Oracle® Enterprise Manager

Command Line Interface

12*c* Release 5 (12.1.0.5)

E17786-20

January 2016

ORACLE®

Oracle Enterprise Manager Command Line Interface, 12*c* Release 5 (12.1.0.5)

E17786-20

Contents

3 Using EM CLI

4 Advanced EM CLI Script Examples

5 Verb Reference

Preface

This manual provides a verb reference, which duplicates and enhances the command-line help, for the Enterprise Manager Command Line Interface (EM CLI). This manual also covers concepts, downloading, deploying, and scripting.

Audience

This guide is written for Enterprise Manager administrators who want to perform operations remotely or script them. The reader should already be familiar with Oracle Enterprise Manager.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following books in the Oracle Enterprise Manager documentation set:

- *Oracle Enterprise Manager Cloud Control Basic Installation Guide*
- *Oracle Enterprise Manager Cloud Control Advanced Installation and Configuration Guide*
- *Oracle Enterprise Manager Cloud Control Administrator's Guide*
- *Oracle Enterprise Manager Cloud Control Upgrade Guide*
- *Oracle Enterprise Manager Cloud Control Extensibility Programmer's Reference*
- *Oracle Database 2 Day DBA*

The latest versions of this and other Oracle Enterprise Manager documentation can be found at:

https://docs.oracle.com/en/enterprise-manager/

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's Changed in this Guide?

The revisions listed below identify information updates, structural changes, as well as relocation of information to other guides.

Since the last revision, the following changes have been made:

- **Added**: EM CLI verbs
- **Moved**: "EM Installation" to section 2.1
- **Moved**: Descriptions of scripting and interactive modes to section 1.2
- **Moved**: "Advanced Script Examples" to section 3.3

1

EM CLI Overview and Concepts

This chapter provides the following topics:

1.1 Overview

The Enterprise Manager Command Line Interface (EM CLI) enables users to access Enterprise Manager functionality through a command-line interface or scripts. It is accessible through classic programming language constructs, enabling tasks to be created and run either from the command-line or programatically. EM CLI enables you to access Enterprise Manager Cloud Control functionality from text-based consoles (shells and command-line windows) for a variety of operating systems.

EM CLI is fully integrated with Enterprise Manager's security and user administration functions, enabling you to carry out operations using EM CLI with the same security and confidentiality as the Enterprise Manager Cloud Control console. For example, you can only see and operate on targets for which you are authorized.

Examples of EM CLI tasks you can accomplish are as follows:

- Create a new Enterprise Manager administrator account.
- Monitor and manage targets, jobs, groups, and blackouts.
- Enable batch/complex tasks on multiple Agents or targets.
- Integrate Enterprise Manager with third-party or custom software through scripts. Actions that are part of a customer's business model can be performed through scripts.

1.2 EM CLI Modes of Operation

EM CLI offers the following modes of operation:

- Standard mode

  In Standard mode, each EM CLI verb entered is a single operating system command. Each command launches EM CLI, executes the command, then terminates.

- Interactive mode

Interactive mode is ideal for adhoc queries or commands for real-time diagnostics or debugging. In this mode, EM CLI is started as a shell and all commands entered on the command line are executed immediately to enable several commands to be executed at will in the same shell.

- Script mode

  In Script mode, an administrator can create a single Python script that includes a sequence of EM CLI commands to be executed with a single invocation.

Each mode uses the same verbs. A verb is a task or action in the form of a user command which exposes Enterprise Manager functionality. Some verbs can include one or more parameters, which are arguments to the specified command. Some of the parameters are required and some are optional.

In the following examples of the `create_group` verb syntax, only the `-name` parameter is required. The other parameters are optional.

- Standard mode example

```
$ emcli create_group -name="name"
      [-type=<group>]
      [-add_targets="name1:type1;name2:type2;..."]
      [-is_propagating="true/false";
```

- Interactive mode example

```
$ emcli
emcli> … other commands…
emcli> create_group(name="name"
       [;type=<group>]
       [;add_targets="name1:type1;name2:type2;..."]
       [;is_propagating="true/false");
emcli> … other commands …
emcli> exit()
$
```

- Script mode example

```
$ emcli @create_group.py;
```

1.2.1 Standard Command-line Mode

This is the traditional and exclusive mode prior to Enterprise Manager Cloud Control version 12.1.0.3. This mode provides a simple command-line interface to Enterprise Manager, and supports the execution of one verb at a time from the command line.

For example:

```
emcli create_group -name=my_group -add_targets="mymachine.myco.com:host"
```

1.2.2 Interactive mode

This mode enables you to create a single interactive session with the server (Oracle Management Services), where you can type in commands, view the output, and potentially respond to or manipulate the output. Interactive mode opens a Jython shell, where you can provide Jython scripts using EM CLI verbs as Jython functions. Jython is a Java implementation of the Python programming language.

Note that when calling a verb in Interactive mode, the arguments are placed inside parentheses. For example:

```
emcli> create_group(name='my_group'..)
```

1.2.3 Script Mode

Script mode is especially effective when performing tasks in bulk mode or many tasks at once. Scripts are useful for accomplishing several tasks, including:

- Listing or setting global target properties
- Listing or setting Agent properties
- Updating database passwords
- Listing group members

This mode enables you to create Jython scripts, store them as files, and then pass these files to EM CLI as an argument, such as ...

```
emcli @createuser.py
```

... where createuser.py is the name of a file containing the Python code to be sent to EM CLI.

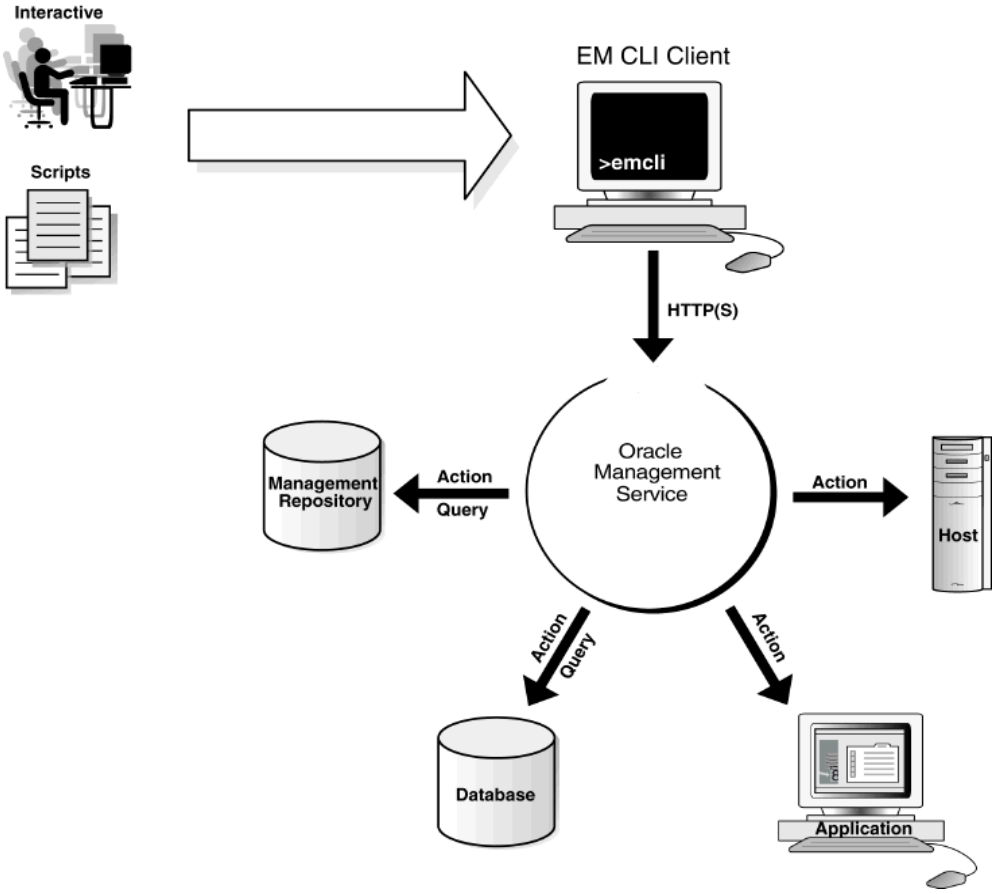
You can create reusable, functional modules using existing EM CLI verbs to generate complex tasks. This intuitive, object-oriented programming model supports encapsulation, loops, functions, exception and error handling, and so forth. These abilities enable you to benefit from all of the powerful features that the Jython programming language offers.

> **See Also:** For more information about using the different modes, see Chapter 3, "Using EM CLI".

1.3 EM CLI Architecture

Figure 1–1 shows the high-level architecture of EM CLI.

Figure 1–1 EM CLI Architecture



EM CLI implements client-server architecture, in which EM CLI is the client, and Oracle Management Services (OMS) is the server.

A typical verb may take zero or more arguments as input. The EM CLI client passes the input to OMS for processing. The EM CLI client connects to OMS and establishes a user session, which is used across verb executions until a logout is initiated.

2

Downloading and Deploying EM CLI

This chapter discusses the following Enterprise Manager Command Line Interface (EM CLI) topics:

2.1 EM CLI Installation

EM CLI provides two installable kits:

- **EM CLI Standard**

  This kit supports the Standard mode only.

- **EM CLI with Scripting mode**

  This kit supports all three modes, but only Interactive and Scripting modes enable you to provide Jython-based scripts.

EM CLI consists of two components used to access the Enterprise Manager framework functionality:

- EM CLI client

  The EM CLI client is a command-line program (Sun Java JRE-based) that sends EM CLI verbs to a specific Oracle Management Service (OMS). In some respects, the EM CLI client functions as a command-line equivalent of an Enterprise Manager Cloud Control console. You can download the EM CLI client on any system within your managed network.

- EM CLI Oracle Management Services (OMS)

  The EM CLI OMS is automatically installed with the OMS and serves as the communication conduit between the EM CLI client and the OMS.

You can download the EM CLI client on any system within your managed network. The EM CLI client is a command-line program (Sun Java JRE-based) that sends EM CLI verbs to a specific Oracle Management Service (OMS). In some respects, the EM CLI client functions as a command-line equivalent of an Enterprise Manager Cloud Control console. The EM CLI OMS is automatically installed with the OMS and serves as the communication conduit between the EM CLI client and the OMS.

For instructions about setting up and running EM CLI, see Chapter 2, "Downloading and Deploying EM CLI".

Note: EM CLI does not support JRockit JVM.

2.2 Downloading and Deploying the EM CLI Client

The EM CLI OMS is automatically installed with the OMS, but you must download and set up the EM CLI client portion. The following instructions cover download procedures for the EM CLI client. The EM CLI client kits are available for public access, so do not require authentication.

As mentioned in Chapter 1, the EM CLI client features two kits: EM CLI Standard and EM CLI with the Script option. The EM CLI Script option includes the Jython Interpreter for Jython script support (described in Chapter 3), as well as all of the features present in the EM CLI Standard kit.

The following sections explain how to download and deploy these two kits.

2.2.1 Requirements

Before downloading the EM CLI client, ensure that the following system requirements have been met:

- Enterprise Manager 12c Cloud Control framework
- Sun Java JRE version 1.6.0_43 or greater
- Workstation running Solaris, Linux, HPUX, Tru64, AIX, or Windows with NTFS

Note: EM CLI does not support JRockit JVM.

2.2.2 Downloading and Deploying the EMC CLI Client for Standard EM CLI

To download the EM CLI client for standard EM CLI only:

1. Obtain the standard EM CLI client kit emclikit.jar using one of the following methods:
   - Download this kit from any 12.1.0.3 or later Cloud Control installation at the following location:

     ```
     https://<your_em_host:port>/em/public_lib_download/emcli/kit/emclikit.jar
     ```

     For example:

     ```
     wget --no-check-certificate
     https://<your_em_host:port>/em/public_lib_download/emcli/kit/emclikit.jar
     ```

   - Download this kit from the Cloud Control console:
     - From the Setup menu, select **Command Line Interface**.
     - In the EM CLI Standard section, click the **Download the EM CLI Standard Kit to your workstation** link.
2. Set your `JAVA_HOME` environment variable and ensure that it is part of your `PATH`. You must be running Sun Java JRE 1.6.0_43 or greater. For example:

   Linux platform:

```
setenv JAVA_HOME /usr/local/packages/j2sdk1.6.0_43
```

Windows platform:

```
C:\Users>set JAVA_HOME=C:\Program Files\Java\jdk1.6.0_43
```

3. Install EM CLI Standard kit into any directory using emclikit.jar. The directory in which EM CLI is installed is called "EM CLI Home" (or "EM CLI Client Directory").

   - For Enterprise Manager Cloud Control version 12.1.0.4.0 and later —

     On a Linux platform, enter:

     ```
     $JAVA_HOME/bin/java -jar emclikit.jar -install_dir=<em_cli_home_dir>
     ```

     On a Windows platform, enter:

     ```
     %JAVA_HOME%\bin\java -jar emclikit.jar -install_dir=<em_cli_home_dir>
     ```

   - For Enterprise Manager Cloud Control versions prior to 12.1.0.4.0 —

     On a Linux platform, enter:

     ```
     $JAVA_HOME/bin/java -jar emclikit.jar client
     -install_dir=<em_cli_home_dir>
     ```

     On a Windows platform, enter:

     ```
     %JAVA_HOME%\bin\java -jar emclikit.jar client
     -install_dir=<em_cli_home_dir>
     ```

4. Change directories to the <em_cli_home_dir> directory where EM CLI is installed, then execute emcli help setup for instructions on how to use the setup verb to configure the EM CLI client for a particular OMS.

   For information on configuring for shared directories environments, see Section 2.2.4, "Using EM CLI With Shared Directories".

2.2.3 Downloading and Deploying the EM CLI Client with the Script Option

---

Note: Before proceeding, click here to see a video tutorial on how to download and get started with EM CLI in Interactive Mode, and click here to see a video tutorial on how to download and get started with EM CLI in Script Mode.

Read the readme.txt file shipped with the EM CLI kit with Scripting mode.

The default behavior for EM CLI with Scripting or Interactive mode is to create its own session, which is not persistent and does not store any user session information on disk. If you have set up EM CLI with Scripting or Interactive mode, the values passed to its options such as -autologin or -trustall would be used by the new session. If you have not set up EM CLI or these options were not passed in the setup verb, you would need to set EM CLI client properties before script execution or launching EM CLI Interactive mode.

- If EM CLI is set up with the -autologin option, the script executes as the auto logged-in user. In the script, you can directly call the verb as a function without having to use login() in it.

- If EM CLI is set up without the -autologin option, the login() function has to be used. If the password is not passed as an argument in the script, you are prompted for the password during script execution.
- If EM CLI is set up with the -trustall option, EMCLI_TRUSTALL or EMCLI_CERT_LOC is not required.

To download the EM CLI client for standard EM CLI as well as Interactive and Script EM CLI:

1. Obtain the EM CLI client kit emcliadvancedkit.jar using one of the following methods:
   - Download this kit from any 12.1.0.3 or later Cloud Control installation at the following location:

     ```
     https://<your_em_host:port>/em/
     public_lib_download/emcli/kit/emcliadvancedkit.jar
     ```

     For example:

     ```
     wget --no-check-certificate
     https://<your_em_host:port>/em/
     public_lib_download/emcli/kit/emcliadvancedkit.jar
     ```

   - Download this kit from the Cloud Control console:
     - From the Setup menu, select **Command Line Interface**.
     - In the EM CLI with Script Option section, click the **Download the EM CLI with Script option kit to your workstation** link.
2. Set your `JAVA_HOME` environment variable and ensure that it is part of your `PATH`. You must be running Sun Java JRE 1.6.0_43 or greater. For example:

   Linux platform:

   ```
   setenv JAVA_HOME /usr/local/packages/j2sdk1.6.0_43
   ```

   Windows platform:

   ```
   C:\Users>set JAVA_HOME=C:\Program Files\Java\jdk1.6.0_43
   ```

3. Install EM CLI with Scripting mode into any directory using emcliadvancedkit.jar. The directory in which EM CLI is installed is called "EM CLI Home" (or "EM CLI Client Directory").
   - For Enterprise Manager Cloud Control version 12.1.0.4.0 and later —

     On a Linux platform, enter:

     ```
     $JAVA_HOME/bin/java -jar emcliadvancedkit.jar
     -install_dir=<em_cli_home_dir>
     ```

     On a Windows platform, enter:

     ```
     %JAVA_HOME%\bin\java -jar emcliadvancedkit.jar
     -install_dir=<em_cli_home_dir>
     ```

   - For Enterprise Manager Cloud Control versions prior to 12.1.0.4.0 —

     On a Linux platform, enter:

     ```
     $JAVA_HOME/bin/java -jar emcliadvancedkit.jar client
     -install_dir=<em_cli_home_dir>
     ```

On a Windows platform, enter:

```
%JAVA_HOME%\bin\java -jar emcliadvancedkit.jar client
-install_dir=<em_cli_home_dir>
```

4. Change directories to the <em_cli_home_dir> directory where EM CLI is installed, then execute emcli help sync for instructions on how to use the sync verb to configure the EM CLI client for a particular OMS.

Note: By default, EM CLI with Scripting mode does not store any user session information on disk. It is tailored to build production-grade Jython modules for Enterprise Manager.

Read the readme.txt file shipped with the EM CLI kit with Scripting mode.

2.2.4 Using EM CLI With Shared Directories

To avoid contention issues when different Enterprise Manager users are accessing the same EM CLI directories, the following configuration is suggested:

1. Set the EMCLI_OPTS environment variable as shown in the following example, using the export Linux operating system command:

```
export EMCLI_OPTS="-Duser.home=/home/user/cli -Demcli.state.dir=/home/user/cli"
```

Since this is an environment variable, you can set this permanently in your session, depending on your operating system.

2. Invoke the setup command, noting the following recommendations:

- Use a different EM CLI state directory per user by defining the directory location with the -dir option.
- Use a different verb jars directory per user by defining the directory location with the -verb_jars_dir option.

For example:

```
$EMCLI_INSTALL_HOME/emcli setup
       -url=https://omsmachine.example.com:em_port/em
       -username="admin"
       -dir="/home/user/cli"
       -verb_jars_dir="/home/user/cli"
```

2.3 Getting Started with EM CLI

After the EM CLI client is downloaded and installed, you are ready to begin using EM CLI. At this point, you can run the EM CLI client out of the installation directory location, or alternatively, you can add it to your `PATH`.

2.3.1 Using Basic Operational Verbs

Immediately after installation, only basic operational verbs are available:

- **argfile** — Execute an EM CLI verb where the verb and any options are contained in a file.
- **help** — Access command-line help for EM CLI verbs.
- **login** — Log in and establish a session with the OMS.

- **logout** — Log out of EM CLI client from Enterprise Manager.
- **setup** — Configure EM CLI to function with a specific OMS.

  (See Section 2.3.2, "Connecting the EM CLI Client to OMS" for important information about this verb.
- **status** — Show EM CLI setup details
- **sync** — Synchronize the EM CLI client with an OMS.
- **version** — List EM CLI verb versions or the EM CLI client version.

EM CLI incorporates a comprehensive command-line help system that provides various levels of assistance. Available from any EM CLI client installation, the help system provides a listing of all available verbs, descriptive overviews for each verb, syntax, as well as usage examples. The command-line help is the definitive EM CLI information source.

2.3.1.1 Using Commands in Standard Mode

To invoke a verb in standard mode, precede the verb with the emcli command. For example, to invoke the help for an overview of all available verbs, enter one of the following commands:

Linux platform:

```
./emcli help
```

Windows platform:

```
>.\emcli help
```

Alternatively, enter the same command followed by the verb name to view a detailed verb description, the verb parameters and options, and usage examples, as in:

Linux platform:

```
./emcli help login
```

Windows platform:

```
>.\emcli help login
```

> **Note:** You can execute EM CLI without using **./** for Linux or **.** for Windows if you set the PATH environment variable to the directory where EM CLI is installed.

2.3.1.2 Using Commands in Interactive Mode

To access command-line verbs in interactive mode, you must first invoke the EM CLI command prompt:

```
$>./emcli
```

To invoke a verb, call the verb name followed by parentheses. For example, enter the following command to invoke the help:

```
emcli> help()
```

To find help for a specific verb, call the help command with the verb within the parentheses surrounded by a single quote:

```
emcli>help('login')
```

Note: The setup and sync commands are not available inside Script and Interactive modes.

Tip: Read the readme.txt file shipped with the advanced kit for more specific examples on how to call the verbs in the EM CLI Client in Script and Interactive modes.

2.3.1.3 Calling a Script

To call a script, you must first invoke the EM CLI command prompt:

```
$>./emcli
```

To run a script, enter `emcli` and provide the script location as shown in the following example, where `my_script.py` is the full path of a valid Python script:

```
%emcli @my_script.py
```

Note: The setup and sync commands are not available inside Script and Interactive modes.

Tip: Read the readme.txt file shipped with the advanced kit for more specific examples on how to call the verbs in the EM CLI Client in Script and Interactive modes.

2.3.2 Connecting the EM CLI Client to OMS

You must run the setup verb to connect the EM CLI client to the OMS running the EM CLI Management Services. Running `setup` installs all available verb-associated command-line help from the EM CLI Management Service. If you have installed EMCLI with the Script option, you can use the sync command instead of the setup command.

Note: If you have followed the instructions in Section 2.2, the set up is already done for you.

You can use one EM CLI client installation to function with multiple OMSes. However, at any time, EM CLI can function with a particular OMS. For either scenario, you need to set up the EM CLI client once for each OMS. You also need to subsequently set the EMCLI_STATE_DIR environment variable to the directory that was specified as the EM CLI client directory for the particular OMS.

To connect the EM CLI client to OMS:

1. Understand the syntax of the setup and sync verbs and their options by entering the following commands or referring to the respective verbs in Chapter 5, "Verb Reference":
   - Command-line EM CLI:

     ```
     ./emcli help setup
     ```

   - Script and Interactive EM CLI:

     ```
     ./emcli help sync
     ```

2. Enter the setup verb with at least the minimally required parameters as shown in This examples:

   - Command-line EM CLI:

     ```
     ./emcli setup -url=http://omsmachine.example.com:em_port/em
       -username=em_user
     ```

   - Script and Interactive EM CLI:

     ```
     ./emcli sync -url=http://omsmachine.example.com:em_port/em
       -username=em_user -trustall
     ```

     If you have already downloaded certificates, you can specify them using the environment variable EMCLI_CERT_LOC. In this case, the -trustall option is not needed.

     Note: Specify the URL you are using to log in to Enterprise Manager through the browser.

   As you observed from step 1, the setup verb has several options, including the following important options:

   - -autologin
   - -noautologin

   In autologin mode, if a session times out, EM CLI automatically logs you in. In the default noautologin mode, if no EM CLI command executes within the 45-minute default session time-out period, you need to log in using the login verb to be able to execute the verbs.

3. Enter your user password for Enterprise Manager when prompted after the EM CLI client connects with the EM CLI Management Services.

After running the `setup` verb, the message "Emcli Setup Successful" appears, and you are ready to begin using EM CLI.

Tip: For complete information on the setup verb and its options, including autogin and noautologin referenced in step 2, see the setup verb.

To configure the EM CLI client to function with multiple Oracle Management Services by implementing multiple setups, see the Examples section for the setup verb.

2.3.3 Configuring an HTTP Proxy Environment

If you are planning to use EM CLI through an HTTP proxy server, you need to set an additional environment variable, `EMCLI_OPTS`, that supplies EM CLI with the requisite proxy host and port information. This examples illustrate setting the `EMCLI_OPTS` environment variable for both Windows and UNIX operating systems.

Example 2–1 Setting EMCLI_OPTS in a Microsoft Windows Environment

```
>set EMCLI_OPTS=-Dhttp.proxyHost=<proxy host> -Dhttp.proxyPort=<proxy port>
```

Example 2–2 Setting EMCLI_OPTS in a UNIX Environment (TCSH)

```
>setenv EMCLI_OPTS "-Dhttp.proxyHost=<proxy host> -Dhttp.proxyPort=<proxy port>"
```

2.3.4 Configuring Log File Settings for EM CLI

EM CLI creates log files to record informational and error messages generated during operation. Not all of the logs in This examples are necessarily present. Logs are created as needed and are appended — they are preserved between invocations of EM CLI. You can safely delete log files any time without affecting the EM CLI operation. The logs help you to troubleshoot any run-time errors.

Note: By default, .emcli.log is only created when an exception or error occurs, or when debugging is enabled. Otherwise, the file does not exist.

This examples show possible log file locations:

```
<EM_CLI_Instance_Home>/.emcli.log
<EM_CLI_Instance_Home>/.emcli.log.1
```

`<EM_CLI_Instance_Home>` refers to the directory specified by the `-dir` option in the latest running of the `setup` verb (with an appended `.emcli` sub-directory). The current `<EM_CLI_Instance_Home>` directory can be identified by executing the `status` verb to display the setup summary.

Log files are limited to a maximum of 0.5 MB. EM CLI alternates between the two log files — as each file reaches the 0.5 MB limit, EM CLI begins writing to the other file, overwriting the oldest log file after `emcli.log.1` has been filled for the first time.

2.3.4.1 Log File Locations

This example show possible log file locations:

Example 2–3 No Configuration Directory Specified with Setup Verb (default location)

```
user.home/.emcli/.emcli.log
user.home/.emcli/.emcli.log.1
```

If you do not specify a configuration directory when you run the `setup` verb (`-dir` option is omitted), EM CLI assumes the `.emcli` configuration directory is located within your local home directory. The log files are placed at the root level of the `.emcli` directory. The `.emcli` directory must be local (not mounted remotely).

Example 2–4 Local Configuration Directory Specified with Setup Verb (-dir=<local directory>

```
local.dir/.emcli/.emcli.log
local.dir/.emcli/.emcli.log.1
```

In this example, the configuration directory is specified using the `-dir` option when the `setup` verb is run. This allows you to specify a local configuration directory if the user home directory is mounted remotely (through NFS, for example).

2.3.4.2 Log File Location and Log Level

You can specify the log file directory and the log level, if desired, using the following variables, which you can set as environment variables:

- EMCLI_LOG_LOC — Sets the log file directory to any desired location.
- EMCLI_LOG_LEVEL — Presets the log level. Allowed values in descending order are:
  - SEVERE (highest level)
  - WARNING
  - INFO
  - CONFIG
  - FINE
  - FINER
  - FINEST (lowest level)

  Additionally, you can use the level OFF to turn off logging, and the level ALL to enable logging of all messages.

2.4 Security and Authentication

To enable EM CLI to function with a particular OMS, configure EM CLI by executing the setup verb. This is a one-time operation for this particular OMS.

Example 2–5 CLI-Enterprise Manager Authentication

```
>emcli setup –url="http[s]://host:port/em" –username="<username>"  [-trustall]
[-novalidate]

>please enter password:
```

You can find out the OMS connection information from any EM CLI client by invoking the `setup` verb without any options. For example:

```
$ emcli setup
Oracle Enterprise Manager Cloud Control 12c Release 4.
Copyright (c) 1996, 2014 Oracle Corporation and/or its affiliates. All rights
reserved.

Instance Home          : /private/emcli/setup/.emcli
Verb Jars Home         : /private/emcli/setup/.emcli
EM URL                 : https://myomshost.us.example.com:5416/em
EM user                : user1
Trust all certificates : true
Auto login             : false
```

You can also invoke the status command, which provides more information than the setup command:

```
$ emcli status
Oracle Enterprise Manager Cloud Control 12c Release 4.
Copyright (c) 1996, 2014 Oracle Corporation and/or its affiliates. All rights
reserved.

Instance Home          : /private/emcli/setup/.emcli
Verb Jars Home         : /private/emcli/setup/.emcli
Status                 : Configured
EMCLI Home             : /private/MWHome/oms/bin
EMCLI Version          : 12.1.0.4.0
```

```
Sun Java JRE Home      : /private/MWHome/jdk16/jdk
Sun Java JRE Version   : 1.6.0_43
Log file               : /private/emcli/setup/.emcli/.emcli.log
EM URL                 : https://myomshost.us.example.com:5416/em
EM user                : sysman
Auto login             : false
Trust all certificates : true
```

2.4.1 HTTPS Trusted Certificate Management

For authenticating an OMS during the SSL server authentication phase of an HTTPS connection handshake, EM CLI searches for trusted certificates in the following key stores:

```
CONFIG_DIR/.emcli/.localkeystore
user.home/.emcli/.keystore
JRE_HOME/lib/security/cacerts
```

`CONFIG_DIR` is the directory specified by the `-dir` option in the latest running of the `setup` verb (with an appended `.emcli` sub-directory).

`JRE_HOME` in a Sun Java JRE installation is typically `JAVA_HOME/jre`.

The Sun Java JRE `keytool` command can manage the key stores. For more information about this tool, see the security documentation for your Sun Java JRE VM installation, or at the time of this writing:

```
http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/keytool.html
```

Not all of the key stores in the list above will necessarily be present.

2.4.2 Secure EM CLI Clients

You can provide credentials to EM CLI in one of two ways:

- Provide credentials at the time of use. See the login and logout verbs for information on credentials.
- Make credentials persistent on the host system where the EM CLI client is running, as might be the case when executing EM CLI verbs from a shell script.

> **Caution:** You should only persist credentials on hosts when the host is a secure EM CLI client, since the only protection available for credentials is the file-system security of the OS.
>
> Oracle also recommends not using persistent credentials if the EM CLI user's home directory is mounted over NFS or any other insecure file system.

2.4.3 Secure Mode for the EM CLI Setup

The EM CLI client installs certain configuration files and a client-side implementation of verbs on the EM CLI client system. The EM CLI client configuration files contain information such as the OMS URL, Enterprise Manager user names, and Enterprise Manager passwords.

By default, the EM CLI client is set up in secure mode. In this mode, EM CLI does not store any Enterprise Manager or SSO passwords on the EM CLI client disk. The command emcli setup -noautologin sets up the EM CLI client in secure mode. By default, -noautologin is true. Therefore, you do not need to specify it if you want to set

up the EM CLI client in secure mode. In secure mode, if the EM CLI session times out due to inactivity, explicit login (using the login verb) is required before invoking any verb.

If you want to set up EM CLI in the insecure auto-login mode, you can use the emcli setup -autologin command. In this mode, if an EM CLI session times out due to inactivity, EM CLI automatically re-establishes the session when a verb needs to execute. However, if you explicitly logged out by running emcli logout, you need to explicitly log in again using emcli login.

- For information on the -noautologin option, see the setup verb.
- For information on logging in, see the login verb.
- For information on logging out, see the logout verb.

2.5 Format Option Availability for Output Data Verbs

Note: The following information regarding the -script option is not to be confused with the Script mode.

For easy parsing of verb output by scripts, a `-script` option is available for all verbs that generate output data. If you use the `-script` option, all output columns become tab-separated (with non-null values), and all rows become newline-separated. You can override the default column and row separators by using the `-format` option in place of `-script`.

```
[-script|-format="name:<format type>;column_separator:<separator_
text>;row_separator:<separator_text>"]
```

Supported `-format` options are shown in Table 2–1.

Table 2–1 Supported "-format" Options

Option	Explanation
`-format="name:pretty"`	Pretty-print the output. This is the default when both `-script` and `-format` are not specified.
`-format="name:script"`	Identical to just specifying `-script`. Columns are tab-separated, and rows are newline-separated.
`-format="name:script;column_separator:<column_sep_string>"`	Causes the verb output to be column-separated by `<column_sep_string>`. Rows are separated by the newline character.
`-format="name:script;row_separator:<row_sep_string>"`	Causes the verb output to be row-separated by `<row_sep_string>`. Columns are separated by the tab character.
`-format="name:script;column_separator:<column_sep_string>;row_separator:<row_sep_string>"`	Causes the verb output to be column-separated by `<column_sep_string>` and row- separated by `<row_sep_string>`.
`-format="name:csv"`	Produces a table with the columns separated by commas and the rows by newlines.

- `-script` is equivalent to `-format="name:script;column_separator:\u0009;row_separator:\u000A"`
- The values for column and row separator are specified as one or more character strings. Any of the characters can be represented by the unicode sequence `\uXXXX` (where `X` is a hex value).

  NOTE: The ASCII character set is represented by `\u00XX`, where `XX` can range from 00 to 7F. For example, the tab character is represented by `\u0009` and the newline character is represented by `\u000A`.
- The `pretty` format type has no attributes.
- In `script` mode, any verb output cells that contain the separator strings are substituted with the unicode values for these strings so that the output does not break any scripts required to parse the output.
- `script` is the only format type for which separators can be specified.
- Separators need not be single characters, and can be specified using both regular characters interspersed with unicode sequences as shown in This example:

Example 2–6 Complex Separator

Separator Specification: `xxx\u0009xxx\u0009`

This separator appears as `xxx` followed by a tab, followed by `xxx`, followed by another tab.

3

Using EM CLI

This chapter discusses the following Enterprise Manager Command Line Interface (EM CLI) topics:

- Using Command-line EM CLI
- Using EM CLI in Interactive or Script Mode
- Advanced Script Examples
- Using the Generic 'List' Verb
- Using the Registered Clients Page

3.1 Using Command-line EM CLI

Command-line EM CLI is the traditional and most direct way of invoking an EM CLI verb. The basic syntax from the system prompt is:

```
emcli verb_name -required_parameter1 -required_parameter2 ... -optional_parameter1
-optional_parameter2 ...
```

The syntax for a particular verb applies to its usage whether it is invoked through the command line or programatically. For example, the syntax for the create_group verb is:

```
emcli create_group
      -name="name"
      [-type=<group>]
      [-add_targets="name1:type1;name2:type2;..."]...
      [-is_propagating="true/false"]
```

This indicates that the -name parameter is required, whereas the -type parameter, enclosed within brackets, is optional, as well as the -add_targets and -is_propagating parameters. This example shows how the verb might be used at the command-line prompt:

```
emcli create_group -name=db_group
      -add_targets="emp_rec:oracle_database"
      -add_targets="payroll:oracle_database"
```

Chapter 5, "Verb Reference" provides the format, descriptions of required and optional parameters, and examples for most EM CLI verbs. Those that are not documented can be found in the online help by entering the following command:

```
emcli help verb_name
```

3.2 Using EM CLI in Interactive or Script Mode

As introduced in Chapter 1, EM CLI provides Interactive and Script modes to enhance and extend the basic functionality offered through the standard command-line invocation. Both Interactive and Script mode provide the same functionality. Unless otherwise stated explicitly, all of the information presented in this chapter pertains to both Interactive and Script modes.

The following sections discuss the fundamental principles associated with the EM CLI interactive or scripting modes:

- Jython Interpreter
- Script and Interactive Mode Syntax
- Interactive Mode — Connecting to an Oracle Management Server (OMS)
- Examples of Standard, Interactive, and Script Verb Invocations
- Writing and Running the First Script
- Invoking an EM CLI Verb Programatically
- Error Exception Handling
- Utility Functions
- Selected Use Cases
- Selected list Verb Use Cases

> **Tip:** For a demonstration of using EM CLI in script and interactive mode, click on the links below to view the following Enterprise Manager Screenwatches:
>
> - Getting Started with EM CLI in Script Mode
> - Downloading and Getting Started with EM CLI in Interactive Mode

3.2.1 Jython Interpreter

Beginning with Enterprise Manager Cloud Control version 12cR3, EM CLI includes an embedded Jython interpreter (Jython 2.5.3), where all of the verbs are registered as functions, known as *EM CLI verb functions* or simply *functions*. Usage of these functions is similar to the corresponding verb. In these functions, the parameters (supplied as key-value pairs) are those present in the verb arguments.

In Interactive mode, the interpreter opens a shell where you can type your commands. Using Script mode, you can run your Jython program by passing it to the interpreter in a non-interactive fashion. For both modes, apart from using the EM CLI verb functions, you can also program in Jython conventionally.

3.2.2 Script and Interactive Mode Syntax

The syntax for these two modes varies slightly.

To run a script:

Enter `emcli` and provide the script location as shown in the following example, where `my_script.py` is the full path of a valid Python script:

```
%emcli @my_script.py
```

To start EM CLI in Interactive mode:

Enter emcli at the command prompt to start an interactive shell, as follows:

Linux platform:

```
% emcli
emcli>
```

Windows platform:

```
C:\Directory> emcli
emcli>
```

Comparing Script and Interactive Modes

To illustrate using the interpreter in both Script and Interactive mode to achieve the same objective, This examples print the current version of the installed EM CLI client. Example 3–1 shows a Python script that uses the version() verb of EM CLI to print the current version. Example 3–2 achieves the same result using the interactive shell. Note that the version verb used in both of these examples has the same signature and functionality.

Example 3–1 Script that Prints the Current Version

For a script named emcli_helloworld.py with the following contents:

```
print 'Hello EMCLI'
print version()
```

The output is:

```
Hello EMCLI
Oracle Enterprise Manager 12c EMCLI Version 12.1.0.4.0
```

Example 3–2 Interactive Input that Prints the Current Version

```
$emcli>print 'Hello EMCLI'
Hello EMCLI
$emcli>version()
Oracle Enterprise Manager 12c EMCLI Version 12.1.0.4.0
```

3.2.3 Interactive Mode — Connecting to an Oracle Management Server (OMS)

Because most of the verbs require a connection to an OMS, you need to set up an OMS connection in an Interactive shell before you can invoke any verb by minimally setting the following required client properties and optionally setting others:

- EMCLI_OMS_URL
- EMCLI_TRUSTALL or EMCLI_CERT_LOC

The following procedure provides a recommended method of setting up these properties and subsequently logging in.

1. Type help('client_properties') in the Interactive shell for more information about the available client properties, as shown in the following output example:

```
emcli>help('client_properties')
EMCLI_OMS_URL             : OMS URL To connect to.
EMCLI_USERNAME            : OMS Username.
EMCLI_AUTOLOGIN           : Possible values are true,false. Default is false.
EMCLI_TRUSTALL            : Possible values are true,false. Default is false.
EMCLI_VERBJAR_DIR         : Location of bindings directory.
EMCLI_CERT_LOC            : Location of a valid certificate file.
EMCLI_LOG_LOC             : Directory where log files will be stored.
```

```
EMCLI_LOG_LEVEL           : Possible values are ALL,INFO,FINE,FINER,WARN,SEVERE
                                  Default is SEVERE.
EMCLI_OUTPUT_TYPE       : Possible values are json, JSON, text, TEXT. Default
                          is json in script mode and text in interactive mode.

status() will list values of all the client properties. set_client_
property(propertyname,value), get_client_property(propertyname), and
clear_client_property(name)can be used to set, get, and clear a client property
```

2. Set the required client properties from either the Interactive shell directly using the set_client_property() function, or as environment variables before a shell is launched.

   - **Function method**

     For example, to connect to an OMS at https://host1.example.com:1234/em in which you want to trust all certificates:

     ```
     emcli>set_client_property('EMCLI_OMS_URL',
           'https://host1.example.com:1234/em')
     emcli>set_client_property('EMCLI_TRUSTALL','true')
     ```

   - **Environment variables method**

     For example, to set the same client properties as environment variables on a Linux platform:

     ```
     % setenv EMCLI_TRUSTALL true
     % setenv EMCLI_OMS_URL https://host1.example.com:1234/em
     ```

     Windows platform:

     ```
     C:\Directory> set EMCLI_TRUSTALL=true
     C:\Directory> set EMCLI_OMS_URL=https://host1.example.com:1234/em
     ```

3. Log into the OMS:

   ```
   emcli>login(username='<user>')
   ```

4. Provide a password at the prompt. You can also alternatively provide a password as shown:

   ```
   emcli>login(username='foo', password='bar')
   ```

Alternatively Logging in with EMCLI_USERNAME

You can use the EMCLI_USERNAME client property to log in as shown in This example for the Linux platform:

```
% setenv EMCLI_USERNAME sysman
    emcli>login()
Enter password :  ******

Login successful
```

Windows platform:

```
C:\Directory> set EMCLI_USERNAME sysman
    emcli>login()
Enter password :  ******

Login successful
```

Displaying the Status of a Session

You can use the status() command to display the status of an EM CLI session, as shown in This example:

```
emcli>status()
    <banner>

    Verb Jars Home (EMCLI_VERBJAR_DIR)  :
                                    /<Location>/int/./bindings/12.1.0.x.0/.emcli
    EM CLI Home (EMCLI_INSTALL_HOME)        : /<Location>/int/.
    EM CLI Version                          : 12.1.0.x.0
    Java Home                               : /jdk6/jre
    Java Version                            : 1.x.0_x
    Log file (EMCLI_LOG_LOC)                : CONSOLE
    Log level (EMCLI_LOG_LEVEL)             : SEVERE
    EM URL (EMCLI_OMS_URL)                  : https://host1.example.com:1234/em
    EM user (EMCLI_USERNAME)                : <user>
    Auto login (EMCLI_AUTOLOGIN)            : false
    Trust all certificates (EMCLI_TRUSTALL) : true
```

Exiting the Interactive Shell

To exit the EM CLI interactive shell, use the exit verb as shown:

```
emcli>exit()
```

3.2.4 Examples of Standard, Interactive, and Script Verb Invocations

This examples contrast these three methods of verb invocations.

Example 1 — String-based Arguments

Standard invocation:

```
% emcli create_user -name='jan.doe' -type='EXTERNAL_USER'
```

Interactive mode invocation:

```
emcli>create_user(name='jan.doe',type='EXTERNAL_USER')
```

Script mode invocation:

```
create_user(name='jan.doe',type='EXTERNAL_USER')
```

Example 2 — List-based Arguments

Standard invocation:

```
% emcli grant_privs -name='jan.doe' \
        -privilege="USE_ANY_BEACON" \
        -privilege="FULL_TARGET;TARGET_NAME=host1.example.com:TARGET_TYPE=host"
```

Interactive mode invocation:

```
emcli>priv_list = ['USE_ANY_BEACON',
'FULL_TARGET;TARGET_NAME=myhost.us.example.com:TARGET_TYPE=host']
```

Script invocation:

```
priv_list=['USE_ANY_BEACON',
'FULL_TARGET;TARGET_NAME=myhost.us.example.com:TARGET_TYPE=host']
```

Example 3 — Flag-based Boolean Arguments

Standard invocation:

```
% emcli get_targets -noheader
```

Interactive mode invocation:

```
emcli>get_targets(noheader=True)
```

Script invocation:

```
get_targets(noheader=True)
```

Advisory Information About Incomplete Commands in Interactive Mode

If you do not complete a command in interactive mode, the Jython interpreter prompts with three dots on the next line. Until the line is complete, Jython continues to generate this prompt. For example:

```
emcli > status(
...
...
...
...)
```

Providing the closing parenthesis executes the status command.

In This example, "\n" completes the line, and reports a syntax error.

```
emcli> get_targets -targets="oracle_database"
...
...
...\n
```

3.2.5 Writing and Running the First Script

To assist you in writing your first script, this section analyzes a sample script that retrieves all targets and prints their names. Example 3–3 shows the entire script.

> **Note:** Line numbers are provided only for explanatory purposes for Table 3–1. For a copy-ready script, see Example A–1 in Appendix A, "Sample Scripts".

Example 3–3 Script That Retrieves All Targets and Prints Their Names

```
 1 #emcli_get_targets.py
 2
 3 #Import all emcli verbs to current program
 4 from emcli import *
 5
 6 def print_target_details(target):
 7    '''
 8    print the target name and target type given a target tuple.
 9    '''
10    print target['Target Name'] + ' ' + target['Target Type']
11
12 #Set the OMS URL to connect to
13 set_client_property('EMCLI_OMS_URL','https://host1.example.com:1234/em')
14 #Accept all the certificates
15 set_client_property('EMCLI_TRUSTALL','true')
16
```

```
17 #Login to the OMS
18 login(username='adminuser')
19
20 #Invoke get_targets and loop over the targets array
21 targets_array = get_targets().out()['data']
22 for target in targets_array:
23       #Call print_target_details function to print the target details
24     print_target_details(target)
```

Note: Observe the method of accessing the JSON response from the verb response get_targets().out()[‘data’]. The get_targets() response provides a handle to the response object, and out()[‘data’] provides a handle over the underlying JSON data. This methodology is consistent for most verbs.

Script Analysis

Table 3–1 provides an analysis of each line of code.

Table 3–1 Line-by-Line Script Analysis

Lines	Description

4	Jython import construct to import all EM CLI verb functions in the current program. You can also selectively import the verb functions. You can use the Jython import function to import all of the functions as wildcards or on an as needed basis explicitly. For example:
	`from emcli import *`
	... imports all of the functions, whereas ...
	`from emcli import get_targets`
	... imports only the get_targets function.

6 - 10	Custom Jython function to print the name and type of a target. It accepts a key value tuple of the form. It accepts a key value tuple of the form {Target Name, Target Type} as the parameters.
13, 15	Necessary connection to OMS in order to retrieve all targets. Before connecting to the OMS, you must first set the OMS connection details using the set_client_property() function. This sets the OMS URL to https://host1.example.com:1234/em and enables the client to trust all certificates. Note that none of these details are stored in disk. These details are stored in memory and only last for a single script execution. For more information on client properties, enter help(‘client_properties’) from the interactive shell. You can define EMCLI_OMS_URL and EMCLI_TRUSTALL variables as environment variables if you do not want to set these in your script. If you have downloaded certificates somewhere, you can also use the environment variable EMCLI_CERT_LOC to point to the certificate directory. In this case, you do not need EMCLI_TRUSTALL.
18	Login function to connect to the OMS. The example uses the Sysman user to log in. This prompts for a password during execution.
21 - 24	Invokes the get_targets() function and captures its response in an array called targets_array. This is in JSON format. This example iterates through this array and uses the custom function print_target_details to print its name and type.

Script Execution

Example 3–4 shows that executing this script retrieves the list of all targets and their types.

Example 3–4 Output of Script that Retrieves All Targets

```
$emcli @emcli_get_targets.py
Enter password :  ******
test.example.com host
EM Management Beacon oracle_beacon
CSAcollector oracle_csa_collector
Oemrep_Database oracle_database
EM Jobs Service oracle_em_service
test.example.com:1838 oracle_emd
Management Services and Repository oracle_emrep
Management_Servers oracle_emsvrs_sys
test.example.com:7654_Management_Service oracle_oms
test.example.com:7654_Management_Service_CONSOLE oracle_oms_console
test.example.com:7654_Management_Service_PBS oracle_oms_pbs
/EMGC_EMGC_DOMAIN/EMGC_DOMAIN weblogic_domain
Logout successful
```

The Logout Successful message indicates that the login session to the OMS is closed at the end of the execution.

3.2.6 Invoking an EM CLI Verb Programatically

As mentioned earlier, all of the verbs are available as global Jython functions with verb options as function parameters. Flag-based options are provided by specifying True as the value. List-based options are provided by constructing a Python list and using it as an argument. Table 3–7 provides more details on this.

3.2.6.1 Accessing Verb Invocation Responses

Every EM CLI verb invocation returns a Response object. The Response object is part of EM CLI, and has the functions listed in Table 3–2.

Table 3–2 Response Object Functions

Function	Description
out()	Provides the verb execution output. The output can be text, or the JSON.isJson() method on the Response object can be used to determine whether the output is JSON. Refer to the section "JSON Processing" for more details.
error()	Provides the error text (if any) of the verb execution if there are any errors or exceptions during verb execution. Refer to the section "Error and Exception Handling" for more details.
exit_code()	Provides the exit code of the verb execution. The exit code is zero for a successful execution and non-zero otherwise. Refer to the section "Error and Exception Handling" for more details.
isJson()	Provides details about the type of output. It returns True if response.out() can be parsed into a JSON object.

Example 3–5 invokes the get_targets verb and prints the output, error, and exit code of the execution.

Note: Line numbers are provided only for illustrative purposes. For a copy-ready script, see Example A–2 in Appendix A, "Sample Scripts".

Example 3–5 Script that Incorporates Functions in the get_targets Verb

```
1  #emcli_introspect_response.py
2
3  #Import all emcli verbs to current program
4  from emcli import *
5
6  #Set the OMS URL to connect to
7  set_client_property('EMCLI_OMS_URL','https://host1.example.com:1234/em')
8  #Accept all the certificates
9  set_client_property('EMCLI_TRUSTALL','true')
10
11 #Login to the OMS
12 login(username='sysman')
13
14 res = get_targets()
15
16 print 'Number of targets:'+str(len(res.out()['data']))
17 print 'Errors           :'+res.error()
18 print 'Exit code        :'+str(res.exit_code())
19 print 'IsJson           :'+str(res.isJson())
```

Line 16 shows that instead of printing the raw response (which will be JSON), the example uses the Jython len() function to print the length of the response array, which is basically the count of all of the targets. Note that the example uses the Jython str() function to convert an integer type to a string.

Example 3–6 shows the execution of the script in Example 3–5.

Example 3–6 Output of Script that Invokes the get_targets Verb

```
$emcli @emcli_introspect_response.py
Enter password :  ******
Number of targets:12
Errors           :
Exit code        :0
IsJson           :True
Logout successful
```

3.2.6.2 JSON Processing

If a verb response is JSON, it can be interactively iterated and accessed. You can use response.isJson() to check whether the verb output is JSON. If the verb output is JSON, response.out()['data'] provides the object in the Jython object model.

JSON processing has been shown in previous examples. Example 3–7 shows another example of this processing. The example uses custom SQL with the list() function, which provides a generic method to retrieve data about managed objects in Enterprise Manager. Custom SQL only works if the OMS user has super user privileges.

Note: Line numbers are provided only for explanatory purposes for Table 3–3. For a copy-ready script, see Example A–3 in Appendix A, "Sample Scripts".

Example 3–7 Script that Incorporates Custom SQL with the list() Function

```
1  #emcli_json_processing.py
2  #Import all EM CLI verbs to current program
3  from emcli import *
4  def format(str):
5     '''
6     Given a string argument returns it back or returns
7     a blank string if it is of None type
8     '''
9     if str is None:
10      return ""
11    return str
12
13 def get_targets_with_props(p_prop_name, p_prop_val):
14 '''
15 Returns targets with given property name and its value. Uses list verb.
16 '''
17 l_sql = "select target_name, target_type, property_value " \
18         "from mgmt$target_properties " \
19         "where property_name = '" + p_prop_name + "' " + "    " \
20         "and property_value like '" + p_prop_val + "'"
21 obj=list(sql=l_sql)
22    return obj
23 #Set the OMS URL to connect to
24 set_client_property('EMCLI_OMS_URL','https://host1.example.com:1234/em')
25 #Accept all the certificates
26 set_client_property('EMCLI_TRUSTALL','true')
27 #Log in to the OMS
28 login(username='sysman')
29 #Find all the targets that have Version property set to release 12
30 l_targets = get_targets_with_props('Version', '12%')
31 for target in l_targets.out()['data']:
32       tn =  target['TARGET_NAME']
33       tt =  target['TARGET_TYPE']
34       pv =  target['PROPERTY_VALUE']
35       print "Name "+tn + " Type =" + tt + " value=" + pv
```

Script Analysis

Table 3–3 provides an analysis of relevant lines of code. The remainder of the program is similar to Example 3–3, which was analyzed in Table 3–1.

Table 3–3 Line-by-Line Script Analysis

Lines	Description
13 - 22	A custom Jython function get_targets_with_props() returns all of the targets with a given property name and value. It uses the list() function or verb to query the targets. This verb, introduced in 12cR3, provides a convenient way to search the Enterprise Manager repository for resources. One of its features is to list the resources matching a given SQL query, which is used in the example. The output of this verb is JSON, which can be accessed using out()['data'].
31 - 35	Iterates over the JSON response and prints the target name and target type.

Script Execution

Example 3–8 shows that executing this script retrieves the list of all targets and their types.

Example 3–8 Output of Script that Incorporates Custom SQL

```
$emcli @emcli_json_processing.py
Enter password :  ******
Name test.example.com:1838 Type =oracle_emd value=12.1.0.3.0
Logout successful
```

3.2.7 Error Exception Handling

If an exception or error occurs during verb execution, an exception of type emcli.exception.VerbExecutionError is raised. emcli.exception.VerbExecutionError extends from RuntimeError and hence stops the execution. You can use standard Jython exception handling to catch this exception.

emcli.exception.VerbExecutionError has the functions listed in Table 3–4.

Table 3–4 Functions for emcli.exception.VerbExecutionError

Function	Description
error()	Provides the error text of the verb execution.
exit_code()	Provides the exit code of the verb execution.

Example 3–9 shows the usage of VerbExecutionError.

> **Note:** Line numbers are provided only for explanatory purposes for Table 3–5. For a copy-ready script, see Example A–4 in Appendix A, "Sample Scripts".

Example 3–9 Script that Incorporates Exception Handling

```
1  #emcli_error_exception_handling.py
2
3  #import all emcli verbs to current program
4  from emcli import *
5  #import the verbexecutionerror
6  from emcli.exception import VerbExecutionError
7
8  #Set the OMS URL to connect to
9  set_client_property('EMCLI_OMS_URL','https://host1.example.com:1234/em')
10 #Accept all the certificates
11 set_client_property('EMCLI_TRUSTALL','true')
12
13 #Login to the OMS
14 login(username='sysman')
15
16 #Create a group
17 res = create_group(name='Jan_Doe_Group')
18
19 print res.out()
20
21 #Try to create the same group again
22 try:
23   #This will trigger an exception as the group exist already
24   create_group(name='Jan_Doe_Group')
25 except VerbExecutionError , e:
26    print e.error()
27    print 'Exit code:'+str(e.exit_code())
```

Script Analysis

Table 3–5 provides an analysis of relevant lines of code.

Table 3–5 Line-by_Line Script Analysis

Lines	Description
4, 6	Imports all EM CLI verbs, and imports VerbExecutionError.
9, 11	Necessary connection to OMS in order to retrieve all targets. Before connecting to the OMS, you must first set the OMS connection details using the set_client_property() function. This sets the OMS URL to https://host1.example.com:1234/em and enables the client to trust all certificates. Note that none of these details are stored in disk. These details are stored in memory and only last for a single script execution. For more information on client properties, enter help('client_properties') from the interactive shell. You can define EMCLI_OMS_URL and EMCLI_TRUSTALL variables as environment variables if you do not want to set these in your script. If you have downloaded certificates somewhere, you can also use the environment variable EMCLI_CERT_LOC to point to the certificate directory. In this case, you do not need EMCLI_TRUSTALL.
14	Login function to connect to the OMS. The example uses the sysman user to log in. This prompts for a password during execution.
22 - 27	Exception use case to create the same group again. This produces a run-time error, which the example is handling in the try except block.

Script Execution

Example 3–10 shows the output of the script shown in Example 3–9.

Example 3–10 Output of Error Exception Handling Script

```
$emcli @emcli_error_exception_handling.py
Enter password :  ******
Group "Jan_Doe_Group:group" created successfully

Error: Group "Jan_Doe_Group:group" already exists

Exit code:1
Logout successful
```

3.2.8 Utility Functions

The functions shown in Table 3–6 are also available in the EM CLI package.

Table 3–6 Additional Functions

Function	Description
last_out()	Returns the output for the last executed EM CLI command. It returns None if an EM CLI command has not been executed in the current session.
last_error()	Returns the error text (if any) for the last executed EM CLI command. It returns None if an EM CLI command has not been executed in the existing session, or all of the previous executions were successful.
clear()	Clears the current shell in Interactive mode.
exit(ret_val)	Exits from the EM CLI Interactive shell with ret_val.

3.2.9 Extending EM CLI with Python Libraries

You can extend EM CLI with end-user Python libraries by doing one of the following:

- Copy modules to the extension directory, as shown in This example:

  ```
  $EMCLI_INSTALL_HOME/extdir
  ```

- Specify the EMCLI_PYTHONPATH environment variable where the Python modules are loaded from.

3.2.10 Selected Use Cases

Table 3–7 shows various use cases and corresponding solution examples for standard EM CLI versus interactive invocations or scripts.

Table 3–7 Use Case Examples

Task/Action	Usage for Standard EM CLI	Usage for EM CLI Interpreter (interactive shell or script)
Invoke a verb with string-based arguments	`% emcli create_user -name='jane.doe' -type='EXTERNAL_USER'`	`create_user(name='jane.doe', type='EXTERNAL_USER')`

Invoke a verb with list-based arguments	`% emcli grant_privs -name='jan.doe' -privilege="USE_ANY_BEACON" \ privilege="FULL_TARGET;TARGET_NAME=host1.example.com:TARGET_TYPE=host"`	`#First construct a list priv_list = ['USE_ANY_BEACON','FULL_TARGET;TARGET_NAME=host1.example.com:TARGET_TYPE=host']`
		
		`#Now use the list grant_privs(name='jan.doe', privilege=priv_list)`

Invoke a verb with flag-based Boolean arguments	`% emcli get_targets -noheader`	`get_targets(noheader=True)`
Using help	`help $verb_name` For example, help get_targets prints the help for the get_targets verb.	`help('$verb_name')` For example, help('get_targets') prints the help for the get_targets verb.

3.3 Advanced Script Examples

This chapter provides examples of using EM CLI to write scripts and automate routine tasks. To use these scripts, Oracle recommends that you are experienced with scripting languages and familiar with Jython (Python for the Java platform).

- Changing Lifecycle Status Properties
- Changing Your Database Password
- Promoting Discovered Targets

3.3.1 Changing Lifecycle Status Properties

To assist you in writing scripts, this section analyzes a sample script that changes the lifecycle status properties.

Example 3–11, "LifeCyclePropertyChange.py" enables an Enterprise Manager administrator to change the lifecycle status of all the Oracle databases (release 11.2) in their test environment from Test to Production. Without this script, you would have to log in to the Enterprise Manager Cloud Control console, and identify all the release

11.2 databases, then manually change the property to Production for each database target from the target's home page.

You can reuse this script whenever there is a request to change a set of targets to a different Lifecycle status.

> **Note:** Line numbers are provided only for explanatory purposes for Table 3–3. For a copy-ready script, see Example A–5 in Appendix A, "Sample Scripts".

Example 3–11 LifeCyclePropertyChange.py

```
#Disclaimer
#EXCEPT WHERE EXPRESSLY PROVIDED OTHERWISE, THE SITE, AND ALL CONTENT PROVIDED ON
#OR THROUGH THE SITE, ARE PROVIDED ON AN "AS IS" AND "AS AVAILABLE" BASIS. ORACLE
#EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED,
#INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
#FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT WITH RESPECT TO THE SITE AND ALL
#CONTENT PROVIDED ON OR THROUGH THE SITE. ORACLE MAKES NO WARRANTY THAT: (A) THE
#SITE OR CONTENT WILL MEET YOUR REQUIREMENTS; (B) THE SITE WILL BE AVAILABLE ON AN
#UNINTERRUPTED, TIMELY, SECURE,OR ERROR-FREE BASIS; (C) THE RESULTS THAT MAY BE
#OBTAINED FROM THE USE OF THE SITE OR ANY CONTENT PROVIDED ON OR THROUGH THE SITE
#WILL BE ACCURATE OR RELIABLE; OR (D) THE QUALITY OF ANY CONTENT PURCHASED OR
#OBTAINED BY YOU ON OR THROUGH THE SITE WILL MEET YOUR EXPECTATIONS.
#ANY CONTENT ACCESSED, DOWNLOADED OR OTHERWISE OBTAINED ON OR THROUGH THE USE OF
#THE SITE IS USED AT YOUR OWN DISCRETION AND RISK. ORACLE SHALL HAVE NO
#RESPONSIBILITY FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM OR LOSS OF DATA THAT
#RESULTS FROM THE DOWNLOAD OR USE OF CONTENT.
#ORACLE RESERVES THE RIGHT TO MAKE CHANGES OR UPDATES TO, AND MONITOR THE USE OF,
#THE SITE AND CONTENT PROVIDED ON OR THROUGH THE SITE AT ANY TIME WITHOUT NOTICE.
 1 from emcli import *
 2
 3 search_list = ['PROPERTY_NAME=\'DBVersion\'','TARGET_TYPE=
 \'oracle_database\'','PROPERTY_VALUE LIKE \'11.2%\'']
 4
 5 if len(sys.argv) == 2:

 6        print login(username=sys.argv[0])
 7        l_prop_val_to_set = sys.argv[1]
 8        l_targets = list(resource="TargetProperties", search=search_list,
columns="TARGET_NAME,TARGET_TYPE,PROPERTY_NAME")
 9        for target in l_targets.out()['data']:
10             t_pn =  'LifeCycle Status'
11             print "INFO: Setting Property name " + t_pn + " to value " + l_
prop_val_to_set
12             print set_target_property_value(property_records=target['TARGET_
NAME']+":"+target['TARGET_TYPE']+":"+t_pn+":"+l_prop_val_to_set)
13  else: ]
14        print "\n ERROR: Property value argument is missing"
15        print "\n INFO: Format to run this file is filename.py <username>
<Database Target LifeCycle Status Property Value>"
```

3.3.1.1 Script Analysis

Table 3–1 provides an analysis of each line of the code.

Table 3–8 Line-by_Line Script Analysis

Lines	Description
1	Jython import construct to import all EM CLI verb functions in the current program.
3	search_list is a variable to pass to the search option in the list verb. This example uses escape characters to represent single quotes. To pass more than one value for the same option in the list verb, define as comma separated values, surrounded by square brackets.
5	Defines an if condition to ensure the user provides two arguments with the script, otherwise the script prints an error message (defined in lines #15, 16)
6	Provides a login to Enterprise Manager. You can remove this if you have set up EM CLI with autologin. For more information about setup and autologin, see Section 2.2.3, "Downloading and Deploying the EM CLI Client with the Script Option" and the setup and the login verbs.
7	l_prop_val_to_set is a variable. This is the property value to be set. Remember that this script is changing this value from Test to Production. You can change this value to any acceptable Lifecycle property value. For a list of valid values, see the modify_lifecycle_stage_name verb or the *Oracle Enterprise Manager Cloud Control Administrator's Guide*.
8	Stores the output of the list verb in l_targets. In the list verb, this script passes the resource as TargetProperties, and search as the search_list variable. This script specifies three columns: ▪ target_name ▪ target_type ▪ property_name
9	Defines a for loop. The data in l_targets is available in JSON format. This loop iterates through the information target property information returned from the list verb. For information about JSON processing, see Section 3.2.6.2, "JSON Processing"
10	Sets t_pn to the LifeCycle Status value.
11	Provides a progress message to the user stating that the script is setting the LifeCycle Status to the value passed to the script from the command line.
12	Defines the set_target_property_value verb, which sets the value using the property_records option. When this verb is set for a target pair, it moves to the next one. This example shows three databases, but in reality, use this script for a larger number of databases.
13 -15	Else statement combined with the if condition in line #5. If the arguments specified in line #5 are not provided correctly, then the script displays one of the following error messages: ▪ Property value argument is missing ▪ Format to run this file is filename.py <username> <Database Target LifeCycle Status Property Value>

3.3.1.2 Script Output

Running Example 3–11, "LifeCyclePropertyChange.py" provides the following output:

Example 3–12 Output from LifeCyclePropertyChange.py

```
$ emcli @myScript.py user Production

Login successful

INFO: Setting Property name LifeCycle Status to value Production for db1
```

```
Properties updated successfully
INFO: Setting Property name LifeCycle Status to value Production for db2
Properties updated successfully
INFO: Setting Property name LifeCycle Status to value Production for db3
Properties updated successfully

Logout successful
```

3.3.2 Changing Your Database Password

To assist you in writing scripts, this section analyzes a sample script that changes the password of your database.

Example 3–13, "dbPasswordChange.py" is useful if you want to reset the database password on a regular basis for security compliance. If you change the database password in a target database, then Enterprise Manager monitoring for that target database is unavailable. To ensure consistent monitoring, you would have to log in to the Enterprise Manager Cloud Control UI, select the required database target and change the password for each and every database, which is very time-consuming.

By using Example 3–13, "dbPasswordChange.py", you can add a number of databases to a group and then change the password for all these databases within the group.

> **Note:** Line numbers are provided only for explanatory purposes for Table 3–9. For a copy-ready script, see Example A–6 in Appendix A, "Sample Scripts".

Example 3–13 dbPasswordChange.py

```
#Disclaimer
#EXCEPT WHERE EXPRESSLY PROVIDED OTHERWISE, THE SITE, AND ALL CONTENT PROVIDED ON
#OR THROUGH THE SITE, ARE PROVIDED ON AN "AS IS" AND "AS AVAILABLE" BASIS. ORACLE
#EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED,
#INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
#FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT WITH RESPECT TO THE SITE AND ALL
#CONTENT PROVIDED ON OR THROUGH THE SITE. ORACLE MAKES NO WARRANTY THAT: (A) THE
#SITE OR CONTENT WILL MEET YOUR REQUIREMENTS; (B) THE SITE WILL BE AVAILABLE ON AN
#UNINTERRUPTED, TIMELY, SECURE,OR ERROR-FREE BASIS; (C) THE RESULTS THAT MAY BE
#OBTAINED FROM THE USE OF THE SITE OR ANY CONTENT PROVIDED ON OR THROUGH THE SITE
#WILL BE ACCURATE OR RELIABLE; OR (D) THE QUALITY OF ANY CONTENT PURCHASED OR
#OBTAINED BY YOU ON OR THROUGH THE SITE WILL MEET YOUR EXPECTATIONS.
#ANY CONTENT ACCESSED, DOWNLOADED OR OTHERWISE OBTAINED ON OR THROUGH THE USE OF
#THE SITE IS USED AT YOUR OWN DISCRETION AND RISK. ORACLE SHALL HAVE NO
#RESPONSIBILITY FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM OR LOSS OF DATA THAT
#RESULTS FROM THE DOWNLOAD OR USE OF CONTENT.
#ORACLE RESERVES THE RIGHT TO MAKE CHANGES OR UPDATES TO, AND MONITOR THE USE OF,
#THE SITE AND CONTENT PROVIDED ON OR THROUGH THE SITE AT ANY TIME WITHOUT NOTICE.

 1 from emcli  import *
 2 from emcli.exception import VerbExecutionError
 3 import sys
 4 import time
 5
 6 def check_job_status(job):
 7   count=0
 8   while (count < 10):
 9    count = count + 1
10    obj = emcli.get_jobs(job_id=job)
```

```
11    #print obj.out()
12    for entry in obj.out()['data']:
13        l_status = entry['Status ID']
14        l_exec_id = entry['Execution ID']
15        #print entry['Status ID']
16        if (l_status == '5'):
17            print "Job completed successfully"
18            count=100
19        elif (l_status == '4'):
20            l_resp = get_job_execution_detail(execution=l_exec_id,
showOutput=True, xml=True)
21            print "Job failed, error details "
22            print "Output " + str(l_resp.out())
23            count=100
24        else:
25            time.sleep(2)
26
27 def update_db_pwd_for_target(p_target_name, p_target_type, p_old_password, p_
new_password):
28    l_target_name = p_target_name
29    l_target_type = p_target_type
30    print "Changing the password for member : name = " + l_target_name + " type
= " + l_target_type
31    try :
32        l_resp = update_db_password (target_name=l_target_name,
33                                     target_type = l_target_type,
34                                     change_at_target="yes",
35                                     user_name="dbsnmp",
36                                     old_password=p_old_password,
37                                     new_password=p_new_password,
38                                     retype_new_password=p_new_password)
39        l_job_submitted = l_resp.out()['JobId']
40        check_job_status(l_job_submitted)
41    except emcli.exception.VerbExecutionError, e:
42        print "ERROR : Change Password failed for name = " + l_target_name + "
type = " + l_target_type
43        print "ERROR : " + e.error()
44
45 def update_db_pwd_for_group(p_group, p_old_password, p_new_password):
46    print "Changing the password for group - " + p_group + " from " + p_old_
password + " to " + p_new_password
47    members = get_group_members(name=p_group).out()['data']
48    for member in members:
49        l_target_name = member['Target Name']
50        l_target_type = member['Target Type']
51        update_db_pwd_for_target(l_target_name, l_target_type, p_old_password,
p_new_password)
52
53
54 #Set the OMS URL to connect to
56 set_client_property('EMCLI_OMS_URL','https://myoms.com/em')
57 #Accept all the certificates
58 set_client_property('EMCLI_TRUSTALL','true')
59
60 login(username=sys.argv[0])
61
62
63 l_grp_name = 'maurGroup'
64
65 l_group_members = ['db1:oracle_database','db2:oracle_database','db3:rac_
```

```
database']
66
67
68
69 res = create_group(name = l_grp_name, add_targets = l_group_members)
70
71 print "Listing members for group " + l_grp_name
72
73 for member in get_group_members(name=l_grp_name).out()['data']:
74    print member
75
76
77 y_n_input = raw_input('Now lets change the password for all the members in this
group(y/n)')
78 if y_n_input != 'y':
79  exit(0)
80
81 l_tgt_username = "dbsnmp"
82 l_old_password = "secret1"
83 l_new_password = "secret2"
84
85 update_db_pwd_for_group(l_grp_name, l_old_password, l_new_password)
```

3.3.2.1 Script Analysis

Table 3–9 provides an analysis of each line of the code.

Table 3–9 Line-by_Line Script Analysis

Lines	Description
1-2	Imports all EM CLI verbs, and imports VerbExecutionError.
3-4	Imports Jython libraries.
6-25	Defines the job where the script updates the database password for each member of the l_grp_name group. After each successful job completion, the script displays a message to the user, and waits 2 seconds before processing the next job, unless there are any errors or all database passwords are updated.
27-43	Defines variables for updating the database password on each target member of the l_grp_name group. While the script successfully updates the database password, it provides the following message to the user before proceeding to update the password of the next database target: Changing the password for member : name = *database_name* type = *database_type*
45-51	Defines a loop to get all members from the l_grp_name group and update the password for each member as defined in line #85. When the script starts processing this loop, it provides this message to the user: Changing the password for group - *l_grp_name* from *l_old_password* to *l_new_password*

Table 3–9 (Cont.) Line-by_Line Script Analysis

Lines	Description
54-58	Necessary connection to OMS to retrieve all targets. Before connecting to the OMS, you must set the OMS connection details using the set_client_property() function. This sets the OMS URL to https://myoms.com/em and enables the client to trust all certificates. Note that none of these details are stored in disk. These details are stored in memory and only last for a single script execution. For more information on client properties, enter help('client_properties') from the interactive shell. You can define EMCLI_OMS_URL and EMCLI_TRUSTALL variables as environment variables if you do not want to set these in your script. If you have downloaded certificates somewhere, you can also use the environment variable EMCLI_CERT_LOC to point to the certificate directory. In this case, you do not need EMCLI_TRUSTALL. For more information, see Section 3.2.3, "Interactive Mode — Connecting to an Oracle Management Server (OMS)".
60	Provides a login to the OMS. You can remove this if you have set up EM CLI with autologin. For more information about setup and autologin, see Section 2.2.3, "Downloading and Deploying the EM CLI Client with the Script Option" and the setup and the login verbs.
63	l_grp_name is a variable for the group name
65	l_group_members is a variable for the array of the database name and type.
69	Adds members from l_group_members to the l_grp_name group
71-74	Provides a message to users while the script is adding members to the l_grp_name group
77-79	Provides a prompt to users to decide if they want to continue with the script. If the user enters n, then the script exits. If the user enters y, then the script proceeds.
81	l_tgt_username is a variable for the user name of the database owner.
82	l_old_password is a variable for the existing password associated with the database owner.
83	l_new_password is a variable for the new password associated with the database owner.
85	Replaces the existing password with the new password for all members of the l_grp_name group.

3.3.2.2 Script Output

Running Example 3–13, "dbPasswordChange.py" provides the following output:

Example 3–14 Output From dbPasswordChange.py

```
$ emcli @myScript.py user
Enter password : **********

Listing members for group maurGroup
('Target Name': 'aixsdbsi', 'Target Type': 'oracle_database')
('Target Name': 'winsidb1', 'Target Type': 'oracle_database')
('Target Name': 'db10g', 'Target Type': 'rac_database')
('Target Name': 'solcdbone', 'Target Type': 'rac_database')
Now let's change the password for all the members in this group (y/n)y
Changing the password for group - maurGroup from secret1 to secret2
Changing the password for member : name = aixsdbsi type = oracle_database
Job completed successfully
Changing the password for member : name = winsidb1 type = oracle_database
Job completed successfully
Changing the password for member : name = db10g type = rac_database
```

```
Job completed successfully
Changing the password for member : name = solcdbone type = rac_database
Job completed successfully

Logout successful
```

3.3.3 Promoting Discovered Targets

To assist you in writing scripts, this section analyzes a sample script that promotes discovered Oracle Database targets.

Consider a corporate environment where databases are added for each user requesting an instance using their UI. Most companies automate the entire process from database creation, adding data files, and so on. As part of your automation process, at the end, you can add this script.

Example 3–15, "promote_discovered_dbs.py" promotes the databases automatically, which are then ready to be monitored by Enterprise Manager. This removes the necessity for you to have to log in to the Enterprise Manager Cloud Control UI and promote the databases manually.

> **Note:** Line numbers are provided only for explanatory purposes for Table 3–10. For a copy-ready script, see Example A–7 in Appendix A, "Sample Scripts".

Example 3–15 promote_discovered_dbs.py

```
#Disclaimer
#EXCEPT WHERE EXPRESSLY PROVIDED OTHERWISE, THE SITE, AND ALL CONTENT PROVIDED ON
#OR THROUGH THE SITE, ARE PROVIDED ON AN "AS IS" AND "AS AVAILABLE" BASIS. ORACLE
#EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED,
#INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
#FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT WITH RESPECT TO THE SITE AND ALL
#CONTENT PROVIDED ON OR THROUGH THE SITE. ORACLE MAKES NO WARRANTY THAT: (A) THE
#SITE OR CONTENT WILL MEET YOUR REQUIREMENTS; (B) THE SITE WILL BE AVAILABLE ON AN
#UNINTERRUPTED, TIMELY, SECURE,OR ERROR-FREE BASIS; (C) THE RESULTS THAT MAY BE
#OBTAINED FROM THE USE OF THE SITE OR ANY CONTENT PROVIDED ON OR THROUGH THE SITE
#WILL BE ACCURATE OR RELIABLE; OR (D) THE QUALITY OF ANY CONTENT PURCHASED OR
#OBTAINED BY YOU ON OR THROUGH THE SITE WILL MEET YOUR EXPECTATIONS.
#ANY CONTENT ACCESSED, DOWNLOADED OR OTHERWISE OBTAINED ON OR THROUGH THE USE OF
#THE SITE IS USED AT YOUR OWN DISCRETION AND RISK. ORACLE SHALL HAVE NO
#RESPONSIBILITY FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM OR LOSS OF DATA THAT
#RESULTS FROM THE DOWNLOAD OR USE OF CONTENT.
#ORACLE RESERVES THE RIGHT TO MAKE CHANGES OR UPDATES TO, AND MONITOR THE USE OF,
#THE SITE AND CONTENT PROVIDED ON OR THROUGH THE SITE AT ANY TIME WITHOUT NOTICE.

 1 from emcli.exception import VerbExecutionError
 2 import sys
 3
 4 alltargets=False
 5 targetparms=0
 6 uname=''
 7 pword=''
 8 url=''
 9 monitor_pw=''
10
11 def helpUsage():
12    print 'Usage: promote_discovered_dbs.py [-help]'
13    print '[-all] Add all discovered Single Instance DBs'
```

```
 14    print '[-targets <target1:target2:...] Add only targets listed'
 15    sys.exit()
 16
 17 for i in range(len(sys.argv)):
 18    if sys.argv[i] in ("-help"):
 19            helpUsage()
 20    elif sys.argv[i] in ("-targets"):
 21            if i+1 < len(sys.argv):
 22                    targetparms = sys.argv[i+1]
 23    else:
 24                    print 'Usage: promote_discovered_dbs.py [-help]'
 25                    print '[-all] Add all discovered Single Instance DBs'
 26                    print '[-targets <target1:target2:...] Add only targets
listed'
 27                    sys.exit()
 28    elif sys.argv[i] in ("-url"):
 29            if i+1 < len(sys.argv):
 30                    url = sys.argv[i+1]
 31    elif sys.argv[i] in ("-username"):
 32            if i+1 < len(sys.argv):
 33                    uname = sys.argv[i+1]
 34    elif sys.argv[i] in ("-password"):
 35            if i+1 < len(sys.argv):
 36                    pword = sys.argv[i+1]
 37    elif sys.argv[i] in ("-monitor_pw"):
 38            if i+1 < len(sys.argv):
 39                    monitor_pw = sys.argv[i+1]
 40    elif sys.argv[i] in ("-all"):
 41                    alltargets = True
 42
 43 # Make sure user did not specify target list and all targets.
 44 if alltargets<>0 and targetparms <>0:
 45    print 'Cannot specify target list and all switch'
 46    print 'Usage: promote_discovered_dbs.py -url <EM URL> -username <username>
-password <password> -monitor_pw <password>'
 47    print '[-all] Add all discovered SI Databses'
 48    print '[-targets <target1:target2:...] Add only list targets'
 49    print '[-help]'
 50    sys.exit()
 51
 52 if len(uname)==0 or len(pword)==0 or len(url)==0:
 53         print 'Missing required arguments (-url, -username, -password)'
 54         print 'Usage: promote_discovered_dbs.py -url <EM URL> -username
<username> -password <password> -monitor_pw <password>'
 55         print '[-all] Add all discovered SI Databses'
 56         print '[-targets <target1:target2:...] Add only list targets'
 57         print '[-help]'
 58         sys.exit()
 59
 60 # Set Connection properties and logon
 61 set_client_property('EMCLI_OMS_URL',url)
 62 set_client_property('EMCLI_TRUSTALL','true')
 63 login(username=uname,password=pword)
 64
 65 cred_str = "UserName:dbsnmp;password:" + monitor_pw + ";Role:Normal"
 66
 67 if targetparms <> 0:
 68    targetparms = targetparms.replace(":",":oracle_database;")+":oracle_
database"
 69    target_array = get_
```

```
targets(unmanaged=True,properties=True,targets=targetparms).out()['data']
 70 elif alltargets:
 71    target_array = get_targets(targets="oracle_
database",unmanaged=True,properties=True ).out()['data']
 72 else:
 73    print 'Missing required arguments (-targets or -all)'
 74    helpUsage()
 75
 76 if len(target_array) > 0:
 77    for target in target_array:
 78            print 'Adding target ' + target['Target Name'] + '...',
 79
 80            for host in str.split(target['Host Info'],";"):
 81                if host.split(":")[0] == "host":]
 82                    print host.split(":")[1]
 83            try:
 84                res1 = add_target(type='oracle_database',name=target['Target
Name'],host=host.split(":")[1], credentials=cred_
str,properties=target['Properties'])
 85                print 'Succeeded'
 86            except VerbExecutionError, e:
 87                print 'Failed'
 88                print e.error()
 89                print 'Exit code:'+str(e.exit_code())
 90 else:
 91    print 'INFO: There are no targets to be promoted. Please verify the targets
in Enterprise Manager webpages.'
 92
```

3.3.3.1 Script Analysis

Table 3–10 provides an analysis of each line of the code.

Table 3–10 Line-by_Line Script Analysis

Lines	Description
1	Imports all EM CLI verbs, and imports VerbExecutionError.
2	Imports Jython libraries.
4-9	Sets variables: ■ uname: User name that allows access to Enterprise Manager ■ pword: Password associated with the user name ■ url: Enterprise Manager URL ■ monitor_pw: Password that allows monitoring of targets
10	Defines input arguments.

11-15	Defines the message displayed to the user if they run the script with invalid or missing arguments:
	`Usage: promote_discovered_dbs.py [-help]`
	`[-all] Add all discovered Single Instance DBs`
	`[-targets <target1:target2:...] Add only targets listed`

17-41	Defines a For loop that checks that the input variables (defined in lines 4 to 9) are valid and present, otherwise the script terminates and displays the message defined in lines 11-15.

Table 3–10 (Cont.) Line-by_Line Script Analysis

Lines	Description
43-50	Defines an If statement to check that the user doesn't provide the `-targets` and the `-all` arguments when running the script. If the user enters both arguments, then the scripts terminates and displays the message defined in lines 11-15.
52-58	Defines an If statement to check that user provides the user name, password, and URL of Enterprise Manager when running the script. If any of these arguments are missing, then the scripts terminates and displays the message defined in lines 11-15.
60-62	Necessary connection to OMS to retrieve all targets. Before connecting to the OMS, you must set the OMS connection details using the set_client_property() function. This sets the OMS URL to https://myoms.com/em and enables the client to trust all certificates. Note that none of these details are stored in disk. These details are stored in memory and only last for a single script execution. For more information on client properties, enter help('client_properties') from the interactive shell. You can define EMCLI_OMS_URL and EMCLI_TRUSTALL variables as environment variables if you do not want to set these in your script. If you have downloaded certificates somewhere, you can also use the environment variable EMCLI_CERT_LOC to point to the certificate directory. In this case, you do not need EMCLI_TRUSTALL. For more information, see Section 3.2.3, "Interactive Mode — Connecting to an Oracle Management Server (OMS)".
63	Provides a login to the OMS. You can remove this if you have set up EM CLI with autologin. For more information about setup and autologin, see Section 2.2.3, "Downloading and Deploying the EM CLI Client with the Script Option" and the setup and the login verbs.
65	Defines a variable for the credential string required for monitoring targets.
67	Defines an if statement to determine if the `-targets` argument is provided and if targets exist.
68	Sets the value for target_params
69	Sets the values for target_array, using the targets (where the list of targets is defined by targetparams), unmanaged, and properties parameters of the get_targets verb. When it is set for the first target, the script then moves on to the next target.
70-71	Defines an else if statement to set the values for target_array if the -all option is provided when running the script, using the targets, unmanaged, and properties parameters of the get_targets verb. When it is set for the first target, the script then moves on to the next target.
72-74	Defines an else statement in case the -targets or -all options are not provided when running the script. If this happens, the script terminates and displays the message defined in lines 11-15.
76-78	Determines if there is data in the array, and if there is data, the script displays a message similar to the following: `Adding target abchost.us.example.com... host.us.example.com`
80-82	Extracts the host name from the host information
83-88	Adds the targets to the Management Repository using the add_target verb and displays the following message: Succeeded If the script fails to add targets to the Management Repository, then it displays the following message: Failed

Table 3–10 (Cont.) Line-by_Line Script Analysis

Lines	Description

90	From line 76, if there are no targets in the array, the script terminates, and displays the following message to the user:
	`INFO: There are no targets to be promoted. Please verify the targets in Enterprise Manager webpages.`
	`Logout successful`

3.3.3.2 Script Output

Running Example 3–15, "promote_discovered_dbs.py" with various options provides the following outputs:

Example 3–16 Output from promote_discovered_dbs.py with -all option

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1 -all
Adding target sid7458.us.example.com... host.us.example.com
Succeeded
Logout successful
```

Example 3–17 Output from promote_discovered_dbs.py with -targets option

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1 -targets
sid7458.us.example.com
Adding target sid7458.us.example.com... host.us.example.com
Succeeded
Adding target db1... host.us.example.com
Succeeded
Adding target db2... host.us.example.com
Succeeded
Adding target db3... host.us.example.com
Succeeded
Logout successful
```

Example 3–18 Output from promote_discovered_dbs.py with -targets option

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1 -targets db1:db2:db3
Adding target db1... host.us.example.com
Succeeded
Adding target db2... host.us.example.com
Succeeded
Adding target db3... host.us.example.com
Succeeded
Logout successful
```

Example 3–19 Output from promote_discovered_dbs.py where the specified targets do not exist

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1 -targets abc
INFO: There are no targets to be promoted. Please verify the targets in Enterprise
Manager webpages.
```

```
Logout successful
```

Example 3–20 Output from promote_discovered_dbs.py where no targets are available for promotion

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1 -all
INFO: There are no targets to be promoted. Please verify the targets in Enterprise
Manager webpages.
Logout successful
```

Example 3–21 Output from promote_discovered_dbs.py where the -all or -targets option is missing

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1
Missing required arguments (-targets or -all)
Usage: promote_discovered_dbs.py [-help]
[-all] Add all discovered Single Instance DBs
[-targets <target1:target2:...] Add only targets listed
Logout successful
```

3.4 Using the Generic 'List' Verb

EM CLI provides dozens of listing verbs, such as list, get, show, and describe. Rather than selecting from all of these choices, EM CLI provides a generic list verb that you can execute with various types of queries.

The generic list verb provides the following benefits:

- Backed by a RESTful web service
- Generates JavaScript Object Notation (JSON) for script use and standard output for command-line use
- Can specify your own custom SQL to retrieve data from the repository using repository views

3.4.1 Selected list Verb Use Cases

The following sections provide examples of using the list verb for various purposes.

3.4.1.1 Listing Registered Resources

The list verb supports describing the registered listable resources.

To list all registered resource groups and resources:

```
emcli list -help
```

To describe a specific resource:

```
emcli list -resource="<resource_name>" -help
```

This provides a list of all of the columns along with descriptions.

To list data:

```
emcli list -resource="<resource_name>"
```

To list a specific number of columns:

```
emcli list -resource="<resource_name>" -columns="col1,col2"
```

This command lists only col1 and col2 columns from the specified resource.

3.4.1.2 Searching for Data

The list verb supports search capabilities.

To search using the list verb:

```
emcli list -resource="<resource_name>" -search="<column>='<value>'"
```

To specify multiple search conditions:

```
emcli list -resource="<resource_name>" -search="<column1> = '<value1>'"
-search="<colum2> = '<value2>'"
```

3.4.1.3 Registering Resources with the Bind Parameter

To list resources with the bind option:

```
emcli list -resource="<resource_name>" -bind="col1 = "'val1'"
```

This is required for a few resources that require bind parameters as specific input.

3.4.1.4 Listing with End-user Defined SQL

To execute user-defined SQL using the -sql option:

```
emcli list -sql='select * from mgmt$target'
```

The SQL provided in the -sql option is executed as the Enterprise Manager user MGMT_VIEW, which has read-only access to the Enterprise Manager published MGMT$ database views in the SYSMAN schema. The -sql option requires Super Administrator privileges.

3.5 Using the Registered Clients Page

The registered clients page shows all of the EM CLI client installations, and enables you to clean up or delete unused installations.

Note the following characteristics of this page:

- Registered clients are only visible to super users.
- You can search for existing registered clients by host name and version.

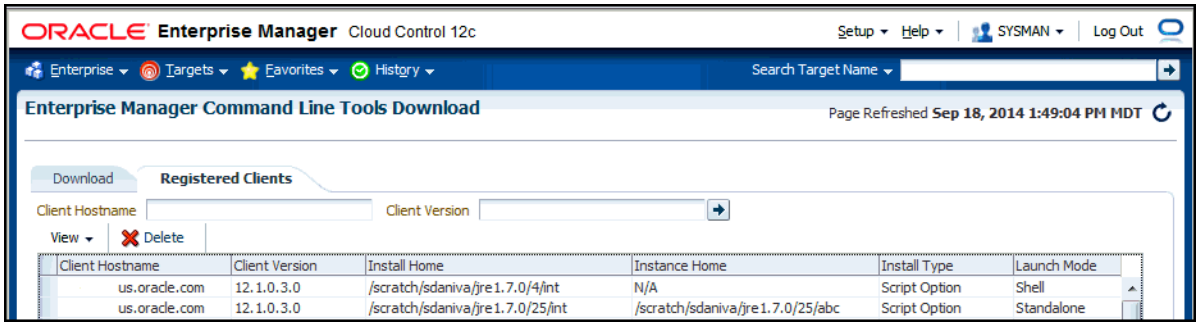
3.5.1 Accessing the Page

To access the Registered Clients page and display its contents:

1. From the Enterprise Manager Cloud Control console Setup menu, select Command Line Interface.

   The Command Line Interface tab appears. The figure below shows the page with a couple of registered clients.

Figure 3–1 Registered Client Entries



2. To select the columns to be displayed, select Columns from the View drop-down, then select Manage Columns. Choose any columns to be hidden from the pop-up, then click **OK**.

Column definitions are as follows:

- Client Hostname — Host name of the client where EM CLI is installed.
- Client Version — Version of the EM CLI client.
- Install Home — Complete path of the directory where the EM CLI client is installed.
- Instance Home — Complete path of the directory of the EM CLI instance home.
- Install Type — Can be Standalone, Shell, or Script.
- Setup By — EM CLI was set up by this OS user.
- Auto Login — Enabled or disabled.
- Trust All Certificates — Enabled or disabled.
- Setup at Time — Time at which EM CLI was set up.
- Last Synced At — Time when the sync verb was last executed.
- Last Login At — Time when the last login occurred.
- Last Logged In User — Enterprise Manger user last logged in.
- EM URL — URL of Enterprise Manager.

3.5.2 Deleting an Entry from the Table

Deleting an entry from the table in this page only deletes it from the repository, but does not delete the files from the client host where EM CLI is installed. After deletion, when EM CLI is launched from the same deleted entries directory, it re-adds the entry into the table.

For example, suppose a row in the table indicates that EM CLI is installed in host.example.com at /u01/dir. If you delete this row from the table, this action does not delete the files in /u01/dir at host.example.com. Now if you log in to host.example.com and execute emcli setup, Enterprise Manger adds the same row into the table which was deleted earlier.

4

Advanced EM CLI Script Examples

This chapter provides examples of using EM CLI to write scripts and automate routine tasks. To use these scripts, Oracle recommends that you are experienced with scripting languages and familiar with Jython (Python for the Java platform).

4.1 Changing Lifecycle Status Properties

To assist you in writing scripts, this section analyzes a sample script that changes the lifecycle status properties.

Example 4–1, "LifeCyclePropertyChange.py" enables an Enterprise Manager administrator to change the lifecycle status of all the Oracle databases (release 11.2) in their test environment from Test to Production. Without this script, you would have to log in to the Enterprise Manager Cloud Control console, and identify all the release 11.2 databases, then manually change the property to Production for each database target from the target's home page.

You can reuse this script whenever there is a request to change a set of targets to a different Lifecycle status.

> **Note:** Line numbers are provided only for explanatory purposes for Table 4–1. For a copy-ready script, see Example A–5 in Appendix A, "Sample Scripts".

Example 4–1 LifeCyclePropertyChange.py

```
#Disclaimer
#EXCEPT WHERE EXPRESSLY PROVIDED OTHERWISE, THE SITE, AND ALL CONTENT PROVIDED ON
#OR THROUGH THE SITE, ARE PROVIDED ON AN "AS IS" AND "AS AVAILABLE" BASIS. ORACLE
#EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED,
#INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
#FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT WITH RESPECT TO THE SITE AND ALL
#CONTENT PROVIDED ON OR THROUGH THE SITE. ORACLE MAKES NO WARRANTY THAT: (A) THE
#SITE OR CONTENT WILL MEET YOUR REQUIREMENTS; (B) THE SITE WILL BE AVAILABLE ON AN
#UNINTERRUPTED, TIMELY, SECURE,OR ERROR-FREE BASIS; (C) THE RESULTS THAT MAY BE
#OBTAINED FROM THE USE OF THE SITE OR ANY CONTENT PROVIDED ON OR THROUGH THE SITE
#WILL BE ACCURATE OR RELIABLE; OR (D) THE QUALITY OF ANY CONTENT PURCHASED OR
#OBTAINED BY YOU ON OR THROUGH THE SITE WILL MEET YOUR EXPECTATIONS.
#ANY CONTENT ACCESSED, DOWNLOADED OR OTHERWISE OBTAINED ON OR THROUGH THE USE OF
```

```
#THE SITE IS USED AT YOUR OWN DISCRETION AND RISK. ORACLE SHALL HAVE NO
#RESPONSIBILITY FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM OR LOSS OF DATA THAT
#RESULTS FROM THE DOWNLOAD OR USE OF CONTENT.
#ORACLE RESERVES THE RIGHT TO MAKE CHANGES OR UPDATES TO, AND MONITOR THE USE OF,
#THE SITE AND CONTENT PROVIDED ON OR THROUGH THE SITE AT ANY TIME WITHOUT NOTICE.
 1 from emcli import *
 2
 3 search_list = ['PROPERTY_NAME=\'DBVersion\'','TARGET_TYPE=
 \'oracle_database\'','PROPERTY_VALUE LIKE \'11.2%\'']
 4
 5 if len(sys.argv) == 2:

 6        print login(username=sys.argv[0])
 7        l_prop_val_to_set = sys.argv[1]
 8        l_targets = list(resource="TargetProperties", search=search_list,
columns="TARGET_NAME,TARGET_TYPE,PROPERTY_NAME")
 9        for target in l_targets.out()['data']:
10             t_pn =  'LifeCycle Status'
11             print "INFO: Setting Property name " + t_pn + " to value " + l_
prop_val_to_set
12             print set_target_property_value(property_records=target['TARGET_
NAME']+":"+target['TARGET_TYPE']+":"+t_pn+":"+l_prop_val_to_set)
13  else: ]
14        print "\n ERROR: Property value argument is missing"
15        print "\n INFO: Format to run this file is filename.py <username>
<Database Target LifeCycle Status Property Value>"
```

4.1.1 Script Analysis

Table 4–1 provides an analysis of each line of the code.

Table 4–1 Line-by_Line Script Analysis

Lines	Description
1	Jython import construct to import all EM CLI verb functions in the current program.
3	search_list is a variable to pass to the search option in the list verb. This example uses escape characters to represent single quotes. To pass more than one value for the same option in the list verb, define as comma separated values, surrounded by square brackets.
5	Defines an if condition to ensure the user provides two arguments with the script, otherwise the script prints an error message (defined in lines #15, 16)
6	Provides a login to Enterprise Manager. You can remove this if you have set up EM CLI with autologin. For more information about setup and autologin, see Section 2.2.3, "Downloading and Deploying the EM CLI Client with the Script Option" and the setup and the login verbs.
7	l_prop_val_to_set is a variable. This is the property value to be set. Remember that this script is changing this value from Test to Production. You can change this value to any acceptable Lifecycle property value. For a list of valid values, see the modify_lifecycle_stage_name verb or the *Oracle Enterprise Manager Cloud Control Administrator's Guide*.
8	Stores the output of the list verb in l_targets. In the list verb, this script passes the resource as TargetProperties, and search as the search_list variable. This script specifies three columns: ▪ target_name ▪ target_type ▪ property_name

Table 4–1 (Cont.) Line-by_Line Script Analysis

Lines	Description
9	Defines a for loop. The data in l_targets is available in JSON format. This loop iterates through the information target property information returned from the list verb. For information about JSON processing, see Section 3.2.6.2, "JSON Processing"
10	Sets t_pn to the LifeCycle Status value.
11	Provides a progress message to the user stating that the script is setting the LifeCycle Status to the value passed to the script from the command line.
12	Defines the set_target_property_value verb, which sets the value using the property_records option. When this verb is set for a target pair, it moves to the next one. This example shows three databases, but in reality, use this script for a larger number of databases.
13 -15	Else statement combined with the if condition in line #5. If the arguments specified in line #5 are not provided correctly, then the script displays one of the following error messages: ▪ Property value argument is missing ▪ Format to run this file is filename.py <username> <Database Target LifeCycle Status Property Value>

4.1.2 Script Output

Running Example 4–1, "LifeCyclePropertyChange.py" provides the following output:

Example 4–2 Output from LifeCyclePropertyChange.py

```
$ emcli @myScript.py user Production

Login successful

INFO: Setting Property name LifeCycle Status to value Production for db1
Properties updated successfully
INFO: Setting Property name LifeCycle Status to value Production for db2
Properties updated successfully
INFO: Setting Property name LifeCycle Status to value Production for db3
Properties updated successfully

Logout successful
```

4.2 Changing Your Database Password

To assist you in writing scripts, this section analyzes a sample script that changes the password of your database.

Example 4–3, "dbPasswordChange.py" is useful if you want to reset the database password on a regular basis for security compliance. If you change the database password in a target database, then Enterprise Manager monitoring for that target database is unavailable. To ensure consistent monitoring, you would have to log in to the Enterprise Manager Cloud Control UI, select the required database target and change the password for each and every database, which is very time-consuming.

By using Example 4–3, "dbPasswordChange.py", you can add a number of databases to a group and then change the password for all these databases within the group.

Note: Line numbers are provided only for explanatory purposes for Table 4–2. For a copy-ready script, see Example A–6 in Appendix A, "Sample Scripts".

Example 4–3 dbPasswordChange.py

```
#Disclaimer
#EXCEPT WHERE EXPRESSLY PROVIDED OTHERWISE, THE SITE, AND ALL CONTENT PROVIDED ON
#OR THROUGH THE SITE, ARE PROVIDED ON AN "AS IS" AND "AS AVAILABLE" BASIS. ORACLE
#EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED,
#INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
#FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT WITH RESPECT TO THE SITE AND ALL
#CONTENT PROVIDED ON OR THROUGH THE SITE. ORACLE MAKES NO WARRANTY THAT: (A) THE
#SITE OR CONTENT WILL MEET YOUR REQUIREMENTS; (B) THE SITE WILL BE AVAILABLE ON AN
#UNINTERRUPTED, TIMELY, SECURE,OR ERROR-FREE BASIS; (C) THE RESULTS THAT MAY BE
#OBTAINED FROM THE USE OF THE SITE OR ANY CONTENT PROVIDED ON OR THROUGH THE SITE
#WILL BE ACCURATE OR RELIABLE; OR (D) THE QUALITY OF ANY CONTENT PURCHASED OR
#OBTAINED BY YOU ON OR THROUGH THE SITE WILL MEET YOUR EXPECTATIONS.
#ANY CONTENT ACCESSED, DOWNLOADED OR OTHERWISE OBTAINED ON OR THROUGH THE USE OF
#THE SITE IS USED AT YOUR OWN DISCRETION AND RISK. ORACLE SHALL HAVE NO
#RESPONSIBILITY FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM OR LOSS OF DATA THAT
#RESULTS FROM THE DOWNLOAD OR USE OF CONTENT.
#ORACLE RESERVES THE RIGHT TO MAKE CHANGES OR UPDATES TO, AND MONITOR THE USE OF,
#THE SITE AND CONTENT PROVIDED ON OR THROUGH THE SITE AT ANY TIME WITHOUT NOTICE.

 1 from emcli  import *
 2 from emcli.exception import VerbExecutionError
 3 import sys
 4 import time
 5
 6 def check_job_status(job):
 7   count=0
 8   while (count < 10):
 9    count = count + 1
10    obj = emcli.get_jobs(job_id=job)
11    #print obj.out()
12    for entry in obj.out()['data']:
13        l_status = entry['Status ID']
14        l_exec_id = entry['Execution ID']
15        #print entry['Status ID']
16        if (l_status == '5'):
17            print "Job completed successfully"
18            count=100
19        elif (l_status == '4'):
20            l_resp = get_job_execution_detail(execution=l_exec_id,
showOutput=True, xml=True)
21            print "Job failed, error details "
22            print "Output " + str(l_resp.out())
23            count=100
24        else:
25            time.sleep(2)
26
27 def update_db_pwd_for_target(p_target_name, p_target_type, p_old_password, p_
new_password):
28    l_target_name = p_target_name
29    l_target_type = p_target_type
30    print "Changing the password for member : name = " + l_target_name + " type
= " + l_target_type
31    try :
```

```
32        l_resp = update_db_password (target_name=l_target_name,
33                                     target_type = l_target_type,
34                                     change_at_target="yes",
35                                     user_name="dbsnmp",
36                                     old_password=p_old_password,
37                                     new_password=p_new_password,
38                                     retype_new_password=p_new_password)
39        l_job_submitted = l_resp.out()['JobId']
40        check_job_status(l_job_submitted)
41    except emcli.exception.VerbExecutionError, e:
42        print "ERROR : Change Password failed for name = " + l_target_name + "
type = " + l_target_type
43        print "ERROR : " + e.error()
44
45 def update_db_pwd_for_group(p_group, p_old_password, p_new_password):
46    print "Changing the password for group - " + p_group + " from " + p_old_
password + " to " + p_new_password
47    members = get_group_members(name=p_group).out()['data']
48    for member in members:
49        l_target_name = member['Target Name']
50        l_target_type = member['Target Type']
51        update_db_pwd_for_target(l_target_name, l_target_type, p_old_password,
p_new_password)
52
53
54 #Set the OMS URL to connect to
56 set_client_property('EMCLI_OMS_URL','https://myoms.com/em')
57 #Accept all the certificates
58 set_client_property('EMCLI_TRUSTALL','true')
59
60 login(username=sys.argv[0])
61
62
63 l_grp_name = 'maurGroup'
64
65 l_group_members = ['db1:oracle_database','db2:oracle_database','db3:rac_
database']
66
67
68
69 res = create_group(name = l_grp_name, add_targets = l_group_members)
70
71 print "Listing members for group " + l_grp_name
72
73 for member in get_group_members(name=l_grp_name).out()['data']:
74    print member
75
76
77 y_n_input = raw_input('Now lets change the password for all the members in this
group(y/n)')
78 if y_n_input != 'y':
79  exit(0)
80
81 l_tgt_username = "dbsnmp"
82 l_old_password = "secret1"
83 l_new_password = "secret2"
84
85 update_db_pwd_for_group(l_grp_name, l_old_password, l_new_password)
```

4.2.1 Script Analysis

Table 4–2 provides an analysis of each line of the code.

Table 4–2 Line-by_Line Script Analysis

Lines	Description
1-2	Imports all EM CLI verbs, and imports VerbExecutionError.
3-4	Imports Jython libraries.
6-25	Defines the job where the script updates the database password for each member of the l_grp_name group. After each successful job completion, the script displays a message to the user, and waits 2 seconds before processing the next job, unless there are any errors or all database passwords are updated.
27-43	Defines variables for updating the database password on each target member of the l_grp_name group. While the script successfully updates the database password, it provides the following message to the user before proceeding to update the password of the next database target: Changing the password for member : name = *database_name* type = *database_type*
45-51	Defines a loop to get all members from the l_grp_name group and update the password for each member as defined in line #85. When the script starts processing this loop, it provides this message to the user: Changing the password for group - *l_grp_name* from *l_old_password* to *l_new_password*
54-58	Necessary connection to OMS to retrieve all targets. Before connecting to the OMS, you must set the OMS connection details using the set_client_property() function. This sets the OMS URL to https://myoms.com/em and enables the client to trust all certificates. Note that none of these details are stored in disk. These details are stored in memory and only last for a single script execution. For more information on client properties, enter help('client_properties') from the interactive shell. You can define EMCLI_OMS_URL and EMCLI_TRUSTALL variables as environment variables if you do not want to set these in your script. If you have downloaded certificates somewhere, you can also use the environment variable EMCLI_CERT_LOC to point to the certificate directory. In this case, you do not need EMCLI_TRUSTALL. For more information, see Section 3.2.3, "Interactive Mode — Connecting to an Oracle Management Server (OMS)".
60	Provides a login to the OMS. You can remove this if you have set up EM CLI with autologin. For more information about setup and autologin, see Section 2.2.3, "Downloading and Deploying the EM CLI Client with the Script Option" and the setup and the login verbs.
63	l_grp_name is a variable for the group name
65	l_group_members is a variable for the array of the database name and type.
69	Adds members from l_group_members to the l_grp_name group
71-74	Provides a message to users while the script is adding members to the l_grp_name group
77-79	Provides a prompt to users to decide if they want to continue with the script. If the user enters n, then the script exits. If the user enters y, then the script proceeds.
81	l_tgt_username is a variable for the user name of the database owner.
82	l_old_password is a variable for the existing password associated with the database owner.
83	l_new_password is a variable for the new password associated with the database owner.

Table 4–2 (Cont.) Line-by_Line Script Analysis

Lines	Description
85	Replaces the existing password with the new password for all members of the l_grp_name group.

4.2.2 Script Output

Running Example 4–3, "dbPasswordChange.py" provides the following output:

Example 4–4 Output From dbPasswordChange.py

```
$ emcli @myScript.py user
Enter password : **********

Listing members for group maurGroup
('Target Name': 'aixsdbsi', 'Target Type': 'oracle_database')
('Target Name': 'winsidb1', 'Target Type': 'oracle_database')
('Target Name': 'db10g', 'Target Type': 'rac_database')
('Target Name': 'solcdbone', 'Target Type': 'rac_database')
Now let's change the password for all the members in this group (y/n)y
Changing the password for group - maurGroup from secret1 to secret2
Changing the password for member : name = aixsdbsi type = oracle_database
Job completed successfully
Changing the password for member : name = winsidb1 type = oracle_database
Job completed successfully
Changing the password for member : name = db10g type = rac_database
Job completed successfully
Changing the password for member : name = solcdbone type = rac_database
Job completed successfully

Logout successful
```

4.3 Promoting Discovered Targets

To assist you in writing scripts, this section analyzes a sample script that promotes discovered Oracle Database targets.

Consider a corporate environment where databases are added for each user requesting an instance using their UI. Most companies automate the entire process from database creation, adding data files, and so on. As part of your automation process, at the end, you can add this script.

Example 4–5, "promote_discovered_dbs.py" promotes the databases automatically, which are then ready to be monitored by Enterprise Manager. This removes the necessity for you to have to log in to the Enterprise Manager Cloud Control UI and promote the databases manually.

> **Note:** Line numbers are provided only for explanatory purposes for Table 4–3. For a copy-ready script, see Example A–7 in Appendix A, "Sample Scripts".

Example 4–5 promote_discovered_dbs.py

```
#Disclaimer
#EXCEPT WHERE EXPRESSLY PROVIDED OTHERWISE, THE SITE, AND ALL CONTENT PROVIDED ON
#OR THROUGH THE SITE, ARE PROVIDED ON AN "AS IS" AND "AS AVAILABLE" BASIS. ORACLE
#EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED,
```

```
#INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
#FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT WITH RESPECT TO THE SITE AND ALL
#CONTENT PROVIDED ON OR THROUGH THE SITE. ORACLE MAKES NO WARRANTY THAT: (A) THE
#SITE OR CONTENT WILL MEET YOUR REQUIREMENTS; (B) THE SITE WILL BE AVAILABLE ON AN
#UNINTERRUPTED, TIMELY, SECURE,OR ERROR-FREE BASIS; (C) THE RESULTS THAT MAY BE
#OBTAINED FROM THE USE OF THE SITE OR ANY CONTENT PROVIDED ON OR THROUGH THE SITE
#WILL BE ACCURATE OR RELIABLE; OR (D) THE QUALITY OF ANY CONTENT PURCHASED OR
#OBTAINED BY YOU ON OR THROUGH THE SITE WILL MEET YOUR EXPECTATIONS.
#ANY CONTENT ACCESSED, DOWNLOADED OR OTHERWISE OBTAINED ON OR THROUGH THE USE OF
#THE SITE IS USED AT YOUR OWN DISCRETION AND RISK. ORACLE SHALL HAVE NO
#RESPONSIBILITY FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM OR LOSS OF DATA THAT
#RESULTS FROM THE DOWNLOAD OR USE OF CONTENT.
#ORACLE RESERVES THE RIGHT TO MAKE CHANGES OR UPDATES TO, AND MONITOR THE USE OF,
#THE SITE AND CONTENT PROVIDED ON OR THROUGH THE SITE AT ANY TIME WITHOUT NOTICE.

 1 from emcli.exception import VerbExecutionError
 2 import sys
 3
 4 alltargets=False
 5 targetparms=0
 6 uname=''
 7 pword=''
 8 url=''
 9 monitor_pw=''
10
11 def helpUsage():
12    print 'Usage: promote_discovered_dbs.py [-help]'
13    print '[-all] Add all discovered Single Instance DBs'
14    print '[-targets <target1:target2:...] Add only targets listed'
15    sys.exit()
16
17 for i in range(len(sys.argv)):
18    if sys.argv[i] in ("-help"):
19            helpUsage()
20    elif sys.argv[i] in ("-targets"):
21            if i+1 < len(sys.argv):
22                    targetparms = sys.argv[i+1]
23    else:
24                    print 'Usage: promote_discovered_dbs.py [-help]'
25                    print '[-all] Add all discovered Single Instance DBs'
26                    print '[-targets <target1:target2:...] Add only targets
listed'
27                    sys.exit()
28    elif sys.argv[i] in ("-url"):
29            if i+1 < len(sys.argv):
30                    url = sys.argv[i+1]
31    elif sys.argv[i] in ("-username"):
32            if i+1 < len(sys.argv):
33                    uname = sys.argv[i+1]
34    elif sys.argv[i] in ("-password"):
35            if i+1 < len(sys.argv):
36                    pword = sys.argv[i+1]
37    elif sys.argv[i] in ("-monitor_pw"):
38            if i+1 < len(sys.argv):
39                    monitor_pw = sys.argv[i+1]
40    elif sys.argv[i] in ("-all"):
41                    alltargets = True
42
43 # Make sure user did not specify target list and all targets.
44 if alltargets<>0 and targetparms <>0:
```

```
 45    print 'Cannot specify target list and all switch'
 46    print 'Usage: promote_discovered_dbs.py -url <EM URL> -username <username>
-password <password> -monitor_pw <password>'
 47    print '[-all] Add all discovered SI Databses'
 48    print '[-targets <target1:target2:...] Add only list targets'
 49    print '[-help]'
 50    sys.exit()
 51
 52 if len(uname)==0 or len(pword)==0 or len(url)==0:
 53         print 'Missing required arguments (-url, -username, -password)'
 54         print 'Usage: promote_discovered_dbs.py -url <EM URL> -username
<username> -password <password> -monitor_pw <password>'
 55         print '[-all] Add all discovered SI Databses'
 56         print '[-targets <target1:target2:...] Add only list targets'
 57         print '[-help]'
 58         sys.exit()
 59
 60 # Set Connection properties and logon
 61 set_client_property('EMCLI_OMS_URL',url)
 62 set_client_property('EMCLI_TRUSTALL','true')
 63 login(username=uname,password=pword)
 64
 65 cred_str = "UserName:dbsnmp;password:" + monitor_pw + ";Role:Normal"
 66
 67 if targetparms <> 0:
 68    targetparms = targetparms.replace(":",":oracle_database;")+":oracle_
database"
 69    target_array = get_
targets(unmanaged=True,properties=True,targets=targetparms).out()['data']
 70 elif alltargets:
 71    target_array = get_targets(targets="oracle_
database",unmanaged=True,properties=True ).out()['data']
 72 else:
 73    print 'Missing required arguments (-targets or -all)'
 74    helpUsage()
 75
 76 if len(target_array) > 0:
 77    for target in target_array:
 78            print 'Adding target ' + target['Target Name'] + '...',
 79
 80            for host in str.split(target['Host Info'],";"):
 81                if host.split(":")[0] == "host":]
 82                    print host.split(":")[1]
 83            try:
 84                res1 = add_target(type='oracle_database',name=target['Target
Name'],host=host.split(":")[1], credentials=cred_
str,properties=target['Properties'])
 85                print 'Succeeded'
 86            except VerbExecutionError, e:
 87                print 'Failed'
 88                print e.error()
 89                print 'Exit code:'+str(e.exit_code())
 90 else:
 91    print 'INFO: There are no targets to be promoted. Please verify the targets
in Enterprise Manager webpages.'
 92
```

4.3.1 Script Analysis

Table 4–3 provides an analysis of each line of the code.

Table 4–3 Line-by_Line Script Analysis

Lines	Description
1	Imports all EM CLI verbs, and imports VerbExecutionError.
2	Imports Jython libraries.
4-9	Sets variables: ■ uname: User name that allows access to Enterprise Manager ■ pword: Password associated with the user name ■ url: Enterprise Manager URL ■ monitor_pw: Password that allows monitoring of targets
10	Defines input arguments.

11-15	Defines the message displayed to the user if they run the script with invalid or missing arguments:
	`Usage: promote_discovered_dbs.py [-help]`
	`[-all] Add all discovered Single Instance DBs`
	`[-targets <target1:target2:...] Add only targets listed`

17-41	Defines a For loop that checks that the input variables (defined in lines 4 to 9) are valid and present, otherwise the script terminates and displays the message defined in lines 11-15.
43-50	Defines an If statement to check that the user doesn't provide the `-targets` and the `-all` arguments when running the script. If the user enters both arguments, then the scripts terminates and displays the message defined in lines 11-15.
52-58	Defines an If statement to check that user provides the user name, password, and URL of Enterprise Manager when running the script. If any of these arguments are missing, then the scripts terminates and displays the message defined in lines 11-15.
60-62	Necessary connection to OMS to retrieve all targets. Before connecting to the OMS, you must set the OMS connection details using the set_client_property() function. This sets the OMS URL to https://myoms.com/em and enables the client to trust all certificates. Note that none of these details are stored in disk. These details are stored in memory and only last for a single script execution. For more information on client properties, enter help('client_properties') from the interactive shell. You can define EMCLI_OMS_URL and EMCLI_TRUSTALL variables as environment variables if you do not want to set these in your script. If you have downloaded certificates somewhere, you can also use the environment variable EMCLI_CERT_LOC to point to the certificate directory. In this case, you do not need EMCLI_TRUSTALL. For more information, see Section 3.2.3, "Interactive Mode — Connecting to an Oracle Management Server (OMS)".
63	Provides a login to the OMS. You can remove this if you have set up EM CLI with autologin. For more information about setup and autologin, see Section 2.2.3, "Downloading and Deploying the EM CLI Client with the Script Option" and the setup and the login verbs.
65	Defines a variable for the credential string required for monitoring targets.
67	Defines an if statement to determine if the `-targets` argument is provided and if targets exist.
68	Sets the value for target_params

Table 4–3 (Cont.) Line-by_Line Script Analysis

Lines	Description
69	Sets the values for target_array, using the targets (where the list of targets is defined by targetparams), unmanaged, and properties parameters of the get_targets verb. When it is set for the first target, the script then moves on to the next target.
70-71	Defines an else if statement to set the values for target_array if the -all option is provided when running the script, using the targets, unmanaged, and properties parameters of the get_targets verb. When it is set for the first target, the script then moves on to the next target.
72-74	Defines an else statement in case the -targets or -all options are not provided when running the script. If this happens, the script terminates and displays the message defined in lines 11-15.
76-78	Determines if there is data in the array, and if there is data, the script displays a message similar to the following: `Adding target abchost.us.example.com... host.us.example.com`
80-82	Extracts the host name from the host information
83-88	Adds the targets to the Management Repository using the add_target verb and displays the following message: Succeeded If the script fails to add targets to the Management Repository, then it displays the following message: Failed

90	From line 76, if there are no targets in the array, the script terminates, and displays the following message to the user:
	`INFO: There are no targets to be promoted. Please verify the targets in Enterprise Manager webpages.`
	`Logout successful`

4.3.2 Script Output

Running Example 4–5, "promote_discovered_dbs.py" with various options provides the following outputs:

Example 4–6 Output from promote_discovered_dbs.py with -all option

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1 -all
Adding target sid7458.us.example.com... host.us.example.com
Succeeded
Logout successful
```

Example 4–7 Output from promote_discovered_dbs.py with -targets option

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1 -targets
sid7458.us.example.com
Adding target sid7458.us.example.com... host.us.example.com
Succeeded
Adding target db1... host.us.example.com
Succeeded
Adding target db2... host.us.example.com
```

```
Succeeded
Adding target db3... host.us.example.com
Succeeded
Logout successful
```

Example 4–8 Output from promote_discovered_dbs.py with -targets option

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1 -targets db1:db2:db3
Adding target db1... host.us.example.com
Succeeded
Adding target db2... host.us.example.com
Succeeded
Adding target db3... host.us.example.com
Succeeded
Logout successful
```

Example 4–9 Output from promote_discovered_dbs.py where the specified targets do not exist

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1 -targets abc
INFO: There are no targets to be promoted. Please verify the targets in Enterprise
Manager webpages.
Logout successful
```

Example 4–10 Output from promote_discovered_dbs.py where no targets are available for promotion

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1 -all
INFO: There are no targets to be promoted. Please verify the targets in Enterprise
Manager webpages.
Logout successful
```

Example 4–11 Output from promote_discovered_dbs.py where the -all or -targets option is missing

```
$ emcli @promote_discovered_dbs.py -url https://host.us.example.com:7799/em
-username sysman -password welcome1 -monitor_pw welcome1
Missing required arguments (-targets or -all)
Usage: promote_discovered_dbs.py [-help]
[-all] Add all discovered Single Instance DBs
[-targets <target1:target2:...] Add only targets listed
Logout successful
```

5 Verb Reference

This chapter provides a complete listing of all EM CLI verbs in categorical as well as alphabetical order. Each verb provides complete syntax and usage information.

5.1 Verb Categories

This section lists all of the verbs for this release in the following categories:

- Pluggable Database Job Verbs
- Deployment Procedure Verbs
- Diagchecks Verbs
- Diagnostic Snapshots Verbs
- Discover and Push to Agents Verbs
- Execute Command Verbs
- Event and Incident Verbs
- Fusion Middleware Provisioning Verbs
- Group Verbs
- Incident Rules Verbs
- Installation Verbs
- Internal Metrics Verbs
- Job Verbs
- Licensing Verbs
- Log Management Verbs
- Masking Verbs
- Metric Collection and Alerts Verbs
- Metric Extension Verbs
- Metric Verbs
- Monitoring Template Verbs
- Notification Verbs
- OMS Configuration Properties
- OMS Plug-in Deployment Verbs
- Oracle Database as Service (DBaaS) Verbs
- Package Fusion Application Problem Verbs
- Patch Verbs
- Ping Subsytem Verbs
- Platform as a Service (PaaS) Verbs
- Pluggable Database Job Verbs
- Privilege Delegation Settings Verbs
- Provisioning Verbs
- Reconfig Job Verbs
- Redundancy Group Verbs
- Refresh Coherence Verbs
- Refresh WLS Domain Verbs
- Report Import/Export Verbs
- Secure Communication Verbs

- Self Update Verbs
- Services Verbs
- Siebel Verbs
- SiteGuard Verbs
- Software Library Verbs
- SSA Verbs
- System Verbs
- Target Data Verbs
- User-defined Metrics (UDM) Migration Verbs
- Upgrade Database Job Verbs
- User Administration Verbs
- User Session Administration Verbs

Basic Operational Verbs

Note: Only these verbs are available immediately after installation.

argfile
help
login
logout
setup
status
sync
version

Add Host Verbs

continue_add_host
get_add_host_status
list_add_host_platforms
list_add_host_sessions
retry_add_host
submit_add_host

Application Data Model Verbs

associate_target_to_adm
export_adm
list_adms
verify_adm

Agent Administration Verbs

get_agent_properties
get_agent_property
modify_monitoring_agent
resecure_agent
restart_agent
secure_agent
set_agent_property
start_agent
stop_agent

unsecure_agent

Agent Upgrade Verbs

get_agent_upgrade_status
get_signoff_agents
get_signoff_status
get_upgradable_agents
signoff_agents
upgrade_agents

Application Data Models Verbs

associate_target_to_adm
export_adm
import_adm
show_operations_list
update_audit_settings

Audit Settings Verbs

disable_audit
enable_audit
show_audit_settings
show_operations_list
update_audit_settings

Bare Metal Provisioning Verbs

bareMetalProvisioning

BI Publisher Reports Verbs

deploy_bipublisher_reports
deploy_bipublisher_selfupdates
grant_bipublisher_roles
revoke_bipublisher_roles
setup_bipublisher
unregister_bipublisher

Big Data Appliance

delete_bda_cluster
discover_bda_cluster
discover_cloudera_cluster
relocate_bda_target
show_bda_clusters

Blackout Verbs

add_blackout_reason
create_blackout
delete_blackout
get_blackout_details
get_blackout_reasons
get_blackout_targets
get_blackouts
stop_blackout

CFW Verbs

cancel_cloud_service_requests
delete_cloud_service_instances
delete_cloud_user_objects
get_cloud_service_instances
get_cloud_service_requests
get_cloud_user_objects

Chargeback Verbs

add_chargeback_entity
assign_charge_plan
assign_cost_center
create_charge_entity_type
create_charge_item
delete_charge_item
export_charge_plans
export_custom_charge_items
get_metering_data
import_charge_plans
import_custom_charge_items
list_chargeback_entities
list_chargeback_entity_types
list_charge_item_candidates
list_charge_plans
list_cost_centers
remove_chargeback_entity
unassign_charge_plan
unassign_cost_center

Cloning Verbs

clone_as_home
clone_crs_home
clone_database
clone_database_home
create_clone
extend_as_home
extend_crs_home
extend_rac_home

Compliance Verbs

associate_cs_targets
export_compliance_group
export_compliance_standard_rule
export_standard
fix_compliance_state
import_compliance_object
remove_cs_target_association

Configuration Data

get_config_searches
get_target_types
run_config_seaches

Configuration/Assocation History

disable_config_history
enable_config_history
set_config_history_retention_period

Configuration Compare

config_compare
get_config_templates

Connector Verbs

publish_change_request_ccc
update_ticket_status

Credential Verbs

clear_credential
clear_default_pref_credential
clear_monitoring_credential
clear_preferred_credential
create_credential_set
create_named_credential
delete_credential_set
delete_named_credential
get_credtype_metadata
get_duplicate_credentials
get_named_credential
list_named_credentials
merge_credentials
modify_named_credential
set_credential
set_default_pref_cred
set_monitoring_credential
set_preferred_credential
show_credential_set_info
show_credential_type_info
test_named_credential
update_host_password
update_monitoring_creds_from_agent
update_password
update_target_password

Credential Verbs - Oracle Database

update_db_password
update_credential_set

Database Job Verbs

create_database
data_transfer
dbimport
delete_database
refresh_database

Database Machine Targets Customer Support Identifier (CSI) Assignment Verbs

assign_csi_for_dbmachine_targets

Database Profile Job Verbs

create_dbprofile
describe_dbprofile_input
edit_dbprofile
list_dbprofiles
refresh_dbprofile

Data Subset Verbs

export_subset_definition
generate_subset
import_subset_definition
import_subset_dump
import_subset_dump
list_subset_definitions

DBaaS Verbs

See "*EM CLI for Administrator Flows*" in the *Enterprise Manager Cloud Administration Guide*.

Pluggable Database Job Verbs

delete_pluggable_database
pdb_backup
pdb_clone_management

Deployment Procedure Verbs

confirm_instance
delete_instance
describe_procedure_input
get_executions
get_instance_data
get_instance_status
get_instances
get_procedure_types
get_procedure_xml
get_procedures
get_retry_arguments
get_runtime_data
ignore_instance
reschedule_instance
resume_instance
retry_instance
save_procedure_input
stop_instance
submit_procedure
suspend_instance
update_and_retry_step
update_procedure_input

Diagchecks Verbs

apply_diagcheck_exclude
define_diagcheck_exclude
diagchecks_deploy_status
diagchecks_deploy_tglist

list_diagcheck_exclude_applies
list_diagcheck_exclusions
list_diagchecks
undeploy_diagchecks
update_diagchecks

Diagnostic Snapshots Verbs

create_diag_snapshot
delete_diag_snapshot

Discover and Push to Agents Verbs

delete_siebel
discover_coherence
discover_fa
discover_gf
discover_siebel
discover_wls
generate_discovery_input
refresh_fa
run_fa_diagnostics

Discovery Prechecks Verbs

fmw_discovery_prechecks

Execute Command Verbs

execute_hostcmd
execute_sql

Event and Incident Verbs

clear_problem
create_resolution_state
delete_resolution_state
get_resolution_states
modify_incident_rule
modify_resolution_state
publish_event

Fusion Middleware Provisioning Verbs

create_fmw_domain_profile
create_fmw_home_profile
create_inst_media_profile
delete_fmw_profile
describe_fmw_profile
list_fmw_profiles

Gold Image Verbs

create_custom_plugin_update
delete_custom_plugin_update
import_custom_plugin_update
list_custom_plugin_updates
list_patches_in_custom_plugin_update

Group Verbs

create_group
delete_group
get_group_members
get_groups
modify_group

Incident Rules Verbs

add_target_to_rule_set
delete_incident_record
remove_target_from_rule_set

Installation Verbs

get_agentimage
get_agentimage_rpm
get_supported_platforms

Internal Metrics Verbs

get_internal_metric
list_internal_metrics

Java EE Application Component Verbs

create_jeeappcom
upload_jeeappcomp_file

Job Verbs

create_job
create_job_from_library
create_library_job
delete_job
delete_library_job
describe_job
describe_job_type
describe_library_job
export_jobs
get_job_execution_detail
get_jobs
get_job_types
import_jobs
resume_job
retry_job
stop_job
submit_job
suspend_job

Licensing Verbs

grant_bipublisher_roles
grant_license_with_validation
revoke_license_no_validation
revoke_license_with_validation

Log Management Verbs

associate_cs_targets

Masking Verbs

export_masking_definition
generate_masking_script
import_masking_definition
list_masking_definitions
reassoc_masking_definition
save_masking_script
submit_masking_job

Metric Collection and Alerts Verbs

clear_stateless_alerts
collect_metric
get_metrics_for_stateless_alerts
get_on _demand_metrics
get_unsync_alerts
metric_control
sync_alerts

Metric Extension Verbs

export_metric_extension
get_unused_metric_extensions
import_metric_extension
publish_metric_extension
save_metric_extension_draft

Metric Verbs

get_threshold
modify_threshold

Monitoring Template Verbs

apply_template
export_template
import_template
list_templates
modify_collection_schedule

Notification Verbs

subscribeto_rule

OMS Configuration Properties

get_oms_config_property
get_oms_inventory
get_oms_logging_property
list_oms_config_properties
list_oms_logging_properties
list_trace
set_logging_property
set_oms_property
trace

OMS CPU Activity Report Verbs

dump_activity_list
generate_activity_report

OMS Plug-in Deployment Verbs

deploy_plugin_on_agent
deploy_plugin_on_server
get_ext_dev_kit
get_plugin_deployment_status
list_plugins_on_agent
list_plugins_on_server
redeploy_plugin_on_agent
undeploy_plugin_from_agent
undeploy_plugin_from_server

Oracle Database as Service (DBaaS) Verbs

config_db_service_target
get_db_sys_details_from_dbname
set_db_service_properties

Package Fusion Application Problem Verbs

package_fa_problem

Patch Verbs

create_patch_plan
delete_patch_plans
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Ping Subsytem Verbs

set_reverse_ping_interval

Platform as a Service (PaaS) Verbs

add_forwarders_for_paas_agent
deregister_forwarder_agents
enable_forwarder_agents
register_forwarder_agents

Pluggable Database Job Verbs

create_pluggable_database
migrate_noncdb_to_pdb

Prerequisite Check Verbs

list_prerequisites

list_prerequisites

Privilege Delegation Settings Verbs

apply_privilege_delegation_setting
clear_default_privilege_delegation_setting
clear_privilege_delegation_setting
create_privilege_delegation_setting
delete_privilege_delegation_settings
list_privilege_delegation_settings
list_target_privilege_delegation_settings
set_default_privilege_delegation_setting
test_privilege_delegation_setting

Provisioning Verbs

provision

Reconfig Job Verbs

convert_to_cluster_database

Redundancy Group Verbs

create_red_group
create_redundancy_group
modify_red_group
modify_redundancy_group
view_redundancy_group

Refresh Coherence Verbs

refresh_coherence

Refresh WLS Domain Verbs

refresh_wls

Report Import/Export Verbs

export_report
get_reports
import_report

Resource Verbs

list

Software Library Verbs

stage_swlib_entity_files

Secure Communication Verbs

get_ca_info
get_onetime_registration_token
secure_agents

Self Update Verbs

apply_update
download_update
export_update

get_update_status
import_update
import_update_catalog
remove_update
verify_updates

Services Verbs

add_beacon
apply_template_tests
assign_test_to_target
change_service_system_assoc
compare_sla
create_aggregate_service
create_service
delete_metric_promotion
delete_sla
delete_test
delete_test_threshold
disable_sla
disable_test
download_ats_test_databank_file
download_ats_test_zip
edit_sl_rule
enable_sla
enable_test
export_sla
extract_template_tests
get_aggregate_service_info
get_aggregate_service_members
get_test_thresholds
import_appreplay_workload
import_sla
list_sla
modify_aggregate_service
remove_beacon
remove_service_system_assoc
run_avail_diag
run_promoted_metric_diag
set_availability
set_key_beacons_tests
set_metric_promotion
set_properties
set_test_threshold
sync_beacon
upload_ats_test_databank_file

Server-generated Alert Metric Verbs

validate_server_generated_alerts

Siebel Verbs

list_siebel_enterprises
list_siebel_servers
update_siebel

SiteGuard Verbs

add_siteguard_aux_hosts
add_siteguard_script_credential_params
add_siteguard_script_hosts
configure_siteguard_lag
create_operation_plan
create_siteguard_configuration
create_siteguard_credential_association
create_siteguard_script
delete_operation_plan
delete_siteguard_aux_host
delete_siteguard_configuration
delete_siteguard_credential_association
delete_siteguard_lag
delete_siteguard_script
delete_siteguard_script_hosts
get_operation_plan_details
get_operation_plans
get_siteguard_aux_hosts
get_siteguard_credential_association
get_siteguard_health_checks
get_siteguard_lag
get_siteguard_script_credential_params
get_siteguard_script_hosts
get_siteguard_scripts
run_prechecks
schedule_siteguard_health_checks
stop_siteguard_health_checks
submit_operation_plan
update_operation_plan
update_siteguard_configuration
update_siteguard_credential_association
update_siteguard_lag
update_siteguard_script

Software Library Verbs

add_swlib_storage_location
create_swlib_entity
create_swlib_folder
list_swlib_entities
list_swlib_entity_subtypes
list_swlib_entity_types
list_swlib_folders
list_swlib_storage_locations
refer_swlib_entity_files
reimport_swlib_metadata
remove_swlib_storage_location
stage_swlib_entity_files
switch_swlib_oms_agent_storage
update_swlib_entity
upload_swlib_entity_files
verify_swlib

Software Maintenance Verbs

db_software_maintenance

SSA Verbs

cleanup_dbaas_requests
create_database_size
create_dbaas_quota
create_paas_zone
create_pool
create_service_template
db_cloud_maintenance
delete_database_size
delete_dbaas_quota
delete_paas_zone
delete_pool verb
delete_service_template
get_dbaas_quota
get_dbaas_request_settings
get_paas_zone_detail
get_pool_allowed_placement_constraints
get_pool_capacity
get_pool_detail
get_pool_filtered_targets
get_saved_configs
get_service_template_detail
get_service_templates
list_database_sizes
rename_service_template
update_dbaas_quota
update_dbaas_request_settings
update_paas_zone
update_pool
update_service_template

System Verbs

create_system
delete_system
get_system_members
modify_system

Target Data Verbs

add_target
add_target_property
change_target_owner
create_assoc
delete_assoc
delete_target
get_target_properties
get_targets
list_allowed_pairs
list_assoc
list_target_property_names
manage_agent_partnership
migrate_to_lifecycle_status

modify_lifecycle_stage_name
modify_target
relocate_targets
remove_target_property
rename_target
set_standby_agent
set_target_property_value

Trace Verbs

generate_ui_trace_report
trace_set_property

User-defined Metrics (UDM) Migration Verbs

abort_udmmig_session
analyze_unconverted_udms
create_udmmig_session
list_unconverted_udms
udmmig_list_matches
udmmig_request_udmdelete
udmmig_retry_deploys
udmmig_session_details
udmmig_submit_metricpicks
udmmig_summary
udmmig_update_incrules

Upgrade Database Job Verbs

upgrade_database

User Administration Verbs

create_role
create_user
delete_role
delete_user
get_supported_privileges
grant_privs
grant_roles
modify_role
modify_user
revoke_privs
revoke_roles

User Session Administration Verbs

list_active_sessions

5.2 -input_file Syntax Guidelines

5.2.1 -input_file Syntax

This option enables you to provide an argument to be specified in a file. For example:

```
emcli xyzverb -input_file="arg1:file1.txt" -input_file="arg2:file2.txt"
```

This string literally translates to:

```
emcli xyzverb -arg1=<contents of file1.txt> -arg2=<contents of file2.txt>

emcli xyzverb -input_file="name:/tmp/b1.txt"  -input_file="type:/tmp/b2.txt"
-input_file="bcnName:/tmp/b3.txt"
```

This example makes User1 an Enterprise Manager user, which is already created on an external user store like the SSO server. The contents of priv_file are view_target;host1.example.com:host. User1 will have view privileges on the host1.example.com:host target.

```
emcli create_user
      -name="User1"
      -type="EXTERNAL_USER"
      -input_file="privilege:/home/user1/priv_file"
```

5.2.2 -input_file for Jobs

For most job verbs, you can specify all of the needed properties in a property file. You can also provide a few properties on the command line. Properties set on the command line override values set in the file.

The property file consists of name=value pairs. For example, put the following into myFile.txt:

```
name=MY JOB 1
    type=OSCommand
    description=this is a test job
    target_list=target1:host
    variable.default_shell_command=ls -l
    schedule.frequency=IMMEDIATE
```

... then run:

```
emcli create_job -input_file=property_file:myFile.txt
```

This creates an OS Command job called "MY JOB 1" using preferred credentials.

Usage of Properties

For the create verbs, all properties set in the file are used. For verbs that act on multiple jobs, like suspend and resume, only "search" properties are used (name, type, targets. and scheduled starting and ending times).

Creating a Property File

The best way to create a property file is to start by describing a job similar to the one you want to create, and/or by describing a job type. This provides a list of which properties are needed by a given job type.

Determining Variables for a Job

Most properties are the same from one job to another. For example:

name, type, description, kind, targetType, cred, schedule notification

The variables needed for a job type change from job to job. Describe a job type to find out which variables it requires.

For example, the following command creates a property file template based on job MYJOB1. This lists the properties set by this job.

```
emcli describe_job [-verbose] -name=MyJob1 > myPropFile.txt
```

This example creates a property file template for an OS Command job. This lists the properties allowed by this job type, including all required and optional variables. Variables marked as deprecated should be avoided.

```
emcli describe_job_type [-verbose] -type=OSCommand > myPropFile.txt
```

5.3 Overriding the Separator and Subseparator

Not all verbs allow separator and subseparator to be overridden. The semi-colon (;) and colon (:) are respectively the default separator and subseparator. The separator is used for arguments that take multiple values, and subseparator is used when the value itself has multiple values. You can override either one of them or both.

The syntax is:

```
separator=<option_for_which_separator_has_to_be_applied>="separator_value"
```

As an example of using the separator and subseperator to create a group containing database2 and database3, the command could be:

```
emcli create_group -name="tstgrp" -add_targets="database2:oracle_database;
database3:oracle_database"
```

Using this command as the basis for modification, these examples show overrides of separator and/or subseperator:

```
emcli create_group -name="tstgrp1" -add_targets="database2:oracle_database,
database3:oracle_database" -separator=add_targets=","

emcli create_group -name="tstgrp2" -add_targets="database2&oracle_database,
database3&oracle_database" -separator=add_targets="," -subseparator=add_
targets="&"

emcli create_group -name="tstgrp3" -add_targets="database2&oracle_database;
database3&oracle_database" -subseparator=add_targets="&"
```

EM CLI Verbs

The following sections provide descriptions, formats, and options for all EM CLI verbs. Some of the verbs also contain one or more examples.

abort_udmmig_session

Aborts the migration of user-defined metrics (UDMs) to metric extensions in a session.

Format

```
emcli abort_udmmig_session
        -session_id=<sessionId>
        [-input_file=specific_tasks:<complete_path_to_file>]

[ ] indicates that the parameter is optional
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

- **input_file**

  Points at a file name that contains a target UDM, one per line in the following format:

  ```
  <targetType>,<targetName>,<collection name>
  ```

  Use targetType=Template to indicate a template. Use * for the collection name to abort all UDMs for a target. The input file should be in UTF-8 format.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

Example 1

This example aborts the specified migration session. The UDM is returned to the unconverted list.

```
emcli abort_udmmig_session -session_id=<sessionId>
```

Example 2

This example partially aborts the migration session by removing the specified UDMs from the session.

```
emcli abort_udmmig_session -session_id=<sessionId> -input_file=specific_
tasks:<complete file path>
```

add_beacon

Adds a beacon to the monitoring set of beacons. All enabled tests are pushed to the beacon.

Format

```
emcli add_beacon
      -name=target_name
      -type=target_type
      -bcnName=beacon_name
      [-dontSetKey]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **bcnName**

  Beacon name to add.

- **dontSetKey**

  Indicates the added beacon is not automatically a key beacon. Only use this option if you do not want the beacon to participate in the availability calculation of the service and tests.

Example

This example adds MyBeacon as a key beacon to the MyTarget service target of type `generic_service`.

```
emcli add_beacon -name='MyTarget' -type='generic_service'
      -bcnName='MyBeacon'
```

add_blackout_reason

Adds a new blackout reason. Only Super Administrators can perform this action.

Format

```
emcli add_blackout_reason -name="<blackout reason>"
```

Example

This example adds the blackout reason "Testing Purposes."

```
emcli add_blackout_reason -name="Testing Purposes"
```

add_chargeback_entity

Adds the given entity to Chargeback.

Format

```
add_chargeback_entity
     -entity_name="eName"
     -entity_type="eType"
     -usage_mode="uMode"
```

Parameters

- **entity_name**

  Name of the entity to be added to Chargeback.

- **entity_type**

  Type of entity to be added to Chargeback.

- **usage_mode**

  Usage mode by which it should be added to Chargeback. You can see the usage modes for a particular entity type by entering list_chargeback_entity_types -entity_type .

See Also

assign_charge_plan
assign_cost_center
list_chargeback_entities
list_chargeback_entity_types
list_charge_plans
list_cost_centers
remove_chargeback_entity
unassign_charge_plan
unassign_cost_center

add_forwarders_for_paas_agent

Adds forwarders for a given PaaS agent.

Format

```
emcli add_forwarders_for_paas_agent
      -paas_agent_name="paas_agent_name"
      -agent_list="agent_list"
```

Parameters

- paas_agent_name

  Agent name of the hybrid agent.

- agent_list

  Forwarder agent list separated by a space.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example forwards paas_agent_1 and paas_agent_2 to paas_agent:

```
emcli add_forwarders_for_paas_agent
      -paas_agent_name=<paas_agent>
      -agent_list="paas_agent_1,paas_agent_2"
```

add_siteguard_aux_hosts

Associates new auxiliary hosts with the system. An auxiliary host can be any host that is not part of the system but is managed by Enterprise Manager Cloud Control. These hosts can be used to execute any script. Any other targets running on this host will not be part of Site Guard operation plan(s).

Format

```
emcli add_siteguard_aux_hosts
        -system_name="name_of_the_system"
        -host_name="name_of_the_auxiliary_host"

[ ] indicates that the parameter is optional
```

Parameters

- **system_name**

  Name of the system.

- **host_name**

  Name of the auxiliary host that the current user needs to add. This host must be managed by Enterprise Manager Cloud Control.

Examples

Example 1

This example adds the auxiliary `host host1.domain.com` to `austin-system`:

```
emcli add_siteguard_aux_hosts
        -system_name="austin-system"
        -host_name="host1.domain.com"
```

Example 2

This example adds auxiliary hosts `host1.domain.com` and `host2.domain.com` to the system `austin-system`:

```
emcli add_siteguard_aux_hosts
        -system_name="austin-system"
        -host_name="host1.domain.com"
        -host_name="host2.domain.com"
```

Example 3

This example associates auxiliary hosts `host1.domain.com` and `host2.domain.com` that are part of `austin-system` to the system:

```
emcli add_siteguard_aux_hosts
        -system_name="austin-system"
        -host_name="host1.domain.com;host2.domain.com"
```

add_siteguard_script_credential_params

Adds a named credential as a parameter for a Site Guard script. The values of user name and password of this credential can be accessed within the script.

See Also: emcli delete_siteguard_script_credential_params, emcli get_siteguard_ script_credential_params

Format

```
emcli add_siteguard_script_credential_params
        -script_id="Id associated with the script"
        -credential_name="name of the credential"
        [-credential_owner="credential owner"]

[ ] indicates that the parameter is optional.
```

Parameters

- script_id

  The script ID.

- credential_name

  The name of the credential.

- credential_owner

  The owner of the credential. This parameter does not need to be specified if the owner of the credential is same as the logged in user.

Examples

Example 1

The following command adds a script ID and credential name to the siteguard script.

```
emcli add_siteguard_script_credential_params
      -script_id="1"
      -credential_name="NAMED_CREDENTIAL_X"
```

Example 2

The following command adds a script ID and credential name to the siteguard script where the credential owner is SG_ADMIN.

```
emcli add_siteguard_script_credential_params
      -script_id="2"
      -credential_name="NAMED_CREDENTIAL_Y"
      -credential_owner="SG_ADMIN"
```

add_siteguard_script_hosts

Adds a host to the Site Guard configuration scripts.

Format

```
emcli add_siteguard_script_hosts
      -script_id=<script_id>
      -host_name=<name1;name2;...>
```

Parameters

- **script_id**

  ID associated with the script.

- **host_name**

  Name of the host where this script will be run. You can specify more than one host name.

Examples

```
emcli add_siteguard_script_hosts
      –script_id=”10”
      –host_name ="host1.domain.com"
```

See Also

create_siteguard_script

get_siteguard_script_hosts

add_swlib_storage_location

Adds a storage location in the software library.

Format

```
emcli add_swlib_storage_location
        -name="location_name"
        -path="location_path"
        [-type="OmsShared|OmsAgent|Http|Nfs|ExtAgent"]
        [-host="hostname"]
        [-credential_set_name="setname"] | [-credential_name="name" - credential_
          owner="owner"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the storage location.

- **path**

  Path of the storage location, which can be a file system path or a URL, depending on the storage type chosen.

- **type**

  Type of storage location. The default is OmsShared.

- **host**

  Target name of the host where the path for the storage location exists. This parameter is required for storage types OmsAgent, Nfs, and ExtAgent. For the Nfs storage type, the host is not required to be a target in Enterprise Manager.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target. This is a required parameter for storage types OmsAgent and ExtAgent. The set names can be one of the following:

  – HostCredsNormal: Default unprivileged credential set

  – HostCredsPriv: Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. This parameter is required for storage types OmsAgent and ExtAgent. This parameter must be specified together with the credential_owner parameter.

- **credential_owner**

  Owner of a named credential stored in the repository. This parameter is required for storage types OmsAgent and ExtAgent. This parameter must be specified together with the credential_name parameter.

Examples

Example 1

This example adds an OMS shared file system storage location named myOMSSharedLocation for the path /u01/swlib .

```
emcli add_swlib_storage_location
        -name="myOMSSharedLocation"
        -path="/u01/swlib"
```

Example 2

This example adds an OMS Agent File system storage location named myOMSAgtLocation for the path /u01/swlib' on host 'fs1.us.acme.com. The named credential MyAcmeCreds owned by ACME_USER is used for reading/writing files from this location.

```
emcli add_swlib_storage_location
        -name="myOMSAgtLocation"
        -path="/u01/swlib"
        -type="OmsAgent"
        -host="fs1.us.acme.com"
        -credential_name="MyAcmeCreds"
        -credential_owner="ACME_USER"
```

delete_siteguard_aux_host

Deletes an auxiliary host associated with the system.

Format

```
emcli delete_siteguard_aux_host
        -system_name="name_of_the_system"
        [-host_name="name_of_the_auxiliary_host"]

[ ] indicates that the parameter is optional
```

Parameters

- **system_name**

  Name of the system whose auxiliary host you want to delete.

- **host_name**

  Name of the auxiliary host that you want to delete. If not specified, all auxiliary hosts associated with the system will be deleted.

Examples

Example 1

This example deletes all of the auxiliary hosts associated with `austin-system`:

```
emcli delete_siteguard_aux_host
        -system_name="austin-system"
```

Example 2

This example deletes the auxiliary host `host1.domain.com` associated with `austin-system`:

```
emcli delete_siteguard_aux_host
        -system_name="austin-system"
        -host_name="host1.domain.com"
```

Example 3

This example deletes the auxiliary host `host2.domain.com` associated with `austin-system`:

```
emcli delete_siteguard_aux_host
        -system_name="austin-system"
        -host_name="host2.domain.com"
```

add_target

Adds a target to be monitored by Enterprise Manager. The target type specified is checked on the Management Agent for existence and for required properties, such as user name and password for host target types, or log-in credentials for database target types. You must specify any required properties of a target type when adding a new target of this type.

For `oracle_database` target types, you must specify Role with the monitoring credentials. If the Role is Normal, the UserName must be `dbsnmp`. Otherwise, the Role must be SYSDBA, and UserName can be any user with SYSDBA privileges.

> **Note:** You cannot use this verb for composite targets. The verb does not support adding an association between a parent target such as IAS and a child target such as OC4J.

Command-Line Format

```
emcli add_target
      -name="name"
      -type="type"
      -host="hostname"
      [-properties="pname1:pval1;pname2:pval2;..."]
      [-separator=properties="sep_string"]
      [-subseparator=properties="subsep_string"]
      [-credentials="userpropname:username;pwdpropname:password;..."]
      [-input_file="parameter_tag:file_path"]
      [-display_name="display_name"]
      [-groups="groupname1:grouptype1;groupname2:grouptype2;..."]
      [-timezone_region="gmt_offset"]
      [-monitor_mode="monitor_mode"]
      [-instances="rac_database_instance_target_name1:target_type1;..."]
      [-force=true|false]
      [-timeout="time_in_seconds"]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
add_target
      (name="name"
      ,type="type"
      ,host="hostname"
      [,properties="pname1:pval1;pname2:pval2;..."]
      [,separator=properties="sep_string"]
      [,subseparator=properties="subsep_string"]
      [,credentials="userpropname:username;pwdpropname:password;..."]
      [,input_file="parameter_tag:file_path"]
      [,display_name="display_name"]
      [,groups="groupname1:grouptype1;groupname2:grouptype2;..."]
      [,timezone_region="gmt_offset"]
      [,monitor_mode="monitor_mode"]
      [,instances="rac_database_instance_target_name1:target_type1;..."]
      [,force=true|false]
      [,timeout="time_in_seconds"])

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks.

- **type**

  Target type. Standard target types include: `host`, `oracle_database`, `oracle_apache`, `oracle_listener`, and `oracle_emd`. To see all available target types available for your environment, check the `$AGENT_HOME/sysman/admin/metadata` directory. A metadata file (XML) exists for each target type.

- **host**

  Network name of the system running the Management Agent that is collecting data for this target instance.

- **properties**

  Name-value pair (that is, `prop_name:prop_value`) list of properties for the target instance. The "name"(s) are identified in the target-type metadata definition. They must appear exactly as they are defined in this file. Metadata files are located in `$AGENT_HOME/sysman/admin/metadata`.

  Note: This verb does not support setting global target properties. It is recommended that you use set_target_property_values to set target properties.

- **separator=properties**

  Specify a string delimiter to use between name-value pairs for the value of the -properties . The default separator delimiter is ";".

  For more information about the separator parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **subseparator=properties**

  Specifies a string delimiter to use between the name and value in each name-value pair for the value of the -properties option. The default subseparator delimiter is ":".

  For more information about the subsepator parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **credentials**

  Monitoring credentials (name-value pairs) for the target instance. The "name"(s) are identified in the target-type metadata definition as credential properties. The credentials must be specified exactly as they are defined in the target's metadata file. Metadata files are located in `$AGENT_HOME/sysman/admin/metadata`.

- **input_file**

  Used in conjunction with the `-credentials` option, this enables you to store specific target monitoring credential values, such as passwords, in a separate file. The `-input_file` specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific monitoring credentials of the `-credentials` option. The tag must not contain colons (:) or semi-colons (;).

For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **display_name**

  Target name displayed in the Enterprise Manager Cloud Control console.

- **groups**

  Name-value pair list of the groups to which this target instance belongs. Follows the format of `groupname:grouptype;groupname2:grouptype2`.

- **timezone_region**

  GMT offset for this target instance. (-7 or -04:00 are acceptable formats.)

- **monitor_mode**

  Either 0, 1, or 2. The default is 0. 1 specifies OMS mediated monitoring, and 2 specifies Agent mediated monitoring.

- **instances**

  Name-value pair list of RAC database instances that the RAC database target has. Database instance targets must be added before trying to add the cluster database.

- **force**

  Forces the target to be added even if the target with the same name exists. Updates the properties of the target with your latest input.

- **timeout**

  Time in seconds for the command to wait to add the target to the Agent. The default is 10 minutes.

Examples

The following two examples add an `oracle_database` target with the name "database." Note how the credentials are specified. The "name"(s) in the name-value pairs come from the `oracle_database` metadata file. They must appear exactly as they are named in that file. This also applies for the property "name"(s). The examples use the base minimum of required credentials and properties for the database target.

Example 1 - Command-Line

```
emcli add_target
      -name="database"
      -type="oracle_database"
      -host="myhost.us.example.com"
      -credentials="UserName:dbsnmp;password:dbsnmp;Role:Normal"
      -properties="SID:semcli;Port:15091;OracleHome:/oracle;
       MachineName:smpamp-example.com"
      -groups="Group1:group;Group2:group"
```

Example 2 - Scripting and Interactive

```
add_target
      (name="database"
      ,type="oracle_database"
      ,host="myhost.us.example.com"
      ,credentials="UserName:dbsnmp;password:dbsnmp;Role:Normal"
      ,properties="SID:semcli;Port:15091;OracleHome:/oracle;
        MachineName:smpamp-example.com"
      ,groups="Group1:group;Group2:group")
```

The following two examples add an oracle_database target with the name "database." The examples illustrate the use of the input_file to camouflage the credentials. The password is actually in a file named at_pwd_file. The input_file argument is used to replace PWD_FILE with the contents of the at_pwd_file in the credentials argument.

Example 3 - Command-Line

```
emcli add_target
      -name="database"
      -type="oracle_database"
      -host="myhost.us.example.com"
      -credentials="UserName:dbsnmp;password:PWD_FILE;Role:Normal"
      -properties="SID:semcli;Port:15091;OracleHome:/oracle;
       MachineName:smpamp-example.com"
      -input_file="PWD_FILE:/emcli_dir/pwdfiles/at_pwd_file"
```

Example 4 - Scripting and Interactive

```
add_target
      (name="database"
      ,type="oracle_database"
      ,host="myhost.us.example.com"
      ,credentials="UserName:dbsnmp;password:PWD_FILE;Role:Normal"
      ,properties="SID:semcli;Port:15091;OracleHome:/oracle;
        MachineName:smpamp-example.com"
      ,input_file="PWD_FILE:/emcli_dir/pwdfiles/at_pwd_file")
```

The following two examples illustrate how to add a RAC database with given installed RAC database instances and clusterware. The examples add a rac_database target with the name cluster_database and the cluster name **newdb_cluster**. A RAC instance is picked up among instances on the given host. This verb should be called after database instances and clusterwares have been installed. monitor_mode is set to 1, because a RAC database is a multi-agent target.

Example 4 - Command-Line

```
emcli add_target
      -name="cluster_database"
      -type="rac_database"
      -host="myhost.us.example.com"
      -monitor_mode="1"
      -properties="ServiceName:service.example.com;ClusterName:
       newdb_cluster"
      -instances="database_inst1:oracle_database;database_inst2:
       oracle_database"
```

Example 5 - Scripting and Interactive

```
emcli add_target
      (name="cluster_database"
      ,type="rac_database"
      ,host="myhost.us.example.com"
      ,monitor_mode="1"
      ,properties="ServiceName:service.example.com;ClusterName:
        newdb_cluster"
      ,instances="database_inst1:oracle_database;database_inst2:
        oracle_database")
```

The following two examples add an `oracle_listener` target with the name mylist. The `LsnrName` is the name of the listener as configured in the `listener.ora` file, and `ListenerOraDir` is the directory containing the `listener.ora` file.

Example 6 - Command-Line

```
emcli add_target
      -name="mylist"
      -type="oracle_listener"
      -host="myhost.example.com"
      -properties="LsnrName:LISTENER;ListenerOraDir:/oracle/lsnr;
       Port:15091;OracleHome:/oracle;Machine:smpamp-sun1.us
```

Example 7 - Scripting and Interactive

```
add_target
      (name="mylist"
      ,type="oracle_listener"
      ,host="myhost.example.com"
      ,properties="LsnrName:LISTENER;ListenerOraDir:/oracle/lsnr;
        Port:15091;OracleHome:/oracle;Machine:smpamp-sun1.us)
```

add_target_property

Adds a new target property for a given target type. All targets of this target type will have this new target property.

Format

```
emcli add_target_property
         -target_type="target_type"
         -property="prop_name"
```

Parameters

- **target_type**

  Target type for which this property needs to be added. To add this property to all existing target types, you can specify a "*" wildcard character.

- **property**

  Name of the property to be created for this target type. Property names are case-sensitive. The property name cannot be the same as the following Oracle-provided target property names (in English):

  Comment, Deployment Type, Line of Business, Location, Contact

Examples

Example 1

This example adds the Owner Name property for all targets of type oracle_database.

```
emcli add_target_property -target_type="oracle_database" -property="Owner Name"
```

Example 2

This example adds the Owner property for all target types.

```
emcli add_target_property -target_type="*" -property="Owner"
```

add_target_to_rule_set

Adds a target to an enterprise rule set.

Privilege Requirements: A Super Administrator can add a target to any enterprise rule set except for predefined (out-of-box) rule sets supplied by Oracle.

Only the owner or co-author of a rule set can add a target to it.

Format

```
emcli add_target_to_rule_set
      -rule_set_name="rule set name"
      -target_name="target name"
      -target_type="internal name for the target type. For example, host"
      [-rule_set_owner=<ruleset owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **rule_set_name**

  Name of an enterprise rule set. This option only applies to rule sets that are associated with a list of targets.

- **target_name**

  Name of the target to be added.

- **target_type**

  Target type of the target to be added. For example, host.

- **rule_set_owner**

  Optionally, you can specify the owner of the rule set.

Examples

Example 1

This example adds the host target *myhost.com* to a rule set named *rules*. This rule set is owned by the administrator *sysman*.

```
emcli add_target_to_rule_set -rule_set_name='rules' -target_name='myhost.com'
-target_type='host' -rule_set_owner='sysman'
```

add_virtual_platform

Adds Oracle Virtual Platform(s) to remotely monitor Xen-based Hypervisor(s). The associated Oracle Server and Oracle Virtual Server running on the Hypervisor will is also added.

You can add multiple Hypervisors at the same time. The command returns the name and the execution identifier of the job submitted to add the target(s).

To delete an Oracle Virtual Platform and its related targets, use the delete_target verb.

Format

```
emcli add_virtual_platform
       -name="host_name/IP_address_or_list_from_an_input_file"
       -agent="agent_target_name"
       [-failover_agent="failover_agent_target_name"]
       -credentials="property_name1:property_value1;property_name2:
           property_value2;..."
       [-wait_for_completion=true|false]
       [-wait_for_completion_timeout=<time_in_minutes>]
       [-separator=credentials="separator_for_key_value_pairs"]
       [-subseparator=credentials="separator_for_key_value_pair"]
       [-input_file="FILE:file_path_or_name:FILE"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  IP address or host name of the Xen-based Hypervisor being added as an Oracle Virtual Platform in Enterprise Manager. There are two ways to provide this value. For only one target, you can directly pass this value at the command line with the name of the Host or the IP address. For multiple values, you can use the "-input_ file" parameter and list the host names, IP addresses, or an IP address range from a file by passing the name of the input file. A new line is used to delimit the host names or the IP addresses. You can specify the host name of a machine, an IP address, or an IP address range on each line.

  See the examples for details.

- **agent**

  Target name of the primary agent used to monitor the Oracle Virtual Platform(s) and related targets.

- **failover_agent**

  Target name of the failover agent used to monitor the Oracle Virtual Platform(s) and related targets.

- **credentials**

  Monitoring credentials (name-value pairs) for the target instance. The "names" are defined in the target type metadata definition as credential properties. Metadata files are located at $AGENT_HOME/sysman/admin/metadata.

  See the examples for details on various options.

- **wait_for_completion**

Flag to indicate if the CLI is going to wait for the submitted job to finish. The default value is false. If the value is true, the progress of the job is printed on the command line as and when the addition of Oracle Virtual Platform(s) Succeeds/Fails.

- **wait_for_completion_timeout**

  Time in minutes after which CLI stops waiting for the job to finish. This parameter is honored only if the value for parameter wait_for_completion is true. A negative or zero value does not wait for the job to finish.

  See the examples for details.

- separator=credentials

  Custom separator for the credential key value pairs. Specify a string delimiter to use between name-value pairs for the values of the -credentials option. The default separator delimiter is ";".

  For more information about the separator parameter, see Section 5.2, "-input_file Syntax Guidelines".

- subseparator=credentials

  Custom separator for a key value pair. Specify a string delimiter to use between name and value in each name-value pair for the values of the -credentials option. The default separator delimiter is ":".

  For more information about the subseparator parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file**

  Optionally use in conjunction with the -credentials option. You can use this option to set specific target monitoring credential values, such as passwords or SSH keys, in a separate file.

  This option specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific -credentials property values.

Examples

Example 1

This example adds an Oracle Virtual Platform with root user host credentials. The value of the property "OVSUsername" is used for the user name, and "OVSPassword" for the password. The value of the property "privilegedUser" indicates if the virtualization-specific metrics are collected(true) or not(false) when monitoring. The password is passed at the command line.

```
emcli add_virtual_platform
      -name=example.com
      -agent=example.com:1838
      -credentials='type:DMOvsBasicCreds;PrivilegeType:none;privilegedUser:true;
          OVSUsername:root;OVSPassword:mypassword'
```

Example 2

This example adds an Oracle Virtual Platform with root user host credentials. The value of the property "OVSUsername" is used for the user name, and "OVSPassword" for the password. The value of the property "privilegedUser" indicates if the virtualization-specific metrics are not collected(true) or not(false) when monitoring. The password of the root user is read from the input file "password.txt".

```
emcli add_virtual_platform
      -name=example.com
      -agent=example.com:1838
      -credentials='type:DMOvsBasicCreds;PrivilegeType:none;privilegedUser:true;
          OVSUsername:root;OVSPassword:PWD_FILE'
      -input_file='PWD_FILE:password.txt'
```

Example 3

This example adds multiple Oracle virtual platforms with root user host credentials. You can specify multiple host names, IP addresses, or an IP address range in the host names list file delimited by a new line.

NOTE: In case of multiple target additions, the same credentials are used across all Hypervisors being added.

Host names list file example:

```
example1.com
192.168.1.0
10.172.10.2-254
```

```
emcli add_virtual_platform
      -input_file='name:hostnames.txt'
      -agent=example.com:1838
      -credentials='type:DMOvsBasicCreds;PrivilegeType:none;privilegedUser:true;
           OVSUsername:root;OVSPassword:mypassword'
```

Example 4

This example adds an Oracle Virtual Platform with Unix Sudo user host credentials. The value of the property "PrivilegeCommand" is used to execute the Sudo command. %RUN_AS% and %COMMAND% are replaced with the user and the command to be executed by the Sudo command. The value of the property "EnablePseudoTerminal" indicates if Sudo requires (true) a tty terminal or not (false). The password is passed at the command line.

```
emcli add_virtual_platform
      -name=example.com
      -agent=example.com:1838
      -credentials='type:DMOvsBasicCreds;PrivilegeType:sudo;privilegedUser:true;
          RunAs:root;PrivilegeCommand:/usr/bin/sudo -S -u %RUN_AS%%COMMAND%;
          EnablePseudoTerminal:false;OVSUsername:root;OVSPassword:mypassword'
```

Example 5

This example adds an Oracle Virtual Platform with Unix PowerBroker user host credentials. The value of the property "PrivilegeCommand" is used to execute the PowerBroker command. %RUN_AS% and %COMMAND% are replaced by you and the command to be executed by PowerBroker. The value of the property "PowerBrokerProfile" is used as the PowerBroker profile. The value of the property "PowerBrokerPasswordPrompt" is used as the PowerBroker password prompt. The password is passed at the command line.

```
emcli add_virtual_platform
      -name=example.com
      -agent=example.com:1838
      -credentials='type:DMOvsBasicCreds;PrivilegeType:powerbroker;RunAs:root;
           PrivilegeCommand:/usr/bin/pbrun -l -u %RUN_AS%
```

```
          %COMMAND%;PowerBrokerProfile:profile;
           PowerBrokerPasswordPrompt:myprompt;
           privilegedUser:true;OVSUsername:root;OVSPassword:mypassword'
```

Example 6

This example adds an Oracle Virtual Platform with a Unix Sudo user who requires SSH key Passphraseless-based authentication. The SSH private key, SSH public key, and password are read from input files.

```
emcli add_virtual_platform
      -name=example.com
      -agent=example.com:1838
      -credentials='type:DMOvsSshKeyCreds;PrivilegeType:sudo;privilegedUser:true;
            RunAs:root;PrivilegeCommand:/usr/bin/sudo -S -u %RUN_AS% %COMMAND%;
            EnablePseudoTerminal:false;SshPrivateKey:PRIVATE_KEY;
            SshPublicKey:PUBLIC_KEY;OVSUsername:sudoer1;OVSPassword:PWD_FILE'
      -input_file='PRIVATE_KEY:id_dsa'
      -input_file='PUBLIC_KEY:id_dsa.pub'
      -input_file='PWD_FILE:password'
```

Example 7

This example adds an Oracle Virtual Platform with a Unix Sudo user who requires SSH key Passphrase-based authentication. The SSH private key, SSH public key, and password are read from input files.

```
emcli add_virtual_platform
      -name=example.com -agent=example.com:1838
      -credentials='type:DMOvsSshKeyCreds;PrivilegeType:sudo;privilegedUser:true;
           RunAs:root;PrivilegeCommand:/usr/bin/sudo -S -u %RUN_AS% %COMMAND%;
           EnablePseudoTerminal:false;PassPhrase:welcome1;
           SshPrivateKey:PRIVATE_KEY;SshPublicKey:PUBLIC_KEY;OVSUsername:sudoer1;
           OVSPassword:PWD_FILE'
      -input_file='PRIVATE_KEY:id_dsa'
      -input_file='PUBLIC_KEY:id_dsa.pub'
      -input_file='PWD_FILE:password'
```

Example 8

This example adds an Oracle Virtual Platform with a Unix PowerBroker user who requires SSH key Passphraseless based authentication. The SSH private key, SSH public key, and password are read from input files.

```
emcli add_virtual_platform
      -name=example.com
      -agent=example.com:1838
      -credentials='type:DMOvsSshKeyCreds;PrivilegeType:powerbroker;
           privilegedUser:true;RunAs:root;PrivilegeCommand:
           /usr/bin/pbrun -l -u %RUN_AS% %COMMAND%;
           PowerBrokerProfile:profile;PowerBrokerPasswordPrompt:myprompt;
           SshPrivateKey:PRIVATE_KEY;SshPublicKey:PUBLIC_KEY;
           OVSUsername:myuser;OVSPassword:PWD_FILE'
      -input_file='PRIVATE_KEY:id_dsa'
      -input_file='PUBLIC_KEY:id_dsa.pub'
      -input_file='PWD_FILE:password'
```

Example 9

This example adds an Oracle Virtual Platform with non-privileged user host credentials. The virtualization metrics for the added target will not be monitored. The password is specified at the prompt.

```
emcli add_virtual_platform
      -name=example.com
      -agent=example.com:1838
      -credentials='type:DMOvsBasicCreds;privilegedUser:false;
           OVSUsername:simpleton;OVSPassword:password'
```

analyze_unconverted_udms

Analyzes UDMs and lists unique UDMs, any possible matches, and templates that can apply these matching metric extensions.

Format

```
emcli analyze_unconverted_udms
      [-session_id=<sessionId>]

[ ] indicates that the parameter is optional
```

Parameters

- **session_id**

  ID of a session to be analyzed. Not specifying a session ID creates an analysis session that contains all unconverted UDMs. You can specify this session ID in future invocations to generate a fresh analysis.

Examples

Example 1

This example lists matches for all unconverted UDMs in existing metric extensions.

```
emcli analyze_unconverted_udms
```

Example 2

This example lists matches for all unconverted UDMs in the specified migration session.

```
emcli list_unconverted_udms -session_id=<sessionId>
```

apply_diagcheck_exclude

Applies a diagnostic check exclusion to a set of target instances. You can exclude certain diagnostic checks by defining an exclusion name. This rule is applied when all diagnostic checks are evaluated for the particular target type so that the checks specified in the rule are excluded.

Format

```
emcli apply_diagcheck_exclude
        -target_type="type"
        -exclude_name="name"
        [-target_name="target_name" ]*

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target.

- **exclude_name**

  Name to use for the exclusion. To create the exclude_name, use the define_diagcheck_exclude verb.

- **target_name**

  Target names to apply the exclusion to.

apply_privilege_delegation_setting

Activates Sudo or PowerBroker settings for specified targets.

Command-Line Format

```
emcli apply_privilege_delegation_setting
         -setting_name="setting"
         -target_type="host/composite"
         [-target_names="name1;name2;..."]
         [-input_file="FILE:file_path"]
         [-force="yes/no"]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
apply_privilege_delegation_setting
         (setting_name="setting"
         ,target_type="host/composite"
         [,target_names="name1;name2;..."]
         [,input_file="FILE:file_path"]
         [,force="yes/no"])

[ ] indicates that the parameter is optional
```

Parameters

- **setting_name**

  Name of the setting you want to apply.

- **target_names**

  List of target names. The newly submitted setting applies to this list of Enterprise Manager targets.

  – All targets must be of the same type.

  – The target list must not contain more than one element if the element's target type is "group."

  – The group referenced above should have at least one host target.

- **target_type**

  Type of targets to which the setting is applied. Valid target types are "host" or "composite" (group).

- **input_file**

  Path of the file that has target names. This enables you to pass targets in a separate file. The file cannot contain any colons (:) or semi-colons (;).

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **force**

  If `yes`, the operation continues and ignores any invalid targets. The default is `no`.

Examples

These examples apply a privilege setting named `sudo_setting`. This setting applies to targets of type `host`, and it is being applied to `host1`, `host2`, and so forth.

Example 1 - Command-Line

```
emcli apply_privilege_delegation_setting
      -setting_name=sudo_setting
      -target_type=host
      -target_names="host1;host2;"
```

Example 2 - Scripting and Interactive

```
apply_privilege_delegation_setting
      (setting_name="sudo_setting"
      ,target_type="host"
      ,target_names="host1;host2")
```

These examples apply a privilege setting named `sudo_setting`. This setting applies to targets of type `host`, and it is being applied to `host1`, `host2`, and so forth. The `force` flag indicates that the setting is applied to all valid targets, and invalid targets are ignored.

Example 3 - Command-Line

```
emcli apply_privilege_delegation_setting
      -setting_name=sudo_setting
      -target_type=host
      -target_names="host1;host2;"
      -force=yes
```

Example 4 - Scripting and Interactive

```
apply_privilege_delegation_setting
      (setting_name="sudo_setting",
       target_type="host",
       target_names="host1;host2", force="yes")
```

These examples apply a privilege setting named `sudo_setting`. This setting applies to targets of type `host`, and host names are selected from `/home/jdoe/file.txt` (one host per line). The `force` flag indicates that the setting is applied to all valid targets, and invalid targets are ignored.

Example 5 - Command-Line

```
emcli apply_privilege_delegation_setting
      -setting_name=sudo_setting
      -target_type=host
      -input_file="FILE:/home/jdoe/file.txt"
      -force=yes
```

Example 6 - Scripting and Interactive

```
apply_privilege_delegation_setting
      (setting_name="sudo_setting"
      ,target_type="host"
      ,input_file="FILE:/home/jdoe/file.txt"
      ,force="yes")
```

apply_template

Applies a monitoring template to a list of specified targets. The parameters to the verb can be supplied in any order.

Format

```
emcli apply_template
        -name="template_name"
        -targets="tname1: ttype1;tname2: ttype2;..."
        [-copy_flags="0" or "1" or "2"]
        [-replace_metrics="0" or "1"]
        [-input_file="FILE1:file_name"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Template name as it exists in the database. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks.

- **targets**

  The targets should be specified in the following sequence:

  TargetName1:TargetType1;TargetName2:TargetType2

  For example:

  ```
  db1:oracle_database;my db group:composite
  ```

  A semi-colon is the target separator. Ideally, non-composite targets should be of the target type applicable to the template. If not, the template is not applied to the indicated target. For composite targets, the template is applied only to the member targets that belong to the target type for which the template is applicable.

- **copy_flags**

  This applies only for metrics with multiple thresholds.

  '0' indicates: Apply threshold settings for key values common to the template and target.

  '1' indicates: Remove key value threshold settings in the target and replace them with key value threshold settings from the template.

  '2' indicates: Apply threshold settings for all key values defined in the template. The default is '0'.

- **replace_metrics**

  0 indicates that the thresholds of the metrics not included in the template but available in the target will not be changed. This is the default value. 1 indicates that the thresholds of the metrics present in the target, but not in the template, will be set to NULL. That is, such metrics in the target will not be monitored and therefore, no alert will be raised for them.

- **input_file**

  You can use this parameter to specify the location of a file, which contains the credentials to be used for the User Defined Metrics (UDMs) if the template contains any UDMs. file_name actually refers to the name of the file along with the

path of the location, which contains the credentials applicable for the UDMs. For example:

```
emcli apply_template -name="template1"  -targets="mydb1:oracle_database"
  -input_file= "FILE1:/usr/template/apply_udm_credentials.txt"
```

This example applies a monitoring template named "template1" to target mydb1 of type oracle_database, and the credentials needed for the UDMs are accessed from the file "/usr/template/apply_udm_credentials.txt".

The contents of the file apply_udm_credentials.txt should be in one of the following formats:

– All UDMs use the same credentials for all targets. For example:

```
credListType:all;
usr_name:joe1;passwrd:pass1;
```

– Each UDM uses its own credentials for all targets. For example:

```
credListType:perUDM;
udm_name:UDM1;usr_name:joe1;passwrd:pass1;
udm_name:UDM2;usr_name:joe2;passwrd:pass2;
```

– Each UDM uses different credentials for different targets. For example:

```
credListType:perTargetperUDM;
udm_name:UDM1;tgt_name:TNAME1;usr_name:joe1;passwrd:pass1;
udm_name:UDM1;tgt_name:TNAME2;usr_name:joe2;passwrd:pass2;
udm_name:UDM2;tgt_name:TNAME1;usr_name:joe3;passwrd:pass3;
udm_name:UDM2;tgt_name:TNAME2;usr_name:joe4;passwrd:pass4;
```

It is important to specify the "credListType" in every input text file that you specify.

For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

Example 1

This example applies a monitoring template named `my_db_template`. This template applies to targets of type `oracle_database`, and it is being applied to `db1` , which is of type `oracle_database`, and `my_db_group`, which is of type `composite`.

For composite targets, the template is only applied to member targets that belong to the target type for which the template is applicable. Since the `copy_flags` is not specified, the default ("Apply threshold settings for monitored objects common to both template and target") is meant.

```
emcli apply_template -name="my_db_template"
          -targets="db1:oracle_database;my_db_group:composite"
```

Example 2

This example applies a monitoring template named `my_db_template`. This template applies to targets of type `oracle_database` and it is being applied to `db1`, which is of type `oracle_database` and `my_db_group`, which is of type `composite`.

For composite targets, the template is applied only to member targets that belong to the target type for which the template is applicable. In this case, since the copy_flags option is specified as 1, the threshold settings on the target will be duplicated.

```
emcli apply_template -name="my_db_template"
          -targets="db1:oracle_database;my_db_group:composite"
          -copy_flags="1"
```

Example 3

This example applies a monitoring template named my_db_template. This template is applicable to targets of type oracle_database, and it is being applied to db1 of type oracle_database and my_db_group of type composite.

For composite targets, the template is applied only to the member targets that belong to the target type for which the template is applicable. In this case, since the copy_flags option is specified as "2", the threshold settings on the target are duplicated, but the keys present only in the target and not present in the template are retained in the target, and their settings are not affected.

```
emcli apply_template -name="my_db_template"
          -targets="db1:oracle_database;my_db_group:composite"
          -copy_flags="2"
```

Example 4

This example applies a monitoring template named my_db_template. This template applies to targets of type oracle_database and it is being applied to db1, which is of type oracle_database and my_db_group, which is of type composite.

For composite targets, the template is applied only to member targets that belong to the target type for which the template is applicable. In this case, since the copy_flags option is specified as "1", the threshold settings on the target will be duplicated. Furthermore, the credentials needed for the UDMs are present in the file /usr/vmotamar/db_credentials.txt.

```
emcli apply_template -name="my_db_template"
          -targets="db1:oracle_database;my_db_group:composite"
          -copy_flags="1" -input_file= "FILE1:/usr/vmotamar/db_credentials.txt"
```

Example 5

This example applies the monitoring template named my_db_template. This template is applicable to targets of type oracle_database. This command applies this template to two targets: target db1 of type oracle_database and target my_db_group of type composite.

For composite targets, the template is applied only to the member targets that belong to the target type for which the template is applicable. In this case, since the copy_flags option is specified as "1", the template is superimposed on the target. All keys in the template are copied to the target, and any extra keys present in the target are deleted. The credentials needed for the UDMs are present in file /usr/user/db_credentials.txt.

The replace_metrics flag set to 1 denotes that the thresholds of the metrics present in the target, but not in the template, are set to NULL. That is, these metrics in the target are not monitored, and therefore, no alert is raised for them.

```
emcli apply_template -name="my_db_template"
          -targets="db1:oracle_database;my_db_group:composite"
          -copy_flags="1" -replace_metrics="1" -input_file=
            "FILE1:/usr/user/db_credentials.txt"
```

apply_template_tests

Applies the variables and test definitions from the file(s) into a repository target.

Format

```
emcli apply_template_tests
      -targetName=target_name
      -targetType=target_type
      -input_file=template:template_filename
      [-input_file=variables:<variable_filename>]
      [-input_file=atsBundleZip:<ats_bundle_zip_filename>]
      [-useBundleDatabankFile]
      [-useFirstRowValues]
      [-overwriteExisting=all | none | <test1>:<type1>;<test2>:<type2>;...]
      [-encryption_key=key]
      [-swlibURN=<URN_for_swlib_entity>]
      [-swlibPath=<Path_for_swlib_entity>]

[ ] indicates that the parameter is optional
```

Parameters

- **targetName**

  Target name.

- **targetType**

  Target type.

- **input_file=template**

  Name of the input file containing the test definitions.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=variables**

  Name of the input file containing the variable definitions. If this attribute is not specified, the variables are extracted from the same file containing the test definitions.

  The variables file format is as follows:

  ```
  <variables xmlns="template">
  <variable name="<name1>" value="<value1>"/>
  <variable name="<name2>" value="<value2>"/>
  ...
  </variables>
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=atsBundleZip**

  Name of the ATS bundle zip defined in the template.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **useBundleDatabankFile**

If you specify this parameter, the bundle databank files are used.

- **useFirstRowValues**

  If you specify this parameter, the first row values are used.

- **overwriteExisting**

  Specifies which tests should be overwritten in case they already exist on the target. The possible values are:

  1. 'none' (default): None of the existing tests on the target will be overwritten.
  2. 'all': If a test with the same name exists on the target, it will be overwritten with the test definition specified in the template file.
  3. <test1>:<type1>;<test2>:<type2>;...: If any of the tests with names <test1>, <test2>, and so forth exist on the target, they are overwritten with the definition in the template file.

- **encryption_key**

  Optional key to decrypt the file contents. This key should be the same as the one used to encrypt the file.

- **swlibURN**

  Loads the software library entity through an URN. The respective entity data such as OATZ zip file and Zip File Name will be associated to the new service test. Either this parameter or the -swlibPath parameter are required to associate the OATS zip file to the service test.

- **swlibPath**

  Loads the software library entity through an entity path. The respective entity data such as OATZ zip file and Zip File Name will be associated to the new service test. Either this parameter or the -swlibURN parameter are required to associate the OATS zip file to the service test.

Examples

You must have the following privileges to perform these examples:

- Operator privilege on the target.
- Operator privilege on all beacons currently monitoring the target. Alternatively, you must have the "use any beacon" privilege.

Example 1

This example applies the test definitions contained in the file `my_template.xml` into the Generic Service target `my_target`, using the key `my_password` to decrypt the file contents. If tests with names `my_website` or `my_script` exist on the target, they are overwritten by the test definitions in the file.

```
emcli apply_template_tests
      -targetName='my_target' -targetType='generic_service'
      -input_file=template:'my_template.xml' -encryption_key='my_password'
      -overwriteExisting='my_website:HTTP;my_script:OS'
```

Example 2

This example applies the test definitions contained in file my_template.xml into the Web Application target my_target using the variable values

specified in file my_variables.xml. If any tests in the target have the same name as tests specified in the template file, they are overwritten.

```
emcli apply_template_tests
          -targetName='my_target' -targetType='website'
          -input_file=template:'my_template.xml' -input_file=variables:
            'my_variables.xml'
          -overwriteExisting='all'
```

apply_update

Applies an update.

Format

```
emcli apply_update
        -id="internal id"
```

Parameters

- **id**

  Internal identification for the update to be applied.

Examples

This example submits a job to apply an update, and prints the job execution ID upon submission.

```
emcli apply_update
        -id="914E3E0F9DB98DECE040E80A2C5233EB"
```

argfile

Executes one or more EM CLI verbs, where both verbs and the associated arguments are contained in an ASCII file. argfile enables you to use verbs with greater flexibility. For example, when specifying a large list of targets to be blacked out (`create_blackout` verb), you can use the `argfile` verb to input the target list from a file.

Multiple EM CLI verb invocations are permitted in this file. You should separate each verb invocation with a new line.

Format

```
emcli argfile <file_name>
      [-delim=<delimiter_string>]

[ ] indicates that the parameter is optional
```

Parameters

- **delim**

  String used as a delimiter between two verbs in the argument file. The default delimiter is a newline character.

assign_charge_plan

Assigns a charge plan to the given entity.

Format

```
assign_charge_plan
      -entity_name="eName"
      -entity_type="eType"
      -plan_name="pName"
      -[entity_guid="entity_guid"]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_name**

  Name of the entity for which the charge plan is to be assigned.

- **entity_type**

  Type of entity for which the charge plan is to be assigned.

- **plan_name**

  Name of the charge plan to be assigned.

- **entity_guid**

  guid of the entity to be added to Chargeback.

  When more than one entity is active in Chargeback with the given entity name and entity type, the command lists all such entities with additional details such as creation date, parent entity name, entity guid, and so forth to choose the correct entity. Select the correct entity from the given list and execute the command again with entity guid as the parameter instead of entity name and entity type.

Example

This example assigns charge plan "plan1" to "db1", an oracle_database entity.

```
emcli assign_charge_plan -entity_name="db1" -entity_type="oracle_database" -plan_
name="plan1"
```

assign_cost_center

Assigns the cost center to the given entity.

Format

```
assign_cost_center
     -entity_name="eName"
     -entity_type="eType"
     -cost_center_name="cName"
     -[entity_guid="entity guid" ]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_name**

  Name of the entity for which the cost center is to be assigned.

- **entity_type**

  Type of entity for which the cost center is to be assigned.

- **cost_center**

  Name of the cost center to be assigned.

- **entity_guid**

  guid of the entity in Chargeback.

  When more than one entity is active in Chargeback with the given entity name and entity type, the command lists all such entities with additional details such as creation date, parent entity name, entity guid, and so forth to choose the correct entity. Select the correct entity from the given list and execute the command again with entity guid as the parameter instead of entity name and entity type.

Example

This example assigns the cost center "cc1" to "db1", an oracle_database entity.

```
emcli assign_cost_center -entity_name="db1" -entity_type="oracle_database" -cost_
center_name="cc1"
```

assign_csi_at_target_level

Assigns or updates the Customer Support IDentifier (CSI) to the given target name and type.

Format

```
emcli assign_csi_at_target_level
      -target_name="Target_name"
      -target_type="Target_type"
      -csi="Customer_Support_IDentifier_value"
      -mos_id="My_Oracle_Support_ID"
```

Parameters

- **target_name**

  Name of the Cloud Control target.

- **target_type**

  Type of Cloud Control target

- csi

  Customer Support Identifier value to be assigned.

- mos_id

  My Oracle Support (MOS) user ID.

Example

```
emcli assign_csi_at_target_level
      -target_name="myhost.us.example.com"
      -target_type="oracle_example_type"
      -csi=12345678
      -mos_id="abc@xyz.com"
```

assign_csi_for_dbmachine_targets

Assigns or updates the Customer Support Identifier (CSI) for all of the associated Exadata, RAC, and database targets for a database machine name.

Format

```
emcli assign_csi_for_dbmachine_targets
      -target_name="database_system_name"
      -csi="customer_support_identifier_value"
      -mos_id="my_oracle_support_ID"
```

Parameters

- **target_name**

  Name of the database system target.

- **csi**

  Customer Support Identifier (CSI) to be assigned.

- **mos_id**

  My Oracle Support (MOS) user ID.

Example

This example assigns the CSI 1234567 to database system abcdef.company.com.

```
emcli assign_csi_for_dbmachine_targets
      -target_name=abcdef.company.com
      -csi=1234567
      -mos_id=abc@xyz.com
```

assign_test_to_target

Assigns a test-type to a target-type. If a test-type t is assigned to target-type T, all targets of type T can be queried with tests of type t.

Format

```
emcli assign_test_to_target
      -testtype=test-type_to_be_assigned
      -type=target_type
      [-tgtVersion]=version_of_target_type

[ ] indicates that the parameter is optional
```

Parameters

- **testtype**

  Test-type to be assigned. Should be the internal name; that is, 'HTTP' instead of 'Web Transaction'.

- **type**

  Service target type.

- **tgtVersion**

  Version of the target type. If not specified, the latest version is used.

Examples

This example assigns test type HTTP to targets of type generic service v2.

```
emcli assign_test_to_target -testtype='HTTP' -type='generic_service'
      -tgtVersion='2.0'
```

associate_cs_targets

Associates the specified standard with the listed targets.

Note: When the standard is provided by Oracle, the <std_name> is the standard internal name.

Format

```
 associate_cs_targets
  -name="<std_name>"
  -version="<std_version>"
  -author="<author_name>"
  -target_list="<target_name>[,<target_name>,<group_name>:Group]*"
  -target_list_file="<file_name>"

[ ] indicates that this parameter is optional
```

Parameters

- name

  Name of the standard.

- version

  Version of the standard.

- author

  Author of the standard. When the standard is provided by Oracle, the <std_name> is the standard internal name, for example, sysman.

- target_list

  Name of the target. This option is useful when a compliance standard is to be associated with one or a small number of targets. Targets are separated by commas. When providing a group target, it should be appended with ":Group". Examples are:

  -target_list="slc0host"

  -target_list="slc0host,slc-host01"

  -target_list="slc0host,host_grps:Group"

- target_list_file

  Name of the file that contains the list of targets. The targets can be either comma-separated values or in a file where the targets are listed on separate lines. Examples are:

  -target_list_file=slc0host,slc0host1,slc0host02

  -target_list_file="slc0host.txt" Where slc0host.txt contains the following lines:

  slc0host

  slc0host01

  slc0host02

  Note: Use either the target_list option or the target_list_file option.

Examples

Example 1

The following example specifies the target_list.

```
emcli associate_cs_targets
-name="secure configuration for host"
-version="1"
-author="sysman"
-target_list="host1,host2,group1:Group"
```

Example 2

The following example specifies the target_list_file.

```
emcli associate_cs_targets
-name="secure configuration for host"
-version="1"
-author="sysman"
-target_list_file="file with target name list"
```

associate_target_to_adm

Associates a target to an existing Application Data Model.

Format

```
emcli associate_target_to_adm
     -adm_name=<application_data_model_name>
     -target_name=<target_name>
     -target_type=<target_type>

[ ] indicates that the parameter is optional
```

Parameters

- **adm_name**

  Application Data Model name to which the target will be associated.

- **target_name**

  Application Data Model name to which the target will be associated.

- **target_type**

  Type of target that will be associated with the Application Data Model.

Output

Success/error messages

Examples

This example associates target test_database to the Application Data Model named Sample_ADM.

```
emcli associate_target_to_adm
       -adm_name=Sample_ADM
       -target_name=test_database
       -target_type=oracle_pdb
```

bareMetalProvisioning

Assigns a test-type to a target-type. If a test-type t is assigned to target-type T, all targets of type T can be queried with tests of type t.

Format

```
emcli bareMetalProvisioning
       [-input_file="config_properties:input_XML"]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Input XML file confirming to the XSD for bare metal provisioning. See below for a detailed XML file used to provision BMP.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Example

```
emcli bareMetalProvisioning
       [-input_file="config_properties:input XML"]
```

XML Example File

```
<?xml version="1.0"?>
<DeployedImage><ImageName>OracleLinux</ImageName><OperatingSystemType>Oracle
Enterprise Linux x86 32 bit</OperatingSystemType>

<!--Specify the Operating system type for this operation. Supported operating
system types are : Oracle Enterprise Linux x86 32 bit, Oracle Enterprise Linux x86
64 bit, RedHat Enterprise Linux x86 32 bit, RedHat Enterprise Linux x86 64 bit,
SUSE Linux Enterprise Server x86 32 bit, SUSE Linux Enterprise Server x86 64
bit-->
<TargetInfo><TargetType>MAC</TargetType>

<!--Specify the target type for this provisioning operation as follows : MAC : If
the provisioning target type is mac address. RE_IMAGE : If reprovisioning the
existing EM targets. Subnet : If provisioning all the targets in a Subnet.-->
<Target><MACAddress>aa:bb:cc:dd:ee:ff</MACAddress><NetworkInterface><InterfaceName
>eth0</InterfaceName><Configuration>Dhcp</Configuration>

<!--Specify the network interface type as follows : Dhcp : If the interface
configuration is to be dynamically assigned from a DHCP server. Static : If the
interface configuration is to be statically configured. Network Profile : If the
interface configuration is to be fetched from a Network Profile.-->
<Type>Non Bonding</Type>

<!--Specify the network interface type as follows : Non Bonding : If the interface
is not part of any bond. Bonding Master : If the interface is supposed to be the
Bonding master of a bond. Bonding Slave : If the interface is supposed to be a
Bonding slave as part of bond.-->
<!-- Following are bonding configuration-->
<BondingMode>1</BondingMode>

<!--Specify the Bonding Mode in case the interface has the role of Bonding
```

```
Master.-->
<SlaveDevices>eth1,eth2</SlaveDevices>

<!--Specify the Slave devices as a csv string in case the interface has the role
of Bonding Master. For ex : eth1,eth2-->
<PrimarySlaveDevice>eth1</PrimarySlaveDevice>

<!--Specify the Primary Slave device in case the interface has the role of Bonding
Master.-->
<ARPInterval>200</ARPInterval><ARPIPTarget>10.177.244.121</ARPIPTarget><ARPFrequen
cy>400</ARPFrequency><ARPDownDelay>200</ARPDownDelay><ARPUpDelay>200</ARPUpDelay>

<!--bonding configuration-->
<!-- If Configuration is Static <IPAddress></IPAddress> <HostName></HostName>
<Netmask></Netmask> <Gateway></Gateway> <DNSServers></DNSServers> -->
<IsBootable>true</IsBootable>

<!--Specify if the network interface is the bootable one.-->
</NetworkInterface></Target>

<!-- If TargetType is RE_IMAGE: replace <MACAddress> with following
<HostName>myhost.us.example.com</HostName>
<BootableMac>aa:bb:cc:dd:ee:ff</BootableMac> -->
<!-- If TargetType is Subnet: replace <MACAddress> with following
<SubnetIP>10.244.177.252</SubnetIP> <SubnetMask>255.255.252.0</SubnetMask> -->
</TargetInfo>

<!-- If TargetType is RE_IMAGE:
<HostTargetsPreferredCredSetName>cred1<.HostTargetsPreferredCredSetName> Specify
the preferred credentials name to be used for accessing the EM Host targetse to be
Re-imaged. -->
<StageServer>myhost.us.example.com</StageServer>

<!--Specify the Stage Server host name. For ex : myhost.mydomain.com-->
<StageStorage>/scratch/stage</StageStorage>

<!--Specify the Stage Storage on the stage server. For ex : /scratch/stage-->
<StageServerPreferredCredSetName>Cred1</StageServerPreferredCredSetName>

<!--Specify the preferred credentials name to be used for accessing the stage
server.-->
<StageServerPrereqs>false</StageServerPrereqs>

<!--Specify if the pre-requisties check should be run on the stage server before
starting the provisioning.-->
<BootServer>myhost.us.example.com</BootServer>

<!--Specify the Boot Server host name. For ex : myhost.mydomain.com-->
<BootServerPreferredCredSetName>Cred1</BootServerPreferredCredSetName>

<!--Specify the preferred credentials name to be used for accessing the boot
server.-->
<BootServerPrereqs>false</BootServerPrereqs>

<!--Specify if the pre-requisties check should be run on the boot server before
starting the provisioning.-->
<DhcpServer>myhost.us.example.com</DhcpServer>

<!--Specify the Dhcp Server host name. This is required only if DHCP automation is
required. For ex : myhost.mydomain.com Note : Dhcp automation is supported only
```

```
for the MAC and RE_IMAGE provisioning types.-->
<DhcpServerPreferredCredSetName>Cred1</DhcpServerPreferredCredSetName>

<!--Specify the preferred credentials name to be used for accessing the Dhcp
server.-->
<DhcpServerPrereqs>false</DhcpServerPrereqs>

<!--Specify if the pre-requisties check should be run on the Dhcp server before
starting the provisioning.-->
<RpmRepository>oelrepos</RpmRepository>

<!--Specify the RPM repository name to be used for provisioning operation. For ex
: OEL4U8repos-->
<RootPassword>password</RootPassword>

<!--Specify the root password for the provisioned machines.-->
<TimeZone>Africa/Algiers</TimeZone>

<!--Specify the time zone for the provisioned machines.-->
<AgentInstallUser>oraem</AgentInstallUser>

<!--Specify the user name for installing EM agent on the provisioned machines. For
ex : oraem-->
<AgentInstallGroup>dba</AgentInstallGroup>

<!--Specify the agent installation user's group for installing EM agent on the
provisioned machines. For ex : dba-->
<AgentBaseInstallationDirectory>/var/lib/oracle/agent12g</AgentBaseInstallationDir
ectory>

<!--Specify a directory for installing EM agent on the provisioned machines. For
ex : /var/lib/oracle/agent12g-->
<OracleInventoryLocation>/var/lib/oracle/oraInventory</OracleInventoryLocation>

<!--Specify a directory for storing oracle installed product's inventory on the
provisioned machines. For ex : /var/lib/oracle/oraInventory-->
<AgentRegistrationPassword>password</AgentRegistrationPassword>

<!--Specify agent registration password for installing EM agent on the provisioned
machines.-->
<AgentRpmUrl>http://myhost.us.example.com/oracle-agt.12.1.0.0.1-i386.rpm</AgentRpm
Url>

<!--Specify a http URL for fetching agent RPM. This is not mandatory if the agent
rom is already placed at the staged location-->
<ReferenceAnaconda/>

<!--Specify a reference anaconda as a string. It will be used to capture
properties like Keyboard, mouse. If not provided they will be defaulted to default
values.-->
<PackageList>@base</PackageList>

<!--Specify the package list to be installed on the provisioned machines.-->
<ACPI>off</ACPI>

<!--Specify the ACPI value for the provisioned machines. Supported values are :
on, off-->
<ParaVirtualizedKernel>false</ParaVirtualizedKernel>

<!--Specify if the provisioned machines should be booted with paravirtualized
```

```
kernels.-->
<PostInstallScript>%post echo "post" </PostInstallScript>

<!--This section provides the option of adding commands to be run on the system
once the installation is complete. This section must start with the %post
command.-->
<FirstBootScript>#!/bin/sh # chkconfig: 345 75 25 # description: Bare Metal
Provisioning First boot service # </FirstBootScript>

<!--This section provides the option of adding commands to run on the system when
it boots for the first time after installation.-->
<RequireTTY>false</RequireTTY>

<!--Specify if tty is required on the provisioned machines.-->
<SeLinux>Disabled</SeLinux>

<!--Specify the SELinux configuration for the provisioned machines. Supported
values are : Disabled, Enforcing, Permissive-->
<MountPointSettings/>

<!--Specify /etc/fstab settings for the provisioned machines.-->
<NISSettings/>

<!--Specify /etc/yp.conf settings for the provisioned machines.-->
<NTPSettings/>

<!--Specify /etc/ntp.conf settings for the provisioned machines.-->
<KernelParameterSettings/>

<!--Specify /etc/inittab settings for the provisioned machines.-->
<FirewallSettings/>

<!--Specify the firewall settings for the provisioned machines.-->
<HardDiskProfiles>

<!--Specify the Hard Disk profiles for the provisioned machines.-->
<HardDiskConfiguration>

<!--Specify the hard disk configuration details-->
<DeviceName>hda</DeviceName>

<!--Specify the device name for the disk. For ex : hda,hdb-->
<Capacity>10000000</Capacity>

<!--Specify the disk capacity in MB. For ex : 1024-->
</HardDiskConfiguration></HardDiskProfiles><PartitionConfigurations>

<!--Specify the partition configurations for the provisioned machines.-->
<PartitionConfiguration>

<!--Specify the partition configuration details.-->
<MountPoint>/</MountPoint>

<!--Specify the mount point for the partition. For ex : /, /root-->
<DeviceName>hda</DeviceName>

<!--Specify the disk name on which this partition has to be configured. For ex :
hda,hdb-->
<SystemDeviceName>/dev/hda1</SystemDeviceName>
```

```
<!--For ex : /dev/hda1-->
<FileSystemType>ext3</FileSystemType>

<!--Specify the File System type for this partition. Supported file system types
are : ext2, ext3, ocfs2, swap, Raid, LVM-->
<Size>4096</Size>

<!--Specify the size in MB for this partition. For ex : 5120-->
</PartitionConfiguration></PartitionConfigurations><RaidConfigurations>

<!--Specify the RAID configurations for the provisioned machines.-->
<RaidConfiguration>

<!--Specify the RAID configuration details-->
<MountPoint>raid.100</MountPoint>

<!--Specify the raid id . For ex : raid.100-->
<RaidLevel>0</RaidLevel>

<!--Specify the RAID Level for this raid device. Supported RAID Levels are : Raid
0, Raid 1, Raid 5, Raid 6-->
<Partitions>/dev/hda1,/dev/hda2</Partitions>

<!--Specify the raid partitions for this raid device as a csv string. For ex :
/dev/hda1, /dev/hda2-->
<FileSystemType>ext3</FileSystemType>

<!--Specify the File System type for this partition. Supported file system types
are : ext2, ext3, ocfs2, swap, LVM-->
</RaidConfiguration></RaidConfigurations><LogicalVolumeGroups>

<!--Specify the Logical Volume Groups for the provisioned machines.-->
<LogicalVolumeGroup>

<!--Specify the logical volume group configuration details-->
<GroupName>LVG1</GroupName>

<!--Specify the Logical group name. For ex : mygrp-->
<Partitions>/dev/hda1</Partitions>

<!--Specify the partitions that take part in this logical volume group as a csv
string. For ex : /dev/hda1, /dev/hda2-->
<Raids>raid.100</Raids>

<!--Specify the RAIDs that take part in this logical volume group as a csv string.
For ex : raid.100, raid.200-->
</LogicalVolumeGroup></LogicalVolumeGroups><LogicalVolumes>

<!--Specify the Logical Volumes for the provisioned machines.-->
<LogicalVolume>

<!--Specify the logical volume configuration details.-->
<MountPoint>/u01</MountPoint>

<!--Specify the mount point for this logical volume. For ex : /, /root-->
<LogicalVolumeName>LV1</LogicalVolumeName>

<!--Specify the logial volume name. For ex : myvols-->
<LogicalGroupName>LVG1</LogicalGroupName>
```

```
<!--Specify the logical group name where this volume should be created. For ex :
mygrp-->
<FileSystemType>ext3</FileSystemType>

<!--Specify the File System type for this partition. Supported file system types
are : ext2, ext3, ocfs2, swap-->
<Size>4096</Size>

<!--Specify the size in MB for this partition. For ex : 5120-->
</LogicalVolume></LogicalVolumes></DeployedImage>
```

cancel_cloud_service_requests

Cancels scheduled cloud service request(s) initiated by the specified user. Note that only scheduled requests can be cancelled.

Format

```
emcli cancel_cloud_service_requests
      -user="username"
      [-family="family"]
      [-ids="id1;id2..."]

[ ] indicates that the parameter is optional
```

Parameters

- **user**

  Name of the user who initiated the requests.

- **family**

  Service family name to use to filter cloud requests.

- **ids**

  List of Request IDs to use to filter cloud requests. Separate each ID with a semicolon (;).

Examples

Example 1

This example cancels all scheduled cloud requests owned by user1.

```
emcli cancel_cloud_service_requests -user="user1"
```

Example 2

This example cancels all cloud requests owned by user1 and belonging to the family1 service family.

```
emcli cancel_cloud_service_requests -user="user1" -family="family1"
```

Example 3

This example cancels cloud requests 1 and 2 owned by user1.

```
emcli cancel_cloud_service_requests -user="user1" -ids="1;2"
```

change_service_system_assoc

Changes the system that hosts a given service.

Format

```
emcli change_service_system_assoc
      -name='name'
      -type='type'
      -systemname='system_name'
      -systemtype='system_type'
      -keycomponents='keycomp1name:keycomp1type[;keycomp2name:keycomp2type;...]'

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service name.

- **type**

  Service type.

- **systemname**

  System on which the service resides.

- **systemtype**

  System type.

- **keycomponents**

  Name-type pair (such as `keycomp_name:keycomp_type`) list of key components in the system used for the service.

Example

This example changes the system for a generic service named `my service` to a generic system named `my system` with specified key components.

```
emcli change_service_system_assoc
      -name='my service' -type='generic_service'
      -systemname='my system' -systemtype='generic_system'
      -keycomponents='database:oracle_database; mytestbeacon:oracle_beacon'
```

change_target_owner

Changes the owner of the target.

Format

```
emcli change_target_owner
        -target="target_name:target_type"
        [-target="target_name:target_type"]
        -owner="current_target_owner_name"
        -new_owner="new_owner_name"

[ ] indicates that the parameter is optional
```

Parameters

- **target**

  Target name and target type to change the owner.

- **owner**

  Name of the existing owner of the target. The default value for this parameter is the currently logged in user.

- **new_owner**

  New owner name of the target.

Example

This example changes the ownership of two targets from admin to admin2.

```
emcli change_target_owner
          -target="abc.oracle.com:host"
          -target="testDBSystem:oracle_database"
          -owner="admin1"
          -new_owner="admin2"
```

cleanup_dbaas_requests

Cleans up requests from the host and Enterprise Manager. Depending on the parameters specified, this verb:

- Cleans up all failed requests from a pool.
- Cleans up all requests from a pool.
- Cleans up failed requests for a specific user.
- Cleans up all requests for a specific user.

Format

```
emcli help cleanup_dbaas_requests
emcli cleanup_dbaas_requests
        [-ids="<request id>"]
        [-pool_name="<pool name>" -pool_type="
                    <database|schema|pluggable_database>"
        [-user="<SSA user name>"]
        [-all]]

[ ] indicates that the parameter is optional.
```

Parameters

- **ids**

  Request ID to be used for filtering Cloud requests, separated by semicolons(;).

- **pool_name**

  Name of the pool from which requests must be cleaned up.

- **pool_type**

  Type of pool. Enter one of the following values:

  – For database pools: database

  – For schema pools: schema

  – For pluggable database pools: pluggable_database

- **user**

  User name to be used for filtering requests for deletions.

- **all**

  If specified, cleans up all requests (successful and failed), cancels the requests that are in a scheduled state, and leaves the requests that are in progress as is. If this parameter is not specified, cleanup is performed on failed requests only.

> **Note:** The ids and pool_name parameters cannot be used together.

Examples

Example 1

This example performs the cleanup for the specified request ID 10.

```
emcli cleanup_dbaas_requests -ids="10"
```

Example 2

This example performs the cleanup for the specified request IDs 10, 11, and 12.

```
emcli cleanup_dbaas_requests -ids="10;11;12"
```

Example 3

This example performs a cleanup of all failed requests.

```
emcli cleanup_dbaas_requests -pool_name="database_pool" -pool_type="database"
```

Example 4

This example performs a cleanup of all requests, both failed and successful. This process essentially resets the pool.

```
emcli cleanup_dbaas_requests -pool_name="database_pool" -pool_type="database" -all
```

Example 5

This example performs a cleanup of all requests (failed and successful) for a specific user. This option is useful in cases where the user is no longer in the system and the administrator wants to clean up all of the service instances owned by this user.

```
emcli cleanup_dbaas_requests -pool_name="database_pool" -pool_type="database" -all
-user="SSA_USER"
```

clear_credential

Clears preferred or monitoring credentials for given users.

Format

```
emcli clear_credential
      -target_type="ttype"
      [-target_name="tname"]
      -credential_set="cred_set"
      [-user="user"]
      [-oracle_homes="home1;home2"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target, which must be "host" if you specify the oracle_homes parameter.

- **target_name**

  Name of the target. Omit this option to clear enterprise-preferred credentials. The target name must be the host name if you specify the oracle_homes parameter.

- **credential_set**

  Credential set affected.

- **user**

  Enterprise Manager user whose credentials are affected. If omitted, the current user's credentials are affected. This value is ignored for monitoring credentials.

- **oracle_homes**

  Name of Oracle homes on the target host. Credentials are cleared for all specified homes.

Examples

```
emcli clear_credential
       -target_type=oracle_database
       -target_name=myDB
       -credential_set=DBCredsNormal
       -user=admin1


emcli clear_credential
       -target_type=oracle_database
       -credential_set=DBCredsNormal
       -user=admin1
```

clear_default_pref_credential

Clears the named credential set as the default preferred credential for the user. The named credential is not deleted from the credential store. Only the user preference to use the named credential as the default preferred credential is cleared.

Format

```
emcli clear_default_pref_cred
        -set_name="set_name"
        -target_type="ttype"
```

Parameters

- **set_name**

  Clears the default preferred credential for this credential set.

- **target_type**

  Target type for the credential set.

Examples

This example clears the default preferred credential set for the host target type for the HostCredsNormal credential set.

```
emcli clear_default_pref_cred
        -set_name=HostCredsNormal
        -target_type=host
```

clear_default_privilege_delegation_setting

Clears the default privilege delegation settings for a specified platform.

Format

Standard Mode

```
emcli clear_default_privilege_delegation_setting
        -platform_list="PLATFORM_DEFAULT"
```

Interactive or Script Mode

```
clear_default_privilege_delegation_setting(
        platform_list="PLATFORM_DEFAULT"
        )

[ ] indicates that the parameter is optional
```

Parameters

- **platform_list**

  Comma-separated list of platforms for which default privilege delegation settings are removed. Supported platforms: Linux, HP-UX, SunOS, and AIX.

Exit Codes

0 on success. A non-zero value means verb processing was not successful.

Examples

Example 1

This example clears the default privilege delegation setting for Linux, HP-UX, SunOS, and AIX platforms.

```
emcli clear_default_privilege_delegation_setting
-platform_list="Linux,HP-UX,SunOS,AIX"
```

clear_monitoring_credential

Clears the monitoring credential set for the target.

Format

```
emcli clear_monitoring_credential
        -set_name="set_name"
        -target_name="target_name"
        -target_type="ttype"
```

Parameters

- **set_name**

  Clears the monitoring credential for this credential set.

- **target_name**

  Clears the preferred credential for this target.

- **target_type**

  Target type for the target/credential set.

Examples

This example clears the monitoring credential set for the target testdb.example.com for the DBCredsMonitoring credential set.

```
emcli clear_monitoring_credential
        -set_name=DBCredsMonitoring
        -target_name=testdb.example.com
        -target_type=oracle_database
```

clear_preferred_credential

Clears the named credential set as the target preferred credential for the user. The named credential is not deleted from the credential store. Only the user preference to use the named credential as the preferred credential is cleared.

Format

```
emcli clear_preferred_credential
        -set_name="set_name"
        -target_name="target_name"
        -target_type="ttype"
```

Parameters

- **set_name**

  Sets the preferred credential for this credential set.

- **target_name**

  Clears the preferred credential for this target.

- **target_type**

  Target type for the target/credential set.

Examples

This example clears the preferred credential set for the host target test.example.com for the HostCredsNormal credential set.

```
emcli clear_preferred_credential
        -set_name=HostCredsNormal
        -target_name=test.example.com
        -target_type=host
```

clear_privilege_delegation_setting

Clears the privilege delegation setting from a given host or hosts.

Format

```
emcli clear_privilege_delegation_setting
        -host_names="name1;name2;..."
        [-input_file="FILE:file_path"]
        [-force="yes/no"]

[ ] indicates that the parameter is optional
```

Parameters

- **host_names**

  Names of the hosts.

- **input_file**

  Path of the file that has the list of hosts. The file should have one host name per line.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **force**

  If set to yes, invalid and unreachable targets are ignored and the setting is removed from all valid and up targets. If set to no, invalid and down targets raise an error. The default is no.

Examples

Example 1

```
emcli clear_privilege_delegation_setting
      -host_names="host1;host2;....."
```

Example 2

```
emcli clear_privilege_delegation_setting
      -host_names="host1;host2;....."
      -force=yes
```

Example 3

```
emcli clear_privilege_delegation_setting
      -input_file="FILE:/home/user/file.txt"
      -force=yes
```

clear_problem

Clears problems matching the specified criteria (problem key, target type, and age). Only users with Manage Target privilege can clear the problems for a target. When a problem is cleared, the underlying incidents and events are also cleared.

By default, the problem notification is not sent out. You can override this by specifying the send_notification option. Clearing the underlying incidents and events does not send out a notification.

Format

```
emcli clear_problem
        -problem_key="problem_key"
        -target_type="target_type"
        -older_than="age_of_problem"
        [-target_name="target_name"]
        [-unacknowledged_only="clear_unacknowledged_problems"]
        [-send_notification="send_notifications_for_problems"]
        [-preview]

[ ] indicates that the parameter is optional
```

Parameters

- **problem_key**

  Problem key of the problem to be cleared

- **target_type**

  Internal type name, such as oracle_database for "Oracle Database." You can use the get_target_types command to get the internal name for a target type.

- **older_tha n**

  Specify the age (in days) of the problem.

- **target_name**

  Name of an existing non-composite target. For example, the name of a single database. You cannot use the name of composite targets (target group).

- **unacknowledged_only**

  If provided, only the unacknowledged problems are cleared. This option does not require any value.

- **send_notification**

  If provided, any applicable notification is sent out for cleared problems. By default, no notification is sent for cleared problems. This parameter does not require any value.

- **preview**

  Gets the number of problems that the command would clear.

Examples

Example 1

This example displays the number of problems matching the specified criteria.

```
emcli clear_problem -problem_key="ORA-600" -target_type="oracle_database"-preview
```

Example 2

This example clears ORA-600 problems across all databases that have occurred (based on the occurrence date of the first incident) for at least 3 days.

```
emcli clear_problem -problem_key="ORA-600" -target_type="oracle_database" -older_
than="3"
```

Example 3

This example clears only unacknowledged problems.

```
emcli clear_problem -problem_key="ORA-600" -target_type="oracle_database"
-older_than="3" -unacknowledged_only
```

Example 4

This example sends applicable notifications when the problem clears. By default, a notification is not sent for the cleared problems.

```
emcli clear_problem -problem_key="ORA-600" -target_type="oracle_database"
-older_than="3" - send_notification
```

clear_stateless_alerts

Clears the stateless alerts associated with the specified target. Only a user can clear these stateless alerts; the Enterprise Manager Agent does not automatically clear these alerts. To find the metric internal name associated with a stateless alert, use the get_metrics_for_stateless_alerts verb.

You cannot use this command to clear stateless alerts associated with diagnostic incidents. You can only clear these alerts in the Enterprise Manager console by clearing their associated Incident or Problem.

Format

```
emcli clear_stateless_alerts
        -older_than=number_in_days
        -target_type=target_type
        -target_name=target_name
        [-include_members]
        [-metric_internal_name=target_type_metric:metric_name:metric_column]
        [-unacknowledged_only]
        [-ignore_notifications]
        [-preview]

[ ] indicates that the parameter is optional
```

Parameters

- **older_than**

  Specify the age of the alert in days. (Specify 0 for currently open stateless alerts.)

- **target_type**

  Internal target type identifier, such as host, oracle_database, and emrep.

- **target_name**

  Name of the target.

- **include_members**

  Applicable for composite targets to examine alerts belonging to members as well.

- **metric_internal_name**

  Metric to be cleaned up. Use the get_metrics_for_stateless_alerts verb to see a complete list of supported metrics for a given target type.

- **unacknowledged_only**

  Only clear alerts if they are not acknowledged.

- **ignore_notifications**

  Use this option if you do not want to send notifications for the cleared alerts. This may reduce the notification sub-system load.

- **preview**

  Shows the number of alerts to be cleared on the target(s).

Examples

This example clears alerts generated from the database alert log over a week old. In this example, no notifications are sent when the alerts are cleared.

```
emcli clear_stateless_alerts -older_than=7 -target_type=oracle_database -tar
get_name=database -metric_internal_name=oracle_database:alertLog:genericErrStack
-ignore_notifications
```

clone_as_home

Clones the specified Application Server Oracle Home or S/W Library component from the target host to specified destinations. For a Portal and Wireless installation, the OID user and password are also needed. For a J2EE instance connected to only a DB-based repository, a DCM Schema password is needed.

Passing Variables Through EM CLI

When working with variables such as `%perlbin%` or `%oracle_home%`, EM CLI passes variable values from the current local environment instead of the variables themselves. To pass variables through an EM CLI command, as might be the case when using the `-prescripts` or `-postscripts` options, you can place the EM CLI command in a batch file and replace all occurrences of `%` with `%%`.

Format

```
emcli clone_as_home
       -input_file="dest_properties:file_path"
       -list_exclude_files="list of files to exclude"
       -isSwLib="true/false"
       -tryftp_copy="true/false"
       -jobname="name of cloning job"
       -iasInstance=instance
       -isIas1013="true/false"
       [-oldIASAdminPassword=oldpass]
       [-newIASAdminPassword=newpass]
       [-oldoc4jpassword=oldpass]
       [-oc4jpassword=newpass]
       [-oiduser=oid admin user]
       [-oidpassword=oid admin password]
       [-dcmpassword=dcm schema password]
       [-prescripts="script name to execute"]
       [-run_prescripts_as_root="true/false"]
       [-postscripts="script to execute"]
       [-run_postscripts_as_root="true/false"]
       [-rootscripts="script name to execute"]
       [-swlib_component ="path:path to component;version:rev"]
       [-source_params="TargetName:name;HomeLoc:loc;HomeName:name;
         ScratchLoc:Scratch dir Location"
       [-jobdesc="description"]

[ ] denotes that the parameter is optional
```

Options

- **input_file="dest_properties:file_path"**

  File containing information regarding the targets.

  Each line in the file corresponds to information regarding one destination.

  Format:

  ```
  Destination Host Name1;Destination Home Loc; Home Name; Scratch
  Location;
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **list_exclude_files**

Comma-separated list of files to exclude. Not required if the source is software lib. "*" can be used as a wild card.

- **isSwLib**

  Specifies whether it is an Oracle Home database or Software Library.

- **ryftp_copy**

  Try FTP to copy or not. You should set the FTP copy option to false when using EM CLI from the command line.

- **jobname**

  Name of the cloning job.

- **iasInstance**

  Name of instance.

- **isIas1013**

  Specifies whether this is a 10.2.3 Ias home.

- **oldoc4jpassword**

  Old OC4j password. (Required for 10.1.3 Ias homes.)

- **oc4jpassword**

  New OC4J password. (Required for 10.1.3Ias homes.)

- **oldIASAdminPassword**

  Old Application Server administrator password. (Not required for 10.1.3 Ias homes.)

- **newIASAdminPassword**

  New Application Server administrator password. (Not required for 10.1.3 Ias homes.)

- **oiduser**

  OID admin user.

- **oidpassword**

  OID admin password.

- **dcmpassword**

  DCM schema password.

- **prescripts**

  Path of script to execute.

  Note: Double-quoted parameters can be passed using an escape (\) sequence. For example:

  ```
  prescripts=" <some value here>=\"some value here\" "
  ```

- **run_prescripts_as_root**

  Run prescripts as "root". By default, the option is set to false.

- **postscripts**

Path of script to execute.

- **run_postscripts_as_root**

  Run postscripts as "root". By default, the option is set to false.

- **rootscripts**

  Path of the script to execute. The job system environment variables (`%oracle_home%`, `%perl_bin%`) can be used for specifying script locations.

- **swlib_component**

  Path to the Software Library to be cloned. "isSwLib" must be true in this case.

- **source_params**

  Source Oracle home information. "isSwLib" must be false in this case.

- **jobdesc**

  Description of the job. If not specified, a default description is generated automatically.

Examples

```
emcli clone_as_home
   -input_file="dest_properties:/home/destinations.txt"
   -list_exclude_files="centralagents.lst"
   -isSwLib="false"
   -tryftp_copy="false"
   -jobname="clone as home"
   -iasInstance="asinstancename"
   -isIas1013="false"
   -oldIASAdminPassword="oldpassword"
   -newIASAdminPassword="newpassword"
   -prescripts="/home/abc/myscripts"
   -run_prescripts_as_root="true"
   -rootscripts="%oracle_home%/root.sh"
   -source_params="TargetName:host.domain.com;HomeLoc=/home/oracle/appserver1;
    HomeName=oracleAppServer1;ScratchLoc=/tmp"
```

clone_crs_home

Creates an Oracle Clusterware cluster given a source Clusterware home location or a Clusterware S/W Library component for specified destination nodes.

Format

```
emcli clone_crs_home
       -input_file="dest_properties:file_path"
       -list_exclude_files="list of files to exclude"
       -isSwLib="true/false"
       -tryftp_copy="true/false"
       -jobname="name of cloning job"
       -home_name="name of home to use when creating Oracle Clusterware cluster"
       -home_location="location of home when creating Oracle Clusterware cluster"
       -clustername=name of cluster to create
       [-isWindows="false/true"]
       [-ocrLoc=ocr location]
       [-vdiskLoc=voting disk location]
       [-prescripts="script name to execute"]
       [-run_prescripts_as_root="true/false"]
       [-postscripts="script to execute"]
       [-run_postscripts_as_root="true/false"]
       [-rootscripts="script name to execute"]
       [-swlib_component ="path:path to component;version:rev"]
       [-source_params="TargetName:name;HomeLoc:loc;HomeName:name;
         ScratchLoc:Scratch dir Location"]
       [-jobdesc="description"]

[ ] denotes that the parameter is optional
```

Options

- **input_file="dest_properties:file_path"**

  File containing information regarding the targets.

  Each line in the file corresponds to information regarding one destination.

  Format:

  ```
  Destination Host Name;Destination Node Name;Scratch
  Location;PVTIC;VirtualIP;
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **list_exclude_files**

  Comma-separated list of files to exclude. Not required if the source is software lib. An asterisk "*" can be used as a wildcard.

- **isSwLib**

  Specifies whether it is an Oracle Home database or Software Library.

- **tryftp_copy**

  Try FTP to copy or not. You should set the FTP copy option to false when using `emcli` from the command line.

- **jobname**

Name of the cloning job.

- **home_name**

  Name of the home to use for all homes in the Oracle Clusterware cluster.

- **home_location**

  Location of the home to use for all homes in the Oracle Clusterware cluster.

- **clustername**

  Name of the cluster to create.

- **isWindows**

  Specify whether the cloning source is on a Windows Platform. This option only applies for creating CRS cloning from a Gold Image source. The default value is false.

- **ocrLoc**

  Oracle Cluster Registry Loaction.

- **vdiskLoc**

  Voting disk location.

- **prescripts**

  Path of the script to execute.

  Note: Double-quoted parameters can be passed using an escape (\) sequence. For example:

  ```
  prescripts=" <some value here>=\"some value here\" "
  ```

- **run_prescripts_as_root**

  Run prescripts as "root". By default, this option is set to false.

- **postscripts**

  Path of the script to execute.

- **run_postscripts_as_root**

  Run postscripts as "root". By default, it is false.

- **rootscripts**

  Path of the script to execute.

- **swlib_component**

  Path to the Software Library to be cloned. "isSwLib" must be true in this case.

- **source_params**

  Source Oracle home info. "isSwLib" must be false in this case.

- **jobdesc**

  Description of the job. If not specified, a default description is generated automatically.

Examples

```
emcli clone_crs_home -input_file="dest_properties:crs.prop"  -isSwLib="true"
  -tryftp_copy="true" -jobname="crs cloning job2" -home_name="cloneCRS1"
  -home_location="/scratch/scott/cloneCRS1 " -clustername="crscluster"
  -ocrLoc="/scratch/shared/ocr" -vdiskLoc="/scratch/shared/vdisk"
  -postscripts="%perlbin%/perl%emd_root%/admin/scripts/cloning/samples/
   post_crs_ create.pl ORACLE_HOME=%oracle_home%"
  -run_postscripts_as_root="true" -rootscripts="%oracle_home%/root.sh"
  -swlib_component="path:Components/crscomp;version:.1"
```

Passing Variables Through EM CLI

When working with variables such as `%perlbin%` or `%oracle_home%`, EM CLI passes variable values from the current local environment instead of the variables themselves. To pass variables through an EM CLI command, as might be the case when using the `-prescripts` or `-postscripts` options, you can place the EM CLI command in a batch file and replace all occurrences of `%` with `%%`.

clone_database

Clones a database.

Format

```
emcli clone_database
      -source_db_name="source_database_name"
      -dest_global_dbname="global_name_of_clone_database"
      -dest_oracle_sid="clone_database_instance_name"
      [-dest_host_name="clone_host_name"]
      [-dest_oracle_home="clone_database_oracle_home"]
      [-source_db_creds_name="source_database_credential_name"]
      [-source_host_creds_name="source_database_host_credential_name"]
      [-dest_host_creds_name="clone_database_host_credential_name"]
      [-asm_inst_creds_name="asm_instance_credential_name"]
      [-dest_target_name="clone_database_name"]
      [-clone_type="clone_type"]
      [-source_staging_area="source_staging_directory"]
      [-dest_staging_area="clone_database_staging_directory"]
      [-dest_storage_type="clone_database_storage_type"]
      [-dest_database_area="clone_database_files_location"]
      [-dest_recovery_area="clone_database_fast_recovery_area"]
      [-dest_listener_selection="clone_database_listener_selection"]
      [[-dest_listener_name="clone_database_listener_name"]
       [-dest_listener_port="clone_database_listener_port"]]
      [-configure_with_oracle_restart"]
      [-job_name="job_name"]
      [-job_desc="job_description"

[ ] denotes that the parameter is optional
```

Options

- **source_db_name**

  Source database Enterprise Manager target name. Can be either a single-instance database or a cluster database instance.

- **dest_global_dbname**

  Clone database global database name. Usually specified as <name>.<domain>, with <name> being used for the db_unique_name and <domain> for the db_domain_name parameters.

- **dest_oracle_sid**

  Clone database instance name.

- **dest_host_name**

  Clone database host name. If not specified, the clone database is created on the same host as the source database.

- **dest_oracle_home**

  Clone database Oracle home. If not specified, the Oracle Home of the source database is used.

- **source_db_creds_name**

  Source database named credential.

- **source_host_creds_name**

  Source database host named credential.

- **dest_host_creds_name**

  Destination (clone) host named credential.

- **asm_inst_creds_name**

  Automatic Storage Management(ASM) named credential.

- **dest_target_name**

  Clone database Enterprise Manager target name.

- **clone_type**

  Type of source database backup that will be used for cloning. Valid values are:

  – DUPLICATE — Database files are moved directly to the clone database host by the Recovery Manager (RMAN).

  – STAGING — Database files are backed-up into the staging area and moved to the clone database host through HTTP.

  – EXISTING_BACKUP — Database files are restored from existing backups to the clone database host by the Recoverey Manager (RMAN).

- **pitr_date**

  Clone database as of the specified date in MM/dd/yyyy hh:mm:a (Month/Date/Year Hours:Minutes:AM/PM marker) format. For example: 03/22/2014 08:25:AM. If not specified, the clone database is created as of the latest point-in-time. This option is applicable when the clone_type is EXISTING_BACKUP.

- **pitr_scn**

  Clone database as of the specified System Change Number of the source database. If not specified, the clone database is created as of the latest point-in-time. This option is applicable when the clone_type is EXISTING_BACKUP.

- **encryption_mode**

  Encryption mode of the existing source database backups. If not specified, the default value is NONE. This option is applicable when the clone_type is EXISTING_BACKUP. Valid values are:

  – WALLET — Backups are encrypted using Oracle Encryption Wallet.

  – PASSWORD — Backups are encrypted using a password.

  – DUAL — Backups are encrypted using both Oracle Encryption Wallet and a password.

- **backups_encryption_creds_name**

  Database named credential for the encrypted backups. This option is applicable if encryption_mode is PASSWORD or DUAL.

  Note: This parameter is applicable only if the clone type is EXISTING_BACKUP and the database backups are encrypted using a password. This database credential should be created in Enterprise Manager of scope GLOBAL with the user name specified as "backup_admin".

- **tape_settings**

  Media management vendor settings if the database backups are on tape. This option is applicable when clone_type is EXISTING_BACKUP.

- **db_backups_location**

  The location of the backups to be transferred to the destination host. Multiple values can be specified using "," as a delimiter. This option is applicable when cloning to a different host and clone_type is EXISTING_BACKUP.

  Note: This parameter is applicable only if the clone type is EXISTING_BACKUP and the database clone occurs on a different host where the source database backups are not visible. If the backups are visible from the destination host, this parameter should NOT be specified.

  - It is recommended that if the size of the database backups is very large, the backups should be taken in a common location visible from the destination host.
  - If the source database backups are on ASM diskgroups, ensure that the diskgroups are mounted at the destination host as these backups are not transferred.
  - When you specify this parameter is specified, all of the available files at this location are transferred to a temporary staging location at the destination host.
  - You can specify multiple values for this parameter with comma (,) as a delimiter.

- **source_staging_area**

  Staging area used to store the backup of source database. This option is applicable when clone_type is STAGING.

- **dest_staging_area**

  Staging area used to store backup files transferred from source host. This option is applicable when clone_type is STAGING.

- **dest_storage_type**

  Clone database storage type. Valid values are:

  – FILE_SYSTEM — Clone database files will be in a regular file system (using Oracle-managed Files).

  – ASM — The clone database will use Automatic Storage Management (ASM).

- **dest_database_area**

  Oracle-managed files (OMF) location for clone database files. This can be a regular file system (if storage_type is FILE_SYSTEM) or an ASM disk group (if storage_type is ASM). If not specified, a default value is used.

- **dest_recovery_area**

  Fast recovery area location. If not specified, a default value is used.

- **dest_listener_selection**

  Clone database listener selection. Valid values are:

  - GRID_INFRA — Use Grid Infrastructure Home listener.
  - DEST_DB_HOME — Use the listener from the clone database Oracle Home.

- **dest_listener_name**

  Clone database listener name. This option is applicable only if dest_listener_selection is DEST_DB_HOME. If not specified, the first existing TCP listener found in the clone database Oracle Home is used. If you specify this option, you must also specify dest_listener_port.

- **dest_listener_port**

  Clone database listener port. This option is applicable only if dest_listener_selection is DEST_DB_HOME. If you specify this option, you must also specify dest_listener_name.

- **configure_with_oracle_restart**

  Configure the clone database with Oracle Restart if the clone host has Oracle Restart configured. Oracle Restart automatically starts the database when required.

- **job_name**

  Unique job name for the clone job in the Enterprise Manager repository.

- **job_desc**

  Job description.

Examples

Example 1

```
emcli clone_database -source_db_name="database" -dest_target_name="dbClone1"
-dest_host_name="host1" -dest_oracle_home="/ade/ngade_gct/oracle" -dest_oracle_
sid="dbClone1" -dest_global_dbname="dbClone1" -dest_listener_selection="DEST_DB_
HOME" -clone_type="DUPLICATE" -dest_storage_type="FILE_SYSTEM"
```

Example 2

```
emcli clone_database -source_db_name="database" -source_db_creds_name="NC_
DBCREDS1" -source_host_creds_name="NC_HOST_CREDS1" -dest_host_name="host1" -dest_
host_creds_name="NC_HOST_CREDS2" -dest_oracle_
home="/u01/app/oracle/product/11.2.0/dbhome_2" -dest_oracle_sid="TESTDB1" -dest_
global_dbname="TESTDB1" -dest_listener_selection="GRID_INFRA" -clone_
type="EXISTING_BACKUP" -dest_storage_type="FILE_SYSTEM" db_backups_
location="/oracle/dir1"
```

Example 3

```
emcli clone_database -source_db_name="database" -source_db_creds_name="NC_
DBCREDS1" -source_host_creds_name="NC_HOST_CREDS1" -dest_host_creds_name="NC_HOST_
CREDS2" -dest_oracle_home="/u01/app/oracle/product/11.2.0/dbhome_2" -dest_oracle_
sid="TESTDB2" -dest_global_dbname="TESTDB2" -dest_listener_selection="GRID_INFRA"
-clone_type="EXISTING_BACKUP" -pitr_date="03/22/2014 08:25:AM" -dest_storage_
type="FILE_SYSTEM"
```

clone_database_home

Clones the specified Oracle Home or S/W Library from the target host to specified destinations. If the isRac option is true, a RAC cluster is created. If the isRac option is true, the home name and location of the RAC cluster are needed.

Format

```
emcli clone_database_home
      -input_file="dest_properties:file_path"
      -list_exclude_files="files_to_exclude"
      -isSwLib="true|false"
      -isRac="true|false"
      -tryftp_copy="true|false"
      -jobname="name_of_cloning_job"
      [-home_name="home_when_creating_RAC_cluster"]
      [-home_location="location_of_home_when_creating_RAC_cluster"]
      [-prescripts="script_name_to_execute"]
      [-run_prescripts_as_root="true|false"]
      [-postscripts="script_to_execute"]
      [-run_postscripts_as_root="true|false"]
      [-rootscripts="script_name_to_execute"]
      [-swlib_component ="path:path_to_component;version:rev"]
      [-source_params="TargetName:name;HomeLoc:loc;HomeName:name;
        ScratchLoc:scratch_dir_location"
      [-jobdesc="description"]

[ ] denotes that the parameter is optional
```

Options

- **input_file=dest_properties**

  File containing information regarding the targets. Each line in the file corresponds to information regarding one destination.

  Format if cloning a database (isRac is false):

  ```
  Destination Host Name1;Destination Home Loc; Home Name; Scratch
  Location;
  ```

  Format if cloning a RAC cluster (isRac is true):

  ```
  Host Name;Node Name;Scratch Location;
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **list_exclude_files**

  Comma-separated list of files to exclude. This is not required if the source is software lib. "*" can be used as a wild card.

- **isSwLib**

  Specifies whether the source is an Oracle Home database or Software Library.

- **isRac**

  Specifies whether cloning in RAC mode. If the isRac option is true, a RAC cluster is created. If the isRac option is true, the home name and location of the RAC cluster are needed.

- **tryftp_copy**

  Try FTP to copy or not. You should set the FTP copy option to false when using EM CLI from the command line.

- **jobname**

  Name of the cloning job.

- **home_name**

  Name of the home to use when creating a RAC cluster.

- **home_location**

  Location of the home to use when creating a RAC cluster.

- **prescripts**

  Path of the script to execute.

  Note: Double-quoted parameters can be passed using an escape (\) sequence. For example:

  ```
  prescripts=" <some value here>=\"some value here\" "
  ```

- **run_prescripts_as_root**

  Run prescripts as "root". By default, it is false.

- **postscripts**

  Path of the script to execute.

- **run_postscripts_as_root**

  Run postscripts as "root". By default it is false.

- **rootscripts**

  Path of the script to execute. You can use the job system environment variables (`%oracle_home%`, `%perl_bin%`) to specify script locations.

- **swlib_component**

  Path to the Software Library to be cloned. "isSwLib" must be true in this case.

- **source_params**

  Source Oracle home info. "isSwLib" must be false in this case.

- **jobdesc**

  Description of the job. If not specified, it is automatically generated.

Examples

```
emcli clone_database_home
      -input_file="dest_properties:clonedestinations"
      -list_exclude_files="*.log,*.dbf,sqlnet.ora,tnsnames.ora,listener.ora"
      -isSwLib="false"
      -isRac="false"
      -tryftp_copy="false"
```

```
      -jobname="clone database home"
      -prescripts="/home/joe/myScript"
      -run_prescripts_as_root="true"
      -rootscripts="%oracle_home%/root.sh"
      -source_params="TargetName:host.domain.com;HomeLoc=/oracle/database1;
HomeName=OUIHome1;ScratchLoc=/tmp"
```

Passing Variables Through EM CLI

When working with variables such as `%perlbin%` or `%oracle_home%`, EM CLI passes variable values from the current local environment instead of the variables themselves. To pass variables through an EM CLI command, as might be the case when using the `-prescripts` or `-postscripts` options, you can place the EM CLI command in a batch file and replace all occurrences of `%` with `%%`.

collect_metric

Performs an immediate collection and threshold evaluation of a set of metrics associated with the specified internal metric name. Metric data collection and threshold evaluation occur asynchronously to the EM CLI call.

You typically use this command when you believe you have resolved an open metric alert or error and would like to clear the event by immediately collecting and reevaluating the metric. This command applies to most metrics except server-generated database metrics.

Use the get_on_demand_metrics verb to see a complete list of supported metrics for a given target.

Format

```
emcli collect_metric
       -target_name=name
       -target_type=type
       -metric_name=metric_name | -collection_name=user_defined_metric_name

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Name of the target.

- **target_type**

  Internal target type identifier, such as host, oracle_database, and emrep.

- **metric_name**

  Internal name that represents a set of metrics that are collected together. Use the get_on_demand_metrics verb to see the supported list of metrics for a given target.

- **collection_name**

  Name of the user-defined metric or SQL user-defined metric. This parameter only applies to user-defined metrics and SQL user-defined metrics.

Examples

Example 1

If you want to collect the "CPU Utilization (%)" metric, look for the appropriate metric internal name (which is Load) using the get_on_demand_metrics command, then run the command as follows:

```
emcli collect_metric  -target_type=host  -target_name=hostname.example.com
-metric_name=Load
```

Example 2

This example immediately collects and evaluates thresholds for the user-defined metric called MyUDM:

```
emcli collect_metric -target_type=host -target_name=hostname.example.com
-collection=MyUDM
```

Example 3

This example immediately collects and evaluates thresholds for the SQL user-defined metric called MySQLUDM:

```
emcli collect_metric -target_type=oracle_database -target_name=database
-collection=MySQLUDM
```

compare_sla

Compares two SLAs as defined by two XML files. This utility outputs the difference trees as sla1_compare.dif and sla2_compare.dif in the specified directory. You can use a diff utility to diff these two files. Compare two sla.xml's to find out the difference.

Format

```
emcli compare_sla
    -input_file=sla1:'first_xml
    -input_file=sla2:'second_xml'
    [-dir='directory']

[ ] indicates that the parameter is optional
```

Parameters

- **input_file=sla1**

  File name for the first XML file.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=sla2**

  File name for the second XML file.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **dir**

  The default is the current directory. If you need to specify another directory, use this option for the output files sla1_compare.dif and sla2_compare.dif.

Example

This example compares two SLAs as defined in sla1.xml and sla2.xml, and outputs sla1_compare.dif and sla2_compare.dif in the current directory.

```
emcli compare_sla
         -input_file=sla1:sla1.xml -input_file=sla2:sla2.xml
```

You can use a standard diff tool to diff the files, such as This example for Linux:

```
diff sla1_compare.dif sla2_compare.dif
```

config_compare

Submits the configuration comparison job.

Format

```
emcli config_compare
      -target_type="oracle_database"
      -first_config="Test Database"
      -second_config="SYSMAN"
      -job_name="Test Compare Job"
      [-schedule=
      {
          start_time:yyyy/MM/dd HH:mm;
          tz:{java timezone ID};
          frequency:interval/weekly/monthly/yearly;
          repeat:#m|#h|#d|#w;
          months:#,#,...;
          days:#,#,...;
          end_time:yyyy/MM/dd HH:mm;
          grace_period:;
       }]
       [-template_id="18"]
       [-job_description="Test Description"]
       [-mapping_display="Tree"]
       [-email_address]
       [-save_mode="save_all|save_only_diffs"]

[ ] indicates that the parameter is optional
```

Parameters

- target_type

  Target type on which the comparison job is being submitted. The value should be the internal name. To get the internal name, execute the following EM CLI command:

  ```
  emcli get_target_types
  ```

- first_config

  Name of the first configuration, which can be either the latest configuration or a saved configuration of a target. If submitting the latest configuration, provide the target name. If submitting a saved configuration, the format should be:

  ```
  "target_name|saved_configuration_name(which is the "name" field from the output
  of "emcli get_saved_configs"
  ```

- second_config

  Names of the second and subsequent configurations, which can contain one or more latest configurations and/or one or more saved configurations of one or more targets. Multiple configurations can be specified, separated by a comma. If the latest configuration needs to be submitted, provide the target name. If the saved configuration needs to be submitted, then the format should be:

  ```
  "target_name|saved_configuration_name(which is the "name" field from the output
  of "emcli get_saved_configs"
  ```

- job_name

Name of the comparison job.

- schedule

  Schedule with which the comparison job must be scheduled. If the schedule option is not provided, the comparison job runs immediately.

  – start_time - Time when comparison job has to start executing. The format is "yyyy/MM/dd HH:mm"

    tz - Timezone ID (optional)

  – frequency - Valid values are once/interval/weekly/monthly/yearly. (optional)

    If frequency is set to interval, repeat must be specified.

    If frequency is set to weekly or monthly, days must be specified.

    If frequency is set to yearly, both days and months must be specified.

  – repeat - Frequency with which the comparison job must be repeated. (Required only if frequency is set to interval.)

  – days - Comma-separated list of days. (Required only if frequency is weekly, monthly, or yearly.) Example: "repeat=1d"

    If frequency is weekly, then the valid range is 1 to 7 inclusive.

    If frequency is monthly or yearly, then the valid range is 1 to 30 inclusive.

  – months - Comma-separated list of months. (Required only if frequency is yearly). Valid range is 1 to 12 inclusive.

  – end_time - End time for comparison job executions. (optional). If it is not specified, the comparison job runs indefinitely. The format is "yyyy/MM/dd HH:mm"

  – grace_period - grace period in minutes(optional)

- template_id

  ID of the template. The value is an integer.

- job_description

  Description of the comparison job.

- mapping_display

  Can be either "tree" or "table." The default value is "tree". This option is only for composite targets. Note: When "template_id" is specified, do not specify mapping_display.

- email_address

  Email address to which notification mail is to be sent, if differences are found.

- save_mode

  Tells the comparison engine whether to save all the results or only the differences. Valid inputs are "save_all" and "save_only_diffs". The save_only_diffs option saves the differences to the Management Repository. Otherwise, all the comparison results are saved. The default value is "save_only_diffs".

Checking the Job Status:

Once submitted, the comparison job's status can be viewed by issuing the following EM CLI command:

```
emcli get_jobs -name="jobName"
```

Aborting the Job:

Once submitted, the comparison job can be aborted by issuing the following EM CLI command:

```
emcli stop_job -name="jobName"
```

Examples

Example 1

This example compares the latest configuration of one target to the latest configurations of multiple targets. All the comparison results will be saved.

```
emcli config_compare
      -target_type="oracle_database"
      -first_config="Test Database"
      -second_config="Test Database","Test_Database"
      -job_name="Test Job" -template_id="18"
      -save_mode="save_all"
```

Example 2

This example compares the latest configuration with the saved configuration specifying a start_time.

```
emcli config_compare
      -target_type="oracle_database"
      -first_config="Test Database"
      -second_config="Test Database|Test Database|oracle_
         database|20140101224530","Test_Database" -job_name="Test Job"
      -schedule="start_time:2014/06/10 15:45"
```

Example 3

This example compares the latest configuration with the latest configuration specifying a repeating frequency of 1 day.

```
emcli config_compare -target_type="oracle_database"
      -first_config="Test Database"
      -second_config="Test Database","Test_Database"
      -job_name="Test Job"
      -schedule="start_time:2014/10/29 2:00;frequency:interval;repeat:1d"
```

Example 4

This example compares the latest configuration with the latest configuration specifying a repeating frequency of 1 week.

```
emcli config_compare
      -target_type="oracle_database"
      -first_config="Test Database"
      -second_config="Test Database","Test_Database"
      -job_name="Test Job"
      -schedule="start_time:2014/08/10 01:00;frequency:interval;repeat:1w"
```

Example 5

This example compares the latest configuration to the latest configuration specifying a repeating frequency of Saturday and Sunday.

```
emcli config_compare
      -target_type="oracle_database"
      -first_config="Test Database"
      -second_config="Test Database","Test_Database"
      -job_name="Test Job"
      -schedule="start_time:2014/08/10 01:00;frequency:weekly;
          days:6,7;grace_period:60;tz:America/New_York"
```

config_db_service_target

Creates a Database as a Service (DBaaS) target for Oracle Public Cloud.

Format

```
emcli config_db_service_target
      -database_unique_name="database unique name"
      -service_grp_name="service group name"
      -cloud_service_name="cloud service name"
      -operation="operation to be performed"
      -schema_name="schema name"
      -tablespace_name="tablespace name"
      [-subscription_id="subscription ID"]
      [-customer_name="customer name"]
      [-csi_number="CSI number"]
      [-connection_service_name="connection service name"]
      [-cloud_service_version="cloud service version"]
      [-l_o_b="line of business"]
```

Parameters

- **database_unique_name**

  Identifies the `DBName` property of a database target on which the DBaaS target will be based.

- **service_grp_name**

  Identifies the DBaaS target service group name. The DBaaS target name will be `service_grp_name_cloud_service_name`.

- **cloud_service_name**

  Identifies the Oracle Public Cloud service name.

- **operation**

  Identifies the operation to be performed on the DBaaS target (for example, "create").

- **schema_name**

  Identifies the name of the schema associated with the DBaaS target.

- **tablespace_name**

  Identifies the name of the tablespace associated with the DBaaS target.

- **subscription_id**

  Provides a value for the Cost Center property of a DBaaS target.

- **customer_name**

  Provides a customer point of contact for the DBaaS target.

- **csi_number**

  Identifies the Customer Support Identifier (CSI) of the DBaaS target.

- **connection_service_name**

  Identifies the name of the Database Service associated with the DBaaS Target.

- **cloud_service_version**

Shows the Oracle Public Cloud Service version of the DBaaS target.

- l_o_b

  Identifies the Line of Business (LOB) of the DBaaS target.

Exit Codes

```
0 On success
Non-zero value means verb processing was not successful.
```

Examples

Example 1

Creates a new Database as a Service (DBaaS) target (`db_serv1`). Specifies the schema, tablespace name, and connection service name. The new DBaaS target is based on a Database target using `db.example.com` as the `DBName` property:

```
emcli config_db_service_target
      -database_unique_name="db.example.com"
      -service_grp_name="db"
      -cloud_service_name="serv1"
      -operation="Create"
      -schema_name="HR"
      -tablespace_name="SYSTEM"
      -connection_service_name="nservice1"
```

configure_log_archive_locations

Configures Log Archive Locations for the root target and its children. To configure Log Archive Location for a target, you should know the configuration parameters, like host name, from where the log archive files are accessible, the credentials to access the host, and the location of the log archive files.

Format

```
emcli configure_log_archive_locations
      -root_target_name="<target_name>"
      -root_target_type="<target_type>"
      [-archive_config_file="<target_archive_config_file_location>"]
      [-no_update]
      [-debug]

[ ] indicates that the parameter is optional.
```

Parameters

- root_target_name

  Name of the root target. A configurable tree target hierarchy will be created with this root target. Example root targets are WebLogic Domain and Fusion Application Instance.

- root_target_type

  Target type of the root target name.

- archive_config_file

  Location of the archive config file. Every line in this file should contain the following 7 fields in the same order.

  target_name,target_type,host_target_name,host_cred_type,host_cred_name or new_cred_user_name,new_cred_password,archive_dir_location

  – target_name

    If this target is part of the root hierarchy, then this target and it's children will be updated with the archive parameters specified.

  – target_type

    Target type of the above target.

  – host_target_name

    Host name from where archive location is accessible. The Management Agent monitoring this target should have Oracle Fusion Middleware plug-in release 12.1.0.4 or later installed.

  – host_cred_type

    Credential type. Possible values are preferred_credentials or named_ credentials or new_credentials.

  – host_cred_name

    Credential set name for preferred_credentials type or Named credential name for named_credentials type.

  – new_cred_user_name

New credential user name for new_credentials type.

- new_cred_password

  New credential password for new_credentials type.

- archive_dir_location

  Directory location where log archive files available.

- no_update

  If this flag is provided, targets which are already configured with the archive properties, will not be updated again.

- debug

  Runs the verb in verbose mode for debugging purposes.

Example

The following example configures Log Archive Locations for Fusion Instance target and its children.

```
emcli configure_log_archive_locations
        -root_target_name=fa1
        -root_target_type=fusion_apps_instance
        -archive_config_file=/scratch/config.txt
```

Sample Archive Config File:

In the case of new credentials:

```
fa1,fusion_apps_instance,adc123.oracle.com,new_credentials,user1,pwd1,/scratch/fa1
```

In the case of preferred credentials:

```
fa1,fusion_apps_instance,adc123.oracle.com,preferred_credentials,credential_
set1,,/scratch/fa1
```

(Because this is preferred credentials, ',,' means the new_cred_password field is not valid and therefore skipped.)

configure_siteguard_lag

Configures the limit of Apply lag and Transport lag for all or selected databases of the system.

Format

```
emcli configure_siteguard_lag
        [-system_name="name_of_the_system"]
        [-target_name="name_of_the_target_database"]
        [-property_name="lag_type"]
        [-value="max_limit_in_seconds"]

[ ] indicates that the parameter is optional.
```

Parameters

- **system_name**

  Name of the system on which lag limits need to be configured.

- **target_name**

  Name of the database on which lag limits need to be configured.

- **property_name**

  Name of the lag property to be configured. Valid values are `ApplyLag` and `TransportLag`.

- **value**

  Limit of the lag. These values are specified in seconds.

Examples

Example 1

This example configures the Apply lag limit of 1000 seconds on all of the databases of `austin-system`:

```
emcli configure_siteguard_lag
        -system_name="austin-system"
        -property_name="ApplyLag"
        -value="1000"
```

Example 2

This example configures the Transport lag limit of 2500 seconds on the database `OID-db` of `austin-system`:

```
emcli configure_siteguard_lag
        -system_name="austin-system"
        -target_name="OID_db"
        -property_name="TransportLag"
        -value="2500"
```

confirm_instance

Confirms a manual step. An instance cannot be confirmed when its status is suspended, stopped, completed, or completed with an error.

Format

```
emcli confirm_instance
      [-instance=<instance_guid>]
      [exec=<execution_guid>]
      [-name=<execution name>]
      [-owner=<execution owner>]
      -stateguid=<state_guid>

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

- **stateguid**

  Comma-separated list of state GUIDs.

Examples

```
emcli confirm_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid=51F762417C4943DEE040578C4E087168

emcli confirm_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid='51F762417C4943DEE040578C4E087168,51F762417C4944DEE040578C4E087168'
```

continue_add_host

Performs resume/continue operations of a previously submitted add host session that has failed at some phase.

Format

```
emcli continue_add_host
        -session_name="session_name"
        -continue_all_hosts | -continue_ignoring_failed_hosts"
        [-wait_for_completion]

[ ] indicates that the parameter is optional
```

Parameters

- **session_name**

  Name of the session you want to continue to the next phase of Agent deployment.

- **continue_all_hosts**

  Continues the session on all hosts, including those on which the current deployment phase failed.

- **continue_ignoring_failed_hosts**

  Continues the session for only the hosts on which the current deployment phase succeeded.

- **wait_for_completion**

  Specifies whether the command should run in synchronous or asynchronous mode. If you specify this option (for synchronous mode), the command waits until the add host session completes before returning control to you on the command line.

Examples

Example 1

This example continues the session 'ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_ AM_PST' to the next phase of deployment on all hosts.

```
emcli continue_add_host -session_name='ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_AM_PST'
-continue_all_hosts
```

Example 2

This example continues the session 'ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_ AM_PST' synchronously to the next phase of deployment only on hosts on which the current phase was successful.

```
emcli continue_add_host -session_name='ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_AM_PST'
-continue_ignoring_failed_hosts -wait_for_completion
```

convert_to_cluster_database

Converts a single-instance database to a Real Application Cluster (RAC) database.

Format

```
emcli convert_to_cluster_database
      -sourceTargetName="Single instance database target to be converted to RAC"
      -sysdbaCreds="Named credentials for SYSDBA user"
      -hostCreds="Named credentials for Host"
      [-newOracleHome="RAC Oracle Home, if moving to differnt home"]
      [-racConfigType="ADMIN_MANAGED | POLICY_MA NAGED"]
      [-nodeList="Comma-separated node names for Admin Managed RAC database"]
      [-serverPoolList="Comma-separated list of server pools for Policy Managed
         database"]
      [-databaseArea="Shared storage location for database files"]
      [-recoveryArea="Shared storage location for recovery files"]
      [-listenerPort="RAC Listener port"]

[ ] indicates that the parameter is optional
```

Parameters

- **sourceTargetName**

  Enterprise Manager target name of the single-instance database to be converted to a RAC database. Database versions 10.2.0.1.0 and above are supported for conversion. The single-instance database target should exist on one of the nodes of the cluster where the RAC database will be created, and the cluster should be an Enterprise Manager target.

- **sysdbaCreds**

  Named database credentials with SYSDBA privileges on the database to be converted to a RAC database.

- **hostCreds**

  Named host credentials of the user who owns the Oracle home installation.

- **newOracleHome**

  RAC Oracle home location of the converted database. You only need to provide this if different from the Oracle home of the single-instance database to be converted.

- **racConfigType**

  RAC configuration type. Valid values are POLICY_MANAGED and ADMIN_MANAGED. POLICY_MANAGED is valid only for database versions 11.2 or higher. The default is ADMIN_MANAGED if not provided.

- **nodeList**

  List of valid node names for an ADMIN_MANAGED RAC database. It should include the node where the single instance database to be converted exists. If not provided, all the nodes in the cluster are used.

- **serverPoolList**

  Comma-separated list of server pool names for a POLICY_MANAGED RAC database. Applicable only for database versions 11.2 or higher.

- **databaseArea**

  New location for data files of the RAC database. This location should be shared across the nodes of the cluster. It can either be a Cluster File System location or an Automatic Storage Management diskgroup. If not specified, the existing database files should already be on shared storage, and files are not moved during RAC conversion.

- **recoveryArea**

  Fast recovery area location of the RAC database. This location should be shared across the nodes of the cluster. It can either be a Cluster File System location or an Automatic Storage Management diskgroup. If not specified, the existing recovery area location should already be on shared storage, and it does not change during RAC conversion.

- **listenerPort**

  Port of the new RAC listener to be created for the new RAC database. If not provided, the existing listener is used. This option is only applicable to 10.2 and 11.1 database versions. For 11.2 or higher database versions, this value is ignored and the RAC database is always registered with the existing listener in the Cloud Infrastructure home.

Examples

Example 1

```
emcli convert_to_cluster_database -sourceTargetName=sidb
 -sysdbaCreds=sysCreds  -hostCreds=hostCreds racConfigType=ADMIN_MANAGED
```

Example 2

```
emcli convert_to_cluster_database -sourceTargetName=sidb
 -sysdbaCreds=sysCreds  -hostCreds=hostCreds racConfigType=POLICY_MANAGED
 -serverPoolList=sp1,sp2 -databaseArea=+DATA -recoveryArea=+RECOVERY
```

Example 3

```
emcli convert_to_cluster_database -sourceTargetName=sidb
 -sysdbaCreds=sysCreds  -hostCreds=hostCreds -nodeList=node1,node2
 -databaseArea=/u01/share/oradata -recoveryArea=/u01/share/fra -listenerPort=1525
```

create_aggregate_service

Defines an aggregate service: name and its sub-services. After the aggregate service is created, you can edit it from the Enterprise Manager Cloud Control console to configure performance and usage metrics to be collected and displayed.

Format

```
emcli create_aggregate_service
      -name='name'
      -type='type'
      -availType=SUB-SERVICE|SYSTEM|TESTS
      -add_sub_services="name1:type1;name2:type2;..."
      -avail_eval_func=and|or
      [-hostName=<host_name>
      [-agentURL=<agent_url>
      [-properties='pname1|pval1;pname2|pval2;...']
      [-timezone_region=<gmt_offset>]
      [-systemname=<system_name>]
      [-systemtype=<system_type>]
      [-keycomponents='keycomp1name:keycomp1type;keycomp2name:keycomp2type;...']
      [-beacons='bcn1name:bcn1isKey;bcn2name:bcn2isKey;...']
      [-input_file='template:Template_file_name;[vars:Variables_file_name]']
      [-sysAvailType=<availability_type>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Aggregate service name.

- **type**

  Aggregate service type.

- **availType**

  Sets availability to either sub-service, system-based, or test-based. Valid values are SUB-SERVICE, SYSTEM, and TESTS.

  If availability is set to SYSTEM, -systemname and -systemtype are required.

  If availability is set to TESTS, -beacons, template file, and variables are required.

- **add_sub_services**

  Sub-services list.

- **avail_eval_func**

  Operator to evaluate availability. If "and" is used, all sub-services, tests, and system-components must be up in order for this aggregate_service to be up.
  If "or" is used, only one of the sub-services, tests, and system-components needs to be up for this aggregate_service to be up.

- **hostName**

  Network name of the system running the Management Agent that is collecting data for this target instance.

- **agentURL**

URL of the Management Agent that is collecting data for this target instance. If you enter the host name, the Agent URL of the host is automatically entered in this field.

- **properties**

  Name-value pair (that is, prop_name | prop_value) list of properties for the service instance.

- **timezone_region**

  Accepts either long formats ("America/Los Angeles") or short formats ("PST"). If you do not provide a time zone, the default OMS time zone is used.

- **systemname**

  System name on which service resides.

- **systemtype**

  Use emcli get_targets to obtain the system type for the system name.

- **keycomponents**

  Name-type pair (that is, `keycomp_name:keycomp_type`) list of key components in the system that are used for the service.

- **beacons**

  Name-isKey pairs that describe the beacons of the service. If isKey is set to `Y`, the beacon is set as a key-beacon of the service. The service should have at least one key beacon if the availability is set to test-based.

- **input_file**

  Template file name is the XML file that includes the template definition. Variable file defines the values for the template.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

  See below for an example of an XML file for this parameter.

- **sysAvailType**

  Type of availability when the availType is system-based. Sets the availability to either SYSTEM_TARGET_DIRECTLY or SELECTED_COMPONENTS_OF_A_SYSTEM.

  If availability is set to SYSTEM_TARGET_DIRECTLY, the system needs to have availability[status] defined. -systemname and -systemtype are required parameters.

  If availability is set to SELECTED_COMPONENTS_OF_A_SYSTEM, -systemname, -systemtype and -keycomponents are required parameters.

  If availability is set to SYSTEM_TARGET_DIRECTLY and if the system does not have availability[status] defined, the availability set is invalid. Therefore, the only option that can be set is SELECTED_COMPONENTS_OF_A_SYSTEM.

Examples

```
emcli create_aggregate_service -name="My_Name"
          -type="aggregate_service"
          -add_sub_services="sub1:type1;sub2:type2"
          -avail_eval_func="and"
```

```
                -availType="SUB_SERVICE"
                 -properties="prop1|value1;prop2|value2"
                -timezone_region="PST"

 emcli create_aggregate_service -name="My_Name"
                -type="aggregate_service"
                -add_sub_services="sub1:type1;sub2:type2"
                -avail_eval_func="or"
                -availType="SYSTEM"
                -systemname="my system" -systemtype="generic_system"
                -sysAvailType="SYSTEM_TARGET_DIRECTLY"
                -timezone_region="PST"

 emcli create_aggregate_service -name="My_Name"
                -type="aggregate_service"
                -add_sub_services="sub1:type1;sub2:type2"
                -avail_eval_func="and"
                -availType="SYSTEM"
                -systemname="my system" -systemtype="generic_system"
                -sysAvailType="SELECTED_COMPONENTS_OF_A_SYSTEM"
                -keycomponents="database:oracle_database;
                 mytestbeacon:oracle_beacon"
                -timezone_region="PST"

emcli create_aggregate_service -name="My_Name"
                -type="aggregate_service"
                -add_sub_services="sub1:type1;sub2:type2"
                -avail_eval_func="or"
                -timezone_region="PST"
                -availType="TESTS"
                -beacons="MyBeacon:Y;MyOtherBeacon:N"
                -properties="prop1|value1;prop2|value2"
                -input_file="template:mytests.xml"
                -input_file="variables:myvariable.xml"
```

XML for input_file Example

The following sample XML file creates a service test of name 'EM Console Service Test' and of type 'Web Transaction'. It defines some properties, such as readTimeout, Collection Interval, and so forth under the <properties> section, which are related to this service test.

This service test has defined step information under <mgmt_bcn_step_with_props>. The name of the step is '1.Access Logout page'. The URL to be monitoried under this step is https://myhost.in.domain.com:14513/em/console/logon/logoff?event=load, which is defined under the properties section of the step.

This XML file also defines some threshold levels for this service test on the transaction level under <txn_thresholds>. For the metric avg_response_time, it states that if the metric value is greater than 6000.0, raise a warning alert, and if the metric value is greater than 12000.0, raise a critical alert.

```
<?xml version = '1.0' encoding = 'UTF-8'?> <transaction-template template_
type="aggregate_service" xmlns="template">
<variables>
<variable name="HOST1" value="myhost.in.domain.com"/>
<variable name="PORT1" value="14513"/>
<variable name="PROTOCOL1" value="https"/>
</variables>
<transactions>
<mgmt_bcn_transaction>
```

```
<mgmt_bcn_txn_with_props>
<mgmt_bcn_txn description="Test for checking the availability of EM
Console/Website" is_representative="true" name="EM Console Service Test"
monitoring="true" txn_type="HTTP"/>
<properties>
<property name="readTimeout" num_value="120000.0" prop_type="2" encrypt="false"/>
<property name="certValidationMode" string_value="1" prop_type="1"
encrypt="false"/>
<property name="maxDownloadSize" num_value="1.0E8" prop_type="2" encrypt="false"/>
<property name="sensitiveValuesProtection" string_value="0" prop_type="1"
encrypt="false"/>
<property name="failureStringModes" string_value="regularText" prop_type="1"
encrypt="false"/>
<property name="UserAgent" string_value="Mozilla/4.0 (compatible; MSIE 6.0;
Windows NT 5.1) OracleEMAgentURLTiming/3.0" prop_type="1" encrypt="false"/>
<property name="successStringModes" string_value="regularText" prop_type="1"
encrypt="false"/>
<property name="variablesModes" string_value="urlEncode" prop_type="1"
encrypt="false"/>
<property name="content" string_value="0" prop_type="1" encrypt="false"/>
<property name="AcceptLanguage" string_value="en" prop_type="1" encrypt="false"/>
<property name="connectionTimeout" num_value="120000.0" prop_type="2"
encrypt="false"/>
<property name="useCache" string_value="yes" prop_type="1" encrypt="false"/>
<property name="stringValidationMode" string_value="1" prop_type="1"
encrypt="false"/>
<property name="granularity" string_value="transaction" prop_type="1"
encrypt="false"/>
<property name="numThreads" num_value="4.0" prop_type="2" encrypt="false"/>
<property name="retries" num_value="1.0" prop_type="2" encrypt="false"/>
<property name="timeout" num_value="300000.0" prop_type="2" encrypt="false"/>
<property name="retryInterval" num_value="5000.0" prop_type="2" encrypt="false"/>
</properties>
<per_bcn_properties/>
</mgmt_bcn_txn_with_props>
<steps_defn_with_props>
<mgmt_bcn_step_with_props>
<mgmt_bcn_step step_number="1" name="1.Access Logout page" step_type="HTTP"/>
<properties>
<property name="req_mode" num_value="1.0" prop_type="2" encrypt="false"/>
<property name="http_method" string_value="G" prop_type="1" encrypt="false"/>
<property name="url" string_
value="{PROTOCOL1}://{HOST1}:{PORT1}/em/console/logon/logoff?event=load" prop_
type="1" encrypt="false"/>
</properties>
</mgmt_bcn_step_with_props>
</steps_defn_with_props>
<stepgroups_defn/>
<txn_thresholds>
<mgmt_bcn_threshold warning_threshold="6000.0" warning_operator="0" critical_
threshold="12000.0" critical_operator="0" num_occurrences="1">
<mgmt_bcn_threshold_key metric_name="http_response" metric_column="avg_response_
time"/>
</mgmt_bcn_threshold>
<mgmt_bcn_threshold warning_threshold="0.0" warning_operator="1" critical_
threshold="0.0" critical_operator="1" num_occurrences="1">
<mgmt_bcn_threshold_key metric_name="http_response" metric_column="status"/>
</mgmt_bcn_threshold>
</txn_thresholds>
<step_thresholds/>
```

```
<stepgroup_thresholds/>
</mgmt_bcn_transaction>
</transactions>
</transaction-template>
```

create_assoc

Creates target association instances.

Format

Standard Mode

```
emcli create_assoc
       -assoc_type="association type"
       -source="source"
       -dest="destination_target"
       [-separator="separator:attribute_name:character"]
       [-subseparator="subseparator:attribute_name:character"]
```

Interactive (Script) Mode

```
create_assoc(
      assoc_type="association type"
      ,source="source"
      ,dest="destination_target"
      [,separator="separator:attribute_name:character"]
      [,subseparator="subseparator:attribute_name:character"]
      )
```

`[ ]` indicates that the parameter is optional.

Parameters

- **source_type**

  Source target type.

- **source**

  Source target.

- **dest**

  Destination target.

- **separator**

  By default, multi-value input attributes use a semicolon (;) as a separator. Specifying this option overrides the default separator value.

  Example: `separator="<attribute_name=sep_char>"` where *attribute_name* is name of the attribute for which you want to override the separator character, and *sep_char* is the new separator character.

  Example: `separator="att=#"`

- **subseparator**

  By default, multi-value input attributes use a colon (:) as a subseparator. Specifying this option overrides the default subseparator value.

  Example: `subseparator="<attribute_name=sep_char>"` where *attribute_name* is name of the attribute for which you want to override the separator character and *sep_char* is the new subseparator character.

  Example: `separator="att=#"`

Note: The name and owner parameters must be used together.

Output

Exit Codes

0 indicates that the verb processing was successful.

Non-zero values indicate that the verb processing was not successful.

Example

This example creates an association of type cluster_contains from target "abc_cluster:cluster" to targets "def.oracle.com:host" and "ghi.oracle.com:host":

```
emcli create_assoc
       -assoc_type="cluster_contains"
       -source="abc_cluster:cluster"
       -dest="def.oracle.com:host;ghi.oracle.com:host"
```

For a list of allowed pairs, enter `emcli list_allowed_pairs` .

create_blackout

Creates a scheduled blackout to suspend any data collection activity on one or more monitored targets.

Format

```
emcli create_blackout
      -name="name"
      -add_targets="name1:type1;name2:type2;..."...
      -reason="reason"
      [-description="description"]
      [-jobs_allowed]
      [-propagate_targets]
      -schedule=
         frequency:once|interval|weekly|monthly|yearly];
         duration:[HH...][:mm...];
         [start_time:yy-MM-dd HH:mm];
         [end_time:yy-MM-dd HH:mm];
         [repeat:#m|#h|#d|#w];
         [months:#,#,...];
         [days:#,#,...];
         [tzinfo:specified|target|repository]
         [tzoffset:#|[-][HH][:mm]]
         [tzregion:...]

[ ] indicates that the parameter is optional
```

Constraints on schedule arguments:

```
frequency:once
    requires => duration or end_time
    optional => start_time, tzinfo, tzoffset
frequency:interval
    requires => duration, repeat
    optional => start_time, end_time, tzinfo, tzoffset
frequency:weekly
    requires => duration, days
    optional => start_time, end_time, tzinfo, tzoffset
frequency:monthly
    requires => duration, days
    optional => start_time, end_time, tzinfo, tzoffset
frequency:yearly
    requires => duration, days, months
    optional => start_time, end_time, tzinfo, tzoffset
```

Parameters

- **name**

  Name of the blackout to create.

- **add_targets**

  Targets to add to the blackout, each specified as `target_name:target_type`. You can specify this parameter more than once.

- **reason**

  Reason for the blackout. If you have `SUPER_USER` privileges (you are an Enterprise Manager Super Administrator), any text string can be used for the reason. The

reason is added to the list of allowable blackout reasons if it is not already in the list. If you do not have `SUPER_USER` privileges, you must specify one of the text strings returned by the `get_blackout_reasons` verb.

- **description**

  Description or comments pertaining to the blackout. The description, limited to 2000 characters, can be any text string.

- **jobs_allowed**

  When you specify this option, jobs are allowed to run against blacked-out targets during the blackout period. If you do not specify this option, jobs scheduled to be run against these targets are not allowed to run during the blackout period. After a blackout has been created, you cannot change the "allowed jobs" from either EM CLI or the Enterprise Manager Cloud Control console.

- **propagate_targets**

  When you specify this option, a blackout for a target of type "host" applies the blackout to all targets on the host, including the Agent. This is equivalent to nodelevel in the emctl command. Regardless of whether you specify this option, a blackout for a target that is a composite or a group applies the blackout to all members of the composite or group.

- **schedule**

  Blackout schedule. Note that the "frequency" argument determines which other arguments are required or optional.

- **schedule=frequency**

  Type of blackout schedule (default is "once").

- **schedule=duration**

  Duration in hours and minutes of the blackout (-1 means indefinite). Hours and minutes each can be up to 6-digits long.

- **schedule=start_time**

  Start date/time of the blackout. The default value is the current date/time. The format of the value is "yy-MM-dd HH:mm", for example: "2003-09-25 18:34"

- **schedule=end_time**

  Last date/time of the blackout. When "frequency" is weekly, monthly, or yearly, only the date portion is used. When "frequency" is interval or once, the date and time are taken into account. The format of the value is "yy-MM-dd HH:mm"; for example: "2003-09-25 18:34"

- **schedule=repeat**

  Time between successive start times of the blackout. The letter following the number value represents the time units: "m" is minutes, "h" is hours, "d" is days, and "w" is weeks.

- **schedule=months**

  List of integer month values in the range 1-12. Each value must have a corresponding "day" value to fully specify (month, day) pairs that indicate the blackout starting days of the year.

- **schedule=days**

When "frequency" is weekly, this is a list of integer day-of-week values in the range 1-7 (1 is Sunday). When "frequency" is monthly, this is a list of integer day-of-month values in the range 1-31 or -1 (last day of the month). When "frequency" is yearly, this is a list of integer day-of-month values in the range 1-31 or -1 (last day of the month); in this case, the month is taken as the corresponding "month" value for each (month, day) pair.

- **schedule=tzinfo**

  Type of timezone. The `tzinfo` argument is used in conjunction with `tzoffset`. Available timezone types are: "specified" (offset between GMT and the target timezone), "target" (timezone of the specified target), and "repository" (repository timezone -- default setting when tzinfo is not specified). See `-schedule=tzoffset` for more information.

- **schedule=tzoffset**

  Value of the timezone. When the `tzinfo` argument is not specified or is "repository", the timezone value is the repository timezone. In this case, the `tzoffset` argument must not be specified. Otherwise, the `tzoffset` argument is required. When `tzinfo` is set to "specified", the `tzoffset` argument specifies the offset in hours and minutes between GMT and the timezone. When `tzinfo` is set to "target", the `tzoffset` argument specifies an integer index (the first is 1) into the list of targets passed as arguments. For example, for a `tzoffset` setting of 1, the timezone of the first target specified in the `-add_targets` parameter is used.

  Note that the timezone is applied to the start time and the end time of the blackout periods. The timezones associated with each target are not taken into account when scheduling the blackout periods (except that when `tzinfo` is set to "target", the specified target's timezone is used for the blackout times).

- **schedule=[tzregion:<...>]**

  Time zone region to use. When you "specify" the tzinfo parameter, this parameter determines which timezone to use for the blackout schedule. Otherwise, it is ignored. It defaults to "GMT".

Examples

Example 1

This example creates blackout `b1` for the specified target (`database2`) to start immediately and last for 30 minutes.

```
emcli create_blackout -name=b1 -add_targets=database2:oracle_database
      -schedule="duration::30"
      -reason="good reason1"
```

Example 2

This example creates blackout `b1` for all targets on `myhost` to start immediately and last until 2007-04-26 05:00 (in the timezone America/New_York).

```
emcli create_blackout -name=b1 -add_targets=myhost:host
      -propagate_targets -jobs_allowed
      -schedule="end_time:2007-04-26 05:00;tzinfo:specified;
        tzregion:America/New_York"
      -reason="good reason2"
```

Example 3

This example creates blackout `b1` for all targets in group `mygroup` to start immediately and last until 2007-04-26 05:00 (in the timezone America/New_York). No jobs are allowed to run during the blackout.

```
emcli create_blackout -name=b1 -add_targets=mygroup:group
      -schedule="end_time:2007-04-26 05:00;tzinfo:specified;
        tzregion:America/New_York"
      -reason="good reason3"
```

Example 4

This example creates blackout `b1` for the specified targets (`database2` and `database3`) to start at 2007-08-24 22:30 and last for 30 minutes. The timezone is the timezone for the `database2` target.

```
emcli create_blackout -name=b1
      -add_targets="database2:oracle_database;database3:oracle_database
      -schedule="frequency:once;start_time:07-08-24
22:30;duration::30;tzinfo:target:tzoffset:1"
      -reason="good reason4"
```

Example 5

This example creates blackout `b1` for the specified targets (`database2` and `database3`) to start at 2007-08-24 22:30 and last for 30 minutes. The timezone is the timezone for the `database3` target.

```
emcli create_blackout -name=b1 -add_targets=database2:oracle_database
      -add_targets=database3:oracle_database
      -schedule="frequency:once;start_time:07-08-24
22:30;duration::30;tzinfo:target;tzoffset:2"
      -reason="good reason5"
```

Example 6

This example creates blackout `b2` for the specified target (`database2`) to start at 2007-08-25 03:00 and every day thereafter, and to last 2 hours each time. The timezone is the repository timezone.

```
emcli create_blackout -name=b2 -add_targets=database2:oracle_database
      -schedule="frequency:interval;start_time:2007-08-25
03:00;duration:2;repeat=1d"
      -reason="good reason"
```

Example 7

This example creates blackout `b2` for the specified target (`database2`) to start immediately and every 2 days thereafter (until 06-12-31 23:59), and to last 2 hours 5 minutes each time. The timezone is the repository timezone.

```
emcli create_blackout -name=b2 -add_targets=database2:oracle_database
      -schedule="frequency:interval;duration:2:5;end_time:06-12-31
23:59;repeat=2d;tzinfo:repository"
      -reason="another good reason"
```

Example 8

This example creates blackout `b4` for all targets on `myhost` and `otherhost` to start every Sunday through Thursday at the current time. The blackout will last 1 hour each time.

```
emcli create_blackout -name=b4 -add_targets="myhost:host;otherhost:host"
      -propagate_targets
```

```
      -schedule="frequency:weekly;duration:1:;days:1,2,3,4,5"
      -reason="very good reason"
```

Example 9

This example creates blackout `b5` for all targets within group `mygroup` to start on the 15th and last day of each month at time 22:30 and last until 2011-11-24 (2011-11-15 will be the actual last blackout date). The blackout will last 1 hour 10 minutes each time. Jobs are allowed to run during the blackouts.

```
emcli create_blackout -name=b5 -add_targets=mygroup:group
      -propagate_targets -jobs_allowed
      -schedule="frequency:monthly;duration:1:10;start_time:06-10-24 22:30;
       end_  time:06-12-24 23:59:days:15,-1"
      -reason="pretty good reason"
```

Example 10

This example creates blackout `b6` for the specified target (`database2`) to start at 13:30 on the following dates of each year: 03-02, 04-22, 09-23. The blackout will last 2 hours each time. Jobs are not allowed to run during the blackouts.

```
emcli create_blackout -name=b6 -add_targets=database2:oracle_database
      -propagate_targets
      -schedule="frequency:yearly;duration:2;start_time:07-08-24
13:30:months=3,4,9;days:2,22,23"
      -reason="most excellent reason"
```

create_charge_entity_type

Creates a custom entity type for an Enterprise Manager target type for which there is no current Chargeback support. There can be only one custom entity type for the specified Enterprise Manager target type.

Format

```
emcli create_charge_entity_type
        -target_type="target_type"
```

Parameters

- **target_type**

  Name of the custom entity type.

Examples

The following example creates a new Chargeback entity type named oracle_apache for the Enterprise Manager Apache target type:

```
emcli create_charge_entity_type
        -target_type="oracle_apache"
```

create_charge_item

Creates a custom charge item for Chargeback based on the properties specified in the referenced file.

Format

```
emcli create_charge_item
        -input_file="property_file:filename"
```

Parameters

The option [-input_file] is the full path of a file that contains the item properties. The following properties can be defined in the file:

- **target_type**

  Target type to which the charge item applies.

- **source_data_type**

  Source data type. Valid values are metric, config, and property.

- **item_name**

  Name of the item.

- **metric_group**

  Metric group name as listed in `list_item_candidates`. This is a required property if source_data_type=metric.

- **metric_column**

  Metric column name as listed in `list_item_candidates`. This is a required property if source_data_type=metric.

- **config_view**

  Config view name as listed in `list_item_candidates`. This is a required property if source_data_type=config.

- **config_key**

  Config key name as listed in `list_item_candidates`. This is a required property if source_data_type=config.

- **config_column**

  Config column name as listed in `list_item_candidates`. This is a required property if source_data_type=config.

- **config_data_source**

  Data source of configuration metric. This is a required property if source_data_type=config.

- **property**

  Property name as listed in `list_item_candidates`. This is a required property if source_data_type=property.

- **item_displayname**

  Item display name.

- **unit**

Unit display name.

- **aggregation_type**

  Type of aggregation to use for this item. Applicable only if data type=number. Valid values are sum and avg. Default value is avg.

- **is_config_condition**

  Item used conditionally in a charge plan. Valid values are 0, 1. Default value is 0.

- **item_category**

  Category of item. Default value is instance. Valid values are cpu, storage, memory, network, and instance.

- **data_type**

  Valid values are string and number. The default value is string for config and property types, and number for metric type.

Examples

Example 1

This example creates a metric custom charge item that bases charges on the average total of processes on a particular host:

```
emcli create_charge_item -input_file="property_file:/home/user/property_file"
Contents of /home/user/property_file:
       target_type=host
       source_data_type=metric
       item_name=total_proc
       metric_group=Load
       metric_column=noOfProcs
       item_displayname=Total Processes
       unit=process
       aggregation_type=avg
       item_category=instance
       data_type=number
```

Example 2

This example creates a configuration custom charge item that can charge different rates for various usage charge items based on the instance region:

```
emcli create_charge_item -input_file="property_file:/home/user/property_file"
Contents of /home/user/property_file:
       target_type=oracle_database
       source_data_type=config
       item_name=custom_config
       config_view=myCustomCCS
       config_key=region
       config_column=country
       config_data_source=regionList.txt
       item_display_name=Region of Instance
       item_category=instance
       data_type=string
```

create_clone

Creates a new cloned database.

Format

```
emcli create_clone
      -inputFile="File containing properties required for cloning a database"
```

Parameters

- inputFile

  The location and name of the file containing the properties required for cloning the database.

Example

The following example creates a cloned database using the parameters contained in the `/u01/files/create_clone.props` file:

```
emcli create_clone
      -inputFile=/u01/files/create_clone.props
```

create_credential_set

Creates a new credential set. Only Enterprise Manager Super Administrators can create new credential sets.

Format

```
  emcli create_credential_set
        -set_name="set_name"
        -target_type="ttype"
        -supported_cred_types="supported_cred_types"
        -monitoring
        [-auth_target_type = "authenticating_target_type"
        [-description ="description]"

[ ] indicates that the parameter is optional
```

Parameters

- **set_name**

  Credential set name to be created.

- **target_type**

  Target type of the new credential set.

- **supported_cred_types**

  Credential types supported by this credential set. You can list the available credential types by using the command show_credential_type_info.

- **monitoring**

  Creates a monitoring credential set.

- **auth_target_type**

  Target type for the supported cred types. The default value is target_type.

- **description**

  Description of the credential set.

Examples

This example creates a new credential set named New_Credential_Set.

```
emcli create_credential_set
        -set_name=New_Credential_Set
        -target_type=host
        -supported_cred_types=HostCreds;HostSSHCreds
        -description="Example credential set"
```

create_custom_plugin_update

Creates a custom plug-in update using a plug-in that is already deployed to a Management Agent. Includes all of the patches that were applied to the source plug-in. Use this in place of Oracle-supplied plug-in versions for all subsequent plug-in deployments on any Management Agent.

Format

```
emcli create_custom_plugin_update
      -agent_name="agent_name"
      -plugin_id="plugin_id"
      [-overwrite]

[ ] indicates that the parameter is optional.
```

Parameters

- **agent_name**

  Management Agent (host:port) on which the plug-in and its patches are deployed.

- **plugin_id**

  ID of the plug-in that should be used for creating the custom plug-in update. To view a list of plug-ins deployed on a Management Agent, run 'emcli list_plugins_on_agent'.

- **overwrite**

  Overwrites and updates an existing custom plug-in update, if a custom plug-in update already exists for that plug-in in the repository. If not provided, the new custom plug-in update is not created for that plug-in. Applies only for subsequent plug-in deployments. Does not automatically redeploy on the Management Agents where the source plug-in was previously deployed. To redeploy on such Management Agents, run 'emcli redeploy_plugin_on_agent'.

Examples

Example 1

The following example creates a custom plug-in update for the `oracle.sysman.db` plug-in that is already deployed on the Management Agent named `host.example.com`. If a custom plug-in update already exists for the `oracle.sysman.db` plug-in, then the command does not overwrite it, and therefore, does not create a new custom plug-in update.

```
emcli create_custom_plugin_update
      -agent_name="host.example.com"
      -plugin_id="oracle.sysman.db"
```

Example 2

The following example creates a custom plug-in update for the `oracle.sysman.db` plug-in, which is already deployed on the Management Agent named `host.example.com`, by overwriting and updating the custom plug-in update that already exists for the `oracle.sysman.db` plug-in in the repository.

```
emcli create_custom_plugin_update
      -agent_name="host.example.com"
      -plugin_id="oracle.sysman.db"
```

```
-overwrite
```

create_database

Creates a database.

Format

```
emcli create_database
       [-dbType="type_of_database"]
       [-hostTargets="list_of_host_targets"]
       [-cluster="cluster_target_name"]
       -oracleHome="Oracle_Home_location"
       -gdbName="global_database_name"
       -templateName="path_and_display_name_of_the_software_library_entity"
       -hostCreds="named_credential_for_OS_user"
       -sysCreds="named_credential_for_SYS_user"
       -systemCreds="named_credential_for_SYSTEM_user"
       -dbsnmpCreds="named_credential_for_DBSNMP_user"
       [-sid="database_system_identifier"]
       [-racConfigType="RAC_configuration_type"]
       [-nodeList="comma-separated_node_names"]
       [-serverPoolList="comma-separated_list_of_server_pools"]
       [-newServerPool="new_server_pool_name_and_cardinality"]
       [-racOneServiceName="service_name_for_RAC_one-node_database"]
       [-templateInSwlib="TRUE|FALSE"]
       [-templateStageLocation="temporary_directory_on_agent_side"]
       [-storageType="FS|ASM"]
       [-dataFileLocation="Location_of_data_files "]
       [-recoveryAreaLocation="Fast_Recovery_Area_location "]
       [-enableArchiving]
       [-useOMF]
       [-listeners="comma-separated_list_of_listeners_database"]
       [-newListener="new_listener_and_port"]

[ ] indicates that the parameter is optional
```

Parameters

- **dbType**

  Type of database that needs to be created. Valid values are:

  – SINGLE_INSTANCE —To create a database on one particular host or a list of hosts.

  – RAC — To create a cluster database on multiple nodes.

  – RACONE — To create a RAC One-node database.

  RAC and RACONE require the use of the cluster parameter.

- **hostTargets**

  Comma-separated list of host targets where a single-instance database needs to be created. This is a mandatory parameter for a SINGLE_INSTANCE database.

- **cluster**

  Cluster target name for the RAC database on which a cluster needs to be created. The target name should be valid and should have at least one node attached to the target. This is a mandatory parameter for RAC and RACONE databases.

- **oracleHome**

Oracle home of the host targets or cluster target. The Oracle home should be present in all of the targets.

- **gdbName**

  Global database name of the database.

- **templateName**

  Fully-qualified path and display name of the software library entity.

- **hostCreds**

  Named host credentials of the user who owns the Oracle Home installation.

- **sysCreds**

  Named database credentials to be used to create the SYS user.

- **systemCreds**

  Named database credentials to be used to create the SYSTEM user.

- **dbsnmpCreds**

  Named database credentials to be used to create the DBSNMP user.

- **sid**

  Database system identifier., which can be a maximum length of 12 for SINGLE_INSTANCE, 8 otherwise. This should be alphanumeric, with the first character being an alpha character.

- **racConfigType**

  RAC configuration type. Valid values are:

  – POLICY_MANAGED

  – ADMIN_MANAGED

  The default is ADMIN_MANAGED if not provided.

- **nodeList**

  List of valid node names for ADMIN_MANAGED RAC databases. If not provided, all the nodes for the given cluster target are used.

- **serverPoolList**

  Comma-separated list of server pool names for POLICY_MANAGED RAC databases.

- **newServerPool**

  Note: You can either use serverPoolList or newServerPool, but not both. For newServerPool, cardinality is mandatory and should be a positive integer greater than 0.

- **racOneServiceName**

  Service name for the RAC One Node database.

- **templateInSwlib**

Boolean value stating whether the template is from the software library. Valid values are TRUE if the template is from the software library, otherwise FALSE. The default is FALSE if you do not provide this parameter.

- **templateStageLocation**

  Fully-qualified path to where the template should be staged on the host target.

- **storageType**

  Type of storage preferred for the database. Valid values are:

  – FS for File System. This is the default if the parameter is not provided.

  – ASM for Automatic Storage Management.

- **dataFileLocation**

  Location of the data files.

- **recoveryAreaLocation**

  Fast Recovery Area location.

- **enableArchiving**

  Indicates whether archiving of the database is required. Valid values are TRUE if archiving is required, otherwise FALSE. The default is FALSE.

- **useOMF**

  Indicates whether to use Oracle Managed Files.

- **listeners**

  Comma-separated list of listeners (name:port) to register the created database. This is for the SINGLE_INSTANCE database type only, and will be ignored for a RAC database.

- **newListener**

  New listener (name:port) creates a new listener and registers the database. This is for the SINGLE_INSTANCE database type only, and will be ignored for a RAC database.

Examples

Example 1

```
emcli create_database -oracleHome=/u01/app/oracle/product/11.2.0/dbhome_2
-gdbName=testdbee -hostCreds=host_named
                      -sysCreds=sys -systemCreds=system -dbsnmpCreds=dbsnmp
-templateName=/u01/app/oracle/product/11.2.0/
dbhome_2/assistants/dbca/templates/General_Purpose.dbc
                      -dbType=SINGLE_INSTANCE -hostTargets=host1
```

Example 2

```
emcli create_database -oracleHome=/u01/app/oracle/product/11.2.0/dbhome_2
-gdbName=testdbee -hostCreds=host_named
                      -sysCreds=sys -systemCreds=system -dbsnmpCreds=dbsnmp
-templateName=/u01/app/oracle/product/11.2.0/
dbhome_2/assistants/dbca/templates/General_Purpose.dbc
                      -dbType=SINGLE_INSTANCE -hostTargets=host1
-newListener=NEWLSNR:1527
```

Example 3

```
emcli create_database -oracleHome=/u01/app/oracle/product/11.2.0/dbhome_2
-gdbName=testRACCli -hostCreds=cluster_named -sysCreds=sys -systemCreds=system
-dbsnmpCreds=dbsnmp
                      -templateName=/u01/app/oracle/product/11.2.0/
dbhome_2/assistants/dbca/templates/General_Purpose.dbc -dbType=RAC
-cluster=cluster1
                      -dataFileLocation=/u01/share/oradata
-recoveryAreaLocation=/u01/share/fra
```

Example 4

```
emcli create_database -oracleHome=/u01/app/oracle/product/11.2.0/dbhome_2
-gdbName=testdbee -hostCreds=cluster_named
                      -sysCreds=sys -systemCreds=system -dbsnmpCreds=dbsnmp
-templateName=/u01/app/oracle/product/11.2.0/
dbhome_2/assistants/dbca/templates/General_Purpose.dbc
                      -dbType=RAC -cluster=cluster1 -racConfigType=POLICY_MANAGED
-newServerPool=sp1:2
```

create_database_size

Specify a database size that overrides values specified in the service template.

Format

```
emcli create_database_size -name="<size name>"
-description="<size description>"
[-attributes="cpu:<number of cores>;memory:<memory in GB>;processes:<max number of
processes>;storage:<Total Storage in GB allocated to database>;"]
[-source_type="Profile Source"]

[ ] indicates that the parameter is optional.
```

Note: Use one or more attributes to specify the database size. The different attributes must be separated by a semicolon (;). Attributes specified using the database_size verb override values specified in the service template.

Parameters

- name

  Creates a name for the database size.

- description

  Creates a description for the database size.

- attributes

  Defines the database size. Attributes must be separated by a semicolon(;). You can specify values for the following attributes:

  cpu: Total number of cpu cores.

  memory: Total maximum in GB.

  processes: Total number of processes that can simultaneously connect to the database.

  storage: Total storage that is allocated to the database (in GB)

Example

The following example creates a database size named Small with a maximum of four CPUs, 50 GB of storage, and 4 GB of memory.

```
emcli create_database_size
      -name=Small
      -description="Small size database"
      -attributes="cpu:4;storage:50;memory:4"
      -source_type="weblogic_domain"
```

create_dbprofile

Creates a new database profile.

Format

```
emcli create_dbprofile
       -input_file=data:"file:path"
       [-schedule=
               [frequency:interval|weekly|monthly|yearly];
               start_time:yy-MM-dd HH:mm;
               end_time:yy-MM-dd HH:mm;
               [repeat:#m];
               [months:#,#,#,...];
               [days:#,#,#,...];
               [tz:{timezone ID}];
               [grace_period:xxx];
      ]
[-purge_policy=DAYS|SNAPSHOTS: number]

[ ] indicates that the parameter is optional.
```

Parameters

- input_file

  A property file which completely describes the type of profile that will be created and the options used.

- schedule

  frequency: Frequency type with which the Database Profile will be created. It can be interval(in minutes), weekly, monthly or yearly

  start_time: Denotes the starting time of Database Profile Component creation in the format yy-MM-dd HH:mm

  end_time: Denotes the end time of Database Profile Component Creation Repetition in the format yy-Mm-dd HH:mm

  repeat: Repetition rate at which Database Profile will be created. If the frequency is interval, then repeat will be in minutes

  months: Number of months after which repetition of Database Profile Component Creation will occur

  days: Number of days after which repetition of Database Profile Component Creation will occur

  tz: Time Zone ID for example tz:America/New_York

  grace_period: A period of time in minutes that defines the maximum permissible delay when attempting to create a Databasse Profile. If the job system cannot start the execution within a time period equal to the scheduled time + grace period, it will set the create Database Profile to be skipped.By default, grace period is indefinite

- purge_policy

  You can purge the collected data based on number of days or count of snapshots. If you do not specify purge_policy, it is defaulted to NONE. Allowed values: DAYS, SNAPSHOT

DAYS specify the number of days after which the data component should be purged.

SNAPSHOT specify the count or number of data components, after which older data will be purged

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example creates a new database profile based on the property file "profile.txt" with the specified schedule and purge policy.

```
emcli create_dbprofile -input_file="data:/tmp/profile.txt"
-schedule="frequency:interval;start_time:14-10-05 05:30;end_time:14-10-12
05:23;repeat:30;grace_period:60;tz:America/New_York" -purge_policy=DAYS:2
```

create_dbaas_quota

Creates a database quota for an SSA user role.

Format

```
emcli create_dbaas_quota
       -role_name="<SSA user role name>"
       -databases="<number of database requests>"
       -schema_services="<number of schema service requests>"
       -pluggable_databases="<number of Pluggable database service requests>"
       -memory="<memory(GB)>"
       -storage="<storage(GB)>"

[ ] indicates that the parameter is optional.
```

Parameters

- **role_name**

  Name of the SSA user role for which the quota is to be created.

- **databases**

  Number of database service requests allowed. For example, for 10 requests enter:

  ```
  -databases="10"
  ```

- **schema_services**

  Number of schema service requests allowed. For example, for 10 requests enter:

  ```
  -schema_services="10"
  ```

- **pluggable_databases**

  Number of pluggable database service requests allowed. For example, for 10 requests enter:

  ```
  -pluggable_databases="10"
  ```

- **memory**

  Amount of memory usage allowed. For example, for 10 GB enter:

  ```
  -memory="10"
  ```

- **storage**

  Amount of storage usage allowed. For example, for 10 GB enter:

  ```
  -storage="10"
  ```

Example

This example assigns the quota for the role My Role:

```
emcli create_dbaas_quota
       -role_name="My Role"
       -databases="10"
       -schema_services="10"
       -pluggable_databases="10"
       -memory="99"
       -storage="99"
```

create_dbprofile

Creates a new database profile.

Format

```
emcli create_dbprofile
       -input_file=data:"file:path"
       [-schedule=
               [frequency:interval|weekly|monthly|yearly];
               start_time:yy-MM-dd HH:mm;
               end_time:yy-MM-dd HH:mm;
               [repeat:#m];
               [months:#,#,#,...];
               [days:#,#,#,...];
               [tz:{timezone ID}];
               [grace_period:xxx];
      ]
[-purge_policy=DAYS|SNAPSHOTS: number]

[ ] indicates that the parameter is optional.
```

Parameters

- **input_file**

  A property file which completely describes the type of profile that will be created and the options used.

- **schedule**

  frequency: Frequency type with which the database profile will be created. It can be interval (in minutes), weekly, monthly, or yearly.

  start_time: Denotes the starting time of database profile component creation in the format yy-MM-dd HH:mm.

  end_time: Denotes the end time of database profile component creation repetition in the format yy-Mm-dd HH:mm.

  repeat: Repetition rate at which the database profile will be created. If the frequency is interval, then repeat is in minutes.

  months: The number of months after which repetition of Database Profile Component Creation will occur.

  days: The number of days after which repetition of Database Profile Component Creation will occur.

  tz: Time zone ID for example tz:America/New_York.

  grace_period: A period of time in minutes that defines the maximum permissible delay when attempting to create a databasse profile. If the job system cannot start the execution within a time period equal to the scheduled time + grace period, it sets the create database profile to be skipped. By default, the grace period is indefinite.

- **purge_policy**

  The policy that specifies how you can purge the collected data based on number of days or count of snapshots. If you do not specify purge_policy, it is defaulted to NONE. The allowed values are DAYS, SNAPSHOT.

DAYS specifies the number of days after which the data component should be purged.

SNAPSHOT specifies the count or number of data components, after which older data will be purged.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example creates a new database profile based on the property file "profile.txt" with the specified schedule and purge policy:

```
emcli create_dbprofile -input_file="data:/tmp/profile.txt"
-schedule="frequency:interval;start_time:14-10-05 05:30;end_time:14-10-12
05:23;repeat:30;grace_period:60;tz:America/New_York" -purge_policy=DAYS:2
```

create_diag_snapshot

Creates a diagnostic snapshot for specified targets.

Format

```
emcli create_diag_snapshot
      -name=<name>
      -desc=<description>
      -start_time=<yyyy/MM/dd HH:mm>
      -end_time=<yyyy/MM/dd HH:mm>
      -targets=<type1:name1;type2:name2;…>
      [-diag_type_odl_target_types=<type1;type2; ….>]
      [-diag_type_odl_online_logs=<true|false>]
      [-diag_type_odl_offline_logs=<true|false>]
      [-diag_type_jvmd_target_types=<type1;type2; ….>]
      [-diag_type_jvmd_properties="<pname1:pval1;pname2:pval2;…>"]
      [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of diagnostic snapshot to be created. Make sure that a diagnostic snapshot does not exists for the specified name.

- **desc**

  Description of the diagnostics snapshot.

- **start_time**

  Start time for collecting the logs. The snapshot will contain all logs between the start time and end time. Make sure that the duration is valid for the snapshot.

- **end_time**

  End time for collecting the logs. The snapshot will contain all logs between the start time and end time. Make sure that the duration is valid for the snapshot.

- **targets**

  Target type and target name list for the snapshot. This list can contain all targets for the specific system. User can choose specific target types in optional parameters for selected diagnostic types.

- **diag_type_odl_target_types**

  Target type list for the Oracle Diagnostic Logging (ODL) diagnostic type. You can select a subset of target types from the target list for snapshot creation.

- **diag_type_odl_online_logs**

  By default, online logs are collected for a snapshot. You can choose to collect online, offline, or both logs for the Oracle Diagnostic Logging (ODL) diagnostic type.

- **diag_type_odl_offline_logs**

  By default, offline/archive logs are not collected for a snapshot. You can choose to collect online, offline, or both logs for the Oracle Diagnostic Logging (ODL) diagnostic type.

- **diag_type_jvmd_target_types**

  Target type list for the JVMD diagnostic type. You can select a subset of target types from the target list for snapshot creation.

- **diag_type_jvmd_properties**

  Properties list to collect logs for the JVMD diagnostic type.

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Examples

Example 1

This example creates a snapshot for EMGC_DOMAIN and EMGC_OMS1 targets with offline logs. The target types (weblogic_domain and weblogic_j2eeserver) belong to the Oracle Diagnostic Logging (ODL) diagnostic type.

```
emcli create_diag_snapshot
      -name=wls_snapshot
      -desc= "Snapshot for Weblogic Domains and Server"
      -start_date="2012/10/02 10:30"
      -end_date="2012/10/03 22:30"
      -targets="weblogic_domain:/EMGC_EMGC_DOMAIN/EMGC_DOMAIN;
                weblogic_j2eeserver: /EMGC_EMGC_DOMAIN/EMGC_DOMAIN/EMGC_OMS1"
```

Example 2

This example creates a snapshot for the weblogic_j2eeserver target type with offline logs. You can filter the target types on top of the target list.

```
emcli create_diag_snapshot
      -name=wls_snapshot
      -desc="Snapshot for Weblogic Domains and Server"
      -start_date="2012/10/02 10:30"
      -end_date="2012/10/03 22:30"
      -targets="weblogic_domain:/EMGC_EMGC_DOMAIN/EMGC_DOMAIN;
              weblogic_j2eeserver:/EMGC_EMGC_DOMAIN/EMGC_DOMAIN/EMGC_OMS1;
              weblogic_j2eeserver:/EMGC_EMGC_DOMAIN/EMGC_DOMAIN/EMGC_ADMIN_SERVER"
      -diag_type_odl_target_types="weblogic_j2eeserver"
      -diag_type_odl_offline_logs=true
```

create_fmw_domain_profile

Creates a Fusion Middleware provisioning profile from a WebLogic Domain.

Format

```
emcli create_fmw_domain_profile
        -name="profile_name"
        -ref_target="reference_target_name"
        [-description="profile_description"]
        [-oh_cred="Oracle_home_owner_credentials"]
        [-includeOh]
        [-schedule=
              start_time:yyyy/MM/dd HH:mm;
              [tz:{java_timezone_ID}];
              [grace_period:xxx];
        ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the profile to be created.

- **ref_target**

  Name of the WebLogic Domain target to be used as a reference to create the profile.

- **description**

  Description of the profile to be created.

- **oh_cred**

  Named credential to be used to access the reference host. The format is:

  ```
  CREDENTIAL_NAME:CREDENTIAL_OWNER
  ```

  All operations are performed on the Administration Server host. Credentials of the Oracle Home owner on the Administration Server host are required. If no named credential is provided, preferred host credentials for the Oracle Home target are used.

- **includeOh**

  Includes the Oracle Home binaries in the profile.

- **schedule**

  Schedule for the Deployment Procedure. If not specified, the procedure executes immediately.

  – start_time: Time when the procedure should start.

  – tz: Time zone ID.

  – grace_period: Grace period in minutes.

Examples

Example 1

This example creates a WebLogic Domain profile for the specified schedule from the given WebLogic Domain target using preferred credentials.

```
emcli create_fmw_domain_profile
      -name="BitlessDomainProfile"
      -ref_target="/Farm01_base_domain/base_domain"
      -description="A domain profile without software bits"
      -schedule="start_time:2014/6/21 21:23;tz:America/New_York;grace_period:60"
```

Example 2

This example immediately creates a WebLogic Domain plus Oracle Home from the given WebLogic Domain target using given named credentials.

```
emcli create_fmw_domain_profile
      -name="DomainProfileWithBits"
      -ref_target="/Farm01_base_domain/base_domain"
      -oh_cred="MY_HOST_CRED:SYSMAN"
      -includeOh
```

create_fmw_home_profile

Creates a Fusion Middleware provisioning profile from an Oracle Home.

Format

```
emcli create_fmw_home_profile
        -name="profile_name"
        -ref_target="reference_target_name"
        [-description="profile_description"]
        [-oh_cred="Oracle_home_owner_credentials"]
        [-schedule=
              start_time:yyyy/MM/dd HH:mm;
              [tz:{java timezone ID}];
              [grace_period:xxx];
        ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the profile to be created.

- **ref_target**

  Name of the Oracle Home target to be used as a reference to create the profile.

- **description**

  Description of the profile to be created.

- **oh_cred**

  Named credential to be used to access the reference host. The format is:

  ```
  CREDENTIAL_NAME:CREDENTIAL_OWNER
  ```

  If no named credential is provided, preferred host credentials for the Oracle Home target are used.

- **schedule**

  Schedule for the Deployment Procedure. If not specified, the procedure executes immediately.

  – start_time: Time when the procedure should start.

  – tz: Time zone ID.

  – grace_period: Grace period in minutes.

Examples

Example 1

This example creates a profile on the specified schedule from the given Oracle Home target using preferred credentials.

```
emcli create_fmw_home_profile
      -name="OhProfile1"
      -ref_target="/Farm01_base_domain/base_domain"
      -description="An Oracle Home profile"
```

```
      -schedule="start_time:2014/6/21 21:23;tz:America/New_York;grace_period:60"
```

Example 2

This example immediately creates a profile from the given Oracle Home target using given named credentials.

```
emcli create_fmw_home_profile
      -name="OhProfile2"
      -ref_target="WebLogicServer_10.3.6.0_myhost.mycompany.com_5033"
      -oh_cred="MY_HOST_CRED:SYSMAN"
```

create_group

Defines a group name and its members. After you create the group, you can edit it from the Enterprise Manager Cloud Control console to configure Summary Metrics to be displayed for group members.

Command-Line Format

```
emcli create_group
      -name="name"
      [-type=<group>]
      [-add_targets="name1:type1;name2:type2;..."]...
      [-is_propagating="true/false"]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
create_group
      (name="name"
      [,type=<group>]
      [,add_targets="name1:type1;name2:type2;..."]...
      [,is_propagating="true/false"])

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the group.

- **type**

  Group type: group. Defaults to "group".

- **add_targets**

  Add existing targets to the group. Each target is specified as a name-value pair `target_name:target_type`. You can specify this option more than once in command-line format.

- **is_propagating**

  Flag that indicates whether or not privilege on the group will be propagated to member targets. The default is false.

Examples

These examples create a database-only group named `db_group`. This group consists of two Oracle databases: `emp_rec` and `payroll`.

Example 1 - Command-Line

```
emcli create_group
      -name=db_group
      -add_targets="emp_rec:oracle_database"
      -add_targets="payroll:oracle_database"
```

Example 2 - Scripting and Interactive

```
create_group
       (name="db_group"
       ,add_targets="emp_rec:oracle_database;payroll:oracle_database")
```

These examples create a mixed member-type group named `my_group` that consists of an Oracle database (`database2`), listener (`dblistener`), and host (`mymachine.myco.com`).

Example 3 - Command-Line

```
emcli create_group
      -name=my_group
      -add_targets="database2:oracle_database;dblistener:oracle_listener
      -add_targets="mymachine.myco.com:host"
```

Example 4 - Scripting and Interactive

```
create_group
       (name="my_group"
       ,add_targets="database2:oracle_database;
        dblistener:oracle_listener;mymachine.myco.com:host")
```

These examples create a host-only group named `my_hosts` that consists of three systems within the `example.com` domain: `smpsun`, `dlsun`, and `supersun`.

Example 5 - Command-Line

```
emcli create_group
      -name=my_hosts
      -add_targets="example.com:host"
      -add_targets="example.com:host;supersun.example.com:host"
```

Example 6 - Scripting and Interactive

```
create_group
       (name="my_hosts"
       ,add_targets="example.com:host;example.com:host;supersun.example.com:host")
```

create_inst_media_profile

Defines a group name and its members. After you create the group, you can edit it from the Enterprise Manager Cloud Control console to configure Summary Metrics to be displayed for group members.

Command-Line Format

```
emcli create_inst_media_profile
       -name="profile_name"
       -host="host_target"
       -version="media_version"
       -platform="media_platform"
       [-description="profile_description"]
       [-host_cred="Oracle_home_owner_credentials"]
       -files=
             WebLogic:WLSFile1;
             SOA:SOAFile1,SOAFile2;
             OSB:OSBFile;
             RCU:RCUFile;

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the profile to be created.

- **host**

  Name of the host target that where all of the installation files are stored.

- **version**

  Version of the installation media.

- **platform**

  Platform for which the installation media is applicable.

- **description**

  Description of the profile to be created.

- **host_cred**

  Named credential to be used to access the files. The format is:

  ```
  CREDENTIAL_NAME:CREDENTIAL_OWNER.
  ```

  If you do not provide a named credential, preferred host credentials for the Oracle Home target are used.

- **files**

  List of files to be uploaded to the Software Library. Acceptable products are WebLogic, SOA, OSB and RCU. An upload for WebLogic is mandatory. The format is:

  ```
  PRODUCT1:FILE1,FILE2;PRODUCT2:FILE3,FILE4
  ```

Examples

Example 1

This example uploads the installation media file for the WebLogic Server to the Software Library from the given location on the given host. Preferred host credentials will be used to access the files.

```
emcli create_inst_media_profile
      -name="WebLogic1036Installer"
      -host="myhost.mycompany.com"
      -description="WebLogic Server 10.3.6.0 installer"
      -version="10.3.6.0"
      -platform="Generic"
      -files="WebLogic:/u01/media/weblogic/wls1036_generic.jar"
```

Example 2

This example uploads the installation media files for SOA and the WebLogic Server to the Software Library from the given location on the given host. The provided named credentials are used to access the files.

```
emcli create_inst_media_profile
      -name="SOA+WLSInstaller"
      -host="myhost.mycompany.com"
      -description="SOA 11.1.1.7.0 and WebLogic Server 10.3.6.0 installer"
      -version="11.1.1.7.0"
      -platform="Generic"
      -host_cred="MY_HOST_CRED:SYSMAN"
      -files="WebLogic:/u01/media/weblogic/
       wls1036_generic.jar;SOA:/u01/media/soa/soa1.zip,/u01/media/soa/soa2.zip"
```

create_jeeappcom

Creates a Java EE Application Component in the software library. On successful creation, the entity revision is displayed under the specified folder in the software library.

Format

```
emcli create_jeeappcomp
        -name="entity_name"
        -folder_id="folder_id"
        [-desc="entity_desc"]
        [-attr="<attr name>:<attr value>"]
        [-prop="<prop name>:<prop value>"]
        [-secret_prop="<secret prop name>:<secret prop value>"]
        [-note="note text"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the entity.

- **folder_id**

  ID of the folder where the entity will be created. The Software Library Home page exposes the identifier for folders and entities as a custom column called Internal ID. By default, this column is hidden.

- **desc**

  A short description about the entity. The new description is visible to all existing revisions.

- **attr**

  A name:value pair for specifying the attributes of an entity. It is represented as "attr_name:attr_value". For specifying values for multiple attributes, repeat the `-attr` option.

- **prop**

  A name:value pair for specifying the configuration properties of an entity. It is represented as prop_name:prop_value. For specifying values for multiple properties, repeat the option.

- **secret_prop**

  A name:value pair for specifying the configuration property and its secret value. Do not provide the secret value on the command line. Instead, enter the property name and press the Enter key. Provide the secret value when you are prompted for it.

- **note**

  Information related to the entity such as changes being made to the entity or modification history that you want to track.

Examples

Example 1

Creates a Java EE Application component called "myJeeAppComp' in the folder identified by folder_id. You can find the folder ID using the custom column called Internal ID available on the Software Library home page. Note that this column is hidden by default.

```
emcli create_jeeappcomp
      -name="myJeeAppComp"
       folder_
id="oracle:defaultService:em:provisioning:1:cat:B13B3B7B086458CFE040E80A19AA560C"
+E34
```

Example 2

Creates entity named 'myJeeAppComp' in the folder identified by folder_id with a short description about the entity. Entity attributes such as PRODUCT, PRODUCT_VERSION, and VENDOR are specified. Value for the DEFAULT_HOME configuration property is also specified. A note that includes information related to the entity is included.

The identifier of the newly created entity revision will be printed on the standard output.

```
emcli upload_jeeappcomp_file
emcli create_jeeappcomp
        -name="myJeeAppComp"
folder_
id="oracle:defaultService:em:provisioning:1:cat:B13B3B7B086458CFE040E80A19AA560C"
        -desc="myJeeAppComp description"
        -attr="PRODUCT:JEEApp"
        -attr="PRODUCT_VERSION:3.0"
        -attr="VENDOR:Vendor"
        -prop="DEFAULT_HOME:/u01/myJeeAppComp3/"
        -note="myJeeAppComp for test servers"
```

create_job

Creates and schedules a job. This verb supports multi-task jobs.

> **Note:** EM CLI permits OS Script jobs to be run against database targets by setting the targetType property for -input_file in the create_job verb. For example:
>
> ```
> targetType=oracle_database
> ```
>
> You can set other target types similarly.

EM CLI supports the following job types:

```
ASMSQLScript
ASSOCIATE_CS_FA
ASSOCIATE_DOMAIN_FA
AssociateClusterASM
BlockAgent
CoherenceCacheAddition
CoherenceNodesRefresh
Config Log Archive Locations
DbMachineDashboard
DiscoverPDBEntities
FusionMiddlewareProcessControl
GlassFishProcessControl
InstallKernelModuleJob
Log Rotation
OSCommand
OpatchPatchUpdate_PA
RMANScript
RefreshFromEMStore
RefreshFromMetalink
RefreshFusionInstance
SOABulkRecovery
SQLScript
ShutdownDB
StartDepartedCohNodes
StartDepartedCohStoreNodes
StartFusionInstance
StartupDB
StatspackPurge
StopFusionInstance
Upgrade Exalogic Systems
WebLogic Control
WebLogic Domain Discover
WebLogic Domain Refresh
```

> **Note:** Not all job types support all target types. Use describe_job_type to determine which target types are supported for a given job type.

Format

```
emcli create_job
   -input_file=property_file:"filename"
```

```
   [-name="job_name"]
   [-type="job_type"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the job.

- **job_type**

  Name of the job type. You can obtain a template property file for the job type by using the describe_job_type verb.

- **input_file**

  Provide the file name to load the properties for creating and scheduling the job. The property file must be accessible to the EM CLI client for reading. Another job of the same job type could also be used to generate the property file using the EM CLI verb describe_job.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Example

This example creates and schedules a job with name MYJOB1 and job type MyJobType1 with the property file present at location /tmp/myjob1_prop.txt .

```
emcli create_job -name=MYJOB1 -job_type=MyJobType1 -input_file="property_
file:/tmp/myjob1_prop.txt"
```

create_job_from_library

Creates a job using a library job as a template. This verb supports multi-task jobs.

Format

```
emcli create_job_from_library
    -lib_job_name="library_job_name"
    -name="new_job_name"
   [-owner="library_job_owner"]
   [-input_file=property_file:"filename"]
   [-appendtargets]

[ ] indicates that the parameter is optional
```

Parameters

- **lib_job_name**

  Library job to use as a template.

- **owner**

  Owner of the job. When this parameter is not specified, the default job owner is the logged in Enterprise Manager administrator.

- **name**

  Name of the new job to be created. You can also specify the name in the property file. If no name is specified, a name is generated from the name of the library job.

- **input_file**

  "filename" can be provided to load the properties for creating the job.

  If you specify a property file, the values in the property file override or append to existing values in the library job. If you do not specify a property file, the library job is submitted unchanged.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **appendtargets**

  Appends targets in the property file to existing targets in the library job. Otherwise, library job targets are overwritten by targets in the property file if they are specified.

Examples

Example 1

This example creates a job named MYJOB based on the library job MYLIBJOB1.

```
emcli create_job_from_library -lib_job_name=MYLIBJOB1 -name=MyJob
```

Example 2

This example creates a job named MYJOB2 based on the library job MYLIBJOB1. Properties in /tmp/myjob1_prop.txt will override library job settings.

```
emcli create_job_from_library -lib_job_name=MYLIBJOB1 -name=MyJob2 -input_
file=property_file:/tmp/myjob1_prop.txt
```

create_library_job

Creates a library job. This verb supports multi-task jobs.

Format

```
emcli create_library_job
   [-name="job_name"]
   [-type="job_type"]
    -input_file=property_file:"filename"

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the job.

- **type**

  Name of the job type. You can obtain a template property file for the job type by using the describe_job_type verb.

- **input_file**

  Provide the file name to load the properties for creating the library job. The property file must be accessible to the EM CLI client for reading. Another library job of the same job type could also be used to generate the property file using the EM CLI verb describe_library_job.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Example

This example creates a library job with the name MYLIBJOB1 and job type MyJobType1 with the property file present at location /tmp/myjob1_prop.txt .

```
emcli create_library_job -name=MYLIBJOB1 -job_type=MyJobType1
-input_file="property_file:/tmp/myjob1_prop.txt"
```

create_named_credential

Creates a named credential. You can provide input parameters using command line arguments or an input properties file. It also supports the input_file tag for passwords and parameter values.

Command-Line Format

```
emcli create_named_credential
       -cred_name=<name>
       -auth_target_type=<authenticating_target_type>
       -cred_type=<credential_type>
       -cred_scope=<credential_scope>
       -cred_desc=<credential_description>
       -target_name=<target_name>
       -target_type=<target_type>
       -test
       -test_target_name=<test_target_name>
       -test_target_type=<test_target_type>
       -input_file=<tag:value>
       -input_bfile=<tag:value>
       -properties_file=<filename>
       -attributes=<p1:v1;p2:v2;...>
```

Scripting and Interactive Format

```
create_named_credential
       (cred_name=<name>
       ,auth_target_type=<authenticating_target_type>
       ,cred_type=<credential_type>
       ,cred_scope=<credential_scope>
       ,cred_desc=<credential_description>
       ,target_name=<target_name>
       ,target_type=<target_type>
       ,test
       ,test_target_name=<test_target_name>
       ,test_target_type=<test_target_type>
       ,input_file=<tag:value>
       ,input_bfile=<tag:value>
       ,properties_file=<filename>
       ,attributes=<p1:v1;p2:v2;...>)
```

Parameters

- **cred_name**

  Credential name, such as MyBackUpCreds. This is required if you do not use properties_file.

- **auth_target_type**

  Authenticating target type (e.g. host). This is required if you do not use properties_file.

- **cred_type**

  Credential type. This is required if you do not use properties_file.

- **cred_scope**

  Possible values are global | instance. The default is global.

- **cred_desc**

  Credential description.

- **target_name**

  This is required when cred_scope is instance.

- **target_type**

  This is required when cred_scope is instance.

- **test**

  Use this to test the credential before saving.

- **test_target_name**

  Use this to supply the target name to test a global credential. This is required when cred_scope is global and the test parameter is used.

- **test_target_type**

  Use this to supply the target type to test a global credential. This is required when cred_scope is global and the test parameter is used.

- **input_file**

  Use this to supply sensitive property values from the file.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_bfile**

  Use this to supply binary property values from the file.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **properties_file**

  Use this to pass all parameters from the file. Values given on the command line take precedence.

- **attributes**

  Specify credential columns as follows:

  ```
  colname:colvalue;colname:colvalue
  ```

  You can change the separator value using -separator=attributes=<newvalue>, and you can change the subseperator value using -subseparator=attributes=<newvalue>.

  For more information about the separator and subseperator parameters , see Section 5.2, "-input_file Syntax Guidelines".

Examples

These examples create a HostCreds named credential with username foo and password bar:

Example 1 - Command Line

```
emcli create_named_credential
        -cred_name=NC1
        -auth_target_type=host
```

```
        -cred_type=HostCreds
        -attributes="HostUserName:foo;HostPassword:bar"
```

Example 2 - Scripting and Interactive

```
create_named_credential
       (cred_name="NC1"
       ,auth_target_type="host"
       ,cred_type="HostCreds"
       ,attributes="HostUserName:foo;HostPassword:bar")
```

These examples create a privilege delegation credential with user name foo, password bar, privilege delegation type SUDO, and RUNAS user root:

Example 3 - Command-Line

```
emcli create_named_credential
        -cred_name=NC1
        -auth_target_type=host
        -cred_type=HostCreds
        -attributes="HostUserName:foo;HostPassword:bar;PDPTYPE:SUDO;RUNAS:root"
```

To use Powerbroker attributes, the string should be:

```
-attributes="HostUserName:foo;HostPassword:bar;PDPTYPE:POWERBROKER;RUNAS:root;
PROFILE:EMGC
```

Example 4 - Scripting and Interactive

```
create_named_credential
       (cred_name="NC1"
       ,auth_target_type="host"
       ,cred_type="HostCreds"
       ,attributes="HostUserName:foo;
        HostPassword:bar;PDPTYPE:SUDO;RUNAS:root")
```

To use Powerbroker attributes, the string should be:

```
,attributes="HostUserName:foo;HostPassword:bar;PDPTYPE:POWERBROKER;RUNAS:root;
PROFILE:EMGC
```

These examples read the password from the mypasswordfile.txt file.

Example 5 - Command-Line

```
emcli create_named_credential
        -cred_name=NC1
        -auth_target_type=host
        -cred_type=HostCreds
        -attributes="HostUserName:foo;HostPassword:tag"
        -input_file="tag:mypasswordfile.txt"
```

Example 6 - Scripting and Interactive

```
create_named_credential
       (cred_name="NC4",
       ,auth_target_type="host"
       ,cred_type="HostCreds"
       ,attributes="HostUserName:foo;HostPassword:tag"
       ,input_file="tag:mypasswordfile.txt")
```

These examples prompt for the password from standard input:

Example 7 - Command-Line

```
emcli create_named_credential
        -cred_name=NC1
        -auth_target_type=host
        -cred_type=HostCreds
        -attributes="HostUserName:foo;HostPassword:"
```

Example 7 - Scripting and Interactive

```
create_named_credential
       (cred_name="NC1"
        ,auth_target_type="host"
        ,cred_type="HostCreds"
        ,attributes="HostUserName:foo;HostPassword:bar")
```

These examples specify prop1.txt as a multi-line Java properties file, in which each line contains a parameter=value format. You can provide the password in the same file or not specify it. If not specified, you are prompted for it.

Example 8 - Command-Line

```
emcli create_named_credential
        -properties_file=prop1.txt
```

Example 9 - Scripting and Interactive

```
create_named_credential
        (properties_file="prop1.txt")
```

create_operation_plan

Creates an operational plan for the Oracle Site Guard operation.

Format

```
emcli create_operation_plan
        -primary_system_name="name_of_primary_system"
        -standby_system_name="name_of_standby_system"
        -system_name="name_of_system"
        -operation="name_of_operation"
        -name="name_of_operation_plan"
        -role="role_associated_with_system"
```

Parameters

- **primary_system_name**

  Name of your system associated with the primary site. Enter this **parameter** for switchover or failover operations.

- **standby_system_name**

  Name of your system associated with the standby site. Enter this **parameter** for switch-over or fail-over operations.

- **system_name**

  Name of the system. Enter this **parameter** for start or stop operations.

- **operation**

  The function of the operation. Examples: switchover, failover, start, or stop.

- **name**

  Name of the operation plan.

- **role**

  Role associated with a system when you run an operation (start or stop).

Examples

Example 1

```
emcli create_operation_plan
      -primary_system_name="BISystem1"
      -standby_system_name="BISystem2"
      -operation="switchover"
      -name="BISystem1-switchover-plan"
```

Example 2

```
emcli create_operation_plan
      -system_name="austin"
      -operation="start"
      -name="BISystem1-start-plan"
      -role="Primary"
```

See Also

emcli get_operation_plans and emcli submit_operation_plan

create_paas_zone

Creates a PaaS Infrastructure Zone.

Format

```
emcli create_paas_zone
       -name="<PaaS Zone name>"
       -credential="<global named credential>"
       [-hosts="<Host1,Host2,Host3...>"]
       [-ovm_zones="<OVMZone1,OVMZone2,OVMZone3...>"]
       [-roles="<ssaRole1,ssaRole2,..>"]
       [-description="<PaaS Zone description>"]
       [-cpu_utilization="<value between 1 and 100>"]
       [-memory_utilization="<value between 1 and 100>"]

[ ] indicates that the parameter is optional.
```

Parameters

- **name**

  Name of the PaaS Infrastructure Zone to be created.

- **credential**

  Global named credentials to be used for provisioning in this PaaS Infrastructure Zone. The credentials should be the same for all hosts. A cloud administrator can only use the named credentials that they own.

- **hosts**

  A comma-separated list of the host targets to be added as members of this Paas Infrastructure Zone.

- **ovm_zones**

  Comma-separated list of the Oracle Virtual Machine (OVM) Zone targets to be added as members of this Paas Infrastructure Zone. You must add at least one host or OVM Zone target for a PaaS Infrastructure Zone to be created.

- **roles**

  Comma-separated list of SSA roles that can access this PaaS Infrastructure Zone. A PaaS Infrastructure Zone can be made available to a restricted set of users through the use of roles. The SSA roles should already be created before executing this EM CLI command.

- **description**

  Description of the PaaS Infrastructure Zone.

- **cpu_utilization**

  Placement policy constraints enable the cloud administrator to set maximum ceilings for any host in the PaaS Infrastructure Zone. This constraint restricts the maximum resource consumption for the host members in a PaaS Infrastructure Zone. For example, a production PaaS Infrastructure Zone might limit CPU utilization to 80%, whereas a development PaaS Infrastructure Zone might allow up to 95 percent utilization. The service instance will be provisioned on the first host that satisfies the placement constraints. The value entered must be between 1 and 100. If not specified, the default value of 80% is used.

- **memory_utilization**

  Placement policy constraint for the PaaS Infrastructure Zone that restricts the percent of memory used. The value entered must be between 1 and 100. If not specified, the default value of 80% is used.

Example

This example creates a Paas Infrastructure Zone with the name My PaaS Zone:

```
emcli create_paas_zone
       -name="My PaaS Zone"
       -credential="ZoneNamedCredentials"
       -hosts="host1.mycompany.com, host2.mycompany.com"
       -roles="SSA_USER_ROLE"
       -description="This is a test PaaS Zone"
       -cpu_utilization="85"
       -memory_utilization="75"
```

create_patch_plan

Creates a new patch plan with the specified name and the patch-target map.

Format

```
emcli create_patch_plan
        -name="name"
        -input_file=data:"file_path"
        [-impact_other_targets="add_all | add_original_only | cancel"]
        [-problems_assoc_patches="ignore_all_warnings | cancel"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the setting.

- **input_file**

  Input data to create a new patch plan. You must provide the data in the property name-value pairs.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **impact_other_targets**

  Action to take when other targets are impacted while adding the patches to the plan. Possible values for this parameter are:

  add_all — Add all impacted targets to the plan.
  add_original_only — Only add original targets to the plan.
  cancel — Cancel the plan creation.

- **problems_assoc_patches**

  Action to take when there are problems associating patches to targets. Possible values for this parameter are:

  ignore_all_warnings — Ignore all warnings.
  cancel — Cancel the plan creation.

See Also

delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan

upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

Examples

```
emcli create_patch_plan -name="plan_name" -input_file=data:"/tmp/patchplan.props"

emcli create_patch_plan -name="plan name" -input_file=data:"/tmp/patchplan.props"
-impact_other_targets="add_all"

emcli create_patch_plan -name="plan name"  -input_file=data:"/tmp/patchplan.props"
-impact_other_targets="add_all" -problems_assoc_patches="ignore_all_warnings"
```

You can use the following sample input file to create a patch plan with two patches:

```
patch.0.patch_id=4518443
         patch.0.release_id=80102010
         patch.0.platform_id=226
         patch.0.language_id=0
         patch.0.target_name=orclws
         patch.0.target_type=oracle_database
         patch.1.patch_id=4424952
         patch.1.release_id=80102030
         patch.1.platform_id=46
         patch.1.language_id=0
         patch.1.target_name=arac
         patch.1.target_type=rac_database
```

create_pool

Creates a software pool.

Format

```
emcli create_pool
       -name="<software pool name>"
       -target_type="<software pool target type>"
       -paas_zone="<Paas Infrastructure Zone of software pool>"
       -members="<Member1, Member2...>"
       [-description="<software pool description>"]
       [-placement_constraints="<constraint1=value1, constraint2=value2...>"]
       [-member_constraints="<constraint1=value1, constraint2=value2>"]
       [-properties="<property1=value1, property2=value2>"]

[ ] indicates that the parameter is optional.
```

Parameters

- **name**

  Name of the software pool to be created.

- **target_type**

  Target type of the software pool to be created, for example "mwaas_zone" for the middleware Pool, "oracle_cloud_zone" for the database pool, and "schaas_pool" for schema pool.

- **paas_zone**

  Name of PaaS Infrastructure Zone in which the software pool is to be created.

- **members**

  Comma-separated list of targets to be added as members of the software pool. The targets to be added must satisfy the member constraints specified.

- **description**

  Description of the software pool.

- **placement_constraints**

  Comma-separated key-value pairs of the placement constraints that enable the self-service administrator to set maximum ceilings for resource utilization. This ability provides protection for the members of the software pool in terms of resource consumption. For example, a production software pool might enforce more conservative limits, whereas a development software pool might enforce more liberal limits.

- **member_constraints**

  Comma-separated key-value pairs that restrict the addition of member targets to a software pool with a set criteria. Execute "emcli get_pool_allowed_member_ constraints -target_type=<Target type>" to retrieve the list of allowed possible member constraints for a pool target type.

- **properties**

  Comma-separated key-value pairs for additional properties that must be specified based on the pool target type.

Example

The following example creates the My Pool software pool:

```
emcli create_pool
       -name="My Pool"
       -target_type="mwaas_zone"
       -paas_zone="My PaaS Zone"
       -members="MyMember"
       -description="This is a test Pool"
       -placement_constraints="MAX_INSTANCES=20"
       -member_constraints="VERSION=10.3.5.0"
```

create_pluggable_database

Creates a pluggable database.

Format

```
emcli create_pluggable_database
-cdbTargetName="CDB_target_name"
-cdbTargetType="CDB_target_type"
-cdbHostCreds="CDB_host_credentials"
-pdbName="new_PDB_name"
-sourceType="DEFAULT|UNPLUGGED_PDB|CLONE"
[-cdbTargetCreds="CDB_target_credentials"]
[-numOfPdbs="number_of_PDBs"]
[-sourceFromSWLIB="Source_from_software_library"]
[-pdbTemplateInSWLIB="URN_of_PDB_template_component"]
[-sourcePDBTempStagingLocation="source_PDB_temporary_staging_location"]
[-unpluggedPDBType="unplugged_PDB_type"]
[-sourcePDBArchiveLocation="source_PDB_archive_location"]
[-sourcePDBMetadataFile="source_PDB_metadata_file"]
[-sourcePDBDatabackup="source_PDB_data_backup"]
[-sourcePDBName="source_PDB_name"]
[-sourceCDBCreds="source_CDB_credentials"]
[-pdbAdminCreds="PDB_admin_crednentials"]
[-useOMF="use_OMF_location"]
[-sameAsSource="store_data_files_in_same_location_as_source_CDB"]
[-newPDBFileLocation="storage_location_for_data_files_of_created_PDB."]
[-createAsClone="create_PDB_as_clone"]
[-lockAllUsers="locks_PDB users_of_new_PDB."]

[ ] indicates that the parameter is optional
```

Parameters

- **cdbTargetName**

  Name of the setting.

- **cdbTargetType**

  Type of setting you want to create.

- **cdbHostCreds**

  Parameter value. Choose one of the following parameters:

- **pdbName**

  Delimiter inserted between name-value pairs for the given name. The default value is a semi-colon (;).

- **sourceType**

  Separator inserted between the name and value in each name-value pair for the given name. The default value is a semi-colon (;).

Examples

Example 1

```
emcli create_pluggable_database -cdbTargetName=database -cdbTargetType=oracle_
database
```

```
-pdbName=pdb -sourceType=UNPLUGGED_PDB -unpluggedPDBType=ARCHIVE
-sourcePDBArchiveLocation=/u01/app/oracle/product/12.1.0/dbhome_
2/assistants/dbca/templates/a.tar.gz
-cdbHostCreds=HOST_CREDS -cdbTargetCreds=DBSNMP
-newPDBFileLocation=/u01/app/oradata/pdb
-pdbAdminCreds=pdb_creds -lockAllUsers
```

Example 2

```
emcli create_pluggable_database -cdbTargetName=database
-cdbTargetType=oracle_database
-pdbName=pdb -numOfPdbs=2 -sourceType=UNPLUGGED_PDB -unpluggedPDBType=RMAN
-sourcePDBMetadataFile=/u01/app/oracle/product/12.1.0/dbhome_
2/assistants/dbca/templates/a.xml
-sourcePDBDatabackup=/u01/app/oracle/product/12.1.0/dbhome_
2/assistants/dbca/templates/a.dfb
-cdbHostCreds=HOST_CREDS -cdbTargetCreds=DBSNMP
-newPDBFileLocation=/u01/app/oradata/pdb
-pdbAdminCreds=pdb_creds -createAsClone
```

create_privilege_delegation_setting

Creates a privilege delegation setting template to apply later. You must create at least one setting to use the `apply_privilege_delegation_setting` verb.

Command-Line Format

```
emcli create_privilege_delegation_setting
        -setting_name="name"
        -setting_type="ttype"
        [-settings="setting"]
        [-separator=settings=";"]
        [-subseparator=settings=","]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
create_privilege_delegation_setting
        (setting_name="name"
        ,setting_type="ttype"
        [,settings="setting"]
        [,separator=settings=";"]
        [,subseparator=settings=","])

[ ] indicates that the parameter is optional
```

Parameters

- **setting_name**

  Name of the privilege delegation setting template.

- **setting_type**

  Type of setting you want to create.

- **settings**

  Parameter value. Choose one of the following parameters:

  %USERNAME% — Name of the user running the command.
  %RUNAS% — Run the command as this user.
  %COMMAND% — Sudo command.

  The %USER%, %RUNAS%,, %COMMAND% are tokens that the end-user has to use as-is while creating/modifying the privilege delegation settings. The system replaces these tokens with the actual values at run time depending on the command being run and for which user. Also, %command% should be upper case %COMMAND% for 10.2.0.5 GC.

- **separator**

  Delimiter inserted between name-value pairs for the given name. The default value is a semi-colon (;).

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **subseparator**

  Separator inserted between the name and value in each name-value pair for the given name. The default value is a semi-colon (;).

For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

These examples create a setting named `sudo_setting`. The setting is of type SUDO, and the Sudo path used is `/usr/local/bin/sudo`. Sudo arguments are:

```
-S
-u
%RUNAS%
%COMMAND%
```

Example 1 - Command-Line

```
emcli create_privilege_delegation_setting
      -setting_name=sudo_setting
      -setting_type=SUDO
      -settings="SETTINGS:/usr/local/bin/sudo -S -u %RUNAS% %COMMAND%"
```

Example 2 - Scripting and Interactive

```
create_privilege_delegation_setting
       (setting_name="sudo_setting",
        setting_type="SUDO",
        settings="SETTINGS:/usr/local/bin/sudo -S -u %RUNAS% %COMMAND%")
```

These examples create a setting named `pb_setting`. The setting is of type POWERBROKER, and the PowerBroker path used is `/etc/pbrun`. Arguments are:

```
%RUNAS%
%PROFILE%
%COMMAND%
```

Example 3 - Command-Line

```
emcli create_privilege_delegation_setting
      -setting_name="pb_setting"
      -setting_type="POWERBROKER"
      -settings="SETTINGS,/etc/pbrun %RUNAS% %PROFILE% %COMMAND%"
      -separator="settings=;"
      -subseparator="settings=,"
```

Example 4 - Scripting and Interactive

```
create_privilege_delegation_setting
      (setting_name=pb_setting
      ,setting_type=POWERBROKER
      ,settings="SETTINGS,/etc/pbrun %RUNAS% %PROFILE% %COMMAND%"
      ,separator="settings=;"
      ,subseparator="settings=,")
```

These examples are similar to examples 3 and 4, except that they also add arguments PASSWORD_PROMPT_STRING and Password.

Example 5 - Command-Line

```
emcli create_privilege_delegation_setting
      -setting_name="pb_setting"
      -setting_type="POWERBROKER"
```

```
      -settings="SETTINGS,/etc/pbrun %RUNAS% %PROFILE% %COMMAND%";
       PASSWORD_PROMPT_STRING,password:"
      -separator="settings=;"
      -subseparator="settings=,"
```

Example 5 - Scripting and Interactive

```
create_privilege_delegation_setting
      (setting_name=pb_setting
      ,setting_type=POWERBROKER
      ,settings="SETTINGS,/etc/pbrun %RUNAS% %PROFILE% %COMMAND%";
       PASSWORD_PROMPT_STRING,password:"
      ,separator="settings=;"
      ,subseparator="settings=,")
```

create_rbk

Creates a retroactive blackout on given targets and updates their availability. Only Enterprise Manager Administrators with OPERATOR privilege on the target can perform this action. The retroactive blackout feature needs to be enabled from the user interface to use this command.

Format

```
emcli create_rbk
  -reason="<blackout_reason>"
  -add_targets="name1:type1;name2:type2;..."
  -schedule="start_time:<yyyy-MM-dd HH:mm:ss>;end_time:<yyyy-MM-dd
   HH:mm:ss>;[tzregion:<timezone_region>;]"
  [-propagate_targets]

[ ] indicates that the parameter is optional
```

Parameters

- **reason**

  Reason to be stored for the retroactive blackout. If you have SUPER_USER privileges (you are an Enterprise Manager Super Administrator), any text string can be used for the reason. The reason is added to the list of allowable blackout reasons if it is not already in the list. If you do not have SUPER_USER privileges, you must specify one of the text strings returned by the get_blackout_reasons verb.

- **add_targets**

  Targets to add to the retroactive blackout. Each target is specified as target_name:target_type. You can specify this option more than once.

- **schedule**

  Schedule for retroactive blackout. The following arguments are mandatory for providing a retroactive blackout schedule:

  – schedule=start_time - The start date/time of the blackout. The format of the value is "yyyy-MM-dd HH:mm:ss". For example: "2013-09-20 12:12:12"

  – schedule=end_time - The end date/time of the blackout. The format of the value is "yyyy-MM-dd HH:mm:ss". For example: "2013-09-20 12:15:00"

  – schedule=tzregion - The timezone region to use. For example: "UTC". If not provided, tzregion is defaulted to UTC.

- **propagate_targets**

  A blackout for a target of type "host" applies the blackout to all non-agent targets on the host. Regardless of whether this option is specified, a blackout for a target that is a composite or a group applies the blackout to all members of the composite or group.

Examples

Example 1

This example creates a retroactive blackout on Oemrep_Database and updates the target's availability record from 2013-09-20 12:12:12 UTC to 2013-09-20 12:15:00 UTC as the blackout.

```
emcli create_rbk -reason="Testing"
      -add_targets="Oemrep_Database:oracle_database"
      -schedule="start_time:2013-09-20 12:12:12;end_time:2013-09-20
       12:15:00;tzregion:UTC"
```

Example 2

This example creates a retroactive blackout for all targets on host example.company.com and updates their availability records from 2013-09-20 12:12:12 UTC to 2013-09-20 12:15:00 UTC as the blackout.

```
emcli create_rbk -reason="Testing"
      -add_targets="example.company.com:host"
      -schedule="start_time:2013-09-20 12:12:12;end_time:2013-09-20
       12:15:00;tzregion:UTC"
      -propagate_targets
```

create_red_group

Defines a redundancy group name and its members. After you create the redundancy group, you can edit it from the Enterprise Manager Cloud Control console to configure charts to be displayed for redundancy group members.

Format

```
emcli create_red_group
      -name="name"
      [-type=<generic_redundancy_group>]
      -add_targets="name1:type1;name2:type2;..."...
      [-owner=<redundancy_group_owner>]
      [-timezone_region=<actual_timezone_region>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the redundancy group.

- **type**

  Redundancy group type. Defaults to `generic_redundancy_group`.

- **add_targets**

  Add existing targets to the redundancy group. Each target is specified as a name-value pair `target_name:target_type`. You can specify this option more than once.

- **owner**

  Owner of the redundancy group.

- **timezone_region**

  Time zone region of this redundancy group.

Example

This example creates a redundancy group named `lsnr_group`. This group consists of two Oracle listeners: `emp_rec` and `payroll`.

```
emcli create_red_group -name=lsnr_group
      -add_targets="emp_rec:oracle_listener"
      -add_targets="payroll:oracle_listener"
```

create_redundancy_group

Creates a redundancy group.

Format

```
emcli create_redundancy_group
       -redundancyGroupName="redGrpName"
       -memberTargetType="tType"
       -memberTargetNames="tName1;tName2"
       [-group_status_criterion=NUMBER|PERCENTAGE]
       [-group_status_tracked=UP|DOWN]
       [-group_status_value=<group_status_value>]
       [-timezone_region=<valid_time_zone_region>]
       [is_propagating=true|false]

[ ] indicates that the parameter is optional
```

Parameters

- **redundancyGroupName**

  Name of the redundancy group.

- **memberTargetType**

  Target type of the constituent member targets.

- **memberTargetNames**

  Member targets for this redundancy group.

- **group_status_criterion**

  This parameter and the next two calculate the status of the Redundancy Group. Consequently, you need to specify all three options together. If this is not to be a capacity group, you need to specify the following combination:

  ```
  -group_status_criterion='NUMBER' -group_status_tracked='UP' -group_status_
  value='1']
  ```

- **group_status_tracked**

  See the parameter above.

- **group_status_value**

  See the group_status_criterion parameter.

  You can specify any value between 1 and 100 if -group_status_criterion= "PERCENTAGE", or any value between 1 and the number of targets present if -group_status_criterion="NUMBER".

- **timezone_region**

  Time zone region of this redundancy group. For a list of valid time zone regions, enter the following command at SQLPLUS:

  ```
  SELECT TZNAME FROM V$TIMEZONE_NAMES
  ```

  You may need to have the SELECT_CATALOG_ROLE role to execute this command.

- **is_propagating**

Indicates whether or not the privilege on the redundancy group will be propagated to member targets. The default value is false.

Examples

Example 1

This example creates a redundancy group with the name 'redGrp1' and with listener, listener2, listener3 as its member targets. The status is calculated as the redundancy group being up if 55 percent of its member targets are up.

```
emcli create_redundancy_group -redundancyGroupName='redGrp1'
      -memberTargetType='oracle_listener'
      -memberTargetNames='listener;listener2;listener3'
      -group_status_criterion='PERCENTAGE'
      -group_status_tracked='UP'
      -group_status_value='55'
```

Example 2

This example creates a 'redGrp1' redundancy group with listener, listener2, and listener3 as its member targets and time zone as PST8PDT. The status is calculated as the redundancy group being up if two of its member targets are up.

```
emcli create_redundancy_group -redundancyGroupName='redGrp1'
         -memberTargetType='oracle_listener'
         -memberTargetNames='listener;listener2;listener3'
         -timezone_region='PST8PDT'
         -group_status_criterion='NUMBER'
         -group_status_tracked='UP'
         -group_status_value='2'
```

create_resolution_state

Creates a new resolution state that describes the state of incidents or problems. Only super administrators can execute this command. The new state is always added between the New and Closed states. You need to specify the exact position of this state in the overall list of states by using the position option. The position can be between 2 and 98.

The state is applicable by default to both incidents and problems. You can use the applies_to option to indicate that the state is applicable only to incidents or problems. A success message is reported if the command is successful. An error message is reported if the create fails.

Format

```
emcli create_resolution_state
      -label="label_for_display"
      -position="display_position"
       [-applies_to="INC|PBLM"]

[ ] indicates that the parameter is optional
```

Parameters

- **label**

  End-user visible label of the state. The label cannot exceed 32 characters. You can change this later if needed.

- **position**

  Position of this state within the overall list of states. This is used when displaying the list of states in the user interface. The position can be between 2 and 98. You can change the position of the state later if needed.

  It is recommended that you set the position with sufficient gaps to facilitate moving states around. For example, if you set the positions to 5, 10, and 15 instead of 2, 3, and 4, it is easier to move a state from position 15 to 9, for instance, in contrast to the latter scheme, in which you would have to move all states to provide space for the reordering.

- **applies_to**

  Indicates that the state is applicable only for incidents or problems. By default, states apply to both incidents and problems. Supported values are "INC" or "PBLM".

Examples

Example 1

This example adds a resolution state that applies to both incidents and problems at position 25.

```
emcli create_resolution_state -label="Waiting for Ticket" -position=25
```

Example 2

This example adds a resolution state that applies to problems only at position 35.

```
emcli create_resolution_state -label="Waiting for SR" -position=35 -applies_
to=PBLM
```

create_role

Creates a new Enterprise Manager admininistrator role.

Command-Line Format

```
emcli create_role
        -name="role_name"
        [-type="type_of_role"]
        [-description="description"]
        [-roles="role1;role2;..."]
        [-users="user1;user2;..."]
        [-privilege="name[;secure_resource_details]]"
        [-separator=privilege="sep_string"]
        [-subseparator=privilege="subsep_string"]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
create_role
        (name="role_name"
        [,type="type_of_role"]
        [,description="description"]
        [,roles="role1;role2;..."]
        [,users="user1;user2;..."]
        [,privilege="name[;secure_resource_details]]"
        [,separator=privilege="sep_string"]
        [,subseparator=privilege="subsep_string"])

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Role name.

- **type**

  Type of role. The default value for this parameter is EM_ROLE. The other possible value is EXTERNAL_ROLE.

- **description**

  Description of the role.

- **roles**

  List of roles to assign to this new role. Currently, the only built-in role is `PUBLIC`.

- **users**

  List of users to whom this role is assigned.

- **privilege**

  Privilege to grant to this role. You can specify this option more than once.
  Note: Privileges are case-insensitive.

  secure_resource_details should be specified as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
  name2=resource_column_value2]..]"
  ```

- **separator**

  Specify a string delimiter to use between name-value pairs for the value of the privilege option. The default separator delimiter is ";" .

- **subseparator**

  Specify a string delimiter to use between name and value in each name-value pair for the value of the privilege option. The default separator delimiter is ";" .

Examples

These examples create a role named `my_new_role` with the one-sentence description - "This is a new role called `my_new_role`". The role combines three existing roles: `role1`, `role2`, and `role3`. The role also has two added privileges: to view the job with ID 923470234ABCDFE23018494753091111 and to view the target `host1.example.com:host`. The role is granted to `johndoe` and `janedoe`.

Example 1 - Command-Line

```
emcli create_role
      -name="my_new_role"
      -desc="This is a new role called my_new_role"
      -roles="role1;role2;role3"
      -privilege="view_job;923470234ABCDFE23018494753091111"
      -privilege="view_target;host1.example.com:host"
      -users="johndoe;janedoe"
```

Example 2 - Scripting and Interactive

```
create_role
      (name="my_new_role"
      ,desc="This is a new role called my_new_role"
      ,roles="role1;role2;role3"
      ,privilege="view_job;923470234ABCDFE23018494753091111"
      ,privilege="view_target;host1.example.com:host"
      ,users="johndoe;janedoe")
```

These examples create a role named my_external_role with a role type of EXTERNAL_ROLE and one-sentence description of "This is an external role."

Example 3 - Command-Line

```
emcli create_role
      -name="my_external_role"
      -type="EXTERNAL_ROLE"
      -desc="This is an external role"
```

Example 4 - Scripting and Interactive

```
create_role
      (name="my_external_role"
      ,type="EXTERNAL_ROLE"
      ,desc="This is an external role")
```

create_service

Creates a service to be monitored by Enterprise Manager.

Format

```
emcli create_service
      -name='name'
      -type='type'
      -availType=test|system
      -availOp=and|or
      [-hostName=<host_name>
      [-agentURL=<agent_url>
      [-properties='pname1|pval1;pname2|pval2;...']
      [-timezone_region=<gmt_offset>]
      [-systemname=<system_name>]
      [-systemtype=<system_type>]
      [-keycomponents='keycomp1name:keycomp1type;keycomp2name:keycomp2type;...']
      [-beacons='bcn1name:bcn1isKey;bcn2name:bcn2isKey;...']
      [-input_file="template:Template file name]"]
      [-input_file="variables:Variable file name]"]
      [-sysAvailType=<availability_type>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service name. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks.

- **type**

  Service type.

- **availType**

  Sets the availability to either test-based or system-based. If availability is set to `test`, template file, beacons, and variable are required arguments. If availability is set to `system`, systemname, systemtype, and keycomponents are required.

- **availOp**

  Availability operator. If `and`, uses all key tests/components to decide availability. If `or`, uses any key tests/components to decide availability.

- **hostName**

  Network name of the system running the Management Agent that is collecting data for this target instance.

- **agentURL**

  URL of the Management Agent that is collecting data for this target instance. If you enter the host name, the Agent URL of the host is automatically entered in this field.

- **properties**

  Name-value pair (that is, prop_name|prop_value) list of properties for the service instance.

- **timezone_region**

  GMT offset for this target instance (-7 or -04:00 are acceptable formats).

- **systemname**

  System name on which service resides.

- **systemtype**

  Type of system for which you want to create the service.

- **keycomponents**

  Name-type pair (that is, `keycomp_name:keycomp_type`) list of key components in the system that are used for the service.

- **beacons**

  Name-isKey pairs that describe the beacons of the service. If isKey is set to `Y`, beacon is set as a key-beacon of the service. The service should have at least one key beacon if the availability is set to test-based.

- **input_file**

  Template file name is the XML file that includes the template definition. Variable file defines the values for the template.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **sysAvailType**

  Type of availability when the availType is system-based. Sets the availability to either system target directly or selected components of a system.

  If availability is set to 'system target directly,' the system needs to have availability[status] defined. systemname and systemtype are required parameters.

  If availability is set to 'selected components of a system,' systemname, systemtype and keycomponents are required parameters.

  If availability is set to 'system target directly,' and if the system does not have availability[status] defined, the availability set is invalid. Therefore, the only option that can be set is 'selected components of a system'.

Examples

Example 1

This example creates a generic service named my_service with specified properties on a generic system named my_system. The availability is set as system-based, and the availability is based on system target status.

```
emcli create_service
      -name='my service' -type='generic_service'
      -availType='system' -availOp='or'
      -sysAvailType='system target directly'
      -properties='prop1:value1; prop2:value2'
      -timezone_region='PST8PDT'
      -systemname='my system' -systemtype='generic_system'
```

Example 2

This example creates a generic service named `my_service` with specified properties on a generic system named `my system` with specified key components. The availability is set as system-based.

```
emcli create_service
      -name='my_service' -type='generic_service'
      -availType='system' -availOp='or'
      -properties='prop1:value1; prop2:value2'
      -timezone_region='PST8PDT'
      -systemname='my system' -systemtype='generic_system'
      -keycomponents='database:oracle_database; mytestbeacon:oracle_beacon'
```

Example 3

This example creates a generic service named `my_service` with specified properties with tests defined in mytests.xml, and beacons MyBeacon as the key beacon and MyOtherBeacon as a non-key beacon. Availability is set as test-based.

```
emcli create_service
      -name='my_service' -type='generic_service'
      -availType='test' -availOp='or'
      -properties='prop1:value1; prop2:value2'
      -timezone_region='PST8PDT'
      -input_file='template:mytests.xml'
      -beacons='MyBeacon:Y;MyOtherBeacon:N'
```

create_service_template

Creates a service template.

Format

```
emcli create_service_template
       -name="<service template>"
       -service_family="<service family>"
       -service_type="<service type>"]
       -pool_target_type="target type of software pools"
       -software_pools="<SwPool1,SwPool2,SwPool3,...>"
       [-roles="<SsaRole1,SsaRole2,..>"]
       [-description="<service template description>"]
       [-input_file="data:<service executable metadata file>"]

[ ] indicates that the parameter is optional.
```

Parameters

- **name**

  Name of the service template to be created.

- **service_family**

  Service family for which the service template is being created, for example DBAAS for database, MWAAS for middleware.

- **service_type**

  Service type for which the service template is being created, for example PhysicalWLS for a physical middleware service template.

- **pool_target_type**

  Target type of the software pools to be associated with the service template.

- **software_pools**

  Comma-separated list of software pools to be associated with the service template.

- **roles**

  Comma-separated list of SSA roles that can access this service template. A service template can be made available to a restricted set of users through the use of roles. The SSA roles should already be created before executing this EM CLI command.

- **description**

  Description of the service template.

- **input_file**

  Contains configuration and profile data in JSON format that will be required for setting values of procedure configuration variables. For example:

  ```
  input_file="data:executable.json"
  ```

Example

This example creates the service template My Service Template:

```
emcli create_service_template
       -name="Middleware service template August"
```

```
-service_family="MWAAS"
-service_type="PhysicalWLS"
-pool_target_type="mwaas_zone"
-software_pools="MyPoolOH"
-roles="SSA_USER_ROLE"
-description="Middleware small instance service template."
-input_file="data:executable.json"
```

create_siteguard_configuration

Creates a site configuration for Site Guard. It associates the systems and their roles.

Format

```
emcli create_siteguard_configuration
      -primary_system_name=<name>
      -standby_system_name=<name1;name2;...>
```

Parameters

- **primary_system_name**

  Name of the system associated with the primary site.

- **standby_system_name**

  Name of the system associated with the standby system. You can specify more than one system name.

Examples

```
emcli create_siteguard_configuration
      -primary_system_name="BISystem1"
      -standby_system_name="BISystem2"
```

See Also

update_siteguard_configuration

delete_siteguard_configuration

create_siteguard_credential_association

Associates the credentials with the targets in a site.

Format

```
emcli create_siteguard_credential_association
      -system_name=<name>
      [-target_name=<name>]
      -credential_type=<type>
      [-credential_name=<name>]
      [-use_preferred_credential=<type>]
      -credential_owner=<owner>

[ ] indicates that the parameter is optional.
```

Parameters

- **system_name**

  Name of the system.

- **target_name**

  Name of the target.

- **credential_type**

  Type of credential, which can be HostNormal, HostPrivileged, WLSAdmin, or DatabaseSysdba.

- **credential_name**

  Name of the credential. If you do not specify this parameter, you need to specify the use_preferred_credential parameter.

- **use_preferred_credential**

  Name of the credential. If you do not specify this parameter, you need to specify the credential_name parameter.

- **credential_owner**

  Owner of the credential.

Examples

Example 1

```
emcli create_siteguard_credential_association
      -system_name="BISystem1"
      -credential_type="HostNormal"
      -credential_name="HOST-SGCRED"
      -credential_owner="sysman"
```

Example 2

```
emcli create_siteguard_credential_association
      -system_name="BIsystem1"
      -target_name="database-instance"
      -credential_type="HostNormal"
      -credential_name="HOST-DBCRED"
      -credential_owner="sysman"
```

create_siteguard_script

Associates scripts (pre-script, post-script, and storage script) with the Site Guard configuration.

Format

```
emcli create_siteguard_script
      -system_name=<name>
      -operation=<name>
      -script_type=<type>
      [-host_name=[<name1;name2;...>]
      -path=<path_of_script>
      [-all_hosts=true|false]
      [-role=Primary|Standby]

[ ] indicates that the parameter is optional.
```

Parameters

- **system_name**

  Name of the system.

- **operation**

  Name of the operation. Examples: Switchover, Failover, Start, or Stop.

- **script_type**

  Type of script, which can be Mount, UnMount, Pre-Script, Post-Script, Failover, or Switchover.

- **host_name**

  Name of the host where this script will run. You can specify this option more than once.

- **path**

  Path to the script.

- **all_hosts**

  Allows the script to run on all the hosts in the system. This parameter overrides the host_name.

- **role**

  Configures the script based on the system role. By default, the script is configured for both primary and standby roles for a given system.

Examples

Example 1

```
emcli create_siteguard_script
      -system_name="BISystem1"
      -operation="Switchover"
      -script_type="Pre-Script"
      -path="/tmp/prescript"
      -all_hosts="true"
      -role="Primary"
```

Example 2

```
emcli create_siteguard_script
      -system_name="BISystem1"
      -operation="Switchover"
      -script_type="Pre-Script"
      -path="/tmp/prescript"
      -host_name="BIHOST1"
      -host_name="BIHOST2"
```

create_swlib_entity

Creates an entity in the software library. Upon successful creation, the entity revision appears under the specified folder on the software library home page.

Format

```
emcli create_swlib_entity
      -name="entity_name"
      -folder_id="folder_id"
      [-type]="type_internal_id"]
      [-subtype]="subtype_internal_id"]
      [-desc="entity_desc"]
      [-attr="<attr_name>:<attr value>"]
      [-prop="<prop_name>:<prop value>"]
      [-secret_prop="<secret_prop_name>:<secret_prop=_value>"]
      [-note="note_text"]

[ ] indicates that the parameter is optionalis optional
```

Parameters

- **name**

  Name of the entity.

- **folder_id**

  Identifier of the folder where the entity is to be created. The software library home page exposes the identifier for folders and entities as a custom column (Internal ID), and is hidden by default.

- **type**

  Use the list_swlib_entity_types verb to identify the type.

- **subtype**

  Internal identifier of the entity subtype, which defaults to the 'Generic Component' subtype for the 'Component' type. Use the list_swlib_entity_types verb to identify the subtype.

- **desc**

  Description of the entity.

- **attr**

  An attribute and its value, separated by a colon (:). To specify values for multiple attributes, repeat this option.

- **prop**

  A configuration property and its value, separated by a colon (:). To specify values for multiple properties, repeat this option.

- **secret_prop**

  A configuration property and its secret value separated by a colon (:). It is recommended to not specify the secret value on the command line. If omitted from the command line, the value is prompted for. To specify values for multiple properties, repeat this option.

- **note**

A note on the entity. For multiple notes, repeat this option.

Examples

Example 1

This example creates an entity named 'myAcmeInstall' under the specified folder. The entity is of type 'Component' and subtype 'Generic Component, by default. The folder identifier value can be found on the software library home page. The software library home page exposes the identifier for folders and entities as a custom column (Internal ID), and is hidden by default.

```
emcli create_swlib_entity
      -name="myAcmeInstall"
      -folder_id=
"oracle:defaultService:em:provisioning:1:cat:B13B3B7B086458CFE040E80A19AA560C"
```

Example 2

This example creates an entity named 'myAcmeInstall' under the specified folder with the specified description. The entity is of type 'Component' and subtype 'Generic Component' by default. Values for the entity attributes, viz. PRODUCT, PRODUCT_VERSION and VENDOR, are specified. The value for the configuration property named DEFAULT_HOME is specified. A note on the entity is also specified. The identifier of the newly created entity revision is printed on the standard output.

```
emcli create_swlib_entity
      -name="myAcmeInstall"
      -folder_id=
"oracle:defaultService:em:provisioning:1:cat:B13B3B7B086458CFE040E80A19AA560C"
       -desc="myAcmeInstall description"
       -attr="PRODUCT:Acme"
       -attr="PRODUCT_VERSION:3.0"
       -attr="VENDOR:Acme Corp"
       -prop="DEFAULT_HOME:/u01/acme3/"
       -note="myAcmeInstall for test servers"
```

create_swlib_folder

Creates a folder in the software library.

Format

```
emcli create_swlib_folder
      -name="folder_name"
      -parent_id="parent_folder_id"
      [-desc="folder_description"]

[ ] indicates that the parameter is optionalis optional
```

Parameters

- **name**

  Name of the folder.

- **parent_id**

  Identifier of the parent folder under which the folder is to be created. To create a folder under the root folder, specify the parent folder identifier as 'ROOT.' The software library home page exposes the identifier for folders and entities as a custom column (Internal ID) and is hidden by default.

- **desc**

  Description of the folder.

Example

This example creates a folder named 'myFolder' under the specified parent folder.

```
emcli create_swlib_folder
       -name="myFolder"
       -parent_id=
"oracle:defaultService:em:provisioning:1:cat:B13B3B7B086458CFE040E80A19AA560C"
       -desc="myFolder description"
```

create_system

Defines a system: name and its members. After the system is created, you can edit the system from the Enterprise Manager Cloud Control console to configure charts to be displayed for system members.

Format

```
emcli create_system
      -name="name"
      [-type=<system>]
      [-add_members="name1:type1:key_member/non_key_member;name2:type2;..."]...
      [-separator=add_members="sep_value"]
      [-subseparator=add_members="subsep_value"]
      -timezone_region="actual_timezone_region"
      [-owner="owner"]
      [-meta_ver="meta_version_of_system_type"]
      [-is_propagating="true|false"]
      [-availability_type="ALL|ANY"]

[ ] indicates that the parameter is optionalis optional
```

Parameters

- **name**

  Name of the system.

- **type**

  System type: generic_system. Defaults to "generic_system".

- **add_members**

  Add existing targets to the system. Each target is specified as a name-value pair `target_name:target_type`. You can specify this option more than once. key_member specifies that this target is a part of the systems availability calculation.

- **separator**

  Name-value pair separator for the given argument.

- **subseparator**

  Separates the name from the value for the given argument.

- **timezone_region**

  Actual time zone region.

- **owner**

  Owner of the system.

- **meta_ver**

  Meta version of the system type. Defaults to "1.0".

- **is_propagating**

  Flag to indicate if the privilege on the system will be propagated to member targets or not. The default value is false.

- **availability_type**

Availability calculation method of the system. Defining this is required if key_member is defined. ALL denotes that all key members must be up in order to mark the system as up. ANY denotes that at least one of the key members must be up in order to mark the system as up.

Examples

Example 1

This example creates a generic system named `db_system` and supports backward compatibility. This system consists of two Oracle databases: `emp_rec` and `payroll`. The owner of this system is `user1`. The meta version of the system type is `3.0`.

```
emcli create_system -name=db_system
      -add_members="emp_rec:oracle_database"
      -add_members="payroll:oracle_database"
      -timezone_region="PST8PDT"
      -owner="user1"
```

Example 2

This example creates a generic system named `my_system` that consists of an oracle database (`database2`), listener (`dblistener`), and host (`mymachine.myco.com`). The owner of this system is the logged-in user. The meta version of the system type is `1.0`. The example supports backward compatibility.

```
emcli create_system -name=my_system
      -add_members="database2:oracle_database;dblistener:oracle_listener
      -add_members="mymachine.myco.com:host"
      -timezone_region="PST8PDT"
```

Example 3

This example creates a generic system named db_system1. This system consists of two Oracle databases: emp_rec and payroll. emp_rec is a key member for the system. The availability calculation method is if ANY of the key members is up, the system is up. The meta version of the system type is 3.0. This example shows the recommended method for creating a system.

```
emcli create_system -name=db_system1
      -add_members="emp_rec$oracle_database$key_member"
      -add_members="payroll$oracle_database"
      -subseparator=add_members="$"
      -timezone_region="PST8PDT"
      -availability_type="ANY"
```

create_udmmig_session

Creates a session to migrate user-defined metrics (UDMs) to metric extensions for targets.

Format

```
emcli create_udmmig_session
      -name=<session_name>
      -desc=<session_description>
      [-udm_choice=<specific_udm_to_convert>]*
      {-target=<type:name_of_target_to_migrate> }*
      | {-input_file=targetList:<complete_path_to_file>};
         {-template=<template_name_to_update> }*
      | {-input_file=templateList:<complete_path_to_file>}
      [-allUdms]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the migration session to be created.

- **desc**

  Description of the migration session to be created.

- **udm_choice**

  Specify if the session should migrate specific UDMs. Otherwise, all UDMs are migrated.

- **target**

  The type:name of the target to be updated. You can specify multiple values.

- **input_file=targetList**

  Specify a file name that contains a list of targets, one per line, in the following format:

  ```
  <targetType>:<targetName>
  ```

  For more information about the input_file parameter see Section 5.2, "-input_file Syntax Guidelines".

- **template**

  Name of the monitoring template to update. You can specify multiple values.

- **input_file=templateList**

  Specify a file name that contains a list of templates, one name per line.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **allUdms**

  Forces the session to contain all UDMs from targets and templates. (The default behavior just selects those not in a session.)

Examples

Example 1

This example creates a new session named hostsession that migrates the UDM hostudm on the target testhost.

```
emcli create_udmmig_session
      -name=hostsession -desc="Convert UDMs for Host Target"
      -udm_choice=hostudm -target=host:testhost
```

Example 2

This example creates a new session named hostsession that migrates all the unconverted UDMs on the target testhost that are not in a session.

```
emcli create_udmmig_session
      -name=hostsession -desc="Convert UDMs for Host Target"
      -target=host:testhost -allUdms
```

create_user

Creates a new Enterprise Manager administrator.

Command-Line Format

```
emcli create_user
        -name="name"
        -password="password"
        [-type="user_type"]
        [-roles="role1;role2;..."]
        [-email="email1;email2;..."]
        [-privilege="name[;secure-resource-details]]"
        [-separator=privilege="sep_string"]
        [-subseparator=privilege="subsep_string"]
        [-profile="profile_name"]
        [-desc="user_description"]
        [-expired="true|false"]
        [-prevent_change_password="true|false"]
        [-department="department_name"]
        [-cost_center="cost_center"]
        [-line_of_business="line_of_business"]
        [-contact="contact"]
        [-location="location"]
        [-input_file="arg_name:file_path"]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
emcli create_user
        (name="name"
        ,password="password"
        [,type="user_type"]
        [,roles="role1;role2;..."]
        [,email="email1;email2;..."]
        [,privilege="name[;secure-resource-details]]"
        [,separator=privilege="sep_string"]
        [,subseparator=privilege="subsep_string"]
        [,profile="profile_name"]
        [,desc="user_description"]
        [,expired="true|false"]
        [,prevent_change_password="true|false"]
        [,department="department_name"]
        [,cost_center="cost_center"]
        [,line_of_business="line_of_business"]
        [,contact="contact"]
        [,location="location"]
        [,input_file="arg_name:file_path"])

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Administrator name.

- **password**

Administrator password.

- **type**

  Type of User. The default value of this parameter is EM_USER. Possible values for this parameter are:

  – EM_USER

  – EXTERNAL_USER

  – DB_EXTERNAL_USER

- **roles**

  List of roles to grant to this administrator. Currently, the built-in roles include `PUBLIC`.

- **email**

  List of e-mail addresses for this administrator.

- **privilege**

  Privilege to grant to this administrator. You can specify this option more than once in command-line format. The original administrator privileges will be revoked. Specify <secure_resource_details> as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
  name2=resource_column_value2]..]"
  ```

  To retrieve the list of system privileges that do not require resource information, execute the get_supported_privileges command.

- **separator**

  Specify a string delimiter to use between name-value pairs for the value of the privilege option. The default separator delimiter is a semi-colon (;).

- **subseparator**

  Specify a string delimiter to use between name and value in each name-value pair for the value of the privilege option. The default subseparator delimiter is a colon (:).

- **profile**

  Database profile name. It uses DEFAULT as the default profile name.

- **desc**

  User description for the user being added.

- **expired**

  Use this option to expire the password immediately. The default is false.

- **prevent_change_password**

  When set to true, you cannot change your own password. The default is false.

- **department**

  Name of the department of the administrator.

- **cost_center**

  Cost center of the administrator in the organization.

- **line_of_business**

Line of business of the administrator.

- **contact**

  Contact information of the administrator.

- **location**

  Location of the administrator.

- **input_file**

  Allow the administrator to provide the value of any argument in a file. The format of the value will be the name_of_argument:file_path_with_file_name. You can specify this option more than once.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

These examples create an Enterprise Manager administrator named `new_admin`. This administrator has two privileges: the ability to view the job with ID 923470234ABCDFE23018494753091111 and the ability to view the target `host1.example.com:host`. The administrator `new_admin` is granted the `PUBLIC` role.

Example 1 Command-Line

```
emcli create_user
      -name="new_admin"
      -password="oracle"
      -email="first.last@example.com;joe.shmoe@shmoeshop.com"
      -roles="public"
      -privilege="view_job;923470234ABCDFE23018494753091111"
      -privilege="view_target;host1.example.com:host"
```

Example 2 - Scripting and Interactive

```
create_user
      (name="new_admin"
      ,password="oracle"
      ,email="first.last@example.com;joe.shmoe@shmoeshop.com"
      ,roles="public"
      ,privilege="view_job;923470234ABCDFE23018494753091111"
      ,privilege="view_target;host1.example.com:host")
```

These examples make User1 an Enterprise Manager user, which is already created on an external user store like the SSO server. The contents of priv_file are view_target;host1.example.com:host . User1 will have view privileges on the host1.example.com:host target.

Example 3 - Command-Line

```
emcli create_user
      -name="User1"
      -type="EXTERNAL_USER"
      -input_file="privilege:/home/user1/priv_file"
```

Example 4 - Scripting and Interactive

```
create_user
      (name="User1"
      ,type="EXTERNAL_USER"
```

```
      ,input_file="privilege:/home/user1/priv_file")
```

These examples make User1 an Enterprise Manager user, provide a description for the user, and prevent the password from being changed. Only another super administrator can change the password. The profile is set as MGMT_ADMIN_USER_ PROFILE.

Example 5 - Command-Line

```
emcli create_user
      -name="User1"
      -desc="This is temp hire."
      -prevent_change_password="true"
      -profile="MGMT_ADMIN_USER_PROFILE"
```

Example 6 - Scripting and Interactive

```
create_user
      (name="User1"
      ,desc="This is temp hire."
      ,prevent_change_password="true"
      ,profile="MGMT_ADMIN_USER_PROFILE")
```

These examples make User1 an Enterprise Manager user, provide a description for the user, and immediately expire the password. When the user logs in the first time, he/she must change the password.

Example 7 - Command-Line

```
emcli create_user
      -name="User1"
      -desc="This is temp hire."
      -expire="true"
```

Example 8 - Scripting and Interactive

```
create_user
      (name="User1"
      ,desc="This is temp hire."
      ,expire="true")
```

These examples make User1 an Enterprise Manager user, and provide a description, department name, cost center, line of business, contact, and location for the administrator.

Example 9 - Command-Line

```
emcli create_user
      -name="User1"
      -password="oracle"
      -desc="This is temp hire."
      -department="dept1"
      -cost_center="testCostCenter"
      -line_of_business="testLineOfBusiness"
      -contact="contact"
      -location="location"
```

Example 10 - Scripting and Interactive

```
emcli create_user
      (name="User1"
      ,password="oracle"
      ,desc="This is temp hire."
      ,department="dept1"
      ,cost_center="testCostCenter"
      ,line_of_business="testLineOfBusiness"
      ,contact="contact"
      ,location="location")
```

data_transfer

Transfers data from source to target.

Format

```
emcli data_transfer
        -inputFile="File containing properties required for transferring data"
```

Parameters

- inputFile

  Location of file containing properties required for transferring data. The mandatory properties required for this job are:

  SOURCE_LOCATION = Location of the data at the source host.

  SRC_HOST = Source host containing the data.

  SRC_HOST_CREDS = Credentials for the host on which the data is located. If the source host is on OPC, this should be Host SSH Credentials.

  DEST_HOST = Destination host where the data should be copied to.

  DEST_HOST_CREDS = Credentials for the host where the data will be copied to. If the destination host is on OPC, this should be Host SSH Credentials.

  DEST_LOCATION = Location on the destination host where the data should be copied.

Example

The following example transfers data from the source to the target contained in the `data_transfer.props` file:

```
emcli data_transfer
      -input_file=data:/u01/files/data_transfer.props
```

dbimport

Imports data from export dumps to the database target specified.

Format

```
emcli dbimport
       -inputFile="File containing properties for importing data to a database"
```

Parameters

- inputFile

  Location of the file containing properties required for importing data to the database.

Example

The following example imports data from export dumps to the database target specified in the `dbimport.props` file:

```
emcli dbimport
      -inputFile=/u01/files/dbimport.props
```

db_cloud_maintenance

Performs database Cloud maintenance tasks.

db_cloud_maintenance -activateSoftware

Activates the new software of the pool.

Format

```
emcli db_cloud_maintenance -activateSoftware
       -pool_name= "pool_name"
       -pool_type= "pool_type" l
       [-force= "force" ]

[ ] indicates that the parameter is optional.
```

Parameters

- pool_name

  The name of the pool.

- pool_type

  The type of the pool.

- force

  Forcibly activates new members.

Example

The following example forcibly activates new members and activates new software for the Oracle Cloud Zone pool with the name POOL.

```
emcli db_cloud_maintenance -activateSoftware
      -pool_name=POOL
      -pool_type=oracle_cloud_zone
      -force=true
```

db_cloud_maintenance -performOperation

Performs a named operation on a specified pool.

Format

```
emcli db_cloud_maintenance -performOperation
       -name= "name"
       -description= "description"
       -purpose= "purpose"
       -pool_name= "pool_name"
       -pool_type= "pool_type"
        [-start_schedule= "start_schedule"]
        [-end_schedule= "end_schedule" ]
        [-input_file= "data:input_file" ]
        [-target_type= "target_type" ]
        [-target_list= "target_list" ]

[ ] indicates that the parameter is optional.
```

Parameters

- name

The name of the operation.

- description

  The description of the operation.

- purpose

  The purpose of the operation.

- pool_name

  The name of the pool.

- pool_type

  The type of the pool.

- start_schedule

  The scheduled start time. The format for start_schedule is yyyy-MM-dd HH:mm:ss, for example start_schedule="2013-11-11 12:15:30". The default start time is immediate.

- end_schedule

  The scheduled end time. The format for end_schedule is yyyy-MM-dd HH:mm:ss, for example end_schedule="2014-11-11 22:30:00". The default end time is indefinite.

- input_file

  Input data for the maintenance action, for example input_file="data:~/input_files/data.

- target_type

  The default target type is identified based on the purpose. For example, if the purpose is DEPLOY_DB, then the default target type becomes oracle_home.

- target_list

  A comma separated list of targets. The target list is the list of entities based on the target type that is selected. For example, if target_type=rac_database target_list,then the target list is "rac1.example.com,rac2.example.com". The default target_list is based on the purpose. For example if the purpose is DEPLOY_DB, the default target list becomes the list of Oracle homes present in the pool.

Example

The following example performs the Update RAC Database operation for the Oracle Cloud Zone pool with the name POOL.

```
emcli -performOperation
      -name="Update RAC Database "
      -description="Update RAC database Instance"
      -purpose="UPDATE_RACDB"
      -start_schedule="start_time:2014/09/01 00:00"
      -end_schedule="start_time:2014/09/01 13:00"
      -pool_name="POOL NAME"
      -pool_type=oracle_cloud_zone
      -target_type=rac_database
      -target_list="rac1.example.com"
```

db_software_maintenance

Performs database software maintenance tasks.

db_software_maintenance -searchImage

Searches the image based on the filters provided. Use '%' for wildcards.

Format

```
emcli db_software_maintenance -searchImage
      [-name_pattern= "name_pattern"]
      [-version_pattern= "version_pattern"]
      [-description_pattern= "description_pattern"]
      [-owner= "owner"]
      [-target_type= "target_type"]
      [-platform_id= "platform_id"]

[ ] indicates that the parameter is optional.
```

Parameters

- name_pattern

  The name pattern.

- version_pattern

  The version pattern.

- description_pattern

  The description pattern.

- version_pattern

  The version pattern.

- target_type

  The target type.

- platform_id

  The platform id, for example 226 for Linux x86_64.

Example 1

The following example searches the database image for names that contain GI.

```
emcli db_software_maintenance -searchImage
      -name_pattern="%GI%"
```

Example 2

The following example searches the database image for the Linux x86_64 platform (platform id 226).

```
emcli db_software_maintenance -searchImage
      -platform_id="226"
```

db_software_maintenance -updateVersionStatus

Updates the version status of the image.

Format

```
emcli db_software_maintenance -updateVersionStatus
      [-version_id= "version_id"]
       -status= "status"

[ ] indicates that the parameter is optional.
```

Parameters

- version_id

  The version id.

- status

  The status of the version, for example DRAFT, ACTIVE, CURRENT, RESTRICTED.

Example

The following example updates the version of the image with the version ID 02A635AOD8D904A4E05362F3E40ADFD8 to CURRENT.

```
emcli db_software_maintenance -updateVersionStatus
      -version_id=02A635AOD8D904A4E05362F3E40ADFD8
      -status=CURRENT
```

db_software_maintenance -createVersion

Creates a new version in an existing image using an existing software library component.

Format

```
emcli db_software_maintenance -updateVersionStatus
      -version_name= "version_name"
      -image_id= "image_id"
      -external_id= "external_id"
      -status= "status"
```

Parameters

- version_name

  The name of the version.

- image_id

  The ID of the image.

- external_id

  The external ID of the version. For example, it will be the Uniform Resource Name (URN) of the Software Library gold image.

- status

  The status of the version, for example DRAFT, ACTIVE, CURRENT, RESTRICTED.

Example

The following example creates a version, Version1.

```
emcli db_software_maintenance -createVersion
      -version_name="Version1"
      -image_id="01B5F14FD57D7B89E05313B2F00A739F"
      -external_id="oracle:defaultService:em:provisioning:1:cmp:COMP
```

```
      Component:SUB_OracleDB:0191172464DD36B6E05313B2F00AB90A:0.1"
      -status=CURRENT
```

db_software_maintenance -deleteImage

Deletes an image.

Format

```
emcli db_software_maintenance -deleteImage
      -image_id= "image_id"
      [-force= "force"]

[ ] indicates that the parameter is optional.
```

Parameters

- image_id

  The ID of the image to be deleted.

- force

  Deletes forcibly even if the image has subscribed targets.

Example

The following example deletes the image with the ID 01B5F14FD57D7B89E05313B2F00A739F.

```
emcli db_software_maintenance -deleteImage
      -image_id="01B5F14FD57D7B89E05313B2F00A739F
```

db_software_maintenance -createImage

Creates a new image.

Format

```
emcli db_software_maintenance -createImage
       -image_name= "image_name"
       -description= "description"
       -type= "type"
       -target_type= "target_type"
       -version= "version"
       -platform_id= "platform_id"
       -status= "status"
```

Parameters

- image_name

  The name of the image.

- description

  The description of the image.

- type

  The type of the image, for example SWLIB if the image version will be in the Software Library.

- target_type

The target type of the image. For example, if the image is being created to manage single instance Oracle Database then the target type is 'oracle_database.

- version

  The RDBMS version of the product, for example 11.2.0.4.0.

- platform_id

  The platform id, for example 226 for Linux x86_64.

- status

  The image status, for example PRODUCTION.

Example

The following example creates a new image with the name GI_11204.

```
emcli db_software_maintenance -createImage
      -image_name="GI_11204"
      -description="GI_11204"
      -type="SWLIB"
      -target_type=cluster -version=11.2.0.4.0
      -platform_id=226 -status=PRODUCTION
```

db_software_maintenance -getImageSubscriptions

Returns the list of subscribed targets.

Format

```
emcli db_software_maintenance -getImageSubscriptions
      -image_id= "image_id"
```

Parameters

- image_id

  The ID of the image.

Example

The following example returns a list of targets for the image with the ID ID01B5F14FD57D7B89E05313B2F00A739F.

```
emcli db_software_maintenance -getImageSubscriptions
      -image_id="01B5F14FD57D7B89E05313B2F00A739F"
```

db_software_maintenance -getTargetSubscriptions

Returns a list of subscriptions for the specified target.

Format

```
emcli db_software_maintenance -getTargetSubscriptions
      -target_name= "target_name"    Target name
      -target_type= "target_type"    Target type
      [-image_type= "image_type"    Image type]

[ ] indicates that the parameter is optional.
```

Parameters

- target_name

The name of the target.

- target_type

  The target type.

- image_type

  The image type.

Example

The following example returns a list of subscriptions for the Oracle Cloud Zone target with the name POOL NAME.

```
emcli db_software_maintenance -getTargetSubscriptions
      -target_name="POOL NAME"
      -target_type="oracle_cloud_zone"
```

db_software_maintenance -getVersions

Returns a list of the versions for the specified image.

Format

```
emcli db_software_maintenance -getVersions
      -image_id= "image_id"
      [-version_status= "version_status"]

[ ] indicates that the parameter is optional.
```

Parameters

- image_id

  The name of the image.

- version_status

  The status filter for the version.

Example

The following example returns a list of versions for the image with theID 01B5F14FD57D7B89E05313B2F00A739F.

```
emcli db_software_maintenance -getVersions
      -image_id="01B5F14FD57D7B89E05313B2F00A739F"
      -version_status=CURRENT
```

db_software_maintenance -getImages

Returns the list of images present in system.

Format

```
emcli db_software_maintenance -getImages
```

Example

The following example returns a list of images present in system.

```
emcli db_software_maintenance -getImages
```

db_software_maintenance -subscribeTarget

Creates new target or modifies the target subscription.

Format

```
emcli db_software_maintenance -subscribeTarget
      -target_name= "target_name"
      -target_type= "target_type"
      -[parent_target_name= "parent_target_name"]
      -[parent_target_type= "parent_target_type"]
      -image_id= "image_id"
      -[version_id= "version_id'

[ ] indicates that the parameter is optional.
```

Parameters

- target_name

  The name of the target.

- target_type

  The target type.

- parent_target_name

  The parent target name.

- parent_target_type

  The parent target type.

- image_id

  The image id.

- version_id

  The version id.

Example

The following example modifies the Oracle Cloud Zone target with the name POOL NAME.

```
emcli db_software_maintenance -subscribeTarget
      -target_name="POOL NAME"
      -target_type=oracle_cloud_zone
      -image_id=FE55AD7AB28974EFE04313B2F00AD4A0
```

db_software_maintenance -unsubscribeTarget

Unsubscribes the specified target.

Format

```
emcli db_software_maintenance -subscribeTarget
      -target_name= "target_name"
      -target_type= "target_type"
      -image_id= "image_id"
      [-version_id= "version_id']

[ ] indicates that the parameter is optional.
```

Parameters

- target_name

  The name of the target.

- target_type

  The target type.

- image_id

  The image id.

Example

The following example unsubscribes the Oracle Cloud Zone target with the name POOL NAME and the image ID FE55AD7AB28974EFE04313B2F00AD4A0.

```
emcli db_software_maintenance -unsubscribeTarget
      -target_name="POOL NAME"
      -target_type=oracle_cloud_zone
      -image_id=FE55AD7AB28974EFE04313B2F00AD4A0
```

db_software_maintenance -getSubscriptionsForContainer

Returns the subscriptions for the container target, for example database pool.

Format

```
emcli db_software_maintenance -getSubscriptionsForContainer
      -image_id= "image_id"
      -target_name= "target_name"
      -target_type= "target_type"
```

Parameters

- image_id

  The image id.

- target_name

  The name of the target in Oracle Enterprise Manager.

- target_type

  The target type.

Example

The following example returns the subscriptions for the Oracle Cloud Zone target with the name POOL NAME and the image ID FE55AD7AB28974EFE04313B2F00AD4A0.

```
emcli db_software_maintenance -getSubscriptionsForContainer
      -target_name="POOL NAME"
      -target_type=oracle_cloud_zone
      -image_id=FE55AD7AB28974EFE04313B2F00AD4A0
```

db_software_maintenance -createSoftwareImage

Creates a new software image for the specified the Oracle home. The createSoftwareImage verb either takes data from a text file or uses the getInputVariableList command.

Format

```
emcli db_software_maintenance -createSoftwareImage
      [-data= "data"]
      [-getInputVariableList= "getInputVariableList"]

[ ] indicates that the parameter is optional.
```

Parameters

- data

  The path of the txt input file.

- getInputVariableList

  Provides the list of variables to be specified in the input file.

Example 1

To get the list of all of the parameters to be passed inside the data file, run the following command:

```
emcli db_software_maintenance -createSoftwareImage
      -getInputVariableList=true
```

Example 2

The following example creates a new image and a version.

```
emcli db_software_maintenance -createSoftwareImage
      -data="input_file=data:"/home/user/input_rac"
```

In this example, the contents of the input_rac file are:

- IMAGE_NAME=DbGoldImage
- IMAGE_DESCRIPTION=Gold Image for 11g db
- REF_TARGET_NAME=ORACLE_HOME
- IMAGE_SWLIB_LOC=Oracle Home Provisioning Profiles/11.2.0.3.0/linux_x64
- REF_HOST_CREDENTIALS=ZONE_CREDS:TESTSUPERADMIN
- WORKING_DIRECTORY=/tmp
- STORAGE_TYPE_FOR_SWLIB=OmsShared
- STORAGE_NAME_FOR_SWLIB=swlib
- VERSION_NAME=Version1

Example 3

The following example creates a new version alone into an existing image.

```
emcli db_software_maintenance -createSoftwareImage
      -data="input_file=data:"/home/user/input_rac"
```

In this example, the contents of the input_rac file are:

- IMAGE_ID=01B5F14FD57D7B89E05313B2F00A739F

- REF_TARGET_NAME=ORACLE_HOME
- IMAGE_SWLIB_LOC=Oracle Home Provisioning Profiles/11.2.0.3.0/linux_x64
- REF_HOST_CREDENTIALS=ZONE_CREDS:TESTSUPERADMIN
- WORKING_DIRECTORY=/tm
- STORAGE_TYPE_FOR_SWLIB=OmsShared
- STORAGE_NAME_FOR_SWLIB=swlib
- VERSION_NAME=Version1

define_diagcheck_exclude

Defines a diagnostic check exclusion with regard to groups and checks to exclude.

Format

```
emcli define_diagcheck_exclude
        -target_type="type"
        -exclude_name="name"
        { [-excl_group="diag_group" ]*
          [-excl_check="diag_check" ]* |
         -input_file=excl_def:<complete_path_to_file> }

[ ] indicates that the parameter is optionalis optional
```

Parameters

- **target_type**

  Type of target.

- **exclude_name**

  Name to use for the exclusion.

- **excl_group**

  Group of diagchecks to exclude.

- **excl_check**

  Name of diagcheck to exclude.

- **input_file**

  For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

delete_assoc

Deletes target association instances.

Format

Standard Mode

```
emcli delete_assoc
       -assoc_type="association type"
       -source="target_name:target_type"
       -dest="target_name1:target_type1[;target_name2:target_type2..]"
       [-separator="separator:attribute_name:character"]
       [-subseparator="subseparator:attribute_name:character"]
```

Interactive (Script) Mode

```
delete_assoc(
      assoc_type="association type"
      ,source="target_name:target_type"
      ,dest="target_name1:target_type1[;target_name2:target_type2..]"
      [,separator="separator:attribute_name:character"
      [,subseparator="subseparator:attribute_name:character"]
      )
```

`[ ]` indicates that the parameter is optional.

Parameters

- **assoc_type**

  Association type.

- **source**

  Target name and target type of the source target.

- **dest**

  Target name and target type of the destination targets.

- **separator**

  By default, multi-value input attributes use a semicolon (;) as a separator. Specifying this option overrides the default separator value.

  Example: `separator="<attribute_name=sep_char>"` where *attribute_name* is name of the attribute for which you want to override the separator character, and *sep_char* is the new separator character.

  Example: `separator="att=#"`

- **subseparator**

  By default, multi-value input attributes use a colon (:) as a subseparator. Specifying this option overrides the default subseparator value.

  Example: `subseparator="<attribute_name=sep_char>"` where *attribute_name* is name of the attribute for which you want to override the separator character, and *sep_char* is the new subseparator character.

  Example: `separator="att=#"`

Output

Exit Codes

0 indicates that the verb processing was successful.

Non-zero values indicate that the verb processing was not successful.

Example

This example deletes associations of type cluster_contains from target "abc_cluster:cluster" to targets "def.oracle.com:host" and "ghi.oracle.com:host":

```
emcli delete_assoc
      -assoc_type="cluster_contains"
      -source="abc_cluster:cluster"
      -dest="def.oracle.com:host;ghi.oracle.com:host"
```

delete_bda_cluster

Deletes the specified Hadoop cluster target and all its children. If this is the last cluster in the BDA target, also deletes the BDA target and all its children.

If Hadoop clusters are spread across multiple racks, performs deletions across the BDA rack. If other clusters exist within the BDA rack, relocates any shared targets before deleting the Hadoop cluster target.

Format

```
emcli delete_bda_cluster
        -cluster="cluster_name"
```

Parameters

- **cluster**

  Name of the cluster to be deleted.

Examples

The following example deletes the acme cluster target and all of its children. If acme is the last cluster in the BDA rack, deletes the rack and all of its children. If there are other clusters in the rack, relocates shared targets before deleting the cluster.

```
emcli delete_bda_cluster
        -cluster="acme"
```

delete_blackout

Deletes a blackout that has already ended or has been fully stopped. You cannot delete a blackout that is either in progress or currently scheduled. You need to first run `stop_blackout`.

Format

```
emcli delete_blackout
      -name="name"
      [-createdby="blackout_creator"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the blackout to delete.

- **createdby**

  Enterprise Manager user who created the blackout. The default is the current user. The `SUPER_USER` privilege is required to delete a blackout created by another user.

Examples

Example 1

This example deletes blackout `backup_monthly` created by the current user.

```
emcli delete_blackout -name=backup_monthly
```

Example 2

This example deletes blackout `db_maintenance` that was created by Enterprise Manager administrator `sysadmin2`. The current user must either be user `sysadmin2` or a user with the `SUPER_USER` privilege.

```
emcli delete_blackout -name=db_maintenance -createdby=sysadmin2
```

delete_charge_item

Deletes the custom charge item from Chargeback.

Format

```
emcli delete_charge_item
        -target_type="target_type"
        -item_name="item_name"
```

Parameters

- **target_type**

  Target type associated with the custom charge item.

- **item_name**

  Name of the custom charge item to be deleted.

Examples

Example 1

This example deletes a custom charge item named total_proc associated with the host target type:

```
emcli delete_charge_item
        -target_type="host"
        -item_name="total_proc"
```

Example 2

This example deletes a custom charge item named custom_config associated with the Oracle Database target type:

```
emcli delete_charge_item
        -target_type="oracle_database"
        -item_name="custom_config"
```

delete_cloud_service_instances

Deletes the cloud service instances based on the specified filter.

Format

```
emcli delete_cloud_service_instances
      -user="username"
      [-family="family"]
      [-type="service type"]

[ ] indicates that the parameter is optional
```

Parameters

- **user**

  Identifies the name of the user to be used for filtering the service instances that are to be deleted.

- **family**

  Identifies the service family name to use to filter cloud requests.

- **type**

  Identifies the Service Type to be used for filtering the service instances that are to be deleted.

Examples

Example 1

This example deletes all cloud instances whose owner is the specified user (`user1`):

```
emcli delete_cloud_service_instances -user="user1"
```

Example 2

This example deletes all cloud instances that are owned by a specified user (`user1`) and belong to a specified service family (`family1`):

```
emcli delete_cloud_service_instances -user="user1" -family="family1"
```

Example 3

This example deletes all cloud instances that are owned by a specified user (`user1`) and belong to a specified service type (`type1`):

```
emcli delete_cloud_service_instances -user="user1" -type="type1"
```

Example 4

This example deletes all cloud instances that are owned by a specified user (`user1`), belong to a specified service family (`family1`), and belong to a specified service type (`type1`):

```
emcli delete_cloud_service_instances -user="user1" -family="family1" -type="type1"
```

delete_cloud_user_objects

Deletes cloud user objects including cloud service instances and requests.

Format

```
emcli delete_cloud_user_objects
      -user="username"
      [-purge]
      [-force]

[ ] indicates that the parameter is optional
```

Parameters

- **user**

  Identifies the name of the user to be used for filtering user objects.

- **purge**

  Sets a flag to purge the completed cloud service requests. Default is **false** unless this parameter is used.

- **force**

  Sets a flag to attempt to cancel In Progress requests. Depending on the job state, there may be some manual cleanup required.

  USE WITH CAUTION. There is no way to undo the operation once started.

Examples

Example 1

Delete all cloud objects owned by a specified user (`user1`) and cancel all scheduled requests:

```
emcli delete_cloud_user_objects -user="user1"
```

Example 2

Delete all cloud objects owned by a specified user (`user1`), cancel all scheduled requests, and purge all completed requests:

```
emcli delete_cloud_user_objects -user="user1" -purge
```

Example 3

Delete all cloud objects owned by a specified user (`user1`), cancel all scheduled requests, and cancel all In Progress requests:

```
emcli delete_cloud_user_objects -user="user1" -force
```

delete_credential_set

Deletes a credential set. Only Enterprise Manager Super Administrators can delete credential sets. Out-of-box credential sets cannot be deleted.

Format

```
emcli delete_credential_set
      -set_name="set_name"
      -target_type="ttype"
```

Parameters

- **set_name**

  Credential set name to be deleted.

- **target_type**

  Target type of the credential set.

Examples

This example deletes a credential set named Old_Credential_Set.

```
emcli delete_credential_set
        -set_name=Old_Credential_Set
        -target_type=host
```

delete_custom_plugin_update

Deletes the custom plug-in update for a plug-in. All subsequent plug-in deployments will use the latest applicable version or revision available with Enterprise Manager Self Update.

Does not automatically redeploy to Management Agents on which this custom plug-in update was previously deployed. Applies only to subsequent plug-in deployments.

Format

```
emcli delete_custom_plugin_update
      -plugin="<plugin_id>:<plugin_version>:<plugin_revision>"
```

Parameters

- **plugin**

  ID, version, and revision of the plug-in. To view the version and revision of a plug-in, run 'emcli list_custom_plugin_updates'.

Example

The following example deletes the custom plug-in update of the 12.1.0.2.0 version of the `oracle.sysman.db2` plug-in.

```
emcli delete_custom_plugin_update
      -plugin="oracle.sysman.db2:12.1.0.2.0"
```

delete_database

Deletes a database and target from Oracle Enterprise Manager.

Format

```
emcli delete_database
      -inputFile="File containing properties required for deleting a database"
```

Parameters

- inputFile

  The location and name of the file containing the properties required for deleting the database.

Example

The following example deletes a database using the parameters contained in the `/u01/files/delete_database.props` file:

```
emcli delete_database
      -inputFile=/u01/files/delete_database.props
```

delete_database_size

Deletes the database size created with the create_database_size verb.

Format

```
emcli delete_database_size
      -name="<Existing size name>"
```

Parameters

- name

  The name of the existing database size.

Example

The following example deletes the database size names Small.

```
emcli delete_database_size
      -name=Small
```

delete_dbaas_quota

Deletes the database quota for an SSA user role.

Format

```
emcli delete_dbaas_quota
       -role_name="<SSA user role  name>"

[ ] indicates that the parameter is optional.
```

Parameters

- **role_name**

  Name of the SSA user role for which the quota is to be deleted.

Example

This example deletes the quota for My Role:

```
emcli delete_dbaas_quota
       -role_name="My Role"
```

delete_dbprofile

Deletes an existing database profile component.

Format

```
emcli delete_dbprofile
     -comp_loc="Database Profile component location and name in software library"
     [-version="Database Profile component version name"]

[ ] indicates that the parameter is optional.
```

Parameters

- comp_loc

  Combination of database profile component location and name.

- version

  Database profile component version name.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example deletes Database profile component with the profile name "RMAN_Profile", version "RMAN_Backup_10_04_14_12_40_PM" and location "Database Provisioning Profiles/11.2.0.4.0/linux_x64".

```
emcli delete_dbprofile -comp_loc="Database Provisioning Profiles/11.2.0.4.0/linux_
x64/RMAN_Profile" -version="RMAN_Backup_10_04_14_12_40_PM"
```

delete_diag_snapshot

Deletes a specified diagnostic snapshot.

Format

```
emcli delete_diag_snapshot
        -name="<diag_snapshot_name>"
        [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the diagnostic snapshot to be deleted. Ensure that the diagnostic snapshot exists for the specified name.

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Examples

This example deletes a diagnostic snapshot with the name of Snapshot1 from Cloud Control.

```
emcli delete_diag_snapshot
      -name="Snapshot1"
```

delete_fmw_profile

Deletes a Fusion Middleware provisioning profile from software library.

Format

```
emcli delete_fmw_profile
      -location="Profile Location"
       -source="source"
       -dest="association type"
       [-separator="separator:attribute_name:character"]
       [-subseparator="subseparator:attribute_name:character"]

[ ] indicates that the parameter is optional.
```

Parameters

- location

  The complete software library path to the profile. Use the list_fmw_profiles verb to identify the complete path.

  Note: The name and owner parameters must be used together.

Example

The following example deletes the Fusion Middleware profile "MyProfile" from software library.

```
emcli delete_fmw_profile
      -location="Fusion Middleware Provisioning/Profiles/MyProfile"
```

delete_group

Deletes a group. Deleting a non-existent group generates the error "Group X does not exist."

Format

```
emcli delete_group
      -name="name"
      [-type=<group>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the group to delete.

- **type**

  Group type: group. Defaults to "group".

Examples

Example 1

This example removes the group `payroll_group` that consists of database target types.

```
emcli delete_group -name=payroll_group
```

Example 2

This example removes the group `my_hosts` that consists of host target types.

```
emcli delete_group -name=my_hosts
```

Example 3

This example removes the group `my_group` that consists of mixed target types.

```
emcli delete_group -name=my_group
```

delete_incident_record

Deletes one or more open incidents based on the provided IDs, up to a maximum of 20 incidents. This removes any *association* with the underlying events and annotates them accordingly. Incident deletion **does not** remove the actual underlying events: These events will remain open.

Privilege Requirements: Only users with Manage Incident privilege can delete the incident.

By default, incidents that have workflow attributes (such as Escalation, Priority, Resolution Status, Acknowledgement, Owner Assignment, or Suppression) set to non-default values will not be deleted unless the `-force` option is used.

Closed incidents, diagnostic (ADR) incidents, and incidents with tickets created cannot be deleted.

The status of each incident deletion is displayed upon command execution.

Format

```
emcli delete_incident_record
-incident_number_list="Comma-separated list of incident numbers"
[-force]
[-preview]

[ ] indicates that the parameter is optional
```

Parameters

- **incident_number_list**

  Comma-separated list of incident numbers (up to 20) to be deleted.

- **force**

  Deletes incidents without checking for their non-default workflow values.

- **preview**

  Displays whether or not specified incidents (by incident number) can be deleted.

Examples

Example 1

This example displays whether or not incidents 173, 1886, 32, 5, and 853 can be deleted.

The command output is shown below.

```
emcli delete_incident_record -incident_number_list="173,1886,32,5,853" -preview

=========
RESULTS
=========
=> Incident 173 can be deleted.

=> Incident 1886 can only be deleted using the -force option, as one or more
incident workflow attributes have been been used.

=> Incident 32 cannot be deleted because there is ticket attached with the
```

```
incident.

=> Incident 5 cannot be deleted because user AdminX does not have at least a
manage incident privilege.

=> Incident 853 can be deleted.
```

Example 2

This example deletes incidents 178, 1886, and 853 without checking for non-default incident workflow values.

The command output is shown below.

```
emcli delete_incident_record -incident_number_list="173,1886,853" -force

=========
RESULTS
=========

=> Incident 173 has been successfully deleted.

=> Incident 1886 has been successfully deleted.

=> Incident 853 has been successfully deleted.
```

delete_instance

Deletes a stopped or completed deployment instance. An instance can only be deleted when its status is stopped, completed, or completed with an error.

Format

```
emcli delete_instance
      [-instance=<instance_guid>]
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

Examples

Example 1

```
emcli delete_instance -instance=16B15CB29C3F9E6CE040578C96093F61
```

Example 2

```
emcli delete_instance -exec=2B15CB29C3F9E6CE040578C96093F16
```

delete_job

Deletes a job or a set of jobs matching the filter criteria. A job cannot be deleted if any of its executions are active. All executions must be in one of the following states:

ABORTED, FAILED, COMPLETED, STOPPED, SKIPPED

Use the `get_jobs` verb to obtain a list of existing jobs along with their job IDs and statuses.

Format

```
emcli delete_job
      [-job_id="ID1;ID2;..."]
      [-name="job name pattern"]
      [-owner="job owner"]
      [-type="job type"]
      [-targets="target name:target type"]
      [-input_file=property_file:"filename"]
      [-preview]

[ ] indicates that the parameter is optional
```

Parameters

- **job_id**

  Semi-colon (;) separated list of job(s) to delete.

  NOTE: This filter cannot be used with other filters.

- **name**

  Name or pattern of the job to delete. To uniquely identify the job, the current user is used.

- **owner**

  Owner of the job(s).

- **type**

  Job type of the job(s).

- **targets**

  Target name and target type of the job(s) to be deleted.

- **input_file**

  The properties for filtering jobs can be specified in "filename". Any jobs matching all the specified filter criteria are deleted. You must specify at least one filter, and the logged in administrator must have the necessary privileges on the matching jobs.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **preview**

  Lists only the jobs to be deleted. In the list of options you specify, if -preview is not one of the options, jobs are deleted, and then these jobs will be listed. If -preview is one of the options, the identical list is shown, but no jobs are deleted.

Examples

Example 1

This example deletes an existing job with the job ID 12345678901234567890123456789012.

```
emcli delete_job -job_id=12345678901234567890123456789012
```

Example 2

This example deletes all jobs of type "Backup."

```
emcli delete_job -type=Backup
```

Example 3

This example stops and deletes a job named MY_JOB owned by the logged in administrator. You can use the stop and delete pattern to delete active jobs.

```
emcli stop_job -name=my_job
emcli delete_job -name=my_job
```

delete_library_job

Deletes a library job you created using the create_library_jobs command.

Format

```
emcli delete_library_job
      -name=<"library_job_name">
      [-owner=<"library_job_owner">]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the library job.

- **owner**

  Owner of the library job if different from the current logged-in EM CLI administrator.

Examples

Example 1

This example deletes the library job "libjob1" owned by the current logged-in Enterprise Manager administrator.

```
emcli delete_library_job -name=libjob1
```

Example 2

This example deletes the library job "libjob2" owned by the Enterprise Manager administrator "emadmin1."

```
emcli delete_library_job -name=libjob2 -owner=emadmin1
```

delete_metric_promotion

Deletes a promoted metric.

Format

```
emcli delete_metric_promotion
      -name=<service_target_name>
      -type=<service_target_type>
      [-category=<usage/performance/business>]
      [-promotedMetricName=<promoted_metric]
      [-promotedMetricColumn=<promoted_metric_column>]
      -promotedMetricKey=<key_value_of_promoted_metric>

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the service target.

- **type**

  Name of the service type.

- **category**

  Defines whether the promoted metric is a usage or a performance metric of a service. This determines the promoted metric name and metric column. If you do not specify this, you must specify the `promotedMetricName` and `promotedMetricColumn`.

- **promotedMetricName**

  Promoted metric name. This is optional if you specify the category .

- **promotedMetricColumn**

  Promoted metric column. This is optional if you specify the category .

- **promotedMetricKey**

  Determines the key value of the promoted metric. It is equivalent to the displayed name of the promoted metric in the user interface.

Examples

This example deletes the promoted performance metric with the key value `mymetric1` on the service `MyTarget`.

```
emcli delete_metric_promotion -name='MyTarget' -type='generic_service'
      -category=Performance -promotedMetricKey=mymetric1
```

delete_named_credential

Deletes an existing named credential.

Format

```
emcli delete_named_credential
        -cred_owner=<owner>
        -cred_name=<name>
```

Parameters

- **cred_owner**

  Credential owner.

- **cred_name**

  Required credential name. This does not support wild cards.

delete_operation_plan

Deletes the specified operation plan from a Site Guard configuration.

Format

```
emcli delete_operation_plan
      -name=<plan_name>
```

Parameters

- **name**

  Name of the operation plan you want to delete.

Example

```
emcli delete_operation_plan
      -name="BISystem1-switchover"
```

delete_patch_plans

Deletes patch plans.

Format

```
emcli delete_patch_plans
        -name="plan_names"

[ ] indicates that the parameter is optional
```

Parameters

name

Specifies the names of the patch plans that you want to delete. Use a comma as a separator if you want to specify multiple patch plans for this parameter.

Examples

The following example deletes the patch plans plan_1, plan_2, and plan_3:

```
emcli delete_patch_plans -name="plan_1,plan_2,plan_3"
```

delete_paas_zone

Deletes a PaaS Infrastructure Zone. A PaaS Infrastructure Zone cannot be deleted if an existing software pool is associated with it.

Format

```
emcli delete_paas_zone
        -name="<name of PaaS Zone>

[ ] indicates that the parameter is optional.
```

Parameters

- **name**

  Name of the existing PaaS Infrastructure Zone.

Example

This example deletes the PaaS Infrastructure Zone with the name My PaaS Zone:

```
emcli delete_paas_zone
        -name="My PaaS Zone"
```

delete_patches

Deletes patches from the software library.

Format

```
emcli delete_patches
      -patch_name=<patch_name>
      -release=<release_id>
      -platform=<platform_id>
```

Parameters

- **patch_name**

  Patch number.

- **release**

  Patch release ID.

- **platform**

  Patch platform ID.

Example

```
emcli delete_patches -patch_name=13741363 -release=80112310 -platform=226
```

See Also

create_patch_plan
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

delete_pluggable_database

Deletes pluggable databases (PDBs).

Format

```
emcli delete_pluggable_database
      -cdbTargetName="CDB_of_target_PDBs"
      -cdbTargetType="CDB_target_type"
      -cdbHostCreds="CDB_host_credentials"
      -cdbTargetCreds="CDB_target_credentials"
      -pdbName="PDB_names"
      [-cdbHostPrivCreds="CDB_host_privileged_credentials"]
      [-ignoreStorageWarnings]
```

`[ ]` indicates that the parameter is optional.

Parameters

- **cdbTargetName**

  Target container database (CDB) that contains the PDBs that you want to delete. Ensure that the target CDB you specify is a valid target in Enterprise Manager.

- **cdbTargetType**

  Database type of the target CDB, which can be `oracle_database`, `rac_database`, and so on.

- **cdbHostCreds**

  Credentials for the host on which the target CDB is located.

- **cdbTargetCreds**

  Credentials for the target CDB.

- **pdbName**

  Names of the PDBs that you want to delete. Ensure that you separate the names using a comma.

- **cdbHostPrivCreds**

  Privileged credentials for the host on which the Snap Clone storage mount points are located. Note that this parameter is required only if you are deleting PDBs that were created using Snap Clone.

- **ignoreStorageWarnings**

  Ignore any storage warnings that may be generated while deleting PDBs that were created using Snap Clone.

Examples

This example deletes the `test_pdb` PDB, which is a part of `test_CDB`, an Oracle single-instance CDB, using `HOST_CREDS` as the CDB host credentials and `DB_CREDS` as the CDB target credentials:

```
emcli delete_pluggable_database -cdbTargetName=test_database
-cdbTargetType=oracle_database -pdbName=test_pdb -cdbHostCreds=HOST_CREDS
-cdbTargetCreds=DB_CREDS
```

delete_pool verb

Deletes a software pool. A software Pool cannot be deleted if there is an existing service template associated with it.

Format

```
emcli delete_pool
       -name="<software pool name>"
       -target_type="<software pool target type>"

[ ] indicates that the parameter is optional.
```

Parameters

- **name**

  The name of the existing software pool.

- **target_type**

  The target type of the existing software pool.

Example

The following example deletes the software pool My Pool:

```
emcli delete_pool
       -name="My Pool"
       -target_type="mwaas_zone"
```

delete_privilege_delegation_settings

Deletes a privilege delegation setting template.

Format

```
emcli delete_privilege_delegation_settings
      -setting_names="setting_name1;setting_name2;setting_name3;"
```

Parameters

- **setting_names**

  Name of the settings you want to delete.

Example

This example deletes the privilege settings for the names `setting_name1`, `setting_name2`, and `setting_name3`.

```
emcli delete_privilege_delegation_settings
        -setting_names="sudo_setting1;sudo_setting2;pbSetting1
```

delete_resolution_state

Deletes an existing resolution state. You typically use this command for resolution states that are no longer used. You need to also specify an alternative resolution state in case there are any references to the state. In this case, the references are changed to this alternative state. This action might require some time.

Only a super administrator can execute this command. A success message is reported if the command is successful. An error message is reported if the deletion fails.

> **Note:** No notifications are sent for any incidents or problems updated in this process.

Format

```
emcli delete_resolution_state
      -label="label of the state to be deleted"
      -alt_res_state_label="alternative resolution state"
```

Parameters

- **label**

  Label of the state to be deleted.

- **alt_res_state_label**

  Alternative state to be used.

Examples

This example deletes the resolution state "Waiting for SR" and replaces any references to this state with the state "Work in Progress".

```
emcli delete_resolution_state  -label="Waiting for SR" -alt_res_state_label="Work
in Progress"
```

delete_role

Deletes an existing Enterprise Manager administrator role.

Format

```
emcli delete_role
      -name="role_name"
```

Parameters

- **name**

  Role name.

Examples

This example deletes the role name `existing_role`.

```
emcli delete_role -name="existing_role"
```

delete_service_template

Deletes a service template.

Format

```
emcli delete_service_template
-name="<service template name>"]
-service_family="<service family name>"]

[ ] indicates that the parameter is optional.
```

Parameters

- **name**

  Name of the existing service template.

- **service_family**

  Service family to which the service template belongs; for example, DBAAS for database and MWAAS for middleware.

Example

This example deletes the service template with name template2 and service family MWAAS:

```
emcli delete_service_template
       -name="Middleware service Template August"
       -service_family="MWAAS"
```

delete_siebel

Deletes one or more Siebel Enterprise instances and their associated targets, such as Siebel servers, component groups, components, work flows, and so on.

Format

```
emcli delete_siebel
      -enterprise=<Siebel_enterprise_1>,<Siebel_enterprise_2>
      [-out_file='<output_file>']
      [<-debug>]

[ ] indicates that the parameter is optional
```

Parameters

- **enterprise**

  Target name of the Siebel enterprise as seen in the Enterprise Manager console. If multiple enterprises need to be deleted at the same time, provide a comma-separated (,) value.

- **out_file**

  Fully-qualified path of the output file. The output of the command is redirected to this file.

  If you include this option, the list of deleted targets are printed in the file. If you do not include this option, the list is printed on the console directly.

- **debug**

  Executes in verbose mode and generates debug log messages in the output.

Examples

This example deletes the Siebel Enterprise instances from Cloud Control. The output of the command is redirected to the deletion_output.txt file.

```
emcli delete_siebel
      -enterprise=SBA80_ent1.example.com,SBA78_ent2.us.example.com
      -out_file='c:\emcli\deletion_output.txt'
```

delete_siteguard_configuration

Deletes the Site Guard configuration. The entire configuration (scripts, credential associations, site associations, operation plans) pertaining to the specified system and all the associated standby systems are deleted.

Format

```
emcli delete_siteguard_configuration
      -primary_system_name=<name> | -standby_system_name=<name>
```

Parameters

- **primary_system_name**

  Name of the primary system. Specify either primary_system_name or standby_system_name.

- **standby_system_name**

  Name of the standby system.

Examples

Example 1

```
emcli delete_siteguard_configuartion
      -primary_system_name="BISystem1"
```

Example 2

```
emcli delete_siteguard_configuration
      -standby_system_name="BISystem2"
```

See Also

create_siteguard_configuration

get_siteguard_configuration

delete_siteguard_credential_association

Deletes the credential association from the Site Guard configuration.

Format

```
emcli delete_siteguard_credential_association
      -system_name=<name>
      [-target_name=<name>]
      -credential_type=<type>

{ } indicates that the parameter is optional
```

Parameters

- **system_name**

  Name of the system.

- **target_name**

  Name of the target.

- **credential_type**

  Type of the credential, which can be HostNormal, HostPrivileged, WLSAdmin, or DatabaseSysdba.

Examples

Example 1

```
emcli create_siteguard_credential_association
      -system_name="BISystem1"
      -credential_type="HostNormal"
      -credential_name="HOST-SGCRED"
      -credential_owner="sysman"
```

Example 2

```
emcli create_siteguard_credential_association
      -system_name="BIsystem1"
      -target_name="database-instance"
      -credential_type="HostNormal"
      -credential_name="HOST-DBCRED"
      -credential_owner="sysman"
```

See Also

create_siteguard_credential_association

update_siteguard_credential_association

get_siteguard_credential_association

delete_siteguard_lag

Updates the limit for Apply lag and Transport lag for all or selected databases of the system.

Format

```
emcli delete_siteguard_lag
        [-system_name="name_of_the_system"]
        [-target_name="name_of_the_database"]
        [-property_name="lag_type"]

[ ] indicates that the parameter is optional
```

Parameters

- **system_name**

  Name of the system whose lag limit property you want to update.

- **target_name**

  Name of the target database whose lag limit property you want to update.

- **property_name**

  Name of the lag property. Valid values for this parameter are `ApplyLag` and `TransportLag`.

Examples

Example 1

This example deletes the `ApplyLag` property on all of the databases configured on `austin-system`:

```
emcli delete_siteguard_lag
        -system_name="austin-system"
        -property_name="ApplyLag"
```

Example 2

The following example deletes the `TransportLag` property on the database `OID-db` configured on `austin-system`:

```
emcli delete_siteguard_lag
        -system_name="austin-system"
        -target_name="OID_db"
        -property_name="TransportLag"
```

delete_siteguard_script

Deletes the specified script from the Site Guard configuration.

Format

```
emcli delete_siteguard_script
      -script_id=<script_id>
```

Parameters

- **script_id**

  ID associated with the script.

Examples

```
emcli delete_siteguard_script
      -script_id="10"
```

See Also

create_siteguard_script

get_siteguard_scripts

delete_siteguard_script_hosts

Deletes the host or hosts associated with a given script.

Format

```
emcli delete_siteguard_script_hosts
      -script_id=<script_id>
      -host_name=<name1;name2;...>
```

Parameters

- **script_id**

  ID associated with the script.

- **host_name**

  Name of the host where this script will be run. You can specify this parameter more than once.

Examples

```
emcli delete_siteguard_script_hosts
      -script_id="10"
      -host_name="BIHOST1"
```

Output Columns

Step Number, Operation Name, Target Name, Target Host, and Error Mode

See Also

create_siteguard_script

add_siteguard_script_hosts

delete_sla

Deletes one or more SLAs for a target.

Format

```
emcli delete_sla
    -targetName=<target_name>
    -targetType=<target_type>
    -slaName=<SLA_name>
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **slaName**

  Name of the SLA.

Example

This example deletes the SLA with the name 'gold_sla' from the target.

```
emcli delete_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla'
```

delete_system

Deletes a system.

Format

```
emcli delete_system
      -name="name"
      [-type=<generic_system>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the system to delete.

- **type**

  System type: generic_system. Defaults to "generic_system".

Examples

This example deletes the system `my_system`.

```
emcli delete_system -name=my_system
```

delete_target

Deletes a specified target from the Enterprise Manager Cloud Control monitoring framework. Deleting a target removes it from the Management Repository and does not physically remove the target itself.

You can use the `get_targets` verb to obtain a list of available targets and their respective types.

Format

```
emcli delete_target
      -name=<name>
      -type=<type>
      [-delete_monitored_targets]
      [-async]
      [-delete_members]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name.

- **type**

  Target type.

- **delete_monitored_targets**

  Deletes the targets monitored by the specified Management Agent. This is only applicable with the oracle_emd target type.

- **async**

  Deletes the target asynchronously.

- **delete_members**

  Deletes all the members of the target as well.

Examples

Example 1

This example deletes the `oracle_database` target with the name `database`.

```
emcli delete_target
      -name="database"
      -type="oracle_database"
```

Example 2

This example deletes the Agent named test.example.com:1836 and all of its monitored targets. The Agent must be marked UNREACHABLE in Enterprise Manger Cloud Control to perform this operation.

```
emcli delete_target
      -name="test.example.com:1836"
      -type="oracle_emd"
      -delete_monitored_targets
```

```
      -async
```

Example 3

This example deletes the example_ias_farm target with the name "farm01_base_domain" and all of its members, such as domain, clusters, servers, application deployments, and so forth.

```
emcli delete_target
      -name="farm01_base_domain"
      -type="example_ias_farm"
      -delete_members
```

delete_test

Deletes a Services test along with its constituent steps and step groups.

Format

```
emcli delete_test
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **testname**

  Name of the test.

- **testtype**

  Type of test.

Example

This example deletes an HTTP test name MyTest for the generic_service target name MyTarget.

```
emcli delete_test -name='MyTarget' -type='generic_service'
                  -testname='MyTest' -testtype='HTTP'
```

delete_test_threshold

Deletes a test threshold.

Format

```
emcli delete_test_threshold
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>
      -metricName=<metric_name>
      -metricColumn=<metric_column>
      [-beaconName=<beacon_name>]
      [-stepName=<step_name>]
      [-stepGroupName=<stepgroup_name>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **testname**

  Name of the test.

- **testtype**

  Type of test.

- **metricName**

  Name of the metric.

- **metricColumn**

  Name of the column.

- **beaconName**

  Name of the beacon.

- **stepName**

  Name of the step.

- **stepGroupName**

  Name of the step group.

Example

```
emcli delete_test_threshold
      -name="Service Name"
      -type="generic_service"
      -testname="Test Name"
      -testtype="HTTP"
      -metricName="http_response"
```

```
                        -metricColumn="timing"
```

delete_user

Deletes an existing Enterprise Manager administrator.

When a user is deleted, all jobs the user creates are stopped and deleted. Also, any blackouts the user creates are deleted. However, a user cannot be deleted if any blackouts the user creates are active at the time the call to delete the user is issued. This situation is considered an invalid state from which to delete a user. First, all of these active blackouts must be stopped, and a thwarted delete user call must be reissued.

Format

```
emcli delete_user
      -name=<user_name>
      [-new_object_owner=<user_name>]
      [-force]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Administrator name.

- **new_object_owner**

  Name of the administrator to assign the secure objects owned by the current administrator being deleted. If you do not specify this option, the secure objects are deleted that are owned by the administrator being deleted.

- **force**

  Deletes the administrator even if the administrator is currently logged in.

Examples

Example 1

This example deletes the Enterprise Manager administrator named `sysman3`.

```
emcli delete_user -name=sysman3
```

Example 2

This example deletes the Enterprise Manager administrator named user1, and assigns all the secure objects owned by user1 to user5.

delete_pluggable_database

Deletes pluggable databases (PDBs).

Format

```
emcli delete_pluggable_database
      -cdbTargetName="CDB_of_target_PDBs"
      -cdbTargetType="CDB_target_type"
      -cdbHostCreds="CDB_host_credentials"
      -cdbTargetCreds="CDB_target_credentials"
      -pdbName="PDB_names"
      [-cdbHostPrivCreds="CDB_host_privileged_credentials"]
      [-ignoreStorageWarnings]
```

`[ ]` indicates that the parameter is optional.

Parameters

- **cdbTargetName**

  The target container database (CDB) that contains the PDBs that you want to delete. Ensure that the target CDB you specify is a valid target in Enterprise Manager.

- **cdbTargetType**

  The database type of the target CDB. It can be `oracle_database, rac_database,` and so on.

- **cdbHostCreds**

  The credentials for the host on which the target CDB is located.

- **cdbTargetCreds**

  The credentials for the target CDB.

- **pdbName**

  The names of the PDBs that you want to delete. Ensure that you separate the names using a comma.

- **cdbHostPrivCreds**

  The privileged credentials for the host on which the Snap Clone storage mount points are located. Note that this parameter is required only if you are deleting PDBs that were created using Snap Clone.

- **ignoreStorageWarnings**

  Specify this option to ignore any storage warnings that may be generated while deleting PDBs that were created using Snap Clone.

Examples

This example deletes the `test_pdb` PDB, which is a part of `test_CDB,` an Oracle single-instance CDB, using `HOST_CREDS` as the CDB host credentials and `DB_CREDS` as the CDB target credentials:

```
emcli delete_pluggable_database -cdbTargetName=test_database
-cdbTargetType=oracle_database -pdbName=test_pdb -cdbHostCreds=HOST_CREDS
-cdbTargetCreds=DB_CREDS
```

deploy_bipublisher_reports

This verb deploys all of the Enterprise Manager Oracle-provided reports, or optionally, specific Enterprise Manager Plug-in reports to the BI Publisher catalog

You can also use this verb to upload a reports jar file (located on the OMS(s)'s file system. The operation does not overwrite existing BI Publisher Reports in the Enterprise Manager reports folder unless you specify the -force option.

> **Note:** This verb requires Enterprise Manager Super Administrator privileges.

Format

```
emcli deploy_bipublisher_reports
    [-force]
    [-all | -reportsjarfile=<reports_jar_file> | (-pluginid=<plugin_id>
     [-pluginversion=<plugin_version>]) )
[ ] indicates that the parameter is optional
```

Parameters

> **Note:** Using -force applies to the entire operation. The absence of all parameters assumes -all.
>
> You can specify the -all option, or -reportsjarfile option, or -pluginid option, but not all three at the same time. If you use the -pluginid option, you can also also include the -pluginversion option.

- **force**

  Overwrites reports. If you use this option, all reports on the BI Publisher server are overwritten with the new copies.

- **all**

  Overwrites reports. If you use this option, all reports on the BI Publisher server are overwritten with the new copies.

- **reportsjarfile**

  Deploys a single Enterprise Manager reports jar file that contains one or more BI Publisher Reports. This jar file is located relative to the OMS's $ORACLE_HOME.

- **pluginid**

  In addition to Enterprise Manager system reports, also deploys any subsequently loaded plug-in-based BI Publisher Reports.

- **pluginversion**

  Limits the plug-ins to a specific version.

Examples

Example 1

This example deploys all platform and plug-in Enterprise Manager Oracle-provided reports, but does not overwrite any existing reports

```
emcli deploy_bipublisher_reports -all
```

Example 2

This example deploys only the Chargeback and Trending reports, and overwrites any existing reports.

```
emcli deploy_bipublisher_reports -force -pluginid=oracle.sysman.emct
-pluginversion=12.1.0.3.0
```

deploy_bipublisher_selfupdates

Deploys Self Update Enterprise Manager reports to the BI Publisher catalog. To deploy all reports in a folder, specify the `-folder` option. To deploy a single report, use the `-folder` and `-report` options. To deploy all Self Update reports, use the `-all` option.

The deploy operation will not overwrite existing BI Publisher reports in the BI Publisher catalog unless the `-force` option is given.

Format

```
emcli deploy_bipublisher_selfupdates
    [-force]
    -all | (-folder=<folder> [-report=<reportname>])

[ ] indicates that the parameter is optional.
```

Parameters

- **force**

  Overwrites any existing reports in the BI Publisher catalog with the copy being deployed.

- **all**

  Deploys all Self Update reports.

- **folder**

  Limits the reports deployed to the specified folder. By default, all reports in the folder are deployed unless the `-report` option is also specified.

- **report**

  Deploys a single, specified report. The `-folder` parameter must be specified when using the -report parameter.

Examples

Example 1

The following example deploys all Self Update reports, but does not overwrite any existing reports:

```
emcli deploy_bipublisher_selfupdates -all
```

Example 2

The following example deploys all Self Update reports to the Compliance Reports folder:

```
emcli deploy_bipublisher_selfupdates -folder="Compliance Reports"
```

Example 3

The following example deploys only the Self Update report named Consolidation Report located in the Compliance Reports folder and overwrites any existing report.

```
 emcli deploy_bipublisher_selfupdates -force -folder="Compliance Reports"
-report="Consolidation Report"
```

deploy_plugin_on_agent

Deploys a plug-in on Management Agents. Agent names must be provided for plug-in deployment.

> **Note:** A plug-in can only be deployed on any Management Agent after it has been successfully deployed on the management server.

Format

```
emcli deploy_plugin_on_agent
        -agent_names=<agent1;agent2>
        -plugin=<plug-in_id[:version>]
        [-discovery_only]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_names**

  Management Agents (host:port) on which the plug-in needs to be deployed.

- **plugin**

  Plug-in ID and version that needs to be deployed. Version is optional, and it defaults to the latest applicable version deployed on the management server. If a later version is available but not certified on the Agent OS platform, the latest version is not picked up.

- **discovery_only**

  To be used when only discovery content needs to be deployed.

Examples

Example 1

This example deploys the latest version of oracle.sysman.db2 on Management Agent myhost1.example.com.

```
emcli deploy_plugin_on_agent -plugin="oracle.sysman.db2"
-agent_names="myhost1.example.com:1838"
```

Example 2

This example deploys version 12.1.0.1.0 of plug-in oracle.sysman.db2 on management agent myhost1.us.example.com.

```
emcli deploy_plugin_on_agent
      -plugin="oracle.sysman.db2:12.1.0.1.0"
      -agent_names="myhost1.us.example.com:1838"
```

deploy_plugin_on_server

Deploys a plug-in on the Management Servers. The deployment process for some plug-ins might restart the Management Servers. If the plug-in is already deployed on one of the servers, this server is skipped. If a lower version of the plug-in is already deployed, the plug-in is upgraded. If a lower revision of the plug-in is already deployed, the new revision is applied.

Format

```
emcli deploy_plugin_on_server
        -plugin=<plug-in_id>[:<version>]
        [-sys_password=<sys_password>]
        [-prereq_check]
        [-use_last_prereq_result]

[ ] indicates that the parameter is optional
```

Parameters

- **plugin**

  ID or ID:Version of the plug-in to be deployed on the Management Servers of the form -plugin=<oracle.sysman.db:12.1.0.1.0>, where the plug-in ID (like oracle.sysman.db) is a required parameter, and the version is optional.

  If do not specify a version, the highest version of the plug-in that has been downloaded is considered for deployment. If multiple revisions of this plug-in version are downloaded, the highest revision is considered for deployment.

- **sys_password**

  Password of the repository DBA SYS. If you do not provide this , you are prompted for the password. This is not required if you use the prereq_check .

- **prereq_check**

  If you provide this option, instead of deploying the plug-in, the verb displays only a check for all the unfulfilled prerequisites for this plug-in deployment to be successful. If you do not provide this option, plug-in deployment follows a prerequisities check.

- **use_last_prereq_result**

  If prerequisites checks have been performed previously for a given set of plug-ins using the -prereq_check option and no other deployment activity occurred for these plug-ins, you can use this option to skip prerequisite checks and start the deployment immediately.

Examples

Example 1

This example deploys the latest downloaded version of Oracle Database plug-in (plug-in ID: oracle.sysman.db) on the management server.

```
emcli deploy_plugin_on_server
      -plugin=oracle.sysman.db
      -sys_password=<sys_password>
```

Example 2

This example deploys the latest downloaded version of a Oracle Database plug-in (plug-in ID: oracle.sysman.db) and Oracle Fusion Middleware plug-in (oracle.sysman.emas) on the management server.

```
emcli deploy_plugin_on_server
      -plugin="oracle.sysman.db;oracle.sysman.emas"
      -sys_password=<sys password>
```

Example 3

This example deploys the Oracle Database plug-in (with version 12.1.0.2.0) and Oracle Fusion Middleware plug-in (version 12.1.0.2.0) on the management server. Since sys password has not been passed on the command line, you are prompted for it.

```
emcli deploy_plugin_on_server
      -plugin="oracle.sysman.db:12.1.0.2.0;oracle.sysman.emas:12.1.0.2.0"
```

Example 4

The folllowing example deploys the Oracle Database plug-in (with version 12.1.0.2..0) and Oracle Fusion Middleware plug-in (12.1.0.2.0) on the management server. Since sys password has not been passed on the command line, you are prompted for it. If a lower version of both plug-ins have already been deployed, they are upgraded to 12.1.0.2.0. If a lower version of only one of the plug-ins is deployed, this generates an error, and you will have to deploy them separately.

```
emcli deploy_plugin_on_server
      -plugin="oracle.sysman.db:12.1.0.2.0;oracle.sysman.emas:12.1.0.2.0"
```

Example 5

This example only performs prerequisite checks on the Oracle Database plug-in and does not actually deploy the plug-in.

```
emcli deploy_plugin_on_server
      -plugin=oracle.sysman.db:11.2.0.1.0 -prereq_check
```

deregister_forwarder_agents

Takes a list of agents and deregisters each agent as a forwarding agent.

Format

```
emcli deregister_forwarder_agents
      -agent_list="agent_list"
[ ] indicates that the parameter is optional.
```

Parameters

- agent_list

  List of agents that need to be deregistered as forwarders. The agents must be separated by space.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example deregisters agent1 and agent2 as forwarding agents.

```
emcli deregister_forwarder_agents
      -agent_list="agent1 agent2..."
```

describe_dbprofile_input

Lists and describes all database profile creation input variables.

Format

```
emcli describe_dbprofile_input
      [-data_mode={EXPORT/DBCA_TEMPLATE/RMAN/STORAGE_SNAPSHOT}]

[ ] indicates that the parameter is optional.
```

Parameters

- data_mode

  Data mode for which the database profile needs to be submitted.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example lists all input variables required for creating a snapshot database profile.

```
emcli describe_dbprofile_input -data_mode=STORAGE_SNAPSHOT
```

describe_fmw_profile

Provides a description of the Fusion Middleware provisioning profile from the software library.

Format

```
emcli describe_fmw_profile
      -location="Profile Location"
```

Parameters

- location

  The complete software library path to the profile. Use the list_fmw_profiles verb to identify the complete path.

  Note: The name and owner parameters must be used together.

Example

The following example displays a description of the Fusion Middleware profile "MyProfile" from software library.

```
emcli describe_fmw_profile
      -location="Fusion Middleware Provisioning/Profiles/MyProfile"
```

describe_job

Describes a job and gets its properties for a job you have submitted from the user interface or using the create_job verb. The output can be redirected into a file and used as a template.

This verb support multi-task jobs.

Format

```
emcli describe_job
     -name=<"job_name">
     [-owner=<"job_owner">]
     [-verbose]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the job to describe.

- **owner**

  Enterprise Manager administrator who owns this job. If not provided, the current EM CLI logged-in administrator is assumed as the owner. The logged-in Enterprise Manager administrator must have at least the view privilege to describe a job.

- **verbose**

  Outputs a help template along with the properties.

Examples

Example 1

This example describes the library job "myJob" owned by the logged-in Enterprise Manager administrator.

```
emcli describe_job -name=myJob
```

Example 2

This example describes the library job "yourJob" owned by the Enterprise Manager administrator "admin1". The logged-in Enterprise Manager administrator has view privilege on this job.

```
emcli describe_job -name=yourJob -owner=admin1
```

Example 3

```
emcli describe_job -name=EMCLI_JOB_2

# Job Name : EMCLI_JOB_2

# Current status of the job is ACTIVE.

# Job Type: OSCommand.
# This job type supports the following target types only :
# host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
```

```
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.

target_list=myHost:host

# Variable: args
# Description: Parameters of the command to run on the target
variable.args=hello

# Variable: command
# Description: Command to run on the target
variable.command=echo

# Credential Usage: defaultHostCred
# Description:
cred.defaultHostCred.myHost:host=NAMED:Admin1:CRED1

schedule.frequency=REPEAT_BY_MINUTES
schedule.startTime=2012-02-01 01:01:01.0
schedule.endTime=2051-02-01 01:01:01.0
schedule.gracePeriod=-1
schedule.months=
schedule.days=
schedule.interval=1
schedule.timezone.type=TIMEZONE_TARGET
schedule.timezone.targetIndex=1
schedule.timezone.zoneOffset=0
schedule.timezone.region=
```

Example 4

```
emcli describe_job -name=EMCLI_JOB_2 -verbose

# Job Name : EMCLI_JOB_2

# Current status of the job is ACTIVE.

# Job Type: OSCommand.
# This job type supports the following target types only :
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.


# Target List.
# In a target list, each member is specified using the target name and target type
# in the fashion:
#    target_name:target_type
# To specify an element of the target list, the following notation is used:
#    job_target_list.1=target_name:target_type
# The suffix "1" after the key word "job_target_list" signifies that the entry is
# for the first element.
# The target target_name:target_type should exists in EM.
# Permissible target types are:
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.
# A sample target list could be:
```

```
# job_target_list.1=<target_name>:host
# job_target_list.2=<target_name>:host
# The target list can only contain targets of the same target type. A cluster,
# group, domain or system
# target must not be intermixed with targets of the other target types.


# Variable List.
# In a variable list, each member is specified in the following way:
# Scalar variable: A variable whose value can be represented as a single string.
#    variable.variable_name=variable_value
# Here "variable" is a keyword. Variable name is the name of the variable whose
# value is being specified.
# Value is specified on the right hand side after the equal to sign.
# Vector variable: A variable whose value is represented as an array or list of
# string values.
#    variable.variable_name.1=value1
#    variable.variable_name.2=value2
# Here the numbers suffixing the variable name signify the entry number in the
# list.
# Large variable: A variable whose value is exceptionally large. Syntax is similar
# to a scalar variable.
#    variable.large_variable_name=a_very_very_big_value


# Credential List.
# This is the list of credential usages declared by the job type.
# Each entry takes the form:
#    cred.credusage_name.target_details=cred_type:cred_details
# Here the prefix "cred" is a keyword signifying that this line represents a
# credential entry.
# "credusage_name" would be substituted with the name of the credential usage
# declared in the job type.
# This is followed by the target details, which take the following form:
#    target_name:target_type
# The value for this credential usage entry is specified using the type of the
# credential and its details.
# "cred_type" can take either "SET" or "NAMED" as its value, depending on whether
# the credential is a credential set or a named credential.
# "cred_details" can specify either the name of a credential set or the name of a
# named credential based on the "cred_type"
# A sample entry for a target target1:host for credential usage defaultHostCred
# for a credential set could look like:
#    cred.defaultHostCred.target1:host=SET:HostCredsNormal
# A sample entry for a target target1:host for credential usage defaultHostCred
# for a named credential could look like:
#    cred.defaultHostCred.target1:host=NAMED:MyNamedCredential
# A sample entry for a target target1:host for credential usage defaultHostCred
# for a named credential shared by EM Admin "admin1" could look like:
#    cred.defaultHostCred.target1:host=NAMED:admin1:MyNamedCredential

# Schedule.
# Specify a schedule for the job. Detailed instructions as per below:
# Frequency: Specifies the frequency of repeatedly submitting instances of this
# job.
#    scheule.frequency=Frequency_Type
# Frequency type could be either of IMMEDIATE, ONCE, WEEKLY, MONTHLY, YEARLY,
# REPEAT_BY_MINUTES, REPEAT_BY_HOURS, REPEAT_BY_DAYS, REPEAT_BY_WEEKS.
# If frequency is IMMEDIATE, then other schedule fields do not matter.
# Start Time: Start time for the schedule.
```

```
#    scheule.startTime=MM-DD-YYYY
# End Time: End time for the schedule.
#    scheule.endTime=MM-DD-YYYY
# Grace Period: Grace period in minutes for the schedule.
#    scheule.graceperiod=
# Months : Months for repetition. January is denoted by 0 and December by 11
#    schedule.months=0,1,2
# Days: Days of the week for repetition. Sunday is denoted by 0 and Saturday by 6.
#    schedule.days=0,1,2
# Timezone: Timezone information is further detailed into type, target index, zone
# offset and region.
#    schedule.timezone.type: either of TIMEZONE_TARGET, TIMEZONE_SPECIFIED,
# TIMEZONE_REGION_SPECIFIED.
#    schedule.timezone.targetIndex : specify the index of the target whose
# timezone is to be used.
#    schedule.timezone.zoneOffset : timezone offset.
#    schedule.timezone.region : timezone region
# Following is a complete schedule section, remove #  and populate the values for
# submission:
# scheule.frequency=ONCE
# schedule.startTime=12-21-2012
# schedule.endTime=12-21-2012
# schedule.gracePeriod=10
# schedule.months=
# schedule.days=
# schedule.timezone.type=TIMEZONE_TARGET
# schedule.timezone.targetIndex=1
# schedule.timezone.zoneOffset=
# schedule.timezone.region=

job_target_list.1=myhost.us.example.com:host

# Variable: args
# Description: Parameters of the command to run on the target
variable.args=hello

# Variable: command
# Description: Command to run on the target
variable.command=echo

# Credential Usage: defaultHostCred
# Description:
cred.defaultHostCred.myhost.us.example.com:host=NAMED:SYSMAN:CRED1

schedule.frequency=REPEAT_BY_MINUTES
schedule.startTime=2012-02-01 01:01:01.0
schedule.endTime=2051-02-01 01:01:01.0
schedule.gracePeriod=-1
schedule.months=
schedule.days=
schedule.interval=1
schedule.timezone.type=TIMEZONE_TARGET
schedule.timezone.targetIndex=1
schedule.timezone.zoneOffset=0
schedule.timezone.region=
```

describe_job_type

Describes the job type and gets its properties. The output can be redirected into a file.

This verb dumps out a properties file for a job type that supports the Job System Generic EM CLI. This file contains some documentation, a list of all required credential usages, and a list of all variables required to create a (library) job instance of the job type.

This verb support multi-task jobs.

Format

```
emcli describe_job_type
     -job_type=<"job_type_internal_name">
     [-verbose]

[ ] indicates that the parameter is optional
```

Parameters

- **job_type**

  Specify the name of the job type to describe. You can use the get_job_types verb to obtain the names of all job types for which a job or library jobs can be created using EM CLI.

- **verbose**

  Outputs a help template along with the properties.

Examples

Example 1

This example describes the job type "MyJobType."

```
emcli describe_job_type -job_type=MyJobType
```

Example 2

This example produces a property file on the console, which can be redirected to a file and used multiple times.

```
emcli describe_job_type -job_type=OSCommand

# Job Type: OSCommand.
# This job type supports the following target types only :
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.


# Variable: args
# Description: Parameters of the command to run on the target
variable.args=

# Variable: command
# Description: Command to run on the target
variable.command=
```

```
# Credential Usage: defaultHostCred
# Description:
cred.defaultHostCred.<target_name>:<target_type>=
```

Example 3

This example with the verbose option generates a property dump with help on how to specify each individual property for the job.

```
emcli describe_job_type -job_type=OSCommand -verbose

# Job Type: OSCommand.
# This job type supports the following target types only :
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.


# Target List.
# In a target list, each member is specified using the target name and target type
# in the fashion:
#    target_name:target_type
# To specify an element of the target list, the following notation is used:
#    job_target_list.1=target_name:target_type
# The suffix "1" after the key word "job_target_list" signifies that the entry is
# for the first element.
# The target target_name:target_type should exists in EM.
# Permissible target types are:
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.
# A sample target list could be:
# job_target_list.1=<target_name>:host
# job_target_list.2=<target_name>:host
# The target list can only contain targets of the same target type. A cluster,
# group, domain or system
# target must not be intermixed with targets of the other target types.

# Variable List.
# In a variable list, each member is specified in the following way:
# Scalar variable: A variable whose value can be represented as a single string.
#    variable.variable_name=variable_value
# Here "variable" is a keyword. Variable name is the name of the variable whose
# value is being specified.
# Value is specified on the right hand side after the equal to sign.
# Vector variable: A variable whose value is represented as an array or list of
# string values.
#    variable.variable_name.1=value1
#    variable.variable_name.2=value2
# Here the numbers suffixing the variable name signify the entry number in the
# list.
# Large variable: A variable whose value is exceptionally large. Syntax is similar
# to a scalar variable.
#    variable.large_variable_name=a_very_very_big_value

# Credential List.
# This is the list of credential usages declared by the job type.
# Each entry takes the form:
```

```
#    cred.credusage_name.target_details=cred_type:cred_details
# Here the prefix "cred" is a keyword signifying that this line represents a
# credential entry.
# "credusage_name" would be substituted with the name of the credential usage
# declared in the job type.
# This is followed by the target details, which take the following form:
#    target_name:target_type
# The value for this credential usage entry is specified using the type of the
# credential and its details.
# "cred_type" can take either "SET" or "NAMED" as its value, depending on whether
# the credential is a credential set or a named credential.
# "cred_details" can specify either the name of a credential set or the name of a
# named credential based on the "cred_type"
# A sample entry for a target target1:host for credential usage defaultHostCred
# for a credential set could look like:
#    cred.defaultHostCred.target1:host=SET:HostCredsNormal
# A sample entry for a target target1:host for credential usage defaultHostCred
# for a named credential could look like:
#    cred.defaultHostCred.target1:host=NAMED:MyNamedCredential
# A sample entry for a target target1:host for credential usage defaultHostCred
# for a named credential shared by EM Admin "admin1" could look like:
#    cred.defaultHostCred.target1:host=NAMED:admin1:MyNamedCredential

# Schedule.
# Specify a schedule for the job. Detailed instructions as per below:
# Frequency: Specifies the frequency of repeatedly submitting instances of this
# job.
#    scheule.frequency=Frequency_Type
# Frequency type could be either of IMMEDIATE, ONCE, WEEKLY, MONTHLY, YEARLY,
# REPEAT_BY_MINUTES, REPEAT_BY_HOURS, REPEAT_BY_DAYS, REPEAT_BY_WEEKS.
# If frequency is IMMEDIATE, then other schedule fields do not matter.
# Start Time: Start time for the schedule.
#    scheule.startTime=MM-DD-YYYY
# End Time: End time for the schedule.
#    scheule.endTime=MM-DD-YYYY
# Grace Period: Grace period in minutes for the schedule.
#    scheule.graceperiod=
# Months : Months for repetition. January is denoted by 0 and December by 11
#    schedule.months=0,1,2
# Days: Days of the week for repetition. Sunday is denoted by 0 and Saturday by 6.
#    schedule.days=0,1,2
# Timezone: Timezone information is further detailed into type, target index, zone
# offset and region.
#    schedule.timezone.type: either of TIMEZONE_TARGET, TIMEZONE_SPECIFIED,
# TIMEZONE_REGION_SPECIFIED.
#    schedule.timezone.targetIndex : specify the index of the target whose
# timezone is to be used.
#    schedule.timezone.zoneOffset : timezone offset.
#    schedule.timezone.region : timezone region
# Following is a complete schedule section, remove #  and populate the values for
# submission:
# scheule.frequency=ONCE
# schedule.startTime=12-21-2012
# schedule.endTime=12-21-2012
# schedule.gracePeriod=10
# schedule.months=
# schedule.days=
# schedule.timezone.type=TIMEZONE_TARGET
# schedule.timezone.targetIndex=1
# schedule.timezone.zoneOffset=
```

```
# schedule.timezone.region=


# Variable: args
# Description: Parameters of the command to run on the target
variable.args=

# Variable: command
# Description: Command to run on the target
variable.command=

# Credential Usage: defaultHostCred
# Description:
cred.defaultHostCred.<target_name>:<target_type>=
```

describe_library_job

Describes a library job and gets its properties. The output can be redirected into a file.

Format

```
emcli describe_library_job
     -name=<"job_name">
     [-owner=<"job_owner">]
     [-verbose]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the library job to describe.

- **owner**

  Enterprise Manager administrator who owns this library job. If not provided, the current EM CLI logged-in administrator is assumed as the owner. The logged-in Enterprise Manager administrator must have at least the view privilege to describe a job.

- **verbose**

  Outputs a help template along with the properties.

Examples

Example 1

This example describes the library job "myLibJob" owned by the logged-in Enterprise Manager administrator.

```
emcli describe_library_job -name=myLibJob
```

Example 2

This example describes the library job "yourLibJob" owned by the Enterprise Manager administrator "admin1". The logged-in Enterprise Manager administrator has view privilege on this library job.

```
emcli describe_library_job -name=yourLibJob -owner=admin1
```

Example 3

```
emcli describe_library_job -name=MYJOB1

# Job Name : MYJOB1

# Current status of the job is ACTIVE.

# Job Type: OSCommand.
# This job type supports the following target types only :
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.
```

```
job_target_list.1=myhost.us.example.com:host

# Variable: args
# Description: Parameters of the command to run on the target
variable.args=hello

# Variable: command
# Description: Command to run on the target
variable.command=echo

# Credential Usage: defaultHostCred
# Description:
cred.defaultHostCred.myhost.us.example.com:host=NAMED:SYSMAN:CRED1

schedule.frequency=REPEAT_BY_MINUTES
schedule.startTime=2012-02-01 01:01:01.0
schedule.endTime=2051-02-01 01:01:01.0
schedule.gracePeriod=-1
schedule.months=
schedule.days=
schedule.interval=1
schedule.timezone.type=TIMEZONE_TARGET
schedule.timezone.targetIndex=1
schedule.timezone.zoneOffset=0
schedule.timezone.region=
```

describe_patch_plan_input

Describes the input data of a patch plan.

Format

```
emcli describe_patch_plan_input
        -name=<name>
```

Parameters

- **name**

  Name of a given patch plan.

Example

```
emcli describe_patch_plan_input -name="plan_name
```

See Also

create_patch_plan
delete_patches
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

describe_procedure_input

Describes the input data of a deployment procedure or a procedure configuration.

Format

```
emcli describe_procedure_input
      [-procedure=<procedure_GUID>]
      [-name=<procedure_name_or_procedure_conf>]
      [-owner=<procedure_owner_or_procedure_config>]
      [-parent_proc=<procedure_of_procedure_config>]

[ ] indicates that the parameter is optional
```

Parameters

- **procedure**

  GUID of the procedure to execute.

- **name**

  Name of the procedure or procedure configuration.

- **owner**

  Owner of the procedure or procedure configuration.

- **parent_proc**

  Procedure of the procedure configuration. This applies to describe a procedure configuration when both a procedure and a procedure configuration have the same name.

Examples

```
emcli describe_procedure_input -procedure=16B15CB29C3F9E6CE040578C96093F61 >
describeDP.properties
```

diagchecks_deploy_status

Gets the status of diagnostic checks deployments against different target types.

Format

```
emcli diagchecks_deploy_status
        [-target_type=<type>]*

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target. You can specify multiple values.

diagchecks_deploy_tglist

Gets the target list for a particular deployment type for a target type.

Format

```
emcli diagchecks_deploy_tgtlist
        -target_type=<type>
        -deploy_type=<CURRENT|OLDER|MISSING|ALL>
        [-show_excludes]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target. You can specify multiple values.

- **deploy_type**

  Deployment type of either CURRENT, OLDER, MISSING, or ALL.

- **show_excludes**

  For targets where excludes have been set, print them.

disable_audit

Disables auditing for all user operations.

Format

```
emcli disable_audit
```

Example

This example disables auditing for all operations.

```
emcli disable_audit
```

disable_config_history

Disables configuration history computation for a target type.

Format

```
emcli disable_config_history
      -target_type="{target type|'*'}"
```

Parameters

- target_type

  Target type for which the configuration history is being disabled. The value should be the internal name or "*" to indicate all target types.

Examples

Example 1

This example disables configuration history computation for the host target type.

```
emcli disable_config_history -target_type="host"
```

Example 2

This example disables configuration history computation for all target types.

```
emcli disable_config_history -target_type="*"
```

disable_sla

Disables an SLA for a target.

Format

```
emcli disable_sla
    -targetName=<target_name>
    -targetType=<target_type>
    -slaName=<SLA_name>
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **slaName**

  Name of the SLA.

Examples

This example disables an SLA named 'gold_sla' for target my_service (generic_service).

```
emcli disable_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla'1
```

disable_test

Disables monitoring of a Services test.

Format

```
emcli disable_test
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **testname**

  Test name.

- **testtype**

  Test type.

Examples

This example disables the HTTP test named `MyTest` for the `generic_service` target named `MyTarget`.

```
emcli disable_test -name='MyTarget' -type='generic_service'
      -testname='MyTest' -testtype='HTTP'
```

discover_bda_cluster

Performs Big Data discovery for the specified host. Can be used for new discovery or for rediscovery of the latest configuration changes.

Format

```
emcli discover_bda_cluster
        -hostname="host_name"
        -host_credential="host_named_cred"
        -ilom_credential="ilom_named_cred"
        -infiniband_credential="ibswitch_named_cred"
        -cloudera_credential="cloudera_named_cred"
        -snmp_string="SNMP_community_string"
```

Parameters

- **hostname**

  The name of host in the Big Data Network.

- **host_credential**

  Named credentials for the `oracle` OS account that owns a Management Agent home.

- **ilom_credential**

  Named credentials for the `root` OS account on an Oracle Integrated Lights Out Manager (Oracle ILOM) server in the Big Data Network.

- **infiniband_credential**

  Named credentials for the `nm2user` OS account on an InfiniBand switch in the Big Data Network.

- **cloudera_credential**

  Named credentials for the `admin` account of the Cloudera Manager that manages the CDH cluster.

- **snmp_string**

  SNMP community string for PDU and Cisco switch traps. The read-only string is public.

Example

The following example performs BDA cluster discovery on the host named `acme101.com`. If the cluster already exists, updates the latest configuration.

```
emcli discover_bda_cluster
         -hostname="acme101.com"
         -host_credential="HOST_CRED"
         -ilom_credential="ILOM_CRED"
         -infiniband_credential="IB_CRED"
         -cloudera_credential="CM_CRED"
         -snmp_string="public"
```

discover_cloudera_cluster

Discovers the Hadoop cluster for the specified Cloudera Manager host. Can also be used for rediscovery of the latest cluster configuration changes.

Format

```
emcli discover_cloudera_cluster
        -hostname = "host_name"
        -cloudera_credential = "cloudera_named_cred"
        -host_credential  = "host_named_cred"
```

Parameters

- **hostname**

  Name of one of the hosts that form the cluster.

- **cloudera_credential**

  Named credentials for the Cloudera Manager managing the cluster.

- **host_credential**

  Named credentials for the specified host.

Example

The following example discovers the Hadoop cluster that includes a host named `acme101.com`, using the provided named credentials:

```
emcli discover_bda_cluster
        -hostname="acme101.com"
        -cloudera_credential="CM_CRED"
        -host_credential="HOST_CRED"
```

discover_coherence

Discovers one or more Coherence clusters.

Format

```
emcli discover_coherence
      -input_file=coherence_discovery_file:file_path
      [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Fully-qualified path to a CSV-formatted file containing one line of details per Coherence cluster. The valid WebLogic version value is 10. The structure of the CSV file is as follows:

  ```
  <Management Node host machine name>,
            <Management Node listen port>,
            <Management Node username - optional>,
            <Management Node password - optional>,
            <Management Node service name - optional>,
            <Agent url>
  ```

  For example:

  ```
  host1.companyA.com,9910,,,,https://host1.companyA.com:3872/emd/main/,
  ```

  For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Examples

This example reads the my_clusters_info.csv file to determine the clusters to be added to Cloud Control.

```
emcli discover_coherence
  -input_file=coherence_discovery_file:"c:\emcli\my_clusters_info.csv
```

discover_fa

Discovers multiple Fusion Applications domains by reading the Fusion Applications domain discovery file and saving the host-wise discovered targets to the Agents provided in the Host Agent Mapping file. If the Host Agent mapping file is not provided, the local Agent (that is, the Agent on the same host as the target) is used to save/monitor the discoverd targets as well. If a local Agent is not found, the default discovery Agent is used to save/monitor the discoverd targets as well.

> **Note:** Although this verb supports discovering multiple Fusion instances at one time by adding all the details in one file, it is advisable to discover each Fusion instance separately using individual EM CLI discover_fa commands run multiple times.

Format

```
emcli discover_fa
      -input_file=fa_domain_discovery_file:file_path
      [-input_file=host_agent_mapping_file:file_path]
      [-input_file=pf_domain_cred_mapping_file:file_path]
      [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file=fa_domain_discovery_file**

  Fully-qualified path to a CSV-formatted file containing one line of details per domain to be added. The valid WebLogic version value is 10. The structure of the CSV file is as follows:

  ```
  <WebLogic Server version>,
  <Administration Server host machine name>,
  <Administration Server listen port>,
  <Administration Server username>,
  <Administration Server password>,
  <External Parameters - optional>,
  <JMX Protocol - required only if SSL enabled>,
  <JMX Service URL - required only if SSL enabled>,
  <Unique Domain Identifier>,
  <Agent URL/>,
  <Discover Down Servers - optional - Default if not specified is false starting
  <PS1. Before PS1 the default for this is true>,
  <Use Same Credentials for All Domains in the Fusion Instance - optional -
  Default if <not specified is true>
  ```

  For example:

  ```
  10,mco01.mycompany.com,7001,weblogic,welcome1,,,,my_farm_
      01,https://mco01.mycompany.com:3872/emd/main/,,
  10,mco01.mycompany.com,7001,weblogic,welcome1,,,,my_farm_
      01,https://mco01.mycompany.com:3872/emd/main/,true,
  10,mco01.mycompany.com,7001,weblogic,welcome1,,,,my_farm_
      01,https://mco01.mycompany.com:3872/emd/main/,true,true
  10,mco01.mycompany.com,7001,weblogic,welcome1,,,,my_farm_
      01,https://mco01.mycompany.com:3872/emd/main/,false,true
  ```

For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=host_agent_mapping_file**

  Fully-qualified path to a CSV-formatted file containing multiple lines of host system names where Managed Servers are to be monitored, and the Agent to be used to monitor each host's Managed Servers.

  For example:

```
mycompany.com,https://mco01.mycompany.com:3872/emd/main
```

For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=pf_domain_cred_mapping_file**

  Fully-qualified path to a CSV-formatted file containing multiple lines of WebLogic admin credentials for each domain of a fusion instance, where the credentials are different from those added in the fa_domain_discovery file.

  The same credentials are used for all the domains in a Fusion Application instance unless the credentials are overwritten in the pf_domain_cred_mapping file.

  For example:

```
<UniqueKey - "<Fusion Instance
     Identifier><CommonDomainDisplayName>">,<Administration Server
     username>,<Administration Server password>,
<UniqueKey - "<Fusion Instance
     Identifier>-<CommonDomainDisplayName>">,<Administration Server
     username>,<Administration Server password>,<Administration Server Host
     Name>

Example:
fi9-FS,weblogic12,welcome1,
fi9-PRJ,faadmin,fusionfa1,
fi9-PRC,faadmin,fusionfa1,myhost.us.example.com
fi9-PRC,,,myhost.us.example.com
```

For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Examples

Example 1

This example reads the my_domains_info.csv file to determine the Fusion Instances to be added to Cloud Control, reads the my_agent_mapping.csv file to determine which Agents should monitor which host's Managed Servers, and reads the my_domain_cred_mapping.csv file to determine which credentials are to be used to discover an individual product family.

```
emcli discover_fa
      -input_file=fa_domain_discovery_file:c:\emcli\my_domains_info.csv
      -input_file=host_agent_mapping_file:c:\emcli\my_agent_mapping.csv
      -input_file=pf_domain_cred_mapping_file:c:\emcli\my_domain_cred_mapping.csv
```

Example 2

```
emcli discover_fa -input_file=fa_domain_discovery_file:/tmp/emcli/
domain_discovery_file.txt -input_file=host_agent_mapping_file:/tmp/emcli/
host_agent_mapping_file.txt -debug
```

Example 3

```
emcli discover_fa -input_file=fa_domain_discovery_file:/tmp/emcli/
domain_discovery_file.txt -input_file=host_agent_mapping_filee:/tmp/emcli/
host_agent_mapping_file.txt -input_file=pf_domain_cred_mapping_file:/tmp/emcli/
pf_domain_cred_mapping_file.txt -debug
```

discover_gf

Discovers Multiple GlassFish Domains by reading the Domain Discovery file and saving the discovered targets of the host to the Agents provided in the Host Agent Mapping file. If the Host Agent mapping file is not provided, the local Agent (the Agent on the same host as the target) is used to save/monitor the discovered targets. If a local Agent is not found, the default discovery Agent is used to save/monitor the discovered targets.

Format

```
$emcli discover_gf
       -input_file=domain_discovery_file:file_path
       [-input_file=host_agent_mapping_file:file_path]
       [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file=domain_discovery_file**

  Fully-qualified path to a CSV-formatted file containing one line of details per domain to be added. The structure of the CSV file is as follows:

  ```
  <Administration Server host machine name>,
  <Administration Server listen port>,
  <Administration Server username>,
  <Administration Server password>,
  <Unique Domain Identifier>,
  <Agent url - optional >,
  <Protocol - optional >,
  <Service URL - optional>,
  <External Parameters - optional>,
  <Discover Down Servers - optional - Default if not specified is false>,\n" +
  ```

  For example:

  ```
  mco01.mycompany.com,4848,admin,welcome1,my_domain_
  01,https://mco01.mycompany.com:3872/emd/main
  mco01.mycompany.com,4848,admin,welcome1,my_domain_
  01,https://mco01.mycompany.com:3872/emd/main,http,,,true
  ```

For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=host_agent_mapping_file**

  Fully-qualified path to a CSV-formatted file containing multiple lines of host system names where Managed Servers are to be monitored, and the Agent to be used to monitor each host's Managed Servers. The structure of the CSV file is as follows:

  ```
  <target_host1>,<save_to_agent1>
  <target_host2>,<save_to_agent3>
  ```

  For example:

  ```
  mycompany.com,https://mco01.mycompany.com:3872/emd/main
  ```

For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Examples

Example 1

```
$emcli discover_gf -input_file=domain_discovery_file:/tmp/emcli/domain_discovery_
file.txt
```

Example 2

```
$emcli discover_gf -input_file=domain_discovery_file:/tmp/emcli/domain_discovery_
file.txt -input_file=host_agent_mapping_file:/tmp/emcli/host_agent_mapping_
file.txt -debug
```

discover_siebel

Discovers Siebel Enterprise instances.

Format

```
emcli discover_siebel
       -input_file=enterprise_info_file:<file_path>
       [-out_file='<fully_qualified_path_of_output_file>']
       [-precheck]
       [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  The input file should be in a CSV format. The structure of the CSV file is as follows:

  ```
  GATEWAY_HOST = < Gateway Server Host >,
  PORT = < Gateway Server Port - optional Default if not specified is 2320 >,
  INSTALL_DIR = < Gateway Server Install Directory - optional >,
  ENTERPRISE_NAME = < Siebel Enterprise Name >,
  SIEBEL_USERNAME = < Siebel Enterprise User Name >,
  SIEBEL_PASSWORD = < Siebel Enterprise Password >,
  DATABASE_USERNAME = < Database User Name >,
  DATABASE_PASSWORD = < Database Password >
  ```

  Note: INSTALL_DIR is a mandatory parameter for discovering Siebel version 8.2.2 and above.

  This example shows discovery of a Siebel Enterprise (siebel) with the gateway located at host 'host1', installed at location 'Location1' and running at port '23201', with a Siebel user name and password of 'sbluser' and 'SBLpass' respectively, and a database user name and password of 'dbuser' and 'DBpass' respectively.

  ```
  GATEWAY_HOST=host1,PORT=23201,INSTALL_DIR=Location1,
  ENTERPRISE_NAME=siebel,SIEBEL_USERNAME=sbluser,
  SIEBEL_PASSWORD=SBLpass,DATABASE_USERNAME=dbuser,
  DATABASE_PASSWORD=DBpass
  ```

  Special cases for commas:

  – If any entry, such as a password, has a comma (,) you need to add it as a backslash comma (\,) in the CSV file. For instance, if SIEBEL_PASSWORD is we,lco,me1 the entry in the CSV file would be SIEBEL_PASSWORD = we\,lc\,ome1 .

  – If any entry, such as a password, has a backslash followed by a comma(\,) you need to add it as as two backslashes followed by a comma(\\,) in the CSV file. For instance, if SIEBEL_PASSWORD is we\,lco\,me1 the entry in the CSV file would be SIEBEL_PASSWORD = we\\,lc\\,ome1 .

  For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **out_file**

Command output is redirected to this file. If not specified, output is printed on the console.

- **debug**

  Executes in verbose mode and generates additional debug log messages in the output. If specified, detailed output is printed.

- **precheck**

  Performs a mock discovery of the Siebel enterprise by executing all of the checks and validations. This option lists the results of these steps to the user for review prior to an actual discovery. It ensures that all prerequisite are met, and discovery does not occur if prerequisites are met.

Examples

Example 1

This example reads the my_enterprise_info.csv file to determine the Siebel Enterprise instances to be added to Cloud Control. The output of the command is redirected to the discovery_output.txt file.

```
emcli discover_siebel
      -input_file=enterprise_info_file:'c:\emcli\my_enterprise_info.csv'
      -out_file='c:\emcli\discovery_output.txt'
      -debug
```

Example 2

This example is the same as the example above, except it adds the -precheck option, which confirms if the precheck is successful, or shows errors if it failed.

```
emcli discover_siebel
      -input_file=enterprise_info_file:'c:\emcli\my_enterprise_info.csv'
      -out_file='c:\emcli\discovery_output.txt'
      -debug
```

discover_wls

Purpose

Used to discover one or more version 8.x, 9.x, 10.x, and 12.x WebLogic Domains (along with Oracle Fusion Middleware 11g software deployed to it), and to specify which Management Agent should monitor which hosts' Managed Servers. Specifying which Management Agent should monitor which hosts' Managed Servers is a feature supported only with versions 9.x, 10.x, and 12.x of the WebLogic Server. If you want to discover version 8.x of the WebLogic Server, you cannot specify which Management Agent to monitor which hosts' Managed Servers; the Management Agent used to perform discovery automatically monitors all WebLogic Servers within the version 8.x domain.

Function

This verb discovers one or more Oracle WebLogic Server Domains. It reads a file labeled domain_discovery_file to discover WebLogic Server versions 8.x, 9.x, 10.x, and 12.x. Note that if you attempt to discover an already discovered WebLogic Server, the discovered WebLogic Server domain will be refreshed.

Requirements

To discover the WebLogic Server, the Administration Server must be up and running. After initial discovery or during refresh of domain membership, the Administration Server is not required to be up for general WebLogic Server monitoring. After initial discovery or during refresh of domain membership, the Managed Server is not required to be up for general WLS monitoring. Oracle recommends ensuring all Managed Servers to be managed by Cloud Control be up during discovery.

domain_discovery_file is required; discovery cannot occur without it. You must create the CSV (comma-separated values) formatted file before performing discovery. To save the discovered components (WebLogic Server versions 9.x, 10.x, and 12.x only) to a specific Management Agent for monitoring, the discover_wls verb reads a second file labeled host_agent_mapping_file. If host_agent_mapping_file does not exist, the Management Agent specified in domain_discovery_file that performs the actual discovery is used as the Agent that monitors all discovered targets.

Usage with generate_discovery_input Verb

The generate_discovery_input verb creates a discovery input file automatically based on the targets discovered from the automatic discovery operation. You can then use this discovery input file in conjunction with the discover_wls verb to further automate the process of promoting discovered domains as fully managed targets.

Format

```
emcli discover_wls
        -input_file=domain_discovery_file:file_path
        [-input_file=host_agent_mapping_file:file_path]
        [-input_file=disable_target_types_file:file_path]
        [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file=domain_discovery_file**

  Fully-qualified path to a CSV (Comma-Separated Values) formatted file that contains one line of details per domain to be added. This is valid for WebLogic Server versions 8.x, 9.x, 10.x, and 12.x. Each line has the format shown for domain_discovery_file in the "File Structures" section below.

  Note the following points about the format of domain_discovery_file:

  Parameters —

  – The order of parameters is fixed. You must provide the parameters in the same order as shown for domain_discovery_file in the "File Structures" section below.

  – If you want to use a comma (,) in any of the parameters provided, you must escape the comma with a backslash as shown in This example, in which a backslash precedes the comma in the password my,pwd:

  ```
  10,domain123.xyx.us,11990,weblogic,my\,pwd,,,farm_
  demo,https://myco01.mycompany.com:3872/emd/main/
  ```

  Delimeters and Requirements —

  – Use a comma (,) as the delimiter.

  – Delimiters must be present even if the corresponding parameter is not provided. See the last line for domain_discovery_file in the "File Structures" section below.

  – If you want to use a comma (,) in one of the parameters provided, you must escape the comma (,) with a backslash. In This example, the password contains a comma:

  ```
  10,mco01.mycompany.com,7001,weblogic,welco\,me1,,,,my_farm_
  01,https://mco01.mycompany.com:3872/emd/main/
  ```

  – If you want to use a backslash in one of the parameters provided, you must escape the backslash with another backslash. In This example, the password contains a backslash:

  ```
  10,mco01.mycompany.com,7001,weblogic,we\,lco \ \ me1,,,,my_farm_
  01,https://mco01.mycompany.com:3872/emd/main/,true,false
  ```

  – The total number of tokens in each line is fixed and should be equal to 10.

  – The order of parameters is fixed. You must provide the parameters in the same order as specified in the sample file structure shown in the "File Structures" section below.

  For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=host_agent_mapping_file**

  Fully-qualified path of the CSV (Comma-Separated Values) formatted file that contains multiple lines of host system names where managed servers are to be monitored, and specifies the Management Agent used to monitor each host's managed servers. This is only valid for WebLogic Server versions 9.x, 10.x, and 12.x. Each line has the following format:

  ```
  <Discovered_target_host_machine_name>,<Agent_URL_to_save/monitor_the_host>
  ```

For example:

```
myco01.mycompany.com,https://myco01.mycompany.com:3872/emd/main/
myco02.mycompany.com,https://myco02.mycompany.com:3872/emd/main/
myco03.mycompany.com,https://myco03.mycompany.com:3872/emd/main/
```

Definitions for the parameters are as follows:

- **Discovered_target_host_machine_name**

  Host machine with installed WebLogic Servers that need to be discovered. Use full host names, such as myco01.mycompany.com instead of myco01.

- **Agent_URL_to_save/monitor_the_host**

  URL for the Management Agent to be used to monitor all discovered targets on the corresponding host.

Delimeters and Requirements —

- Use a comma (,) as the delimiter.
- The total number of tokens in each line is fixed and should be equal to 2.
- The order of parameters is fixed. You must provide the parameters in the same order as shown in the sample file structure shown in the "File Structures" section below.
- <target_host1> and <save_to_agent1> are both mandatory parameters.

For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=disable_target_types_file**

  Fully-qualified path to a CSV (Comma-Separated Values) formatted file containing multiple lines of internal target type names that should not be discovered.

  For example:

  ```
  oracle_soa_composite
  j2ee_application
  ```

  If the discover_wls verb is run against a Fusion Applications WebLogic Server domain, the disabled target types can include Fusion Applications target types.

- **debug**

  Runs this verb in verbose mode for debugging purposes.

File Structures

domain_discovery_file for WebLogic Server version 8.x

This example shows the structure of a sample domain_discovery_file for WebLogic Server version 8.x. The same Management Agent is used to discover and save the targets. OPT signifies an optional parameter. The last entry shows the format when the optional parameters, Administration Server Home Directory and Trusted Keystore Filename, are not provided.

```
<WebLogic Server version>,
<Administration Server host machine name>,
<Administration Server listen port>,
<Administration Server username>,
```

```
<Administration Server password>,
<Trusted Keystore Filename - required only if SSL enabled>,
<Administration Server Home Directory>,
<Agent Host>,
<Agent Host username>,
<Agent Host password>
```

For example:

```
8,mco02.mycompany.com,7001,weblogic,welcome1,,/u01/wls/,mco02.mycompany.com,oracle
,oracle
```

Definitions for the parameters are as follows for WebLogic Server version 8:

- **WebLogic Server version**

  Valid value is 8. This example shows a sample entry in domain_discovery_file to discover WebLogic Server version 8:

  ```
  8,myhost.us.mycompany.com,7001,weblogic,welcome1,,,myhost.us.mycompany.com,
  oracle,welcome1
  ```

- **Administration Server Host Machine Name**

  Full host name of the WebLogic Administration Server that needs to be discovered; for example, myhost.us.mycompany.com. This is a mandatory parameter.

- **Administration Server Listen Port**

  Listen port of the WebLogic Administration Server.

- **Administration Server Username**

  Login user name for the WebLogic Administration Server.

- **Administration Server Password**

  Login password for the WebLogic Administration Server.

- **Trusted Keystore Filename**

  Absolute path of the Trusted Keystore Filename. This is required if the Administration Server's port is SSL enabled. If the Management Agent is on a different system than the WebLogic Server to be managed, you must manually copy the Trusted Keystore file to an accessible directory on the Management Agent system prior to discovery, and then use this path.

- **Administration Server Home Directory**

  Absolute path of the directory where the weblogic.jar file is located. If the Management Agent is on a different system than the Administration Server, you must manually copy the weblogic.jar file (located in the <WEBLOGIC_HOME>/server/lib/ directory) to an accessible directory on the Management Agent system prior to discovery, and then use this path.

- **Agent Host**

  Host name of the Management Agent used to discover and monitor the targets.

- **Agent Host Username | Agent Host Password**

  Credentials of the operating system user of the Management Agent host. These credentials are used to discover any Oracle WebLogic Server domains.

domain_discovery_file for WebLogic Server versions 9.x, 10.x and 12.x

This example shows the structure of a sample domain_discovery_file for WebLogic Server versions 9.x, 10.x, and 12.x. OPT signifies an optional parameter. The last entry shows the format when optional parameters External Parameters, JMX Protocol, JMX Service URL, and Management Agent URL are not provided.

```
<WebLogic Server version>,
<Administration Server host machine name>,
<Administration Server listen port>,
<Administration Server username>,
<Administration Server password>,
<External Parameters - optional>,
<JMX Protocol - Required only if SSL enabled>,
<JMX Service URL - Required only if SSL enabled>,
<Unique Domain Identifier>,
<Agent URL/>,
<Discover Down Servers - optional - Default if not specified is false
   (starting with the Fusion Middleware Plug-in 12.1.0.3 release. Before this, the
    default was true)>,
<Use Credential Store - optional - Default if not specified is false>
<Enable Refresh Job - optional - Default if not specified is false>
<Use Host Name in Service URL - optional - Default if not specified is false>
```

For example:

```
10,mco01.mycompany.com,7001,weblogic,welcome1,,,,my_farm_
01,https://mco01.mycompany.com:3872/emd/main/,false,false
```

Definitions for the parameters are as follows:

- **WebLogic Server Version**

  Valid values are 9, 10, or 12. This example shows a sample entry in domain_discovery_file to discover WebLogic Server version 10:

  ```
  10,myco01.mycompany.com,7001,weblogic,welcome1,,,,soa_farm,
  https://myco02.mycompany.com:8723/emd/main/
  ```

- **Administration Server Host**

  Full host name of the WebLogic Administration Server that needs to be discovered; for example, myco01.mycompany.com. This is a mandatory parameter.

- **Port**

  Listen port of the WebLogic Administration Server.

- **Username**

  Login user name for the WebLogic Administration Server.

- **Password**

  Login password for the WebLogic Administration Server.

- **External Parameters**

  These parameters are passed to the Java process, which connects to the Administration Server. All of these parameters must begin with -D.

- **JMX Protocol**

  The Management Agent makes a JMX connection to the Administration Server to discover the domain's members. Valid values are t3, t3s, iiop, and iiops. If you do not provide a protocol, the t3 default is used.

- **JML Server URL**

  Makes a JMX connection to the Administration Server. If you do not specify this parameter, it is created based on the input parameters.

- **Unique Domain Identifier**

  Creates a unique target name. This parameter can contain only alphanumeric characters and the special character '_' and cannot contain any other special characters.

- **Agent URL**

  URL for the Management Agent used to discover the targets. If you do not provide a value, the local Management Agent present on the target WebLogic Server is used. If a Management Agent is not found on the target WebLogic Server, an error is displayed.

- **Discover Down Servers**

  If this value is true, the servers that are down are discovered. If false, the servers that are down are not discovered.

- **Use Credential Store**

  If this value is set to true, the verb retrieves the WebLogic credentials from the credential store.

Examples

This example reads the my_domains_info.csv file to determine the domains to be added to Cloud Control, and reads the my_agent_mapping.csv file to determine which Management Agents should monitor which host's managed servers.

```
emcli discover_wls
    -input_file=domain_discovery_file:\emcli\my_domains_info.csv
    -input_file=host_agent_mapping_file:\emcli\my_agent_mapping.csv
    -debug
```

This example manually redirects the output of discover_wls to a file using standard output redirect.

```
emcli discover_wls input_file=domain_discovery_file:"<fully_qualified_path_of_
domain_discovery_file/domain_discovery_file.csv>" > /tmp/emcli/output_file.out
```

download_ats_test_databank_file

Downloads the specified databank file corresponding to the given ATS test. If no databank alias is specified, the command downloads all databanks for the test.

Format

```
emcli download_ats_test_databank_file
        -name=<target_name>
        -type=<target_type>
        -testname=<test_name>
        -testtype=<test_type>
        [-databankAlias=<databank_alias>]
        [-output_dir=<output_directory>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the target.

- **type**

  Name of the target type.

- **testname**

  Name of the test.

- **testtype**

  Type of test.

- **databankAlias**

  Databank alias.

- **output_dir**

  Output directory. If the directory does not exist, it is created.

Examples

Example 1

This example downloads the databank corresponding to alias1 for the specified test.

```
emcli download_ats_test_databank_file -name="Service Name"
                                      -type="generic_service"
                                      -testname="Test Name"
                                      -testtype="OATS"
                                      -databankAlias="alias1"
```

Example 2

This example downloads all databanks corresponding to the specified test.

```
emcli download_ats_test_databank_file -name="Service Name"
                                      -type="generic_service"
                                      -testname="Test Name"
                                      -testtype="OATS"
```

download_ats_test_zip

Downloads the zip bundle corresponding to the specified ATS test.

Format

```
emcli download_ats_test_zip
        -name=<target_name>
        -type=<target_type>
        -testname=<test_name>
        -testtype=<test_type>
        [-output_dir=<output_directory>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the target.

- **type**

  Name of the target type.

- **testname**

  Name of the test.

- **testtype**

  Type of test.

- **output_dir**

  Output directory. If the directory does not exist, it is created.

Examples

```
emcli download_ats_test_zip -name="Service_Name"
                            -type="Generic_Service"
                            -testname="Test_Name"
                            -testtype="OATS"
                            -output_dir="outputDirectory"
```

download_update

Downloads an update.

Format

```
emcli download_update
        -id="internal id"
```

Parameters

- **id**

  Internal identification for the update to be downloaded.

Examples

This example submits a job to download an update, and prints the job execution ID upon submission.

```
emcli download_update
        -id="914E3E0F9DB98DECE040E80A2C5233EB"
```

dump_activity_list

Prints the list of all current activities.

Format

```
emcli dump_activity_list
```

edit_dbprofile

Edits the schedule and purge policy of an existing database profile.

Format

```
emcli edit_dbprofile
      -comp_loc="Database Profile component location in software library"
        [-schedule=
          [NONE]|[frequency:interval|weekly|monthly|yearly];
          start_time:yy-MM-dd HH:mm;
          end_time:yy-MM-dd HH:mm;
          [repeat:#m];
          [months:#,#,#,...];
          [days:#,#,#,...];
          [tz:{java timezone ID}];
          [grace_period:xxx];
        ]
        [-purge_policy= DAYS|SNAPSHOTS: number]

[ ] indicates that the parameter is optional.
```

Parameters

- comp_loc

  A combination of the database profile location and name.

- schedule
  - frequency: The frequency type with which the databse profile will be created. It can be an interval (in minutes), weekly, monthly, or yearly.
  - start_time: Denotes the start time of Database Profile Component Creation in the format yy-MM-dd HH:mm.
  - end_time: Denotes the end time of Database Profile Component Creation Repetition in the format yy-MM-dd HH:mm
  - repeat: The repetition rate at which database profile will be created. If the frequency is an interval, then repeat is in minutes.
  - months: The number of months after which the repetition of Database Profile Component Creation will occur.
  - days: The number of days after which repetition of Database Profile Component Creation will occur.
  - tz: The time zone ID, for example tz:America/New_York.
  - grace_period: A period of time in minutes that defines the maximum permissible delay when attempting to create a databasse profile. If the job system cannot start the execution within a time period equal to the scheduled time plus the grace period, it will set the create database profile to be skipped. By default, the grace period is indefinite.

- purge_policy

  You can purge the collected data based on a specified number of days (DAYS) or a count of snapshots (SNAPSHOT). If the purge_policy parameter is not specified, then it is defaulted to NONE.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example edits the schedule and purge policy database profile RMAN_Profile with the location `Database Provisioning Profiles/11.2.0.4.0/linux_x64`.

```
emcli edit_dbprofile
      -comp_loc="Database Provisioning Profiles/11.2.0.4.0/linux_x64/RMAN_Profile"
      -schedule="frequency:interval;start_time:14-10-05 05:30;end_time:
         14-10-12 05:23;repeat:30;grace_period:60;tz:America/New_York"
      -purge_policy=DAYS:2
```

edit_sl_rule

Edits the service-level rule for the specified service.

Format

```
emcli edit_sl_rule
      -name="target name"
      -type="target type"
      [-expSL="expected service level value"]
      [-repeatSequence="days repeat sequence"]
      [-startTime="start time"]
      [-endTime="end time"]
      [-availStatesInclude="included availability states"]
      [-availStatesExclude="excluded availability states"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Identifies the target name.

- **type**

  Identifies the target type. Use `emcli get_targets` to get the target type.

- **expSL**

  Specifies the expected service-level rule. Values must be any number between 0 and 100.

- **repeatSequence**

  Specifies the days in which the service-level rule is to be applied. Identify the days value from these comma-separated values: MON, TUE, WED, THU, FRI, SAT, SUN.

- **startTime**

  Specifies the time of day that the application of the service-level rule is to begin. Enter the time format as: HH:min

- **endTime**

  Specifies the time of day that the application of the service-level rule is to end. Enter the time format as: HH:min

- **availStatesInclude**

  Specifies the availability states (apart from UP) that are to be included while computing the service-level rule. Values are: BLACKOUT | UNKNOWN

- **availStatesExclude**

  Specifies the availability states (apart from UP) that are to be excluded while computing the service-level rule. Values are: BLACKOUT | UNKNOWN

Examples

Example 1

Update the MyService service-level rule to begin at 6 a.m. on Mondays and Tuesdays:

```
emcli edit_sl_rule
      -name="MyService"
      -type="generic_service"
      -expSL="90.0"
      -repeatSequence="MON,TUE"
      -startTime="06:00"
      -endTime="23:00"
      -availStatesInclude="BLACKOUT"
      -availStatesExclude="UNKNOWN"
```

enable_audit

Enables auditing for ALL and BASIC user operations. For other operations, see the update_audit_settings verb.

Format

```
emcli enable_audit
      [-level=basic]

[ ] indicates that the parameter is optional
```

Parameters

- **level=basic**

  Enables auditing for BASIC user operations.

Examples

Example 1

This example enables auditing for all operations.

```
emcli enable_audit
```

Example 2

This example enables auditing for LOGIN, LOGOUT, DB_LOGIN, and DB_LOGOUT.

```
emcli enable_audit -level=basic
```

enable_config_history

Enables configuration history computation for a target type.

Format

```
emcli enable_config_history -target_type="{target type|'*'}"
```

Parameters

- **target_type**

  Target type for which the configuration history is being enabled. The value should be the internal name or "*" to indicate all target types.

Examples

Example 1

This example enables configuration history computation for the host target type.

```
emcli enable_config_history -target_type="host"
```

Example 2

This example enables configuration history computation for all target types.

```
emcli enable_config_history -target_type="*"
```

enable_forwarder_agents

Takes a list of agents and marks each agent as a forwarder agent.

Format

```
emcli enable_forwarder_agents
      -agent_list="agent_list"

[ ] indicates that the parameter is optional.
```

Parameters

- agent_list

  List of agents that need to be registered as forwarders. The agents must be separated by space.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example enables agent1 and agent2 as forwarding agents.

```
emcli enable_forwarder_agents
      -agent_list="agent1 agent2..."
```

enable_sla

Enables an SLA for a target.

Format

```
emcli enable_sla
    -targetName=<target_name>
    -targetType=<target_type>
    -slaName=<SLA_name>
    [-now]
    [-versionStart=<MM/dd/yyyy hh:mm a>]

[ ] indicates that the parameter is optional
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **slaName**

  Name of the SLA.

- **now**

  Enables the SLA now, or uses versionStart for a specific time.

- **versionStart**

  Specifies when the computation of the SLA should start.

Examples

Example 1

This example immediately enables an SLA named 'gold_sla' for target my_service (generic_service).

```
emcli enable_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla' -versionNum=2 -now
```

Example 2

This example enables a SLA named 'gold_sla' for target my_service (generic_service). It becomes active and starts computing at '09/23/2012 3:30 PM'.

```
emcli enable_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla' -versionNum=2 -versionStart='09/23/2012 3:30 PM'
```

enable_test

Enables monitoring of a Services test. It pushes the Service test collection to all the beacons.

Format

```
emcli enable_test
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **testname**

  Test name.

- **testtype**

  Test type.

Examples

This example enables the HTTP test named `MyTest` for the `generic_service` target named `MyTarget`.

```
emcli enable_test -name='MyTarget' -type='generic_service'
      -testname='MyTest' -testtype='HTTP'
```

execute_hostcmd

Executes a host command across a set of targets.

Format

```
emcli execute_hostcmd
      -cmd=<host_command"
      -osscript=<script_to_be_executed>
      -targets=<name1:type1;name2:type2;...>
      -credential_set_name=<name>
      [-input_file=<parameter_tag:script_file>]

[ ] indicates that the parameter is optional
```

Parameters

- **cmd**

  Host_command can be any valid host command or group of host commands.

- **osscript**

  OS script to be executed with the cmd parameter.

- **targets**

  List of target-name, target-type pairs. The host command is executed across this list of Enterprise Manager targets. All targets must be of the type `host` or `composite`, which represents a group of targets. If it is a group, the group is expanded to extract all the host targets, and the host command is executed across these host targets.

- **credential_set_name**

  The `credential_set_name` parameter refers to the set name of the preferred credentials stored in the Enterprise Manager repository. If this parameter is not present, `HostCredsNormal` is used for executing host commands. For the `host` target type, two credential sets exist:

  – `HostCredsNormal` — Default unprivileged credential set for a host target

  – `HostCredsPriv` — Privileged credential set for a host target

  The credential set parameter can only be specified when the override credential parameters such as `username` and `password` are not present.

  If provided, the you must fully specify the override credential parameters. For host command, `username` and `password` must be specified together.

- **input_file**

  Used in conjunction with `-osscript` , this enables you to load the contents of an OS script. The `-input_file` specifies a mapping between a tag and a local file path. The tag is specified in lieu of actual osscript contents of the `-osscript` . The tag must not contain colons (:) or semi-colons (;).

  For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

Example 1

This example executes the host command `ls -l;` against the target `stach.example.com:host` and host targets contained in the group `grp`. The stored `HostCredsPriv` preferred credentials are used for all the targets.

```
emcli execute_hostcmd
      -cmd="ls -l;"
      -credential_set_name="HostCredsPriv"
      -targets="stach.example.com:host;grp:composite"
```

Example 2

This example loads the contents of the script `/scratch/dba_scripts/shellscript.sh` into the value of `-osscript` and executes it against target `reference.example.com:host` and host targets contained in the group `grp`. The stored `HostCredsNormal` preferred credentials are used for all the targets.

```
emcli execute_hostcmd
      -cmd="/bin/sh -s"
      -osscript="FILE"
      -input_file="FILE:/scratch/dba_scripts/shellscript.sh"
      -credential_set_name="HostCredsNormal"
      -targets="reference.example.com:host;grp:composite"
```

execute_sql

Executes a SQL command across a set of targets.

Format

```
emcli execute_sql
      -sql=<sql_command>
      -targets=<name1:type1;name2:type2;...>
      -credential_set_name=<name>
      [-input_file=<parameter_tag:script_file>]

[ ] indicates that the parameter is optional
```

Parameters

- **sql**

  "sql command" is a single SQL statement.

- **targets**

  List of target-name, target-type pairs. The SQL command executes across this list of Enterprise Manager targets. All targets must be of the type `oracle_database` or `composite`, which represents a group of targets. If it is a group, the group expands to extract all the database targets, and the SQL command is executed across these database targets.

- **credential_set_name**

  Refers to the set name of the preferred credentials stored in the Enterprise Manager repository. If this parameter is not present, the `DBCredsNormal` and `DBHostCreds` credential set is used for executing SQL commands. For each target type, several credential sets exist:

  – HostCredsNormal — Default unprivileged credential set for a host target

  – HostCredsPriv — Privileged credential set for a host target

  – DBHostCreds — Host credential set for an `oracle_database` target

  – DBCredsNormal — Default normal credential set for an oracle_database target

  – DBCredsSYSDBA — sysdba credential set for an `oracle_database` target

  You can only specify the `credential_set_name` parameter when the override credential parameters such as `[db_|host_]username` and `[db_|host_]password` are not present. If provided, the override credential parameters must be specified fully. For the SQL commands, `db_username`, `db_password`, `db_role`, `host_username`, and `host_password` must be present.

- **input_file**

  Used in conjunction with the `-sql` option, this option enables you to load the contents of a SQL script. The `-input_file` option specifies a mapping between a tag and a local file path. The tag is specified in lieu of an actual SQL command for the `-sql` . The tag must not contain colons (:) or semi-colons (;).

  For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

Example 1

This example executes the SQL command `select * from sysman.mgmt_targets;` against the target `database:oracle_database` and database targets contained in the group `grp`. The stored SYSDBA preferred credentials are used for all the targets.

```
emcli execute_sql
      -sql="select * from sysman.mgmt_targets;"
      -credential_set_name="DBCredsSYSDBA"
      -targets="database:oracle_database;grp:composite"
```

Example 2

This example loads the contents of the script `/scratch/dba_scripts/enterprise_schema.sql` into the value of `-sql`, and executes it against target `database:oracle_database` and database targets contained in the group `grp`. The stored SYSDBA preferred credentials are used for all the targets.

```
emcli execute_sql
      -sql="FILE"
      -input_file="FILE:/scratch/dba_scripts/enterprise_schema.sql"
      -credential_set_name="DBCredsSYSDBA"
      -targets="database:oracle_database;grp:composite"
```

Example 3

This example executes the SQL command against "asm:osm_instance" and ASM targets contained in the group 'grp'. The SYSASM preferred credentials are used for all the targets.

```
emcli execute_sql
      -sql="select * from sysman.mgmt_targets;"
      -credential_set_name="ASMCredsSYSASM"
      -targets="asm:osm_instance;grp:composite"
```

export_adm

Exports an Application Data Model to the specified directory with the specified file name.

Format

```
emcli export_adm
     -adm_name=<application_data_model_name>
     [-directory=<directory_path>]
     [-file_name=<file_name>]

[ ] indicates that the parameter is optional
```

Parameters

- **adm_name**

  Application data name that will be exported.

- **directory**

  Directory where the Application Data Model is to be exported. If the directory is not specified, the file is saved in the current directory.

- **file_name**

  Name of the file where the Application Data Model will be exported. If the file name is not specified, the default file name is the same as the specified Application Data Model name. If the file name does not have an extension, '.xml' is the default extension.

Output

Success/error messages.

Examples

Example 1

This example exports the Application Data Model Sample_ADM to the sample_adm.xml file.

```
emcli export_adm
       -directory=/home/user
       -adm_name=Sample_ADM
       -file_name=sample_adm.xml
```

export_charge_plans

Exports charge plan metadata to an XML file.

Format

```
emcli export_charge_plans
        [-charge_plan="plan_name"[-entity_type = chargeback_entity_type]]
        [-start_date=ddmmyyyy]
        -file=file_name

[ ] indicates that the parameter is optional
```

Parameters

- **charge_plan**

  Name of the charge plan to be exported. If this parameter is not specified, all charge plan metadata is exported.

- **entity_type**

  Name of the Chargeback entity type whose charge plan is to be exported. If this parameter is not specified, all entity type charge rates in the charge plan are exported.

- **start_date**

  Start date of the report cycle whose charge plan metadata is to be exported. The start date value must be in ddmmyyyy format. If this parameter is not specified, the start date of the current report cycle is used.

- **file**

  Absolute path to which to export the metadata.

Examples

Example 1

This example exports metadata of all charge plans that were active in the current report cycle to the file /home/allplans.xml:

```
emcli export_charge_plans
        -file=/home/allplans.xml
```

Example 2

This example exports metadata of charge plan Plan A, if active in the current report cycle, to the file /home/plans.xml:

```
emcli export_charge_plans
        -charge_plan="Plan A"
        -file=/home/plans.xml
```

Example 3

This example exports metadata of the host entity type associated with charge plan Plan A, if active in the current report cycle, to the file /home/plans.xml:

```
emcli export_charge_plans
        -charge_plan="Plan A"
        -entity_type=host
```

```
        -file=/home/plans.xml
```

Example 4

This example exports metadata of charge plan Plan A, if active in the report cycle starting on 01062014, to the file /home/plans.xml:

```
emcli export_charge_plans
        -charge_plan="Plan A"
        -start_date=01062014
        -file=/home/plans.xml
```

export_compliance_group

Exports a compliance group definition and all of its element definitions given the name, author, and version.

Format

```
emcli export_compliance_group
      -name=<name>
      -author=<author>
      -version=<name>
      -output_file=<file>
```

Parameters

- **name**

  Name of the group to be exported.

- **author**

  Author of the group to be exported.

- **version**

  Version of the group to be exported.

- **output_file**

  Name of the exported file.

Examples

Example 1

```
emcli export_compliance_group \
     -name="foo" \
     -author="Jonas" \
     -version="99" \
     -output_file="$HOME/reports/group.xml"
```

export_compliance_standard_rule

Exports a rule to the specified files.

Format

```
export_compliance_standard_rule
  -name=<name>
  -target_type=<target_type>
  -output_file=<file>
```

Parameters

- **name**

  Name of the rule to be exported.

- **target_type**

  Target type of the rule to be exported.

- **output_file**

  Name of the exported file.

Examples

Example 1

```
emcli export_compliance_standard_rule \
     -name="foo" \
     -target_type="weblogic_j2eeserver" \
     -output_file="$HOME/reports/rule.xml"
```

export_custom_charge_items

Exports user-defined charge item metadata to the specified XML file.

Format

```
emcli export_custom_charge_items
        -entity_type="entity_type"
        -file=output_file
```

Parameters

- **entity_type**

  Name of the entity whose charge item metadata you want to export.

- **file**

  Full path of the file to which to write user-defined charge items associated with the specified entity type.

Examples

This example writes user-defined charge item metadata associated with the host entity type to the myhost.xml file in the home directory:

```
emcli export_custom_charge_items
        -entity_type="host"
        -file=/home/myhost.xml
```

export_jobs

Exports all matching job definitions in Enterprise Manager, including Corrective Actions. System jobs and nested jobs are excluded.

Format

```
emcli export_jobs
   -export_file=<zip_file_name>"
   [-name="job_name1;job_name2;..."]
   [-type="job_type1;job_type2;..."]
   [-targets="tname1:ttype1;tname2:ttype2;..."]
   [-owner="owner1;owner2;..."]
   [-preview]

[ ] indicates that the parameter is optional
```

Parameters

- **export_file**

  Zip file name to be created.

- **name**

  Job name pattern to be used for filtering. Semicolon-separated job names can be provided. When filtering by a single value, wildcard char(% or _) can also be used. Wildcard "%"matches one or more characters. "_"(underscore) matches exactly one character.

- **type**

  Job type pattern to be used for filtering. Semicolon-separated job types can be provided. When filtering by sa ingle value, wildcard chars(% or _) can be used.

- **targets**

  Target name, type pattern to be used for filtering. Semicolon-separated target names and types can be provided. When filtering by a single value, wildcard chars(% or _) can be used.

- **owner**

  Owner of the jobs to be used for filtering. Semicolon-separated job owners can be provided.

- **preview**

  Jobs in the Enterprise Manager site matching the filter criteria are shown to stdout. Jobs are not exported to any file.

Output Columns

Success/Error messages.

Examples

Example 1

This example exports job definitions for jobs MYJOB1 and MYJOB2 to job_data.zip:

```
emcli export_jobs -name=MYJOB1;MYJOB2 -export_file=jobsdata.zip
```

Example 2

This example exports job definitions for any jobs owned by user name starting with ADMIN.

```
emcli export_jobs -owner=ADMIN% -export_file=jobsdata.zip
```

Example 3

This example exports job definitions for jobs MYJOB1 and MYJOB2 to job_data.zip:

```
emcli export_jobs -name=MYJOB1;MYJOB2 -export_file=jobsdata.zip
```

Example 4

This example exports job definitions for any jobs owned by user name starting with ADMIN.

```
emcli export_jobs -owner=ADMIN% -export_file=jobsdata.zip
```

export_masking_definition

Exports a masking definition in XML format.

Format

```
emcli export_masking_definition
     -definition_name=<masking_definition_name>
     [-path=file_path]
     [-file=file_name]

[ ] indicates that the parameter is optional
```

Parameters

- **definition name**

  Masking definition name.

- **path**

  Path for the file name to save the masking script. The file name is auto-generated. -path and -file are mutually exclusive. Only an absolute path is allowed.

- **file**

  File name to save the masking script. The file name must include the absolute path. -path and -file are mutually exclusive.

Output Columns

Success/Error messages.

Examples

Example 1

This example exports the masking definition mask_hr_data to an XML file at the specified path:

```
emcli export_masking_definition
      -definition_name=mask_hr_data
      -path=/tmp/
```

Example 2

This example exports the masking definition mask_hr_data to an XML file named abc.xml:

```
emcli export_masking_definition
      -definition_name=mask_hr_data
      -file=/tmp/abc.xml
```

export_metric_extension

Exports a metric extension archive file.

Format

```
emcli export_metric_extension
       -file_name=<metric_extension_archive_name>
       -target_type=<metric_extension_target_type>
       -name=<metric_extension_name>
       -version=<metric_extension_version>
```

Parameters

- **file_name**

  Name of the metric extension archive file to export into.

- **target_type**

  Target type of the metric extension.

- **name**

  Name of the metric extension.

- **version**

  Version of the metric extension to be exported.

Example

This example creates an archive of a metric extension of a given target type, name, and version.

```
emcli export_metric_extension -file_name=<name of the metric extension archive>
-target_type=<target type of the metric extension> -name=<name of the metric
extension -version=<version of the metric extension>
```

export_report

Exports an Information Publisher report definition and all of its element definitions given its title and owner.

Format

```
emcli export_report
      -title=<report_title>
      -owner=<report_owner>
      -output_file=<file>
```

Parameters

- **title**

  Title of the report to export. To export copies of Oracle-provided reports, the title value should be the internal report title stored in the repository. To avoid using the internal title, make a copy of the report and provide your own custom title, then use your title to export the report.

- **owner**

  The owner of the report to export. The logged-in emcli user must have view privilege for the report. Target names are not exported. The report is uniquely defined using title and owner, so both must be supplied.

- **output_file**

  Name of the exported file.

Examples

```
emcli export_report
      -title=Maintenance_Report
      -owner=SHIFT1_OPERATOR
      -output_file=$HOME/reports/maint_report.xml
```

export_sla

Extracts the configuration details of an SLA into a local file. If you do not specify slaName and/or version, multiple SLA are exported to the same output file.

Format

```
emcli export_sla
    -targetName=<target_name>
    -targetType=<target_type>
    [-slaName=<SLA_name>]
    -output_file=<output_filename>

[ ] indicates that the parameter is optional
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **slaName**

  Name of the SLA.

- **output_file**

  Output file name of the template. If the file does not exist, it is created; if it already exists, it is overwritten. (This assumes that the extract operation was successful. If the operation fails, no files are created, and any existing files remain unchanged.)

Example

This example creates an output file named 'service_sla.xml' that contains configuration details of the 'gold_sla' SLA for the target 'my_service'.

```
emcli export_sla
          -targetName='my_service'
          -targetType='generic_service'
          -slaName='gold_sla'
          -output_file='service_sla.xml'
```

export_standard

Exports a standard from the repository to an XML file.

Format

```
emcli export_standard
      -name=<name>
      -author=<author>
      -version=<name>
      -output_file=<file>
```

Parameters

- **name**

  Name of the standard to be exported.

- **author**

  Author of the standard to be exported.

- **version**

  Author of the standard to be exported.

- **output_file**

  Name of the exported file.

Example

```
emcli export_standard \
     -name=foo \
     -author=Curly \
     -version=99 \
     -output_file=$HOME/reports/standard.xml
```

export_subset_definition

Exports the specified subset definition as an XML file at the specified directory location.

Format

```
emcli export_subset_definition
   -subset_name=<subset_definition_name>
   [-file_name=<file_name>]
   [-directory=<directory_path>]

[ ] indicates that the parameter is optional
```

Parameters

- **subset_name**

  Subset definition name to export.

- **file_name**

  File name to save the exported file. If you do not specify the file name, it is saved under the subset definition name. If it is specified without an extension, '.xml' is used as the default extension.

- **directory**

  Directory location to save the exported file. If you do not specify a directory, the file is saved in the current directory.

Output

Export success or error message.

Examples

Example 1

This example exports a subset definition with the name HR_Subset as XML in the current directory.

```
emcli export_subset_definition -subset_name=HR_Subset
```

Example 2

This example exports a subset definition with the name HR_Subset as XML with the name HR_Subset_Export at the directory path /scratch/subset.

```
emcli export_subset_definition -directory=/scratch/subset -subset_name=HR_Subset
-file_name=HR_Subset_Export
```

export_template

Exports a monitoring template and also exports UDMs in the template. You can export a template to the file system in the form of an XML file, or you can print it on standard output in XML form.

Format

```
emcli export_template
      -name=<name>
      -target_type=<target_type>
      [-output_file=<file_for_exported_template>]
      [-archive]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the template. The name and target type uniquely identify a template.

- **target_type**

  Target type of the template.

- **output_file**

  Specifies the file to output the template. If not specified, the template prints to `stdout`.

- **archive**

  Indicates that the template must be exported as a zip file. When a Metric Extension is included in the template, this option is required to export the template as a zip file.

Examples

Example 1

This example shows that template XML specified by name `HOST_TEMP1` and target type `host` will be output to the screen.

```
emcli export_template -name=HOST_TEMP1 -target_type=host
```

Example 2

This example shows that template XML specified by name `HOST_TEMP1` and target type `host` will be created in the `test.xml` file.

```
emcli export_template -name=HOST_TEMP1 -target_type=host -output_file=test.xml
```

Example 3

This example shows that the template archive specified by name HOST_TEMP1 and target type host will be created in the test.zip file.

```
emcli export_template -name=HOST_TEMP1  -target_type=host -output_file=test.zip
-archive
```

export_update

Exports a Self Update archive file from Enterprise Manager to the specified location.

Format

```
emcli export_update
        -id="internal id"
        -dir="dir"
        -omslocal
emcli export_update
        -id="internal id"
        -dir="dir"
        -host="hostname"
        [-credential_set_name="setname"] | -credential_name="name"
          -credential_owner="owner"

[ ] indicates that the parameter is optional
```

Parameters

- **id**

  Internal identification for the update to be exported.

- **dir**

  Complete path of the directory where the update is to be exported.

- **omslocal**

  Flag specifying that the directory is accessible from the OMS.

- **host**

  Target name for a host target where the update is to be exported.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target. Can be one of the following:

  HostCredsNormal — Default unprivileged credential set

  HostCredsPriv — Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. You must specify this option along with the credential_owner option.

- **credential_owner**

  Owner of a named credential stored in the repository. You must specify this option along with the credential_name option.

Examples

Example 1

This example exports the update archive file to /u01/common/. The directory must exist on the OMS host. In a multiple OMS setup, the request can be processed by any OMS, so the directory should be accessible from the OMS processing the request. This

usually means that the directory must be on a shared location accessible from all OMSes.

```
emcli export_update
        -id="914E3E0F9DB98DECE040E80A2C5233EB"
        -dir="/u01/common/"
        -omslocal
```

Example 2

This example exports the update archive file to /u01/common/ on host host1.example.com. The host must be the managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The preferred unprivileged credentials for host host1.example.com are used to push the remote file.

```
emcli export_update
        -id="914E3E0F9DB98DECE040E80A2C5233EB"
        -dir="/u01/common/"
        -host="host1.example.com"
        -credential_set_name="HostCredsNormal"
```

Example 3

This example exports the update archive file to /u01/common/ on host host1.example.com. The host must be the managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The named credentials "host1_creds" owned by user "admin1" are used to push the remote file.

```
emcli export_update
        -id="914E3E0F9DB98DECE040E80A2C5233EB"
        -dir="/u01/common/"
        -host="host1.example.com"
        -credential_name="host1_creds"
        -credential_owner="admin1"
```

extend_as_home

Clones the specified Application Server Oracle Home or Software Library component from the target host to specified destinations. The new hosts join an existing cluster. For a Portal and Wireless install, OID user and password are also needed. For a J2EE instance connected to only a database-based repository, a DCM Schema password is needed.

Passing Variables Through EM CLI

When working with variables such as `%perlbin%` or `%oracle_home%`, EM CLI passes variable values from the current local environment instead of the variables themselves. To pass variables through an EM CLI command, as might be the case when using the `-prescripts` or `-postscripts` options, you can place the EM CLI command in a batch file and replace all occurrences of `%` with `%%`.

Format

```
emcli extend_as_home
       -input_file="dest_properties:file_path"
       -list_exclude_files="list of files to exclude"
       -isSwLib="true/false"
       -tryftp_copy="true/false"
       -jobname="name of cloning job"
       -iasInstance=instance
       -clustername=name of the cluster to join
       -oldIASAdminPassword=oldpass
       -newIASAdminPassword=newpass
       [-oiduser=oid admin user]
       [-oidpassword=oid admin password]
       [-dcmpassword=dcm schema password]
       [-prescripts=script name to execute"]
       [-run_prescripts_as_root="true/false"]
       [-postscripts=script to execute"]
       [-run_postscripts_as_root="true/false"]
       [-rootscripts=script name to execute"]
       [-swlib_component ="path:path to component;version:rev"]
       [-source_params="TargetName:name;HomeLoc:loc;HomeName:name;
         ScratchLoc:Scratch dir Location"
       [-jobdesc="description"]

[ ] denotes that the parameter is optional
```

Options

- **input_file=dest_properties**

  File containing information regarding the targets. Each line in the file corresponds to information regarding one destination.

  Format:

  ```
  Destination Host Name1;Destination Home Loc; Home Name; Scratch Loca
  ```

  For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=list_exclude_files**

  Comma-separated list of files to exclude. This is not required if the source is a Software Library. You can use an asterisk "*" as a wildcard.

For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **isSwLib**

  Specifies whether it is an Oracle Home database or Software Library.

- **tryftp_copy**

  Try FTP to copy or not. You should set the FTP copy option to false when using EM CLI from the command line.

- **jobname**

  Name of the cloning job.

- **iasInstance**

  Application Server instance.

- **clustername**

  Name of the cluster to join.

- **oldIASAdminPassword**

  Old Application Server administrator password.

- **newIASAdminPassword**

  New Application Server administrator password.

- **oiduser**

  OID administrator user.

- **oidpassword**

  OID administrator password.

- **dcmpassword**

  DCM schema password.

- **prescripts**

  Path of the script to execute.

  Note: Double-quoted parameters can be passed using an escape (\) sequence. For example:

  ```
  prescripts=" <some value here>=\"some value here\" "
  ```

- **run_prescripts_as_root**

  Run prescripts as `root`. By default, this option is set to false.

- **postscripts**

  Path of the script to execute.

- **run_postscripts_as_root**

  Runs postscripts as `root`. By default, this option is set to false.

- **rootscripts**

  Path of the script to execute. You can use the job system environment variables (`%oracle_home%`, `%perl_bin%`) to specify script locations.

- **swlib_component**

  Path to the Software Library to be cloned. `isSwLib` must be true in this case.

- **source_params**

  Source Oracle home information. `isSwLib` must be false in this case.

- **jobdesc**

  Description of the job. If not specified, a default description is generated automatically.

Examples

```
emcli extend_as_home
   -input_file="dest_properties:/home/destinations.txt"
   -list_exclude_files="centralagents.lst"
   -isSwLib="false"
   -tryftp_copy="false"
   -jobname="extend as home"
   -iasInstance="asinstancename"
   -isIas1013="false"
   -clustername=ascluster
   -oldIASAdminPassword="oldpassword"
   -newIASAdminPassword="newpassword"
   -prescripts="/home/abc/myscripts"
   -run_prescripts_as_root="true"
   -rootscripts="%oracle_home%/root.sh"
   -source_params="TargetName:host.domain.com;HomeLoc=/home/oracle/appserver1;
    HomeName=oracleAppServer1;ScratchLoc=/tmp"
```

extend_crs_home

Extends an Oracle Clusterware cluster, using an Oracle Clusterware source home location or an Oracle Clusterware Software Library component, to specified destinations. If a component is used, you must provide information for a host that is part of the current cluster, along with the Oracle Home name and home location. When cloning from a source home, you do not need to pass source information twice (`-srchost`, `-home_name`, and `-home_location`). This information is extracted from the home. These are only needed when cloning from a Software Library component.

Format

```
emcli extend_crs_home
   -input_file="dest_properties:file_path"
   -list_exclude_files="list of files to exclude"
   -clusternodes="node1;node2;node3;node4"
   -clustername="name of cluster to create"
   -isSwLib="true/false"
   -tryftp_copy="true/false"
   -jobname="name of cloning job"
    [-srchost=name of a host node present on the cluster being extended"]
    [-home_name="home name on a host for the existing Oracle Clusterware
      cluster"]
    [-home_location="location on a host for the existing Oracle Clusterware
      cluster"]
    [-prescripts=script name to execute"]
    [-run_prescripts_as_root="true/false"]
    [-postscripts=script to execute"]
    [-run_postscripts_as_root="true/false"]
    [-rootscripts=script name to execute"]
    [-swlib_component ="path:path to component;version:rev"]
    [-source_params="TargetName:name;HomeLoc:loc;HomeName:name;
      ScratchLoc:Scratch dir Location"]
    [-jobdesc="description"]

[ ] denotes that the parameter is optional
```

Options

- **input_file**

  File containing information regarding the targets. Each line in the file corresponds to information regarding one destination.

  Format:

  ```
  Destination Host Name1;Destination Node Name;Scratch
  Location;PVTIC;VirtualIP;
  ```

  For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **list_exclude_files**

  Comma-separated list of files to exclude. Not required if the source is a Software Library. You can use an asterisk "*" as a wildcard.

- **clusternodes**

  List of current nodes in the cluster.

- **clustername**

  Name of the cluster to create.

- **isSwLib**

  Specifies whether it is an Oracle Home database or Software Library.

- **tryftp_copy**

  Try FTP to copy or not. You should set the FTP copy option to false when using EM CLI from the command line.

- **jobname**

  Name of the Cloning job.

- **srchost**

  Name of a host that is part of the Oracle Clusterware cluster being extended.

- **home_name**

  Name of home used by all the current Oracle Clusterware cluster nodes.

- **home_location**

  Home location used by all the current Oracle Clusterware cluster nodes.

- **prescripts**

  Path of the script to execute.

  Note: Double-quoted parameters can be passed using an escape (\) sequence. For example:

  ```
  prescripts=" <some value here>=\"some value here\" "
  ```

- **run_prescripts_as_root**

  Run prescripts as `root`. By default, this option is set to false.

- **postscripts**

  Path of the script to execute.

- **run_postscripts_as_root**

  Run postscripts as `root`. By default, this option is set to false.

- **rootscripts**

  Path of the script to execute. You can use the job system environment variables (`%oracle_home%`, `%perl_bin%`) to specify script locations.

- **swlib_component**

  Path to the Software Library to be cloned. `isSwLib` must be true in this case.

- **source_params**

  Source Oracle home info. `isSwLib` must be false in this case.

- **jobdesc**

  Description of the job. If not specified, a default description is generated automatically.

Examples

```
emcli extend_crs_home -input_file="dest_properties:crs.prop"  -list_exclude_
files=""
   -isSwLib="false"
   -tryftp_copy="false" -jobname="crs extend job"
   -home_name="cloneCRS1"
   -home_location="/scratch/scott/cloneCRS1 "
   -clusternodes="node1;node2" -clustername="crscluster"
   -postscripts="%perlbin%/perl%emd_root%/admin/scripts/cloning/samples/
     post_crs_extend.pl ORACLE_HOME=%oracle_home%"
   -run_postscripts_as_root="false" -rootscripts="%oracle_home%/root.sh"
   -source params="TargetName:testhost;HomeLoc:
     /scratch/scott/cloneCRS1;HomeName:cloneCRS1;ScratchLoc:/tmp"
```

Passing Variables Through EM CLI

When working with variables such as `%perlbin%` or `%oracle_home%`, EM CLI passes variable values from the current local environment instead of the variables themselves. To pass variables through an EM CLI command, as might be the case when using the `-prescripts` or `-postscripts` options, you can place the EM CLI command in a batch file and replace all occurrences of `%` with `%%`.

extend_rac_home

Extends a RAC cluster by cloning a specified Oracle Home location or a RAC Software Library component to specified destinations. If a component is used, you must provide information for a host that is part of the current cluster, along with the Oracle Home name and home location. When cloning from a source home, this information is automatically extracted from the home.

Format

```
emcli extend_rac_home
   -input_file="dest_properties:file_path"
   -list_exclude_files="list of files to exclude"
   -isSwLib="true/false"
   -tryftp_copy="true/false"
   -jobname="name of cloning job"
   -clusternodes="node1;node2;node3;node4"
    [-srchost=name of a host node present on the RAC cluster being extended"]
    [-home_name="home name on a host for the existing RAC cluster"]
    [-home_location="location on a host for the existing RAC cluster"]
    [-prescripts="script name to execute"]
    [-run_prescripts_as_root="true/false"]
    [-postscripts="script to execute"]
    [-run_postscripts_as_root="true/false"]
    [-rootscripts="script name to execute"]
    [-swlib_component ="path:path to component;version:rev"]
    [-source_params="TargetName:name;HomeLoc:loc;HomeName:name;
      ScratchLoc:Scratch dir Location"]
    [-jobdesc="description"]

[ ] denotes that the parameter is optional
```

Options

- **input_file**

  File containing information regarding the targets. Each line in the file corresponds to information regarding one destination.

  Format:

  ```
  Destination Host Name;Destination Node Name;Scratch Location;
  ```

  For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **list_exclude_files**

  Comma-separated list of files to exclude. Not required if the source is a Software Library. You can use an asterisk "*" as a wildcard.

- **isSwLib**

  Specifies whether it is an Oracle Home database or Software Library.

- **tryftp_copy**

  Try FTP to copy or not. You should set the FTP copy option to false when using EM CLI from the command line.

- **jobname**

  Name of the cloning job.

- **clusternodes**

  Current nodes in the cluster.

- **srchost**

  Name of a host that is part of the RAC cluster being extended.

- **home_name**

  Name of the home used by all the current RAC cluster nodes.

- **home_location**

  Home location used by all the current RAC cluster nodes.

- **prescripts**

  Path of the script to execute.

  Note: Double-quoted parameters can be passed using an escape (\) sequence. For example:

  ```
  prescripts=" <some value here>=\"some value here\" "
  ```

- **run_prescripts_as_root**

  Run prescripts as `root`. By default, this option is set to false.

- **postscripts**

  Path of the script to execute.

- **run_postscripts_as_root**

  Run postscripts as `root`. By default, this option is set to false.

- **rootscripts**

  Path of the script to execute.

- **swlib_component**

  Path to the Software Library being cloned. `isSwLib` must be true in this case.

- **source_params**

  Source Oracle home info. `isSwLib` must be false in this case.

- **jobdesc**

  Description of the job. If not specified, a default description is generated automatically.

Examples

```
emcli extend_rac_home
       -input_file="dest_properties:clonedestinations"
       -list_exclude_files="*.log,*.dbf,sqlnet.ora,tnsnames.ora,listener.ora"
       -isSwLib="false"
       -tryftp_copy="false"
       -jobname="clone database home"
       -clusternodes="node1;node2"
       -prescripts="/home/joe/myScript"
       -run_prescripts_as_root="true"
```

```
-rootscripts="%oracle_home%/root.sh"
-source_params="TargetName:host.domain.com;HomeLoc:/oracle/database1;
 HomeName:OUIHome1;ScratchLoc:/tmp"
```

Passing Variables Through EM CLI

When working with variables such as `%perlbin%` or `%oracle_home%`, EM CLI passes variable values from the current local environment instead of the variables themselves. To pass variables through an EM CLI command, as might be the case when using the `-prescripts` or `-postscripts` options, you can place the EM CLI command in a batch file and replace all occurrences of `%` with `%%`.

extract_template_tests

Extracts variables and test definitions from a repository template into a local file.

Format

```
emcli extract_template_tests
      -templateName=<template_name>
      -templateType=<template_type>
      -output_file=<output_filename>
      [-encryption_key=<key>]

[ ] indicates that the parameter is optional
```

Parameters

- **templateName**

  Name of the template.

- **templateType**

  Type of template.

- **output_file**

  Name of the output file. If the file does not exist, it will be created; if it already exists, it will be overwritten. (This is assuming the extract operation was successful; if the operation fails, no files are created, and any existing files are left unchanged.)

- **encryption_key**

  Key to encrypt the file contents. The same key should be used to decrypt the file.

Example

This example creates a file named `my_template.xml` containing the variable values and test definitions of the Web Application template `my_template`. The file contents are encrypted using the key `my_password`.

```
emcli extract_template_tests
      -templateName=my_template -templateType=website
      -output_file=my_template.xml -encryption_key=my_password
```

Note:

- The emcli user must have operator privilege on the repository template to perform this operation.
- Beacon-related information is not exported to the file. In particular, the list of monitoring beacons, as well as any beacon-specific properties or thresholds, are not exported.
- The values of password variables are not exported.

fix_compliance_state

Removes stale associations/results related to targets that have been deleted.

Format

```
fix_compliance_state
```

Example

```
emcli fix_compliance_state
```

fmw_discovery_prechecks

Checks if the host configuration is collected or not. If it is not yet collected, it initiates a configuration collection for the hosts.

Format

```
emcli fmw_discovery_prechecks
      -hostnames=<comma separated list of host names>
```

Parameters

- hostnames

  CSV (Comma Separated Value) list of host names.

Exit Codes

0 if verb processing is successful.

A non-zero value indicates that verb processing was unsuccessful.

Example

The following example checks if the host configuration is collected:

```
emcli fmw_discovery_prechecks
      -hostnames="host1.domain.com,host2.domain.com"
```

generate_activity_report

Generates a current activity report for OMS.

Format

```
emcli generate_activity_report
```

generate_discovery_input

Fusion Middleware Plug-in Release 12.1.0.6 introduced this verb to further automate the process of adding several WebLogic Domains to the Cloud Control console. You can run this verb after automatic discovery has already discovered several WebLogic Domains.

This verb creates a discovery input file automatically based on the targets discovered from the automatic discovery operation. You can then use this discovery input file in conjunction with the discover_wls verb to further automate the process of promoting discovered domains as fully managed targets. Consequently, you do not need to manually create a discovery input file to perform domain discovery from EM CLI.

Format

```
emcli generate_discovery_input
        -out_file=<fully_qualified_path_of_output_file>
```

Parameters

- **out_file**

  Location where the output file will be generated. Verify that the OMS user has write permissions on the specified location. If you are invoking the verb from the EM CLI client, verify that you have read permissions on the specified location.

Examples

This example creates the ouput file /tmp/myFile.csv.

```
emcli generate_discovery_input -out_file=/tmp/myFile.csv
```

generate_ui_trace_report

Generates and downloads a UI Page Trace Report to identify slow rendering pages.

Command-line Format

```
emcli generate_ui_trace_report
      [-user_name="user_name"]
      [-start_time="start_time"]
      [-duration="duration"]

[ ] indicates that the parameter is optional
```

Interactive or Script Format

```
generate_ui_trace_report(
         [user_name="user_name"]
         [,start_time="start_time"]
         [,duration="duration"]
         )

[ ] indicates that the parameter is optional
```

Parameters

- **user_name**

  User name for which the pager performance trace report needs to be generated. The default is the current logged-in user.

- **start_time**

  Start time in MM:dd:yy format from where the report needs to be generated. The default is the current time - 1 hour.

- **duration**

  Duration in hh:mm format for which the report needs to be generated. The default is 1 hour.

Exit Codes

- 0 — Successful
- 100 — Only super users can generate the UI page performance report for other users.
- 101 — Cannot generate the report. No data found for the given parameters. Please change the input parameters.
- 102 — Cannot generate the report. Invalid duration. Duration should be specified in HH:MI format and should be less than 24 hours.

Examples

This example generates and downloads the UI page performance report for the past 6 hours.

```
emcli generate_ui_trace_report -duration 06:00
```

generate_masking_script

Generates a masking script for the given masking definition.

Format

```
emcli generate_masking_script
      -definition_name=masking_definition_name
      [-tablespace_name=tablespace_name]
      [-parameters=<name1:value1;name2:value2;...>]
      [-credential_name=cred_name]
      [-input_file=<parameter_tag:file_path>]
      [-generate_export=Y|N]
      [-generate_mask=Y|N]
      [-script | -format=[name:<pretty|script|csv>];
                         [column_separator:column_sep_string];
                         [row_separator:row_sep_string];

[ ] indicates that the parameter is optional
```

Parameters

- **definition_name**

  Name of the masking definition.

- tablespace_name

  Name of the masking definition.

- **parameters**

  List of name-value pairs that represent the credentials required for connecting to the database instance. The supported parameters are db_username, db_password, and db_role.

- **credential_name**

  Name of the database credential. This parameter is mandatory when the db_username and db_password parameters are not specified.

- **input_file**

  Used in conjunction with the 'parameters' option, this enables you to store parameter values, such as username and password, in a separate file. This specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific parameter values of the 'parameters' . The tag must not contain colons (:) or semi-colons (;).

  For information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **generate_export**

  Specify whether to generate a script to export masked data from the specified source database using Oracle Data Pump. Specify Y or N.

- **generate_mask**

  Specify whether to generate a script to replace sensitive data in-place with masked data on a specified (nonproduction) database. Specify Y or N.

- **script**

This is equivalent to -format='name: script'.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

Output

Success or error messages as well as the impact report (if generated).

Examples

Example 1

The following example generates a script for the masking definition named mask_hr_data::

```
emcli generate_masking_script
      -definition_name=mask_hr_data
      -parameters=db_username:system;db_password:password;db_role:NORMAL
```

Example 2

The following example generates a script for the masking definition named mask_hr_data. The database password is read from the pwd.txt file:

```
emcli generate_masking_script
      -definition_name=mask_hr_data
      -parameters=PWD_FILE
      -input_file=PWD_FILE:pwd.txt
```

Example 3

The following example reads the database credentials from the named credential DB_NC and generates a masking script for the masking definition named mask_hr_data:

```
emcli generate_masking_script
      -definition_name=mask_hr_data
      -credential_name=DB_NC
```

generate_subset

Generates a subset using the specified subset definition and target database.

Format

```
emcli generate_subset
      -subset_name=<subset_definition_name>
      -target_name=<target_name>
      -target_type=<target_type>
      [-in_place_delete]
      [-db_pref_cred_name=<DBCredsNormal | DBCredsSYSDBA>]
      [-db_cred_name=<database_credential_name>]
      [-host_cred_name=<host_credential_name>]
      [-rule_parameters=<rule_parameters>]
      [-subset_directory=<database_directory_object_name>]
      [-custom_directory_path=<custom_directory_path> ]
      [-use_external_directory]
      [-external_directory=<external_directory_object_name>]
      [-export_file_name=<export_file_name>]
      [-max_file_size=<maximum_file_size>]
      [-max_threads=<maximum_number_of_threads>]
      [-compress_dump_file=<compress_dump>]
      [-encrypt_dump_file]
      [-encryption_password=<encryption_password>
      [-confirm_encryption_password=<encryption_password_confirmation>]
      [-seed_flag]
      [-seed_password=<seed_password>]
      [-log_file_name=<log_file_name>]
      [-job_name=<job_name>]
      [-job_description=<job_description>]
      [-is_non_prod_env]

[ ] indicates that the parameter is optional
```

Parameters

- **subset_name**

  Name of the existing subset definition to generate the subset.

- **target_name**

  Database target name.

- **target_type**

  Type of target. The possible values for target type are 'oracle_database', 'rac_ database', and 'oracle_pdb'.

- **in_place_delete**

  Determines whether to generate a subset operation. The default operation is exporting data into a dump file. Set this flag to delete data from the specified target.

- **db_pref_cred_name**

  Name of preferred credentials stored in the Enterprise Manager repository. You must provide a value for either db_pref_cred_name_or db_cred_name.
  The valid values for this parameter are:

  - DBCredsNormal — Default normal credential set for an oracle_database target.
  - DBCredsSYSDBA — SYSDBA credential set for an oracle_database target.

- **db_cred_name**

  Name of existing credentials stored in the Enterprise Manager repository to connect a selected target database.

- **host_cred_name**

  Name of existing host credentials stored in the Enterprise Manager repository to access the target host. If you do not specify a value, the preferred host credentials set for this target are used to access the target host.

- **rule_parameters**

  Maps values to rule parameter names.You must specify the value for this parameter if any of the rule parameters have missing values. However, you can also override the specified values using this option.

  For example:

  ```
  -rule_parameters="department_id_param:80;order_id_param:2400"
  ```

- subset_directory

  Directory location object name to save dump and log files. For example:

  ```
  DATA_PUMP_DIR
  ```

- **custom_directory_path**

  User-specified directory location on the target host to save dump and log files. You must provide a value for either subset_directory or custom_directory_path. For example:

  ```
  /scratch/user/subset_dir
  ```

- **use_external_directory**

  Enables the external directory (clustered/shared file system or ASM) for a faster export dump. If this parameter is set, you need to provide a value for external_directory.

- **external_directory**

  External directory (clustered/shared file system or ASM) for a faster export dump. For example:

  ```
  DATA_PUMP_DIR
  ```

- **export_file_name**

  File name to save the dump file. If not specified, the default value is EXPDAT%U.DMP. You can wildcard a set of dump files using '%U' in the file name.

- **max_file_size**

  Maximum file size in MB. If not specified, the default value is 100.

- max_threads

  Maximum number of threads created for export operation. If not specified, the default value is 1.

- **compress_dump_file**

  Enables data compression during the export operation.

- encrypt_dump_file

  Enables data encryption during the export operation.

- **encryption_password**

  Password key to encrypt data during export operation. If encrypt_dump_file is set and a value for this parameter is not specified, you are prompted for the encryption password. For a secure operation, it is recommended that passwords should not be stored in the scripts, but specified instead when prompted for them.

- **confirm_encryption_password**

  The value for this parameter should be the same as encryption_password. If encrypt_dump_file is set and the value for this parameter is not specified, you are prompted for confirmation of the encryption password. For a secure operation, it is recommended that passwords should not be stored in the scripts, but specified instead when prompted for the them.

- **seed_flag**

  Indicates that the subset definition contains one or more masking definitions, and any of the masking definitions contains a substitute or encrypt format.

- **seed_password**

  Seed string to be used if the subset definition contains one or more masking definitions and any of the masking definitions contains a substitute or encrypt format. If seed_flag is set and a value for this parameter is not specified, you are prompted for the seed password. The seed can be any text string. For a secure operation, it is recommended that passwords should not be stored in the scripts, but specified instead when prompted for them.

- log_file_name

  File name to save the log file. The default value is EXPDAT.LOG.

- **is_non_prod_env**

  Confirmation that the specified database is not a production database. This parameter is mandatory for the in-place delete option.

- **job_name**

  Generates the subset job name.

- **job_description**

  Job description.

Output

Success or error messages as well as the job name if applicable.

Examples

Example 1

This example exports data into a dump file.

```
emcli generate_subset
     -subset_name=hr_subset
```

```
     -target_name=sample_database
     -target_type=oracle_database
     -export_file_name=EXPDAT.DMP
     -db_cred_name=db_cred
     -host_cred_name=host_cred
     -subset_directory=DATA_PUMP_DIR
```

Example 2

This example exports data into a dump file using preferred database and host credentials. For preferred host credentials, you do not need to specify any parameters.

```
emcli generate_subset
      -subset_name=hr_subset
      -target_name=sample_database
      -target_type=oracle_database
      -export_file_name=EXPDAT.DMP
      -db_pref_cred_name=DBCredsNormal
      -subset_directory=DATA_PUMP_DIR
```

Example 3

This example exports data into a dump file at a user-specified location.

```
emcli generate_subset
      -subset_name=hr_subset
      -target_name=sample_database
      -custom_directory_path=/scratch/user/custom_location
      -target_type=oracle_database
      -export_file_name=EXPDAT.DMP
      -db_pref_cred_name=DBCredsNormal
```

Example 4

This example exports data into a dump file using rule parameters.

```
emcli generate_subset
      -subset_name=hr_subset
      -target_name=sample_database
      -target_type=oracle_database
      -export_file_name=EXPDAT.DMP
      -db_cred_name=db_cred
      -host_cred_name=host_cred
      -subset_directory=DATA_PUMP_DIR
      -rule_parameters="department_id:80;order_id:2400"
```

Example 5

This example exports data into a dump file with encryption enabled. You are prompted for encryption_password and confirm_encryption_password.

```
emcli generate_subset
      -subset_name=hr_subset
      -target_name=sample_database
      -encryption_password
      -target_type=oracle_database
      -export_file_name=EXPDAT.DMP
      -db_cred_name=db_cred
      -host_cred_name=host_cred
      -subset_directory=DATA_PUMP_DIR
```

Example 6

This example deletes data from the specified target database. To delete data from a target database, just set in_place_delete and is_non_prod_env flags as in the example.

```
emcli generate_subset
      -subset_name=hr_subset
      -target_name=sample_database
      -target_type=oracle_database
      -db_cred_name=db_cred
      -host_cred_name=host_cred
      -subset_directory=DATA_PUMP_DIR
      -in_place_delete
      -is_non_prod_env
```

generate_ui_trace_report

Generates a user interface page performance trace report, which enables you to identify slow rendering pages.

Format

Standard Mode

```
emcli generate_ui_trace_report
     [-user_name="user_name"]
     [-start_time="start_time"]
     [-duration="duration"]
```

Interactive or Script Mode

```
generate_ui_trace_report(
     [user_name="user_name"]
     [,start_time="start_time"]
     [,duration="duration"]
     )
```

```
[ ] indicates that the parameter is optional.
```

Parameters

- user_name

  User name for which the page performance trace report will be generated. Default is the currently logged in user.

- start_time

  Start time in mm:dd:yy format from where page performance trace report has to be generated. Default is current time - 1 hour.

- duration

  Duration in hh:mm format for which the page performance trace report has to be generated. Default is 1 hour.

Example

The following example generates and downloads the UI page performance trace report for the last 6 hours.

```
emcli generate_ui_trace_report -duration 06:00
```

get_add_host_status

Displays the latest status of an Add Host session.

Format

```
emcli get_add_host_status
      -session_name="Session name"
      [-details]
      [-show_only_failed_hosts]
      [-host_name="Host name"]
      [-noheader]
      [-script | -format=
                 [name:<pretty|script|csv>];
                 [column_separator:"column_sep_string"];
                 [row_separator:"row_sep_string"];
        ]

[ ] indicates that the parameter is optional.
```

Parameters

- **session_name**

  Name of the session whose status you want to view.

- **details**

  Displays additional information for the given session.

- **show_only_failed_hosts**

  Displays only the hosts on which the Add Host operation failed.

- **host_name**

  Displays the details of the provided host.

- **noheader**

  Display tabular output without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Host, Platform Name, Initialization, Remote Prerequisite, Agent Deployment, Error

Examples

Example 1

This example displays the add host status for the session 'ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_AM_PST'

```
emcli get_add_host_status -session_name=ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_AM_PST
```

Example 2

This example displays the add host status for the session 'ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_AM_PST', with additional information.

```
emcli get_add_host_status -session_name=ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_AM_PST
-details
```

Example 3

This example displays the detailed status of host 'example.com' for the session 'ADD_HOST_SYSMAN_Jun_6_2013_11:26:43_PM_PDT'.

```
emcli get_add_host_status -session_name=ADD_HOST_SYSMAN_Jun_6_2013_11:26:43_PM_PDT
-host_name=example.com
```

Example 4

This example displays only the failed hosts for the session 'ADD_HOST_SYSMAN_Jun_6_2013_11:26:43_PM_PDT'.

```
emcli get_add_host_status -session_name=ADD_HOST_SYSMAN_Jun_6_2013_11:26:43_PM_PDT
-show_only_failed_hosts
```

get_agentimage

Gets the Management Agent image for the particular platform and version provided as inputs.

Format

```
emcli get_agentimage
   -destination=<download_directory>
   -platform="<platform>"
   [-version=<version>]

[ ] indicates that the parameter is optional.
```

Parameters

- **destination**

  Directory where you want to download the Management Agent software. Ensure that you have write permission on this location.

  If the destination directory is titled with two or more words separated by a space, enclose the directory name with double-quotes. For instance, if the destination directory is titled /tmp/linuxagentimage, enter the value as -destination="/tmp/linuxagentimage"

- **platform**

  Platform for which you want to download the software; this must match one of the platforms for which the software is available on the OMS host. Use the emcli get_supported_platforms command to determine this.

- **version**

  Version of the Management Agent software that you want to download. If you do not specify this , the version defaults to the OMS version.

Examples

```
emcli get_agentimage -destination=/tmp/agtImage -platform=Linux x86
-version=12.1.0.1.0
```

get_agentimage_rpm

Gets the Management Agent image for the Linux platform and version provided as inputs, then converts the image as rpm.

Format

```
emcli get_agentimage_rpm
   -destination=<download_directory>
   -platform=<platform>
   [-version=<version>]

[ ] indicates that the parameter is optional.
```

Parameters

- **destination**

  Directory where you want to download the .rpm file. Ensure that you have write permission on this location.

  If the destination directory is titled with two or more words separated by a space, enclose the directory name with double-quotes. For instance, if the destination directory is titled /tmp/linuxagentimage, enter the value as -destination="/tmp/linuxagentimage"

- **platform**

  Platform for which you want to download the .rpm file; this must match one of the platforms for which the software is available on the OMS host. Use the emcli get_supported_platforms command to determine this.

- **version**

  Version of the Management Agent for which you want to download the .rpm file. If you do not specify this , the version defaults to the OMS version.

Examples

```
emcli get_agentimage_rpm -destination=/tmp -platform=Linux x86 -version=12.1.0.1.0
```

get_agent_properties

Displays Management Agent properties. You can use this command if you have view privilege for the Management Agent.

Format

```
emcli get_agent_properties
      -agent_name="<agent_target_name>"
      [-all]
      [-format="<format_name>"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **all**

  Shows all Management Agent properties. By default, only basic properties appear.

- **format**

  Format to display Management Agent properties. Valid values are pretty, script, and csv. By default, values are displayed in pretty format.

Examples

This example shows all of the Management Agent properties in CSV format:

```
emcli get_agent_properties -agent_name=agent.example.com:11850
        -all
        -format=csv
```

get_agent_property

Displays the value of a specific Management Agent property. You can use this command if you have view privilege for the Management Agent.

Format

```
emcli get_agent_property
      -agent_name=<agent_target_name>
      -name=<agent_property_name>
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **name**

  Name of the Management Agent property.

Examples

This example shows the current value of the UploadInterval property in emd.properties.

```
emcli get_agent_property -agent_name=agent.example.com:11850
        -name=UploadInterval
```

get_agent_upgrade_status

Shows Agent upgrade results.

Format

```
emcli get_agent_upgrade_status
      [-agent]
      [-job_name]
      [-status]

[ ] indicates that the parameter is optional
```

Parameters

- **agent**

  Shows the upgrade job details of the specified Agent names or Agent name patterns separated by commas.

- **job_name**

  Shows the upgrade job details of the specified job name.

- **status**

  Shows the upgrade job details with the specified status.

Permutations for combinations of parameters are as follows:

No parameters — Shows <JOB NAME, JOB STATUS, NUMBER OF AGENTS IN THE JOB, JOB START TIME, JOB END TIME> for each job.

-job_name only — Shows <AGENT_NAME, UPGRADE STATUS OF AGENT, UPGRADE START TIME, UPGRADE END TIME> for each Agent in the job, where job name is passed in the -job_name parameter.

-agent only — Shows <JOB NAME, UPGRADE STATUS OF AGENT IN THE JOB, UPGRADE START TIME, UPGRADE END TIME> for each job where the Agent is present and the Agent name passed in the -agent parameter.

-agent and -status only — Shows <JOB NAME, UPGRADE START TIME, UPGRADE END TIME> for each job in which the Agent and Agent upgrade status are passed in -agent and -status, respectively.

-job_name and -agent only — Shows <JOB STEP NAME, JOB STEP STATUS, JOB STEP START TIME, JOB STEP END TIME> for each step in the job for the Agent passed in the -job_name and -agent parameters.

-job_name and -status only — Shows <AGENT_NAME, UPGRADE START TIME, UPGRADE END TIME> for each Agent in the job in which the Agent upgrade status is passed in -job_name and -status, respectively

-job_name, -agent, and -status — Shows <JOB STEP NAME, JOB STEP START TIME, JOB STEP END TIME> for each step in the job for the Agent in which the step status is passed in -job_name , -agent , and -status, respectively

-status only — Shows <JOB NAME, NUMBER OF AGENTS IN THE JOB, JOB START TIME, JOB END TIME> for each job in which job status is passed in the -status parameter.

Examples

Example 1

This example shows the Agent upgrade job details for the Agent xyz.domain.com:1243 .

```
emcli get_agent_upgrade_status -agent="xyz.domain.com:1243"
```

Example 2

This example shows the Agent upgrade job details for the job UPGRADE_JOB123 .

```
emcli get_agent_upgrade_status  -job_name="UPGRADE_JOB123"
```

Example 3

This example shows the Agent upgrade job details with the status Inprogress.

```
emcli get_agent_upgrade_status -status="Inprogress"
```

get_aggregate_service_info

Gets time zone and availability evaluation function information of an aggregate's service instance.

Format

```
emcli get_aggregate_service_info
      -name=<name>
      -type=<type>
      [-noheader]
      [-script|-format=
               [name:<pretty|script|csv>];
               [column_separator:<sep_string>];
               [row_separator:<row_sep_string>]
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Aggregate service name.

- **type**

  Aggregate service type.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli get_aggregate_service_info -name=My_Name
     -type=aggregate_service
```

get_aggregate_service_members

Gets sub-services of an aggregate service instance.

Format

```
emcli get_aggregate_service_members
      -name=<name>
      -type=<type>
      [-noheader]
      [-script|-format=
                     [name:<pretty|script|csv>];
                     [column_separator:<sep_string>];
                     [row_separator:<row_sep_string>]
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Aggregate service name.

- **type**

  Aggregate service type.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli get_aggregate_service_members -name=My_Name
      -type=aggregate_service
```

get_blackout_details

Gets detailed information for a specified blackout.

Format

```
emcli get_blackout_details
      -name=<name>
      [-createdby=<blackout_creator>]
      [-noheader]
      [-script | -format=
                 [name:<pretty|script|csv>];
                 [column_separator:<column_sep_string>];
                 [row_separator:<row_sep_string>];
      ]
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the blackout.

- **createdby**

  Enterprise Manager user who created the blackout. The default is the current user. For displaying details of a blackout created using emctl, use -createdby="<SYSTEM>".

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

Output Columns

Status, Status ID, Run Jobs, Next Start, Duration, Reason, Frequency, Repeat, Days, Months, Start Time, End Time, TZ Region, TZ Offset

Examples

Example 1

This example shows detailed information for blackout `blackout1` that the current user created.

```
emcli get_blackout_details -name=blackout1
```

Example 2

This example shows detailed information for blackout `blackout1` that user `joe` created.

```
emcli get_blackout_details -name=blackout1 -createdby=joe
```

get_blackout_reasons

Lists all blackout reasons, one per line.

Format

```
emcli get_blackout_reasons
```

Examples

This example lists all blackout reasons, one per line.

```
emcli get_blackout_reasons
```

get_blackout_targets

Lists targets for a specified blackout.

Format

```
emcli get_blackout_targets
      -name=<name>
      [-createdby=<blackout_creator>]
      [-noheader]
      [-script | -format=
                 [name:<pretty|script|csv>];
                 [column_separator:<column_sep_string>];
                 [row_separator:<row_sep_string>];
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the blackout.

- **createdby**

  Enterprise Manager user who created the blackout. The default is the current user. For listing details of a blackout created using emctl, use -createdby="<SYSTEM>".

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

Output Columns

Target Name, Target Type, Status, Status ID

Examples

Example 1

This example lists targets in the blackout `blackout1` the current user created.

```
emcli get_blackout_targets -name=blackout1
```

Example 2

This example lists targets in the blackout `blackout1` that user `joe` created.

```
emcli get_blackout_targets -name=blackout1 -createdby=joe
```

get_blackouts

Lists all blackouts or just those for a specified target or one or more hosts. Only the blackouts the user has privilege to view are listed.

Format

```
emcli get_blackouts
      [-target=<name1:type1> | -hostnames=<host1;host2;...>]
      [-noheader]
      [-script | -format=
                        [name:<pretty|script|csv>];
                        [column_separator:<column_sep_string>];
                        [row_separator:<row_sep_string>];
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **target**

  Lists blackouts for this target. When neither this nor the `-hostnames` option is specified, all blackouts the user has privilege to view are listed.

- **hostnames**

  Lists blackouts that have a target on one of the specified hosts. The host name is just the target name part of the host target. For example, specify `host.example.com`, rather than `host.example.com:host`. When neither this nor the `-target` option is specified, all blackouts the user has privilege to view are listed.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

Output Columns

Name, Created By, Status, Status ID, Next Start, Duration, Reason, Frequency, Repeat, Start Time, End Time, Previous End, TZ Region, TZ Offset

Examples

Example 1

This example shows all blackouts with some details.

```
emcli get_blackouts
```

Example 2

This example shows all blackouts that cover the target `database2:oracle_database`.

```
emcli get_blackouts -target=database2:oracle_database
```

Example 3

This example shows all blackouts that cover some target on host `myhost.example.com`.

```
emcli get_blackouts -hostnames=myhost.example.com
```

Example 4

This example shows all blackouts that cover some target on host `myhost.example.com` or on host `yourhost.example.com`.

```
emcli get_blackouts -hostnames=myhost.example.com
-hostnames=yourhost.example.com
```

get_ca_info

Displays information about all of the Certificate Authorities (CA) created since the Cloud Control installation. It also displays the Management Agent names whose certificates are issued by the CA(s) when you specify the -details option. The following information is retrieved from the Cloud Control repository:

- Unique identifier of the Certificate Authority (CA) in the Cloud Control repository
- CA description
- CA creation date
- CA expiration date
- Number of Management Agents registered to this CA
- Number of secured Management Agents not registered to any CA

Format

```
emcli get_ca_info
        [-ca_id=<id1;id2;...>]
        [-details]

[ ] indicates that the parameter is optional
```

Parameters

- **ca_id**

  Specifies the Certificate Authority ID.

- **details**

  For each Certificate Authority, displays the list of Management Agent names whose certificates are issued by it.

Examples

This example shows output for the CA with the ID of 2 specified.

```
emcli get_ca_info -ca_id=2

 Info about CA with ID: 2
 CA is configured
 DN: EMAILADDRESS=Enterprise.Manager@myomshost.mycompany.com,
 CN=myomshost.mycompany.com, OU=EnterpriseManager on myomshost.mycompany.com,
 O=EnterpriseManager on myomshost.mycompany.com, L=EnterpriseManager on
 myomshost.mycompany.com1, ST=CA, C=US, DC=com
 Serial# : 87539237298512593900
 Valid From: Mon Oct 25 17:01:15 UTC 2011
 Valid Till: Thu Oct 22 17:01:12 UTC 2020
 Number of Agents registered with CA ID 2 is 1

 Number of Agents to be re-secured, as OMS is secured using force_newca
    : 1
```

Regarding the force_newca option in the last line, the output shows that a new certificate was created with the ID of 2. Two Management Agents have been re-secured to be registered with this new certificate. The OMS running on myomshost.mycompany.com has been re-secured to be registered with the new

certificate created. There is still a Management Agent that needs to be secured to be registered to the new certificate. To retrieve the Management Agent name, you need to run the command "emcli get_ca_info -ca_id=2 -details," which is shown in the next example.

This example displays the Management Agent names registered with the CA(s) for ID 2.

```
emcli get_ca_info -ca_id=2 -details

     Info about CA with ID: 2
     CA is configured
     DN: EMAILADDRESS=Enterprise.Manager@myomshost.mycompany.com,
      CN=myomshost.mycompany.com, OU=EnterpriseManager on myomshost.mycompany.com,
       O=EnterpriseManager on myomshost.mycompany.com, L=EnterpriseManager on
       myomshost.mycompany.com2, ST=CA, C=US, DC=com
     Serial# : 87539237298512593900
     Valid From: Mon Oct 25 17:01:15 UTC 2011
     Valid Till: Thu Oct 22 17:01:12 UTC 2020
     Number of Agents registered with CA ID 2 is 1
     usagent1.mycompany.com:20872

    Following Agents needs to be re-secured, as OMS is secured using force_newca
     :

     ukagent1.mycompany.com:1830
```

get_cloud_service_instances

Retrieves the list of cloud service instances. All instances are printed if you do not specify any options.

Format

```
emcli get_cloud_service_instances
      [-user="username"]
      [-family="family"]
      [-type="service type"]

[ ] indicates that the parameter is optional
```

Parameters

- **user**

  Identifies the name of the user to be used for filtering service instances.

- **family**

  Identifies the name of the service family to be used for filtering service instances.

- **type**

  Identifies the type of service to be used for filtering service instances.

Examples

Example 1

This example shows all cloud service instances:

```
emcli get_cloud_service_instances
```

Example 2

This example shows all cloud instances owned by a specified user (`user1`):

```
emcli get_cloud_service_instances -user="user1"
```

Example 3

This example shows all cloud instances that belong to a specified service family (`family1`):

```
emcli get_cloud_service_instances -family="family1"
```

Example 4

This example shows all cloud instances that belong to a specified service type (`type1`):

```
emcli get_cloud_service_instances -type="type1"
```

get_cloud_service_requests

Retrieves a list of cloud service requests. All requests are printed if you do not provide any options. Options cannot be used simultaneously.

Format

```
emcli get_cloud_service_requests
      [-user="username"]
      [-family="family"]
      [-ids="id1;id2..."]

[ ] indicates that the parameter is optional
```

Parameters

- **user**

  Identifies the name of the user to be used for filtering service instances.

- **family**

  Identifies the name of the service family to be used for filtering service instances.

- **ids**

  Lists the Request IDs to be used for filtering cloud requests. Separate each ID with a semicolon (;).

Examples

Example 1

This example shows all cloud service requests:

```
emcli get_cloud_service_requests
```

Example 2

This example shows all cloud service requests created by a specified user (`user1`):

```
emcli get_cloud_service_requests -user="user1"
```

Example 3

This example shows all cloud service requests that belong to a specified service family (`family1`):

```
emcli get_cloud_service_requests -family="family1"
```

Example 4

This example shows all cloud service requests with a specific request ID (`1` and `2`):

```
emcli get_cloud_service_requests -ids="1;2"
```

get_cloud_user_objects

Retrieves a list of cloud user objects, cloud service instances, and cloud service requests. All objects are printed if you do not provide the -user option.

Format

```
emcli get_cloud_user_objects
      [-user="username"]

[ ] indicates that the parameter is optional
```

Parameters

- **user**

  Identifies the name of the user to be used for filtering user objects.

Examples

Example 1

This example shows all cloud user objects, cloud service instances, cloud service requests, and any other objects:

```
emcli get_cloud_user_objects
```

Example 2

This example shows all cloud user objects, cloud service instances, cloud service requests, and any other objects for a specified user (`user1`):

```
emcli get_cloud_user_objects -user="user1"
```

get_config_searches

Displays information about saved configuration searches.

Format

```
emcli get_config_searches
      [-target_type="<name_or_pattern>"]
      [-format="name:<format_option>"]
      [-noheader]

[ ] indicates that the parameter is optional.
```

Parameters

- target_type

  A string matching the target type on which the configuration search is based. Use the internal target type name. Specify the full name or a pattern match using "%" as a wildcard.

- format

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.
  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings may be specified to change these defaults.
  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

    format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

- noheader

  Display tabular output without column headers.

Example

The following example shows all the configuration searches saved where the target type names contain the string "data".

```
emcli get_config_searches
      -target_type="%data%"
```

get_config_templates

Gets all of the comparison templates.

Format

```
emcli get_config_templates
      [-target_type="oracle_database"]
      [-template_name="host_template"]
      [-owner="SYSMAN"]
      [-list_default_templates="yes"]
      [-list_oracle_provided_templates="no"]
      [-format="[name:<pretty|script|csv>];
            [column_separator:"column_sep_string"];
            [row_separator:"row_sep_string"]"]
      [-noheader]

[ ] indicates that the parameter is optional
```

Parameters

- target_type

  Target type on which the comparison template is created. The value should be the internal name. To get the internal name, execute the following EM CLI command:

  ```
  emcli get_target_types
  ```

- template_name

  Name of the template, which can be a full value or a pattern match using "%". The value should be an internal name.

- owner

  Owner of the comparison template, which can be a full value or a pattern match using "%".

- list_default_templates

  Valid inputs are "yes" and "no". If the value of this option is "yes", the result will contain default templates. If the value of this option is "no", the result will not contain default templates. If this option is not specified, the result shows all templates.

- list_oracle_provided_templates

  Valid inputs are "yes" and "no". If this option is provided, the result will be only templates provided by Oracle. If the value of this option is "yes", the result contains Oracle-provided templates. If the value of this option is "no", the result will not contain Oracle-provided templates. If this option is not specified, the result shows all templates.

- format

  Format specification (default is -format="name:pretty").

  – format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  – format="name:script" sets the default column separator to a tab, and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

- noheader

  Displays tabular output without column headers.

Output columns:

Template ID
Template Name
Target Type
Default — Displays "Yes" if the template is the default, "No" otherwise
Oracle Provided — Displays "Yes" if the template is provided by Oracle, "No" otherwise
Owner
Saved Time
Time Zone
Description

Examples

Example 1

This example shows all of the comparison templates created on target type "Test Database".

```
emcli get_config_templates -target_type="oracle_database"
```

Example 2

This example shows all of the comparison templates created on target type "Test Database" and having the name "Test Database Template".

```
emcli get_config_templates -target_type="oracle_database" -template_name="Test
Database Template"
```

Example 3

This example shows all of the comparison templates created by the user name "Test Admin" that are created on target type "Test Database" and having the template_name as "Test Database Template".

```
emcli get_config_templates -target_type="oracle_database" -template_name="Test
Database Template" -owner="Test Admin"
```

Example 4

This example shows all of the comparison templates having the name that contains the string "Test".

```
emcli get_config_templates -template_name="%Test%"
```

Example 5

This example shows all of the comparison templates created by users whose user name contains the string "Admin".

```
emcli get_config_templates -owner="%Admin%"
```

Example 6

This example shows all of the default comparison templates.

```
emcli get_config_templates -list_default_templates="yes"
```

Example 7

This example shows all of the comparison templates provided by Oracle.

```
emcli get_config_templates -list_oracle_provided_templates="yes"
```

get_connection_mode

Gets the My Oracle Supporrt (MOS) connection mode. The two MOS connection modes are online and offline.

Format

```
emcli get_connection_mode
```

Parameters

None.

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

get_credtype_metadata

Prints credential-type information for a credential type. The verb prints credential column names. These column names should be used as parameter names for the create_named_credential and modify_named_credential verbs.

Format

```
emcli get_credtype_metadata
        -auth_target_type=<ttype>
        -cred_type=<name>
```

Parameters

- **auth_target_type**

  Authenticating target type.

- **cred_type**

  Credential type.

Examples

```
emcli get_credtype_metadata
          -auth_target_type=host
          -cred_type=HostCreds
```

get_dbaas_quota

Lists the database quota setup for SSA user roles.

Format

```
emcli get_dbaas_quota
```

Example

The following example successfully retrieves quotas for roles:

```
emcli get_dbaas_quota
```

It displays information similar to the following:

```
ROLE_NAME: SSA_USER_ROLE1 NUMBER_OF_SCHEMA_SERVICES: 99
MEMORY: 99
STORAGE: 99
NUMBER_OF_PLUGGABLE_DATABASES: 99NUMBER_OF_DB_INSTANCES : 99

Quotas for Roles retrieved successfully
```

get_dbaas_request_settings

Lists the database request settings.

Format

```
emcli get_dbaas_request_settings
```

Example

The following example successfully retrieves database request settings:

```
emcli get_dbaas_request_settings
```

It displays information similar to the following:

```
Future Reservation Length  : 2 Months
 Maximum Archive Duration : 10 Weeks
 Default Retirement Period : 1 Years
Request Settings retrieved successfully.
```

get_db_sys_details_from_dbname

Retrieves the details of an Oracle Database System target from a Database Unique Name.

Format

```
emcli get_db_sys_details_from_dbname
      -db_unique_name="database unique name"
```

Parameters

- db_unique_name

  Identifies the database unique name of the database target. You can find this name on the Last Collected page of the database target, or you can query for it.

Example

The following example shows how to retrieve the details of the `company_e_commerce` database:

```
emcli get_db_sys_details_from_dbname -db_unique_name="company_e_commerce"
```

get_duplicate_credentials

Gets all the target-scoped named credentials that are the same as the given target-scoped named credential. Duplicate credentials are redundant. Named credentials can be managed better if reused. The same named credential can be reused for all of the usages.

Format

```
emcli get_duplicate_credential
        -cred_name=<cred_name>
        [-cred_owner=<cred_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **cred_name**

  Searches duplicates of this credential.

- **cred_owner**

  Owner of the credential, which defaults to the current user.

Example

This example gets all of the credentials that are the same as the named credential MyOracleCredential and credential owner Joe.

```
emcli get_duplicate_credential
        -cred_name=MyOracleCredential
        -cred_owner=Joe
```

get_executions

Gets a list of executions of a submission using a submission GUID.

Format

```
emcli get_executions
   -instance=<Instance_GUID>
```

Parameters

- **instance**

  Displays all executions of a submission.

Output Columns

ExecutionGUID, Name, Status

Examples

```
emcli get_executions instance=16B15CB29C3F9E6CE040578C96093F61
```

get_ext_dev_kit

Downloads the Extensibility Development Kit to your local system. This verb has no parameters and only downloads a kit called edk.zip to the directory where you execute the command. After extracting the contents, you can use this kit to develop extensible components (plug-ins) of Enterprise Manager.

Format

```
emcli get_ext_dev_kit
```

Parameters

None.

get_group_members

Lists the members of the specified group.

Note that targets are only listed once, even though they can be in more than one sub-group of the group.

Format

```
emcli get_group_members
      -name=<name>
      [-type=<group>]
      [-depth=#]
      [-noheader]
      [-expand_non_groups]
      [-script | -format=
                 [name:<pretty|script|csv>];
                 [column_separator:<column_sep_string>];
                 [row_separator:<row_sep_string>];
      ]
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the group.

- **type**

  Group type: group. Defaults to `group`.

- **depth**

  Lists target members in sub-groups to the depth specified. The default is 1. When the depth is set to 0, no group target members are listed, and only the group's existence is verified. When the depth is set to -1, all group and sub-group target members are listed; in this case no groups appear in the output. Note that a target is listed at most once, even though it can be a member of several sub-groups.

- **noheader**

  Displays tabular information without column headers.

- **expand_non_groups**

  Lists members of aggregates and the aggregate target. By default, only sub-group target members are listed.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

- `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
- `format=`column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
- row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Target Name, Target Type

Examples

Example 1

This example lists the databases in group `db2_group`.

```
emcli get_group_members -name=db2_group
```

Example 2

This example verifies that group `my_hosts:group` exists.

```
emcli get_group_members -name=my_hosts -depth=0
```

Example 3

This example lists the unique targets in group `my_group:group` and its sub-groups.

```
emcli get_group_members -name=my_group -depth=-1
```

Example 4

This example lists the unique targets in group my_group:group and its sub-groups/aggregates. The aggregate targets are also listed.

```
emcli get_group_members -name=my_group -depth=-1 -expand_non_groups
```

get_groups

Lists all groups.

Format

```
emcli get_groups
      [-noheader]
      [-script | -format=
                 [name:<pretty|script|csv>];
                 [column_separator:<column_sep_string>];
                 [row_separator:<row_sep_string>];
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Target Name, Target Type

Example

This example lists all groups.

```
emcli get_groups
```

get_instance_data

Downloads instance submission data.

Format

```
emcli get_instance_data
        [-instance=<instance_guid>]
        [-exec=<execution_guid>]
        [-name=<execution name>]
        [-owner=<execution owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

Output

Instance properties data.

Examples

```
emcli get_instance_data -instance=16B15CB29C3F9E6CE040578C96093F61 > data.xml
```

get_instance_status

Displays the procedure instance status identified by the GUID on the command line.

> **Tip:** See also get_instances on page 5-407 and get_job_execution_detail on page 5-409.

Format

```
emcli get_instance_status
        -instance=<instance_guid>
        [-exec=<execution_guid>]
        [-name=<execution_name>]
        [-owner=<execution_owner>]
        [-xml [-details] [-showJobOutput [-tailLength=<last_n_characters>]]]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Display the details of a procedure instance identified by the GUID number. You can find the GUID number by using the emcli get_instances command.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

- **xml**

  Shows the complete status of each of the steps in XML format.

- **details**

  Displays more details for the command output. This option also requires the -xml option.

- **showJobOutput**

  Shows the output or errors for the job execution steps. This option also requires the -xml option.

- **tailLength**

  Limits the number of characters in the job step output or error. This option also requires the -showJobOutput option.

  <Last N Characters> is a positive non-zero number until which the characters are chosen from the end of the job step output. The system sets the maximum permissible characters to dump. If you do not provide this option, the maximum permissible characters are dumped.

Output Columns

GUID, Procedure Type, Instance Name, Status

Status Values

Possible status/return values are as follows:

SUCCEEDED

FAILED

EXECUTING

COMPLETED

Examples

Example 1

This example shows procedure details in CSV format:

```
emcli get_instance_status -guid=12345678901234567890123456789012
```

Example 2

This example shows details in XML format:

```
emcli get_instance_status -guid=16B15CB29C3F9E6CE040578C96093F61 -xml -details
```

Example 3

This example shows details in XML format with complete output:

```
emcli get_instance_status -guid=16B15CB29C3F9E6CE040578C96093F61 -xml -details
-showJobOutput
```

Example 4

This example shows details in XML format with the last 1024 characters of output:

```
 emcli get_instance_status -guid=16B15CB29C3F9E6CE040578C96093F61 -xml
-showJobOutput -tailLength=1024
```

See Also

get_instances

get_job_execution_detail

get_instances

Displays a list of procedure instances.

Tip: See also get_procedure_types on page 5-435.

Format

```
emcli get_instances
      [-type=<procedure_type>]

[  ] indicates that the parameter is optional
```

Parameters

- **type**

  Displays all the procedure instances of type procedure_type.

Output Columns

Instance GUID, Execution GUID, Procedure Type, Instance Name, Status

Examples

Example 1

This example lists all procedure instances:

```
emcli get_instances
```

Example 2

This example lists all procedure instances of type 'PatchOracleSoftware':

```
emcli get_instances -type=PatchOracleSoftware
```

See Also

get_procedure_types

get_internal_metric

Gets the value of an internal metric from the specified OMS. This verb obtains metric values for any of the internal metrics returned by the list_internal_metrics verb.

Format

```
emcli get_internal_metric
        -metric_name=<metric name>
        [-script | -format=
                          [name:<pretty|script|csv>];
                          [column_separator:"column_sep_string"];
                          [row_separator:"row_sep_string"];
        [-oms_name=<specific oms name> ]
        ]

[ ] indicates that the parameter is optional.
```

Parameters

- metric_name

  The name of the internal metric whose value you want to extract from the OMS. A list of internal metrics can be obtained using the list_internal_metrics verb.

- oms_name

  The name of the target OMS. The explicit OMS name can be found in the Cloud Control console Management Services page. To navigate to this page, from the Setup menu select Manage Cloud Control and then Management Services. In the Servers area, look for the full name of the Management Service (<host name>:<port number>_Management_Service).

  Note: You only need to specify the oms_name option if you are attempting to access a specific OMS in a multi-OMS environment. If you omit the oms_name option, the get_internal_metric verb will access the OMS running the current instance of EMCLI.

Examples

Example 1

The following example extracts metric values for the metric "pbc_ WorkManagerStatistics" from the OMS named "myserver.myco.com:17999_ Management_Service".

```
emcli get_internal_metric -metric_name=pbs_WorkManagerStatistics -oms_
name=myserver.myco.com:17999_Management_Service
```

Example 2

The following example extracts metric values for the metric "pbc_ WorkManagerStatistics" from the OMS currently running EMCLI.

```
emcli get_internal_metric -metric_name=pbs_WorkManagerStatistics
```

get_job_execution_detail

Displays details of a job execution.

Format

```
 emcli get_job_execution_detail
        -execution=<"execution_id">
        [-xml [-showOutput [-tailLength=<"length">]]]

[ ] indicates that the parameter is optional
```

Parameters

- **execution**

  Specifies that the ID of the job execution (execution_id) is the job execution ID.

- **xml**

  Shows the execution details as XML.

- **showOutput**

  Shows the output of the steps inside the job execution. You can only use this option in conjunction with the -xml option.

- **tailLength**

  Limits the display of the output to the number of characters from the end of the output. (length) is in characters. You can only use this option in conjunction with the -showOutput option. If you do not specify this option, a system-generated hard limit is enforced.

Examples

Example 1

This example shows the details in CSV format:

```
emcli get_job_execution_detail -execution=12345678901234567890123456789012345678901
```

Example 2

This example shows the details in XML format:

```
emcli get_job_execution_detail -execution=12345678901234567890123456789012 -xml
```

Example 3

This example shows the details in XML format with complete output:

```
emcli get_job_execution_detail -execution=12345678901234567890123456789012 -xml
-showOutput
```

Example 4

This example shows the details in XML format with last N chars output:

```
emcli get_job_execution_detail -execution=12345678901234567890123456789012 -xml
-showOutput -tailLength=1024
```

get_jobs

Lists existing jobs.

Command-Line Format

```
emcli get_jobs
        [-name="job_name_pattern"]
        [-owner="job_owner"]
        [-job_ids="ID1;ID2;..."]
        [-targets="type1:name1;type2:name2;..."]
        [-status_ids="status1;status2;..."]
        [-noheader]
        [-script | -format=
                          [name:<pretty|script|csv>];
                          [column_separator:"column_sep_string"];
                          [row_separator:"row_sep_string"];
        ]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
get_jobs
      [(name="job_name_pattern"]
      [,owner="job_owner"]
      [,job_ids="ID1;ID2;..."]
      [,targets="type1:name1;type2:name2;..."]
      [,status_ids="status1;status2;..."]
      [,noheader=True|False]
      [,script=True|False | ,format=
                  [name:<pretty|script|csv>];
                  [column_separator:"column_sep_string"];
                  [row_separator:"row_sep_string"];
      ])

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Job name pattern to filter on.

- **owner**

  Owner of the jobs to filter on.

- **job_ids**

  Lists job IDs to use as the output filters.

- **targets**

  Lists targets (as name-type pairs) to use as the output filters.

- **status_ids**

  Lists numeric status IDs to use as the output filters.

  The numeric codes for all possible job statuses are as follows:

  – `ABORTED(Error)=3`

  - ACTION_REQUIRED_STATUS=22
  - COMPLETED(Successful)=5
  - EXECUTING(Running)=2
  - FAILED=4
  - INACTIVE=14
  - MISSING_CREDS_STATUS=21
  - QUEUED=15
  - REASSIGNED_STATUS=20
  - SCHEDULED=1
  - SKIPPED=18
  - STOPPED=8
  - STOP_PENDING=12
  - SUSPENDED_AGENT_DOWN=7
  - SUSPENDED_BLACKOUT=11
  - SUSPENDED_EVENT=10
  - SUSPENDED_LOCK=9
  - SUSPEND_PENDING=13
  - SUSPENDED_USER=6
  - TARGET_NOT_READY_STATUS =26

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Name, Type, ID, Execution ID, Scheduled, TimeZone, Completed, Status, Status ID, Owner, Target Type, Target Name

Examples

These examples show the jobs with the specified job IDs 12345678901234567890123456789012 and 09876543210987654321098765432100:

Example 1 - Command-Line

```
emcli get_jobs
      -job_ids="12345678901234567890123456789012;09876543210987654321098765432100"
```

Example 2 - Script and Interactive

```
get_jobs
    (job_ids="12345678901234567890123456789012;
       09876543210987654321098765432100")
```

These examples show all jobs run against a host target named `mainhost.example.com` that are scheduled or have completed.

Example 3 - Command-Line

```
emcli get_jobs
      -status_ids="1;5"
      -targets="mainhost.example.com:host"
```

Example 4 - Script and Interactive

```
get_jobs
     (status_ids="1;5",
      targets="mainhost.example.com:host")
```

These examples show all jobs run against an Oracle database target named `payroll` that have failed. Tabular output is generated using tabs as column separators and newlines as row separators.

Example 5 - Command-Line

```
emcli get_jobs
      -status_ids="4"
      -targets="payroll:oracle_database"
      -script
```

Example 6 - Scripting and Interactive

```
get_jobs
      (status_ids="4"
      ,targets="payroll:oracle_database"
      ,script=True)
```

These examples list all jobs whose names start with BACKUP.

Example 7 - Command-Line

```
emcli get_jobs
      -name="backup%"
```

Example 8 - Scripting and Interactive

```
emcli get_jobs
      (name="backup%")
```

These examples list all jobs owned by admin1.

Example 9 - Command-Line

```
emcli get_jobs
      -owner="admin1"
```

Example 10 - Scripting and Interactive

```
emcli get_jobs
      (owner="admin1")
```

get_job_types

Lists all the job types that can be used to create jobs, library jobs, and multi-task jobs from EM CLI.

EM CLI supports the following job types:

```
ASMSQLScript
ASSOCIATE_CS_FA
ASSOCIATE_DOMAIN_FA
AssociateClusterASM
BlockAgent
CoherenceCacheAddition
CoherenceNodesRefresh
Config Log Archive Locations
DbMachineDashboard
DiscoverPDBEntities
FusionMiddlewareProcessControl
GlassFishProcessControl
InstallKernelModuleJob
Log Rotation
OSCommand
OpatchPatchUpdate_PA
RMANScript
RefreshFromEMStore
RefreshFromMetalink
RefreshFusionInstance
SOABulkRecovery
SQLScript
ShutdownDB
StartDepartedCohNodes
StartDepartedCohStoreNodes
StartFusionInstance
StartupDB
StatspackPurge
StopFusionInstance
Upgrade Exalogic Systems
WebLogic Control
WebLogic Domain Discover
WebLogic Domain Refresh
```

Format

```
emcli get_job_types
   [-type="job_type_pattern"]
   [-target_type="target_type"]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Job type internal name pattern. Specify all or part of the job type name.

- **target**

  Target type on which the job type will run.

get_metering_data

Gets usage details.

Format

```
emcli get_metering_data
     [-start_date=<start_date_in_mmddyyyy>]
     [-end_date=<end_date_in_mmddyyyy>]
     [-charge]
     [-cost_center=<cost_center_name>]
     [-target_type=<target_type>
     [-target_name=<target_name>]

[ ] indicates that the parameter is optional
```

Parameters

- **start_date**

  Report cycle start date in mmddyyyy. If you do not specify the report cycle start date, the latest report cycle is used.

- **end_date**

  Report cycle end date in mmddyyyy. If you do not specify the report cycle end date, the latest report cycle is used.

- **charge**

  Prints charge relation information.

- **cost_center**

  Cost center name. If you do not specify the cost center name, the logged in user is used as the cost center name.

- **target_type**

  If you do not specify the target type, all targets are used. Supported target types for this release are oracle_database, oracle_vm_guest, host, and weblogic_j2eeserver. This parameter is not valid without the target_name parameter.

- **target_name**

  If you do not specify the target name, all targets of a given target type are used. This parameter is not valid without the target_type parameter.

Examples

Example 1

This example shows the latest cycle usage data for the logged in user.

```
emcli get_metering_data
```

Example 2

This example shows usage data for the cost center cost_center_internal_name for the report cycle with a starting date of 10012011.

```
emcli get_metering_data
    -start_date=10012011
```

```
    -cost_center=cost_center_internal_name
```

Example 3

This example shows charge data for the my_target Oracle Guest VM target for cost center cost_center_internal_name for a report cycle with a starting date of 10012011.

```
emcli get_metering_data
    -start_date=10012011
    -cost_center=cost_center_internal_name
    -target_type=oracle_vm_guest
    -target_name=my_target
    -charge
```

get_metrics_for_stateless_alerts

For the specified target type, lists the metrics whose alerts are stateless and thus can be manually cleared. Both the metric name and metric internal name are provided in the output of this command. To clear the stateless alerts associated with the specified metric, use the clear_stateless_alerts verb.

Format

```
emcli get_metrics_for_stateless_alerts
      -target_type=type
```

Parameters

- **target_type**

  Internal target type identifier, such as host, oracle_database, oc4j, oracle_emrep, and oracle_emd.

Examples

This example provides a list of all metrics for which stateless alerts can be manually cleared for any Oracle database (internal name for the target type is oracle_database).

```
emcli get_metrics_for_stateless_alerts -target_type=oracle_database
```

get_named_credential

Displays named credential details.

Command-Line Format

```
emcli get_named_credential
        -cred_owner=<owner>
        -cred_name=<name>
        -out=<filename>
```

Scripting and Interactive Format

```
get_named_credential
        (cred_owner=<owner>
        ,cred_name=<name>
        ,out=<filename>)
```

Parameters

- **cred_owner**

  Owner of the credential.

- **cred_name**

  Required credential name.

- **out**

  Output file name. The same file can be used as the input properties file for create_named_credential and modify_named_credential.

Examples

These examples display the details of the named credential NC1 owned by the current logged in user.

Example 1 - Command-Line

```
emcli get_named_credential
      -cred_name=NC1
```

Example 2 - Scripting and Interactive

```
get_named_credential
       (cred_name="NC1")
```

These examples display the details of the named credential NC2 owned by the Administrator CREDS_MGR.

Example 3 - Command-Line

```
emcli get_named_credential
      -cred_name=NC2
      -cred_owner=CREDS_MGR
```

Example 4 - Scripting and Interactive

```
get_named_credential
      (cred_name="NC2"
```

```
,cred_owner="CREDS_MGR")
```

get_oms_config_property

Gets the property value corresponding to the specified property name.

Format

```
emcli get_oms_config_property
        -property_name="propertyName"
        [-oms_name="omsName"]
        [-details]

[ ] indicates that the parameter is optional
```

Parameters

- **property_name**

  Name of the property whose value must be retrieved.

- **oms_name**

  Name of the mangaement server for which the property must be retrieved.

- **details**

  Specifies details about from where the property value has been derived, and also the global and default values for the property.

Examples

Example 1

This example retrieves the property value set for the property name "propName" from the management server myhost:1159_Management_Service.

```
get_oms_config_property -property_name=propName -oms_name="myhost:1159_Management_
Service"
```

Example 2

This example retrieves the property value set for the property name "propName" from all the management servers.

```
get_oms_config_property -property_name=propName
```

Example 3

This example retrieves the property value set for the property name "propName" from all the management servers with details.

```
get_oms_config_property -property_name=propName -details
```

get_oms_inventory

Displays the OMS version, plug-in details and patches applied on each home.

Format

```
emcli get_oms_inventory
      [-xml]|[-map]

[ ] indicates that the parameter is optional.
```

Parameters

- xml

  Displays the output in xml format.

- map

  Diplays the output in name:value format

get_oms_logging_property

Gets the property value corresponding to the specified logging property name.

Format

```
emcli get_oms_logging_property
        -property_name="propertyName"
        [-oms_name="omsName"]
        [-details]

[ ] indicates that the parameter is optional
```

Parameters

- **property_name**

  Name of the logging property whose value must be retrieved.

- **oms_name**

  Name of the mangaement server for which the property must be retrieved.

- **details**

  Specifies details about from where the property value has been derived, and also the global and default values for the logging property.

Examples

Example 1

This example retrieves the property value set for the property name "propName" from the management server myhost:1159_Management_Service.

```
get_oms_logging_property -property_name=propName -oms_name="myhost:1159_
Management_Service"
```

Example 2

This example retrieves the property value set for the property name "propName" from all the management servers.

```
get_oms_logging_property -property_name=propName
```

get_on _demand_metrics

Gets a list of metrics that can be immediately collected with the collect_metric EM CLI verb. From this list, identify the metric you are interested in under the Metric Name column, then use its corresponding Metric Internal name in the collect_metric verb.

Format

```
emcli get_on_demand_metrics
      -target_type=type
      -target_name=name
```

Parameters

- **target_type**

  Internal target type identifier, such as host, oracle_database, oc4j, oracle_emrep, and oracle_emd.

- **target_name**

  Name of the target.

Examples

This example shows a list of collectible metrics for the host target called hostname.example.com.

```
emcli get_on_demand_metrics -target_type=host -target_name=hostname.example.com
```

get_onetime_registration_token

Generates an agent registration token for one-time use.

Format

Standard Mode

```
emcli get_onetime_registration_token
[-validity="number of minutes"]
```

Interactive or Script Mode

```
get_onetime_registration_token([validity="number of minutes"]
)

[ ] indicates that the parameter is optional.
```

Parameters

- validity1

  Number of minutes the registration token is valid. The default validity is 15 minutes. The maximum validity allowed is 720 minutes.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Examples

Example 1

The following command creates a one time registration token with validity of 25 minutes.

```
emcli get_onetime_registration_token
      -validity=25
```

Example 2

The following command creates a one time registration token with validity of 15 minutes.

```
emcli get_onetime_registration_token
```

get_operation_plan_details

Provides detailed step-by-step information about the specified operation plan.

Format

```
emcli get_operation_plan_details
      -name="plan name"
```

Parameters

- **name**

  Name of the operation plan.

Examples

```
emcli get_operation_plan_details
      -name="BISystem1-switchover"
```

See Also

create_operation_plan

get_operation_plans

get_operation_plans

Lists all configured operation plans.

Format

```
emcli get_operation_plans
      -name=<operation plan_name>
      -operation=<operation_name>
```

Parameters

- **name**

  Name of the operation plan.

- **operation**

  Name of the operation, such as switchover, failover, start, or stop.

Output Columns

Plan Name, Operation Name, Configuration GUID

Examples

```
emcli get_operation_plans
      -name="austin-switchover"
      -operation="switchover"
```

See Also

create_operation_plan

submit_operation_plan

get_paas_zone_detail

Retrieves the PaaS Infrastructure Zone details.

Format

```
emcli get_paas_zone_detail
      -name="<Name of PaaS Zone>"
```

Parameters

- **name**

  Name of the existing PaaS Infrastructure Zone

Example

This example retrieves the PaaS Infrastructure Zone details for My PaaS Zone:

```
emcli get_paas_zone_detail
    -name="My PaaS Zone"
```

It displays the following information:

```
Name                            My PaaS Zone
Description                     This is a test PaaS Zone
Named Credentials               ZoneNamedCredentials
Number of Hosts                 2
Roles                           CLOUD_ADMIN_ROLE
Maximum Memory Allocation (%)   75
Maximum CPU Utilization (%)     85
```

Note: To retrieve the members of this PaaS Infrastructure Zone, run:

```
emcli get_system_members -name="My PaaS Zone" -type="self_service_
zone"
```

get_patch_plan_data

Gets patch plan user-editable data.

Format

```
emcli get_patch_plan_data
        -name="name"
```

Parameters

- **name**

  Name of a given patch plan.

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

Example

```
emcli get_patch_plan_data -name="plan_name"
```

get_pool_allowed_placement_constraints

Retrieves the list of placement constraints for a pool target type.

Format

```
emcli get_pool_allowed_placement_constraints
      -target_type="<Target type of Software Pool>"
```

Parameters

- **target_type**

  Target type of the software pool.

Example

The following example retrieves the list of placement constraints for the mwaas_zone target type:

```
emcli get_pool_allowed_placement_constraints
     -target_type="mwaas_zone"
```

It displays the following output:

```
Name                        Description
MAX_INSTANCES               Maximum Number of Java Servers (per host)
```

get_pool_capacity

Retrieves the capacity details for a software pool including CPU utilization, memory allocation, and number of instances per host.

Format

```
emcli get_pool_capacity
    -name="<Software Pool name>"
    -target_type="<Target type of Software Pool>"
```

Parameters

- **name**

  Name of an existing Software Pool.

- **target_type**

  Target type of the Software Pool.

get_pool_detail

Retrieves details for a software pool.

Format

```
emcli get_pool_detail
     -name="<Software Pool name>"
     -target_type="<Target type of Software Pool>"
```

Parameters

- **name**

  Name of an existing software pool.

- **target_type**

  Target type of the software pool.

Example

This example retrieves details for the MyPool software pool:

```
emcli get_pool_detail
       -name="My Pool"
      -target_type="mwaas_zone"
```

It displays the following information:

```
Name                          My Pool
Target Type                   mwaas_zone
Description                   This is a test Pool
Paas Infrastructure Zone      My PaaS Zone
Number of Members             1
Placement Constraints         MAX_INSTANCES : 25
Member Constraints            VERSION : 10.3.5.0
```

Note: To retrieve the members of this software pool, run:

```
emcli get_system_members -name="My Pool" -type="mwaas_zone"
```

---

get_pool_filtered_targets

Retrieves the filtered targets available for software pool creation based on the given criteria.

Format

```
emcli get_pool_filtered_targets
      -target_type="<Target type of Software Pool>"
      -paas_zone="<Paas Infrastructure Zone of Software Pool>"
      [-member_constraints="<constraint1=value1, constraint2=value2>"
]
```

[] indicates that the parameter is optional.

Parameters

- **target_type**

  Target type of the Software Pool.

- **paas_zone**

  Name of PaaS infrastructure zone within which the filtered targets are to be retrieved.

- **member_constraints**

  Comma separated key value pairs that restrict the addition of member targets to a software pool with a set criteria.

Example

The following example retrieves the list of allowed possible member constraints for a pool target type:

```
emcli get_pool_allowed_member_constraints -target_type=<Target type>
```

get_plugin_deployment_status

Displays the status of a specific plug-in deployment or undeployment activity as well as the list of steps.

Format

```
emcli get_plugin_deployment_status
       [-plugin_id="plugin_id"]

[ ] indicates that the parameter is optional
```

Parameters

- **plugin_id**

  ID of the plug-in for which you need to view the deployment/undeployment status. If not provided, the command shows the status of the latest plug-in being deployed, or the last one that was deployed or undeployed.

Examples

Example 1

Displays the status of the last plug-in deployment/undeployment activity.

```
emcli get_plugin_deployment_status
```

Example 2

This example displays the status of the last deployment/undeployment activity of a specific plug-in.

```
emcli get_plugin_deployment_status -plugin_id=oracle.sysman.db
```

get_procedures

Gets a list of deployment procedures and pre-saved procedure configurations.

Tip: See also get_procedure_types on page 5-435.

Format

```
emcli get_procedures [-type=<procedure_type>]
      [-parent_proc=<procedure_associate>]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Displays all the deployment procedures of type procedure_type.

- **parent_proc**

  Procedure associated with procedure configurations.

Output Columns

GUID, Procedure Type, Name, Display Type, Version, Created By, Procedure Name

See Also

get_procedure_types

get_procedure_xml

get_procedure_types

Gets the list of all deployment procedure types.

Format

```
emcli get_procedure_types
```

Output Column

Procedure Type

Example

This example lists all procedure types:

```
emcli get_procedure_types
```

get_procedure_xml

Gets the deployment procedure XML file. XML is printed on standard output.

Format

```
emcli get_procedure_xml
      -procedure=[procedure_guid]
      [-name=<procedure_name>]
      [-owner=<procedure_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **procedure**

  Procedure GUID.

- **name**

  Procedure name.

- **owner**

  Procedure owner.

Output

Deployment procedure XML.

Examples

```
emcli get_procedure_xml -procedure=16B15CB29C3F9E6CE040578C96093F61 > proc.xml
```

get_reports

Returns a list of Information Publisher reports owned by or viewable by all users or a specified user. The output of this report is space-separated, quoted strings for the report title and owner, with each report on its own line.

Format

```
emcli get_reports
   [-owner="<report_owner>"]

[ ] indicates that the parameter is optional
```

Parameters

- **owner**

  Enables listing of viewable reports that a specific Enterprise Manager owns.

Output

Space-separated quoted strings for the report title and owner, with each report on its own line.

Examples

```
emcli get_reports –owner=username
“report 1”,”username”
“example report 2”,”username”


emcli get_reports
“report A”,”username1”
“report 1”,”username2”
“example report 2”,”username2
```

get_resolution_states

Gets the list of existing resolution states used in managing incidents and problems. It also prints the display position of states. It does not list the fixed "New" and "Closed" resolution states.

Format

```
emcli get_resolution_states
```

Parameters

None.

Examples

This example shows sample output for Incident defined states of OnHold, Waiting, and Processed, and Problem defined states of OnHold and Processed.

```
Incident resolution states
   5     OnHold
  10     Waiting
  25     Processed

Problem resolution states
   5     OnHold
  25     Processed
```

get_retry_arguments

Get arguments of failed steps that can be retried.

Format

```
emcli get_retry_arguments
      [-instance=<instance_guid>]
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
      [-stateguid=<state_guid>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

- **stateguid**

  State GUID.

Examples

```
emcli get_retry_arguments -instance=16B15CB29C3F9E6CE040578C96093F61
```

```
emcli get_retry_arguments -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid=51F762417C4943DEE040578C4E087168
```

get_runtime_data

Downloads the execution run-time properties data. The execution can be retrieved by using the instance GUID, execution GUID, or a name value pair.

Format

```
emcli get_runtime_data
        [-instance={instance_guid}]
        [-exec={execution_guid}]
        [-name={execution name}]
        [-owner={execution owner}]

[ ] indicates that the parameter is optional.
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

> **Note:** The name and owner parameters must be used together.

Example

This example displays the execution run-time properties data.

```
emcli get_runtime_data -exec=16B15CB29C3F9E6CE040578C96093F61 > data.xml
```

get_saved_configs

Lists the saved configurations.

Format

```
emcli get_saved_configs
        [-target_type="<target_type>"]
        [-target_name="<target_name>"]
        [-owner="<owner>"]
        [-format=name:<pretty|script|csv>;
        [column_separator:"column_sep_string"];
        [row_separator:"row_sep_string"];]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Internal type name, such as oracle_database for "Oracle Database." You can use the get_target_types command to get the internal name for a target type.

- **target_name**

  Name of the target.

  Either specify the complete name or a pattern match using "%".

- **owner**

  Owner of the saved configuration.

  This can be a full value or a pattern match using "%".

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

- **noheader**

  Display tabular output without column headers.

Output Columns

Name (Saved configuration name, the concatenation of target name, target type and saved time in YYYYMMDDHH24MISS format), Target Type, Target Name, Saved Time (Format of the time is: yyyy/MM/dd HH:mm), Time Zone, Owner, Description

Examples

Example 1

This example lists all of the saved configurations created on target type "host":

```
emcli get_saved_configs -target_type="host"
```

Example 2

This example lists all of the saved configurations created on target type "host" and target name "test host":

```
emcli get_saved_configs -target_type="host" -target_name="test host"
```

Example 3

The example lists all of the saved configurations created by user with name "test user" and created on target type "host" and target name "test host":

```
emcli get_saved_configs -target_type="host" -target_name="test host" -owner="test
user"
```

Example 4

This example lists all of the saved configurations whose target name contains the string "Test":

```
emcli get_saved_configs -target_name="%Test%"
```

Example 5

This example lists all of the saved configurations created by users whose user name contains the string "Admin":

```
emcli get_saved_configs -owner="%Admin%"
```

get_service_template_detail

Retrieves the Service Template details.

Format

```
emcli get_service_template_detail
        -name="<Service_Template_name>"
        -service_family="<Service_family_name>"

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the existing Service Template.

- **service_family**

  Service family to which the Service Template belongs. Examples: DBAAS for Database, and MWAAS for Middleware.

Examples

```
emcli clear_problem
      -problem_key="ORA-600"
      -target_type="oracle_database"-preview
```

displays the following output:

```
Name Middleware service template August
Service Family MWAAS
Description Middleware small instance service template
Roles CLOUD_USER_ROLE_1
Software Pools mwaas_zone:pool1
Configurations
{
"type" : "CDP"
"serviceFamily" : "MWAAS",
"payloads": null,
"configurations" : [ {
"name" : "WebLogic Username *",
"value" : "weblogic",
"id" : "wlsUserName",
"displayName" : null,
"description" : "Username for the WebLogic Server",
"values" : null,
"required" : false,
"secret" : false,
"subconfigurations" : null
}, {
"name" : "WebLogic Password *",
"value" : "Welcome_123",
"id" : "wlsUserPassword",
"displayName" : null,
"description" : "Password for the WebLogic Server",
"values" : null,
"required" : false,
"secret" : true,
```

```
"subconfigurations" : null
},{
"name" : "Topology",
"value" : "1",
"id" : "topology",
"displayName" : null,
"description" : "Enter 1 for single cluster, 0 for no cluster. For physical
provisioning it is auto populated based on the profile selected. For virtual
provisioning it is defaulted to 1. Please change based on the actual topology of
the assembly. ",
"values" : null,
"required" : false,
"secret" : false,
"subconfigurations" : null
},
```

Note that all configurations are not shown in the example above.

get_service_templates

Lists the available service templates.

Format

```
emcli get_service_templates
        [-service_family="<Service_family_name>"]

[ ] indicates that the parameter is optional
```

Parameters

- **service_family**

  Service family name used for filtering the service templates. Example: DBAAS for Database, and MWAAS for Middleware

get_signoff_agents

Shows the available Agents for sign-off.

If you do not specify any options, the command shows all Agents available for sign-off. If you specify more than one option, the command shows the union of Agents available for sign-off belonging to each option passed.

Format

```
emcli get_signoff_agents
      [-agents="List_of_agents"]
      [-platforms="List_of_platforms"]
      [-versions="list_of_versions"]
      [-groups="list_of_group_names"]
      [-output_file="location_of_output_file"]

[ ] indicates that the parameter is optional
```

Parameters

- **agents**

  List of Agents for sign-off matching Agent names or Agent names pattern separated by commas.

- **platforms**

  Lists Agents available for sign-off on the specified platforms.

- **versions**

  Lists Agents available for sign-off with the specified version.

- **groups**

  Lists Agents available for sign-off belonging to the specified groups.

- **output_file**

  Adds the Agents into the output file, which can be submitted for a clean-up job to remove old Oracle Management Agent homes and old Oracle home targets, and back up directories of upgraded Oracle Management Agents.

Examples

Example 1

This example shows the list of Agents for clean up that match the Agents specified in the option.

```
emcli get_signoff_agents -agents="abc%,xyz.domain.com:1243"
```

Example 2

This example shows the list of Agents for clean up that match the platform specified in the option.

```
emcli get_signoff_agents -platforms="Linux x86,Microsoft Windows x64 (64-bit)"
```

Example 3

This example shows the list of Agents for clean up that match the versions specified in the option.

```
emcli get_signoff_agents -versions="12.1.0.1.0,12.1.0.2.0"
```

Example 4

This example shows the list of Agents for clean up that match the group names specified in the option.

```
emcli get_signoff_agents -groups="GROUP1,GROUP2"
```

Example 5

This example adds the list of Agents for clean up to the /scratch/agents_file.txt file.

```
emcli get_signoff_agents -output_file="/scratch/agents_file.txt"
```

get_signoff_status

Shows Agent sign-off results.

Format

```
emcli get_signoff_status
      [-agent="full_agent_name"]
      [-job_name="job_name"]
      [-status="status"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent**

  Shows the sign-off job details of the specified Agent names or Agent names pattern separated by commas.

- **job_name**

  Shows the sign-off job details of the specified job name.

- **status**

  Shows the sign-off job details of the specified status.

Permutations for combinations of parameters are as follows:

No parameters — Shows <JOB NAME, JOB STATUS, NUMBER OF AGENTS IN THE JOB, JOB START TIME, JOB END TIME> for each job.

-job_name — Shows <AGENT_NAME, STATUS OF JOB, START TIME, END TIME> for each Agent in the job, where tje job name is passed in the -job_name parameter.

-status only — Shows <JOB NAME, NUMBER OF AGENTS IN THE JOB, JOB START TIME, JOB END TIME> for each job, where the job status is passed in -status parameter.

-agent only — Shows <JOB NAME, STATUS OF JOB, START TIME, END TIME> for each job, where the Agent is present and the Agent name is passed in the -agent parameter.

-job_name and -agent only — Shows <JOB STEP NAME, JOB STEP STATUS, JOB STEP START TIME, JOB STEP END TIME> for each step in the job for the Agent passed in -job_name , -agent parameter

-job_name, -agent, and -status — Shows <JOB STEP NAME, JOB STEP START TIME, JOB STEP END TIME> for each step in the job for the Agent having step status passed in -job_name , -agent , and -status respectively.

-job_name and -status — Shows <AGENT_NAME, START TIME, END TIME> for each Agent in the job having an Agent upgrade status passed in -job_name and -status respectively.

-agent and -status — Shows <JOB NAME, START TIME, END TIME> for each job having the Agent and clean-up status passed in -agent and -status respectively.

Examples

Example 1

This example shows the sign-off job details for agent xyz.domain.com:1243 .

```
emcli get_signoff_status -agent=xyz.domain.com:1243
```

Example 2

This example shows the sign-off job details with the job name cleanup_123.

```
emcli get_signoff_status -job_name="cleanup_123"
```

Example 3

This example shows the sign-off job details with the status Success.

```
emcli get_signoff_status -status="Success"
```

get_siteguard_aux_hosts

Derives the list of all the auxiliary hosts associated with the system.

Format

```
emcli get_siteguard_aux_hosts
        -system_name=name_of_the_system
```

Parameter

system_name

Name of the system whose list of auxiliary hosts you want to view.

Example

This example derives the list of auxiliary hosts configured on austin-system:

```
emcli get_siteguard_aux_hosts
        -system_name="austin-system"
```

get_siteguard_credential_association

Lists the credential associations configured for a system.

Format

```
emcli get_siteguard_credential_association
      [-system_name=<name_of_system>]
      [-target_name=<name_of_target>]
      [-credential_type=<type_of_credential>]

[ ] indicates that the parameter is optional
```

Parameters

- **system_name**

  Name of the system.

- **target_name**

  Name of the target.

- **credential_type**

  Type of the credential, which can be HostNormal, HostPrivileged, WLSAdmin, or DatabaseSysdba.

Output Columns

Target Name, Credential Name, Credential Type

Examples

Example 1

```
emcli get_siteguard_credential_association
      -system_name="austin-system"
      -credential_type="HostNormal"
```

Example 2

```
emcli create_siteguard_credential_association
      -system_name="austin-system"
      -target_name="austin-database-instance"
      -credential_type="HostNormal"
```

See Also

create_siteguard_credential_association

update_siteguard_credential_association

get_siteguard_health_checks

Displays the schedule of health checks for an operation plan.

Format

```
emcli get_siteguard_health_checks
        [-operation_plan="name_of_the_operation_plan"]

[ ] indicates that the parameter is optional
```

Parameter

operation_plan

Name of the operation plan for which health checks have been scheduled.

Example

This example displays information about the health checks scheduled on a system for the `austin-switchover` operation plan:

```
emcli get_siteguard_health_checks
        -operation_plan="austin-switchover"
```

get_siteguard_lag

Retrieves and shows the configured limit for the Apply lag and Transport lag for all or selected databases of the system.

Format

```
emcli get_siteguard_lag
        [-system_name="name_of_the_system"]
        [-target_name="name_of_the_target_database"]
        [-property_name="lag_type"]

[ ] indicates that the parameter is optional
```

Parameters

- **system_name**

  Name of the system whose configuration details you want to view.

- **target_name**

  Name of the database whose lag configuration details you want to view.

- **property_name**

  Name of the lag property configured. Valid values are `ApplyLag` and `TransportLag`.

Examples

Example 1

This example displays the details of the Apply lag limit configured on all of the databases of the system `austin-system`:

```
emcli get_siteguard_lag
        -system_name="austin-system"
        -property_name="ApplyLag"
```

Example 2

This example displays the details of the Transport lag limit configured on the database `OID-db` of `austin-system`:

```
emcli get_siteguard_lag
        -system_name="austin-system"
        -target_name="OID_db"
        -property_name="TransportLag"
```

get_siteguard_script_credential_params

Retrieves all credentials parameters for a Site Guard script.

Format

```
emcli get_siteguard_script_credential_params
        -script_id="Id associated with the script"
        -credential_name="name of the credential"
        [-credential_owner="credential owner"]

[ ] indicates that the parameter is optional.
```

Parameters

- script_id

  The script ID.

- credential_name

  Name of the credential. If this parameter is not specified, all credentials associated as parameters for the script will be listed.

- credential_owner

  The owner of the credential. If this argument is not specified, all credentials associated as parameters for the script will be listed.

Examples

Example 1

The following command retrieves the Site Guard credential parameters for the script with the ID 1 and name NAMED_CREDENTIAL_X.

```
emcli get_siteguard_script_credential_params
      -script_id="1"
      -credential_name="NAMED_CREDENTIAL_X"
```

Example 2

The following command retrieves the Site Guard credential parameters for all scripts with the script ID of 3.

```
emcli get_siteguard_script_credential_params
      -script_id=3"
```

Example 3

The following command retrieves the Site Guard credential parameters for all scripts with the script ID of 3, owned by the user SG_ADMIN.

```
emcli get_siteguard_script_credential_params
      -script_id="3"
      -credential_owner="SG_ADMIN"
```

get_siteguard_script_hosts

Lists the host or hosts associated with any script where the script is designated to run.

Format

```
emcli get_siteguard_script_hosts
      [-script_id=<script_id>]

[ ] indicates that the parameter is optional
```

Parameters

- **script_id**

  ID associated with the script.

Output Columns

Host Name

Examples

```
emcli get_siteguard_script_hosts
      -script_id="10"
```

See Also

create_siteguard_script

add_siteguard_script_hosts

get_siteguard_scripts

Obtains the Site Guard scripts associated with the specified system.

Format

```
emcli get_siteguard_scripts
      -system_name=<system_name>
      -operation=<operation_name>
      [-script_type=<type_of_script>]
      [-role=<role_of_system>]
```

Parameters

- **system_name**

  Name of the system.

- **operation**

  Name of the operation, such as switchover, failover, start, or stop.

- **script_type**

  Type of the script. For example: mount, unmount, pre-script, post-script, failover, or switchover.

- **role**

  Filters the scripts based on the role associated with the system. For example: Primary or Standby.

Output Columns

Script, ID, Type, Operation, Path, Role

Examples

Example 1

```
emcli get_siteguard_scripts
      -system_name="BISystem1"
      -operation="Switchover"
      -script_type="Pre-Script"
```

Example 2

```
emcli get_siteguard_scripts
      -system_name="austin-system"
      -operation="Switchover"
      -script_type="Pre-Script"
      -role="Primary"
```

See Also

create_siteguard_script

delete_siteguard_scripts

get_supported_platforms

Lists the platforms for which the Management Agent software is available on the OMS host.

Format

```
emcli get_supported_platforms
```

Output

The output of the command appears like This example:

```
--------------------------
Platform Name : Linux x86
--------------------------
```

get_supported_privileges

Gets the list of available privileges in Enterprise Manager based on the type specified.

Format

```
emcli get_supported_privileges
        -type="ResourceType"
        [-noheader]
        [-script | -format=
                          [name:<pretty|script|csv>];
                          [column_separator:"column_sep_string"];
                          [row_separator:"row_sep_string"];
        ]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Type of privileges to retrieve from Enterprise Manager. Possible values are:

  – ALL (default value)

  – SYSTEM

  – TARGET

  – JOB

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format=`column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Privilege Name, Privilege Type, Resource Class, Resource GUID Column, Resource ID Columns

get_system_members

Lists the members of the specified system.

Format

```
emcli get_system_members
      -name="name"
      [-type=<generic_system>]
      [-depth=#  (default 1)]
      [-noheader]
      [-script | -format=
            [name:<pretty|script|csv>];
            [column_separator:"column_sep_string"];
            [row_separator:"row_sep_string"];
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the system.

- **type**

  System type: `generic_system`. Defaults to `generic_system`.

- **depth**

  Lists target members in sub-systems to the specified depth. When the depth is set to 0, no system target members are listed, and only the system's existence is verified. When the depth is set to -1, all system and sub-system target members are listed.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`. In interactive and script mode, the value must be True or False.

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Source Target Name, Member Target Name, Member Target Type, Level

Examples

Example 1

This example lists the databases in system `db2_system`.

```
emcli get_system_members -name=db2_system
```

Example 2

This example verifies that system `my_system:generic_system` exists.

```
emcli get_system_members -name=my_system -depth=0
```

Example 3

This example lists the unique targets in `system my_system:generic_system` and its sub-systems.

```
emcli get_system_members -name=my_system -depth=-1
```

get_target_properties

Lists all the property names for the target type provided.

Format

```
emcli get_target_properties
         -target_type="target_type"
```

Parameters

- **target_type**

  Target type for which you want to list user-defined property names.

Examples

```
emcli get_target_properties -target_type="host"

Comment
Contact
Deployment Type
Line of Business
Location
Target properties fetched successfully
```

get_target_types

Obtain target types and their details for the input plug-in.

Format

```
emcli get_target_types
      -plugin="Plug-in Id for which the targets types needs to be retrieved"
```

Output columns:

Display Target Type, Target Type Is Composite (Y/N)

Parameters

- plugin

  Plug-in ID for which the target types needs to be retrieved.

Example

The following example shows all target types for the database plug-in:f

```
emcli get_target_types
      -plugin=oracle.sysman.db
```

get_targets

Gets status and alert information for targets.

Command-Line Format

```
emcli get_targets
        [-targets="[name1:]type1;[name2:]type2;..."]
        [-alerts]
        [-noheader]
        [-script | -format=
                          [name:<pretty|script|csv>];
                          [column_separator:"column_sep_string"];
                          [row_separator:"row_sep_string"];
        ]
        [-limit_rows="maximum_targets_to_be_retrieved"]
        [-config_search="configuration_search_UI_name"]
        [-unmanaged]
        [-properties]
        [-separator_properties="properties_sep_string"]
        [-subseparator_properties="properties_subsep_string"]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
get_targets
      [(targets="[name1:]type1;[name2:]type2;..."]
      [,alerts=True|False]
      [,noheader=True|False]
      [,script=True|False | ,format=
                [name:<pretty|script|csv>];
                [column_separator:"column_sep_string"];
                [row_separator:"row_sep_string"];
      ]
      [,-limit_rows="maximum_targets_to_be_retrieved"]
      [,-config_search="configuration_search_UI_name"]
      [,-unmanaged]
      [,-properties]
      [,-separator_properties="properties_sep_string"]
      [,-subseparator_properties="properties_subsep_string"])

[ ] indicates that the parameter is optional
```

Parameters

- **targets=name:type**

  Name or type can be either a full value or a pattern match using `%`. Also, name is optional, so the type can be specified alone.

- **alerts**

  Shows the count of critical and warning alerts for each target. In scripting and interactive mode, the value needs to be set to either True or False.

- **noheader**

  Display tabular output without column headers. In scripting and interactive mode, the value needs to be set to either True or False.

- **script**

  This is equivalent to `-format="name:script"`. In scripting and interactive mode, the value needs to be set to either True or False.

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format=`column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

- **limit_rows**

  Maximum number of targets to be retrieved. This defaults to 2000 rows if not specified.

- **config_search**

  The search UI name should be the display name of the configuration search.

- **unmanaged**

  Gets unmanaged targets (no status or alert information).

- properties

  Maximum number of targets to be retrieved. This defaults to 2000 rows if not specified.

- **separator_properties**

  Displays unmanaged target properties with separator_properties.

- **subseparator_properties**

  Displays unmanaged target properties with subseparator_properties.

Output Columns

Status ID, Status, Target Type, Target Name, Critical, Warning

Examples

These examples show all targets. Critical and Warning columns are not included.

Example 1 - Comand-Line

```
emcli get_targets
```

Example 2 - Scripting and Interactive

```
get_targets()
```

These examples show all targets. Critical and Warning columns are shown.

Example 3 - Command-Line

```
emcli get_targets
      -alerts
```

Example 4 - Scripting and Interactive

```
get_targets
      (alerts=True)
```

These examples show all `oracle_database` targets.

Example 5 - Command-Line

```
emcli get_targets
      -targets="oracle_database"
```

Example 6 - Scripting and Interactive

```
get_targets
      (targets="oracle_database")
```

These examples show all targets whose type contains the string `oracle`.

Example 7 - Command-Line

```
emcli get_targets
      -targets="%oracle%"
```

Example 8 - Scripting and Interactive

```
get_targets
      (targets="%oracle%")
```

These examples show all targets whose name starts with `databa` and type contains `oracle`.

Example 9 - Command-Line

```
emcli get_targets
      -targets="databa%:%oracle%"
```

Example 10 - Scripting and Interactive

```
get_targets
      (targets="databa%:%oracle%")
```

These examples show status and alert information on the Oracle database named `database3`.

Example 11 - Command-Line

```
emcli get_targets
      -targets="database3:oracle_database"
      -alerts
```

Example 12 - Scripting and Interactive

```
get_targets
      (targets="database3:oracle_database"
      ,alerts=True)
```

These examples show name and type information for unmanaged host targets.

Example 13 - Command-Line

```
emcli get_targets
      -targets="host"
      -unmanaged
```

Example 14 - Scripting and Interactive

```
get_targets
    (targets="host"
    ,unmanaged)
```

These examples show name, type, and properties for unmanaged host targets with the specified separators. By default, the separator_properties is ";" and the subseparator_ properties is ":".

Example 15 - Command-Line

```
emcli get_targets
      -unmanaged -properties
      -separator_properties=,
      -subseparator_properties==
```

Example 16 - Scripting and Interactive

```
get_targets
    (unmanaged -properties
    ,separator_properties=,
    ,subseparator_properties==)
```

get_test_thresholds

Shows test thresholds.

Format

```
emcli get_test_thresholds
     -name=<target_name>
     -type=<target_type>
     -testname=<test_name>
     -testtype=<test_type>
     [-script|-format=
              [name:"pretty|script|csv"];
              [column_separator:"sep_string"];
              [row_separator:"row_sep_string"]
     ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name.

- **type**

  Target type.

- **testname**

  Test name.

- **testtype**

  Test type.

- **script**

  This is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format=`column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli get_test_thresholds -name="Service Name"
                -type="generic_service"
                -testname="Test Name"
                -testtype="HTTP"
```

get_threshold

Obtains threshold information for a given target and metric.

Format

```
emcli get_threshold
      -target_name="tname"
      -target_type="ttype"
      [-metric="metric_group"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Name of the target associated with the threshold.

- **target_type**

  Type of target associated with the threshold.

- **metric**

  Metric group associated with the threshold. The default without this option is to show the threshold of all metrics.

Examples

Example 1

This example gets the threshold data for the Load category on the host myhost.example.com.

```
emcli get_threshold
          -target_name="myhost.example.com"
          -target_type="host"
          -metric="Load"
```

Example 2

This example gets the DiskActivitybusy threshold for the Disk Activity on the host myhost.oracle.com.

```
emcli get_threshold
          -target_name="myhost.oracle.com"
          -target_type="host"
          -metric="DiskActivity"
```

get_unsync_alerts

Gets a list of alerts that are out-of-sync between the Management Agent and the repository for the specified target. You would typically use this command when you think that the Management Agent has not uploaded the latest alert to the repository. Under these circumstances, the repository would be out-of-sync with the Management Agent state.

Format

```
emcli get_unsync_alerts
      -target_type="type"
      -target_name="name"
```

Parameters

- **target_type**

  Internal target type identifier, such as host, oracle_database, emrep, and so forth.

- **target_name**

  Name of the target.

Output Column

Status

Examples

This example shows the out-of-sync alert states for the host target type and abc.example.com target name:

```
emcli get_unsync_alerts -target_type=host -target_name=abc.example.com
```

get_unused_metric_extensions

Gets a list of metric extensions deployed to Agents, but not attached to any targets.

Format

```
emcli get_unused_metric_extensions
```

Parameters

None.

get_update_status

Gets the latest status of an update.

Format

```
emcli get_update_status
        -id="internal id"
```

Parameters

- **id**

  Internal identification for the update.

Examples

This example displays the latest update status.

```
emcli get_update_status
        -id="914E3E0F9DB98DECE040E80A2C5233EB"
```

get_upgradable_agents

Shows upgradable Agents. If you do not specify any options, the command shows all upgradable Agents. If you specify more than one option, the command shows the union of upgradable Agents belonging to each option specified.

Format

```
emcli get_upgradable_agents
      [-agents="full_agent_name"]
      [-platforms="list_of_platforms"]
      [-versions="list_of_versions"]
      [-groups="list_of_group_names"]
      [-output_file="output_file_location"]

[ ] indicates that the parameter is optional
```

Parameters

- **agents**

  Lists upgradable Agents matching Agent names or an Agent names pattern.

- **platforms**

  Lists upgradable Agents on the specified platforms.

- **versions**

  Lists upgradable Agents with the specified version.

- **groups**

  Lists upgradable Agents belonging to the specified groups.

- **output_file**

  Lists upgradable Agents and adds them to the specified file.

Examples

Example 1

This example lists upgradable Agents matching the pattern abc% and xyz.domain.com agent.

```
emcli get_upgradable_agents -agents="abc%,xyz.domain.com:1243"
```

Example 2

This example lists upgradable Agents on the platforms Linux x86 and Microsoft Windows x64 (64-bit).

```
emcli get_upgradable_agents -platforms="Linux x86,Microsoft Windows x64 (64-bit)"
```

Example 3

This example lists upgradable Agents with version 12.1.0.1.0 and 12.1.0.2.0

```
emcli get_upgradable_agents -versions="12.1.0.1.0,12.1.0.2.0"
```

Example 4

This example lists upgradable Agents belonging to groups GROUP1 and GRP2.

```
emcli get_upgradable_agents -groups="GROUP1,GRP2"
```

Example 5

This example lists upgradable Agents and adds them to the file /scratch/agents_file.txt.

```
emcli get_upgradable_agents -output_file="/scratch/agents_file.txt"
```

grant_bipublisher_roles

Grants roles for accessing the BI Publisher catalog.

Format

```
emcli grant_bipublisher_roles
    (-roles="role1[;role2;...role_n]"
    [-users="user"]
    [-external_role="grantee_group"])

[ ] indicates that the parameter is optional
```

Parameters

- **roles**

  Grants one or more roles to BI Publisher. Specify one or more roles separated by a semicolon.

- **users**

  Users to receive the granted role.

- **external_role**

  Group to assign the role.

Examples

Example 1

This example grants one role to a group.

```
emcli grant_bipublisher_roles -roles="EMBIPViewer" -external_role="TESTGROUPNAME"
```

Example 2

This example grants more than one role to a group.

```
emcli grant_bipublisher_roles -roles="EMBIPViewer;EMBIPAuthor"
      -external_role="TESTGROUPNAME"
```

Example 3

This example grants one role to a user.

```
emcli grant_bipublisher_roles -roles="EMBIPViewer"
      -users="TESTUSERNAME"
```

Example 4

This example grants one role to multiple users.

```
emcli grant_bipublisher_roles -roles="EMBIPViewer"
      -users="TESTUSERNAME;TESTUSERNAME2"
```

Example 5

This example grants more than one role to multiple users and a group.

```
emcli grant_bipublisher_roles -roles="EMBIPViewer;EMBIPAuthor"
      -external_role="TESTGROUPNAME"
      -users="TESTUSERNAME;TESTUSERNAME2
```

grant_license_no_validation

Grants licenses on a set of user-specified packs, or all packs to a set of user-specified targets, or all targets belonging to the input licensable target type.

For 11g database targets, you cannot enable or disable the Database Diagnostic and Tuning Packs through the user interface. You need to set the control_management_pack_access initialization parameter to manage your licenses. For information about this parameter, see the Enterprise Database Management chapter of *Oracle Enterprise Manager Licensing Information*.

> **Tip:** You can use this verb to grant licenses for standalone target types, such as hosts and databases, but you cannot use this verb to grant licenses for the parent Application Server (oracle_ias) target type, which has dependent target types of OC4J, Jserv, Web Cache, and so forth. To do this, use the grant_license_with_validation verb instead.
>
> For example, for pack ias_config and an Application Server target of AS1 with an associated dependent target of OC4J1, this verb grants a license to AS1, but this does not propagate to OC4J1.

Format

```
emcli grant_license_no_validation
      -type="target_type"
      [-targets="tname1;tname2;..."]
      [-packs="pack1;pack2;..."]
      [-file="file_name"]
      [-displayAllMessages]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Target type as it exists in the database. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks. You can specify only one target type at a time; for example, -type="oracle_database".

- **targets**

  Targets should be specified in the following sequence:

  ```
  TargetName1;TargetName2;
  ```

  For example:

  ```
  -targets="database1;database2;database3;"
  ```

  The semi-colon (;) is the target separator.

  See the "Examples" section below for information about providing arguments for the targets .

- **packs**

  License packs should be specified in the following sequence:

  ```
  pack1;pack2;
  ```

For example:

```
-packs="db_diag;db_config;"
```

The semi-colon (;) is the pack separator.

See the "Examples" section below for information about providing arguments for the packs .

- **file**

  Specify the file name, including the complete path. For example:

  ```
  -file="/usr/admin1/db_license.txt"
  ```

  The file should contain the list of targets and packs according to the following cases:

  – If you only need to provide a list of targets, use the following format:

    ```
    targets=database1;database2;database3;
    ```

  – If you only need to provide a list of packs, use the following format:

    ```
    packs=db_diag;db_config;
    ```

  – If you need to provide a list of both targets and packs, use the following format:

    ```
    targets=database1;database2;database3;
    packs=db_diag;db_config;
    ```

- **displayAllMessages**

  Displays all messages. Only error messages are displayed by default. "=value" is not allowed on the command line.

Examples

Example 1 and Example 2 below grant licenses to specific packs for specific targets. In order to know which target types and pack names you can pass as arguments, you can use the view named mgmt_license_view to see a list of licensable targets, their target types, and the list of packs licensed on them.

To obtain this information, do the following:

1. Access SQL*Plus with your username and password, using sysman or other user that has access to sysman.mgmt_license_view.
2. Select a distinct pack name from sysman.mgmt_license_view, where:

   ```
   target_type=<oracle_database>
   ```

This example shows pack names for an Oracle database you specify as the target type.

```
PACK_NAME
------------------
db_config
provisioning
db_sadm
db_tuning
db_diag
provisioning_db
db_chgmgt
```

```
7 rows selected.
```

Based on this information, to grant a license to the database1 target for the db_chgmgt pack, you would enter the following command:

```
emcli grant_license_no_validation -type="oracle_database" -targets="database1"
-packs="db_chgmgt"
```

The only limitation of mgmt_license_view is that it only lists the packs for a target type where the pack is granted to at least one target of that type. That is, if the pack is not granted to any target of that type, mgmt_license_view cannot provide any information.

Example 1

This example grants the license to the db_diag and db_config packs to database1, database2, and database3 targets (oracle_database target type):

```
emcli grant_license_no_validation -type="oracle_database"
          -targets="database1;database2;database3;" -packs="db_diag;db_config;"
```

Example 2

This example grants the license to the db_diag and db_config packs to all database targets in the setup:

```
emcli grant_license_no_validation -type="oracle_database"
       -packs="db_diag;db_config;"
```

Example 3

This example grants the license to all packs (applicable to database targets) to database1, database2, and database3 targets in the setup:

```
emcli grant_license_no_validation -type="oracle_database"
         -targets="database1;database2;database3;"
```

Example 4

This example grants the license to all packs (applicable to database targets) to all database targets in the setup:

```
emcli grant_license_no_validation -type="oracle_database"
```

Example 5

This example uses a text file to pass targets and pack names as the argument. It grants the license to the db_diag and db_config packs to the database1, database2, and database3 targets (oracle_database target type):

```
emcli grant_license_no_validation -type="oracle_database"
          -file="/usr/admin1/db_license.txt"
             targets=database1;database2;database3;
                   packs=db_diag;db_config;
```

... where the content of the "/usr/admin1/license/db_license.txt" file is as follows:

```
targets=database1;database2;database3;
packs=db_diag;db_config;
```

grant_license_with_validation

Grants licenses on a set of user-specified packs, or all packs to a set of user-specified targets, or all targets belonging to the input licensable target type as per business rules.

For 11g database targets, you cannot enable or disable the Database Diagnostic and Tuning Packs through the user interface. You need to set the control_management_pack_access initialization parameter to manage your licenses. For information about this parameter, see the Enterprise Database Management chapter of *Oracle Enterprise Manager Licensing Information*.

> **Tip:** You can use this verb to grant licenses for standalone target types, such as hosts and databases, and you also use this verb to grant licenses for the parent Application Server (oracle_ias) target type, which has dependent target types of OC4J, Jserv, Web Cache, and so forth.
>
> For example, for pack ias_config and an Application Server target of AS1 with an associated dependent target of OC4J1, this verb grants a license to AS1 and also propagates to OC4J1 (and all other dependent targets associated with AS1).
>
> To grant licenses for only standalone target types, use the grant_license_no_validation verb.

Format

```
emcli grant_license_with_validation
      -type="target_type"
      [-targets="tname1;tname2;..."]
      [-packs="pack1;pack2;..."]
      [-file="file_name"]
      [-displayAllMessages]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Target type as it exists in the database. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks. You can specify only one target type at a time; for example, -type="oracle_database".

- **targets**

  Targets should be specified in the following sequence:

  ```
  TargetName1;TargetName2;
  ```

  For example:

  ```
  -targets="database1;database2;database3;"
  ```

  The semi-colon (;) is the target separator.

  See the "Examples" section below for information about providing arguments for the targets .

- **packs**

  License packs should be specified in the following sequence:

```
pack1;pack2;
```

For example:

```
-packs="db_diag;db_config;"
```

The semi-colon (;) is the pack separator.

See the "Examples" section below for information about providing arguments for the packs .

- **file**

  Specify the file name, including the complete path. For example:

  ```
  -file="/usr/admin1/db_license.txt"
  ```

  The file should contain the list of targets and packs according to the following cases:

  – If you only need to provide a list of targets, use the following format:

    ```
    targets=database1;database2;database3;
    ```

  – If you only need to provide a list of packs, use the following format:

    ```
    packs=db_diag;db_config;
    ```

  – If you need to provide a list of both targets and packs, use the following format:

    ```
    targets=database1;database2;database3;
    packs=db_diag;db_config;
    ```

- **displayAllMessages**

  Displays all messages. Only error messages are displayed by default. "=value" is not allowed on the cmd line.

Examples

Example 1 and Example 2 below grant licenses to specific packs for specific targets. In order to know which target types and pack names you can pass as arguments, you can use the view named mgmt_license_view to see a list of licensable targets, their target types, and the list of packs licensed on them.

To obtain this information, do the following:

1. Access SQL*Plus with your username and password, using sysman or other user that has access to sysman.mgmt_license_view.
2. Select a distinct pack name from sysman.mgmt_license_view, where:

   ```
   target_type=<oracle_database>
   ```

This example shows pack names for an Oracle database you specify as the target type.

```
PACK_NAME
--------------------
db_config
provisioning
db_sadm
db_tuning
db_diag
provisioning_db
```

```
db_chgmgt

7 rows selected.
```

Based on this information, to grant a license to the database1 target for the db_chgmgt pack, you would enter the following command:

```
emcli grant_license_with_validation -type="oracle_database" -targets="database1"
-packs="db_chgmgt"
```

The only limitation of mgmt_license_view is that it only lists the packs for a target type where the pack is granted to at least one target of that type. That is, if the pack is not granted to any target of that type, mgmt_license_view cannot provide any information.

Example 1

This example grants a license to the db_diag and db_config packs to database1, database2, and database3 targets (oracle_database target type):

```
emcli grant_license_with_validation -type="oracle_database"
          -targets="database1;database2;database3;" -packs="db_diag;db_config;"
```

Example 2

This example grants a license to the db_diag and db_config packs to all database targets in the setup:

```
emcli grant_license_with_validation -type="oracle_database"
       -packs="db_diag;db_config;"
```

Example 3

This example grants a license to all packs (applicable to database targets) to database1, database2, and database3 targets in the setup:

```
emcli grant_license_with_validation -type="oracle_database"
         -targets="database1;database2;database3;"
```

Example 4

This example grants a license to all packs (applicable to database targets) to all database targets in the setup:

```
emcli grant_license_with_validation -type="oracle_database"
```

Example 5

This example uses a text file to pass targets and pack names as the argument. It grants a license to the db_diag and db_config packs to the database1, database2, and database3 targets (oracle_database target type):

```
emcli grant_license_with_validation -type="oracle_database"
          -file="/usr/admin1/db_license.txt"
             targets=database1;database2;database3;
                   packs=db_diag;db_config;
```

where the content of the "/usr/admin1/license/db_license.txt" file is as follows:

```
targets=database1;database2;database3;
packs=db_diag;db_config;
```

grant_privs

Grants the privileges to the existing Enterprise Manager user or Enterprise Manager Role.

> **Note:** To replace an existing Enterprise Manager administrator role, use the modify_role verb.

Format

```
emcli grant_privs
        -name="username|rolename"
        -privilege="name[;secure_resource_details]"
        [-grant_all_targets_on_host="yes|no"]
        [-separator=privilege="sep_string"]
        [-subseparator=privilege="subsep_string"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  User name or role name to which privileges will be assigned.

- **privilege**

  Privilege to be granted to the Enterprise Manager user or role. You can specify this parameter more than once.

  Specify secure_resource_details as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1
  [:resource_column_name2=resource_column_value2]..]"
  ```

  Optionally, you can drop resource column names from this parameter if you provide resource information in the order described by emcli get_supported_ privileges. See the "See Also" section below for more information.

- **grant_all_targets_on_host**

  Indicates if the privilege needs to be granted on all targets of the host specified as part of the privilege parameter. The default value is no.

- **separator=privilege**

  Specify a string delimiter to use between name-value pairs for the value of the -privilege option. The default separator delimiter is a semi-colon (;).

- **subseparator=privilege**

  Specify a string delimiter to use between the name and value in each name-value pair for the value of the -privilege option. The default subseparator delimiter is a colon (:).

Examples

Example 1

This example grants these privileges to user1:

- Privilege to use any beacon
- Full control of the jobs with ID 923470234ABCDFE23018494753091111
- Full control on the target host1.example.com:host
- Full control on the credential cred1:user2
- View Privilege on target with ID 123451234ABCDFE23018494753092222

```
emcli grant_privs
      -name="user1"
      -privilege="USE_ANY_BEACON"
      -privilege="FULL_JOB;923470234ABCDFE23018494753091111"
      -privilege="FULL_TARGET;TARGET_NAME=host1.example.com:TARGET_TYPE=host"
      -privilege="FULL_CREDENTIAL;CRED_NAME=cred1:CRED_OWNER=user2"
      -privilege="FULL_CREDENTIAL;CRED_GUID=123451234ABCDFE23018494753092222"
```

Example 2

This example grants target privileges to EM Role : Role1:

```
emcli grant_privs
      -name="Role1"
      -privilege="FULL_TARGET;TARGET_NAME=host1.example.com:TARGET_TYPE=host"
```

Example 3

This example grants FULL_TARGET privilege on all targets on host host1.example.com to user1.

```
emcli grant_privs
      -name="user1"
      -privilege="FULL_TARGET;TARGET_NAME=host1.exemple.com:TARGET_TYPE=host"
      -grant_all_targets_on_host="yes"
```

Example 4

This example uses the separator and subseparator parameters to grant FULL_TARGET privilege on host1.example.com to user1.

```
emcli grant_privs
      -name="user1"
      -privilege="FULL_TARGET->TARGET_NAME=host1.example.com@@TARGET_TYPE=host"
      -separator=privilege="->"
      -subseparator=privilege="@@"
```

See Also

To see the complete list of privileges and resource column names, execute the following command:

```
emcli get_supported_privileges
```

To see the list of SYSTEM privileges, which do require resource information:

```
emcli get_supported_privileges -type=SYSTEM
```

To see the list of TARGET privileges:

```
emcli get_supported_privileges -type=TARGET
```

To see the list of JOB privileges:

```
emcli get_supported_privileges -type=JOB
```

grant_roles

Grants roles to an existing Enterprise Manager user or Enterprise Manager role.

Format

```
emcli grant_roles
     -name="username|rolename"
     [-roles="role1;role2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  User name or role name to which roles will be assigned.

- **roles**

  Roles that will be granted to an Enterprise Manager user or role. You can specify this option more than once.

Examples

```
emcli grant_roles
      -name="user1"
      -roles="SUPER_USER"

emcli grant_roles
      -name="Role1"
      -roles="BLACKOUT_ADMIN;MAINTAIN_TARGET"
```

help

Shows a summary of all verbs or command-line help for individual EM CLI verbs.

> **Note:** EM CLI must be set up and configured before command line help is available for all verbs.

Format

```
emcli help [verbname]

[ ] indicates that the parameter is optional
```

Parameters

None.

Examples

Example 1

This example provides an overview for all available verbs:

```
emcli help
```

Example 2

This example provides the description, syntax, and usage examples for the add_target verb:

```
emcli help add_target
```

ignore_instance

Ignores a failed step. An instance cannot be ignored when it completes, completes with an error, is suspended, or is stopped.

Format

```
emcli ignore_instance
      -instance=<instance_guid>
      [exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
      [-stateguid=<state_guid>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

- **stateguid**

  Comma-separated list of state GUIDs.

Example

```
emcli ignore_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid=51F762417C4943DEE040578C4E087168
```

import_adm

Imports an Application Data Model from the specified XML file.

Format

```
emcli import_adm
     -file=<file_name>
     -adm_name=<application_data_model_name>
     -target_name=<target_name>
     -target_type=<target type>
     [-desc=<description>]

[ ] indicates that the parameter is optional
```

Parameters

- **file**

  File name with the absolute path of the XML file.

- **adm_name**

  Model name with which the Application Data Model will be imported.

- **target_name**

  Target for which the Application Data Model will be created.

- **target_type**

  Target type of the target for which the Application Data Model will be created.

- **desc**

  Application Data Model description.

Output

Success/error messages.

Examples

This example imports the Application Data Model from the sample_adm_import.xml file as Sample_ADM.

```
emcli import_adm
       -file=/home/user/sample_adm_import.xml
       -adm_name=Sample_ADM
       -target_name=test_database
       -target_type=oracle_pdb
       -desc="Application Data Model for EBS"
```

import_appreplay_workload

Imports a workload metadata XML file and creates a new application replay workload object. A Workload metadata XML file, which is stored in the workload root directory, is automatically generated as part of the workload capture process. The XML file contains a pointer to the actual raw captured workload data files. If you are importing a workload captured by one Enterprise Manager system to another, make sure the workload storage location specified in the XML file is reachable and contains the workload data files.

Format

```
emcli  import_appreplay_workload
        -input_file=template:<input_filename>

[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Fully-qualified path to a workload metadata XML file. The workload XML file is automatically created during capture. However, you may need to make necessary changes to the XML file before you import. For example, you may want to change the workload name in the exported file and rename the XML file to match the workload name. You may also need to modify the storage locations to point to where the workload data files are located if you have moved the captured data files.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

import_charge_plans

Imports charge plan metadata from the specified file.

Format

```
emcli import_charge_plans
        [-charge_plan="plan_name"[-entity_type=entity_type_name]]
        [<-create|-create_revision|-validate|-describe>]
        [-start_date=ddmmyyyy]
        -file=file_name

[ ] indicates that the parameter is optional
```

Parameters

- **charge_plan**

  Name of the charge plan to import. If this parameter is not specified, imports all charge plans within the file.

- **entity_type**

  Name of the Chargeback entity type whose charge rate metadata to import from the specified charge plan within the file. If this parameter not specified, import all entity type charge rates from the charge plan.

- **create**

  Import charge plan metadata to create a charge plan.

- **create_revision**

  Import charge plan metadata to create a charge plan revision.

- **validate**

  Validate the charge plan metadata file. This is the default action if no import operation is specified.

- **describe**

  Describe the charge plan metadata in the specified file.

- **start_date**

  Start date in ddmmyyyy format of the report cycle for the applicable charge plan import operation. If this parameter not specified, uses the start date of the current report cycle.

- **file**

  Absolute path of the XML file containing the charge plan metadata to import.

Examples

Example 1

This example validates charge plan metadata in /home/allplans.xml:

```
emcli import_charge_plans
        -file=/home/allplans.xml
        -validate
```

Example 2

This example describes charge plan metadata in /home/allplans.xml:

```
emcli import_charge_plans
        -file=/home/allplans.xml
        -describe
```

Example 3

This example imports Plan A charge plan metadata in /home/plans.xml to create a plan, using the current report cycle start date as the plan's effective date:

```
emcli import_charge_plans
        -charge_plan="Plan A"
        -file=/home/plans.xml
        -create
```

Example 4

This example imports Plan B charge plan metadata in /home/plans.xml to create a plan, with an effective start date of 01092014:

```
emcli import_charge_plans
        -charge_plan="Plan B"
        -file=/home/plans.xml
        -create
        -start_date=01092014
```

Example 5

This example imports Plan C charge plan metadata in /home/plans.xml to create a plan revision with an effective start date of 01092014:

```
emcli import_charge_plans
        -charge_plan="Plan C"
        -file=/home/plans.xml
        -create_revision
        -start_date=01092014
```

Example 6

This example imports Chargeback host entity type metadata from Plan D in /home/plans.xml to create a plan using the current report cycle start date as the plan's effective date:

```
emcli import_charge_plans
        -charge_plan="Plan D"
        -file=/home/plans.xml
        -entity_type=host
        -create
```

import_compliance_object

Imports a compliance object into the repository.

Format

```
 import_compliance_object
      -files=file1;file2;...  [-overwrite] [-deep]

[ ] indicates that the parameter is optional
```

Parameters

- **files**

  Files to be imported.

- **overwrite**

- **deep**

Examples

```
emcli import_compliance_object
       -files=file1.xml;file2.xml -overwrite
```

import_custom_charge_items

Imports user-defined charge item metadata from the specified file.

Format

```
emcli import_custom_charge_items
        -file=file_name
        [-validate]

[ ] indicates that the parameter is optional
```

Parameters

- **file**

  Absolute path of the XML file from which to import user-defined charge item metadata.

- **validate**

  Validates the XML file.

Examples

Example 1

This example imports user-defined charge item metadata from /home/host.xml:

```
emcli import_custom_charge_items
        -file=/home/host.xml
```

Example 2

This example validates user-defined charge item metadata in /home/host.xml:

```
emcli import_custom_charge_items
        -file=-file=/home/host.xml
        -validate
```

import_custom_plugin_update

Imports a custom plug-in update that was created using the Extensibility Development Kit. The imported plug-in update is used for all subsequent plug-in deployments on the Management Agents.

Format

```
emcli import_custom_plugin_update
      -archive="<path_to_plugin_update_archive>"
       [-overwrite]
```

Parameters

- **archive**

  Absolute path to the custom update archive file.

- **overwrite**

  Overwrites an existing custom plug-in update, if a custom plug-in update already exists for that plug-in. If not provided, the custom plug-in update is not imported for that plug-in. Applies only to subsequent plug-in deployments. Does not automatically redeploy on the Management Agents where the already-existing plug-in was previously deployed. To redeploy on such Management Agents, run the emcli redeploy_plugin_on_agent verb.

Examples

Example 1

The following example imports the `12.1.0.4.0_oracle.sysman.db.008.zip` archive file from the `/u01/oracle/plugin_updates/` location, assuming no custom plug-in update already exists for that plug-in.

```
emcli verb_name
      -archive="/u01/oracle/plugin_updates/12.1.0.4.0_oracle.sysman.db.008.zip"
```

Example 2

The following example imports the `12.1.0.4.0_oracle.sysman.db.008.zip` archive file from the `/u01/oracle/plugin_updates/` location, and overwrites any existing custom plug-in update that already exists for that plug-in.

```
emcli import_custom_plugin_update
      -archive="/u01/oracle/plugin_updates/12.1.0.4.0_oracle.sysman.db.008.zip"
      -overwrite
```

import_jobs

Imports all job definitions into Enterprise Manager, including Corrective Actions from a zip file. Library jobs are created. The EM CLI logged-in user is set as the library job owner.

Format

```
emcli import_jobs
   -import_file=<zip_file_name>"
   [-name="job name1;job_name2;..."]
   [-type="job type1;job_type2;..."]
   [-targets="tname1:ttype1;tname2:ttype2;..."]
   [-owner="owner1;owner2;..."]
   [-preview]
   [-force]
   [-stoponerror]

[ ] indicates that the parameter is optional
```

Parameters

- **import_file**

  Zip file name that contains job definitions.

- **name**

  Job name to be used for filtering. Semicolon-separated job names can be provided. Filtering by using a wildcard character is not supported.

- **type**

  Job type to be used for filtering. Semicolon-separated job types can be provided. Filtering by using a wildcard character is not supported.

- **targets**

  Target name and target type to be used for filtering. Semicolon-separated target names and types can be provided. Filtering by using a wildcard character is not supported.

- **owner**

  Job owner to be used for filtering. Semicolon-separated owners can be provided. Filtering by using a wildcard character is not supported.

- **preview**

  Prints the job definitions in the zip file. Filter values provided are used to show only matching job definitions. Jobs are not created in Enterprise Manager.

- **force**

  Updates the job record if it already exists. Otherwise, the job record is created. When this option is not specified, the default behavior of the system is to always create jobs from the import file.

- **stoponerror**

  Stops the import operation is after the first failure of the job import and rolls back the transaction. All jobs created by using this EM CLI session are deleted.

Output Columns

Success/Error messages.

Examples

Example 1

This example displays all job definitions in the zip file.

```
emcli import_jobs -preview -import_file=job_data.zip
```

Example 2

This example displays all job definitions owned by SYSMAN or ADMIN from the zip file.

```
emcli import_jobs -owner=SYSMAN;ADMIN -preview -import_file=job_data.zip
```

Example 3

This example import jobs MYJOB1 and MYJOB2.

```
emcli import_jobs -name=MYJOB1;MYJOB2 -import_file=job data.zip
```

Example 4

This example imports all jobs owned by user SYSMAN:

```
emcli export_jobs -owner=ADMIN% -export_file=jobsdata.zip
```

Example 5

This example imports all job definitions into Enterprise Manager.

```
emcli import_jobs  -import_file=job data.zip
```

Example 6

This example imports all job definitions into Enterprise Manager. If the job already exists, the details are edited. Otherwise, a new job is created.

```
emcli import_jobs  -import_file=job data.zip -force
```

Example 7

This example imports all job definitions into Enterprise Manager, and on the first failure, rolls back the jobs created in this session. The remaining jobs from the import file are not processed. Otherwise, a new job is created.

```
emcli import_jobs  -import_file=job data.zip -stoponerror
```

import_masking_definition

Imports a masking definition from the specified XML file.

Format

```
emcli import_masking_definition
     -file=/tmp/file_name.xml
```

Parameters

- **file**

  Path of the file containing the masking definition in XML format.

Output

Success or error messages.

Examples

This example imports the masking definition from the hr_mask.xml file.

```
emcli import_masking_definition
       -file=/tmp/hr_mask.xml
```

import_metric_extension

Imports a metric extension archive file.

Format

```
emcli import_metric_extension
        -file_name=<metric_extension_archive>
        -rename_as=<metric_extension_to_import_as>
```

Parameters

- **file_name**

  Name of the metric extension archive file to be imported.

- **rename_as**

  Imports the metric extension using the specified name, replacing the name given in the archive.

Examples

This example imports the masking definition from the hr_mask.xml file.

```
emcli import_metric_extension
      -file_name=<file name>
      -rename_as=<metric extension name>
```

import_report

Imports one or more Information Publisher report definitions from an XML file(s) using the title in the XML file and the currently logged-in CLI user as the owner of the report. If the report/owner already exists, the operation fails for this report with an accompanying error message. (You can override this with the -force option.) The report will be changed to a just-in-time report with the target type from the exported report.

You will need to edit schedules and access privileges using the Enterprise Manager user interface. The system enforces title/owner uniqueness, so an error occurs if a report with the same title and owner already exists.

Format

```
emcli import_report
      -files="file1;file2;..."
      [-force]

[ ] indicates that the parameter is optional
```

Parameters

- **files**

  List of path/file name(s) of XML file(s) that contain valid report definition(s).

- **force**

  First delete the report (and all jobs and saved copies) if a report with the same title/owner exists.

Examples

```
emcli import_report
     -files="$HOME/reports/maint_report1.xml;$HOME/reports/file2.xml"
```

import_sla

Imports an SLA configuration XML file for a target. This verb provides the functionality of creating a new SLA, creating a new version, and creating a new copy.

Note: The XML file can only contain one SLA to be imported; that is, when export_sla has successfully exported a file when slaName and version are specified.

Note: The target must have the metrics required by the SLA template's SLI. If the template's SLI calls for a metric not found in the target, the SLI cannot be created.

Format

```
emcli import_sla
      -targetName=<target name>
      -targetType=<target type>
      -input_file=slaTemplate:<input filename>
      [-slaName=<SLA name>]

[ ] indicates that the parameter is optional
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **input_file**

  Name of the input file. There can only be one SLA root node in the XML document.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **slaName**

  Specifying this name overrides the name contained in the SLA template XML file. This effectively creates a new SLA version series starting with version 1.

Examples

This example creates an SLA named 'gold_sla' for the target my_service (generic_service).

```
emcli import_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla' -input_file=slaTemplate:'service_sla.xml'
```

import_subset_definition

Imports a subset definition from the specified XML file.

Format

```
emcli import_subset_definition
      -adm_name=<Application_Data_Model_Name>
      -subset_name=<Subset_Definition_Name>
      -target_name=<Target_Database_Name>
      -target_type=<Target_Type>
      -file=<Import_File>
      [-db_pref_cred_name=<DBCredsNormal | DBCredsSYSDBA>]
      [-db_cred_name=<Database_Credential_Name>]
      [-description=<Description>]
      [-job_name=<Job_Name>]
      [-job_description=<Job_Description>]

[ ] indicates that the parameter is optional
```

Parameters

- **adm_name**

  Application Data Model (ADM) name.

- **subset_name**

  Name of the imported subset definition.

- **target_name**

  Target database name.

- **target_type**

  Type of target. Possible values are 'oracle_database', 'rac_database' and 'oracle_pdb'.

- **file**

  Fully-qualified file name of the file in XML format.

- **db_cred_name**

  Name of existing credentials stored in the Enterprise Manager repository to connect selected target database.

  You must provide a value for either db_cred_name or db_pref_cred_name.

- **db_pref_cred_name**

  Name of preferred credentials stored in the Enterprise Manager repository.

  You must provide a value for either db_cred_name or db_pref_cred_name.

  Valid values for this parameter are:

  – DBCredsNormal: Default normal credential set for an oracle_database target.

  – DBCredsSYSDBA: SYSDBA credential set for an oracle_database target.

- **description**

  Description for the imported subset definition.

- **job_name**

Job name for the import subset definition operation.

- **job_description**

  Job description.

Examples

Example 1

This example imports a subset definition from an XML file at path /scratch/samples/HR_subset.xml.

```
emcli import_subset_definition
      -adm_name=adm
      -file=/scratch/samples/HR_subset.xml
      -subset_name=HR_Subset
      -db_cred=cred
      -target_name=sample_database
      -target_type=oracle_database
```

Example 2

This example imports a subset definition from an XML file at path /scratch/samples/HR_subset.xml using preferred normal database credentials.

```
emcli import_template -files="e1.xml;e2.xml;e3.xml"
```

import_subset_dump

Imports the dump file into the specified target database.

Format

```
emcli import_subset_dump
      -target_name=<Target Database>
      -target_type=<Target Database type>
      [-db_cred_name=<Database Credential Name>]
      [-db_pref_cred_name=<DBCredsNormal | DBCredsSYSDBA> ]
      [-host_cred_name=<Host Credential Name>]
      [-subset_directory=<Database Directory Object Name> ]
      [-custom_directory_path=<Custom Directory Path>]
      [-use_external_directory]
      [-external_directory=<External Directory Object Name>]
      [-export_file_name=<Exported Dump File Name>]
      [-max_imp_threads=< Maximum Number of Import Threads>]
      [-encrypted_dump_file]
      [-encryption_password=<Encryption Password>]
      [-import_type=<ALL | DATA_T_L| DATA_A_L> ]
      [-tablespace_map=<Tablespace Map>]
      [-schema_map=<Schema Map>]
      [-log_file_name=<Log file name>]
      [-job_name=<Job Name>]
      [-job_description=<Job Description>]
      [-oid_transform]

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Name of the existing target database.

- **target_type**

  Type of target. Possible values target type are 'oracle_database', 'rac_database', and 'oracle_pdb'.

- **db_cred_name**

  Name of existing credentials stored in the Enterprise Manager repository to connect selected target database. You must provide a value for either db_pref_cred_name_or db_cred_name.

- db_pref_cred_name

  Name of preferred credentials stored in the Enterprise Manager repository.

  Valid values are:

  – DBCredsNormal — Default normal credential set for an oracle_database target.

  – DBCredsSYSDBA — SYSDBA credential set for an oracle_database target.

  You must provide a value for either db_pref_cred_name_or db_cred_name.

- **host_cred_name**

Name of existing host credentials stored in the Enterprise Manager repository to access the target host.

- **subset_directory**

  Database Directory where the dump file is stored. For example: DATA_PUMP_DIR

  You must provide a value for either subset_directory or custom_directory_path.

- **custom_directory_path**

  User-specified directory location on the target host where the dump file is present. For example: /scratch/user/subset_dir

  You must provide a value for either subset_directory or custom_directory_path.

- use_external_directory

  Flag to enable using an external directory (clustered/shared file system or ASM) for faster import processing. If you do not set this parameter, you must provide a value for external_directory.

- **external_directory**

  External directory location (clustered/shared file system or ASM) object for faster host access. For example: DATA_PUMP_DIR

- **export_file_name**

  Name of the dump file to import. If not specified, the default value is EXPDAT%U.DMP.

- max_imp_threads

  Maximum number of import threads. If not specified, the default value is 1.

- **encrypted_dump_file**

  Set this option if an encryption password was specified during the export operation. If you use this option, you must also provide a value for encryption_ password.

- **encryption_password**

  Password to decrypt encrypted data during an import operation. The specified password should be same as that specified during the export operation. If the encrypted_dump_file option is set and a value for this option is not specified, you are prompted for the encryption password.

  For a secure operation, it is recommended that passwords not be stored in the scripts, but instead specified when prompted for them.

- import_type

  Drives an import operation. Valid values are:

  – ALL: Import both metadata and data.

  – DATA_T_L: Data within the preexisting table will be removed. Data in the import source will replace it.

  – DATA_A_L: Data contained within the table to be imported will be appended to the end of the preexisting table.

  The default value is ALL.

- **tablespace_map**

This password is required to re-map data from one tablespace to another. For example:

```
-tablespace_map="source_tbsp1:target_tbsp1;source_tbsp2:target_tbsp2"
```

- **schema_map**

  This password is required to re-map data from one schema to another.

  ```
  -schema_map="source_schema1:target_schema1;source_schema2:target_schema2"
  ```

- log_file_name

  If not specified, the default value is IMPORT.LOG.

- oid_transform

  By default, the exported OID is imported during table or type creation. Set this option to create a new OID. This is useful when some of the objects already exist in the database and a cloned copy is required. However, selecting this option will cause breakage in REF columns that point to the table.

- **job_name**

  Import subset dump operation job name.

- job_description

  Job description.

Output

Success or error message along with the job name if applicable.

Examples

Example 1

This example imports dump(E.dmp) located at the DATA_PUMP_DIR directory into the target sample_database.

```
emcli import_subset_dump -db_cred_name=db_cred -export_file_name=E.dmp -host_cred_
name=host_cred -subset_directory=DATA_PUMP_DIR -target_type=oracle_database
-target_name=sample_database -import_type=All
```

Example 2

This example imports dump(E.dmp) located at the DATA_PUMP_DIR directory into the target sample_database using preferred database and host credentials.

```
emcli import_subset_dump -export_file_name=E.dmp -db_pref_cred_name=DBCredsNormal
-subset_directory=DATA_PUMP_DIR -target_type=oracle_database -target_name=sample_
database -import_type=All
```

Example 3

This example imports dump(E.dmp) located at the DATA_PUMP_DIR file directory target sample_database with oid transformation enabled.

```
emcli import_subset_dump -db_cred_name=db_cred -export_file_name=E.dmp -host_cred_
name=host_cred -subset_directory=DATA_PUMP_DIR -target_type=oracle_database
-target_name=sample_database -import_type=All -oid_transform
```

Example 4

This example imports the encrypted data dump(E.dmp) located at the DATA_PUMP_DIR directory into the target sample_database. You are prompted for encryption_password.

```
emcli import_subset_dump -db_cred_name=db_cred -export_file_name=E.dmp -host_cred_
name=host_cred -subset_directory=DATA_PUMP_DIR -target_type=oracle_database
-target_name=sample_database -import_type=All -encrypted_dump_file
```

Example 5

This example imports dump(E.dmp) located at the DATA_PUMP_DIR file directory target sample_database with schema mapping.

```
emcli import_subset_dump -db_cred_name=db_cred -export_file_name=E.dmp -host_cred_
name=host_cred -subset_directory=DATA_PUMP_DIR -target_type=oracle_database
-target_name=sample_database -import_type=All -schema_map="HR:HR_COPY;OE:OE_COPY"
```

Example 6

This example imports dump(E.dmp) located at the custom directory location(\scratch\user\custom_dir) into the target sample_database with schema mapping.

```
emcli import_subset_dump -db_cred_name=db_cred -export_file_name=E.dmp -host_cred_
name=host_cred -custom_directory_path=\scratch\user\custom_dir -target_
type=oracle_database -target_name=sample_database -import_type=All -schema_
map="HR:HR_COPY;OE:OE_COPY"
```

import_template

Imports a monitoring template from an XML or zip file. The resulting definition is saved in the repository.

Format

```
emcli import_template
      -files="file1;file2;..."
```

Parameters

- **files**

  Path/file name of an XML file, which contains a valid template definition. You can specify multiple files with this option by separating each file with a semi-colon (;).

Examples

Example 1

This example imports a template from `template.xml`.

```
emcli import_template -files="template.xml"
```

Example 2

This example imports three templates — one from each of the files specified.

```
emcli import_template -files="e1.xml;e2.xml;e3.xml"
```

Example 3

This example imports a template from the template.zip file along with any metric extensions.

```
emcli import_template -files="template.zip"
```

import_update

Imports a Self Update archive file into Enterprise Manager. Upon successful import, the update is displayed on the Self Update Home in downloaded status for further action.

Format

```
emcli import_update
        -file="file"
        -omslocal
emcli import_update
        -file="file"
        -host="hostname"
        [-credential_set_name="setname"] | -credential_name="name"
        -credential_owner="owner"

[ ] indicates that the parameter is optional
```

Parameters

- **file**

  Complete path name of the update archive file.

- **omslocal**

  Flag specifying that the file is accessible from the OMS.

- **host**

  Target name for a host target where the file is available.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target. Can be one of the following:

  HostCredsNormal — Default unprivileged credential set

  HostCredsPriv — Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. You must specify this along with the credential_owner .

- **credential_owner**

  Owner of a named credential stored in the repository. You must specify this option along with the credential_name option.

Examples

Example 1

This example imports the file update1.zip. The file must be present on the OMS host. In a multiple OMS setup, any OMS can process the request, so the file should be accessible from the OMS processing the request. This usually means that the file must be kept on a shared location accessible from all OMSes.

```
emcli import_update
        -file="/u01/common/update1.zip"
```

```
        -omslocal
```

Example 2

This example imports the file update1.zip that is present on the host host1.example.com. The host must be a managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The preferred unprivileged credentials for host host1.example.com are used to retrieve the remote file.

```
emcli import_update
        -file="/u01/common/update1.zip"
        -host="host1.example.com"
        -credential_set_name="HostCredsNormal"
```

Example 3

This example imports the file update1.zip that is present on the host host1.example.com. The host must be a managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The named credentials "host1_creds" owned by user "admin1" are used to retrieve the remote file.

```
emcli import_update
        -file="/u01/common/update1.zip"
        -host="host1.example.com"
        -credential_name="host1_creds"
        -credential_owner="admin1"
```

import_update_catalog

Imports a Self Update master catalog file when Enterprise Manager is configured in offline mode. All updates present in the catalog are processed, and the applicable updates are displayed on the Self Update Home for further action.

Format

```
emcli import_update_catalog
        -file="file"
        -omslocal
        -file="file"
        -host="hostname"
        [-credential_set_name="setname"] | -credential_name="name"
         -credential_owner="owner"

[ ] indicates that the parameter is optional
```

Parameters

- **file**

  Complete path name of the self update catalog file.

- **omslocal**

  Flag specifying that the file is accessible from the OMS.

- **host**

  Target name for a host target where the file is available.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target. Can be one of the following:

  HostCredsNormal — Default unprivileged credential set

  HostCredsPriv — Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. You must specify this along with the credential_owner option.

- **credential_owner**

  Owner of a named credential stored in the repository. You must specify this option along with the credential_name option.

Examples

Example 1

This example imports the master catalog file p9984818_121000_Generic.zip. The file must be present on the OMS host. In a multiple OMS setup, the request can be processed by any OMS, so the file should be accessible from the OMS processing the request. This usually means that the file must be kept on a shared location accessible from all OMSes.

```
emcli import_update_catalog
        -file="/u01/common/p9984818_121000_Generic.zip"
        -omslocal
```

Example 2

This example imports the master catalog file p9984818_121000_Generic.zip that is present on the host host1.example.com. The host must be a managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The preferred unprivileged credentials for host host1.example.com are used to retrieve the remote file.

```
emcli import_update_catalog
        -file="/u01/common/p9984818_121000_Generic.zip"
        -host="host1.example.com"
        -credential_set_name="HostCredsNormal"
```

Example 3

This example imports the master catalog file p9984818_121000_Generic.zip that is present on the host host1.example.com. The host must be a managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The named credentials "host1_creds" owned by user "admin1" are used to retrieve the remote file.

```
emcli import_update_catalog
        -file="/u01/common/p9984818_121000_Generic.zip"
        -host="host1.example.com"
        -credential_name="host1_creds"
        -credential_owner="admin1"
```

list

Lists resource data. The maximum number of rows displayed is controlled by OMS property oracle.sysman.core.dataservice.max_fetch_rows. When the property is not set, it uses the default value of 2000.

Format

```
emcli list
        [-help]
        [-resource="list_resource_name"]
        [-columns="column_options"]
        [-colsize="column_sizes"]
        [-search="search_options"]
        [-bind="bind_parameters"]
        [-sql="sql"]
        [-script | -format=
                    [name:<pretty|script|csv>];
                    [column_separator:"column_sep_string"];
                    [row_separator:"row_sep_string"];
        ]
        [-noheader]

[ ] indicates that the parameter is optional
```

Parameters

- **help**

  Lists all resource names with their descriptions. You can use this option with the -resource option to see more details about the resource.

- **resource**

  Resource name for which data is displayed.

- **columns**

  Specify columns as shown, separated by commas:

  ```
  -columns="colname,colname,colname"
  ```

  Example:

  ```
  -columns="COL1,COL3,COL5"
  ```

  Specify column size and width as shown below. A colon precedes the size for a given column.

  ```
  -columns="colname:colsize,colname,colname"
  ```

  Example:

  ```
  -columns="COL1:30,COL3,COL5"
  ```

- **colsize**

  Resizes column widths. Most resource columns have some default widths. You can override them with this option.

  Example: -colsize="col1:30,col2:5"

- **search**

You can specify multiple search options. The usage is -search="ColumnName Operator 'Value'. The search value must be enclosed in quotes unless searching for null or not null.

The following operators are supported:

= !+ > < >= <= like

The option also supports is null and is not null.

- **bind**

  Use for resources that require specific input. The usage is -bind="Name Operator Value".

- **sql**

  Specifies arbitrary SQL against views. This query is executed as MGMT_VIEW user.

- **script**

  Sets the default column separator to a tab and the default row separator to a newline. You can change the column and row separator strings to change these defaults.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

- **noheader**

  Displays tabular output without column headers.

Output

When run in script mode, returns JSON output that can be easily parsed.

Exit Codes:

- 0 — Appears when successful.
- 1 — Appears when the list service fails to process the request.

Examples

These examples list all resource names.

Example 1 - Command-Line

```
emcli list -help
```

Example 2 - Scripting and Interactive

```
list (help)
```

These examples list column information about the 'Administrators' resources. They also list which columns users can search.

Example 3 - Command-Line

```
emcli list
      -help
      -resource=Administrators
```

Example 4 - Scripting and Interactive

```
list
     (help
     ,resource=Administrators)
```

These examples list all data for the 'Administrators' resource.

Example 5 - Command-Line

```
emcli list -resource=Administrators
```

Example 6 - Scripting and Interactive

```
list (resource=Administrators)
```

These examples list only user_name and user_type columns.

Example 7 - Command-Line

```
emcli list
      -resource=Administrators
      -columns="USER_NAME,USER_TYPE"
```

Example 8 - Scripting and Interactive

```
list
      (resource=Administrators
      ,columns="USER_NAME,USER_TYPE")
```

These examples show details about SYSMAN users.

Example 9 - Command-Line

```
emcli list
      -resource=Administrators
      -columns="USER_NAME,USER_TYPE"
      -search="USER_NAME = 'SYSMAN'"
```

Example 10 - Scripting and Interactive

```
list
      (resource=Administrators
      ,columns="USER_NAME,USER_TYPE"
      ,search="USER_NAME = 'SYSMAN'")
```

list_active_sessions

Lists active sessions on all OMSes in the environment. By default, the verb prints a summary for each OMS.

Format

```
emcli list_active_sessions
        [-details
        [-table]
        [-script]
        [-format=name:value;name:value]
        [-noheader]]

[ ] indicates that the parameter is optional
```

Parameters

- **details**

  Displays active user sessions on each OMS. The output format is non-tabular.

- **table**

  Prints details in table format.

- **script**

  Prints output that can be processed by script.

- **format**

  Supports the following name/value pairs:

  csv — Output will be comma-separated

  script — Output will be in a format that can be processed by script. You can also specify row_separator and column_separator.

- **noheader**

  Skips the header.

Examples

```
emcli list_active_sessions
    emcli list_active_sessions -details
    emcli list_active_sessions -details -table
    emcli list_active_sessions -details -table -script
    emcli list_active_sessions -details -table -script -noheader
    emcli list_active_sessions -details -table -format="name:csv"
    emcli list_active_sessions -details -table -format="name:script;row_
separator:@@;column_separator:!"
```

list_add_host_platforms

Lists the platforms on which the Add Host operation can be performed.

Format

```
emcli list_add_host_platforms
        [-all]
        [-noheader]
        [-script | -format=
                    [name:<pretty|script|csv>];
                    [column_separator:"column_sep_string"];
                    [row_separator:"row_sep_string"];
        ]

[ ] denotes that the parameter is optional
```

Parameters

- **all**

  Displays all of the platforms, including those for which the Agent software is not available.

- **noheader**

  Displays tabular output without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format=`column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Platform ID, Platform Name

Examples

Example 1

This example displays the platforms for which the agent software is available so that the Add Host operation can be performed.

```
emcli list_add_host_platforms
```

Example 2

This example displays all of the platforms, including those for which the Agent software is not available.

```
emcli list_add_host_platforms -all
```

list_add_host_sessions

Lists all of the Add Host sessions.

Format

```
emcli list_add_host_sessions
        [-host_name="Host name"]
        [-session_name="Session name"]
        [-match_all]
        [-noheader]
        [-script | -format=
                    [name:<pretty|script|csv>];
                    [column_separator:"column_sep_string"];
                    [row_separator:"row_sep_string"];
        ]

[ ] denotes that the parameter is optional
```

Parameters

- **host_name**

  Displays all of the Add Host sessions that the provided host is a part of.

- **session_name**

  Displays all of the sessions that match the session name provided.

- **match_all**

  Displays results that match all of the provided query criteria. By default, the results that match any of the provided query criteria are displayed.

- **noheader**

  Displays tabular output without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Session Name, Deployment Type, Host, Initialization, Remote Prerequisite, Agent Deployment

Examples

Example 1

This example displays all of the Add Host sessions.

```
emcli list_add_host_sessions
```

Example 2

This example displays all of the Add Host sessions that the host 'example.com' was part of.

```
emcli list_add_host_sessions -host_name=example.com
```

Example 3

This example displays all of the Add Host sessions whose session name contains the string 'Jan_1'.

```
emcli list_add_host_sessions -session_name=Jan_1
```

Example 4

This example displays all of the Add Host sessions that the host 'example.com' was part of, OR whose session name contains the string 'Dec_25'.

```
emcli list_add_host_sessions -host_name=example.com -session_name=Dec_25
```

Example 5

This example displays all of the Add Host sessions that the host 'example.com' was part of, AND whose session name contains the string 'Jan_15'.

```
emcli list_add_host_sessions -host_name=example.com -session_name=Jan_15 -match_
all
```

list_adms

Lists the names, source target name, and application suites of existing Application Data Models.

Format

```
emcli list_adms
```

Output

List of Application Data Models.

Examples

This example lists all Application Data models.

```
emcli list_adms
```

list_allowed_pairs

Lists allowed association types for the specified source and destination target types.

Format

Standard Mode

```
emcli list_allowed_pairs
       -source_type="source type"
       -dest_type="dest type"]
       [-noheader]
       [-script]
       [-format="[name:<pretty|script|csv>];[column_separator:
       "column_sep_   string"];[row separator:"row_sep_string"]"]
```

Interactive (Script) Mode

```
list_allowed_pairs(
      source_type="source type"
      [,dest_type="dest type"]
      [,noheader=True/False]
      [,script=True/False]
      [,format="[name:<pretty|script|csv>];[column_separator:
      "column_sep_string"];[row_separator:"row_sep_string"]"]
      )

[ ] indicates that the parameter is optional.
```

Parameters

- **source_type**

  Source target type.

- **dest_type**

  Destination target type.

- **noheader**

  Displays the output in tabular output without column headers.

- **script**

  Prints the output in a format that can be used in scripting.

- **format**

  Specifies how the output is formatted. The default value is "name:pretty", which prints the output table in a readable format not intended to be parsed by scripts. Other format options include:

  – format="name:script"
  Sets the default column separator to a tab and the default row separator to a newline in the output. You can override the column and row separator strings with your own values.

  – format="name:script;column_separator:<*column_sep_string*>"
  Causes the verb output to be column-separated by <*column_sep_string*>. Rows are separated by the newline character.

- format="name:script;row_separator:<*row_sep_string*>"
  Causes the verb output to be row-separated by <*row_sep_string*>.
- format="name:script;column_separator:<*column_sep_string*>;row_
  separator:<*row_sep_string*>"
- Causes the verb output to be column-separated by <*column_sep_string*> and row- separated by <*row_sep_string*>.
- format="name:csv"
  Sets the default column separator to a comma and the default row separator to a newline in the output.

Output

Exit Codes

0 indicates that the verb processing was successful.

Non-zero values indicate that the verb processing was not successful.

Example

This example lists allowed associations for the source target type "cluster" and the destination target type "host":

```
emcli list_allowed_pairs
       -source_target_type="cluster"
       -dest_target_type="host"
```

list_aru_languages

Lists ARU language information.

Format

```
emcli list_aru_languages
        [-name="language_name" | -id="language_id"]
        [-noheader]
        [-script | -format=
                   [name:<pretty|script|csv>];
                   [column_separator:"column_sep_string"];
                   [row_separator:"row_sep_string"];

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Language name.

- **id**

  Language ID.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_aru_languages
      emcli list_aru_languages -noheader
      emcli list_aru_languages -name="language name" -format="name:pretty"
      emcli list_aru_languages -id="language id" -format="name:script"
```

See Also

create_patch_plan

delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

list_aru_platforms

Lists ARU platform information.

Format

```
emcli list_aru_platforms
        [-name="platform_name" | -id="platform_id"]
        [-noheader]
        [-script | -format=
                          [name:<pretty|script|csv>];
                          [column_separator:"column_sep_string"];
                          [row_separator:"row_sep_string"];

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Platform name.

- **id**

  Platform ID.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_aru_platforms
      emcli list_aru_platforms -noheader
      emcli list_aru_platforms -name="platform_name" -format="name:pretty"
      emcli list_aru_platforms -id="platform id" -noheader -format="name:script"
```

See Also

create_patch_plan

delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

list_aru_products

Lists ARU product information.

Format

```
emcli list_aru_products
        [-name="product_name" | -id="product_id"]
        [-noheader]
        [-script | -format=
                          [name:<pretty|script|csv>];
                          [column_separator:"column_sep_string"];
                          [row_separator:"row_sep_string"];
        ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Product name.

- **id**

  Product ID.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_aru_products
      emcli list_aru_products -id="product id"
      emcli list_aru_products -name="product name"
      emcli list_aru_products -id="product id" -noheader
      emcli list_aru_products -id="product id" -noheader -script
      emcli list_aru_products -id="product id" -noheader  -format="name:pretty"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

list_aru_releases

Lists ARU release information.

Format

```
emcli list_aru_releases
        [-name="release_name" | -id="release_id" | -productId="product_id"]
        [-noheader]
        [-script | -format=
                          [name:<pretty|script|csv>];
                          [column_separator:"column_sep_string"];
                          [row_separator:"row_sep_string"];
        ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Release name.

- **id**

  Release ID.

- **productId**

  Product ID.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format=`column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_aru_releases
      emcli list_aru_releases -noheader
      emcli list_aru_releases -name="release_name" -format="name:pretty"
```

```
emcli list_aru_releases -id="release id" -format="name:script"
emcli list_aru_releases -productId="product id" -noheader -format="name:csv"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

list_assoc

Lists associations between the specified source and destination targets.

Format

Standard Mode

```
emcli list_assoc
      -source="target_name:target_type"
      -dest="target_name:target_type"[-subseparator="subseparator:attribute_
             name:character"]
      [-noheader]
      [-script
      [-format="[name:<pretty|script|csv>];[column_separator:
               "column_sep_string"];[row_separator:"row_sep_string"]"]
```

Interactive (Script) Mode

```
list_assoc(
      source="target_name:target_type"
      ,dest="target_name:target_type"
      [,subseparator="subseparator:attribute_name:character"]
      [,noheader=True/False]
      [,script=True/False]
      [,format="[name:<pretty|script|csv>];[column_separator:
      "column_sep_string"];[row_separator:"row_sep_string"]"]
      )

[ ] indicates that the parameter is optional.
```

Parameters

- **source**

  Source target.

- **dest**

  Destination target.

- **subseparator**

  By default, multi-value input attributes use a colon (:) as a subseparator. Specifying this option overrides the default subseparator value.

  Example: `subseparator="<attribute_name=sep_char>"` where *attribute_name* is the name of the attribute for which you want to override the separator character, and *sep_char* is the new subseparator character.

  Example: `separator="att=#"`

- **noheader**

  Displays the output in tabular output without column headers.

- **script**

  Prints the output in a format that can be used in scripting.

- **format**

Specifies how the output is formatted. The default value is "name:pretty", which prints the output table in a readable format not intended to be parsed by scripts. Other format options include:

- format="name:script"
  Sets the default column separator to a tab and the default row separator to a newline in the output. You can override the column and row separator strings with your own values.
- format="name:script;column_separator:<*column_sep_string*>"
  Causes the verb output to be column-separated by <*column_sep_string*>. Rows are separated by the newline character.
- format="name:script;row_separator:<*row_sep_string*>"
  Causes the verb output to be row-separated by <*row_sep_string*>.
- format="name:script;column_separator:<*column_sep_string*>;row_separator:<*row_sep_string*>"
- Causes the verb output to be column-separated by <*column_sep_string*> and row- separated by <*row_sep_string*>.
- format="name:csv"
  Sets the default column separator to a comma and the default row separator to a newline in the output.

Output

Exit Codes

0 indicates that the verb processing was successful.

Non-zero values indicate that the verb processing was unsuccessful.

Example

This example lists all associations between the source target "abc_cluster:cluster" and the destination target "def.oracle.com:host":

```
emcli list_assoc
       -source="abc_cluster:cluster"
       -dest="def.oracle.com:host"
```

list_chargeback_entities

List all of the entities added into Chargeback

Format

```
list_chargeback_entities
```

See Also

add_chargeback_entity
assign_charge_plan
assign_cost_center
list_chargeback_entity_types
list_charge_plans
list_cost_centers
remove_chargeback_entity
unassign_charge_plan
unassign_cost_center

list_chargeback_entity_types

Lists all of the entity types supported by Chargeback.

Format

```
list_chargeback_entity_types
     -[usage_mode]
     -[entity_type="eType"]

[ ] indicates that the parameter is optional
```

Parameters

- **usage_mode**

  Lists all of the entity types supported by Chargeback and the corresponding usage modes.

- **entity_type**

  Lists all of the usage modes supported for the particular entity type "eType".

Examples

Example 1

This example lists all of the entity types supported by Chargeback.

```
list_chargeback_entity_types
```

Output:

```
Entity Type--------------------Entity Type Display Name
----------------------------------------------------------------------
1. oracle_database--------Database Instance
2. host-------------------------Host
```

Example 2

This example lists all of the entity types supported by Chargeback and the corresponding usage modes.

```
list_chargeback_entity_types -usage_mode
```

Output:

```
Entity Type---------------Entity Type Display Name--------------------Usage Mode
----------------------------------------------------------------------
1. oracle_database------------------Database
Instance----------------------dbMetered
2. oracle_database------------------Database
Instance----------------------dbByService
3. oracle_database------------------Database
Instance----------------------cdbBypdb
4. host--------------------------------Host--------------------------hostMetered
```

Example 3

This example lists all of the usage modes supported for the particular entity type.

```
list_chargeback_entity_types -entity_type="oracle_database"
```

Output:

```
Entity Type-------------------Entity Type Display Name----------------Usage Mode
-------------------------------------------------------------------------------
1. oracle_database----------Database Instance------------------------dbMetered
2. oracle_database----------Database Instance------------------------dbByService
3. oracle_database----------Database Instance------------------------cdbBypdb
```

See Also

add_chargeback_entity
assign_charge_plan
assign_cost_center
list_chargeback_entities
list_charge_plans
list_cost_centers
remove_chargeback_entity
unassign_charge_plan
unassign_cost_center

list_charge_item_candidates

Lists the charge items that can be registered to Chargeback.

Format

```
emcli  list_charge_item_candidates
        -target_type=target_type
        -source_data_type=<metric|config|property>
        [-target_name=target_name]
        [-config_name=config_name]
        [-config_data_source=target_name]
        [-all]

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Name of a target type.

- **source_data_type**

  Type of source data. Valid values are metric, config, and property.

- **target_name**

  If specified, metering and charge data are retrieved only for the named target. If you do not specify a valid target name, or if the specified target has not been enabled, then no data is generated. If this parameter is not specified, All targets for the specified target-type are included. Required if source_data_type=config.

- **config_name**

  Name of a configuration. Required if source_data_type=config.

- **config_data_source**

  Data source of the configuration. Required if source_data_type=config.

- **all**

  Applies only when source_date=metric. Displays all items, including out-of-box metrics of target type. Without this option, only user-defined metric extensions, are displayed.

Examples

Example 1

This example lists the metric extensions created for the Oracle Database target type:

```
emcli list_charge_item_candidates
        -target_type="oracle_database"
        -source_data_type="metric"
```

Example 2

This example lists the configuration items of the myCustomCCS configuration for an Oracle Database target named myDatabase:

```
emcli list_charge_item_candidates
```

```
-target_type="oracle_database"
-source_data_type="config"
-target_name="myDatabase"
-config_name="myCustomCCS"
-config_data_source="CCSDataSource"
```

list_charge_plans

Lists the charge plans in Chargeback.

Format

```
list_charge_plans
     [[-entity_type="entity_type" [-all]]
     [-charge_plan="charge_plan_name" [-all]]
     [-all]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_type**

  Entity type for which the charge plans are to be listed.

- **charge_plan**

  Lists details about a specific charge plan.

- all

  Lists all active and future plans.

Examples

Example 1

This example lists all of the charge plans in Chargeback.

```
list_charge_plans
```

Example 2

This example lists all of the active and future charge plans in Chargeback.

```
list_charge_plans -all
```

Example 3

This example lists all of the active charge plans that support the "eType" entity type.

```
list_charge_plans -entity_type="eType"
```

Example 4

This example lists all of the active and future charge plans that support the "eType" entity type.

```
list_charge_plans -entity_type="eType" -all
```

Example 5

This example provides details about the active version of the charge plan.

```
list_charge_plans -charge_plan="chargePlanName"
```

Example 6

This example provides details about the active and future versions of the charge plan.

```
list_charge_plans -charge_plan="chargePlanName" -all
```

See Also

add_chargeback_entity
assign_charge_plan
assign_cost_center
list_chargeback_entities
list_chargeback_entity_types
list_cost_centers
remove_chargeback_entity
unassign_charge_plan
unassign_cost_center

list_cost_centers

Lists the cost centers in various formats depending on the options given.

Format

```
list_cost_centers
     -[[cost_center_name="cName"]
     -[parent]
     -[children]
     -[top]
     -[leaf]]

[ ] indicates that the parameter is optional
```

Parameters

- **cost_center_name**

  Name of the cost center for which further details like parent/children/top/leaf should be listed.

- **parent**

  Provides the parent cost center of the given cost center.

- **children**

  Provides the list of child cost centers of the given cost center.

- **top**

  Provides the hierarchy of the given cost center from the top.

- **leaf**

  Provides the leaf nodes of the given cost center.

Examples

Example 1

This example lists all of the cost centers.

```
list_cost_centers
```

Example 2

This example provides the parent of the given cost centers.

```
list_cost_centers -cost_center_name="c11" -parent
```

Output:

```
Parent Node
-----------------
c1
```

Example 3

This example provides a list of all the child cost centers of the given cost center

```
list_cost_centers -cost_center_name="c1" -children
```

Output:

```
Children Nodes
----------------------
c11
c12
```

Example 4

This example provides the top hierarchy of the given cost center.

```
list_cost_centers -cost_center_name="c111" -top
```

Output:

```
Hierarchy
-------------
[c1]----->[c11]----->[c111]
```

Example 5

This example provides the leaf nodes of the given cost center.

```
list_cost_centers -cost_center_name="c1" -leaf
```

Output:

```
Leaf Nodes
---------------
c111
c112
c12
```

See Also

add_chargeback_entity
assign_charge_plan
assign_cost_center
list_chargeback_entities
list_chargeback_entity_types
list_charge_plans
remove_chargeback_entity
unassign_charge_plan
unassign_cost_center

list_custom_plugin_updates

Lists all of the custom plug-in updates imported to Enterprise Manager to date. Only one custom plug-in update can be imported for each plug-in version and revision combination.

Format

```
emcli list_custom_plugin_updates
```

Example

The following example lists all of the custom plug-in updates imported to Enterprise Manager to date.

```
emcli list_custom_plugin_updates
```

list_database_sizes

Lists all of the database sizes that have been created.

Format

```
emcli list_database_sizes
      [-name="<Existing size name>"]

[ ] indicates that the parameter is optional.
```

Parameters

- name

  A complete or a partial string. If the name parameter is specified, only database sizes that include the specified string are returned.

Examples

Example 1

The following command finds all database sizes that have been created.

```
emcli list_database_sizes
```

Output:

```
Name:Extra-Small
Description:Extra-small database size
CPU(cores):4
Memory(GB):4
Storage(GB):Not Specified
Processes(Units):Not Specified

Name:Small
Description:Small database
CPU(cores):8
Memory(GB):8
Storage(GB):Not Specified
Processes(Units):Not Specified

Name:Medium
Description:Medium
CPU(cores):8
Memory(GB):16
Storage(GB):Not Specified
Processes(Units):Not Specified
```

Example 2

The following command finds all database sizes that include 'Extra' in the name string.

```
emcli list_database_sizes
-name="Extra*"
```

Output:

```
Name:Extra-Small
Description:Extra-small database size
```

```
CPU(cores):4
Memory(GB):4
Storage(GB):Not Specified
Processes(Units):Not Specified
```

Example 3

The following command finds all database sizes that include 'Extra-Small' in the name string.

```
emcli list_database_sizes
-name="Extra-Small*"
```

Output:

```
Name:Extra-Small
Description:Extra-small
CPU(cores):4
Memory(GB):4
Storage(GB):Not Specified
Processes(Units):Not Specified
```

list_dbprofiles

Lists all the database profiles.

Format

```
emcli list_dbprofiles
      [-details]

[ ] indicates that the parameter is optional.
```

Parameters

- details

  Shows the details for each database profile.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example lists all the existing database profiles in detail:

```
emcli list_dbprofiles -details

Output:
    Name=RMAN Profile,Location=Database Provisioning Profiles/11.2.0.4.0/linux_
x64/,Type=RMAN,Status=Ready,Description=Database Reference Profile 04-11-2014
12:40 PM from database.mycompany.com
    Version : 11.2.0.4.0,contains=Structure and
Data,removalOverdue=0,sourceDatabaseName=database.mycompany.com.
    Name=DB Template,Location=Database Provisioning Profiles/11.2.0.4.0/linux_
x64/,Type=DBCA_TEMPLATE,Status=Ready,Description=Database Reference Profile
03-11-2014 04:55 PM from database.mycompany.com
    Version : 11.2.0.4.0,contains=Structure
only,removalOverdue=0,sourceDatabaseName=database.mycompany.com.
    Name=Snapshot Profile,Location=Database Provisioning
Profiles/11.2.0.4.0/linux_x64/,Type=SNAPSHOT,Status=Ready,Description=Database
Reference Profile 05-11-2014 03:09 PM from database.mycompany.com
    Version : 11.2.0.4.0,contains=Structure and
Data,removalOverdue=2,sourceDatabaseName=database.mycompany.com.
```

list_diagchecks

Gets the list of diagnostic check exclusions defined for a target type.

Format

```
emcli list_diagchecks
        -target_type="type"
        [-version="<diag_version>" ]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target.

- **version**

  Diagnostic version. Defaults to the latest version.

list_diagcheck_exclude_applies

Displays the list of targets using a diagcheck exclusion.

Format

```
emcli list_diagcheck_exclude_applies
     -target_type="target type"
     -exclude_name="name"
```

Parameters

- target_type

  The target type.

- exclude_name

  The exclusion name.

list_diagcheck_exclusions

Gets the list of diagnostic check exclusions defined for a target type.

Format

```
emcli list_diagcheck_exclusions
        -target_type="type"
```

Parameters

- **target_type**

  Type of target.

list_fmw_profiles

Lists all available Fusion Middleware provisioning profiles in the software library.

Format

```
emcli list_fmw_profiles
      [-source_type="Profile Source"]

[ ] indicates that the parameter is optional.
```

Parameters

- source_type

  Specify one source type to view only profiles of that type. Valid values are weblogic_domain, oracle_home, or install_media.

Example

The following example displays all available Weblogic domain provisioning profiles in the software library.

```
emcli list_fmw_profiles
      -source_type="weblogic_domain"
```

list_internal_metrics

Lists all available internal metrics in an OMS.

Format

```
emcli list_internal_metrics
        [-oms_name=<specific oms name> ]

[ ] indicates that the parameter is optional.
```

Parameters

- oms_name

  The name of the target OMS . The explicit OMS name can be found in the Cloud Control console Management Services page. To navigate to this page, from the Setup menu, select Manage Cloud Control and then Management Services. In the Servers area, look for the full name of the Management Service (<host name>:<port number>_Management_Service).

  Note: You only need to specify the oms_name option if you are attempting to access a specific OMS in a multi-OMS environment. If you omit the oms_name option, the list_internal_metric verb will access the OMS running the current instance of EMCLI..

Examples

Example 1

The following example generates a list of internal metrics from an Enterprise Manager repository named "myserver.myco.com:17999_Management_Service".

```
emcli list_internal_metrics -oms_name=myserver.myco.com:17999_Management_Service
```

Example 2

The following example generates a list of internal metrics from the OMS currently running EMCLI.

```
emcli list_internal_metrics
```

list_masking_definitions

Gets the list of masking definitions for an associated target and its script status.

Format

```
emcli list_masking_definitions
     [-definition_name=<masking_defn_name_filter>]
     [-adm_name=<application_data_model_filter>]
     [-target_type=<target_type_filter>]
     [-target_name=<target_name_filter>]
     [-string_match]
     [-script | -format=[name:<pretty|script|csv>];
                        [column_separator:"column_sep_string"];
                        [row_separator:"row_sep_string"];
     ]
     [-noheader]

[ ] indicates that the parameter is optional
```

Parameters

- **definition_name**

  Masking definition name filter. This can be either a full value or a pattern match (%).

- **adm_name**

  Application Data Model (ADM) name. This can be either a full value or a pattern match (%).

- **target_type**

  Database target type. This can be either 'oracle_database' or 'rac_database'.

- **target_name**

  Database target name. This can be either a full value or a pattern match (%).

- **string_match**

  Uses an exact string match for a target_name and definition_name match.

- **script**

  This option is equivalent to -format='name: script' .

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format=`column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

- **noheader**

  Suppresses printing of column headers.

Output Columns

Masking Definition, Database, Status

Examples

Example 1

This example lists all masking definitions.

```
emcli list_masking_definitions
```

Example 2

This example lists the masking definition named mask_hr_data.

```
emcli list_masking_definitions -definition_name=mask_hr_data
```

Example 3

This example lists all masking definitions with names starting with credit_card.

```
emcli list_masking_definitions  -definition_name=credit_card%
```

Example 4

This example lists all masking definitions created on a database named testdb.

```
emcli list_masking_definitions -target_name=testdb
```

Example 5

This example lists all masking definitions created on databases with names starting with test.

```
emcli list_masking_definitions -target_name=test%
```

Example 6

This example lists the masking definition named mask_hr_data created on a database named testdb.

```
emcli list_masking_definitions -definition_name=mask_hr_data -target_name=testdb
```

Example 7

This example lists all masking definitions with names starting with credit and created on databases with names starting with test.

```
emcli list_masking_definitions -definition_name=credit% -target_name=test%
```

Example 8

This example lists all masking definitions without printing the column headers.

```
emcli list_masking_definitions -noheader
```

list_named_credentials

Lists the named credentials. You can list the credentials you own or have explicit access to.

Format

```
emcli list_named_credentials
      [-cred_name="cred_name"]
      [-cred_owner="cred_owner"]
      [-script | -format=[name:<pretty|script|csv>];
                         [column_separator:column_sep_string];
                         [row_separator:row_sep_string];
      [-separator="separator:attname:charseq"]
      [-noheader]

[ ] indicates that the parameter is optional
```

Parameters

- **cred_name**

  Credential name to filter the list of credentials displayed.

- **cred_owner**

  Credential owner to filter the list of credentials displayed.

- **script**

  This is equivalent to -format='name: script'.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

- **separator**

  Multi-value attributes use the semi-colon character as the separator. When data contains this character, you can override its value. For example:

  ```
  separator="<attributename=sep_char"
  ```

... where 'attributename' is the name of the attribute for which you want to override the separator character, and 'sepchar' is the new separator character. For example:

```
separator="att=#"
```

- **noheader**

  Suppresses printing of column headers in tabular output.

Exit Codes

0 if successful. A non-zero value means that verb processing was unsuccessful.

Examples

This example lists credentials matching credential names containing 'NC'.

```
emcli list_named_credentials -cred_name="NC"
```

list_oms_config_properties

Lists the OMS configuration properties.

Format

```
emcli list_oms_config_properties
        [-oms_name="omsName"]
        [-details]

[ ] indicates that the parameter is optional
```

Parameters

- **oms_name**

  Name of the OMS from where the properties have to be retrieved.

- **details**

  Displays the details about from where the property value has been derived, and also the global and default values for the property.

Examples

Example 1

This example lists the entire set of properties.

```
list_oms_config_properties
```

Example 2

This example lists all the properties set on the management server myhost:1159_ Management_Service.

```
list_oms_config_properties -oms_name="myhost:1159_Management_Service"
```

list_oms_logging_properties

Lists the logging configuration properties.

Format

```
emcli list_oms_logging_properties
        [-oms_name="omsName"]
        [-details]

[ ] indicates that the parameter is optional
```

Parameters

- **oms_name**

  Name of the OMS from where the logging properties have to be retrieved.

- **details**

  Displays the details about from where the property value has been derived, and also the global and default values for the logging property.

Examples

Example 1

This example lists the entire set of logging properties.

```
list_oms_logging_properties
```

Example 2

This example lists all the logging properties set on the management server myhost:1159_Management_Service.

```
list_oms_logging_properties -oms_name="myhost:1159_Management_Service"
```

list_patch_plans

Lists existing patch plans. You can list all the existing patch plans and can also list the existing patch plans whose names match the specified pattern.

Format

```
emcli list_patch_plans
        [-name="name"]
        [-noheader]
        [-script | -format=
                          [name:<pretty|script|csv>];
                          [column_separator:"column_sep_string"];
                          [row_separator:"row_sep_string"];
        ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Plan name used for searching patch plans. If you do not specify this parameter, the patch plan whose name is the same as the specified name, or contains the specified name string, will be listed. If you do not specify this option, all of the existing patch plans are listed.

- **noheader**

  Suppresses printing of column headers.

- **script**

  This option is equivalent to -format='name: script' .

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_patch_plans
      emcli list_patch_plans -name="plan name" -noheader
      emcli list_patch_plans -name="plan name" -noheader -script
      emcli list_patch_plans -name="plan name" -noheader -format="name:pretty"
      emcli list_patch_plans -name="plan name" -noheader
      -format="name:pretty";column_separator="separator"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

list_patches_in_custom_plugin_update

Lists all of the patches included in the custom plug-in update for a particular plug-in.

Format

```
emcli list_patches_in_custom_plugin_update
      -plugin="<plugin_id>:<plugin_version>:<plugin_revision>"
      [-discovery]

[ ] indicates that the parameter is optional.
```

Parameters

- **plugin**

  ID, version, and revision of the plug-in. To view the version and revision of a plug-in, run the emcli list_custom_plugin_updates verb.

- **discovery**

  Lists even patches with the discovery component of the plug. If not passed, only patches with the monitoring component of the plug-in are listed.

Examples

Example 1

The following example lists all of the patches included in the custom plug-in update of the 12.1.0.2.0 version of the `oracle.sysman.db2` plug-in. The patch list includes patches that contain only the monitoring component of the plug-in.

```
emcli list_patches_in_custom_plugin_update
      -plugin="oracle.sysman.db2:12.1.0.2.0"
```

Example 2

The following example lists all the patches included in the custom plug-in update of the 12.1.0.2.0 version of the `oracle.sysman.db2` plug-in. The patch list includes patches that contain not only the monitoring component but also the discovery component of the plug-in.

```
emcli list_patches_in_custom_plugin_update
      -plugin="oracle.sysman.db2:12.1.0.2.0"
      -discovery
```

list_plugins_on_agent

Lists all of the plug-ins deployed on the management Agents.

Format

```
emcli list_plugins_on_agent
           [-agent_names="agent1,agent2,agent3 "
           [-all]
           [-include_discovery]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_names**

  All of the management Agents(host:port) whose deployed plug-ins need to be listed. If you do not provide any Agent names, plug-ins on all Agents are listed. String literals with a wildcard (*) expression are accepted. For example:

  ```
  emcli list_plugins_on_agent -agent_names='adc*,st*93'
  ```

- **all**

  Lists plug-ins on all the management's Agents.

- **include_discovery**

  Includes discovery components of the plug-ins. By default, discovery components of the plug-ins are ignored.

Examples

Example 1

This example lists plug-ins on the Agent abc.example.com.

```
emcli list_plugins_on_agent -agent_names=abc.example.com:3872
```

Example 2

This example lists plug-ins for both of the Agents as well as their discovery components.

```
emcli list_plugins_on_agent -agent_names=
abcd.example.com:3872,efgh.example.com:3872 -include_discovery
```

Example 3

This example lists plug-ins for all Agents with the name that matches one of the regular expressions adc* or st*93.

```
emcli list_plugins_on_agent -agent_names='adc*,st*93'
```

Example 4

This example lists plug-ins for all of the management Agents.

```
emcli list_plugins_on_agent -all
```

list_plugins_on_server

Lists all of the plug-ins that are deployed on the OMS instances.

Format

```
emcli list_plugins_on_server
      [-details]

[ ] indicates that the parameter is optional.
```

Parameters

- **details**

  Displays the plug-in home location.

Examples

Example 1

The following example lists all the plug-ins that are deployed on the OMS instances.

```
emcli list_plugins_on_server
```

Example 2

The following example lists all of the plug-ins, with their plug-in home locations, which are deployed on the OMS instances.

```
 emcli list_plugins_on_server
      -details
```

list_prerequisites

Displays a list of Enterprise Manager repository-related prerequisites.

Format

```
emcli list_prerequisites
      -db_user=<database_user>
      -db_password=<database_password>
      -db_role=<database_role>
      -repos_user=<repository_user>
      [-prerequisite_xml_root_dir=<xml_root_directory_for_platform_prerequisites>]
      [-prerequisite_resource_locs="<xml_resource_location_for_platform/
      plug-in_prerequisites>"]
      [-log_loc=<location_for_log_files_of_EMPrereqKit_tool>]
      [-upgrade_version=<EM_version_to_which_upgrade_is_being>]
      [-configuration_type=<configuration/deployment_type>]

[ ] indicates that the parameter is optional.
```

Parameters

- db_user

  Database user account with which a connection to the database can be established, for example SYS.

- db_password

  Database user account password. If you do not provide here, you will be prompted for the password.

- db_role

  Database role. For example, sysdba. Required only when the -db_user value is SYS.

- repos_user

  Repository user account with which the prerequisite checks can be run, for example, SYSMAN. Required only when the -db_user value is SYS.

- prerequisite_xml_root_dir

  Absolute path to the `requisites/list` directory where of the all prerequisite XMLs are located. This is an optional parameter and if not provided, the value is calculated internally. The XML files can be in a subdirectory within the `requisites/list` directory, but make sure the path that you enter leads only up to the `list` directory. For example, `$<OMS_HOME>/install/requisites/list`.

- prerequisite_resource_locs

  Absolute path to the directory where the plug-in opar files or the platform/plug-in binaries, which contains XML files for platform or plug-in prerequisite checks, are located. This is an optional parameter. For plug-in opar files, use the format `plugin_id=<<absolute_path_.opar_file>>`. For the plug-in home directory use the format `plugin_id=<<plugin_home>>`.

- log_loc

  Absolute path to a directory where the logs of the execution of the Enterprise Manager prerequisite kit can be stored.

- upgrade_version

  The Enterprise Manager version to which the upgrade is being done. For example, 12.1.0.3. If you have downloaded the Enterprise Manager prerequisite resources for two future versions, for example v1 and v2 through Self-Update then with -upgrade_version, you can see or run the prerequisite of the specified version.

- configuration_type

  Configuration or deployment type. For example, MINI, SMALL, MEDIUM, LARGE. This is an optional parameter, and if not provided, it will be calculated internally.

Examples

Example 1

Displays a list of Enterprise Manager repository-related prerequisites with the configuration type MEDIUM.

```
emcli list_prerequisites
      -db_user=SYS
      -db_password=pwd
      -db_role=sysdba
      -repos_user=SYSMAN
      -prerequisite_xml_root_dir=$ORACLE_HOME/install/requisites/list
      -configuration_type=MEDIUM
```

Example 2

Displays a list of Enterprise Manager repository-related prerequisites with upgrade version 12.1.0.4.

```
emcli list_prerequisites
      -db_user=SYS
      -db_password=pwd
      -db_role=sysdba
      -repos_user=SYSMAN
      -prerequisite_xml_root_dir=$ORACLE_HOME/install/requisites/list
      -upgrade_version=12.1.0.4.0
```

Example 3

Displays a list of Enterprise Manager repository-related prerequisites with the prerequisite resource location `oracle.sysman.db=<<MW_HOME>>/plugins/oracle.sysman.db.oms.plugin_x.x.x.x.x,oracle.sysman.emas=<<Absolute directory path>>/x.x.x.x.x_oracle.sysman.emas_2000_0.opar'`.

```
emcli list_prerequisites
     -db_user=SYS
     -db_password=pwd
     -db_role=sysdba
     -repos_user=SYSMAN
     -prerequisite_resource_locs="oracle.sysman.db=
     <<MW_HOME>>/plugins/oracle.sysman.db.oms.plugin_x.x.x.x.x,
     oracle.sysman.emas=<<Absolute directory path>>/
     x.x.x.x.x_oracle.sysman.emas_2000_0.opar"
```

list_privilege_delegation_settings

Lists privilege delegation setting templates available on the server that apply to targets.

Format

```
emcli list_privilege_delegation_settings
       [-setting_type="SUDO/POWERBROKER]"
       [-noheader]
       [-script | -format=
                [name:<pretty|script|csv>];
                [column_separator:"column_sep_string"];
                [row_separator:"row_sep_string"];
       ]

[ ] indicates that the parameter is optional
```

Parameters

- **setting_type**

  Setting type. All applicable settings are displayed if you do not specify this option.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_privilege_delegation_settings
  -setting_type="SUDO"
```

list_siebel_enterprises

Lists the Siebel enterprises currently discovered in Enterprise Manager.

Format

```
eemcli list_siebel_enterprises
```

Example

This example lists the Siebel enterprises that are discovered in Enterprise Manager.

```
emcli list_siebel_enterprises
```

For example, the listed Siebel enterprises that are displayed are:

```
siebel_enterprise: siebel_slc01nqr.us.example.com
siebel_enterprise: siebel_slc01qhn.us.example.com
```

list_siebel_servers

Lists the Siebel servers present in the specified Siebel enterprise.

Format

```
emcli list_siebel_servers -enterprise=<Siebel enterprise>
```

Parameters

- **enterprise**

  Indicates the fully-qualified name of the Siebel enterprise.

  For example, to list servers under a Siebel enterprise `<Seibel enterprise>`, enter the option as: `-enterprise=<Siebel enterprise>`.

  Note: The command `emcli list_siebel_enterprises` can be used to list the currently monitored Siebel enterprises in EM.

Example

This example lists the Siebel servers present in the `siebel_slc01nqr.us.example.com` Siebel enterprise in Enterprise Manager.

```
emcli list_siebel_servers -enterprise=siebel_slc01nqr.us.example.com
```

list_sla

Lists the SLA life-cycle status and version information for a target. If you specify the slaName, the command prints the summary information of the different versions. If you do not specify the slaName, the command prints all the available SLA version series for a target. When you specify the version, this commands prints only summary information for the specified version.

Format

```
emcli list_sla
    -targetName=<target_name>
    -targetType=<target_type>
    [-slaName=<SLA_name>]

[ ] indicates that the parameter is optional
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **slaName**

  Name of the SLA.

Examples

Example 1

This example prints the SLA information for one SLA.

```
emcli list_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla' -version=2
```

Example 3

This example prints the SLA information for all SLAs of a target.

```
emcli list_sla
      -targetName='my_service' -targetType='generic_service'
```

list_subset_definitions

Gets the list of subset definitions, Application Data Models, and target names.

Format

```
emcli list_subset_definitions
      [-subset_name=<subset_definition_name_filter>]
      [-adm_name=<application_data_model_filter>]
      [-target_name=<target_name_filter>]
      [-string_match]
      [-script | -format=[name:<pretty|script|csv>];
                     [column_separator:"column_sep_string"];
                     [row_separator:"row_sep_string"];
      ]
      [-noheader]

[ ] indicates that the parameter is optional
```

Parameters

- **subset_name**

  Filter for the subset definition name. This can either be a full value or a pattern match(%).

- **adm_name**

  Filter for the Application Data Model (ADM) name. This can be either a full value or a pattern match(%).

- **target_name**

  Filter for the database target name. This can be either a full value or a pattern match (%).

- **string_match**

  Uses an exact string match for the subset definition name, target name, and ADM name.

- **script**

  This option is equivalent to -format='name: script'.

- **format**

  Format specification (default is -format="name: pretty").

  – -format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  – -format="name:script" sets the default column separator to a tab and the default row separator to a newline.

  – -format="name:csv" sets the column separator to a comma and the row separator to a newline.

- **noheader**

  Suppresses the printing of column headers.

Output Columns

Subset Definition, Adm Name, Target Name

Examples

Example 1

This example prints the SLA information for one SLA.

```
emcli list_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla' -version=2
```

Example 3

This example prints the SLA information for all SLAs of a target.

```
emcli list_sla
      -targetName='my_service' -targetType='generic_service'
```

list_swlib_entities

Lists the entities in the software library based on the specified filter criteria. The results are printed in the following order:

Display Name, Revision, Description, Status, Type, Subtype, Maturity, Owner, [Folder Path, Folder Id, Entity Rev Id]

Format

```
emcli list_swlib_entities
        [-name="entity_name"]
        [-folder_id="folder_internal_id"]
        [-desc="entity_desc"]
        [-attr="<attr_name>:<attr_value>"]
        [-type]="type_internal_id"]
        [-subtype]="subtype_internal_id"]
        [-maturity]="maturity"]
        [-owner]="owner"]
        [-status]="status"]
        [-show_folder_path]
        [-show_folder_id]
        [-show_entity_rev_id]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the entity. The value specified for this option is considered for a case-insensitive match.

- **folder_id**

  Internal identifier of the parent folder. The value specified for this option is considered for an exact match.

- **desc**

  Description of the entity. The value specified for this option is considered for a case-insensitive match.

- **attr**

  An attribute and its value, separated by a colon (:). For specifying values for multiple attributes, repeat the option. The value specified for this option is considered for an exact match.

  You can only use this parameter with the type parameter.

- **type**

  Internal identifier of the entity type. Use the list_swlib_entity_types verb to identify the type.

- **subtype**

  Internal identifier of the entity sub-type. Use the list_swlib_entity_subtypes verb to identify the sub-type.

- maturity

  Maturity of the entity revision. Can be one of:

MAT_Untested

MAT_Beta

MAT_Production

- owner

  Owner of the entity revision.

- status

  Status of the entity revision. Can be one of:

  STATE_Incomplete

  STATE_Ready

  STATE_Deleted

- **show_folder_path**

  Enables printing of the internal path of each entity's folder.

- **show_folder_id**

  Enables printing of the internal ID of each entity's folder. If specified, the value is printed after the value for show_folder_path.

- **show_entity_rev_id**

  Enables printing of the internal ID of each entity. If specified, the value is printed after the value for show_folder_id.

Examples

This example lists all folders under the specified parent folder, and also prints the internal identifier for each folder in the list.

```
emcli list_swlib_entities
      -name="myEntity"
      -type="COMP_Component"
      -attr="PRODUCT:Oracle Database"
      -show_folder_id
```

list_swlib_entity_subtypes

Lists the entity subtypes available in the software library for a specified entity type.

Format

```
emcli list_swlib_entity_subtypes
        [-entity_type_id="type_internal_name"]
        [-show_subtype_id]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_type_id**

  Internal identifier of the type.

- **show_subtype_id**

  Enables printing of the internal identifier for the subtype.

Examples

This example lists all subtypes available in the software library for the type 'COMP_ Component.'

```
emcli list_swlib_entity_subtypes
        -entity_type_id="COMP_Component"
        -show_subtype_id
```

list_swlib_entity_types

Lists the entity types available in the software library.

Format

```
emcli list_swlib_entity_types
        [-show_type_id]

[ ] indicates that the parameter is optional
```

Parameters

- **show_type_id**

  Enables printing of the internal identifier for the type.

Examples

This example lists all of the types available in the software library.

```
emcli list_swlib_entity_types
        -show_type_id
```

list_swlib_folders

Lists folders in the software library.

Format

```
emcli list_swlib_folders
        [-parent_id="parent_folder_id"]
        [-show_folder_path]
        [-show_folder_id]

[ ] indicates that the parameter is optional
```

Parameters

- **parent_id**

  Internal identifier of the parent folder.

- **show_folder_path**

  Enables printing of the internal path for the folder.

- **show_folder_id**

  Enables printing of the internal identifier for the folder.

Examples

This example lists all folders under the specified parent folder, and prints the internal identifier for each folder in the list.

```
emcli list_swlib_folders
      -parent_id=
"oracle:defaultService:em:provisioning:1:cat:B13B3B7B086458CFE040E80A19AA560C"
      -show_folder_id
```

list_swlib_storage_locations

Lists storage locations configured in the software library.

Format

```
emcli list_swlib_storage_locations
        [-type="OmsShared|OmsAgent|Http|Nfs|ExtAgent"]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Type of the storage location. The default is OmsShared.

Examples

This example lists all locations configured for storage type 'OmsAgent.'

```
emcli +_locations
        -type="OmsAgent"
```

list_target_privilege_delegation_settings

Lists current privilege delegation settings for targets.

Format

```
emcli list_target_privilege_delegation_settings
      -target_names="name1;name2;name3"
      [-input_file="FILE:file_path"]
      [-noheader]
      [-script | -format=
               [name:<pretty|script|csv>];
               [column_separator:"column_sep_string"];
               [row_separator:"row_sep_string"];
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **target_names**

  List of targets. All targets must be of the host type. Either target_names or input_file must be present.

- **input_file**

  Path of the file that has the list of targets. The file should have one target name per line.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **noheader**

  Display tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_target_privilege_delegation_settings
```

```
          -target_names="host.example.com;host2.example.com;

emcli list_target_privilege_delegation_settings
          -input_file="FILE:/home/nqureshi/targets.txt"

emcli list_target_privilege_delegation_settings
          -target_names="host.example.com;host2.example.com;
```

list_target_property_names

Lists property names for the global properties.

Format

```
emcli list_target_property_names
```

Parameters

None.

list_templates

Lists monitoring templates and their display names.

Format

```
emcli list_templates
        [-target_type="target_type"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Template's target type. If specified, all templates defined for this target type are displayed.

Examples

Example 1

This example lists all templates.

```
emcli list_templates
```

Exmaple 2

This example lists all templates defined for the host target type.

```
emcli list_templates -target_type="host"
```

list_trace

Displays the list of OMS traces for the Oracle Management System.

Format

```
emcli list_trace
```

Parameters

None.

list_unconverted_udms

Retrieves the list of UDMs that are not yet in a migration session.

Format

```
emcli list_unconverted_udms
      [-templates_only]

[ ] indicates that the parameter is optional
```

Parameters

- **templates_only**

  Only lists unconverted UDMs in templates.

Examples

Example 1

This example displays all the UDMs that are not part of a migration session.

```
emcli list_unconverted_udms
```

Exmaple 2

This example displays all the UDMs that are in a template and not part of a migration session.

```
emcli list_unconverted_udms -templates_only
```

login

Logs into Enterprise Manager with the given credentials and sets up a session with the OMS.

Note: To avoid an uncommon occurrence in which multiple emcli sessions are created on the OMS, Oracle recommends that you enter the login command before running a script containing EM CLI commands.

Tip: See also logout on page 5-583.

Command-Line Format

```
emcli login
      -username=<EM_Console_Username>
      [-password=<EM_Console_Password>]
      [-force]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
login
      (username="<EM_Console_Username>"
      [,password="<EM_Console_Password>"]
      [,force=True|False])

[ ] indicates that the parameter is optional
```

Parameters

- **username**

  Enterprise Manager user name to be used by all subsequent EM CLI commands when contacting the OMS.

- **password**

  Enterprise Manager user password. If you do not specify this , you are prompted for the password interactively.

  Note: Providing a password on the command line is insecure and should be avoided.

- **force**

  Force a login even if there is an existing session. The value must be set to either True or False for scripting and interactive format.

Examples

These examples show a login as a different user using newly specified credentials, then a subsequent login using the previous credentials.

Example 1 - Command-Line

```
emcli logout
emcli login -user=new_user -pass=new_user_pass
emcli <verb-name>
emcli logout
emcli login -user=old_user -pass=old_user_pass
```

Example 2 - Scripting and Interactive

```
logout()
login(username="new_user", password="new_user_pass")
<verb-name>
logout()
login(username="old_user", password="old_user_pass")
```

logout

Terminates the existing session with the OMS. This verb and the login verb are useful when you need to run a particular verb as a different user. After a logout, you need to invoke either the setup verb or login verb before invoking any other emcli verb.

Tip: See also login on page 5-581.

Note: Verbs executed after 'emcli logout' may fail with the message "Error: Session expired. Run emcli login to establish a session." You need to run the login verb to log in to EM CLI after an 'emcli logout'.

Format

```
emcli logout
```

Parameters

None.

Examples

This example shows a login as a different user using newly specified credentials, then a subsequent login using the previous credentials.

```
emcli logut
emcli login -user=new_user -pass=new_user_pass
emcli <verb-name>
emcli logout
emcli login -user=old_user -pass=old_user_pass
```

manage_agent_partnership

Overrides Enterprise Manager's default behavior of automatically assigning partner agents to agents. A partner agent is an agent that, in addition to its other functions, is assigned to another agent as its "partner" in order to remotely monitor the availability of that agent and its host. A partner agent is typically in close network proximity, for example, in the same subnet, with the agent that it remotely monitors. An agent can be a partner (remote monitor) of multiple agents. An agent can only have one partner agent assigned to it.

This verb is not meant to be commonly used. It is provided to support special circumstances where an administrator might want to explicitly assign agent partnerships or exclude agents from being partners or exclude agents from being remotely monitored by other agents.

Format

Standard Mode

```
emcli manage_agent_partnership
      [-add_agent_partnership]
      [-remove_agent_partnership]
      [-enable_agent_partnership]
      [-disable_agent_partnership]
      [-partner_agent="partneragent"]
      [-monitored_agent="monitoredagent"]
```

Interactive (Script) Mode

```
manage_agent_partnership(
      [add_agent_partnership=True/False]
      [,remove_agent_partnership=True/False]
      [,enable_agent_partnership=True/False]
      [,disable_agent_partnership=True/False]
      [,partner_agent="partneragent"]
      [,monitored_agent="monitoredagent"]
      )

[ ] indicates that the parameter is optional.
```

Parameters

- **add_agent_partnership**

  Assigns a partner agent to an agent. You must also specify the monitored_agent and partner_agent parameters.

- **remove_agent_partnership**

  Removes the partnership between a partner agent and the agent that it monitors. For a remotely monitored agent, to remove the relationship between itself and its partner agent, the monitored_agent parameter must be specified. The partner_ agent can be optionally specified. For a partner agent to remove the relationships between itself and all agents that it remotely monitors, the partner_agent parameter must be specified. If the monitored_agent parameter is not specified, then all partnerships that the partner agent currently has will be deleted.

- **disable_agent_partnership**

Prevents an agent from being a partner agent or from being a monitored agent depending on the additional parameters used. If the partner_agent parameter is used, then it prevents the specified agent from being a partner agent (remotely monitoring other agents). If the monitored_agent parameter is used, it prevents the specified agent from being remotely monitored by any agent.

- **enable_agent_partnership**

  Enables an agent to become a partner agent or a monitored agent based on the additional parameters used. If the partner_agent parameter is used, it enables the specified agent to be a partner agent (remotely monitor other agents). If the monitored_agent parameter is used, it enables the specified agent to be remotely monitored by another agent. Only one of these actions can be specified.

- **monitored_agent**

  The name of the agent that is remotely monitored by another agent. It is typically in the form *host:port*, for example m`yhost.example.com:1830`.

- **partner_agent**

  The name of the agent that will remotely monitor the availability of another agent and its host. It is typically in the form *host:port*, for example m`yhost.example.com:1830`.

Output

Exit Codes

0 indicates that the verb processing was successful.

Non-zero values indicate that the verb processing was not successful.

Examples

Example 1

This example assigns agent2 as the partner agent for agent1:

```
emcli manage_agent_partnership
      -add_agent_partnership
      -monitored_agent=agent1.example.com:1830
      -partner_agent=agent2.example.com:1833
```

Example 2

This example unassigns agent2 as the partner agent for agent1. If agent1 does not have a partner agent, then an exception is thrown.

```
emcli manage_agent_partnership
      -remove_agent_partnership
      -monitored_agent=agent1.example.com:1830
      -partner_agent=agent2.example.com:1833
```

Example 3

This example unassigns agent2 as the partner agent for all the agents that it remotely monitors. If agent1 is not a partner agent for any agent, then an exception is thrown.

```
emcli manage_agent_partnership
      -remove_agent_partnership
      -partner_agent=agent2.example.com:1833
```

Example 4

This example prevents agent3 from being assigned a partner agent. This means agent3 cannot be remotely monitored by another agent.

```
emcli manage_agent_partnership
      -disable_agent_partnership
      -monitored_agent=agent3.example.com:1830
```

Example 5

This example prevents agent4 from being a partner agent for any agent. This means agent4 cannot be used to remotely monitor other agents.

```
emcli manage_agent_partnership
      -disable_agent_partnership
      -partner_agent=agent4.example.com:1833
```

Example 6

This example allows agent3 to be assigned a partner agent to remotely monitor it.

```
emcli manage_agent_partnership
      -enable_agent_partnership
      -monitored_agent= agent3.example.com:1830
```

Example 7

This example allows agent4 to become a partner for other agents. This means agent4 can be used to remotely monitor other agents.

```
emcli manage_agent_partnership
      -enable_agent_partnership
      -partner_agent= agent4.example.com:1833
```

merge_credentials

Merges all the references of named credentials provided in the source_credential_list into the destination_credential. The verb expects all the named credentials provided to be equivalent. You can list equivalently named credentials using the command emcli get_duplicate_credentials. All the matching duplicate credentials can be merged using the flag merge_all.

Format

```
emcli merge_credentials
        -destination_credential="destination_cred_name[:destination_cred_owner]"
        [-source_credential_list="source_credential_list"]
        [-merge_all]
        [-merge_without_testing]

[ ] indicates that the parameter is optional.
```

Parameters

- **destination_credential**

  Destination credentials to merge the references.

- **source_credential_list**

  Source-named credential list.

- **merge_all**

  Finds all the duplicate credentials and merges.

- **merge_without_testing**

  Merges the credentials without testing the destination credential.

Examples

Example 1

This example merges the named credentials MyOracleCredential2 and MyOracleCredential3 into MyOracleCredential1. If MyOracleCredential1 is equivalent to MyOracleCredential2 and MyOracleCredential3, all the usages of MyOracleCredential2 and MyOracleCredential3 are replaced with MyOracleCredential1.

```
emcli merge_credentials
        -destination_credential="MyOracleCredential1:ADMIN1"
        -source_credential_list=
          "MyOracleCredential2:ADMIN1;MyOracleCredential3:ADMIN3"
```

Example 2

This example finds all the named credentials equivalent to MyOracleCredential1 and merges their usages with MyOracleCredential1.

```
emcli merge_credentials
        -destination_credential=MyOracleCredential1
        -merge_all
```

metric_control

For the specified target type, lists the metrics whose alerts are stateless and therefore can be manually cleared. Both the metric name and metric internal name are provided in the output of this command. To clear the stateless alerts associated with the specified metric, use the clear_stateless_alerts verb.

Tip: See also clear_stateless_alerts on page 5-82.

Format

```
emcli metric_control
         -command=command
         -target_type=type
         -metric_name=name

[ ] indicates that the parameter is optional
```

Parameters

- **command**

  Can be one of the following:

  - disable_metric — Disables loading of the specified metric .
  - enable_metric — Reenables loading of the specified metric.
  - list_disabled_metrics — Lists the metrics currently disabled for loading.
  - flush_metadata_cache — Flushes the metric API metadata cache target_type.

- **target_type**

  Internal target type identifier (host, oracle_database, oc4j, oracle_emrep, oracle).

- **metric_name**

  Internal name of the metric (for example, load for the host target type).

Example

This example disables the loading of the Load metric on the host target type.

```
emcli metric_control -command=disable_metric -target_type=host -metric_name=Load
```

migrate_noncdb_to_pdb

Migrates a non-container database (non-CDB) as a PDB.

Format

```
emcli migrate_noncdb_to_pdb
         -cdbTargetName="CDB_target_name"
         -cdbTargetType="oracle_database | rac_database"
         -cdbDBCreds="CDB_credentials"
         -cdbHostCreds="CDB_host_credentials"
         -migrationMethod="DATAPUMP | PLUG_AS_PDB"
         -noncdbTargetName="migrate_target"
         -noncdbTargetType="oracle_database | rac_database"
         -noncdbDBCreds="migrate_target_credentials"
         -noncdbHostCreds="migrate_target_host_credentials"
         -pdbName="PDB_name"
         -pdbAdminName="PDB_admin_name"
         -pdbAdminPassword="PDB_admin_password"
         [-exportDir="temporary_file_system_location_for_export"]
         [-importDir="temporary_file_system_location_for_import"]
         [-useOMF="Y | N"]
         [-createAsClone="Y | N"]
         [-dataFilesLoc="datafile_location"]
         [-encryptionPwd="encryption_password"]
         [-cdbWalletPwd="wallet_password"]
         [-objectExistsAction="SKIP | REPLACE"]
         [-precheck="YES | NO | ONLY"]
         [-ignoreWarnings="Y | N"]

[ ] indicates that the parameter is optional.
```

Parameters

- cdbTargetName

  Specifies the target container database (CDB) to which the selected database will be migrated as a PDB. The target CDB must be a valid target in Enterprise Manager.

- cdbTargetType

  Specifies the type of the target CDB.

- cdbDBCreds

  Specifies the credentials for the CDB to which the selected database will be migrated as a PDB.

- cdbHostCreds

  Specifies the credentials for the host on which the target CDB resides.

- migrationMethod

  Specifies the method that will be used to migrate the selected database into the CDB. This parameter can accept the following values:

  – DATAPUMP: Uses Oracle Data Pump Full Transportable Export and Import to export data from the selected non-CDB and import data into a newly created PDB. This option is supported for non-CDBs of version 11.2.0.3 or higher.

  - PLUG_AS_PDB: Uses the DBMS_PDB package to generate an XML metadata file. The XML metadata file describes the database files of the non-CDB that are used to plug the non-CDB into a CDB. To use this option, the non-CDB must be a 12c Oracle Database.

- noncdbTargetName

  Specifies the name of the non-CDB that you want to migrate as a PDB.

- noncdbTargetType

  Specifies the target type of the non-CDB that you want to migrate as a PDB.

- noncdbDBCreds

  Specifies the credentials of the non-CDB that you want to migrate as a PDB.

- noncdbHostCreds

  Specifies the credentials of the host on which the non-CDB that you want to migrate resides.

- pdbName

  Specifies the name of the PDB that you want to create.

- pdbAdminName

  Specifies the user name of the PDB administrator that will be created for the new PDB.

- pdbAdminPassword

  Specifies the password of the PDB administrator that will be created for the new PDB.

- exportDir

  Specifies the file system location on the non-CDB host where the exported datapump files (dump and data files) will be stored. This location will be cleaned up after a successful migration. The default export directory is the location pointed to by the DATA_PUMP_DIR directory object on the non-CDB. Ensure that you specify a location on which the non-CDB Oracle home owner has read and write permissions.

- importDir

  Specifies the file system location on the CDB host that will be used to temporarily stage the migration metadata and/or datafiles. This location will be cleaned up after a successful migration. The default import directory is the location pointed to by the DATA_PUMP_DIR directory object on the CDB. Ensure that you specify a location on which the CDB Oracle home owner has read and write permissions.

- useOMF

  Uses the OMF location as the datafile location. This parameter accepts the following values:

  Y: Accepted only if the CDB uses OMF. Ignored otherwise.

  N (Default)

- createAsClone

  Specifies whether the new PDB must be created as a clone. It is accepted only if -migrationMethod is specified as PLUG_AS_PDB. This parameter accepts the following values:

Y

N (default)

- dataFilesLoc

  Specifies the file system location on the CDB host where the datafiles for the newly created PDB will be stored. If the CDB uses ASM, then a disk group name can also be specified as the datafile location. The default datafile location is the location pointed to by the DATA_FILE_DIR directory object on the CDB. Ensure that you specify a location on which the CDB Oracle home owner has read and write permissions. This parameter is ignored if -useOMF is Y.

- encryptionPwd

  Specifies the password to decrypt/encrypt the datapump dump files. This parameter is mandatory if the non-CDB contains encrypted tablespaces.

- cdbWalletPwd

  Specifies the wallet password to open the wallet on the CDB. This parameter is mandatory if the non-CDB contains encrypted tablespaces.

- objectExistsAction

  Specifies the action that must be taken if the exported object having the same name is found on the target CDB. This parameter accepts the following values:

  – SKIP: (Default)

  – REPLACE

- precheck

  Specifies whether to run pre-requisite checks during the migration job. This parameter accepts the following values:

  – YES (Default): Runs pre-requisite checks and proceeds to database migration if there are no errors during the pre-requisite checks.

  – NO: Proceeds to the database migration directly. Does not run the pre-requisite checks.

  – ONLY: Runs pre-requisite checks only. Does not migrate the database.

- ignoreWarnings

  Specifies whether to ignore the warnings (if any) during the pre-requisite checks, and proceed with the migration. This parameter is used only when -precheck is set to YES. It is ignored otherwise. This parameter accepts the following values:

  – Y (Default): Ignores the warnings and proceeds with the migration.

  – N: Does not proceed with the migration if warnings are found.

[ ] indicates that the parameter is optional.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example migrates the non-CDB target NON_CDB_1 as a PDB named NEW_PDB, using the datapump method, the non-CDB target credentials NON_CDB_DB_CREDS, and the non-CDB host credentials NON_CDB_HOST_CREDS, specifying

the administrator user name of the newly created PDB as pdbAdmin, and the administrator password as welcome:

```
emcli migrate_noncdb_to_pdb -migrationMethod=datapump -noncdbTargetName=NON_CDB_1
-noncdbTargetType=oracle_database -noncdbHostCreds=NON_CDB_HOST_CREDS
-noncdbDBCreds=NON_CDB_DB_CREDS -cdbTargetName=CDB_NAME -cdbTargetType=oracle_
database -cdbHostCreds=CDB_HOST_CREDS -cdbDBCreds=CDB_DB_CREDS -pdbName=NEW_PDB
-pdbAdminName=pdbAdmin -pdbAdminPassword=welcome
```

migrate_to_lifecycle_status

Migrates to the lifecycle state from the deployment type.

Format

```
emcli migrate_to_lifecycle_status
        -deployment_values="value1;value2;value3
        -lifecycle_stage_values="Stage;Stage;Production
```

Parameters

- **deployment_values**

  Deployment type values.

- **lifecycle_stage_values**

  Lifecycle stage values

modify_aggregate_service

Modifies an aggregate service instance.

Format

```
emcli modify_aggregate_service
      -name="name"
      -type="type"
      [-add_sub_services="name1:type1;name2:type2;..."]
      [-del_sub_services="name1:type1;name2:type2;..."]
      [-avail_eval_func="function_to_evaluate_availability."]
      [-timezone_region="timezone_region"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Aggregate service name.

- **type**

  Aggregate service type.

- **add_sub_services**

  Sub-services to be added.

- **del_sub_services**

  Sub-services to be deleted.

- **avail_eval_func**

  PL/SQL function to evaluate the availability of the aggregate service. Use [or | and] for the predefined evaluation helper function.

- **timezone_region**

  Time zone region of the service.

Examples

```
emcli modify_aggregate_service -name="My_Name"
      -type="aggregate_service"
      -add_sub_services="sub1:type1;sub2:type2"
      -del_sub_services="sub3:type3"
      -avail_eval_func="my_pkg.my_eval_func"
      -timezone_region="CST"
```

modify_collection_schedule

Modifies the collection schedule of a collection setup for metrics and policies for the specified set of targets. Combining all the metrics, running a script, and collecting the data is referred to as a collection. The collection has various attributes associated with it, such as the collection schedule, upload frequency, and so forth.

Format

```
emcli modify_collection_schedule
      -targetType=ttype
      -targetNames=tname1;tname2;tname3...
      -collectionName=collname
      [-collectionStatus=Enabled or Disabled]
      [-freqType={Minute}{Hour}{Day}{Week}{Weekly}{Month}
      [-freqValue={any integer value for Minute/Hour/Day/Week}{One or more from
       Mon...Sun for Weekly}{One or more from 1;2..31 or Last for Month}
      [-preview=Y or N]

[ ] indicates that the parameter is optional
{ } indicates that you can select one of the s in the series shown

Note: All of the parameters and choices are case-insensitive
```

Parameters

- **targetType**

  You must specify a single target type value, and it should be the same as specified in the repository.

  > **Note:** Only individual target types are currently supported.

- **targetNames**

  The target name should be the same as exists in the repository. All of the targets should be the same target type you specified in the targetType parameter. Use a semicolon (;) to separate the names. Changes to the collection schedule will be executed for only valid target name and target type combinations. For example:

  host1;host2;host3

- **collectionName**

  The collection name should be exactly the same as exists in the repository or the corresponding collections .xml file present on the Management Agent.

  Access files from the following locations to determine the collection to be modified. Select the desired collection and provide it as input to the EM CLI utility.

  – $AGENT_HOME/sysman/admin/metadata/<targetType>.xml

    This file is shipped as a part of the setup and contains information regarding the metrics for this target type.

  – $AGENT_HOME/sysman/admin/default_collection/<targetType>.xml

This file is shipped as a part of the setup and contains the collections shipped by default.

– $AGENT_HOME/sysman/emd/collection/<targetType_targetName>.xml

Whenever changes have occurred for any particular target, this file is automatically generated. Collections for user-defined metrics are available in this file.

- **collectionStatus**

  Enables or disables the collection. The default is Enabled. If Disabled, freqType and freqValue are ignored.

- **freqType**

  You can specify one of the following values:

  Minute (default)
  Hour
  Day
  Week
  Weekly
  Month

  For Week, you must specify an integer value as the frequency value. For instance, if you specify freqType='WEEK' and freqValue='2', the collection occurs every two weeks.

  For Weekly, the possible values are Mon, Tue, Wed, Thu, Fri, Sat, Sun. For instance, if you specify freqType='Weekly' and freqValue='Tue;Thu;Sun', the collection occurs every Tuesday, Thursday, and Sunday of a week.

  The schedule is modified based on your selection. You do not need to specify a value (and the value will be ignored) if the collectionStatus parameter is set to Disabled.

  If you use this parameter, you must also use the freqValue parameter.

- **freqValue**

  You can specify one of the following values:

  – You must specify an integer value if the freqType is any one of Minute, Hour, Day, or Week. The default value is 5.

  – For Weekly, specify one or more choices from Mon, Tue, Wed, Thu, Fri, Sat, and Sun. If the collection occurs on any particular day(s) of the week, you must specify the corresponding value(s) against the Weekly option.

  – For Monthly, specify one or more choices from 1...31 or Last. If the collection occurs on any particular date(s) in a month, you must specify the corresponding value(s) against the Monthly option.

  You do not need to specify a value (and the value will be ignored) if the collectionStatus parameter is set to Disabled.

  If you use this parameter, you must also use the freqType parameter.

- **preview**

  Provides a preview of the changes that would occur if this verb is executed. The default value for this option is Y (Yes), whether you specify the option or not. If

you specify N, the changes to the collection schedule are executed for both the repository and Management Agent.

Examples

Example 1

This example changes the collection schedule to collect once every 5 minutes for hosts host1, host2, and host3. DiskActivity is a collection item associated with a host target type. The preview flag is set to Y, so the changes are not executed, but you can see the metrics affected if the changes were implemented.

```
emcli modify_collection_schedule -targetType="host"
      -targetNames="host1;host2;host3" -collectionName="DiskActivity"
      -freqType="Minute" -freqValue="5" -preview="Y"
```

Example 2

This example changes the collection schedule to collect once every 15 hours for host host1. Inventory is a collection item associated with a host target type. The preview flag is set to N, so the changes are executed for the associated metrics for both the repository and Management Agent.

```
emcli modify_collection_schedule -targetType="host"
      -targetNames="host1" -collectionName="Inventory"
      -freqType="Hour" -freqValue="15" -preview="N"
```

Example 3

This example changes the collection schedule to collect on Monday and Thursday every week for hosts host1 and host2. Inventory is a collection item associated with a host target type. The preview option is not specified, but since the value is Y whether you specify the option or not, the changes are not executed, but you can see the metrics affected if the changes were implemented.

```
emcli modify_collection_schedule -targetType="host"
      -targetNames="host1;host2" -collectionName="Inventory"
      -freqType="Weekly" -freqValue="Mon;Thu"
```

Example 4

This example changes the collection schedule to collect on the 1st, 5th, 23rd, and last day of every month for hosts host1 and host2. Inventory is a collection item associated with a host target type.

```
emcli modify_collection_schedule -targetType="host"
      -targetNames="host1;host2" -collectionName="Inventory"
      -freqType="Month" -freqValue="1;5;23;Last"
```

Example 5

This example disables the collection schedule for hosts host1 and host2. Inventory is a collection item associated with a host target type.

```
emcli modify_collection_schedule -targetType="host"
      -targetNames="host1;host2" -collectionName="Inventory"
      -collectionStatus="Disabled"
```

modify_group

Adds or removes targets from an existing group.

An error is not generated when attempting to delete a non-existent target in the group or when attempting to add a target that already exists in the group.

Command-Line Format

```
emcli modify_group
      -name="name"
      [-type=<group>]
      [-add_targets="name1:type1;name2:type2;..."]...
      [-delete_targets="name1:type1;name2:type2;..."]...
      [-privilege_propagation=true|false]
      [-drop_existing_grants=yes|no]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
modify_group
      (name="name"
      [,type="<group>"]
      [,add_targets="name1:type1;name2:type2;..."]...
      [,delete_targets="name1:type1;name2:type2;..."]...
      [,privilege_propagation="true|false"]
      [,drop_existing_grants="yes|no"])

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the group to modify.

- **type**

  Group type: group. Defaults to `group`.

- **add_targets**

  Targets to add, each specified as `target_name:target_type`. You can specify this option more than once for command-line format.

- **delete_targets**

  Targets to delete, each specified as `target_name:target_type`. You can specify this option more than once for command-line format.

- **privilege_propagation**

  Enables or disables the privilege propagation flag for the group. Converts the normal group to a privilege propagating group and vice versa.

- **drop_existing_grants**

  Drops the existing grants on a group during privilege propagation conversion. This parameter is only applicable with the privilege_propagation parameter. The default value is yes.

Examples

These examples modify group db2_group by adding database database:oracle_database and deleting database database2:oracle_database from the group.

Example 1 - Command-Line

```
emcli modify_group
      -name=db2_group
      -add_targets=database:oracle_database
      -delete_targets=database2:oracle_database
```

Example 2 - Scripting and Interactive

```
modify_group
      (name="db2_group",
       add_targets="database:oracle_database",
       delete_targets="database2:oracle_database")
```

These examples modify group my_hosts by adding host yourhost.example.com:host to the group.

Example 3 - Command-Line

```
emcli modify_group
      -name=my_hosts
      -add_targets=yourhost.example.com:host
```

Example 4 - Scripting and Interactive

```
modify_group
       (name="my_hosts",
        add_targets="yourhost.example.com:host")
```

These examples modify group my_group by adding targets group_a:group and database:oracle_database and deleting the nonexistent target nogroup:group from the group.

Example 5 - Command-Line

```
emcli modify_group
      -name=my_group
      -add_targets=group_a:group
      -add_targets=database:oracle_database
      -delete_targets=nogroup:group
```

Example 6 - Scripting and Interactive

```
modify_group
      (name="my_group",
       add_targets="group_a:group;database:oracle_database",
       delete_targets="nogroup:group")
```

These examples convert group my_group to privilege propagating, ignores if already converted, and drops all of its existing grants.

Example 7 - Command-Line

```
emcli modify_group
          -name=my_group
```

```
          -privilege_propagation=true
```

Example 8 - Scripting and Interactive

```
modify_group
       (name="my_group",
        privilege_propagation="true")
```

These examples convert group my_group to non-privilege propagating, ignores if already converted, and retains all of its existing grants on my_group.

Example 9 - Command-Line

```
emcli modify_group
          -name=my_group
          -privilege_propagation=false
          -drop_existing_grants=no
```

Example 10 - Scripting and Interactive

```
modify_group
       (name="my_group",
        privilege_propagation="false",
        drop_existing_grants="no")
```

modify_incident_rule

Enables or disables a specific incident rule or rule set. (Updates all rules in the rule set.)

Format

```
emcli modify_incident_rule
      -action=enable|disable
      -type=ruleset|rule
      -rule_set_name=<name_of_rule_set>
      [-owner=<owner_of_rule_set>]
      [-rule_name=<name_of_rule>]

[ ] indicates that the parameter is optional
```

Parameters

- **action**

  Action to be performed. Supported actions are enable and disable.

- **type**

  Disables a specific rule or the entire rule set.

- **rule_set_name**

  Name of the rule set to which you would like to apply the action.

- **owner**

  Owner of the rule set. If multiple rule sets exist with same name, the rule set owner is used to identify the rule set.

- **rule_name**

  Name of the specific rule to which the action will apply.

Examples

Example 1

This example enables 'rule set 1' and all child rules.

```
emcli modify_incident_rule -action='enable' -type='ruleset' -rule_set_name='rule
set 1'
```

Example 2

This example disables 'rule set 1' and all child rules.

```
emcli modify_incident_rule -action='disable' -type='ruleset' -rule_set_name='rule
set 1'
```

Example 3

This example enables a single rule named 'rule 1' within 'rule set 1'.

```
emcli modify_incident_rule -action='enable' -type='rule' -rule_set_name='rule set
1' -rule_name='rule 1'
```

Example 4

This example disables a single rule named 'rule 1' within 'rule set 1'.

```
emcli modify_incident_rule -action='disable' -type='rule' -rule_set_name='rule set
1' -rule_name='rule 1'
```

modify_lifecycle_stage_name

Changes the life-cycle stage name. Only super users can run this command.

The Lifecycle Status property of the target has special semantics. The property does priority processing of events related to the target. Therefore, events from mission-critical targets have a higher priority than events from development targets. If you change the name, make sure to use a name that reflects its corresponding priority, because the same priority continues to be maintained regardless of the name change.

Format

```
emcli modify_lifecycle_stage_name
      -name="current_name"
      -new_name="new_name"
```

Parameters

- **name**

  Current life-cycle stage name. The available list in the order of decreasing priority is:

  - MissionCritical
  - Production
  - Stage
  - Test
  - Development

- **new_name**

  New life-cycle stage name. The new name is not translated into your locale and will be displayed as is. The new name should only contain alpha characters.

  When you change the existing name to a new name, all existing targets are updated with the new property value. For instance, if name=MissionCritical and new_name=Production, all existing targets are updated with Production.

Examples

```
emcli modify_lifecycle_stage_name
      -name="Test"
      -new_name="Test_staging"
```

modify_monitoring_agent

Changes the Agents configured to monitor targets in a WebLogic Domain.

Format

```
emcli modify_monitoring_agent
        -target_name=<target_name>
        [-target_type=weblogic_domain]
        [-assign_local_agent]
        -debug

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Complete target name of domain to be modified.

- **target_type**

  Default value is weblogic_domain, and is the only valid target type.

- **assign_local_agent**

  Globally assigns each target in the WebLogic Domain, such as WebLogic Server, to be monitored by the Agent installed on each target's host. That is, after running the verb with this option, each target in the domain is monitored by its local Agent. The local Agent is assigned if a local Agent is found. Otherwise, the monitoring Agent of the target is not changed.

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Examples

This example changes the Agents configured to monitor targets in a WebLogic Domain.

```
emcli modify_monitoring_agent
      -target_name=/prod_my_domain/my_domain
      -assign_local_agent
```

modify_named_credential

Updates an existing named credential. You can provide input parameters using command line arguments or an input properties file. It also supports the input_file tag for passwords and parameter values.

Format

```
emcli modify_named_credential
       -cred_name=<name>
       -new_cred_name<name>
       -cred_type=<credential_type>
       -cred_scope=<credential_scope>
       -cred_desc=<credential_description>
       -target_name=<target_name>
       -target_type=<target_type>
       -test
       -test_target_name=<test_target_name>
       -test_target_type=<test_target_type>
       -input_file=<tag|value>
       -properties_file=<filename>
       -attributes=<p1:v1;p2:v2;...>
       -remove_old_attributes
```

Parameters

- **cred_name**

  Credential name, such as MyBackUpCreds. This is required if you do not use the properties_file option.

- **new_cred_name**

  New credential name.

- **cred_type**

  Credential type.

- **cred_scope**

  Possible values are global instance. The default is global.

- **cred_desc**

  Credential description.

- **target_name**

  This is required when cred_scope is instance.

- **target_type**

  This is required when cred_scope is instance.

- **test**

  Use this parameter to test the credential before saving.

- **test_target_name**

  Use this parameter to supply the target name to test a global credential. This is mandatory when the scope is global and the test option is used.

- **test_target_type**

Use this parameter to supply the target type to test a global credential. This is mandatory when the scope is global and the test option is used.

- **input_file**

  Use this option to supply sensitive property values from the file.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **properties_file**

  Use this option to pass all parameters from the file. Values given on the command line take precedence.

- **attributes**

  Specify credential columns as follows:

  ```
  colname:colvalue;colname:colvalue
  ```

  You can change the separator value using -separator=attributes=<newvalue>, and you can change the subseperator value using -subseparator=attributes=<newvalue>.

- **remove_old_attributes**

  Unsets all existing credential column values.

Examples

Example 1

This example updates credentials to foo and bar:

```
emcli modify_named_credential
        -cred_name=NC1
        -attributes="HostUserName:foo;HostPassword:bar"
```

Example 2

This example updates the password to bar:

```
emcli modify_named_credential
        -cred_name=NC1
        -attributes="HostPassword:bar"
```

Example 3

This example reads the password from the mypasswordfile.txt file.

```
emcli modify_named_credential
        -cred_name=NC1
        -attributes="HostUserName:foo;HostPassword:tag"
        -input_file="tag:mypasswordfile.txt"
```

Example 4

This example prompts for the password from standard input:

```
emcli modify_named_credential
        -cred_name=NC1
        -attributes="HostUserName:foo;HostPassword:"
```

Example 5

This example specifies prop1.txt as a multi-line Java properties file, in which each line contains a parameter=value format. You can provide the password in the same file or not specify it. If not specified, you are prompted for it.

```
emcli modify_named_credential
        -properties_file=prop1.txt
```

modify_red_group

Adds or removes targets from an existing redundancy group. An error is not generated when attempting to delete a non-existent target in the redundancy group.

Format

```
emcli modify_red_group
      -name="name"
      -type=<generic_redundancy_group>
      [-add_targets="name1:type1;name2:type2;..."]...
      [-delete_targets="name1:type1;name2:type2;..."]...
      [-owner=<redundancy_group_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the group to modify.

- **type**

  Redundancy Group type: generic_redundancy_group. Defaults to `generic_redundancy_group`.

- **add_targets**

  Targets to add, each specified as `target_name:target_type`. You can specify this option more than once.

- **delete_targets**

  Targets to delete, each specified as `target_name:target_type`. You can specify this option more than once.

- **owner**

  Owner of the redundancy group.

Examples

This example modifies redundancy group servers by adding `Server1:generic_apache` and deleting `Server5:generic_apache` from the redundancy group.

```
emcli modify_red_group -name=Servers
      -add_targets=HTTP_Server1:generic_apache
      -delete_targets=Server5:generic_apache
```

modify_redundancy_group

Modifies a redundancy group.

Format

```
emcli modify_redundancy_group
        -redundancyGroupName="redGrpName"
        [-owner="new_owner"]
        [-memberTargetType="tType"]
        [-add_targets="tName1;tName2"]
        [-delete_targets="tName3;tName4"]
        [-group_status_criterion="NUMBER" or "PERCENTAGE"]
        [-group_status_tracked="UP" or "DOWN"]
        [-group_status_value=<status_value>]
        [-privilege_propagation=true|false]
        [-drop_existing_grants=yes|no]

[ ] indicates that the parameter is optional
```

Parameters

- **redundancyGroupName**

  Name of the redundancy group.

- **owner**

  Valid owner to be specified.

- **memberTargetType**

  Target type of the constituent member targets. You need to specify this parameter if you specify either add_targets or delete_targets.

- **add_targets**

  Member targets to be added to this redundancy group.

- **delete_targets**

  Member targets to be deleted from this redundancy group.

- **group_status_criterion**

  This option and the next two calculate the status of the Redundancy Group. Consequently, you need to specify all three options together. If this is not to be a capacity group, you need to specify the following combination:

  ```
  -group_status_criterion='NUMBER' -group_status_tracked='UP'
  -group_status_value='1']
  ```

- **group_status_tracked**

  See the option above.

- **group_status_value**

  See the group_status_criterion .

  You can specify any value between 1 and 100 if -group_status_criterion= "PERCENTAGE", or any value between 1 and the number of targets present if -group_status_criterion="NUMBER".

- **privilege_propagation**

  Enables or disables the privilege propagation flag for the group. Converts the normal group to a privilege-propagating group and vice versa.

- **drop_existing_grants**

  Drops the existing grants on a group during privilege propagation conversion. This parameter is only applicable with the privilege_propagation parameter. The default value is yes.

Examples

This example changes the configuration of the 'redGrp1' redundancy group to add listener, listener2, and listener3 to its existing members, and delete listener4 and listener5 from its existing members.

```
emcli modify_redundancy_group -redundancyGroupName='redGrp1'
       -memberTargetType='oracle_listener'
       -add_targets='listener;listener2;listener3'
       -delete_targets='listener4;listener5'
       -group_status_criterion='NUMBER'
       -group_status_tracked='UP'
       -group_status_value='2'
```

modify_resolution_state

Modifies an existing resolution state that describes the state of incidents or problems. Only super administrators can execute this command. You need to specify the updated label as well as the updated position. The position can be between 2 and 98, and cannot be in use by another resolution state.

You can also optionally indicate that the state should apply to both incidents and problems. A success message is reported if the command is successful. An error message is reported if the change fails.

Format

```
emcli modify_resolution_state
        -label="old_label_of_state"
        -new_label="new_label_for_display"
        -position="new_display_position"
        [-applies_to=BOTH]

[ ] indicates that the parameter is optional
```

Parameters

- **label**

  Old label of the state to be modified.

- **new_label**

  End-user visible label of the state. The label cannot exceed 32 characters.

- **position**

  Position of this state within the overall list of states. This is used when displaying the list of states in the user interface. The position can be between 2 and 98.

  It is recommended that you set the position with sufficient gaps to facilitate moving states around. For example, if you set the positions to 5, 10, and 15 instead of 2, 3, and 4, it is easier to move a state from position 15 to 9, for instance, in contrast to the latter scheme, in which you would have to move all states to provide space for the reordering.

- **applies_to**

  Indicates that the state is applicable for incidents and problems. The only supported value is "BOTH."

Examples

Example 1

This example updates the resolution state with the old label "Waiting for TT" with the new label "Waiting for Ticket," and if necessary, changes the position to 25.

```
emcli modify_resolution_state  -label="Waiting for TT" -new_label="Waiting for
Ticket" -position=25
```

Example 2

This example updates the resolution state with the old label "SR Waiting" with the new label "Waiting for SR," and if necessary, changes the position to 35. It also makes the state applicable to incidents and problems.

```
emcli modify_resolution_state  -label="SR Waiting" -new_label="Waiting for SR"
-position=35 -applies_to=BOTH
```

modify_role

Modifies an existing Enterprise Manager administrator role.

> **Note:** Omit an argument to leave its value unchanged.
>
> To update a role and add targets to the role, use the grant_privs verb.

Format

```
emcli modify_role
      -name="role_name"
      [-description="description"]
      [-roles="role1;role2;..."]
      [-privilege="name[;secure-resource-details]]"
      [-separator=privilege="sep_string"]
      [-subseparator=privilege="subsep_string"]
      [-users="user1;user2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Role name.

- **description**

  Replaces the description of the role.

- **roles**

  Replaces the list of roles assigned to this existing role. Currently, the only built-in role is PUBLIC.

- **privilege**

  Replaces privileges granted to this role. You can specify this option more than once. Specify <secure_resource_details> as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
  name2=resource_column_value2]..]"
  ```

- **separator**

  Specify a string delimiter to use between name-value pairs for the value of the -privilege option. The default separator delimiter is a semi-colon (;).

- **subseparator**

  Specify a string delimiter to use between name and value in each name-value pair for the value of the -privilege option. The default subseparator delimiter is a colon (:).

- **users**

  Replaces the list of users to whom this role is assigned.

Examples

Example 1

This example modifies a role named `existing_role` with the one-sentence description `"This role was changed`." The role combines three existing roles: `role1`, `role2`, and `role3`. The role also has two added privileges: to view the job with ID `923470234ABCDFE23018494753091111` and to view the target `host1.example.com:host`. The role is granted to `johndoe` and `janedoe`.

```
emcli modify_role
      -name="existing_role"
      -desc="This role was changed"
      -roles="role1;role2;role3"
      -privilege="view_job;923470234ABCDFE23018494753091111"
      -privilege="view_target;host1.example.com:host"
      -users="johndoe;janedoe"
```

Example 2

This example modifies a role named `existing_role` by assigning `role4`, `role5`, and `role6` to it. The description, privileges, and users associated with this role remain unchanged.

```
emcli modify_role
      -name="existing_role"
      -roles="role4;role5;role6"
```

modify_system

Adds or removes targets from an existing system. An error is not generated when attempting to delete a non-existent target in the system or when attempting to add a target that already exists in the system.

If you specify both the `-add_members` and `-delete_members` options in the same command, the members specified by `-delete_members` are deleted first, then the members specified by `-add_members` are added.

Format

```
emcli modify_system
         -name="name"
         [-type=<generic_system>]
         [-add_members="name1:type1:key_member|non_key_member;name2:type2;..."]...
              [-separator=add_members="sep_value"]
              [-subseparator=add_members="subsep_value"]
         [-delete_members="name1:type1;name2:type2;..."]...
              [-separator=delete_members="sep_value"]
              [-subseparator=delete_members="subsep_value"]
         [-owner="new_owner"]
         [-privilege_propagation=true|false]
         [-drop_existing_grants=yes|no]
         [-availability_type="ALL/ANY"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the system to modify.

- **type**

  System type: generic_system. Defaults to `generic_system`.

- **add_members**

  Targets to add, each specified as target_name:target_type. You can specify this more than once. key_member specifies that this target is a part of the systems availability calculation. non_key_member specifies that this target is not a part of the systems availability calculation.

- **delete_members**

  Member targets to be removed from the system, each specified as target_name:target_type. You can specify this option more than once.

- **owner**

  New owner of the system.

- **privilege_propagation**

  Enables or disables the privilege propagation flag for the group. Converts the normal group to a privilege propagating group and vice versa.

- **drop_existing_grants**

Drops existing grants on a group when conversion occurs in privilege propagation nature. This parameter is only applicable with the privilege_propagation parameter. The default value is yes.

- **availability_type**

  Availability calculation method of the system. Defining this is required if key_member is defined. ALL denotes that all key members must be up in order to establish the system as UP. ANY denotes that at least one of the key members must be up in order to establish the system as UP.

Examples

Example 1

This example modifies system `db2_system` by adding database `database:oracle_database` and deleting database `database2:oracle_database` from the system. The new owner of the system is `user2`.

```
emcli modify_system -name=db2_system
      -add_members=database:oracle_database
      -delete_members=database2:oracle_database
      -owner=user2
```

Example 2

This example modifies system `my_hosts` by adding host `yourhost.example.com:host` to the system.

```
emcli modify_system -name=my_hosts
      -add_members=yourhost.example.com:host
```

Example 3

This example modifies system `my_system` by adding targets `system_a:generic_system` and `database:oracle_database`, and deleting the nonexistent target `nosystem:generic_system` from the system.

```
emcli modify_system -name=my_system
      -add_members=system_a:generic_system
      -add_members=database:oracle_database
      -delete_members=nosystem:generic_system
```

Example 4

This example modifies system db2_system by adding database database1 as a key member, adding databases database2 and database3 as non-key members, and deleting database4 and database5. The availability computation is impacted, since database1 is now part of the availability computation for the db2_system. If database4 and database5 were key members, they are no longer part of the availability computation for the db2_system.

Specifying separator and subseparator is optional. Separator defaults to ; and subseparator defaults to : .

```
emcli modify_system -name=db2_system -type=generic_system
  [add_members=database1:oracle_database:key_member,database2:oracle_database]
     [separator=add_members=","]
     [subseparator=add_members=":"]
  [add_members=database3:oracle_database:non_key_member]
  [delete_members=database4:oracle_database,database5:oracle_database]
     [separator=delete_members=","]
```

```
[subsrparator=delete_members=":]"
```

modify_target

Modifies a target instance definition.

> **Note:** To change the monitoring password of a database target, either use update_db_password (at the RAC level), or use modify_target with the following options:
>
> ```
> -credentials="UserName:newuser;password:PWD_FILE;Role:SYSDBA"
> -input_file="PWD_FILE:at_pwd_file"
> ```

Format

```
emcli modify_target
      -name="name"
      -type="type"
      [-properties="pname1:pval1;pname2:pval2;..."]...
      [-separator=properties="sep_string"]
      [-subseparator=properties="subsep_string"]
      [-credentials="userpropname:username;pwdpropname:password;..."]
      [-input_file="parameter_tag:file_path"]
      [-display_name="display name"]
      [-on_agent]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name.

- **type**

  Target type.

- **properties**

  Name-value pair list of properties for the target instance. The "name"(s) are identified in the target-type metadata definition. They must appear exactly as they are defined in that file. Metadata files are located in `$AGENT_ORACLE_HOME/sysman/admin/metadata`.

  > **Note:** This verb does not support setting global target properties. It is recommended that you use set_target_property_values to set target properties.

- **separator=properties**

  Specifies a string delimiter to use between name-value pairs for the value of the `-properties` option. The default separator delimiter is ";".

- **subseparator=properties**

  Specifies a string delimiter to use between name and value in each name-value pair for the value of the `-properties` option. The default subseparator delimiter is ":".

- **credentials**

Monitoring credentials (name-value pairs) for the target instance. The "name"(s) are identified in the target-type metadata definition as credential properties. They must appear exactly as they are defined in that file. Metadata files are located in `$AGENT_ORACLE_HOME/sysman/admin/metadata`.

- **input_file**

  Used in conjunction with the `-credentials` option, this option enables you to store specific target monitoring credential values, such as passwords, in a separate file. The `-input_file` option specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific monitoring credentials of the `-credentials` option. The tag must not contain colons (:) or semi-colons (;).

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **display_name**

  Sets the target display name.

- **on_agent**

  Propagates changes to the Management Agent collecting this target's metrics.

Examples

Example 1

This example modifies the display name to `New Name DB` for the database with the internal name `database`.

```
emcli modify_target
      -name="database"
      -type="oracle_database"
      -display_name="New Name DB"
```

Example 2

This example modifies the credentials for the `oracle_database` target with the name `database`. This example illustrates the use of the `input_file` to camouflage the credentials. The password is actually in a file named `at_pwd_file`. The `input_file` argument replaces `PWD_FILE` with the contents of the `at_pwd_file` in the `credentials` argument. The `on_agent` flag ensures that the changes are propagated to the Management Agent collecting for this target.

```
emcli modify_target
      -name="database"
      -type="oracle_database"
      -credentials="UserName:newuser;password:PWD_FILE;Role:SYSDBA"
      -input_file="PWD_FILE:at_pwd_file"
      -on_agent
```

Example 3

This example modifies the display name and properties for the `oracle_database` target with the name `database`. The `on_agent` flag ensures that the changes are propagated to the Management Agent collecting for this target.

```
emcli modify_target
      -name="database"
      -type="oracle_database"
      -display_name="New Name DB"
      -properties="SID=newsid|Port=15091|OracleHome=/oracle"
```

```
      -properties="MachineName=smpamp-sun1.example.com"
      -separator=properties="|"
      -subseparator=properties="="
      -on_agent
```

Example 4

This example modifies an `oracle_database` target type with the name `payroll_db`. In this example, the display name for this database (target name that is displayed in the Enterprise Manager UI) is being changed to `payroll`. The port number is being changed to `15067`, and the Oracle Home is being changed to `/oradb`. The administrator (`dbsnmp`), whose previous default role was `normal`, is being changed to `sysdba`. This example also illustrates the use of the `input_file` to camouflage the credentials. The password is actually in a file named `at_pwd_file`. The `-input_file` argument replaces `PWD_FILE` with the contents of `at_pwd_file` in the `-credentials` option.

```
emcli modify_target
      -name="payroll_db"
      -type="oracle_database"
      -credentials="UserName:Fred;password:PWD_FILE;Role:sysdba"
      -properties="Port:15067;OracleHome:/oradb"
      -input_file="PWD_FILE:at_pwd_file"
      -display_name=payroll
      -on_agent
```

modify_threshold

Edits threshold settings for a given target and metric

Format

```
emcli modify_threshold
        -target_name="tname"
        -target_type="ttype"
        [-metric="met"]
        [-column="col"]
        [-key_columns="val1;val2;..."]
        [-warning_threshold="warn"]
        [-critical_threshold="crit"]
        [-occurrences="occur"]
        [-prevent_override="0 or 1"]
        [-force]
        [-input_file="FILE:cli_input.txt"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Name of the target associated with the threshold.

- **target_type**

  Type of target associated with the threshold.

- **metric**

  Metric category associated with the threshold.

- **column**

  Metric column associated with the threshold.

- **key_columns**

  Values of the key columns associated with the threshold. If you do not specify this option for a key-based metric, an EM CLI occurs.

- **warning_threshold**

  New warning threshold value. Specify " " for no warning threshold. If warning and critical thresholds are incoherent depending on the comparison operator, an EM CLI error occurs .

  Use -force to save the provided thresholds.

  To keep the previous value (if any), omit this option.

- **critical_threshold**

  New critical threshold value. Specify " " for no warning threshold. If warning and critical thresholds are incoherent depending on the comparison operator, an EM CLI error occurs .

  Use -force to save the provided thresholds.

  To keep the previous value (if any), omit this option.

- **occurrences**

Number of times a threshold can be violated before causing an alert. To keep the previous value (if any), omit this option.

- **prevent_override**

  Prevents thresholds modification of this metric from future Apply Template operations on this target. Periodic Apply Template operations are submitted on targets managed by Administration Groups, which can override the metric thresholds you set if the prevent_override flag is not set.

  An error occurs if prevent_override is not set in database, you have not provided prevent_override, and the target is managed by Administration Groups. To continue without using prevent_override, use -force.

  To keep the previous value (if any), omit this option.

- **force**

  Saves the provided thresholds incase recommended in previous error messages.

- **input_file**

  Provides threshold details for multiple metrics in a text file.

  Do not provide metric, column, key_columns, warning_threshold, critical_threshold, occurrences and prevent_override in this command when using the input_file option.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

  You can provide the details for multiple metrics in the input file as shown:

```
START_RECORD 1
metric , Filesystems
column , available
key_columns , ab;cd;
warning_threshold , 15
critical_threshold , 50
occurrences , 3
prevent_override , 1
END_RECORD 1

START_RECORD 2
metric , Load
column , cpuUtil
warning_threshold , 15
critical_threshold , 50
occurrences , 3
prevent_override , 1
END_RECORD 2
```

  To set the thresholds for the "Al Others" key, provide the details as shown:

```
START_RECORD 1
metric , Filesystems
column , available
key_columns , ;
warning_threshold , 15
critical_threshold , 50
occurrences , 1
END_RECORD 1
```

Examples

Example 1

This example sets the critical threshold value to "0" for the Load metric, and the cpuUtil column on the host "myhost.example.com". The warning threshold value and response action (if any) remain unchanged.

```
emcli modify_threshold
          -target_name="myhost.example.com"
          -target_type="host"
          -metric="Load"
          -column="cpuUtil"
          -critical_threshold="0"
          -prevent_override="0"
          -force
```

Example 2

This example sets the DiskActivitybusy threshold for the DiskActivitydevice called sd0 on the host myhost.example.com.

```
emcli modify_threshold
          -target_name="myhost.example.com"
          -target_type="host"
          -metric="DiskActivity"
          -column="DiskActivitybusy"
          -key_columns="sd0;"
          -warning_threshold="55"
          -critical_threshold="65"
          -occurrences="3"
```

Example 3

This example sets the sessions.active threshold for the name my_module and the oc4j_ear my_ear on the oc4j myOC4J.example.com.

```
emcli modify_threshold
          -target_name="myOC4J"
          -target_type="oc4j"
          -metric="oc4j_web_module_rollup"
          -column="session.active"
          -key_columns="my_module;my_ear;"
          -warning_threshold="1000"
          -critical_threshold="2000"
          -occurrences="4"
          -force
```

Example 4

This example uses emcli_input.txt for metric and threshold details.

```
emcli modify_threshold
          -target_name="myOC4J"
          -target_type="oc4j"
          -input_file="FILE:/home/emcli_input.txt"
```

Example 5

This example uses emcli_input.txt for metric and threshold details, and the -force option for all the metrics provided in the input file.

```
emcli modify_threshold
          -target_name="myOC4J"
```

```
-target_type="oc4j"
-input_file="FILE:/home/emcli_input.txt"
-force
```

modify_user

Modifies an existing Enterprise Manager administrator.

Format

```
emcli modify_user
        -name="name"
        [-type="type_of_user"]
        [-password="password"]
        [-roles="role1;role2;..."]
        [-email="email1;email2;..."]
        [-privilege="name[;secure_resource_details]]"
        [-separator=privilege="sep_string"]
        [-subseparator=privilege="subsep_string"]
        [-profile="profile_name"]
        [-desc="user_description"]
        [-expired="true|false"]
        [-prevent_change_password="true|false"]
        [-department="department_name"]
        [-cost_center="cost_center"]
        [-line_of_business="line_of_business"]
        [-contact="contact"]
        [-location="location"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Administrator name.

- **type**

  Converts the type of user from a repository user to an external user and vice versa. Possible values for this parameter are EM_USER or EXTERNAL_USER.

- **password**

  Replaces the administrator password with the specified password.

- **roles**

  Replaces current roles with the specified list of Enterprise Manager roles to grant to this administrator. Currently, the built-in roles include PUBLIC.

- **email**

  Replaces current email addresses for this administrator with the specified list. To delete all email addresses for this administrator, specify an empty string.

- **privilege**

  Privilege to grant to this administrator. You can specify this option more than once. The original administrator privileges will be revoked. Specify <secure_resource_details> as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
  name2=resource_column_value2]..]"
  ```

  To retrieve the list of system privileges that do not require resource information, execute the get_supported_privileges command.

- **profile**

  Database profile name. When not passed, this value is not altered.

- **desc**

  User description

- **expired**

  True immediately expires the password. The default is false.

- **prevent_change_password**

  True prevents a user from updating his/her password. The default is false.

- **department**

  Department name of the administrator.

- **cost_center**

  Cost center of the administrator in the organization.

- **line_of_business**

  Line of business of the administrator.

- **contact**

  Contact information for the administrator.

- location

  Location of the administrator.

Examples

Example 1

This example modifies the `new_admin` administrator. The user will have two privileges: to view the job with ID `923470234ABCDFE230184947530911111` and to view the target `host1.example.com:host`. The user will also be granted role `PUBLIC`. The user email addresses will be set to `first.last@example.com` and `joe.shmoe@shmoeshop.com`.

```
emcli modify_user
      -name="new_admin"
      -password="oracle"
      -email="first.last@example.com;joe.shmoe@shmoeshop.com"
      -roles="public"
      -privilege="view_job;923470234ABCDFE230184947530911111"
      -privilege="view_target;host1.example.com:host"
```

Example 2

This example deletes all the email addresses and privileges for administrator `new_admin`. Note that `-privilege=""` and `-privilege` are equivalent if specified at the command line in a UNIX shell.

```
emcli modify_user
      -name="new_admin"
      -email="
      -privilege=""
```

modify_virtual_platform

Modifies the Oracle Virtual Platform target's monitoring agent, fail-over agent, or the monitoring credentials. Only the properties of the target needing modification must be specified when modifying a target of that type. For all of the parameters not passed, the existing values are retained.

Format

```
emcli modify_virtual_platform
       -name="target_name"
       -agent="agent_target_name"
       [-failover_agent="failover_agent_target_name"]
       -credentials="property_name1:property_value1;property_name2:
           property_value2;..."
       [-wait_for_completion=true|false]
       [-wait_for_completion_timeout=<time_in_minutes>]
       [-separator=credentials="separator_for_key_value_pairs"]
       [-subseparator=credentials="separator_for_key_value_pair"]
       [-input_file="FILE:file_path"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the Oracle Virtual Platform to modify.

- **agent**

  Target name of the primary agent used to monitor the Oracle Virtual Platform and related targets.

- **failover_agent**

  Target name of the failover agent used to monitor the Oracle Virtual Platform and related targets.

- **credentials**

  Monitoring credentials (name-value pairs) for the target instance. The "names" are defined in the target type metadata definition as credential properties. Metadata files are located at $AGENT_HOME/sysman/admin/metadata.

  See the examples for details on various options.

- **wait_for_completion**

  Flag to indicate if the CLI is going to wait for the submitted job to finish. The default value is false. If the value is true, the progress of the job is printed on the command line as and when the addition of Oracle Virtual Platform(s) Succeeds/Fails.

- **wait_for_completion**

  Flag to indicate if the CLI is going to wait for the submitted job to finish. The default value is false. If the value is true, the CLI waits and prints the job output on the command line when the modification of Oracle Virtual Platform(s) Succeeds/Fails.

- **wait_for_completion_timeout**

Time in minutes after which CLI stops waiting for the job to finish. This parameter is honored only if the value for parameter wait_for_completion is true. A negative or zero value does not wait for the job to finish.

See the examples for details.

- separator=credentials

  Custom separator for the credential key value pairs. Specify a string delimiter to use between name-value pairs for the values of the -credentials option. The default separator delimiter is ";".

  For more information about the separator parameter, see Section 5.2, "-input_file Syntax Guidelines".

- subseparator=credentials

  Custom separator for a key value pair. Specify a string delimiter to use between name and value in each name-value pair for the values of the -credentials option. The default subseparator delimiter is ":".

  For more information about the subseparator parameters see Section 5.2, "-input_file Syntax Guidelines".

- **input_file**

  File path with a credential secret value. Optionally use in conjunction with the -credentials option. You can use this option to set specific target monitoring credential values, such as passwords or SSH keys, in a separate file.

  This option specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific -credentials property values.

Examples

Example 1

This example modifies the primary monitoring Agent of the Oracle Virtual Platform to the specified Agent target.

```
emcli modify_virtual_platform -name=exampletarget -agent=example.com:1838
```

Example 2

This example modifies the primary monitoring Agent to a specified Agent target, and the failover agent to none.

```
emcli modify_virtual_platform -name=exampletarget -agent=example.com:1828
-failover_agent=""
```

Example 3

This example modifies the Oracle Virtual Platform's credentials with root user host credentials. The value of the property "OVSUsername" is used for the user name and "OVSPassword" for the password. The value of the property "privilegedUser" indicates if the virtualization-specific metrics are collected (true) or not (false) when monitoring. The password is passed at the command line.

```
emcli modify_virtual_platform
          -credentials='type:DMOvsBasicCreds;PrivilegeType:none;
              privilegedUser:true;OVSUsername:root;OVSPassword:mypassword'
```

Example 4

This example modifies the Oracle Virtual Platform's credentials with root user host credentials. The value of the property "OVSUsername" is used for the user name and "OVSPassword" for the password. The value of the property "privilegedUser" indicates if the virtualization-specific metrics are collected (true) or not (false) when monitoring. The password of the root user is read from the input file "password.txt".

```
emcli modify_virtual_platform
      -name=exampletarget
      -credentials='type:DMOvsBasicCreds;PrivilegeType:none;
          privilegedUser:true;OVSUsername:root;OVSPassword:PWD_FILE'
      -input_file='PWD_FILE:password.txt
```

Example 5

This example modifies the Oracle Virtual Platform's credentials with Unix Sudo user host credentials. The value of the property "PrivilegeCommand" is used to execute the Sudo command. %RUN_AS% and %COMMAND% are replaced with the user and the command to be executed by Sudo command. The value of the property "EnablePseudoTerminal" indicates if Sudo requires (true) a tty terminal or not (false). The password is passed at the command line.

```
emcli modify_virtual_platform
      -name=exampletarget
      -credentials='type:DMOvsBasicCreds;PrivilegeType:sudo;
          privilegedUser:true;RunAs:root;
          PrivilegeCommand:/usr/bin/sudo -S -u %RUN_AS% %COMMAND%;
          EnablePseudoTerminal:false;OVSUsername:root;OVSPassword:mypassword'
```

Example 6

This example modifies the Oracle Virtual Platform's credentials with Unix PowerBroker user host credentials. The value of the property "PrivilegeCommand" is used to execute the PowerBroker command. %RUN_AS% and %COMMAND% are replaced by the user and the command to be executed by PowerBroker. The value of the property "PowerBrokerProfile" is used as the PowerBroker profile. The value of the property "PowerBrokerPasswordPrompt" is used as the PowerBroker password prompt. The password is passed at the command line.

```
emcli modify_virtual_platform
      -name=exampletarget
      -credentials='type:DMOvsBasicCreds;PrivilegeType:powerbroker;
         RunAs:root;privilegedUser:true;OVSUsername:root;OVSPassword:mypassword;
         PrivilegeCommand:/usr/bin/pbrun -l -u %RUN_AS% %COMMAND%;
         PowerBrokerProfile:profile;PowerBrokerPasswordPrompt:myprompt'
```

Example 7

This example modifies the Oracle Virtual Platform's credentials with a Unix Sudo user who requires SSH key Passphrase-less based authentication. The SSH private key, SSH public key, and password are read from input files.

```
emcli modify_virtual_platform
      -name=exampletarget
      -credentials='type:DMOvsSshKeyCreds;PrivilegeType:sudo;
          privilegedUser:true;RunAs:root;
          PrivilegeCommand:/usr/bin/sudo -S -u %RUN_AS% %COMMAND%;
          EnablePseudoTerminal:false;
          SshPrivateKey:PRIVATE_KEY;SshPublicKey:PUBLIC_KEY;
          OVSUsername:sudoer1;OVSPassword:PWD_FILE'
      -input_file='PRIVATE_KEY:id_dsa'
```

```
      -input_file='PUBLIC_KEY:id_dsa.pub'
      -input_file='PWD_FILE:password'
```

Example 8

This example modifies the Oracle Virtual Platform's credentials with a Unix PowerBroker user who requires SSH key Passphrase-less based authentication. The SSH private key, SSH public key, and password are read from input files.

```
emcli modify_virtual_platform
      -name=exampletarget
      -credentials='type:DMOvsSshKeyCreds;PrivilegeType:powerbroker;
          privilegedUser:true;RunAs:root;
          PrivilegeCommand:/usr/bin/pbrun -l -u %RUN_AS% %COMMAND%;
          PowerBrokerProfile:profile;PowerBrokerPasswordPrompt:myprompt;
          SshPrivateKey:PRIVATE_KEY;SshPublicKey:PUBLIC_KEY;
          OVSUsername:myuser;OVSPassword:PWD_FILE'
          -input_file='PRIVATE_KEY:id_dsa'
          -input_file='PUBLIC_KEY:id_dsa.pub'
          -input_file='PWD_FILE:password'
```

Example 9

This example modifies the Oracle Virtual Platform's credentials with non- privileged user host credentials. The virtualization metrics for the added target will not be monitored. The password is specified at the prompt.

```
emcli modify_virtual_platform
      -name=exampletarget
      -credentials='type:DMOvsBasicCreds;privilegedUser:false;
          OVSUsername:simpleton;OVSPassword:password'
```

package_fa_problem

This verb accomplishes the following tasks:

- Packages a Fusion Applications problem by reading details from a pre-written input file.
- Optionally attaches metrics, custom dumps, and reports by reading details from pre-written heap dumps and database AWR (Automatic Workload Repository) files.
- Uploads the finalized package to Oracle Support and reports the number of the draft Service Request created for the package if no SR is supplied.

Format

```
emcli package_fa_problem
      -input_file=incident_packaging_file:file_path
      [-input_file=heap_dumps_file:file_path]
      [-input_file=db_awr_file:file_path]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file=incident_packaging_file**

  Fully-qualified path to a CSV formatted file containing one line of details for the Fusion Applications problem to be packaged.

  The structure of the CSV file is as follows:

  ```
  <Full target name>,
  <Target type>,
  <Problem key>,
  <Host credential name - for using named credentials only>,
  <Host username - for using new credentials only>,
  <Host password - for using new credentials only>,
  <Target credential name - for using named credentials only>,
  <Target username - for using new credentials only>,
  <Target password - for using new credentials only>,
  <Boolean for adding host metrics - optional - default is true>,
  <Boolean for adding WebLogic metrics - optional - default is true>,
  <Boolean for adding JVM dump - optional - default is true>,
  <Boolean for adding heap dumps - optional - default is false>,
  <Boolean for adding Automatic Workload Repository (AWR) reports - optional -
  default is false>,
  <My Oracle Support username>,
  <My Oracle Support password>,
  <Service Request (SR) number - required if no CSI given>,
  <Customer Support Identifier (CSI) - required if no SR number given>
  ```

  For example:

  ```
  /HCMDomain/Server_1/SetupApp,fusion_apps_j2ee_
  app,Other-1,,username,mypassword,,FAadmin,fusionfa1,,,,,GENERIC@oracle.com,,3-
  6586541801
  /HCMDomain/Server_1/SetupApp,fusion_apps_j2ee_app,Other-1,HOST_CREDS,,,WLS_
  CREDS,,,false,false,false,true,true,GENERIC@oracle.com,,,15427437
  /HCMDomain/Server_1/SetupApp,fusion_apps_j2ee_
  app,Other-1,,,,,,,,false,,,true,GENERIC@oracle.com,,3-6586541801
  ```

Note the following points about the format of incident_packaging_file:

- The delimiter used is a comma (,).
- The order of parameters is fixed. You must provide the parameters in the same order as specified above in the sample file structure.
- Delimiters must be present even if the corresponding parameter is not provided.
- If you want to use a comma in one of the parameters provided, you must escape the comma with a backslash, as shown in This example in which the password has a comma:

```
/HCMDomain/Server_1/SetupApp,fusion_apps_j2ee_
app,Other-1,,username,mypassword,,FAadmin,fusion\,fa1,,,,,GENERIC@oracle.c
om,,3-6586541801
```

- If you want to use a backslash in one of the parameters provided, you must escape the backslash with a backslash, as shown in This example in which the password has a comma:

```
/HCMDomain/Server_1/SetupApp,fusion_apps_j2ee_
app,Other-1,,username,mypassword,,FAadmin,fusion\
\fa1,,,,,,GENERIC@oracle.com,,3-6586541801
```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=heap_dumps_file**

  Fully qualified path to a CSV formatted file containing multiple lines of fully qualified paths to heap dump files to be included in the package. The files whose locations are provided in the file are added as heap dumps to the package.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=db_awr_file**

  Fully-qualified path to a CSV formatted file containing multiple lines of databases and the credentials used to generate reports for the package. The AWR reports generated by the databases provided in the file are added to the package, assuming that the credentials, if needed, are provided and valid.

  The structure of the CSV file is as follows:

```
<Database name as used in EM>,
<credential name - for using named credential only>,
<username - for using new credential only>,
<password - for using new credential only>,
<role - optional, for using new credential only>
```

  For example:

```
Oemrep_database (preferred credentials set in Enterprise Manager)
Oemrep_database,MY_DB_CREDS
Oemrep_database,,sysman,sysman
Oemrep_database,,sysman,sysman,normal
```

  Note the following points about the format of db_awr_file:

  - The delimiter used is a comma (,).

– The order of parameters is fixed. You must provide the parameters in the same order as specified above in the sample file structure.

For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

Example 1

This example shows a fully-qualified path to a CSV formatted file containing one line of details for the Fusion Applications problem to be packaged.

```
/HCMDomain/Server_1/SetupApp,fusion_apps_j2ee_
app,Other-1,,username,mypassword,,FAadmin,fusionfa1,,,,,,GENERIC@oracle.com,,
3-6586541801
/HCMDomain/Server_1/SetupApp,fusion_apps_j2ee_app,Other-1,HOST_CREDS,,,
WLS_CREDS,,,false,false,false,true,true,GENERIC@oracle.com,,,15427437
/HCMDomain/Server_1/SetupApp,fusion_apps_j2ee_
app,Other-1,,,,,,,,false,,,true,GENERIC@oracle.com,,3-6586541801
```

Example 2

This example shows a fully-qualified path to a CSV formatted file containing multiple lines of databases and the credentials used to generate reports for the package.

```
Oemrep_database (preferred credentials set in Enterprise Manager)
Oemrep_database,MY_DB_CREDS
Oemrep_database,,sysman,sysman
Oemrep_database,,sysman,sysman,normal
```

pdb_backup

Takes a backup of the data files and metadata xml of a given pluggable database (PDB).

Format

```
emcli pdb_backup
      -inputFile="File containing properties required for taking backup of PDB"
```

Parameters

- inputFile

  Location of the file containing properties required for taking a backup of a PDB.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example takes a back of the PDB contained in the `pdb_backup.props` file:

```
emcli pdb_backup
      -input_file=data:/u01/files/pdb_backup.props
```

Contents of pdb_backup.props:

```
TARGET_HOST_LIST=xyz.abccorp.com
HOST_NORMAL_NAMED_CRED=XYZ_CRED:CRED_OWNER
SRC_CDB_NAMED_CRED=CDB1_CRED:CRED_OWNER
SRC_CDB_TARGET_NAME=CDB1
SRC_CDB_TARGET_TYPE=oracle_database
SRC_PDB_TARGET_NAME=CDB1_PDB1
BACKUP_LOCATION=/scratch/pdbBackup
WORK_DIR_LOCATION=/tmp
ORACLE_HOME_LOC=/scratch/d121hmcasm/product/12.1.0/dbhome_1
```

pdb_clone_management

Creates a new cloned PDB.

Format

```
emcli pdb_clone_management
     [-cloneToOracleCloud = Clone PDB to Container database (CDB) on Oracle Cloud]
     -input_file = pdb_input_file

[ ] indicates that the parameter is optional.
```

Parameters

- cloneToOracleCloud

  Specifies if the destination CDB is on Oracle Cloud.

- input_file

  Location of the file containing properties required for cloning a PDB. The allowed properties for this job are:

  SRC_CDB_TARGET = Enterprise Manager target name of the CDB containing the source PDB.

  SRC_CDB_TYPE = Enterprise Manager target type of the CDB containing the source PDB.

  SRC_CDB_CREDS = Named credentials for the source CDB.

  SRC_HOST = Enterprise Manager target name of host containing source CDB. If not provided, will be defaulted from CDB.

  SRC_HOST_CREDS = Named credentials for the source target host.

  SRC_PDB_TARGET = Enterprise Manager target name of the source PDB.

  SRC_WORK_DIR = Work directory at source host where files will be temporarily stored. If not provided, will be defaulted to agent work directory.

  DEST_HOST = Enterprise Manager target name of host containing destination CDB. If not provided, will be defaulted from CDB.

  DEST_HOST_CREDS = Named credentials for the destination target host. If destination host is on OPC, this should be Host SSH credentials.

  DEST_LOCATION = Data file location at the destination where the new PDB will be hosted.

  DEST_CDB_TARGET = Enterprise Manager target name of CDB where the new PDB should be cloned.

  DEST_CDB_TYPE = Enterprise Manager target type of destination CDB.

  DEST_CDB_CREDS = Named credentials for the destination CDB.

  DEST_PDB_NAME = Name of the new PDB.

  EXISTING_BACKUP = Absolute location of the existing backup in the file system, if it should be used to clone new PDB.

  EXISTING_BACKUP_METADATA = Absolute location of the metadata template of the backup. Required, if EXISTING_BACKUP is provided.

BACKUP_TYPE = [TAR | | OSIMAGE | | RMAN]

If existing backup is provided, this represents the type of the backup. If not, this represents the type of backup that should be taken during job execution. If both, EXISTING_BACKUP and BACKUP_TYPE are not provided, source PDB will be unplugged and copied over to destination for creating new clone. After the data files are copied, the source PDB will be plugged back.

Mandatory properties:

SRC_PDB_TARGET, SRC_HOST_CREDS, SRC_CDB_CREDS, SRC_WORK_DIR, DEST_HOST_CREDS, DEST_LOCATION, DEST_CDB_TARGET, DEST_CDB_TYPE, DEST_CDB_CREDS, DEST_PDB_NAME

Clone Types:

Full Clone - Live backup: Takes a backup of the source PDB and creates a new PDB. BACKUP_TYPE specifies the type of backup.

Full Clone - Existing Backup: Uses an existing backup of the source PDB and creates a new PDB. BACKUP_TYPE specifies the type of backup.

EXISTING_BACKUP: Specifies the backup name and EXISTING_BACKUP.

METADATA: Specifies the metadata for the backup.

Full Clone - Unplug/Plug: Unplugs the source PDB, creates a new PDB at the destination using the unplugged source, and plugs the source back.

Example

The following example creates a new cloned PDB from the information contained in the `pdb_clone.props` file.

```
emcli pdb_clone_management
      -input_file=data:/u01/files/pdb_clone.props
```

provision

Provisions a hardware server using configuration properties from the input file. The configuration properties required for a component can be viewed from the Cloud Control console. After you make a provisioning request, you can view the status of the request from the Enterprise Manager Cloud Control console by using the assignment name (specified by you or the automatically generated name returned to you).

Format

```
emcli provision
      -image="path_to_image"
      -network="network_profile_path"
      -bootserver="boot_server_name"
      -stageserver="stage_server_name"
      -stgcredentials="username"
      -schedule="type:immediate/onetime;timezone:zone;
       startdt:startdate;starttm:time"
      -resettimeout="time"
      -target="hardware_server_label"
      -input_file="config_properties:file_path"
      -assignment="assignment_name"
      [-desc="assignment_description"]

[ ] indicates that the parameter is optional
```

Parameters

- **image**

  Path to the image (includes the image name). This is the image used for provisioning.

- **network**

  Path name of the network profile.

- **bootserver**

  Name of the boot server.

  Format: hostName:Directory Path

- **stageserver**

  Name of the stage server. hostName:Directory Path.

- **Stgcredentials**

  User name of the stage server.

- **schedule**

  Time when provisioning should be scheduled. This is a string argument that contains multiple name-value pairs separated by `;'. This is used to schedule the provisioning operation. "type" can be `immediate' or `onetime'. If "type" is not immediate, the other values are expected in the Time Zone: string, which is a timezone ID of the format:

  zone Sign TwoDigitHours:Minutes

  zone: Time zone ID (GMT, PDT, and so forth)

  Sign: one of "+ -"

TwoDigitHours: Digit Digit

Minutes: Digit Digit

Digit: One of 0 1 2 3 4 5 6 7 8 9

Startdt: Date string of the format: MM/DD/YY

Starttm: Time string of the format: HH:MM

- **resettimeout**

  Reset timeout for the hardware server in minutes.

- **target**

  Target hardware server is specified using the hardware label type.

- **input_file**

  File containing configuration properties.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **assignment**

  Name of the assignment.

- **desc**

  Assignment description. The description is automatically generated if not specified.

Examples

This example submits a job to provision `myimage` on a target with the label of `mylabel`. The job runs immediately with a reset timeout of 100 minutes. Image properties are picked from `properties.txt` that overrides the default image. `properties.stageserver` is used as the staging server, and `/private/share` as the staging storage with `joe` as the user name.

```
emcli provision
      -image="Images/myimage"
      -network="Networks/networkprofile"
      -bootserver="booservername.example.com"
      -stageserver="stageserver.example.com:/private/share"
      -stgcredentials="joe"
      -schedule="type:immediate"
      -resettimeout="100"
      -target="mylabel"
      -input_file="config_properties:properties.txt"
      -assignment="provision mylabel"
```

publish_change_request_ccc

Sends change request data to the Change Management Connector, and data processed into the Configuration Change Console. Some of the properties (such as connector_guid, target, and facet) are to be specified as part of customization. All of the data should be able to be mapped to the data required in publishChangeRequest.xsd after XSLT.

Format

```
emcli publish_change_request_ccc
    -connector_guid="ConnectorGUID"
    -change_id="change_ID"
    -last_modified_date="last_modified_date"
    -properties_list="list_of_Change_Management_specific_properties"
    -date_format="Date_format_in_Change_Management_System"
```

Parameters

- **connector_guid**
- **change_id**
- **last_modified_date**
- **properties_list**

  Specify all relevant properties of the Change Management System required for CCC to process a change request.

  The properties are name,value pairs to be specified as prop_name1=value1;prop_name2=value2 with no quotes for values.

  prop_name and values cannot contain the equals sign (=) or semi-colons (;).

- **date_format**

  Specify a date format in the Change Management System:

  MM/dd/yyyy hh:mm:ss if the date field in change management is "09/14/2011 5:38:24 AM"

publish_event

Publishes a user-reported event to Enterprise Manager. This event is published as an event of the "User-reported event" class. Only users with Manage Target privilege can publish these events for a target. An error message is reported if the publish fails.

After an event is published with a severity other than CLEAR (see below), end-users with appropriate privileges can manually clear the event from the user interface, or you can publish a new event using a severity level of CLEAR and the same details to report clearing of the underlying situation.

Format

```
emcli publish_event
        -target_name="target_name"
        -target_type="target_type_internal_name"
        -message="message_for_event"
        -severity="severity_level"
        -name="event_name"
        [-key="sub_component_name"
        [-context="name1=value1;name2=value2;.."]
        [-separator=context="alt._pair_separator"]
        [-subseparator=context="alt._name-value_separator"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Target name.

- target_type

  Target type name.

- message

  Message to associate for the event. The message cannot exceed 4000 characters.

- **severity**

  Numeric severity level to associate for the event. The supported values for severity level are as follows:

  "CLEAR"
  "MINOR_WARNING"
  "WARNING"
  "CRITICAL"
  "FATAL"

- **name**

  Name of the event to publish. The event name cannot exceed 128 characters.

  This is indicative of the nature of the event. Examples include "Disk Used Percentage," "Process Down," "Number of Queues," and so on. The name must be repeated and identical when reporting different severities for the same sequence of events. This should not have any identifying information about a specific event; for example, "Process xyz is down." To identify any specific components within a target that the event is about, see the key below.

- **key**

  Name of the sub-component within a target this event is related to. Examples include a disk name on a host, name of a tablespace, and so forth. The key cannot exceed 256 characters.

- **context**

  Additional context that can be published for a given event. This is a series of strings of format name:value separated by a semi-colon. For example, it might be useful to report the percentage size of a disk when reporting space issues on the disk. You can override the default separator ":" by using the subseperator , and the pair separator ";" by using the separator .

  The context names cannot exceed 256 characters, and the values cannot exceed 4000 characters.

- **separator**

  Set to override the default ";" separator. You typically use this option when the name or the value contains ";". Using "=" is not supported for this option.

- **subseparator**

  Set to override the default ":" separator between the name-value pairs. You typically use this option when the name or value contains ":". Using "=" is not supported for this .

Examples

Example 1

This example publishes a warning event for "my acme target" indicating that a HDD restore failed, and the failure related to a component called the "Finance DB machine" on this target.

```
emcli publish_event  -target_name="my acme target" -target_type="oracle_acme"
-name="HDD restore failed" -key="Finance DB machine" -message="HDD restoration
failed due to corrupt disk" -severity=WARNING
```

Example 2

This example publishes a minor warning event for "my acme target" indicating that a HDD restore failed, and the failure related to a component called the "Finance DB machine" on this target. It specifies additional context indicating the related disk size and name using the default separators. Note the escaping of the \ in the disk name using an additional "\".

```
emcli publish_event  -target_name="my acme target" -target_type="oracle_acme"
-name="HDD restore failed" -key="Finance DB machine" -message="HDD restoration
failed due to corrupt disk" -severity=MINOR_WARNING -context="disk
size":800GB\;"disk name":\\uddo0111245
```

Example 3

This example publishes a critical event for "my acme target" indicating that a HDD restore failed, and the failure was related to a component called the "Finance DB machine" on this target. It specifies additional context indicating the related disk size and name. It uses alternate separators, because the name of the disk includes the ":" default separator.

```
emcli publish_event  -target_name="my acme target" -target_type="oracle_acme"
-name="HDD restore failed" -key="Finance DB machine" -message="HDD restoration
failed due to corrupt disk" -severity=CRITICAL -context="disk size"^800GB\;"disk
name"^\\sdd1245:2 -subseparator=context=^
```

publish_metric_extension

Publishes a metric extension for use by all administrators. The metric extension must currently be a deployable draft.

Format

```
emcli publish_metric_extension
       -target_type=<metric_extension_target_type>
       -name=<metric_extension_name>
       -version=<metric_extension_version>
```

Parameters

- **target_type**

  Target type of the metric extension.

- **name**

  Name of the metric extension.

- **version**

  Version of the metric extension to be published.

Example

This example publishes a metric extension of a given target type, name, and version.

```
emcli publish_metric_extension -target_type=<target type of the metric extension>
-name=<name of the metric extension -version=<version of the metric extension>
```

reassoc_masking_definition

Reassociates an existing masking definition with another database target.

Format

```
emcli reassoc_masking_definition
   -definition_name=masking definition name
   -target_name=database target name
   -target_type=database target type
   [-parameters=name1:value1;name2:value2;...]
   [-credential_name=credential_name]
   [-input_file=parameter_tag:file_path]

[ ] indicates that the parameter is optional
```

Parameters

- **definition_name**

  Masking definition name.

- **target_name**

  New database target name with which to associate the masking definition.

- **target_type**

  New database target type with which to associate the masking definition.

- **parameters**

  List of name-value pairs that represent the credentials required for connecting to the database instance. The supported parameters are db_username, db_password, and db_role.

- **credential_name**

  Name of the database credential. This parameter is mandatory when the db_username and db_password parameters are not specified.

- **input_file**

  Used in conjunction with the parameters option, this option enables you to store parameter values, such as username and password, in a separate file. This option specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific parameter values for the parameters . The tag must not contain colons (:) or semi-colons (;).

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Output

Success or failure message along with the details.

Examples

Example 1

This example reassociates the masking definition mask_hr_data with the new database target testdb2 :

```
emcli reassoc_masking_definition
     -definition_name=mask_hr_data
     -target_name=testdb2
     -parameters="db_username:system;db_password:password;db_role:NORMAL"
```

Example 2

This example reassociates the masking definition mask_hr_data with the new database target testdb2. The database password is read from the pwd.txt file.

```
emcli reassoc_masking_definition
      -definition_name=mask_hr_data
      -target_name=testdb2
      -parameters="db_username:system;db_password:PWD_FILE;db_role=SYSDBA"
      -input_file="PWD_FILE:pwd.txt"
```

Example 3

This example reads the credentials from the preferred credential set DBCredsNormal and reassociates the masking definition.

```
      emcli reassoc_masking_definition
      -definition_name=mask_hr_data
      -target_name=testdb2
```

Example 4

This example reads the credentials from the preferred credential set DBCredsSYSDBA and reassociates the masking definition.

```
      emcli reassoc_masking_definition
      -definition_name=mask_hr_data
      -target_name=testdb2
      -credential_set_name=DBCredsSYSDBA
```

redeploy_plugin_on_agent

Redeploys an existing plug-in on the Management Agents.

Format

```
emcli redeploy_plugin_on_agent
      {-agent_names="agent1[;agent2...]" | -group_name="group1"}
      -plugin="plug-in_id:version"
      [-redeploy_noprompt]
      [-include_dependent_agents]

[ ] indicates that the parameter is optional.
```

Parameters

- **agent_names**

  List of Management Agents (host:port) on which the plug-in should be redeployed.

- **plugin**

  ID and version of the plug-in that should be redeployed on the Management Agents.

- **redeploy_noprompt**

  Redeploys the same plug-in that is already available in the Plug-in manager inventory, without prompting you to confirm the redeployment.

- **include_dependent_agents**

  Includes all of the dependent Management Agents and proceeds with plug-in redeployment.

Examples

Example 1

The following example redeploys 12.1.0.2.0 version of the `oracle.sysman.db2` plug-in on the Management Agent named host.example.com:

```
emcli redeploy_plugin_on_agent
      -agent_names="host.example.com:1838"
      -plugin="oracle.sysman.db2:12.1.0.2.0"
```

Example 2

The following example redeploys 12.1.0.2.0 version of the `oracle.sysman.db2` plug-in on the Management Agent named `host.example.com` without prompting you for any confirmation:

```
emcli redeploy_plugin_on_agent
      -agent_names="host.example.com:1838"
      -plugin="oracle.sysman.db2:12.1.0.2.0"
      -redeploy_noprompt
```

Example 3

The following example redeploys 12.1.0.2.0 version of the `oracle.sysman.db2` plug-in on the Management Agent named ost.example.com and also on other Management Agents that are dependent on the Management Agent named `myhost1.example.com`.

```
emcli redeploy_plugin_on_agent
      -agent_names="host.example.com:1838"
      -plugin="oracle.sysman.db2:12.1.0.2.0"
      -include_dependent_agents
```

refer_swlib_entity_files

Refers one or more files from an entity revision in the software library.

Format

```
emcli refer_swlib_entity_files
        -entity_rev_id="entity_rev_id"
        -file="<relative_file_path>[;<new_file_name>]" | [-removefile="<existing_
           file_name>"]
        -refer_storage="<storage_location_name>;<storage_type>"
        [-use_latest_revision]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_rev_id**

  Identifier of the entity revision. The Software Library home page exposes the identifier for folders and entities as a custom column (Internal ID) and is hidden by default.

- **file**

  Relative path of the file to be referred from the specified storage location. The file name stored in the software library is defaulted to the name of the file being referred. You can optionally specify a different file name, separated by a semi-colon (;).

- **removefile**

  Name of the file to be removed. This is an existing file carried forward from the specified entity revision.

  Alternatively, you can specify the following values:

  ALL — Remove all existing files (no carry forward).
  NONE — Retain all carried forward files.

  The default is NONE.

- **refer_storage**

  The storage location and type for referring to files, separated by a semi-colon (;). The location specified must be in 'active' status. The storage type can be Http, Nfs, or ExtAgent.

- **use_latest_revision**

  Indicates that the latest revision of the entity be used instead of the revision identified by entity_rev_id.

Example

This example refers the file 'scripts/perl/script1.pl' in the HTTP reference file location 'myScripts' from the entity revision identified. The file name associated will be 'new_script.pl'. The identifier of the updated revision is output.

```
emcli refer_swlib_entity_files
        -entity_rev_id="oracle:defaultService:em:provisioning:1:cmp:
         COMP_Component:SUB_Generic:B1B1880C6A8C62AAE040548C42832D14:0.1"
```

```
-file="scripts/perl/script1.pl;new_script.pl"
-refer_storage="myScripts;Http"
-use_latest_revision
```

refresh_coherence

Refreshes one or more Coherence clusters.

Format

```
emcli refresh_coherence
        -input_file=coherence_refresh_file:file_path
        [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Fully-qualified path to a CSV-formatted file listing Coherence cluster target per line. For example:

  ```
  ClusterA
  ClusterB
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Examples

This example reads the my_clusters_name.csv file to determine the clusters to be refreshed to Cloud Control, and then refreshes them.

```
emcli refresh_coherence
  -input_file=coherence_refresh_file:c:\emcli\my_clusters_names.csv
```

refresh_database

Refreshes the database from the latest data in the source database. This command places the database target under blackout and the database is deleted from the Oracle Home. The database is then recreated from the latest data in the source database and the target is removed from blackout.

> **Note:** This command only applies to full clone test master databases created using the Database Cloning wizard. It does not apply to thin clone databases.

Format

```
emcli refresh_database
      -target_name="database target name"
      -target_type="database target type"
      -input_file=data:"file:path"
```

Parameters

- target_name

  The target name of the database to be refreshed.

- target_type

  The target type of the database to be refreshed.

- input_file

  The input file containing parameters for the temporary staging location and all passwords to be set:

  – DB_TEMPLATE_STAGE

    The staging area used to store files transferred from the source host.

  – COMMON_DB_SYSTEM_PASSWORD

    The password to be set for SYSTEM user.

  – COMMON_DB_DBSNMP_PASSWORD

    The password to be set for the DBSNMP user.

  – COMMON_DB_SYS_PASSWORD

    The password to be set for SYS user.

  – ASMSYSPWD

    The ASM SYS password required to log in to ASM. This password is only required if the database files are on ASM.

Example

The following example refreshes the Oracle database with the name 'database' using the parameters contained in the `/tmp/a.txt` file:

```
emcli refresh_database
      -target_name="database"
      -target_type="oracle_database"
```

```
      -input_file=data:"/tmp/a.txt"
```

In this example, the `/tmp/a.txt` has the following content:

```
DB_TEMPLATE_STAGE=/tmp
COMMON_DB_SYSTEM_PASSWORD=welcome
COMMON_DB_DBSNMP_PASSWORD=welcome
COMMON_DB_SYS_PASSWORD=welcome
ASMSYSPWD=welcome
```

refresh_dbprofile

Creates a new snapshot under the specified database profile.

Format

```
emcli refresh_dbprofile
      -comp_loc="Database Profile component location in software library"
```

Parameters

- comp_loc

  A combination of the database profile location and name.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example creates a new snapshot of the database profile RMAN_Profile with the location `Database Provisioning Profiles/11.2.0.4.0/linux_x64`.

```
emcli refresh_dbprofile
      -comp_loc="Database Provisioning Profiles/11.2.0.4.0/linux_x64/RMAN_Profile"
```

refresh_fa

Refreshes a Fusion Application instance.

If the -delete_targets option is not passed, this verb submits a job to refresh all of the WebLogic Domains of the given Fusion Instance.

If the -delete_targets option is passed, this verb removes the targets that are not present.

If both -add_targets and -delete_targets options are passed, this verb adds, updates, and removes targets that are not present in the WebLogic Domains of the Fusion Instance.

Format

```
emcli refresh_fa
     -name=<Fusion_Instance_name>
     [-delete_targets]
     [-add_targets]

[ ] indicates that the parameter is optional
```

Parameters

- **name=<Name of the Fusion Instance>**

  Target name of the Fusion Application instance.

- **delete_targets**

  Deletes the specified Fusion Application instance targets from the Enterprise Manager Cloud Control monitoring framework. Deleting a target removes it from the Management Repository and does not physically remove the target itself.

- **add targets**

  Adds specified Fusion Application instance targets to be monitored by Enterprise Manager. The target type specified is checked on the Management Agent for existence and for required properties, such as user name and password for host target types, or log-in credentials for database target types. You must specify any required properties of a target type when adding a new target of this type.

Examples

This example refreshes the Fusion Application instance:

```
emcli refresh_fa -name=fa1
emcli refresh_fa -name=fa1 -delete_targets -add_targets
emcli refresh_fa -name=fa1 -delete_targets
```

refresh_wls

Enables/disables a refresh for one or more Oracle WebLogic Server Domains (target type --> weblogic_domain). This verb reads a file labeled domain_refresh_file in order to refresh the WebLogic Server. The domain_refresh_file is required; refresh cannot occur without it. You must create the file prior to performing refresh.

Format

```
emcli refresh_wls
        -input_file=domain_refresh_file:file_path
        [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Fully-qualified path of the CSV(Comma-Separated Values) file that contains multiple lines of the Target name and Refresh action (Enable/Disable refresh of the WLS domains/farms to be refreshed).

  Note the following advisory information about the format of domain_refresh_file:

  - The target name should be the fully-qualified name of the domain target.
  - Every target is treated as type weblogic_domain.
  - Valid values of the refresh option are "E", "D", and "R". "E" enables a refresh for the WLS Domain, "D" disables the refresh for the WLS Domain, and "R" removes targets that are deleted from the WebLogic Domain.
  - A comma (,) is used as the delimiter.
  - The total number of tokens in each line is fixed, and should be equal to 2.
  - The order of parameters is fixed. You must provide the parameters in the same order as specified below in the sample file structure for domain_refresh_file:

    ```
    /Farm01_base_domain/base_domain,D
    /Farm02_base_domain/base_domain,E
    /Farm03_base_domain/base_domain,R
    ```

    The first entry disables the refresh for target /Farm01_base_domain/base_domain, the second entry enables a refresh for target /Farm02_base_domain/base_domain, and the third entry removes targets from Enterprise Manager that are deleted from /Farm03_base_domain/base_domain.

    For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Example

```
$emcli refresh_wls
          -input_file=domain_refresh_file:/tmp/refresh/emcli/
             domain_refresh_file.csv -debug
```

register_forwarder_agents

Takes a list of agents and registers each agent as a forwarding agent.

Format

```
emcli register_forwarder_agents
      -agent_list="agent_list"
```

Parameters

- agent_list

  List of agents that need to be registered as forwarders. The agents must be separated by space.

Exit Codes

0 if successful. A non-zero value indicates that verb processing was unsuccessful.

Example

The following example registers agent1 and agent2 as forwarding agents.

```
emcli register_forwarder_agents
      -agent_list="agent1 agent2..."
```

reimport_swlib_metadata

Re-imports software library metadata from the OMS and deployed plug-in Oracle Homes. Any Oracle-owned entity with missing files is restored to the corresponding upload storage location.

Format

```
emcli reimport_swlib_metadata
```

Parameters

None.

relocate_bda_target

Relocates monitoring agents for BDA targets. Use it to relocate monitoring of a specific target to another agent on a destination host, or use it to relocate monitoring of all shared targets on a cluster to other agents on the same BDA rack.

Format

```
emcli relocate_bda_target
        -target="target_name" -dest_host="destination_host_name" | -all_shared
        -cluster="cluster_name">
```

Parameters

- **target**

  A target in the BDA network.

- **dest_host**

  Name of the host where monitoring of the specified target is to be relocated.

- **all_shared**

  Specifies to relocate monitoring of all shared targets in the named cluster.

- **cluster**

  Name of the cluster for whom monitoring is to be relocated.

Examples

Example 1

The following example relocates monitoring of the target `hdfs_USA_acme` to the agent on host `acme101.com`:

```
emcli relocate_bda_target
         -target="hdfs_USA_acme"
         -dest_host="acme101.com"
```

Example 2

The following example relocates monitoring of all shared targets on the cluster `acme101` to other valid agents on the same BDA rack:

```
emcli relocate_bda_target
         -all_shared
         -cluster="acme101"
```

relocate_targets

Moves all of the collections and blackouts for targets from the source Agent to the destination Agent, and makes the destination Agent the monitoring Agent for these targets in Enterprise Manager.

Format

```
emcli relocate_targets
       -src_agent=<source_agent_target_name>
       -dest_agent=<dest_agent_target_name>
       -target_name=<name_of_target_to_be_relocated>
       -target_type=<type_of_target_to_be_relocated>
       -copy_from_src
       -changed_param=<propName>:<propValue>
       -input_file:dupTargets=<targets_contents>
       -input_file:moveTargets="complete path to file containing targets with
         overridden property values"
       -copy_from_src [-changed_param=<propName>:<propValue>]*
       [-ignoreRelatedTargets]
       [-noHostColumnUpdate]
       [-ignoreTimeSkew=yes]
       [-changed_param=MachineName:mmmm ]
       [-force=yes]

[ ] indicates that the parameter is optional
```

Note: To relocate a composite target, you must specify the input_file:dupTargets , and you cannot combine -target_type or -target_name.

Note: For non-Sysman users, Full Any Target and Add Any Target privileges should be granted.

Modes

There are two modes for this verb:

- **Create Mode**

  This mode creates a list of targets on the destination Management Agent that already exists and is monitored by the source Management Agent in Enterprise Manager. It moves all the collections and blackouts for these targets from the source Management Agent to the destination Management Agent, and makes the destination Agent the monitoring Agent for these targets in Enterprise Manager.

  ```
  emcli relocate_targets -src_agent=<source_agent>
        -dest_agent=<destination_agent>
        -input_file=dupTarget:<complete_path_to_file>;
        [-ignoreTimeSkew=yes]
  ```

  Tip: See the Examples section for more samples of the create mode.

- **Exist Mode**

  In this mode, the target also exists at the destination.

```
emcli relocate_targets
      -src_agent=<source_agent_target_name>
      -dest_agent=<destination_agent_target_name>
      -target_name=<target_name>
      -target_type=<target_type>
      [-ignoreTimeSkew=yes]
      [-force=yes]
```

In all cases, relocation moves all collections and blackouts for these targets from the source Agent to destination Agent, and makes the destination Agent the monitoring Agent for these targets in Enterprise Manager.

Parameters

- **src_agent**

  Management Agent currently monitoring the targets. If srcAgent is not known, enter currentOwner as the argument.

- **dest_agent**

  Management Agent that should monitor the targets.

- **target_name**

  Name of the target that needs to be moved.

- **target_type**

  Type of target that needs to be moved.

- **changed_param**

  The value of the propName property in the target should be changed to propValue.

- **input_file=dupTargets**

  Takes a file name that contains all the targets and its properties as seen in `targets.xml`. The contents of the file must have the same format as `targets.xml`.

  To relocate a composite target, you must specify the input_file:dupTargets, and you cannot combine -target_type or -target_name.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **input_file=moveTargets**

  Takes a file name that contains a list of targets, one per line, in the following format:

  ```
  <targetType>:<targetName>[;<propName>=<propValue>]*
  ;lkj;lkj;lkj
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **copy_from_src**

  Copies target properties from the source Agent.

- **ignoreTimeSkew**

  If specified, the target is relocated, ignoring the time skew between the source and destination Agent.

- **ignoreRelatedTargets**

  Moves related targets when not specified. Specified to move onlhy the targets on the command line.

- **noHostColumnUpdate**

  Preserves the host of the relocated tartet when specified. Otherwise, the host is updated to be the new Agent's host.

- **changed_param**

  Specify the new MachineName as part of relocate operation, as it is different for each host.

- **force**

  If the command is executed with the `-force=yes` switch, the composite target is automatically relocated with its related targets. If the command is executed without this switch, an error message appears if it is a composite target.

Output

Output message of the command execution.

Examples

Example 1

The following Create Mode example creates a target on the destination Agent by copying the target property content from the source Agent, while allowing some property values to be changed.

```
emcli relocate_targets
      -src_agent=<source_agent>
      -dest_agent=<destination_agent>
      -target_name=<target_name>
      -target_type=<target_type>
      -copy_from_src
      [-ignoreTimeSkew=yes]
      [-changed_param=<Propname>:<Value>]*
```

Example 2

The following Create Mode example creates a list of targets on the destination Agent specified in the moveTargets file. You can specify property value overrides.

```
emcli relocate_targets
      -src_agent=<source_agent>
      -dest_agent=<destination_agent>
      -input_file=moveTargets:<complete_file_path>
      [-ignoreTimeSkew=yes]
```

Example 3

This example creates a list of targets on the destination Agent that already exists and is monitored by the source Agent in Enterprise Manager.

```
emcli relocate_targets
       -src_agent=<source agent>
       -dest_agent=<destination agent>
       {-ignoreTimeSkew=yes}
       -input_file=dupTarget:<complete file path>;
```

Example 4

The following example creates a target on the destination Agent by copying the target property content from the source Agent while allowing some property values to be changed.

```
emcli relocate_targets
        -src_agent=<source agent>
        -dest_agent=<destination agent>
        -target_name=<target name>
        -target_type=<target type>
        -copy_from_src
        {-ignoreRelatedTargets}
        {-noHostColumnUpdate}
        {-ignoreTimeSkew=yes}
        [-changed_param=<Propname>:<Value>]*
```

Example 5

This example creates a list of targets on the destination Agent specified in the moveTargets file. You can specify property value overrides.

```
emcli relocate_targets
        -src_agent=<source agent>
        -dest_agent=<destination agent>
        {-ignoreTimeSkew=yes}
        -input_file=moveTargets:<complete file path>;
```

Example 6

This example relocates a database target with the MachineName property.

```
emcli relocate_targets
        -src_agent=source.example.com:1830
        -dest_agent=destination.example.com:1830
        -target_name=ABC
        -target_type=oracle_database
        -copy_from_src
        -force=yes
        -ignoreTimeSkew=yes
        -changed_param=MachineName:dbsdpb2-vip.unix.lch.com
```

remove_beacon

Removes a beacon from the monitoring set of beacons.

Format

```
emcli remove_beacon
      -name=<target_name>
      -type=<target_type>
      -bcnName=<beacon_name>

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **bcnName**

  Beacon name to remove.

Examples

This example removes MyBeacon from the MyTarget service target of type generic_service.

```
emcli remove_beacon -name='MyTarget' -type='generic_service'
      -bcnName='MyBeacon'
```

remove_chargeback_entity

Removes the given entity from Chargeback.

Format

```
remove_chargeback_entity
     -entity_name="eName"
     -entity_type="eType"
     -[ entity_guid="entity guid" ]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_name**

  Name of the entity to be removed from Chargeback.

- **entity_type**

  Type of entity to be removed from Chargeback.

- **entity_guid**

  guid of the entity to be removed to Chargeback.

  When more than one entity is active in Chargeback with the given entity name and entity type, the command lists all such entities with additional details such as creation date, parent entity name, entity guid, and so forth to choose the correct entity. Select the correct entity from the given list and execute the command again with entity guid as the parameter instead of entity name and entity type.

Examples

This example removes "db1", an oracle_database entity, from Chargeback.

```
emcli remove_chargeback_entity -entity_name="db1" -entity_type="oracle_database"
```

See Also

add_chargeback_entity
assign_charge_plan
assign_cost_center
list_chargeback_entities
list_chargeback_entity_types
list_charge_plans
list_cost_centers
unassign_charge_plan
unassign_cost_center

remove_cs_target_association

Removes the specified standard target associations.

Note: When the standard is provided by Oracle, the <std_name> is the standard internal name.

Format

```
remove_cs_target_association
 -name="<std_name>"
 -version="<std_version>"
 -author="<author_name>"
 -target_list="<target_name>[,<target_name>]*"
 -target_list_file="<file_name>"
```

Parameters

- name

  Name of the standard.

- version

  Version of the standard.

- author

  Author of the standard.

- target_list

  Name of the target. Use this parameter when removing the compliance standard association from a small number of targets. Targets are separated by commas. When providing a group target, it should be appended with ":Group". Examples are:

  ```
  -target_list="slc0host"
  -target_list="slc0host,slc-host01"
  -target_list="slc0host,host_grps:Group"
  ```

- target_list_file

  Name of the file that contains the list of targets. The targets can be either comma-separated values or in a file where the targets are listed on separate lines. Examples are:

  ```
  -target_list_file=slc0host,slc0host1,slc0host02
  -target_list_file="slc0host.txt" Where slc0host.txt contains the following
  lines:
       slc0host
       slc0host01
       slc0host02
  ```

  Note: Use either the target_list option or the target_list_file option.

Examples

Example 1

The following example removes the standard target association named "secure configuration for host" and uses the target_list option to remove the targets associated with the standard.

```
emcli remove_cs_target_association
    -name="secure configuration for host"
    -version="1"
    -author="sysman"
    -target_list="host1,host2"
```

Example 2

The following example removes the standard target association named "secure configuration for host" and uses the target_list_file option to remove the targets associated with the standard. The targets listed in the file are either comma separated values or each target is listed on a separate line.

```
emcli remove_cs_target_association
    -name="secure configuration for host"
    -version="1"
    -author="sysman"
    -target_list_file="file with target name list"
```

remove_service_system_assoc

Removes the system for a given service.

Format

```
emcli remove_service_system_assoc
      -name='name'
      -type='type'
```

Parameters

- **name**

  Service name.

- **type**

  Service type.

Examples

This example removes the system for the generic service named my service.

```
emcli remove_service_system_assoc
      -name='my service' -type='generic_service'
```

remove_swlib_storage_location

Removes a storage location from the software library. The alternate storage location where the existing files need to be migrated should also be specified. For upload file storage types, OMS shared and the OMS Agent file system, a job is submitted to perform the migration of files, subsequent to which the location is removed. For these upload file storage types, the alternate location need not be of the same storage type, which is not the case for locations of referenced file storage types.

Format

```
emcli remove_swlib_storage_location
        -name="src_location_name"
        -type="OmsShared|OmsAgent|Http|Nfs|ExtAgent"
        -migrate_to_loc="dest_location_name"
        [-migrate_to_type="OmsShared|OmsAgent|Http|Nfs|ExtAgent"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the storage location to be removed.

- **type**

  Type of storage location, which can be one of:

  ```
  OmsShared
  OmsAgent
  Http
  Nfs
  ExtAgent
  ```

- **migrate_to_loc**

  Name of the alternate storage location where existing files need to be migrated.

- **migrate_to_type**

  Type of the alternate storage location, which can be one of:

  ```
  OmsShared
  OmsAgent
  Http
  Nfs
  ExtAgent
  ```

  The default is the storage type of the location being removed.

  Note: This option can be different from the type option specified only for OmsShared and OmsAgent storage types. For all other storage types, migrating files across storage types is not supported, and therefore, type and migrate_to_type (if specified) must be the same.

Examples

Example 1

This example removes an OMS shared file system storage location named 'myOMSSharedLocation' and migrates all of its files to another OMS shared file system storage location named 'myNewOMSSharedLocation'. A job is submitted for performing the file migration. The location being removed will be moved to 'Inactive' status during file migration and subsequently removed.

```
emcli remove_swlib_storage_location
        -name="myOMSSharedLocation"
        -type="OmsShared"
        -migrate_to_loc="myNewOMSSharedLocation"
```

Example 2

This example removes an OMS shared file system storage location named 'myOMSSharedLocation' and migrates all of its files to an OMS Agent file system storage location named 'myNewAGTLocation'. A job is submitted for performing the file migration. The location being removed will be moved to 'Inactive' status during file migration and subsequently removed.

```
emcli remove_swlib_storage_location
        -name="myOMSSharedLocation"
        -type="OmsShared"
        -migrate_to_loc="myNewAGTLocation"
        -migrate_to_type="OmsAgent"
```

Example 3

This example removes an HTTP storage location named 'myHTTPLocation' and migrates all of its files to another HTTP storage location named 'myNewHTTPLocation'.

```
emcli remove_swlib_storage_location
        -name="myHTTPLocation"
        -type="Http"
        -migrate_to_loc="myNewHTTPLocation"
```

remove_target_from_rule_set

Removes a target from an enterprise rule set.

Privilege Requirements: Super Administrators can add a target to any enterprise rule set except for predefined (out-of-box) rule sets supplied by Oracle.

Only the owner or co-author of a rule set can add a target to it.

Format

```
emcli remove_target_from_rule_set
      -rule_set_name="rule set name"
      -target_name="target name"
      -target_type="internal name for target type"
      [-rule_set_owner=<ruleset owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **rule_set_name**

  Name of an enterprise rule set. This option only applies to rule sets associated with a list of targets.

- **target_name**

  Name of the target to be removed.

- **target_type**

  Type of the target to be removed. For example, *host*.

- **rule_set_owner**

  Optionally, you can specify the owner of the rule set.

Examples

The following example removes the host target *myhost.com* from a rule set named *rules*. This rule set is owned by the administrator sysman.

```
emcli remove_target_from_rule_set -rule_set_name='rules' -target_name='myhost.com'
-target_type='host' -rule_set_owner='sysman'
```

remove_target_property

Removes the target property from all targets of the specified target type. This also removes all values associated with this target property.

Format

```
emcli remove_target_property
      -target_type="target_type"
      -property="property_name"
```

Parameters

- **target_type**

  Target type for which you want to remove this property. To remove this property from all target types for which it is defined, you can specify the "*" wildcard character.

- **property**

  Name of the property you want to remove. Property names are case-sensitive. You cannot remove the following Oracle-provided target properties:

  Comment, Deployment Type, Line of Business, Location, Contact

Examples

Example 1

This example removes the target property Owner from all targets of type oracle_database. This also removes all values associated with this target property.

```
emcli remove_target_property  -target_type="oracle_database" -property="Owner"
```

Example 2

This example removes the target property Owner from all targets. This also removes all values associated with this property for all target types.

```
emcli remove_target_property  -target_type="*" -property="Owner"
```

remove_update

Removes an update.

Format

```
emcli remove_update
        -id="internal id"
```

Parameters

- **id**

  Internal identification for the update to be removed.

Examples

This example submits a job to remove the update, and prints the job execution ID upon submission.

```
emcli remove_update
      -id="914E3E0F9DB98DECE040E80A2C5233EB"
```

rename_service_template

Renames a Service Template.

Format

```
emcli rename_service_template
      -name_old="<Current_Name_of_Service_Template>"
      -name_new="<New_Name_of_Service_Template>"
      -service_family="<Name_of_Service_Family>"
```

Parameters

- **name_old**

  Current name of the Service Template.

- **name_new**

  New name of the Service Template.

- **service_family**

  Name of the Service Family.

Examples

```
emcli rename_service_template
      -name_old="Web_Logic"
      -name_new="Web_Logic_V1"
      -service_family="MWAAS"
```

displays the following output:

```
Service Template renamed from "Web_Logic" to "Web_Logic_V1" successfully
```

rename_target

Renames the repository-side target.

Format

```
emcli rename_target
      -target_type=<type1>
      -target_name=<old_target1>
      -new_target_name=<new_target1>
```

Parameters

- **target_type**

  Target type of the target being renamed.

- **target_name**

  Existing name of the target.

- **new_target_name**

  New name of the target.

Examples

This example renames the repository-side target.

```
emcli rename_target
      -target_type="oracle_em_service"
      -target_name="TestService1"
      -new_target_name="NewTestService1"
```

reschedule_instance

Reschedules a submitted procedure instance. You can only reschedule scheduled instances.

Format

```
emcli reshedule_instance
      -instance=<instance_guid>
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
      -schedule=
          start_time:yyyy/MM/dd HH:mm;
          [tz:<java_timezone_ID>];
          [grace_period:xxx]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  GUID of the instance to execute.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

- **schedule**

  Schedule for the procedure instance:

  – **start_time** — When the procedure should start.

  – **tz —** Optional time zone ID.

  – **grace_period** — Optional grace period in minutes.

Examples

```
emcli reschedule_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-schedule="start_time:2011/8/21 21:23;tz:America/New_York;grace_period:60"
```

resecure_agent

Resecures a Management Agent already secured. This verb requires operator privilege or full privilege on the Management Agent.

Format

```
emcli resecure_agent
        -agent_name="agent_target_name"
        -registration_pwd="registration_password"
        [-host_username="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name="credential_name"]
        [-credential_setname="credential_setname_of_agent"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **registration**

  Registration password to securely communicate with OMS.

- **host_username**

  User name of the OS user (on the host) who owns the Management Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Management Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Management Agent. Example: "HostCreds".

Examples

Example 1

```
emcli resecure_agent -agent_name="agent.example.com:1234"
                          -registration_pwd="test_pwd"
                          -host_username="test_user"
                          -host_pwd="test"
```

Example 2

```
emcli resecure_agent -agent_name="agent.example.com:1234"
                          -registration_pwd="test_pwd"
                          -credential_name="MyMachineCredential"
```

Example 3

```
emcli resecure_agent -agent_name="agent.example.com:1234"
                          -registration_pwd="test_pwd"
                          -credential_setname="HostCreds"
```

restart_agent

Restarts a Management Agent. This verb requires operator privilege or full privilege on the Management Agent.

Format

```
emcli restart_agent
        -agent_name="agent_target_name"
        [-host_username="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name="credential_name"]
        [-credential_setname="credential_setname_of_agent"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **host_username**

  User name of the OS user (on the host) who owns the Management Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Management Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Management Agent. Example: "HostCreds".

Examples

Example 1

```
emcli restart_agent -agent_name="agent.example.com:1234"
                          -host_username="test_user"
                          -host_pwd="test"
```

Example 2

```
emcli restart_agent -agent_name="agent.example.com:1234"
                          -credential_name="MyMachineCredential"
```

Example 3

```
emcli restart_agent -agent_name="agent.example.com:1234"
                          -credential_setname="HostCreds"
```

resume_instance

Resumes a suspended deployment instance.

Format

```
emcli resume_instance
      -instance=<instance_guid>
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  GUID of the instance.

- **exec**

  GUID of the execution.

- **name**

  Name of the execution.

- **owner**

  Owner of the execution.

Examples

```
emcli resume_instance -instance=16B15CB29C3F9E6CE040578C96093F61
```

resume_job

Resumes a job or set of jobs. Resumes job executions on any of the targets scheduled to start within the beginning and ending time window.

> **Note:** Suspend and resume operate either at the job or the execution level, but not both. If job executions were previously suspended, they must be resumed by execution matching. If a job was suspended, it must be resumed by job matching; it is not be possible to resume it by executions.

Format

```
emcli resume_job
   [-name="job_name_pattern"]
   [-owner="job_owner"]
   [-type="job_type"]
   [-targets="target_name:target_type"]
   [-input_file=property_file:"filename"]
   [-preview]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name or pattern of the job(s) to resume.

- **owner**

  Owner of the job(s).

- **type**

  Job type of the job(s).

- **targets**

  Target name and target type of the job(s).

- **input_file**

  Specify the filtering properties of the file in "filename."

  Any jobs matching all the specified filter criteria are resumed. You must specify at least one filter, and the logged in administrator must have the necessary privileges on the matching jobs.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **preview**

  Only lists the jobs that would be resumed.

Examples

Example 1

This example resumes a job named MYJOB.

```
emcli resume_job -name=MyJob
```

Example 2

This example resumes all jobs owned by User1.

```
emcli resume_job -owner=User1
```

Example 3

This example resumes all jobs of type Backup whose name starts with BK.

```
emcli resume_job -name=BK% -type=Backup
```

Example 4

This example resumes all jobs on db target orcl_123.

```
emcli resume_job -targets=orcl_123:oracle_database
```

Example 5

This example resumes jobs or job executions matching search criteria in suspend_prop.txt.

```
resume_job -input_file=property_file:/tmp/suspend_prop.txt
```

If the same file is used for both suspend and for resume, the set of jobs or executions resumed should overlap, but might not be identical. The criteria may match more or fewer jobs or executions than previously.

retry_add_host

Retries a failed add host session.

Format

```
emcli retry_add_host
        -session_name="session_name"
        -retry_using_same_inputs | -update_inputs_and_retry"
        [-host_names="host_names"]
        [-platform="platform_id"]
        [-installation_base_directory="installation_base_directory"]
        [-credential_name="credential_name"]
        [-credential_owner="credential_owner"]
        [-instance_directory="instance_directory"]
        [-port="agent_port"]
        [-deployment_type="type_of_agent_deployment"]
        [-privilege_delegation_setting="privilege_delegation_setting"]
        [-additional_parameters="parameter1 parameter2 ..."]
        [-source_agent="source_agent"]
        [-master_agent="master_agent"]
        [-preinstallation_script="preinstallation_script"]
        [-preinstallation_script_on_oms]
        [-preinstallation_script_run_as_root]
        [-postinstallation_script="postinstallation_script"]
        [-postinstallation_script_on_oms]
        [-postinstallation_script_run_as_root]
        [-wait_for_completion]

[ ] indicates that the parameter is optional
```

Parameters

- **session_name**

  Name of the session you want to retry.

- **retry_using _same_inputs**

  Retries the Add Host session using the same inputs.

- **update_inputs_and_retry**

  Updates the inputs and retries the Add Host session.

- **host_names**

  Names of the hosts where the Agents need to be installed, separated by a semicolon.

- **platform**

  ARU platform ID of the hosts where the Agent needs to be installed.

- **installation_base_directory**

  Directory where you want to install the Agent. Provide this parameter in double-quotes if it is an MS-DOS/Windows style path.

- **credential_name**

  Named credential to be used for installing the Agent.

- **credential_owner**

Owner of the named credential.

- **instance_directory**

  Instance directory of the Agent. Provide this parameter in double-quotes if it is an MS-DOS/Windows style path.

- **port**

  Port on which the Agent should communicate with the OMS.

- **deployment_type**

  Type of Agent deployment, which can be FRESH, CLONE, or SHARED. By default, it is the deployment type of the failed session you want to retry.

- **privilege_delegation_setting**

  Privilege delegation setting you want to use for installing an Agent and running the root script.

- **additional_parameters**

  Additional parameters you want to use for installing an Agent.

- **source_agent**

  Source Agent you want to use for installing a cloned Agent.

- **master_agent**

  Master Agent you want to use for installing a shared Agent.

- **preinstallation_script**

  Script you want to run before installing the Agent. Provide this parameter in double-quotes if it is an MS-DOS/Windows style path.

- **preinstallation_script_run_as_root**

  Use this option if you want to run the pre-installation script as the root user.

- **preinstallation_script_on_oms**

  Use this option if the pre-installation script resides on the OMS host.

- **postinstallation_script**

  Script you want to run after installing the Agent. Provide this parameter in double-quotes if it is an MS-DOS/Windows style path.

- postinstallation_script_on_oms

  Use this option if the post-installation script resides on the OMS host.

- **postinstallation_script_run_as_root**

  Use this option if you want to run the post-installation script as the root user.

- **wait_for_completion**

  Runs the Add Host operation synchronously.

Examples

Example 1

This example retries the session 'ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_AM_PST' using the same inputs.

```
emcli retry_add_host session_name='ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_AM_PST'
-retry_using_same_inputs
```

Example 2

This example retries the session 'ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_AM_PST' by updating the input port to 5678.

```
emcli retry_add_host session_name='ADD_HOST_SYSMAN_Dec_17_2012_2:02:28_AM_PST'
-update_inputs_and_retry -port=5678
```

retry_instance

Retries a failed instance or failed step.

Format

```
emcli retry_instance
      [-instance=<instance_guid>]
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
      [-stateguid=<state_guid>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  GUID of the instance.

- **exec**

  GUID of the execution.

- **name**

  Name of the execution.

- **owner**

  Owner of the execution.

- **stateguid**

  Comma-separated list of state GUIDs.

Examples

```
emcli retry_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid=51F762417C4943DEE040578C4E087168

emcli retry_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid='51F762417C4943DEE040578C4E087168,51F762417C4944DEE040578C4E087168'
```

retry_job

Restarts a previously failed job execution.

Format

```
emcli retry_job
      -exec_id="executionID"
      [-noheader]
      [-script | -format=
            [name:<pretty|script|csv>];
            [column_separator:"column_sep_string"];
            [row_separator:"row_sep_string"];
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **exec_id**

  ID of the job execution to be retried. Use the get_jobs verb to obtain specific job execution IDs.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.
  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.
  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.
  - `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.
  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns:

Execution ID

Examples

This example restarts the job execution with Id 12345678901234567890123456789012 and displays a new execution ID.

```
emcli retry_job -exec_id=12345678901234567890123456789012
```

revoke_bipublisher_roles

Revokes roles for accessing the BI Publisher catalog.

Format

```
emcli revoke_bipublisher_roles
      (-roles="role1[;role2;...role_n]"
      [-users="user"]
      [-external_role="group"])

[ ] indicates that the parameter is optional
```

Parameters

- **roles**

  Revokes one or more roles from the BI Publisher. Specify one or more roles separated by a semicolon.

- **users**

  Users to revoke the granted role.

- **external_role**

  Name of the external group to apply the revocation.

Examples

Example 1

This example revokes one role from a group.

```
emcli revoke_bipublisher_roles -roles="EMBIPViewer" -external_role="TESTGROUPNAME"
```

Example 2

This example revokes more than one role from a group.

```
emcli revoke_bipublisher_roles -roles="EMBIPViewer;EMBIPAuthor"
      -external_role="TESTGROUPNAME"
```

Example 3

This example revokes one role from a user.

```
emcli revoke_bipublisher_roles -roles="EMBIPViewer"
      -users="TESTUSERNAME"
```

Example 4

This example revokes one role from multiple users.

```
emcli revoke_bipublisher_roles -roles="EMBIPViewer"
      -users="TESTUSERNAME;TESTUSERNAME2"
```

Example 5

This example revokes more than one role from multiple users and a group.

```
emcli revoke_bipublisher_roles -roles="EMBIPViewer;EMBIPAuthor"
      -external_role="TESTGROUPNAME"
      -users="TESTUSERNAME;TESTUSERNAME2
```

revoke_license_no_validation

Revokes licenses on a set of user-specified packs, or all packs to a set of user-specified targets, or all targets belonging to the input licensable target type.

For 11g database targets, you cannot enable or disable the Database Diagnostic and Tuning Packs through the user interface. You need to set the control_management_pack_access initialization parameter to manage your licenses. For information about this parameter, see the Enterprise Database Management chapter of *Oracle Enterprise Manager Licensing Information*.

> **Tip:** You can use this verb to revoke licenses for standalone target types, such as hosts and databases, but you cannot use this verb to revoke licenses for the parent Application Server (oracle_ias) target type, which has dependent target types of OC4J, Jserv, Web Cache, and so forth. To do this, use the revoke_license_with_validation verb instead.
>
> For example, for pack ias_config and an Application Server target of AS1 with an associated dependent target of OC4J1, this verb revokes the license to AS1, but this does not propagate to OC4J1.

Format

```
emcli revoke_license_no_validation
      -type="target_type"
      [-targets="tname1;tname2;..."]
      [-packs="pack1;pack2;..."]
      [-file="file_name"]
      [-displayAllMessages]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Target type as it exists in the database. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks. You can specify only one target type at a time; for example, -type="oracle_database".

- **targets**

  Targets should be specified in the following sequence:

  ```
  TargetName1;TargetName2;
  ```

  For example:

  ```
  -targets="database1;database2;database3;"
  ```

  The semi-colon (;) is the target separator.

  See the "Examples" section below for information about providing arguments for the targets .

- **packs**

  License packs should be specified in the following sequence:

  ```
  pack1;pack2;
  ```

For example:

```
-packs="db_diag;db_config;"
```

The semi-colon (;) is the pack separator.

See the "Examples" section below for information about providing arguments for the pack .

- **file**

  Specify the file name, including the complete path. For example:

  ```
  -file="/usr/admin1/db_license.txt"
  ```

  The file should contain the list of targets and packs according to the following cases:

  – If you only need to provide a list of targets, use the following format:

    ```
    targets=database1;database2;database3;
    ```

  – If you only need to provide a list of packs, use the following format:

    ```
    packs=db_diag;db_config;
    ```

  – If you need to provide a list of both targets and packs, use the following format:

    ```
    targets=database1;database2;database3;
    packs=db_diag;db_config;
    ```

- **displayAllMessages**

  Displays all messages. Only error messages are displayed by default. "=value" is not allowed on the command line.

Examples

Example 1 and Example 2 below revoke licenses of specific packs for specific targets. In order to know which target types and pack names you can pass as arguments, you can use the view named mgmt_license_view to see a list of licensable targets, their target types, and the list of packs licensed on them.

To obtain this information, do the following:

1. Access SQL*Plus with your username and password, using sysman or other user that has access to sysman.mgmt_license_view.
2. Select a distinct pack name from sysman.mgmt_license_view, where:

   ```
   target_type=<oracle_database>
   ```

This example shows pack names for an Oracle database you specify as the target type.

```
PACK_NAME
-------------------
db_config
provisioning
db_sadm
db_tuning
db_diag
provisioning_db
db_chgmgt
```

```
7 rows selected.
```

Based on this information, to revoke a license to the database1 target for the db_chgmgt pack, you would enter the following command:

```
emcli revoke_license_no_validation -type="oracle_database" -targets="database1"
-packs="db_chgmgt"
```

The only limitation of mgmt_license_view is that it only lists the packs for a target type where the pack is granted to at least one target of that type. That is, if the pack is not granted to any target of that type, mgmt_license_view cannot provide any information.

Example 1

This example revokes the license of the db_diag and db_config packs to database1, database2, and database3 targets (oracle_database target type):

```
emcli revoke_license_no_validation -type="oracle_database"
          -targets="database1;database2;database3;" -packs="db_diag;db_config;"
```

Example 2

This example revokes the license of the db_diag and db_config packs to all database targets in the setup:

```
emcli revoke_license_no_validation -type="oracle_database"
       -packs="db_diag;db_config;"
```

Example 3

This example revokes the license of all packs (applicable to database targets) to database1, database2, and database3 targets in the setup:

```
emcli revoke_license_no_validation -type="oracle_database"
         -targets="database1;database2;database3;"
```

Example 4

This example revokes the license of all packs (applicable to database targets) to all database targets in the setup:

```
emcli revoke_license_no_validation -type="oracle_database"
```

Example 5

This example uses a text file to pass targets and pack names as the argument. It revokes the license of the db_diag and db_config packs to the database1, database2, and database3 targets (oracle_database target type):

```
emcli revoke_license_no_validation -type="oracle_database"
          -file="/usr/admin1/db_license.txt"
             targets=database1;database2;database3;
                   packs=db_diag;db_config;
```

where the content of the "/usr/admin1/license/db_license.txt" file is as follows:

```
targets=database1;database2;database3;
packs=db_diag;db_config;
```

revoke_license_with_validation

Revokes licenses on a set of user-specified packs, or all packs to a set of user-specified targets, or all targets belonging to the input licensable target type as per business rules.

For 11g database targets, you cannot enable or disable the Database Diagnostic and Tuning Packs through the user interface. You need to set the control_management_pack_access initialization parameter to manage your licenses. For information about this parameter, see the Enterprise Database Management chapter of *Oracle Enterprise Manager Licensing Information*.

> **Tip:** You can use this verb to revoke licenses for standalone target types, such as hosts and databases, and you also use this verb to revoke licenses for the parent Application Server (oracle_ias) target type, which has dependent target types of OC4J, Jserv, Web Cache, and so forth.
>
> For example, for pack ias_config and an Application Server target of AS1 with an associated dependent target of OC4J1, this verb revokes the license to AS1 and also propagates to OC4J1 (and all other dependent targets associated with AS1).
>
> To revoke licenses for only standalone target types, use the revoke_license_no_validation verb.

Format

```
emcli revoke_license_with_validation
      -type="target_type"
      [-targets="tname1;tname2;..."]
      [-packs="pack1;pack2;..."]
      [-file="file_name"]
      [-displayAllMessages]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Target type as it exists in the database. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks. You can specify only one target type at a time; for example, -type="oracle_database".

- **targets**

  Targets should be specified in the following sequence:

  ```
  TargetName1;TargetName2;
  ```

  For example:

  ```
  -targets="database1;database2;database3;"
  ```

  The semi-colon (;) is the target separator.

  See the "Examples" section below for information about providing arguments for the targets .

- **packs**

  License packs should be specified in the following sequence:

```
pack1;pack2;
```

For example:

```
-packs="db_diag;db_config;"
```

The semi-colon (;) is the pack separator.

See the "Examples" section below for information about providing arguments for the packs.

- **file**

  Specify the file name, including the complete path. For example:

  ```
  -file="/usr/admin1/db_license.txt"
  ```

  The file should contain the list of targets and packs according to the following cases:

  – If you only need to provide a list of targets, use the following format:

    ```
    targets=database1;database2;database3;
    ```

  – If you only need to provide a list of packs, use the following format:

    ```
    packs=db_diag;db_config;
    ```

  – If you need to provide a list of both targets and packs, use the following format:

    ```
    targets=database1;database2;database3;
    packs=db_diag;db_config;
    ```

- **displayAllMessages**

  Displays all messages. Only error messages are displayed by default. "=value" is not allowed on the command line.

Examples

Example 1 and Example 2 below revoke licenses of specific packs for specific targets. In order to know which target types and pack names you can pass as arguments, you can use the view named mgmt_license_view to see a list of licensable targets, their target types, and the list of packs licensed on them.

To obtain this information, do the following:

1. Access SQL*Plus with your username and password, using sysman or other user that has access to sysman.mgmt_license_view.
2. Select a distinct pack name from sysman.mgmt_license_view, where:

   ```
   target_type=<oracle_database>
   ```

This example shows pack names for an Oracle database you specify as the target type.

```
PACK_NAME
-------------------
db_config
provisioning
db_sadm
db_tuning
db_diag
provisioning_db
```

```
db_chgmgt

7 rows selected.
```

Based on this information, to revoke a license to the database1 target for the db_chgmgt pack, you would enter the following command:

```
emcli revoke_license_with_validation -type="oracle_database" -targets="database1"
-packs="db_chgmgt"
```

The only limitation of mgmt_license_view is that it only lists the packs for a target type where the pack is granted to at least one target of that type. That is, if the pack is not granted to any target of that type, mgmt_license_view cannot provide any information.

Example 1

This example revokes the license of the db_diag and db_config packs to database1, database2, and database3 targets (oracle_database target type):

```
emcli revoke_license_with_validation -type="oracle_database"
          -targets="database1;database2;database3;" -packs="db_diag;db_config;"
```

Example 1

This example revokes the license of the db_diag and db_config packs to database1, database2, and database3 targets (oracle_database target type):

```
emcli revoke_license_with_validation -type="oracle_database"
          -targets="database1;database2;database3;" -packs="db_diag;db_config;"
```

Example 2

This example revokes the license of the db_diag and db_config packs to all database targets in the setup:

```
emcli revoke_license_with_validation -type="oracle_database"
       -packs="db_diag;db_config;"
```

Example 3

This example revokes the license of all packs (applicable to database targets) to database1, database2, and database3 targets in the setup:

```
emcli revoke_license_with_validation -type="oracle_database"
          -targets="database1;database2;database3;"
```

Example 4

This example revokes the license of all packs (applicable to database targets) to all database targets in the setup:

```
emcli revoke_license_with_validation -type="oracle_database"
```

Example 5

This example uses a text file to pass targets and pack names as the argument. It revokes the license of the db_diag and db_config packs to the database1, database2, and database3 targets (oracle_database target type):

```
emcli revoke_license_with_validation -type="oracle_database"
          -file="/usr/admin1/db_license.txt"
             targets=database1;database2;database3;
                   packs=db_diag;db_config;
```

where the content of the "/usr/admin1/license/db_license.txt" file is as follows:

```
targets=database1;database2;database3;
packs=db_diag;db_config;
```

revoke_privs

Revokes the privileges from an existing Enterprise Manager user or Enterprise Manager role.

Format

```
emcli revoke_privs
        -name="username|rolename"
        [-privilege="name[;secure-resource-details]]"
        [-separator=privilege="sep_string"]
        [-subseparator=privilege="subsep_string"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  User name or role name from which privileges will be revoked.

- **privilege**

  Privilege to grant to this administrator. You can specify this option more than once. The original administrator privileges will be revoked. Specify <secure_resource_details> as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
  name2=resource_column_value2]..]"
  ```

- **separator**

  Specify a string delimiter to use between name-value pairs for the value of the -privilege option. The default separator delimiter is a semi-colon (;).

- **subseparator**

  Specify a string delimiter to use between name and value in each name-value pair for the value of the -privilege option. The default subseparator delimiter is a colon (:).

Examples

Example 1

For user1, This example revokes full control of the jobs with ID 923470234ABCDFE23018494753091111, and revokes full control on the target host1.example.com:host:

```
emcli revoke_privs
          -name="user1"
          -privilege="FULL_JOB;923470234ABCDFE23018494753091111"
          -privilege="FULL_TARGET;host1.example.com:host"
```

Example 2

This example revokes the target privileges from Enterprise Manager role Role1:

```
emcli revoke_privs
          -name="Role1"
          -privilege="FULL_TARGET;host1.example.com:host"
```

revoke_roles

Revokes the roles to an existing Enterprise Manager user or Enterprise Manager role.

Format

```
emcli revoke_roles
      -name="username|rolename"
      [-roles="role1;role2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  User name or role name from which roles will be revoked.

- **roles**

  Roles, which will be revoked from the Enterprise Manager user or role. You can specify this option more than once.

Examples

```
emcli revoke_roles
        -name="user1"
        -roles="SUPER_USER"

emcli revoke_roles
        -name="Role1"
        -roles="BLACKOUT_ADMIN;MAINTAIN_TARGET"
```

run_avail_diag

Runs diagnostics for an availability algorithm for a test-based service. This is mostly useful when the "last calculated" time stamp is running behind the current time, and the service status has been unresponsive for some time.

Format

```
emcli run_avail_diag
      -name=<target_name>
      -type=<target_type>
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

Examples

```
emcli run_avail_diag  -name='MyTarget' -type='generic_service'
```

run_config_seaches

DRuns a configuration search using a specified search name.

Format

```
emcli run_config_search
      -search_name="<Configuration Search UI Name>"
      [-target_name="<target name>"]
      [-on_host="<hostname>"]
      [-memberof="<group name>"]
      [-format=name:<pretty|script|csv>;
      [column_separator:"column_sep_string"];
      [row_separator:"row_sep_string"];

[ ] indicates that the parameter is optional.
```

Parameters

- search_name

  A display name for the configuration search.

- target_name

  Name of the target. It can be a full value or a pattern match using "%".

- on_host

  Name of the host where the target is running. It can be a full value or a pattern match using "%".

- memberof

  Group name or composite name of which the target is a member. It can be a full value or a pattern match using "%".

- format

  Format specification (default is -format="name:pretty").

  – format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  – format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings may be specified to change these defaults.

  – format="name:csv" sets the column separator to a comma and the row separator to a newline.

    format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

- column_separator

Specifies the column separator.

- row_separator

  Specifies the row separator.

Examples

Example 1

The following command shows the results of the search named "Search File Systems on Hosts":

```
emcli run_config_search
      -search_name="Search File Systems on Hosts"
```

Example 2

The following command shows the results of the search named "Search File Systems on Hosts" where the target name starts with "oracle":

```
emcli run_config_search
      -search_name="Search File Systems on Hosts"
      -target_name="oracle%"
```

Example 3

The following command shows the results of the search named "Search File Systems on Hosts" where the target name starts with "oracle" and the host name containing the pattern "host" are members of the group "group1":

```
emcli run_config_search
      -search_name="Search File Systems on Hosts"
      -target_name="oracle%"
      -on_host="%host%"
      -memberof="group1"
```

run_fa_diagnostics

Runs diagnostics checks to identify issues related to discovery, monitoring, and other features.

Format

```
emcli run_fa_diagnostics
      -input_file=fa_domain_discovery_file:file_path
      [-input_file=host_agent_mapping_file:file_path]
      [-input_file=pf_domain_cred_mapping_file:file_path]
      [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Fully-qualified path to a CSV-formatted file containing one line of details for each Fusion Instance to be added. The valid Fusion Instance WebLogic Server version is 10.

  The structure of the CSV file for WebLogic Server version 10.x and above is as follows:

  ```
  <WebLogic Server version>,
  <Administration Server host machine name>,
  <Administration Server listen port>,
  <Administration Server username>,
  <Administration Server password>,
  <External Parameters - optional>,
  <JMX Protocol - required only if SSL is enabled>,
  <JMX Service URL - required only if SSL is enabled>,
  <Unique Domain Identifier>,
  <Agent URL>,
  <Discover Down Servers - optional - Default if not specified is false>,
  <Use Same Credentials for All Domains in the Fusion Instance - optional -
  Default if not specified is true>,
  <Discover Application Versions - optional - Default if not specified is true>
  ```

  For example:

  ```
  fa1-CRM,weblogic,welcome1,
  fa1-FIN,weblogic1,welcome2,
  fa2-CRM,weblogic,welcome.host.example.com
  ```

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Examples

This example reads the my_domains_info.csv file to determine the Fusion Instances to run diagnostic checks, reads the my_agent_mapping.csv file to determine which Management Agents to use for running discovery tests, and reads the my_domain_cred_mapping.csv file to determine which credentials to use to discover the individual product family.

```
emcli run_fa_diagnostics
-input_file=fa_domain_discovery_file:c:\emcli\my_domains_info.csv
```

```
-input_file=host_agent_mapping_file:c:\emcli\my_agent_mapping.csv
-input_file=pf_domain_cred_mapping_file:c:\emcli\my_domain_cred_mapping.csv
```

run_prechecks

Submits the pre-check operation for any given operation plan.

Format

```
emcli run_prechecks
      -operation_plan=<operation_plan_name>
```

Parameters

- **operation_plan**

  Name of the operation plan.

Examples

```
emcli run_prechecks
      -operation_plan="BISystem1-switchover"
```

run_prerequisites

Runs a list of Enterprise Manager repository-related prerequisites.

Format

```
emcli run_prerequisites
      -db_user=<database_user>
      -db_password=<database_password>
      -db_role=<database_role>
      -repos_user=<repository_user>
      [-prerequisite_xml_root_dir=<xml_root_directory_for_platform_prerequisites>]
      [-prerequisite_resource_locs="<xml_resource_location_for_platform/
      plug-in_prerequisites>"]
      [-log_loc=<location_for_log_files_of_EMPrereqKit_tool>]
      [-upgrade_version=<EM_version_to_which_upgrade_is_being_done_eg_12.1.0.3>]
      [-configuration_type=<configuration/deployment_type_
       eg_MINI/SMALL/MEDIUM/LARGE>]

[ ] indicates that the parameter is optional.
```

Parameters

- db_user

  Database user account with which a connection to the database can be established, for example SYS.

- db_password

  Database user account password. If you do not provide here, you will be prompted for the password.

- db_role

  Database role. For example, sysdba. Required only when the -db_user value is SYS.

- repos_user

  Repository user account with which the prerequisite checks can be run, for example, SYSMAN. Required only when the -db_user value is SYS.

- prerequisite_xml_root_dir

  Absolute path to the `requisites/list` directory where of the all prerequisite XMLs are located. This is an optional parameter and if not provided, the value is calculated internally. The XML files can be in a subdirectory within the `requisites/list` directory, but make sure the path that you enter leads only up to the `list` directory. For example, `$<OMS_HOME>/install/requisites/list`.

- prerequisite_resource_locs

  Absolute path to the directory where the plug-in opar files or the platform/plug-in binaries, which contains XML files for platform or plug-in prerequisite checks, are located. This is an optional parameter. For plug-in opar files, use the format `plugin_id=<<absolute_path_.opar_file>>`. For the plug-in home directory use the format `plugin_id=<<plugin_home>>`.

- log_loc

  Absolute path to a directory where the logs of the execution of the Enterprise Manager prerequisite kit can be stored.

- upgrade_version

  The Enterprise Manager version to which the upgrade is being done. For example, 12.1.0.3. If you have downloaded the Enterprise Manager prerequisite resources for two future versions, for example v1 and v2 through Self-Update then with -upgrade_version, you can see or run the prerequisite of the specified version.

- configuration_type

  Configuration or deployment type. For example, MINI, SMALL, MEDIUM, LARGE. This is an optional parameter, and if not provided, it will be calculated internally.

Examples

Example 1

Runs a list of Enterprise Manager repository-related prerequisites with the configuration type MEDIUM.

```
emcli list_prerequisites
      -db_user=SYS
      -db_password=pwd
      -db_role=sysdba
      -repos_user=SYSMAN
      -prerequisite_xml_root_dir=$ORACLE_HOME/install/requisites/list
      -configuration_type=MEDIUM
```

Example 2

Runs a list of Enterprise Manager repository-related prerequisites with upgrade version 12.1.0.4.

```
emcli list_prerequisites
      -db_user=SYS
      -db_password=pwd
      -db_role=sysdba
      -repos_user=SYSMAN
      -prerequisite_xml_root_dir=$ORACLE_HOME/install/requisites/list
      -upgrade_version=12.1.0.4.0
```

Example 3

Runs a list of Enterprise Manager repository-related prerequisites with the prerequisite resource location `oracle.sysman.db=<<MW_HOME>>/plugins/oracle.sysman.db.oms.plugin_x.x.x.x.x,oracle.sysman.emas=<<Absolute directory path>>/x.x.x.x.x_oracle.sysman.emas_2000_0.opar'`.

```
emcli list_prerequisites
     -db_user=SYS
     -db_password=pwd
     -db_role=sysdba
     -repos_user=SYSMAN
     -prerequisite_resource_locs="oracle.sysman.db=
     <<MW_HOME>>/plugins/oracle.sysman.db.oms.plugin_x.x.x.x.x,
     oracle.sysman.emas=<<Absolute directory path>>/
     x.x.x.x.x_oracle.sysman.emas_2000_0.opar"
```

run_promoted_metric_diag

Runs promoted metric diagnostics.

Format

```
emcli run_promoted_metric_diag
        -name=<target_name>
        -type=<target_type>
        -promotedMetricName=<metric_name>
        -promotedColumn=<metric_type>
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **promotedMetricName**

  Promoted metric name.

- **promotedColumn**

  Promoted metric type.

Examples

```
emcli run_promoted_metric_diag  -name='MyTarget' -type='generic_service'
-promotedMetricName='metric1' -promotedColumn='Performance'
```

save_masking_script

Saves a masking script already generated to the specified path or file.

Format

```
emcli save_masking_script
     -definition_name=<masking_definition_name>
     [-path=file path]
     [-file=file name]

[ ] indicates that the parameter is optional
```

Parameters

- **definition_name**

  Masking definition name.

- **path**

  Path for the file name to save the masking script. File name is automatically generated. The path and file options are mutually exclusive. Only an absolute path is allowed.

- **file**

  File name to save the masking script. The file name must include the absolute path. Either the path or file option must be specified.

Output

Success or error messages

Examples

Example 1

This example saves the masking script for the definition named mask_hr_data to the /tmp directory:

```
emcli save_masking_script
       -definition_name=mask_hr_data
       -path=/tmp/
```

Example 2

This example saves the masking script for the definition named mask_hr_data to /tmp/abc.sql :

```
emcli save_masking_script
       -definition_name=mask_hr_data
       -file=/tmp/abc.sql
```

save_metric_extension_draft

Save a deployable draft of a metric extension. The metric extension must currently be in an editable state. Once saved as a draft, the metric extension is no longer editable.

Format

```
emcli save_metric_extension_draft
       -target_type=<metric_extension_target_type>
       -name=<metric_extension_name>
       -version=<metric_extension_version>
```

Parameters

- **target_type**

  Target type of the metric extension.

- **name**

  Name of the metric extension.

- **version**

  Version of the metric extension to be saved to the draft.

save_procedure_input

Configures a deployment procedure for execution.

Format

```
emcli save_procedure_input
        [-name="procedure_configuration_name"]
        [-owner="procedure_configuration_owner"]
        [-procedure="procedure_guid"]
        -input_file="file_path\file_name"
        [-grants="access_levels_for_users"]
        [-schedule=
            start_time:yyyy/MM/dd HH:mm;
            tz:{java timezone ID};
            grace_period:xxx;
        ]
        [-notification="procedure status"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the configuration for the procedure.

- **owner**

  Owner of the Procedure configuration.

- **procedure**

  GUID of the procedure to execute.

- **input_file**

  GUID of the procedure to execute. The file_path should point to a file containing the data property file.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **grants**

  Specifies users and their corresponding access levels as a string of user:privilege pairs, each separated by a semi-colon (;). The user is an Enterprise Manager user name, and the privilege is either VIEW_JOB or FULL_JOB.

  See the example below.

- **schedule**

  Schedule for the deployment procedure. If not specified, the procedure is executed immediately.

  - **start_time** — When the procedure should start.
  - **tz** — Optional timezone ID.
  - **grace_period** — Optional grace period in minutes.

- **notification**

  Status of the procedure.

Example

```
emcli save_procedure_input
    -name=configProcedure -procedure=16B15CB29C3F9E6CE040578C96093F61
    -input_file=/home/data.properties -grants="user1:VIEW_JOB;user2:FULL_JOB"
    -schedule="start_time:2011/8/21 21:23;tz:America/New_York;grace_period:60"
    -notification="scheduled, action required, running"
```

schedule_siteguard_health_checks

Schedules health checks for an operation plan. Optionally, configured users can be notified about scheduled health-check reports.

Format

```
emcli schedule_siteguard_health_checks
   -operation_plan=[name_of_the_operation_plan]
   -schedule=
              start_time:yyyy| MM | dd HH:mm;
              [tz:"java timezone ID";]
              [frequency:interval | weekly | monthly| yearly;]
              [repeat:tx;]
              [end_time:yyyy | MM | dd HH:mm;]
              [grace_period:xxx;]
   [-notify="true" | "false"]
   [-email="email_address_to_be_notified"]

[ ] indicates that the parameter is optional
```

Parameters

- **operation_plan**

  Name of the operation plan for which health checks must be scheduled.

- **schedule**

  Time when health checks need to run. The possible values for this parameter are:

  – **start_time**

    Date and time when health checks need to be executed.

  – **tz**

    Time zone ID to run health checks.

  – **frequency**

    Frequency at which you want to execute health checks. The valid values for this parameter are `once`, `interval`, `weekly`, `monthly`, and `yearly`. If the frequency is set to `interval`, then the values for the parameter `repeat` must be specified. If the frequency is set to `weekly` or `monthly`, then the days when the health check needs to be executed must be specified. If frequency is set to `yearly`, then both days and months when the health checks need to be executed must be specified.

  – repeat

    Frequency of repetition of the health checks. You need to enter the values for this parameter only if the frequency is set to 'interval'. You need to specify one of the following values for this option:

    * days

      Enter the list of days that the health checks need to be executed for the specified operation plan. Use commas to separate the items in the list. This value is required only if the frequency is set to `weekly`, `monthly`, or `yearly`. If frequency is set to `weekly`, then the valid range is `1` to `7`. If the frequency is set to `monthly` or `yearly`, then the valid range is `1` to `30`.

* months

  Enter the list of months that the health checks need to be executed for the specified operation plan. Use commas to separate the items in the list. This value is required only if the frequency is set to `monthly`. If the frequency is set to `monthly`, then the valid range is `1` to `12`.

– end_time

  Time when the health check should end. This parameter is optional. If the values for this parameter are not specified, the health checks run indefinitely.

– grace_period

  Values of the grace period for the health check scheduled for the specified operation plan. Enter the values in minutes.

- **notify**

  Emails the health check reports to the configured users. If this parameter is set to `true`, then the configured users receive an email notification of the health-check execution report.

- **email**

  Email address of the configured users who should be notified about the health-check reports. The email addresses specified need to be those of registered users.

Examples

Example 1

This example schedules a health check for the `austin-switchover` operation plan to start on `2014/06/10` at `3:45 p.m.`:

```
emcli schedule_siteguard_health_checks
        -operation_plan="austin-switchover"
        -schedule="start_time:2014/06/10 15:45"
```

Example 2

This example schedules a health check for the `austin-switchover` operation plan to start on `2014/10/29` at `2:00 a.m.` and to run daily. The example also notifies the configured user by sending an email to `admin@example.com`:

```
emcli schedule_siteguard_health_checks
        -operation_plan="austin-switchover"
        -schedule="start_time:2014/10/29 2:00;frequency:interval;repeat:1d"
        -notify="true"
        -email="admin@example.com"
```

Example 3

This example schedules a health check for the `austin-failover` operation plan to start on `2014/08/10` at `1:00 a.m.` and to run weekly:

```
emcli schedule_siteguard_health_checks
        -operation_plan="austin-failover"
        -schedule="start_time:2014/08/10 01:00;frequency:interval;repeat:1w"
```

Example 4

This example schedules a health check for the `austin-failover` operation plan to start on `2014/08/10` at `1:00 a.m.,New York` timezone. The example also schedules the health check to run on Saturday and Sunday of every week, with a grace period of 60 minutes:

```
emcli schedule_siteguard_health_checks
        -operation_plan="austin-failover"
        -schedule="start_time:2014/08/10 01:00;frequency:weekly;days:6,7;grace_
period:60;tz:America/New_York"
```

search_patches

Searches patches from the ARU site or software library with the specified search criteria.

Format

```
emcli search_patches
        [-swlib]
        [-patch_name="patch_name"]
        [-product="product_id" [-include_all_products_in_family]]
        [-release="release_id"]
        [-platform="platform_id" | -language="language_id"]
        [-type="patch | patchset"]
        [-noheader]
        [-script | -xml | -format=
                                [name:<pretty|script|csv>];
                                [column_separator:"column_sep_string"];
                                [row_separator:"row_sep_string"];
        ]

[ ] indicates that the parameter is optional
```

Parameters

- **swlib**

  Searches patches in the software library if this parameter is provided, whether the current connection mode is online or offline.

- **patch_name**

  Patch name, number, or Sun CR ID. This option is only valid in Simple Search mode. If you provide this option, the Simple Search mode is enabled. If the options specific to Advanced Search mode are provided along with this option, they will not take effect.

- **product**

  Patch product/product family ID. Run the command "emcli list_aru_products" to search the product ID.

- **include_all_products_in_family**

  Takes the specified product ID as a product family ID and includes all products in this product family while searching patches. This option is valid only when you provide the 'product' option.

- **release**

  Patch release ID. Run the command "emcli list_aru_releases" to search for the release ID.

- **platform**

  Patch platform ID. Run the command "emcli list_aru_platforms" to search for the platform ID.

- **language**

  Patch language ID. Run the command "emcli list_aru_languages" to search for the language ID.

- **type**

  Patch type.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **xml**

  Displays the patch information in XML format.

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli search_patches -patch_name=6880880  -platform=226 -swlib

emcli search_patches -patch_name=6880880  -platform=226 -language=0 -xml

emcli search_patches -product=9480  -release=80102030  -platform=226 -type=patch
-format=name:pretty

emcli search_patches -product=9480  -release=80102030 type=patch -xml

emcli search_patches -product=9480  -release=80102030 -script

emcli search_patches -product=9480 -release=80102030 type=patchset
-format=name:csv
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans

set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

secure_agent

Secures an Agent.

Format

```
emcli secure_agent
        -agent_name="agent_target_name"
        -registration_pwd="registration_password"
        [-host_username ="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name ="credential_name"]
        [-credential_setname ="credential_setname_of_agent"]

[ ] incicates that the paramter is optional
```

Parameters

- **agent_name**

  Name of the Agent target.

- **regustration_pwd**

  Registration password to secure the communication with OMS.

- **host_username**

  User name of the OS user (on the host) who owns the Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Agent. Example: "HostCreds".

Examples

Example 1

```
emcli secure_agent -agent_name="agent.example.com:1234"
                   -registration_pwd="test_pwd"
                   -host_username="test_user"
                   -host_pwd="test"
```

Example 2

```
emcli secure_agent -agent_name="agent.example.com:1234"
                   -registration_pwd="test_pwd"
                   -credential_name="MyMachineCredential"
```

Example 3

```
emcli secure_agent -agent_name="agent.example.com:1234"
                   -registration_pwd="test_pwd"
                   -credential_setname="HostCreds"
```

secure_agents

Secures Agents by providing a list of Agent names, a group name, and input file. If a group name is provided, Enterprise Manager resolves this to a list of Agents that monitor targets in this group. You can also provide an Agent list with an input file to this EM CLI command. For all of these options, you must provide either a user name or password, or the user must have been configured with preferred credentials on Agent targets. This verb submits a job with the list of Agents and the credentials provided as input, and outputs the Job Name and Job ID that you can use to track the status of the job.

This verb also calculates the list of Agents to resecure by filtering out invalid Agents, Agents that are not secure, Agents that are down, and Agents that already have an active job execution. This verb also filters out Agents that are already secured by the correct CA, but you can disable this particular filter by using the -disable_ca_check option .

Format

```
emcli secure_agents
        [-agt_names="agt1;agt2;..."] [-agt_names_file="<file>"]
        [-group_name="group_name"]
        [-use_pref_creds]
        [-username="username"]
        [-password="password"]
        [-disable_ca_check]

[ ] incicates that the paramter is optional
```

Parameters

- **agt_names**

  Semicolon-separated list of Agent names.

- **agt_names_file**

  Absolute path of file containing list of Agent names, each on a new line.

- **group_name**

  Identifies the list of Agents to secure. Enterprise Manager resolves the list of Agents that monitor (not just members of the group) the list of targets in the group.

- **use_pref_creds**

  Uses preferred credentials configured for the Agent to execute the secureAgent job.

- **username**

  User name to execute the secureAgent job at the Agent.

- **password**

  User password to execute the secureAgent job at the Agent.

- **disable_ca_check**

  Disables the check to verify if the Agents are secured with the latest CA.

Examples

```
emcli secure_agents -agt_names="agent_host1:1831;agent_host2:3872" -use_pref_creds

emcli secure_agents -agt_names="agent_host1:1831;agent_host2:3872"
-username=oracleagt

emcli secure_agents -agt_names_file=/tmp/agents_list.txt -use_pref_creds

emcli secure_agents -agt_names_file=/tmp/agents_list.txt -username=oracleagt
```

set_agent_property

Modifies a specific Management Agent property. You can use this command if you have operator privilege for the Management Agent.

Format

```
emcli set_agent_property
      -agent_name="<agent_target_name>"
      -name="<agent_property_name>"
      -value="<agent_property_value>"
      [-new]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **name**

  Name of the Management Agent property you want to modify.

- **value**

  New value for the Management Agent property.

- **new**

  Denotes whether this is a new Agent property being added.

Examples

Example 1

This example sets the value of the UploadInterval property in emd.properties to 15.

```
emcli set_agent_property -agent_name="agent.example.com:1234"
      -name=UploadInterval
      -value=15
```

Example 2

This example sets the value of new property 'newprop' in emd.properties to 15.

```
emcli set_agent_property -agent_name="agent.example.com:1234"
      -name=newprop
      -value=15
      -new
```

set_availability

Changes the availability definition of a given service.

Format

```
emcli set_availability
      -name=<target_name>
      -type=<target_type>
      -availType=TESTS|SYSTEM|SUB_SERVICE
      -availOp=and|or
      [-sysAvailType=SYSTEM_TARGET_DIRECTLY|SELECTED_COMPONENTS_OF_A_SYSTEM]
      [-keycomponents=<'keycomp1name:keycomp1type;
        keycomp2name:keycomp2type;...'>]
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type. Aggregate services target type are also supported. Use the get_targets verb to get the target type of a target.

- **availType**

  Type of availability. Switches the availability to either test-based, system-based, or subservice-based. SUB_SERVICE is supported only for aggregate services.

- **availOp**

  If `and`, it uses all key tests/components to decide availability.

  If `or`, it uses any key tests/components to decide availability.

- **sysAvailType**

  Type of availability when the availType is system-based. Sets the availability to either SYSTEM_TARGET_DIRECTLY or SELECTED_COMPONENTS_OF_A_SYSTEM .

  – If availability is set to 'system target directly', the system associated with the service needs to define availability[status], systemname, and systemtype are required arguments.

  – If availability is set to 'selected components of a system', systemname, systemtype, and keycomponents are required arguments.

  – If availability is set to 'system target directly', and if availability[status] is not defined, the availability set is invalid. Therefore, the only option that can be set is 'selected components of a system'.

- **keycomponents**

  Name-type pair (that is, keycomp_name:keycomp_type) list of key components in the system used for the service.

Examples

Example 1

This example sets the availability of service MyTarget to be based on all key-tests.

```
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='test' -availOp='and'
```

Example 2

This example sets the availability of service MyTarget to be based on any key-test.

```
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='test' -availOp='or'
```

Example 3

This example sets the availability of service MyTarget to be based on any key components of a system.

```
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='system' -availOp='or'
                       -keycomponents='database:oracle_database; host1:host'
```

Example 4

This example sets the availability of service MyTarget to be based on system targets availability.

```
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='system' -availOp='and'
                       -sysAvailType='system target directly'
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='system' -availOp='and'
                       -sysAvailType='selected components of a system'
                       -keycomponents='database:oracle_database; host1:host'
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='system' -availOp='or'
                       -sysAvailType='selected components of a system'
                       -keycomponents='database:oracle_database; host1:host'
```

set_config_history_retention_period

Sets the amount of time for which the configuration history is retained.

Format

```
emcli set_config_history_retention_period
      -period="Retention period in months"
```

Parameters

- **period**

  Retention period in months. The value must be in the range of 1 to 60 inclusive.

Example

This example sets the retention period to 12 months.

```
emcli set_config_history_retention_period
-period=12
```

set_connection_mode

Sets the new MOS connection mode.

Format

```
emcli set_connection_mode
        -mode="online | offline"
```

Examples

```
emcli set_connection_mode -mode="offline"

emcli set_connection_mode -mode="online"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

set_credential

Sets preferred credentials for given users.

> **Note:** This command does not support the COLLECTION credential sets.

Format

```
emcli set_credential
      -target_type="ttype"
      [-target_name="tname"]
      -credential_set="cred_set"
      [-user="user"]
      -columns="col1:newval1;col2:newval2;PDP:SUDO/POWERBROKER;RUNAS:oracle;
         PROFILE:user1..."
      [-input_file="tag1:file_path1;tag2:file_path2;..."]
      [-oracle_homes="home1;home2"]
      [-monitoring]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target. This must be "host" if the `-oracle_homes` parameter is specified.

- **target_name**

  Name of the target. Omit this argument to set enterprise preferred credentials. This must be the host name if the `-oracle_homes` parameter is specified.

- **credential_set**

  Credential set affected.

- **user**

  Enterprise Manager user whose credentials are affected. If omitted, the current user's credentials are affected.

- **columns**

  Name and new value of the column(s) to set. Every column of the credential set must be specified. Alternatively, a tag from the `-input_file` argument can be used so that the credential values are not seen on the command line. You can specify this argument more than once.

- **input_file**

  Path of the file that has the `-columns` argument(s). This is used to hide passwords. Each path must be accompanied by a tag referenced in the `-columns` parameter. You can specify this option more than once.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **oracle_homes**

  Name of Oracle homes on the target host. Credentials will be added/updated for all specified homes.

Note: The list of columns and the credential sets they belong to is included in the metadata file for each target type. This and other credential information is in the `<CredentialInfo>` section of the metadata.

- **monitoring**

  Flag indicating that credentials affected are monitoring credentials. If omitted, the credentials affected are preferred credentials. Monitoring credentials require specifying the target_name option.

Examples

Example 1

```
emcli set_credential
      -target_type=oracle_database
      -target_name=myDB
      -credential_set=DBCredsNormal
      -user=admin1
      -column="username:joe;password:newPass;role:newRole"
```

Example 2

In this example, `FILE1` is a tag to refer to the contents of `passwordFile`. Note that Example 2 has the same effect as Example 1.

```
emcli set_credential
      -target_type=oracle_database
      -target_name=myDB
      -credential_set=DBCredsNormal
      -user=admin1
      -column=FILE1
      -input_file=FILE1:passwordFile
```

Example 3

In this example, the contents of the `passwordFile`: is `username:joe;password:newPass;role:newRole`

```
emcli set_credential
      -target_type=host
      -target_name=host.example.com
      -credential_set=OHCreds
      -user=admin1
      -column="OHUsername:joe;OHPassword:newPass"
      -oracle_homes="database1;mydb"
```

set_db_service_properties

Sets and updates the Database as a Service (DBaaS) target properties by providing the database unique name of an underlying database target and property name/value for the service target.

Format

```
emcli set_db_service_properties
      -db_unique_name="database unique name"
      -property_name="property name"
      -property_value="property value"
```

Parameters

- db_unique_name

  The database name of the database target on which the DBaaS target relies. You can find this name on the Last Collected page of the database target, or you can query for it.

- property_name

  The target property name of a DBaaS target (for example, `company_gtp_cost_center`, `company_gtp_line_of_bus`, `company_gtp_contact`).

- property_value

  Value you want to assign to the database target's property name of the DBaaS target.

Example

The following example shows how to set a property value of `web_group1` for the `dev_cost_center` property name of the `company_e_commerce` database:

```
emcli set_db_service_properties -db_unique_name="company_e_commerce" -property_
name="dev_cost_center" -property_value="web_group1"
```

set_default_pref_cred

Sets a named credential as a default preferred credential. If you decide to use preferred credentials for an Enterprise Manager operation and preferred credentials are not set for the target, the default credentials for this target type that you set are used. Default credentials are set at the target-type level.

Format

```
emcli set_default_pref_cred
        -set_name="set_name"
        -target_type="ttype"
        -credential_name="cred_name"
        [-credential_owner ="owner]"
        [-test]
        [-test_target_name="test_target_name"]

[ ] indicates that the parameter is optional
```

Parameters

- **set_name**

  Sets the preferred credential for this credential set.

- **target_type**

  Target type for the credential set.

- **credential_name**

  Name of the credential.

- **credential_owner**

  Owner of the credential. This defaults to the currently logged-in user.

- **test**

  Tests the credential before setting it as the default credential.

- **test_target_name**

  Tests the target name if the global credential is set as the default preferred credential.

Examples

Example 1

This example sets the named credential MyHostCredentials as the default preferred credential for the target type host as HostCredsNormal.

```
emcli set_default_pref_credential
        -set_name=HostCredsNormal
        -target_type=host
        -credential_name=MyHostCredentials
        -credential_owner="Joe"
```

Example 2

This example sets the named credential MyHostCredentials as the default preferred credential for the target type host as HostCredsNormal. The command tests the named

credential MyHostCredentials against server1.example.com before setting it as a default preferred credential.

```
emcli set_default_pref_cred
        -set_name=HostCredsNormal
        -target_type=host
        -credential_name=MyHostCredential
        -credential_owner="Joe"
        -test
        -test_target_name=server1.example.com
```

set_default_privilege_delegation_setting

Sets the default privilege delegation settings for one or more platforms.

Format

Standard Mode

```
emcli set_default_privilege_delegation_setting
        -default_setting_list="platform1:setting_name1;platform2:setting_name2"
        [-separator="separator:attribute_name:character"]
        [-subseparator="subseparator:attribute_name:character"]
```

Interactive or Script Mode

```
set_default_privilege_delegation_setting(
        default_setting_list="platform1:setting_name1;platform2:setting_name2"
        [,separator="separator:attribute_name:character"]
        [,subseparator="subseparator:attribute_name:character"]
        )

[ ] indicates that the parameter is optional
```

Exit Codes

0 on success. A non-zero value means verb processing was not successful.

Parameters

- **default_setting_list**

  List of default settings per platform. Supported platforms: Linux, HP-UX, AIX, SunOS.

- **separator**

  By default, multi-value input attributes use the semicolon (;) character as a separator. Specifying this option overrides the default separator value.

  Example: `separator="<attribute_name=sep_char>"` where attribute_name is the name of the attribute for which you want to override the separator character, and sep_char is new separator character.

  Example: `separator="att=#"` changes the separator character to a pound sign (#).

- **subseparator**

  By default, multi-value input attributes use the colon (:) character as the sub-separator. Specifying this option overrides the default sub-separator value.

  Example: `subseparator="<attribute_name=sep_char"` where `attribute_name` is the name of the attribute for which you want to override the separator character, and `sep_char` is the new sub-separator character.

  Example: subseparator="att=#" changes the sub-separator character to a pound sign.

Examples

Example 1

This example sets the privilege delegation setting to SUDO1 for Linux platforms and SUDO2 for HP-UX platforms.

```
emcli set_default_privilege_delegation_setting
          -default_setting_list="Linux:SUDO1;HP-UX:SUDO2"
```

Example 2

This example sets the privilege delegation setting to SUDO_SETTING_1 for Linux and SUDO_SETTING_2 for HP-UX. The default separator has been changed to a comma (,) and the subseparator to a hash tag (#).

```
emcli set_default_privilege_delegation_setting
          -default_setting_list="Linux#SUDO_SETTING_1,HP-UX#SUDO_SETTING_2"
          -separator="default_setting_list=,"
          -subseparator="default_setting_list=#"
```

set_key_beacons_tests

Defines key beacons and tests of the service.

Format

```
emcli set_key_beacons_tests
      -name=<target_name>
      -type=<target_type>
      [-beacons=<beacon_names>]+
      [-tests='test1:type1;test2:type2;...']+
      [-removeKey]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **beacons**

  Names of beacons to set as key (or non-key).

- **tests**

  Names and types of tests to set as key (or non-key).

- **removeKey**

  If specified, the mode is (remove key); that is, the specified tests and beacons will be set as non-key.

  If not specified, the mode is (add key); that is, the specified tests and beacons will be set as key.

Examples

Example 1

This example sets `MyTest/HTTP`, `MyTest2/FTP` and `MyBeacon` as non-key elements of service `MyTarget/generic_service`.

```
emcli set_key_beacons_tests -name='MyTarget' -type='generic_service'
      -tests='MyTest:HTTP;MyTest2:FTP'
      -beacons='MyBeacon' -removeKey
```

Example 2

This example sets `MyBeacon` and `MyBeacon2` as key beacons of service `MyTarget/generic_service`.

```
emcli set_key_beacons_tests -name='MyTarget' -type='generic_service'
      -beacons='MyBeacon;MyBeacon2'
```

set_logging_property

Sets the property value corresponding to the specified logging property name.

Format

```
emcli set_logging_property
        -property_name="propertyName"
        [-oms_name="omsName"]
        -property_value="propertyValue"

[ ] indicates that the parameter is optional
```

Parameters

- **property_name**

  Name of the logging property whose value needs to be set.

- **oms_name**

  Name of the management server where the logging property needs to be set.

- **property_value**

  Value to be set.

Examples

Example 1

This example sets the value for the property name "propName" on the management server myhost:1159_Management_Service to "propValue."

```
set_logging_property -property_name=propName -property_value=propValue
-oms_name="myhost:1159_Management_Service"
```

Example 2

This example sets the value for the property name "propName" to "propValue" on all of the management servers.

```
set_logging_property -property_name=propName -property_value=propValue
```

set_metric_promotion

Creates or edits a metric promotion based on a test or system.

Format

```
emcli set_metric_promotion
      -name=<service_target_name
      -type=<service_target_type
      ***[-category=Usage/Performance/Business]
      -basedOn=SYSTEM|TESTS|SUB_SERVICE
      -aggFunction=AVG|MAX|MIN|SUM|COPY
      [-promotedMetricName=<promoted_metric>]
      [-promotedMetricColumn=<promoted_metric_column>]
      -promotedMetricKey=<key_value_of_promoted_metric>
      [-metricName=<dependent_metric_name>]
      -column=<dependent_metric_column>
      *[-depTargetType=<target_type_of_dependent_targets>]
      *#[-depTargets='target1;target2...']
      *#[-depTargetKeyValues='target1:key11|key12|key13..;
          target2:key21|key22|key23..']
      *[-depMetricKeyValue=<dependent_metric_key_column>]
      **[-testname=<dependent_test_name]
      **[-testtype=<dependent_test_type]
      **[-metricLevel=TXN|STEP|STEPGROUP]
      **[-beacons='bcn1;bcn2..']
      **[-depTestComponent=<step_or_stepgroup_name>]
      [-threshold='critical_threshold_value;warning_threshold_value;
         threshold_operator (EQ|LE|LT|GT|GE)']
      -mode=CREATE|EDIT
      #[-includeRuleBasedTargets = YES|NO]
      [-targetFilter = ALL|STARTS_WITH:<filter>|ENDS_WITH:<filter>|
        CONTAINS:<filter>|EQUALS:<filter>]
[ ] indicates that the parameter is optional.
```

Key:

* — Might be required if `basedOn` is set to SYSTEM
** — Might be required if `basedOn` is set to TESTS
*** — Might be required if `basedOn` is set to `SUB_SERVICE`
\# — One of these values is required for system-based metrics.

Parameters

- **category**

  Defines whether the promoted metric is a usage, performance, or business metric of a service. Category is used to determine the promoted metric name and metric column. If you do not specify this option, you must specify the `promotedMetricName` and `promotedMetricColumn` options.

- **basedOn**

  Determines whether the promotion is test-based or system-based.

- **aggFunction**

  Determines the aggregate function to be used to compute the promoted metric. `AVG/MAX/MIN/SUM` takes average, max, min, and sum of the dependent metrics,

respectively. `COPY` only copies over a single dependent metric to the promoted metric.

- **promotedMetricName**

  Promoted metric name. This is optional if the category is specified.

- **promotedMetricColumn**

  Promoted metric column. This is optional if the category is specified.

- **promotedMetricKey**

  Required argument that determines the key value of the promoted metric. It is equivalent to the displayed name of the promoted metric in the UI.

- **metricName**

  Required argument if the dependent metric column is collected by more than one metric.

- **column**

  Dependent metric column.

- **depTargetType**

  All dependent targets should be of this target type.

- **depTargets**

  Specifies the dependent targets. This argument is ignored if you specify `depTargetKeyValues`.

- **depTargetKeyValues**

  Specifies the key values associated with the dependent targets. Specify multiple key values for a single target by repeating the entry in the following format: `'tgt1:key1;tgt1:key2...'`

- **depMetricKeyValue**

  Required if the dependent metric is a transpose metric. It is the key value that applies to all the dependent targets.

- **testname**

  Defines the name of the test to be used in promoting the metric.

- **testtype**

  Defines the type of test to be used in promoting the metric.

- **metricLevel**

  Some metrics can be promoted on step-level. This option defines the level to be used during promotion.

- **beacons**

  List of beacons to be used for promoting the metric data.

- **depTestComponent**

  If `metricLevel` is not `TXN`, this option is required to specify which step or which step group is being promoted.

- **threshold**

Defines a threshold on the promoted metric.-mode: The mode can be CREATE or EDIT.

- **includeRuleBasedTargets**

  If YES, the system member targets available at the time of metric evaluation are considered for metric evaluation. The default is NO. This option is applicable only for system-based metrics.

- **targetFilter**

  The given target filter value is compared with target names of system member targets. The member targets that meet this filter value will participate in the metric evaluation. For a target filter, wild cards such as *, % an so forth are not accepted. e.g., ALL, STARTS_WITH:EM, ENDS_WITH:EM, CONTAINS:EM and EQUALS:EM.

Examples

Example 1

This example creates a promoted Performance metric with key value mymetric1 on service MyTarget using MyTest/HTTP. The promoted metric takes the maximum of the dns_time metric column returned by the MyBeacon and mybcn1 beacons. It also has a threshold with 'greater or equal to' operator (GE) with the critical value set to 200 and warning value set to 100.

```
emcli set_metric_promotion -name='MyTarget' -type='generic_service'
      -category=Performance -basedOn=test -aggFunction=MAX
      -testname='MyTest' -testtype=HTTP
      -beacons='MyBeacon, mybcn1'
      -promotedMetricKey=mymetric1 -column=dns_time -metricName=http_response
      -metricLevel=TXN -threshold='200;100;GE' -mode=CREATE
```

Example 2

This example creates a promoted Usage metric with key value mymetric1 on service MyTarget. The dependent target is 'myhost.mydomain.com' with type host. The promoted metric just copies the cpuUtil column of the Load metric.

```
emcli set_metric_promotion -name='MyTarget' -type='generic_service'
      -category=Usage -basedOn=system -aggFunction=COPY
      -promotedMetricKey=mymetric1 -column=cpuUtil -metricName=Load
      -depTargets='myhost.mydomain.com' -depTargetType=host
      -mode=CREATE
```

Example 3

This example creates a promoted Usage metric with the key value AppServerComponentUsage on service MyTarget. The dependent target is 'myapp_server' with type 'oracle_ias'. The promoted metric computes the average value of the cpu.component metric column for the specified key values.

```
emcli set_metric_promotion -name='MyTarget' -type='generic_service'
      -category=Usage -basedOn=system -aggFunction=AVG
      -promotedMetricKey=AppServerComponentUsage -depTargetType=oracle_ias
      -column=cpu.component
      -metricName=opmn_process_info
      -depTargetKeyValues='myapp_server:petstore;myapp_server:http_server'
      -mode=CREATE
```

Example 4

This example creates a promoted business metric with key value ordersCount on service MyTarget. The dependent targets are 'onlineOrderService1' and 'onlineOrderService2' with type 'generic_service'. The promoted metric computes the total number of orders from all dependent targets. Note that if the category is business, the category of the metrics from dependent targets should also be business. In the example below, the metric category of the metric named 'Business' is 'Business'.

```
emcli set_metric_promotion -name='MyTarget' -type='generic_service'
          -category=Business -basedOn=system -aggFunction=SUM
          -promotedMetricKey='ordersCount'
          -depTargets='onlineOrderService1;onlineOrderService2'
          -depTargetType='generic_service'
          -metricName='Business' -column='BusinessValue'
          -depMetricKeyValue='Number of Orders Placed'
          -mode=CREATE
```

Example 5

This example creates a promoted performance metric based on a system with the key value of EMCLIRule1 on service MyTarget. The dependent target is weblogic_ j2eeserver. The underlying metric column is cpuUsage.percentage and the metric name is jvm. No dependent targets are selected during metric promotion. The options includeRuleBasedTargets=YES and targetFilter=ALL consider all dependent targets available at the time of metric execution.

```
emcli set_metric_promotion
      -name="MyTarget" -type="generic_service"
      -category=Performance -promotedMetricKey=EMCLIRule1
      -column='cpuUsage.percentage'
      -metricName='jvm' -depTargetType="weblogic_j2eeserver"
      -mode=CREATE -basedOn=system -aggFunction="AVG"
      -includeRuleBasedTargets="YES"
      -targetFilter="ALL"
```

Example 6

This example is like Example 5, except targetFilter=CONTAINS:EM considers the dependent targets that contain 'EM' in the target name for the metric evaluation.

```
emcli set_metric_promotion
      -name="MyTarget" -type="generic_service"
      -category=Performance -promotedMetricKey=EMCLIRule1
      -column='cpuUsage.percentage'
      -metricName='jvm' -depTargetType="weblogic_j2eeserver"
      -mode=CREATE -basedOn=system -aggFunction="AVG"
      -includeRuleBasedTargets="YES"
      -targetFilter="CONTAINS:EM"
```

Example 7

This example edits an existing promoted performance metric EMCLIRule1 based on a system on the service MyTarget. The dependent target is weblogic_j2eeserver. The underlying metric column is cpuUsage.percentage and the metric name is jvm. The existing rule-based metric is edited to a static metric that considers only provided dependent targets '/EMGC_EMGC_DOMAIN/EMGC_ DOMAIN/EMGC_ADMINSERVER and /EMGC_EMGC_DOMAIN/EMGC_ DOMAIN/
EMGC_OMS1' during metric evaluation.

```
emcli set_metric_promotion
```

```
-name="MyTarget" -type="generic_service"
-category=Performance -promotedMetricKey=EMCLIRule1
-aggFunction="MIN"
-mode="Edit" -basedOn=system
-column='cpuUsage.percentage'
-depTargets="/EMGC_EMGC_DOMAIN/EMGC_DOMAIN/EMGC_ADMINSERVER;
  /EMGC_EMGC_DOMAIN/EMGC_DOMAIN/EMGC_OMS1"
-depTargetType="weblogic_j2eeserver"
```

set_monitoring_credential

Sets a monitoring credential set for a target. You can provide input parameters using command line arguments or the input properties file. It also supports the input_file parameter for passwords and parameter values.

Format

```
emcli set_monitoring_credential
        -target_name=<target_name>
        -target_type=<ttype>
        -set_name=<set_name>
        -cred_type=<credential_type>
        -auth_target_type=<auth_ttype>
        -test
        -input_file=<tag|value>
        -properties_file=<filename>
        -attributes=<p1:v1;p2:v2;...>
```

Parameters

- **target_name**

  Sets the monitoring credential for this target.

- **target_type**

  Target type for the target.

- **set_name**

  Sets the monitoring credential for this credential set name.

- **cred_type**

  Credential type for the credential to set as the monitoring credential.

- **auth_target_type**

  Authenticating target type. Defaults to target_type.

- **test**

  Tests the credential against the target(s) before setting the monitoring credential.

- **input_file**

  Supplies sensitive property values from the file.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **properties_file**

  Passes all parameters from the file. Values provided on the command line take precedence.

- **attributes**

  Specify credential columns as follows:

  ```
  colname:colvalue;colname:colvalue
  ```

You can change the separator value using -separator=attributes=<newvalue>, and you can change the subseperator value using -subseparator=attributes=<newvalue>.

Examples

Example 1

This example sets the monitoring credential set DBCredsMonitoring for the target testdb.example.com:oracle_database with user name foo, password bar, and role normal.

```
emcli set_monitoring_credential
        -target_name=testdb.example.com
        -target_type=oracle_database
        -set_name=DBCredsMonitoring
        -cred_type=DBCreds
        -attributes="DBUserName:foo;DBPassword:bar;DBRole:normal"
```

Example 2

This example reads the password from the mypasswordfile.txt file.

```
emcli set_monitoring_credential
        -target_name=testdb.example.com
        -target_type=oracle_database
        -set_name=DBCredsMonitoring
        -cred_type=DBCreds
        -attributes="DBUserName:foo;DBPassword:tag;DBRole:normal"
        -input_file="tag:mypasswordfile.txt"
```

Example 3

This example prompts for the password from standard input.

```
emcli set_monitoring_credential
        -target_name=testdb.example.com
        -target_type=oracle_database
        -set_name=DBCredsMonitoring
        -cred_type=DBCreds
        -attributes="DBUserName:foo;DBRole:normal;DBPassword:"
```

Example 4

This example specifies prop1.txt as a multi-line Java properties file, in which each line contains a parameter=value format. You can provide the password in the same file or not specify it. If not specified, you are prompted for it.

```
emcli set_monitoring_credential
        -properties_file=prop1.txt
```

Example 5

This example sets the monitoring credential set DBCredsMonitoring for the target testdb.oracle.com:oracle_database with a user name of foo, password of bar, and role of normal. The credential is tested before setting the monitoring credential.

```
emcli set_monitoring_credential
        -target_names="testdb1;testdb2"
        -target_type=oracle_database
        -set_name=DBCredsMonitoring
        -cred_type=DBCreds
        -attributes="DBUserName:foo;DBPassword:bar;DBRole:normal"
```

```
-test
```

set_oms_property

Sets the property value corresponding to the specified property name.

Format

```
emcli set_oms_property
        -property_name="propertyName"
        [-oms_name="omsName"]
        -property_value="propertyValue"

[ ] indicates that the parameter is optional
```

Parameters

- **property_name**

  Name of the property whose value needs to be set.

- **oms_name**

  Name of the management server for which the property needs to be set.

- **property_value**

  Property value to be set.

Examples

Example 1

This example sets the value for the property name "propName" on the management server myhost:1159_Management_Service to "propValue."

```
set_oms_property -property_name=propName -property_value=propValue -oms_
name="myhost:1159_Management_Service"
```

Example 2

This example sets the value for the property name "propName" to "propValue" on all of the management servers.

```
set the value for the property name "propName" to "propValue" on all the
management servers
```

set_patch_plan_data

Sets user-editable data. The get_patch_plan_data verb is useful when used preceding this verb.

Format

```
emcli set_patch_plan_data
        -name="name"
        -input_file=data:"file_path"
        [-impact_other_targets="add_all|add_original_only|cancel"]
        [-problems_assoc_patches="ignore_all_warnings|cancel"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Sets the preferred credential for this credential set.

- **input_file**

  Sets the preferred credential for this target.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **impact_other_targets**

  Target type for the target/credential set.

- **problems_assoc_patches**

  Name of the credential.

Examples

```
emcli set_patch_plan_data -name="plan name"
-input_file=data:"/tmp/patchplan.pros"

emcli set_patch_plan_data -name="plan name"
-input_file=data:"/tmp/patchplan.pros" -impact_other_targets="add_all"

emcli set_patch_plan_data -name="plan name"
-input_file=data:"/tmp/patchplan.pros" -impact_other_targets="add_all"
-problems_assoc_patches="ignore_all_warnings"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches

set_connection_mode
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

set_preferred_credential

Sets a named credential as a target-preferred credential for the user.

Format

```
emcli set_preferred_credential
        -set_name="set_name"
        -target_name="target_name"
        -target_type="ttype"
        -credential_name="cred_name"
        [-credential_owner ="owner]"
        [-test]

[ ] indicates that the parameter is optional
```

Parameters

- **set_name**

  Sets the preferred credential for this credential set.

- **target_name**

  Sets the preferred credential for this target.

- **target_type**

  Target type for the target/credential set.

- **credential_name**

  Name of the credential.

- **credential_owner**

  Owner of the credential. This defaults to the currently logged in user.

- **test**

  Tests the credential against the target_name before setting the preferred credential.

Examples

Example 1

This example sets the named credential MyHostCredentials as the target preferred credential for the target test.example.com:host as HostCredsNormal.

```
emcli set_preferred_credential
        -set_name=HostCredsNormal
        -target_name=test.oracle.com
        -target_type=host
        -credential_name=MyHostCredentials
        -credential_owner="Joe"
```

Example 2

This example sets the named credential MyDBCredentials as the target preferred credential for the target myDB:oracle_database as Normal Database Credentials. The command tests the named credential against myDB:oracle_database before setting the preferred credential.

```
emcli set_preferred_credential
        -target_type=oracle_database
        -target_name=myDB
        -set_name=DBCredsNormal
        -credential_name=MyDBCredentials
        -credential_owner="Joe"
        -test
```

Example 3

This example sets the named credential MyDBCredentials as the target preferred credential for the target myDB:oracle_database as SYSDBA database credentials.

```
emcli set_preferred_credential
        -target_type=oracle_database
        -target_name=myDB
        -set_name=DBCredsSYSDBA
        -credential_name=MyDBCredentials
        -credential_owner="Joe"
```

set_properties

Sets the property for a test or beacons.

Format

```
emcli set_properties
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name
      -testtype=<test_type>
      [-beacons=<beacon_names>]
      [-properties='prop1:value1;prop2:value2;..']+

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **testname**

  Name of the test to set the property on.

- **testtype**

  Type of test to set the property on.

- **beacons**

  Names of the beacons to set the property on.

- **properties**

  Names and values of the properties to be set (can be multiple).

Examples

Example 1

This example sets the property `timeout` to `30000` and granularity to `transaction` for the test `MyTest` defined on `MyTarget` for all beacons.

```
emcli set_properties -name='MyTarget' -type='generic_service'
      -testname='MyTest' -testtype='HTTP'
      -propertyName='timeout:30000;granularity:transaction'
```

Example 2

This example sets the property value to `30000` of the test `MyTest` defined on `MyTarget` for only `MyBeacon` and `MyBeacon2`. This only works if the specified properties can be set on a per beacon level.

```
emcli set_properties -name='MyTarget' -type='generic_service'
      -testname='MyTest' -testtype='HTTP'
      -bcnName='MyBeacon;MyBeacon2'
      -propertyName='timeout' -propertyValue='30000'
```

set_reverse_ping_interval

Modifies the maximum waiting time for the Management Agents. You need to provide Agent names for the modification.

Format

```
emcli set_reverse_ping_interval
        -agent_names="agent1[;agent2...]"|-all_agents
        -value=" "|-reset_to_default

[ ] indicates that the parameter is optional
```

Parameters

- **agent_names**

  Management agents (host:port) on which the modification needs to be performed.

- **all_agents**

  Use only when all Agents need to be modified with the new value.

- **value**

  New value to which the existing waiting time needs to be updated.

- **reset_to_default**

  Use when the value needs to be reset to the default value.

Examples

Example 1

This example modifies the existing waiting time with the new value provided, which in this case is 240.

```
emcli set_reverse_ping_interval -agent_names="myhost1.example.com:1838" -value=240
```

Example 2

This example modifies the existing waiting time for the provided Agents with the default value in the Ping System.

```
emcli set_reverse_ping_interval -agent_
names="myhost1.example.com:1838;myhost2.example.com:4352" -reset_to_default
```

set_standby_agent

Permits targets to relocate from one Management Agent to another. This verb always populates a table that determines which targets from the source Management Agent to the destination Management Agent are permitted to relocate for the Enterprise Manager target.

Format

```
emcli set_standby_agent
       -src_agent=<source_agent>
       -dest_agent=<destination_agent>
       -target_name=<target_name>
       -target_type=<target_type>

[ ] indicates that the parameter is optional
```

Parameters

- **src_agent**

  Management Agent currently monitoring the targets. If srcAgent is not known, enter currentOwner as the argument.

- **dest_agent**

  Management Agent for which you want to monitor the targets.

- **target_name**

  Name of the target to be moved.

- **target_type**

  Type of target to be moved.

Output

Output message of the command execution.

set_target_property_value

Sets the value of a target property for a specified target. Any prior values of the target property are overwritten. When assigning values to the Oracle-provided target properties, use the English names of these target properties:

Comment, Lifecycle Status, Line of Business, Location, Contact

Acceptable values for Lifecyle Status are:

- Development
- MissionCritical
- Production
- Stage
- Test

For cluster target types, the value of the target property automatically propagates to all of its member targets. This happens even without the -propagate_to_members parameter. The propagate_to_members parameter is used for aggregate non-cluster targets where the desired behavior is to propagate the target property values to members of the aggregate target. Note that it will propagate to current members of the aggregate, and not targets that are added in the future.

> **Note:** You can only set up and propagate one property at a time to members.

Format

```
emcli set_target_property_value
      -property_records="target_name:target_type:property_name:property_value"
      [-separator=property_records="sep_string"]
      [-subseparator=property_records="subsep_string"]
      [-input_file="parameter_tag:file_path"]
      [-propagate_to_members]

[ ] indicates that the parameter is optional
```

Parameters

- **property_records**

  List of property records. The following parts comprise each property record:

  <target_name>:<target_type>:<property_name>:property_value>

  - target_name — Target name of the target for which you want to update the property.
  - target_type — Target type of the target.
  - property_name — Name of the property whose value you want to update. Property names are case sensitive. You can execute the list_target_property_ names verb for a list of possible property names.
  - property_value — Value to be assigned/updated for the property.

- **separator**

When specifying multiple property records, use the separator string delimiter as a delimiter between property records. The default separator delimiter is ";".

- **subseparator**

  String delimiter to be used between parts of a property record. The default subseparator delimiter is ":".

- **input_file**

  Used in conjunction with the -property_records option, this option enables you to provide the property records in a file. This option specifies a mapping between a tag and a local file path. The tag is specified in lieu of property records. The tag cannot contain colons (:) or semi-colons (;) .

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **propagate_to_members**

  Used for group and system targets to also propagate the property to all of its members.

Examples

Example 1

This example sets the 'Owner Name' property to Jane Smith for the database test_database.

```
emcli set_target_property_value
      -property_records="test_database:oracle_database:Owner Name:Jane Smith"
```

Example 2

This example sets the Owner property to Jane Smith for the database test_db, and also sets the Asset Number property to 100 for the database test_db1.

```
emcli set_target_property_value
          -property_records="test_db:oracle_database:Owner:Jane Smith;
           test_db1:oracle_database:Asset Number:100"
```

Example 3

This example takes the input of the property records from the specified file /temp/rec_file.

```
emcli set_target_property_value
          -property_records="REC_FILE" -input_file="REC_FILE:/temp/rec_file"
```

The file /temp/rec_file would contain entries such as:

```
test_db:oracle_database:Owner:Jane Smith;test_db1:oracle_database:Asset Number:100
```

Example 4

This example sets the Owner property to Jane Smith for the test_db database, and sets the Asset Number property to 100 for the test_db1 database. The separator used within the records is "," and the subseparator is "@" .

```
emcli set_target_property_value
           -property_records="test_db@oracle_database@Owner@
            Jane Smith,test_db1@oracle_database@AssetNumber@100"
```

Example 5

```
emcli set_target_property_value -property_records="MyProdGroup:composite:LifeCycle
Status:Production" -propagate_to_members
```

set_test_threshold

Sets a test threshold.

Format

```
emcli set_test_threshold
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>
      -metricName=<metric_name>
      -metricColumn=<metric_column>
      -occurrences=<occurrences>
      [-warningThres=<warning_threshold>]
      [-criticalThres=<critical_threshold>]
      [-operator=<operator>]
      [-beaconName=<beacon_name>]
      [-stepName=<step_name>]
      [-stepGroupName=<stepgroup_name>]

[ ] indicates that the parameter is optional
```

Examples

```
emcli set_test_threshold -name="Service Name"
              -type="generic_service"
              -testname="Test Name"
              -testtype="HTTP"
              -metricName="http_response"
              -metricColumn="timing"
              -occurrences=1
              -warningThres=100000
```

setup

Configures EM CLI to work with a specific management server.

You can set up the EM CLI client either in secure mode by specifying the -noautologin option, or unsecure mode by specifying the -autologin otpion. -noautologin is the default, so if you do not specify either option, the EM CLI client is automatically set up in secure mode.

The configuration directory will contain log files generated by EM CLI to record informational and error messages generated during operations.

Format

```
emcli setup
        -url="http[s]://host:port/em"
        -username=<EM_console_username]
        [-password=<password_of_user>]
        -dir=<local_emcli_config_directory>
        [-localdirans=yes|no]
        [-licans=yes|no]
        [-trustall]
        [-certans=yes|no]
        [-nocertvalidate]
        [-novalidate]
        [-autologin]
        [-noautologin]
        [-noregister]
        [-custom_attrib_file=<custom_attr_file_path>]

[ ] indicates that the parameter is optional
```

Parameters

- **url**

  URL of the Oracle Management Server (OMS). `host` specifies the host of the OMS. `port` specifies the listening port of the OMS. Both http and https protocols are supported. (https is recommended for security reasons).

- **username**

  Enterprise Manager user name to be used by all subsequent EM CLI commands when contacting the OMS.

  If the SSO user is also an Enterprise Manager user (that is, authenticated in LDAP/OID), you can only register EM CLI with the ssousername. After you enable SSO for the OMS, you cannot subsequently register EM CLI with only username.

- **password**

  Enterprise Manager user password. If you do not specify this option, you are prompted for the password interactively.

  Note: Providing a password on the command line is insecure and should be avoided.

- **dir**

Directory where an EM CLI configuration directory will be created. This directory must be on a locally mounted file system. A warning and confirmation is issued for an HTTPS URL if the directory is not heuristically identified as such (unless you specify `trustall`). The directory can be relative to the working directory where setup is called, or it can be absolute. This option defaults to the user's home directory.

- **localdirans**

  Indicates whether the setup directory given with the -dir option is a local directory. Specify yes to indicate that the setup directory is local, and specify no to indicate that the setup directory is non-local.

- **licans**

  Indicates whether the license is accepted or not accepted by the user. Specify yes to accept the license, or specify no to not accept the license.

- **trustall**

  Automatically accepts any server certificate from the OMS, which results in lower security.

- **certans**

  Indicates whether the certificate needs to be trusted without having to prompt the user. Specify yes to trust the certificate, and specify no to not trust the certificate.

- **nocertvalidate**

  Does not validate the host name in the SSL certificate provided by the OMS.

- **novalidate**

  Does not authenticate the Enterprise Manager user name or SSO user name against the OMS. Assume the given user name is valid. This enables the configuration to be stored (Enterprise Manager URL and user) without validating or connecting to Enterprise Manager. This might be useful in scenarios where Enterprise Manager is not up when you do run the setup command.

- **autologin**

  In this mode, credentials are stored on the EM CLI client system. Autologin mode is preserved until emcli logout is executed. If the session has expired when a verb is executed, login is automatically performed and the verb is executed.

  Verbs executed after emcli logout may fail with the message "Error: Session expired. Run emcli login to establish a session." You need to run the login verb to log in to EM CLI after an emcli logout. After the Enterprise manager user's password has changed, you need to log in with the ID and the new password. The new password will subsequently be stored.

  Note that noautologin is the default mode.

- **noautologin**

  In this default mode, credentials are not stored on the EM CLI client system. If the session has expired when a verb is executed, you have to explicitly run the login verb and then run the required verb.

- **noregister**

  Does not register this EM CLI instance.

- **custom_attrib_file**

Path name of a file containing Audit Custom Attribute values. This option is required when the OMS is configured for Audit Custom Attributes. If you do not provide custom_attrib_file, you are prompted to enter the values of the custom attributes.

The file can contain up to three lines, each containing the description of one custom attribute. Each line should be of the form:

```
<attr-name>#<attr-displayname>#<isMandatory>#<attr-value>
```

- **#** — Field separator.
- **attr-name** — Name of the attribute.
- **attr-displayname** — Display name of the attribute.
- **isMandatory** — 1 if the attribute is mandatory, otherwise 0.
- **attr-value** — Value of the custom attribute.

Examples

```
emcli setup -url=http://omsmachine.example.com:7770/em -username=sysman
```

To configure the EM CLI Client to function with multiple OMSes by implementing multiple setups, do the following:

1. Set up the EM CLI client for OMS1 at location dir1:

   ```
   emcli setup -dir=<dir1> -url=<Url of OMS1> -user=<EM Username for OMS1>
   ```

2. Set up the EM CLI client for OMS2 at location dir2:

   ```
   emcli setup -dir=<dir2> -url=<Url of OMS1> -user=<EM Username for OMS2>
   ```

3. Set the environment variable EMCLI_STATE_DIR to point to the setup directory for OMS1:

   ```
   setenv EMCLI_STATE_DIR <dir1>
   ```

   This sets the EM CLI Client to function with OMS1.

4. Set the environment variable EMCLI_STATE_DIR to point to the setup directory for OMS2:

   ```
   setenv EMCLI_STATE_DIR <dir2>
   ```

   This sets the EM CLI Client to function with OMS2.

setup_bipublisher

Sets up a relationship between Enterprise Manager and a BI Publisher Web Application. If a relationship already exists, you must provide the -force option. The Enterprise Manager System Reports are deployed to the newly configured BI Publisher Web Application. To just change the registration details without deploying the reports, use the -nodeploy option. Detailed status messages are provided for all operations.

Use the -force option to overwrite existing copies of reports if they exist. If you do not want to deploy following setup, you can specify the -nodeploy option.

> **Note:** This verb requires Enterprise Manager Super Administrator privileges.

Format

```
emcli setup_bipublisher
    [-force]
    -protocol=http|https
    -host=<hostname>
    -port=<portnumber>
    -uri=xmlpserver
    [-nodeploy]

[ ] indicates that the parameter is optional
```

Parameters

- **force**

  Overwrites existing copies of reports if they exist. The following scenarios are affected by this option:

  - If a relationship exists between Enterprise Manager and a BI Publisher Web Application, this option overrides it with a new relationship (-host, -port, and so forth).
  - If you do not specify -nodeploy, this option causes deployed reports to overwrite any that may already exist for the BI Publisher Web Application in the "Enterprise Manager Cloud Control" folder.

- **protocol**

  Must be either http or https.

- **host**

  Name of the host that is running, or server load balancer fronting the BI Publisher Web Application.

- **port**

  Port number of the web service.

- **uri**

  Web application context root, which must be xmlpserver.

- **nodeploy**

Specify if you do not want to deploy following setup. Suppresses the Enterprise Manager BI Publisher System Reports to the BI Publisher Web Application. You can do this later with the deploy_bipublisher_reports cli.

Examples

Example 1

```
emcli setup_bipublisher
     -protocol=https
     -host=www.somehost.com
     -port=7801
     -uri=xmlpserver
```

Example 2

This example reconfigures the BI Publisher Managed Server (BIP) inside the WebLogic Server console to listen on a different port (9704):

```
emcli setup_bipublisher -protocol=https -host=somehost.com -port=9704
-uri=xmlpserver -force -nodeploy
```

Example 3

This example sets up BI Publisher behind a load balancer with a host name of slb.somedomain.com on port 9754:

```
emcli setup_bipublisher -proto=https -host=slb.somedomain.com -port=9754
-uri=xmlpserver -force -nodeploy
```

show_bda_clusters

Lists all Hadoop clusters in the BDA network. If a host is specified, lists all Hadoop clusters in the network where the host is present.

Format

```
emcli show_bda_clusters
        [-host="host_name"]

[ ] indicates that the parameter is optional
```

Parameters

- **host_name**

  Name of a particular host in the BDA network.

Examples

Example 1

The following example lists all Hadoop cluster targets in the BDA network:

```
emcli show_bda_clusters
```

Example 2

The following example lists all Hadoop clusters in the network where the host `acme101.com` is present:

```
emcli show_bda_clusters
         -host="acme101.com"
```

show_audit_settings

Shows the following details of the current audit settings:

— Audit Switch
— Externalization Switch
— Directory
— File Prefix
— File Size
— Data Retention Period

Format

```
emcli show_audit_settings
      -view="SUMMARY|DETAIL"
```

show_credential_set_info

Displays the parameters of credential sets defined with target types.

Format

```
emcli show_credential_set_info
       [-target_type="<target_type>"]
       [-set_name="<credential_set_name>"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target. The default is to display the credential set defined for all target types.

- **set_name**

  Name of the credential set. The default is to display all credential sets defined for a target type.

Examples

Example 1

This example displays the details of all credential sets defined with all target types:

```
emcli show_credential_set_info
```

Example 2

This example displays all credential sets defined with the oracle_database target type:

```
emcli show_credential_set_info -target_type=oracle_database
```

Example 3

This example displays the details of the HostUDMCreds credential set defined for the host target type.

```
emcli show_credential_set_info -target_type=host
      -set_name=HostUDMCreds
```

show_credential_type_info

Displays the parameters of credential types defined for target types.

Format

```
emcli show_credential_type_info
       [-target_type="<target_type>"]
       [-type_name="<credential_type_name>"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target. The default is to display the credential set defined for all target types.

- **type_name**

  Name of the credential type. The default is to display all credential types defined for a target type.

Examples

Example 1

This example displays the details of all credential types defined with all target types:

```
emcli show_credential_type_info
```

Example 2

This example displays all credential types defined with the oracle_database target type:

```
emcli show_credential_type_info -target_type=oracle_database
```

Example 3

This example displays the details of the HostUDMCreds credential type defined for the oracle_database target type.

```
emcli show_credential_type_info -target_type=oracle_database
      -type_name=HostUDMCreds
```

Example 4

This example shows output for various credential types.

```
emcli show_credential_type_info -target_type=host

Target Type   Cred Type Name  Cred Type Column Name    Key Column
host          HostCreds       HostPassword             No
                              HostUserName             Yes
              HostSSHCreds    SSH_PUB_KEY              No
                              SSH_PVT_KEY              No
                              USERNAME                 Yes
              ProvisionCreds  InstallPassword          No
                              InstallUserName          Yes
                              OMSRegistrationPassword  No
```

	ProvCompPasswd	No
WBEMCreds	WBEMPassword	No
	WBEMUserName	Yes

show_operations_list

Shows the list of all auditable Enterprise Manager operations names.

Format

```
emcli show_operations_list
```

Output

Output appears as shown in This example:

```
ADD_AGENT_REGISTRATION_PASSWORD
AGENT_REGISTRATION_PASSWORD_USAGE
AGENT_RESYNC
APPLY_TEMPLATE
AUDIT_EXPORT_SETTINGS
AUDIT_SETTINGS
CHANGE_PASSWORD
CHANGE_PREFERRED_CREDENTIAL
CREATE_PG_SCHED
CREATE_ROLE
CREATE_TEMPLATE
CREATE_UDP
CREATE_UDPG
CREATE_USER
DELETE_AGENT_REGISTRATION_PASSWORD
DELETE_JOB
DELETE_PG_EVAL
DELETE_PG_SCHED
DELETE_ROLE
DELETE_TEMPLATE
DELETE_UDP
DELETE_UDPG
DELETE_USER
EDIT_AGENT_REGISTRATION_PASSWORD
EDIT_JOB
EDIT_PG_SCHED
EDIT_TEMPLATE
EDIT_UDP
EDIT_UDPG
EVALUATE_UDP
FILE_TRANSFER
GET_FILE
GRANT_JOB_PRIVILEGE
GRANT_ROLE
GRANT_SYSTEM_PRIVILEGE
GRANT_TARGET_PRIVILEGE
IMPORT_UDP
JOB_OUTPUT
LOGIN
LOGOUT
MODIFY_METRIC_SETTINGS
MODIFY_POLICY_SETTINGS
MODIFY_ROLE
MODIFY_USER
PUT_FILE
REMOTE_OPERATION_JOB
REMOVE_PRIVILEGE_DELEGATION_SETTING
REPOSITORY_RESYNC
```

```
REVOKE_JOB_PRIVILEGE
REVOKE_ROLE
REVOKE_SYSTEM_PRIVILEGE
REVOKE_TARGET_PRIVILEGE
SAVE_MONITORING_SETTINGS
SET_PRIVILEGE_DELEGATION_SETTING
SUSPEND_JOB
```

show_patch_plan

Shows the details of a particular patch plan.

Format

```
emcli show_patch_plan
        -name="name"
        [-info [-showPrivs]] [-actions [-onlyShowEnabled]]
        [-patches]
        [-targets]
        [-deplOptions]
        [-analysisResults]
        [-conflictFree]
        [-impactedTargets]
        [-deploymentProcedures]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Plan name. If you only provide this parameter with no other options, the full details of the patch plan are shown.

- **info**

  Shows the generic information of the given patch plan.

- **show_Privs**

  Shows the user privileges on the given patch plan along with the generic information.

- **actions**

  Show the actions that are possible to be taken on the given patch plan.

- **onlyShowEnabled**

  Only show the enabled actions on the given patch plan.

- **patches**

  Shows details of the patches contained in the given patch plan.

- **targets**

  Shows details of the targets contained in the given patch plan.

- **deplOptions**

  Shows details of the deployment options contained in the given patch plan.

- **analysisResults**

  Shows details of the analysis results of the given patch plan.

- **conflictFree**

  Shows details of the conflict-free patches of the given patch plan.

- **impactedTargets**

  Shows details of the impacted targets of the given patch plan.

- **deploymentProcedures**

  Shows the deployment procedure of the given patch plan.

Examples

```
emcli show_patch_plan -name="plan name"

emcli show_patch_plan -name="plan name" -info

emcli show_patch_plan -name="plan name" -actions -onlyShowEnabled

emcli show_patch_plan -name="plan name" -info -showPrivs
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

signoff_agents

Performs Agent sign-off prerequisites and submits the Agent sign-off job.

Format

```
emcli signoff_agents
      -agents="List_of_agents" | -input_file="agents_file:Location of_output file"
      [-job_name="Name_of_job"]

[ ] indicates that the parameter is optional
```

Parameters

- **agents**

  Submits a job to clean up old Agent homes matching Agent names or an Agent names pattern separated by commas.

- **input_file**

  Checks whether Agents specified in the file are available for sign-off, and submits the Agent sign-off job.

  You can pass all of these parameters in a response file. The usage is:

  ```
  -input_file="response_file:/scratch/response_file.txt"
  ```

  You must provide the file name with the full path, and each parameter should be given in each line. If you pass a parameter both in the command line and in a response file, the command-line option is given precedence.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **job_name**

  Submits the clean-up job with the job name specified in this option.

Examples

Example 1

This example submits a job to clean up the old Agent homes on Agent names matching the pattern abc% and on the xyz.domain.com Agent.

```
emcli signoff_agents -agents="abc%,xyz.domain.com:1243"
```

Example 2

This example submits a job to clean up the old Agent homes on the Agents specified in the file.

```
emcli signoff_agents -input_file="agents_file:/scratch/agents_file.txt"
```

Example 3

This example submits job cleanup_123 to clean up the old Agent homes on Agent names matching the pattern abc% and on the xyz.domain.com Agent.

```
emcli signoff_agents -agents="abc%,xyz.domain.com:1243" -job_name="cleanup_123"
```

stage_swlib_entity_files

Stages files of an entity from the Software Library to a host target.

Format

```
emcli stage_swlib_entity_files
        -entity_rev_id="entity_rev_id"
        -host="hostname"
        [-file="<file name specified during upload>"
        [-credential_set_name="setname"] | [-credential_name="name"
            -credential_owner="owner"]
        [-stage_path="<storage location name>;<storage type>"]
        [-use_latest_revision]
        [-overwrite_files]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_rev_id**

  Identifier of the entity revision. The Software Library Home page exposes the identifier for folders and entities as a custom column (Internal ID). However, this is hidden by default.

- **host**

  Target name of the host where the files are to be staged.

- **file**

  Name of the file associated with the entity revision.

- **credential_set_name**

  Set name of the preferred credential stored in the Management Repository for the host target. The value can be one of the following:

  – HostCredsNormal — Default unprivileged credential set

  – HostCredsPriv — Privileged credential set

- **credential_name**

  Name of a named credential stored in the Management Repository. You must specify this option with the -credential_owner option.

- **credential_owner**

  Owner of a named credential stored in the Management Repository. You must specify this option with the -credential_name option.

- **stage_path**

  Location on the host where the files are to be copied.

- **use_latest_revision**

  Indicates that the upload should occur to the latest revision of the entity instead of the revision identified by entity_rev_id.

- **overwrite_files**

Indicates that the file should be overwritten in the stage location. By default, files will not be overwritten.

Examples

This example copies the file 'myfile.zip' associated with the specified entity revision to '/u01/stage_loc' on host 'fs1.us.acme.com' using the named credential 'MyAcmeCreds' owned by 'ACME_USER.'

```
emcli stage_swlib_entity_files
        -entity_rev_id="oracle:defaultService:em:provisioning:1:cmp:
           COMP_Component:SUB_Generic:B1B1880C6A8C62AAE040548C42832D14:0.1"
        -file="myfile.zip"
        -stage_path="/u01/stage_loc"
        -host="fs1.us.acme.com"
        -credential_name="MyAcmeCreds"
        -credential_owner="ACME_USER"
```

start_agent

Starts up a Management Agent. This verb requires operator privilege or full privilege on the Management Agent.

Format

```
emcli start_agent
        -agent_name="agent_target_name"
        [-host_username="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name="credential_name"]
        [-credential_setname="credential_setname_of_agent"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **host_username**

  User name of the OS user (on the host) who owns the Management Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Management Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Management Agent. Example: "HostCreds".

Examples

Example 1

```
emcli start_agent -agent_name="agent.example.com:1234"
                          -host_username="test_user"
                          -host_pwd="test"
```

Example 2

```
emcli start_agent -agent_name="agent.example.com:1234"
                          -credential_name="MyMachineCredential"
```

Example 3

```
emcli start_agent -agent_name="agent.example.com:1234"
                          -credential_setname="HostCreds"
```

status

Shows whether EM CLI is configured or not, and shows the EM CLI setup details. It also displays the Java home, version, EM CLI home, and all of the EM CLI configuration detials if it is configured.

Command-Line Format

```
emcli status
```

Scripting and Interactive Format

```
status()
```

Parameters

None.

Output

This example shows output when EM CLI setup has not been done:

```
Oracle Enterprise Manager Cloud Control 12c Release 12.1.0.0.0.
Copyright (c) 1996, 2011 Oracle Corporation and/or its affiliates. All rights
reserved.

Instance Home : /home/sumadas
Status        : Not Configured
```

This example shows output after EM CLI setup has been done:

```
Oracle Enterprise Manager Cloud Control 12c Release 12.1.0.0.0.
Copyright (c) 1996, 2013 Oracle Corporation and/or its affiliates. All rights
reserved.

Instance Home           : /ade/sumadas_emcli/oracle/work/.emcli
Status                  : Configured
EMCLI Home              : /ade/sumadas_emcli/emcore/emcli/bin
EMCLI Version           : 12.1.0.0.0
Java Home               : /ade_autofs/nfsdo_base/EMGC/MAIN/LINUX/110811/jdk6/jre
Java Version            : 1.6.0_24
Log file                : /ade/sumadas_emcli/oracle/work/.emcli/.emcli.log
EM URL                  : https://dadvma0121.example.com:14487/em
EM user                 : SYSMAN
Auto login              : true
Trust all certificates  : true
```

This example shows output in interactive shell mode:

```
emcli>status()
Oracle Enterprise Manager 12c EM CLI with Scripting option Version 12.1.0.3.0.
Copyright (c) 1996, 2013 Oracle Corporation and/or its affiliates. All rights
reserved.

Verb Jars Home (EMCLI_VERBJAR_DIR)      : <EMCLI_
LOCATION>/int/./bindings/12.1.0.3.0/.emcli
EM CLI Home (EMCLI_INSTALL_HOME)        : <EMCLI_LOCATION>/int/.
EM CLI Version                          : 12.1.0.3.0
Java Home                               : /jdk6/jre
Java Version                            : 1.6.0_43
```

```
Log file (EMCLI_LOG_LOC)                : CONSOLE
Log level (EMCLI_LOG_LEVEL)             : SEVERE
EM URL (EMCLI_OMS_URL)                  : https://<hostname>:<port>/em
EM user (EMCLI_USERNAME)                : sysman
Auto login (EMCLI_AUTOLOGIN)            : false
Trust all certificates (EMCLI_TRUSTALL) : true
```

stop_agent

Shuts down a Management Agent. This verb requires operator privilege or full privilege on the Agent.

Format

```
emcli stop_agent
        -agent_name="agent_target_name"
        [-host_username="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name="credential_name"]
        [-credential_setname="credential_setname_of_agent"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **host_username**

  User name of the OS user (on the host) who owns the Management Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Management Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Management Agent. Example: "HostCreds".

Examples

Example 1

```
emcli stop_agent -agent_name="agent.example.com:1234"
                          -host_username="test_user"
                          -host_pwd="test"
```

Example 2

```
emcli stop_agent -agent_name="agent.example.com:1234"
                          -credential_name="MyMachineCredential"
```

Example 3

```
emcli stop_agent -agent_name="agent.example.com:1234"
                          -credential_setname="HostCreds"
```

stop_blackout

Stops a blackout.

You can stop a blackout before it has fully started, for example, when it has a "Scheduled" status. You can also stop a blackout while it is in effect.

Format

```
emcli stop_blackout
      -name="name"
      [-createdby="blackout_creator"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the blackout to stop.

- **createdby**

  Enterprise Manager user who created the blackout. The default is the current user. The `SUPER_USER` privilege is required to stop a blackout created by another user.

Examples

Example 1

This example stops blackout `backup_db3` created by the current user.

```
emcli stop_blackout -name=backup_db3
```

Example 2

This example stops blackout `weekly_maint` created by user `joe`. The current user must either be user `joe` or a user with the `SUPER_USER` privilege.

```
emcli stop_blackout -name=weekly_maint -createdby=joe
```

stop_instance

Stops a scheduled, failed, or running deployment instance.

Format

```
emcli stop_instance
      [-instance=<instance_guid>]
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
```

Parameters

- **instance**

  GUID of the instance.

- **exec**

  GUID of the execution.

- name

  Name of the execution.

- **owner**

  Owner of the execution.

Examples

```
emcli stop_instance -instance=16B15CB29C3F9E6CE040578C96093F61
```

stop_job

Stops a specified job. You can use the `get_jobs` verb to obtain a list of job IDs and names.

Format

```
emcli stop_job
      [-job_id="ID1;ID2;..."]
      [-name="job_name_pattern"]
      [-owner="job_owner"]
      [-type="job_type"]
      [-targets="target_name:target_type"]
      [-input_file=property_file:"filename"]
      [-preview]

[ ] indicates that the parameter is optional
```

Parameters

- **job_id**

  Semi-colon (;) separated list of job(s) to stop.

  Note: This filter cannot be used with other filters.

- **name**

  Name or pattern of the job(s) to stop.

- **owner**

  Owner of the job(s).

- **type**

  Job type of the job(s).

- **targets**

  Targret name and target type of the job(s) to stop.

- **input_file**

  The properties for filtering jobs may be specified in "filename."

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **preview**

  Lists only the jobs to stop.

Examples

Example 1

This example stops a job with the specified ID.

```
emcli stop_job -job_id=12345678901234567890123456789012
```

Example 2

This example stops all jobs owned by the Administrator "Jennifer".

```
emcli stop_job -owner=Jennifer
```

Example 3

This example stops a job named "BACKUP_WEDNESDAY", owned by the logged-in administrator and scheduled to run in the future.

```
emcli stop_job -name=Backup_Wednesday
```

stop_siteguard_health_checks

Retrieves and shows the configured limit for Apply lag and Transport lag for all or selected databases of the system.

Format

```
emcli stop_siteguard_health_checks
        [-operation_plan=name_of_the_operation_plan]

[ ] indicates that the parameter is optional
```

Parameter

- **operation_plan**

  Name of the operation plan for which execution of health checks must be stopped.

Examples

Example 1

This example displays the details of the Apply lag limit configured on all of the databases of the system `austin-system`:

```
emcli get_siteguard_lag
        -system_name="austin-system"
        -property_name="ApplyLag"
```

Example 2

This example stops health checks for operation plan `austin-switchover`:

```
emcli stop_siteguard_health_checks
        -operation_plan="austin-switchover"
```

submit_add_host

Submits an Add Host session that installs management Agents on unmanaged hosts, thereby converting them to managed hosts.

Format

```
emcli submit_add_host
      -host_names=<host_list>
      -platform=<platform_id>
      -installation_base_directory=<installation_base_directory>
      -credential_name=<credential_name>
      [-instance_directory=<instance_directory>]
      [-credential_owner=<credential_owner>]
      [-properties_file=<properties_file>]
      [-session_name=<deployment_session_name>]
      [-privilege_delegation_setting=<privilege_delegation_setting>]
      [-port=<agent_port>]
      [-deployment_type=FRESH|SHARED|CLONE]
      [-preinstallation_script=<preinstallation_script_location>]
      [-preinstallation_script_on_oms]
      [-preinstallation_script_run_as_root]
      [-postinstallation_script=<postinstallation_script_location>]
      [-postinstallation_script_on_oms]
      [-postinstallation_script_run_as_root]
      [-additional_parameters=<parameter1 parameter2 parameter3 .... >]
      [-wait_for_completion]
      [-source_agent=<clone_source_agent_name>]
      [-master_agent=<master_agent_name>]

[ ] indicates that the parameter is optional
```

Parameters

- **host_names**

  Names of the hosts where the Agents need to be installed, separated by a semi-colon.

- **platform**

  ARU platform ID of the hosts where the Agent needs to be installed. To show the list of supported agent platforms, run the command emcli list_add_host_platforms -all.

- **installation_base_directory**

  Directory where you want to install the Agent. Provide this parameter in double-quotes if it is an MS-DOS/Windows-style path.

- **credential_name**

  Named credential to be used for installing the Agent.

- **instance_directory**

  Instance directory of the Agent. Provide this parameter in double-quotes if it is an MS-DOS/Windows-style path.

- **credential_owner**

  Owner of the named credential owner.

- **session_name**

  Session name that uniquely identifies the Add Host session.

- **privilege_delegation_setting**

  Privilege delegation setting you want to use to install an Agent and run the root script.

- **port**

  Port on which the Agent should communicate with the OMS.

- **deployment_type**

  Type of Agent deployment, which can be FRESH, CLONE, or SHARED. The default is FRESH.

- **preinstallation_script**

  Script you want to run before installing the Agent. Provide this parameter in double-quotes if it is an MS-DOS/Windows-style path.

- **preinstallation_script_on_oms**

  Use this option if the pre-installation script resides on the OMS host.

- **preinstallation_script_run_as_root**

  Use this option if you want to run the pre-installation script as the root user.

- **postinstallation_script**

  Script to run after installing the Agent. Provide this parameter in double-quotes if it is an MS-DOS/Windows-style path.

- **postinstallation_script_on_oms**

  Use this option if the post-installation script resides on the OMS host.

- **postinstallation_script_run_as_root**

  Use this option if you want to run the post-installation script as the root user.

- **additional_parameters**

  Additional parameters you want to use to install an Agent.

- **wait_for_completion**

  Runs the Add Host operation synchronously. If you specify this option, the command waits until the add host session completes before returning control to you on the command line.

- **source_agent**

  Source Agent you want to use to install a cloned Agent. The source Agent name should have the format of "agent host name:agent port". For example: foo.example.com:3872 .

- **master_agent**

  Master Agent you want to use to install a shared Agent. The master Agent name should have the format of "agent host name:agent port". For example: foo.example.com:3872 .

Examples

Example 1

This example submits an Add Host session on the host 'example.com', having platform ID '226' with '/opt/agent' as the installation base directory,using the named credential 'oracle' and privilege delegation setting /usr/bin/sudo -u %RUNAS% %COMMAND%.

```
emcli submit_add_host -host_names="example.com" -platform=226 -credential_
name=oracle -installation_base_directory=/opt/agent -privilege_delegation_
setting="/usr/bin/sudo -u %RUNAS% %COMMAND%"
```

Example 2

This example submits an Add Host session on the host 'example2.com', having platform ID '233' with 'C:\agent' as the installation base directory, and using the named credential 'oracle'.

```
emcli submit_add_host -host_names=example2.com -platform=233 -installation_base_
directory="C:\agent" -credential_name=oracle
```

Example 3

This example submits an Add Host session using the inputs provided in the properties file '/opt/inputs.txt'.

```
emcli submit_add_host -properties_file=/opt/inputs.txt
```

The contents of the inputs.txt file is as follows:

```
host_names="example1.com;example2.com"
platform=226
credential_name=oracle
installation_base_directory=/opt/agent
privilege_delegation_setting="/usr/bin/sudo -u %RUNAS% %COMMAND%"
```

Example 4

This example submits an Add Host session of type 'CLONE' on the host 'example.com', having platform ID '226' with '/opt/agent' as the installation base directory. 'example1.com:3872' is the source agent, using the named credential 'oracle'.

```
emcli submit_add_host -host_names=example.com -platform=226 -installation_base_
directory=/opt/agent -credential_name=oracle
-deployment_type=CLONE -source_agent=example1.com:3872
```

Example 5

This example submits an Add Host session of type 'SHARED' on the host 'example.com', having platform ID '226' with '/opt/agent' as the instance directory. 'example1.com:3872' is the master agent, using the named credential 'oracle'.

```
emcli submit_add_host -host_names=example.com -platform=226 -instance_
directory=/opt/agent -credential_name=oracle
-deployment_type=SHARED -master_agent=example1.com:3872
```

submit_job

Creates and submits a job. This verb has been deprecated in favor of create_job. For more information, refer to this verb in this chapter, or enter:

```
emcli help create_job
```

submit_masking_job

Submits a masking job and returns the display job ID and execution ID.

Format

```
emcli submit_masking_job
    -definition_name=<masking_defn_name>
    -target_name=<database_target_name>
    -target_type=<database_target_type>
    -parameters=name1:value1;name2:value2;...
    [-host_pref_creds_name=<preferred_host_credentials_name>
      OR -host_cred_name=<host_credential_name>]
    [-db_pref_creds_name=<preferred_db_credentials_name>
      OR -db_cred_name=<db_credential_name>]
    [-encryption_key=<encryption_key_string]
    [-script_file_location=<script_file_location>]
    [-script_file_name=<script_file_name>]
    [-input_file=PWD_FILE_TAG:<credentials_file_name>]
    [-script | -format=[name:<pretty|script|csv>];
         [column_separator:"column_sep_string"];
         [row_separator:"row_sep_string"];
    ]

[ ] indicates that the parameter is optional
```

Note: Unless values for the Host and DB credentials are specified in the -parameters parameter, either the host_pref_creds_name or host_cred_name parameter should be specified. Similarly, either the db_pref_creds_name or the db_cred_name parameter should be specified.

Parameters

- **definition_name**

  Masking definition name.

- **target_name**

  Database target name to mask.

- **target_type**

  Database target type to mask.

- **parameters**

  List of name-value pairs that represent the credentials required to connect to the database instance. The supported parameters are 'db_username', 'db_password', 'db_role', 'db_cred_name', 'host_username', 'host_password', and 'host_cred_name'. If PDP needs to be used, additional parameters to be specified are 'PDP','RUNAS', and 'PROFILE'. The 'PROFILE' option is only applicable for Powerbroker.

- **host_pref_creds_name**

  Type of preferred credentials to use to connect to the database host, which can either be HostCredsNormal or HostCredsPriv.

- **host_cred_name**

Credential name to use to connect to the database host.

- **db_pref_creds_name**

  Type of preferred credentials to use to connect to the database instance, which can either be DBCredsNormal or DBCredsSYSDBA.

- **db_cred_name**

  Credential name to use to connect to the database instance.

- **encryption_key**

  Specify an encryption key if the masking definition involves usage of a substitute format.

- **script_file_location**

  Location where the SQL script is to be copied and executed. Default values of $ORACLE_HOME/dbs are used if a value is not specified.

- **script_file_name**

  Name of the script file to store the masking SQL script. If you do not specify a name, a system-generated file name is used.

- **input_file**

  Used in conjunction with the 'parameters' option, this option enables you to store parameter values, such as user name and password, in a separate file. The 'input_file' option specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific parameter values of the 'parameters' option. You can specify multiple -input_file parameters. The result would be a combination of all of the files.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **script**

  This option is equivalent to -format="name:script" .

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

Example 1

This example submits a masking job for the definition name MASKING_DEF and returns the job ID and execution ID.

```
emcli submit_masking_job -definition_name=MASKING_DEF -target_name=testdb -target_
type=oracle_database -parameters="db_username:sys;db_password:password;db_
role:SYSDBA;db_cred_name:DBCREDS;host_username:test;host_password:password;host_
cred_name:HOSTCREDS"
```

Example 2

This example takes the credentials from the provided input files host_creds.txt and db_creds.txt.

```
emcli submit_masking_job -definition_name=MASKING_DEF -target_name=testdb -target_
type=oracle_database  -parameters="HOST_CREDS;DB_CREDS" -input_file=HOST_
CREDS:host_creds.txt -input_file=DB_CREDS:db_creds.txt
```

It is also possible to specify both of the credentials in one file and use only one -input_ file tag. If PDP must be used, you must provide values in the parameters/input_file as follows:

- SUDO:

  ```
  db_username:sys;db_password:password;db_role:SYSDBA;host_username:user2;host_
  password:password;PDP:SUDO;RUNAS:user1
  ```

- POWERBROKER:

  ```
  db_username:sys;db_password:password;db_role:SYSDBA;host_username:user2;host_
  password:password;PDP:POWERBROKER;RUNAS:user1;PROFILE:profile
  ```

Example 3

This example uses the named database credential DB_NC, the named host credential HOST_NC, and submits the masking job. If the masking definition involves usage of the substitute format, uses the encryption key as 'abcd'.

```
emcli submit_masking_job -definition_name=email2 -target_name=testdb -target_
type=oracle_database -db_cred_name=DB_NC -host_cred_name=HOST_NC -encryption_
key=abcd -script_file_location=/tmp -script_file_name=email1.sql
```

Example 4

This example uses the saved default preferred host and database credentials. If the masking definition involves usage of the substitute format, it uses the encryption key as 'abcd'. It submits a masking job for the given definition name and returns job id and execution id.

```
emcli submit_masking_job -definition_name=MASKING_DEF -target_name=testdb -target_
type=oracle_database -host_pref_creds_name="HostCredsPriv" -db_pref_creds_
name="DBCredsSYSDBA"
```

submit_operation_plan

Submits the specified operation plan for execution.

Format

```
emcli submit_operation_plan
      -name=<operation_plan_name>
      [-run_prechecks=true|false]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the operation plan.

- **run_prechecks**

  Optionally run pre-checks by specifying either true or false.

Examples

```
emcli submit_operation_plan
      -name="austin-switchover"
      -run_prechecks="true"
```

See Also

emcli create_operation_plan

emcli get_operation_plans

submit_patch_plan

Submits action on a given patch plan, such as analyzing, preparing, deploying, and switchbacking, or finds the next action automatically, then runs it.

Format

```
emcli submit_patch_plan
        -name="name"
        -action="action name"
```

Parameters

- **name**

  Patch plan name.

- **action**

  Action to submit on the given patch plan.

Examples

```
emcli submit_patch_plan -name="plan name"

emcli submit_patch_plan -name="plan name" -action="analyze"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

submit_procedure

Submits a deployment procedure or a pre-saved procedure configuration.

Format

```
emcli submit_procedure
      -input_file=data:"file_path"
      [-procedure="procedure_guid"]
      [-name="procedure_name"]
      [-owner="procedure_owner"]
      [-parent_proc="procedure_of_procedure_config"]
      [-instance_name="procedure_instance_name"]
      [-grants="users_and_their_corresponding_access_levels"]
      [-schedule=
          start_time:yyyy/MM/dd HH:mm;
          tz:{java timezone ID};
          grace_period:xxx;
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Input data for the Deployment Procedure. The `file_path` should point to a file containing the data properties file.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **procedure**

  GUID of the procedure to execute.

- **name**

  Name of the procedure or procedure configuration.

- **owner**

  Owner of the procedure or procedure configuration.

- **parent_proc**

  Procedure of the procedure configuration, this applies to a procedure configuration when there is both a procedure and a procedure configuration with the same name.

- **instance_name**

  Name of the procedure instance.

- **grants**

  Users and their corresponding access levels designated as a string of user:privilege pairs each separated by ; .

  where:

  user = Enterprise Manager user name

  privilege = VIEW_JOB or FULL_JOB

- **schedule**

  Schedule for the deployment procedure. If not specified, the procedure is executed immediately.

  `start_time` — When the procedure should start

  `tz` — Optional time zone ID

Output Columns

Instance GUID

Examples

```
emcli submit_procedure -input_file=data:data.properties
-procedure=16B15CB29C3F9E6CE040578C96093F61 -grants="user1:VIEW_JOB;user2:
FULL_JOB" -schedule="start_time:2006/6/21 21:23;tz:America/New_York;
grace_period:60" -instance_name="MyProcedureInstance_0001"
```

subscribeto_rule

Subscribes the user to a rule with email notification.

It is not an error to specify email addresses that are already in the `assignto` user's preferences.

A message appears if the outgoing mail server (SMTP) has not been set up. When you specify the `-fail_if_no_mail_server`, this condition is an error and prevents the subscribe from occurring; otherwise, this condition is a warning that does not affect the success of this command.

Format

```
emcli subscribeto_rule
        -ruleset_name="ruleset_name"
        -rule_name="rule_name"
        -owner="rule_owner"
        [-assignto="em_username"]
        [-email="email_address";...]
        [-fail_if_no_mail_server]

[ ] indicates that the parameter is optional
```

Parameters

- **ruleset_name**

  Name of the incident rule set.

- **rule_name**

  Name of the rule.

- **owner**

  Owner of the rule set.

- **assignto**

  User to subscribe to the notification rule. If the `assignto` user is not the current user, or if the owner of the rule is not the current user, the super-user privilege is needed. The default is the current user.

- **email**

  List of email addresses to associate with the rule to which the `assignto` user is being subscribed. These addresses are first added to the preferences of the `assignto` user (duplicates are ignored) before being assigned to the notification rule. The email addresses are added only if the current user has the privilege to subscribe the `assignto` user to the rule.

- **fail_if_no_mail_server**

  A message appears if the outgoing mail server (SMTP) has not been set up. When you specify the `-fail_if_no_mail_server` option, this condition is an error and prevents the subscribe from occurring; otherwise, this condition is a warning that does not affect the success of this command.

Examples

Example 1

This example subscribes the current user to the rule `"Agent Upload Problems"` using the current user's email addresses for notification. The current user must have the `SUPER_USER` (or have `sysman`) privilege for this to succeed, since `sysman` owns the rule. Also, the current user must already have at least one email address in his/her preferences for this command to succeed.

```
emcli subscribeto_rule -name="Agent Upload Problems" -owner=sysman
```

Example 2

This example first adds the two specified email addresses to the preferences for user `joe`. Then user `joe` is subscribed to the rule `"Agent Upload Problems"` using joe's email addresses for notification. The current user must have `SUPER_USER` privilege (or be `joe`) for this command to succeed.

```
emcli subscribeto_rule -name="Agent Upload Problems" -owner=sysma
      -assignto=joe -email="joe@work.com;joe@home.com"
```

suspend_instance

Suspends a running deployment instance.

Format

```
emcli suspend_instance
      -instance=<instance_guid>
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  GUID of the instance.

- **exec**

  GUID of the execution.

- **name**

  Name of the execution.

- **owner**

  Owner of the execution.

Examples

```
emcli suspend_instance -instance=16B15CB29C3F9E6CE040578C96093F61
```

suspend_job

Suspends a job or set of jobs matching the filter criteria. Executions on any of the targets and scheduled to start within the beginning and ending time window are suspended.

Format

```
emcli suspend_job
   [-name="job_name_pattern"]
   [-owner="job_owner"]
   [-type="job_type"]
   [-targets="target_name:target_type"]
   [-input_file=property_file:"filename"]
   [-preview]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name or pattern of the job(s) to suspend.

- **owner**

  Owner of the job(s).

- **type**

  Job type of the job(s).

- **targets**

  Target name and target type of the job(s).

- **input_file**

  Specify the filtering properties of the file in "filename."

  Any jobs matching all the specified filter criteria are resumed. You must specify at least one filter, and the logged in administrator must have the necessary privileges on the matching jobs.

  If the property file is provided, criteria can be read from it as well as the command line. You can specify the execution targets and/or starting and ending time window in this file. All other properties in this file are ignored.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **preview**

  Only lists the jobs that would be suspended.

Examples

Example 1

This example suspends a job named MYJOB.

```
emcli suspend_job -name=MyJob
```

Example 2

This example suspends all jobs owned by User1.

```
emcli suspend_job -owner=User1
```

Example 3

This example suspends all jobs of type Backup whose name starts with BK.

```
emcli suspend_job -name=BK% -type=Backup
```

Example 4

This example suspends all jobs on database target orcl_123.

```
emcli suspend_job -targets=orcl_123:oracle_database
```

Example 5

This example suspends jobs or job executions matching search criteria in suspend_prop.txt. If the property file contains job details, matching jobs are suspended. If the property file contains time or target details, matching executions are suspended. If the property file contains job, time, and target details, matching executions of the matching jobs are suspended.

```
emcli suspend_job -input_file=property_file:/tmp/suspend_prop.txt
```

switch_swlib_oms_agent_storage

Modify a Software Library OMS Agent storage location to change the associated OMS Host and the credential for accessing the location.

Format

```
emcli switch_swlib_oms_agent_storage
        -name="location_name"
        -host="hostname"
        [-credential_set_name="setname"] | [-credential_name="name"
           -credential_owner="owner"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of an existing OMS Agent storage location.

- **host**

  Target name of the OMS host where the file system path for the storage location exists.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target. The value can be one of the following:

  – HostCredsNormal — Default unprivileged credential set

  – HostCredsPriv — Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. You must specify this option with the -credential_owner option.

- **credential_owner**

  Owner of a named credential stored in the repository. You must specify this option with the -credential_name option.

Examples

This example modifies the OMS Agent file system storage location named 'myOMSAgtLocation' to use the specified host 'fs1.us.acme.com', and the named credential 'MyAcmeCreds' owned by 'ACME_USER' for reading/writing files from/to this location.'

```
emcli switch_swlib_oms_agent_storage
        -name="myOMSAgtLocation"
        -host="fs1.us.acme.com"
        -credential_name="MyAcmeCreds"
        -credential_owner="ACME_USER"
```

sync

Synchronizes the EM CLI client with an OMS. After synchronization, all verbs and associated command-line help available to this OMS become available at the EM CLI client. Synchronization occurs automatically during a call to setup.

There are two ways to perform sync:

- With parameters
- Without parameters

  sync connects to the same OMS against which it has been set up and downloads the latest jar files.

Command-Line Format

```
emcli sync
         [-url="http[s]://host:port/em"]
         [-username=<EM_username>]
         [-password=<EM_user's_password>]
         [-trustall]
         [-novalidate]

[ ] indicates that the parameter is optional.
```

Parameters

- **url**

  URL of the Enterprise Manager OMS. Both http and https are supported, but https is recommend for security purposes.

- **username**

  User name to be used by all subsequent EM CLI commands when contacting the OMS.

- **password**

  Enterprise Manager user's password. If you do not specify this option, you are interactively prompted for the password. Providing a password on the command line is insecure and should be avoided.

- **trustall**

  Automatically accepts any server certificate from the OMS, which results in lower security. Also indicates that the setup directory is local and trusted. Either pass this option or the set environment variable EMCLI_CERT_LOC, which has the certificate keystore file. If the file is not present, the system stores the certificate at this location.

- **novalidate**

  Does not authenticate the Enterprise Manager user name against the OMS. Assumes that the given username is valid.

Examples

This example synchronizes the EM CLI client with the OMS by connecting as Enterprise Manager user john_doe. The user is prompted for the password interactively.

```
emcli sync
          -url="https://mymachine.example.com"
          -username=john_doe
          -trustall
          -novalidate
```

sync_alerts

Synchronizes all alerts for the specified target between the Agent and the repository. You typically use this command when you think that the Agent has not uploaded the latest alert to the repository, and the repository is therefore out of sync with the Agent state.

To determine if alerts are out of sync between the Agent and the repository for the specified target, run the get_unsync_alerts command.

Format

```
emcli sync_alerts
        -target_type=type
        -target_name=name
        -agent_name=agent
```

Parameters

- **target_type**

  Internal target-type identifier (host, oracle_database, emrep, and so forth).

- **target_name**

  Name of the target.

- **agent_name**

  Name of the Agent.

Examples

Example 1

This example synchronizes alert states for target_type "host" and target_name "hostname.oracle.com".

```
emcli sync_alerts -target_type=host -target_name=hostname.oracle.com
```

Example 2

This example synchronizes alert states for all targets that the Agent "hostname.xyz.com:port" monitors.

```
emcli sync_alerts -agent_name=hostname.xyz.com:port
```

sync_beacon

Synchronizes a beacon that is monitoring the target (reloads all collections to the beacon).

Format

```
emcli sync_beacon
      -name=target name
      -type=target type
      -bcnName=beacon name
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **bcnName**

  Beacon name to synchronize.

Examples

This example synchronizes `MyBeacon`, which is monitoring the `MyTarget` target of type `generic_service`.

```
emcli sync_beacon -name='MyTarget' -type='generic_service'
      -bcnName='MyBeacon'
```

test_named_credential

Tests the named credentials provided in the list. Instance credentials are tested against the credential target. Global credentials are tested against the target provided.

Format

```
emcli test_named_credential
       -cred_names=<cred_name_list>
       [-target_name=<target_name>]
       [-target_type=<target_type>]
```

Parameters

- **cred_names**

  List of credential names to be tested.

- **target_name**

  Target name to test the global credentials. Instance credentials are tested against their respective targets.

- **target_type**

  Target type to test the global credentials.

Examples

Example 1

This example tests the instance named credentials NC1 owned by the current logged in user and NC2 owned by ADMIN1.

```
emcli test_named_credential
        -cred_names="NC1;NC2:ADMIN1"
```

Example 2

This example tests the global host named credentials NC1, NC2, and NC3 against the target testhost.example.com.

```
emcli test_named_credential
        -cred_names="NC1;NC2;NC3"
        -target_name="testhost.example.com"
        -target_type="host
```

test_privilege_delegation_setting

Tests privilege delegation settings on a specified host.

Format

Standard Mode

```
emcli test_privilege_delegation_setting
        -host_name="Host Name"
        -cred_name="Cred Name"
        [-cred_owner="Cred Owner"]
```

Interactive or Script Mode

```
test_privilege_delegation_setting(
        host_name="Host Name"
        ,cred_name="Cred Name"
        [,cred_owner="Cred Owner"]
        )

[ ] indicates that the parameter is optional
```

Parameters

- **host_name**

  Target name of the host.

- **cred_name**

  Credential name.

- **cred_owner**

  Credential owner

Exit Codes

0 on success. A non-zero value means verb processing was not successful.

Examples

Example 1

This example tests the privilege delegation settings for a host named *my_host* and credentials named *my_cred*.

```
emcli test_privilege_delegation_setting
              -host_name="my_host"
              -cred_name="my_cred"
```

Example 2

This example tests the privilege delegation settings for a host named "my_host" and credential named "my_cred" owned by "owner1."

```
emcli test_privilege_delegation_setting
              -host_name="host"
              -cred_name="cred"
              -cred_owner="owner1"
```

trace

Enables or disables tracing for OMS.

Format

```
emcli trace
        -enable="true|false"
        -user="username"
```

Parameters

- enable

  Specify true to enable and false to disable.

- user

  Name of the user.

Example

This example enables tracing for user sysman.

```
emcli trace -enable=true -user=sysman
```

trace_set_property

Sets the property name and corresponding value for the trace facility. These values are not persistent.

Format

```
emcli trace_set_property
     -name=<property name>
     -value=<property value>
```

Parameters

- name

  Property name.

- value

  Property value.

Example

The following example enables tracing for the user.

```
emcli trace_set_property -name=trace.backgroundthreads -value=true
```

udmmig_list_matches

Lists all the metric extensions that match the UDMs in a given migration session.

Format

```
emcli udmmig_list_matches
        -session_id=<sessionId>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

udmmig_request_udmdelete

Deletes the UDMs that have been replaced by Metric Extensions.

Format

```
emcli udmmig_request_udmdelete
        -session_id=<sessionId>
        -input_file=metric_tasks:<complete_path_to_file>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

- **input_file**

  Specify a file name that contains a target, UDM, one per line in the following format:

  ```
  <targetType>,<targetName>,<collection name>
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Example

This example attempts to delete the UDM from all relevant targets. This step is indicative of the end of the migration process. The file input_tasks lists the locations where the UDM is present.

```
emcli udmmig_request_udmdelete -session_id=<sessionId> -input_file=metric_
tasks:input_tasks
```

udmmig_retry_deploys

Retries the deployment of metric extensions to a target.

Format

```
emcli udmmig_retry_deploys
        -session_id=<sessionId>
        -input_file=metric_tasks:<complete path to file>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

- **input_file**

  Specify a file name that contains a target, UDM, one per line in the following format:

  ```
  <targetType>,<targetName>,<collection name>
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Example

This example attempts to deploy the metric extension to all targets where the UDM was present. The file input_tasks lists these locations.

```
emcli udmmig_retry_deploys -session_id=<sessionId> -input_file=metric_tasks:input_
tasks
```

udmmig_session_details

Provides details of the specified migration session, including the targets, templates, UDMs, and metric extensions involved.

Format

```
emcli udmmig_session_details
        -session_id=<sessionId>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

udmmig_submit_metricpicks

Supply the metric picks to use to replace UDMs per target in a session.

Format

```
emcli udmmig_submit_metricpicks
        -session_id=<sessionId>
        -input_file=metric_picks:<complete_path_to_file>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

- **input_file**

  Specify a file name that contains a target, UDM, metric pick, one per line in the following format:

  ```
  <targetType>,<targetName>,<collection name>,[N/E],<metric>,<column>
  ```

  Use N if a new metric should be created, or E if an existing metric is referenced.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Example

This example provides the mapping from UDM to the metric extension through the file input_picks.

```
emcli udmmig_submit_metricpicks -session_id=<sessionId> -input_file=metric_
picks:input_picks
```

udmmig_summary

Displays all the active migration sessions in the system.

Format

```
emcli udmmig_summary
      [-showAll]

[ ] indicates that the parameter is optional
```

Parameters

- **showAll**

  Prints out all sessions including those that are complete. By default, only in-progress sessions are listed.

udmmig_update_incrules

Updates incident rules that reference UDMs with a reference to replacing a metric extension.

Format

```
emcli udmmig_update_incrules
        -session_id=<sessionId>
        -input_file=udm_inc_rules:<complete_path_to_file>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

- **input_file**

  Specify a file name that contains a rule, UDM, metric, one per line in the following format:

  ```
  <ruleset id>,<rule id>,<udm name>,<metric name>
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

unassign_charge_plan

Unassigns the charge plan associated with the specified entity.

Format

```
unassign_charge_plan
      -entity_name="eName"
      -entity_type="eType"
      -[entity_guid="entity_guid"]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_name**

  Name of the entity for which the charge plan is to be unassigned.

- **entity_type**

  Type of entity for which the charge plan is to be unassigned.

- **entity_guid**

  guid of the entity to be added to Chargeback.

  When more than one entity is active in Chargeback with the given entity name and entity type, the command lists all such entities with additional details such as creation date, parent entity name, entity guid, and so forth to choose the correct entity. Select the correct entity from the given list and execute the command again with entity guid as the parameter instead of entity name and entity type.

Example

This example unassigns charge plan associated to "db1", an oracle_database entity.

```
emcli unassign_charge_plan -entity_name="db1" -entity_type="oracle_database"
```

See Also

add_chargeback_entity
assign_charge_plan
assign_cost_center
list_chargeback_entities
list_chargeback_entity_types
list_charge_plans
list_cost_centers
remove_chargeback_entity
unassign_cost_center

unassign_cost_center

Unassigns the cost center from the given entity.

Format

```
unassign_cost_center
     -entity_name="eName"
     -entity_type="eType"
     -[entity_guid="entity guid" ]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_name**

  Name of the entity for which the cost center is to be unassigned.

- **entity_type**

  Type of entity for which the cost center is to be unassigned.

- **entity_guid**

  guid of the entity in Chargeback.

  When more than one entity is active in Chargeback with the given entity name and entity type, the command lists all such entities with additional details such as creation date, parent entity name, entity guid, and so forth to choose the correct entity. Select the correct entity from the given list and execute the command again with entity guid as the parameter instead of entity name and entity type.

Example

This example unassigns the cost center associated to "db1", an Oracle database entity.

```
emcli unassign_cost_center -entity_name="db1" -entity_type="oracle_database"
```

See Also

add_chargeback_entity
assign_charge_plan
assign_cost_center
list_chargeback_entities
list_chargeback_entity_types
list_charge_plans
list_cost_centers
remove_chargeback_entity
unassign_charge_plan

undeploy_diagchecks

Undeploys diagcheck scripts for targets.

Format

```
emcli undeploy_diagchecks
      {-target_name=<target_name_to_be_updated>
       -target_type=<target_type_to_be_updated> }
      | {-input_file=targetList:<complete_path_to_file>};
```

Parameters

- **target_name**

  Name of the target to be updated.

- **target_type**

  Type of target to be updated.

- **input_file**

  Specify a file name that contains a list of targets, one per line in the following format:

  ```
  <targetType>:<targetName>
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

undeploy_plugin_from_agent

Undeploys an Enterprise Manager plug-in from the Management Agents. Undeploying a plug-in from a Management Agent removes all targets of any type belonging to this plug-in from Enterprise Manager.

Defaults to the version currently deployed on the given Management Agent.

Format

```
emcli undeploy_plugin_from_agent
        -plugin="pluginId[:pluginVersion]"
        -agent_names="agent1;agent2"
```

Parameters

- **plugin**

  Plug-in ID and version to be undeployed. Version is optional, and it defaults to the latest version deployed on the management server.

- **agent_names**

  Management Agents (host:port) from which the plug-in is to be undeployed.

Examples

Example 1

This example undeploys the oracle.sysman.db2 plug-in of version 11.2.0.1.0 from Management Agents myhost1.example.com:1159 and myhost2.example.com:1159.

```
undeploy_plugin_from_agent -plugin=oracle.sysman.db2:11.2.0.1.0
-agent_names="myhost1.example.com:1159;myhost2.example.com:1159"
```

Example 2

This example undeploys the oracle.sysman.db2 plug-in of the latest version from the Agent myhost1.example.com:1159.

```
undeploy_plugin_from_agent -plugin=oracle.sysman.db2
-agent_names="myhost1.example.com:1159"
```

undeploy_plugin_from_server

Undeploys a plug-in from the Oracle Management Server.

> **Note:** You need to undeploy the plug-in from all Management Agents before you can undeploy it from the management server.

Format

```
emcli undeploy_plugin_from_server
        -plugin="plug-inId"[:"pluginVersion"]
        [-sys_password="sys_password"]

[ ] indicates that the parameter is optional
```

Parameters

- **plugin**

  This is of the form -plugin=<oracle.sysman.db:12.1.0.1.0> where the plug-in id (like oracle.sysman.db) is a required parameter and the version is optional.

  You do not need to provide a version in the -plugin="plugin_id" field, because at any given time, only one version of the plug-in can be deployed on the management server. Therefore, the version is implicit. Contrast this with providing a version during deployment, because you could have downloaded more than one version.

- **sys_password**

  The repository sys user password. If not provided at the console, it will be prompted for.

Examples

Example 1

This example undeploys the "oracle.sysman.db2" plug-in from the Oracle Management Server.

```
undeploy_plugin_from_server -plugin="oracle.sysman.db2" -sys_password=knl_test7
```

Example 2

This example prompts you for sys_password.

```
emcli undeploy_plugin_from_server -plugin="oracle.sysman.db2"
```

unregister_bipublisher

Unregisters a previously set up relationship between Enterprise Manager and a previously set up relationship (using setup_bipublisher). You can also use this verb to determine the status of the relationship between Enterprise Manager and BI Publisher if you do not specify the -force option.

> **Note:** This verb requires Enterprise Manager Super Administrator privileges.

Format

```
emcli unregister_bipublisher
    [-force]
```

Parameters

- **force**

  This option severs the relationship. The BI Publisher managed server is not stopped or uninstalled.

Examples

Example 1

```
emcli unregister_bipublisher

Error: The BI Publisher Web Application named
"https://somehost.somedomain.com:9704/xmlpserver" is registered. Use -force option
to overwrite this.
```

Example 2

```
emcli unregister_bipublisher -force

BI Publisher "https://somehost.somedomain.com:9704/xmlpserver" has been
unregistered for use with Enterprise Manager.
```

unsecure_agent

Unsecures a secured Management Agent. This verb requires operator privilege or full privilege on the Management Agent.

Format

```
emcli unsecure_agent
        -agent_name="agent_target_name"
        [-host_username ="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name ="credential_name"]
        [-credential_setname ="credential_setname_of_agent"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **host_username**

  User name of the OS user (on the host) who owns the Management Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Management Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Management Agent. Example: "HostCreds"

Examples

Example 1

```
emcli unsecure_agent -agent_name="agent.example.com:1234"
                          -host_username="test_user"
                          -host_pwd="test"
```

Example 2

```
emcli unsecure_agent -agent_name="agent.example.com:1234"
                          -credential_name="MyMachineCredential"
```

Example 3

```
emcli unsecure_agent -agent_name="agent.example.com:1234"
                          -credential_setname="HostCreds"
```

update_and_retry_step

Updates arguments of the failed step and retries it.

Format

```
emcli update_and_retry_step
      -stateguid=<state_guid>
      [-instance=<instance_guid>]
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
      [-args="command1:value1;command2:value2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **stateguid**

  State GUID.

- **instance**

  GUID of the instance.

- **exec**

  GUID of the execution.

- **name**

  Name of the execution.

- **owner**

  Owner of the execution.

- **args**

  Arguments of the step to be updated during retry. The format of the arguments are name-value pairs. Name and value are separated by a colon (:), and each pair is separated by a semicolon (;). The arguments take scalar data and list data. The format of list data should be like [a,b,c].

  For the full list of arguments that can be updated, see the get_retry_arguments verb.

Examples

```
emcli update_and_retry_step -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid=51F762417C4943DEE040578C4E087168 -args="command:ls"
```

update_audit_settings

Updates the current audit settings in the repository and restarts the OMS.

Format

```
emcli update_audit_settings
      -audit_switch="ENABLE|DISABLE"
      -operations_to_enable="name_of_operations_to_enable"
      -operations_to_disable="name_of_operations_to_disable"
      -externalization_switch="ENABLE|DISABLE"
      -directory="directory_name"
      -file_prefix="file_prefix"
      -file_size="file_size"
      -data_retention_period="data_retention_period"
```

Parameters

- **audit_switch**

  Audit switch to enable auditing across Enterprise Manager.

- **operations_to_enable**

  Enables auditing for specified operations. To enable all operations, specify ALL. This parameter is invalid if auditing is disabled.

- **operations_to_disable**

  Disables auditing for specified operations. To disable all operations, specify ALL. This parameter is invalid if auditing is disabled.

- **externalization_switch**

  Enable the audit data export service. The default value is DISABLE.

- **directory**

  Database directory that is configured with an OS directory where the export service archives the audit data files. This directory is required to externalize audit data in Enterprise Manager Cloud Control. The update_audit_settings verb assumes that this directory has already been created. The following example creates the database directory EMDIR from the directory `/tmp/em_audit_data` with read/write permissions for the SYSMAN user:

  ```
  SQL>create directory EMDIR as '/tmp/em_audit_data';
  Directory created.
  SQL>grant read,write on directory "EMDIR" to SYSMAN;
  Grant succeed.
  ```

- **file_prefix**

  File prefix to be used by the export service to create the file name where audit data is to be written. The default value is em_audit.

- **file_size**

  Maximum value of each file size. The default value for this is 5000000 bytes.

- **data_retention_period**

  Maximum period the Enterprise Manager repository stores audit data. The default value is 365 days.

Examples

Example 1

This example enables all operations except LOGIN and LOGOUT:

```
emcli update_audit_settings
          -audit_switch="ENABLE"
          -operations_to_enable="ALL"
          -operations_to_disable="LOGIN;LOGOUT"
```

Example 2

```
emcli update_audit_settings
    -externalization_switch="ENABLE"
    -directory="EM_DIR"
    -file_prefix="my_audit"
    -file_size="10000"
    -data_retention_period="60"
```

update_credential_set

Update privileges required to get/set global preferred credentials. You can update privileges for a single credential set, for all credential sets of a specific target type, or for the entire system (all target types).

Format

Standard Mode

```
emcli update_credential_set
        [-set_name="set_name"]
        [-target_type="ttype"]
        [-get_priv="get_priv"]
        [-update_priv="update_priv"]
        [-update_default_priv="update_default_priv"]
```

Interactive or Script Mode

```
update_credential_set(
        [set_name="set_name"]
        [,target_type="ttype"]
        [,get_priv="get_priv"]
        [,update_priv="update_priv"]
        [,update_default_priv="update_default_priv"]
        )
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **set_name**

  Credential set name for which privileges need to be updated.

- **target_type**

  Target type for the target/credential set.

- **get_priv**

  Name of the privilege required to *get* the system scoped credential set.

- **update_priv**

  Name of the privilege required to *set/clear* the system scoped credential set.

- **update_default_priv**

  Name of the privilege required to *set/clear* the default global scoped preferred credentials for the set.

Exit Codes

0 on success. A non-zero value indicates that verb processing was not successful.

Examples

This example changes the privilege *get_priv* to *VIEW_TARGET* across all target types and all credential set names.

```
emcli update_credential_set -get_priv=VIEW_TARGET
```

update_database_size

Lists all of the database sizes that have been created.

Format

```
emcli update_database_size
      -name="<Existing size name>"
      -description="<Size description>"
      [-attributes="cpu:<number of cores>;memory:<memory in GB>;processes:
         <max number of processes>;storage:<Total storage in GB allocated
         to database>;"]

[ ] indicates that the parameter is optional.
```

Parameters

- name

  The name of the existing database size.

- description

  Updates the description of the database size.

- attributes

  Defines the database size. Attributes must be separated by a semicolon(;). You can specify values for the following attributes:

  cpu: Total number of cpu cores.

  memory: Total maximum in GB.

  processes: Total number of processes that can simultaneously connect to the database.

  storage: Total storage that is allocated to the database (in GB)

Example

The following command updates the description and attributes of the database size with the name Small.

```
emcli update_database_size
      -name=Small
      -description="Small size database"
      -attributes="cpu:4;storage:50;memory:4;processes:remove"
```

update_db_password

Updates the target database password change in the Enterprise Manager Credential sub-system and can change the password on the target database as well. This verb also propagates the collection or monitoring credentials to Enterprise Manager Management Agents.

Command-Line Format

```
emcli update_db_password
       -target_name="tname"
       -user_name="user_name"
       [-target_type="ttype"]
       [-change_all_references="yes/no"]
       [-change_at_target="yes/no"]
       [-input_file="tag1:file_path1;tag2:file_path2;..."]

[ ] indicates that the parameter is optional
```

Scripting and Interactive Format

```
update_db_password
       (target_name="tname"
       ,user_name="user_name"
       [,target_type="ttype"]
       [,change_all_references="yes/no"]
       [,change_at_target="yes/no"]
       [,input_file="tag1:file_path1;tag2:file_path2;..."])

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Name of the target.

- **user_name**

  Name of the database user.

- **target_type**

  Type of target. The possible values for target type in this verb are -oracle_database and -rac_database. The default value for this parameter is oracle_database. For the rac_database type, the password should be changed at the database and not at the individual instance level.

- **change_all_references**

  Specify if the password must be changed for all references in Enterprise Manager. Possible values are:

  – **yes** — Update all password references in Enterprise Manager for a DBSNMP user who has an old password that matches the new password.

  – **no** — Update the password for the currently logged in user.

  The default value of this option is Yes.

- **change_at_target**

Specify whether the password must also be changed on the target. This is not supported for a SYS user.

- **yes** — Change the password on the target database.
- **no** — Update the password only on Enterprise Manager.

The default value of this option is No.

- **input_file**

  Path of the file that has old and new passwords. Use this option to hide passwords displayed on the command line. You must accompany each path with a tag referenced in the password options.

  When you execute this verb with the input_file option, you are prompted to enter the following values in non-echo mode:

  -old_password
  -new_password
  -retype_new_password

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

Example 1 - Command-Line

```
emcli update_db_password
      -target_name=myDB
      -user_name=Admin1
```

Example 2 - Scripting and Interactive

```
update_db_password
       (target_name="myDB",
        user_name="Admin1")
```

Example 3 - Command-Line

```
emcli update_db_password
      -target_name=myDB
      -user_name=Admin1
      -change_at_target=yes
```

Example 4 - Scripting and Interactive

```
update_db_password
      (target_name="myDB",
       user_name="Admin1",
       change_at_target="yes")
```

update_diagchecks

Updates diagnostic check scripts for targets.

Format

```
emcli update_diagchecks
      -target_name=<target_name_to_be_updated>
      -target_type=<target_type_to_be_updated>
      [-input_file=targetList:<complete_path_to_file>]
```

Parameters

- **target_name**

  Name of the target to be updated.

- **target_type**

  Type of the target to be updated.

- **input_file**

  Specify a file name that contains a list of targets, one per line in the following format:

  ```
  <targetType>:<targetName>
  ```

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

update_host_password

Updates the changed host password in the credential sub-system. For collection or monitoring credentials, the password change is optionalso propagated to the Enterprise Manager Management Agent.

Format

```
emcli update_host_password
      -target_name="tname"
      -user_name="user_name"
      [-change_all_references="yes/no"]
      [-input_file="tag1:file_path1;tag2:file_path2;..."]

[ ] indicates that the parameter is optional
```

Note: When you execute this verb, you are prompted to enter the following values in non-echo mode:

-old_password
-new_password
-retype_new_password

Parameters

- **target_name**

  Name of the target.

- **user_name**

  Name of the database user.

- **change_all_references**

  Specifies if the password must be changed for all references in Enterprise Manager for the given user.

  Possible values are:

  – Yes — Updates all references in Enterprise Manager for this password.

  – No — Updates the password for the current logged-in user. This is the default.

- **input_file**

  File path that has old and new passwords. This hides passwords. You must accompany each path with a tag referenced in the password.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

Example 1

This example asks the user to enter the values of the old and new passwords, then retype the new password to update the new password in Enterprise Manager for this target reference.

```
emcli update_host_password
```

```
      -target_name=myHost
      -user_name=Admin1
```

Example 2

This example asks the user to enter the values of the old and new passwords, then retype the new password to update the new password in Enterprise Manager for all users' credentials referenced with the myHost target name and Admin1 user name.

```
emcli update_host_password
      -target_name=myHost
      -user_name=Admin1
      -change_all_references=yes
```

update_monitoring_creds_from_agent

Finds all of the targets on the Management Agent, retrieves the monitoring credentials, and updates them in the Management Repository. In 11g Release 1 (11.1.0.0), the monitoring credentials for some targets were stored only on the Management Agent.

> **Note:** Although -emd_list and -update_all are shown syntactically as optional, you must provide either one or the other.

Format

```
emcli update_monitoring_creds_from_agent
       [-emd_list=<emd_list>]
       [-update_all]

[ ] indicates that the parameter is optional
```

Parameters

- **emd_list**

  List of EMD URLs. You must provide either this parameter or the update_all option.

- **update_all**

  Update in the repository for all targets that have monitoring credentials on the Agents but not in the repository. You must provide either this parameter or the emd_list option.

Exit Codes

0 if successful. A non-zero value means that verb processing was not successful.

Examples

Example 1

This example finds all the targets monitored by host1.example.com:1832 and host2.example.com:1832 that have monitoring credentials on the Agent but not in the management repository, and updates the monitoring credentials in the management repository.

```
emcli update_monitoring_creds_from_agent
       -emd_list="host1.example.com:1832;host2.example.com:1832"
```

Example 2

This example finds all the targets that have monitoring credentials on the Management Agents but not in the management repository, and updates the monitoring credentials in the repository.

```
emcli update_monitoring_creds_from_agent
       -update_all
```

update_operation_plan

Updates the SiteGuard operation plan.

Format

```
emcli update_operation_plan
        [-name=<plan_name>]
        [-step_number=<step_number>]
        [-target_host=<host_name>]
        [-error_mode=<error_mode>]
        [-enabled=<true|false>]
        [-execution_mode=<Serial|Parallel>]
        [-move=<Up|Down>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the operation plan.

- **step_number**

  Number of the step that should be updated.

- **target_host**

  Target host name. Specifying this updates all of the steps involving this target host.

See Also

emcli create_operation_plan

emcli get_operation_plan_details

Examples

```
emcli update_operation_plan -name="austin-switchover"
             -step_number="1"
             -error_mode="Continue"
             -enabled="true"
             -execution_mode="Serial"

emcli update_operation_plan -name="austin-switchover"
             -step_number="5"
             -move="Up"

emcli update_operation_plan -name="austin-switchover"
             -target_host="myhost.domain.com"
             -error_mode="Continue"
             -enabled="true"
```

update_dbaas_quota

Updates the database quota for an SSA user role.

Format

```
emcli update_dbaas_quota
        -role_name="<SSA_User_Role_name>"
        -databases="<Number_of_Database_Requests>"
        -schema_services="<Number_of_Schema_Service_Requests>"
        -pluggable_databases="<Number_of_Pluggable_Database_Service_Requests>"
        -memory="<Memory(GB)>"
        -storage="<Storage(GB)>"
```

Parameters

- **role_name**

  Name of an SSA user role for which the quota is to be updated.

- **databases**

  Number of database service requests allowed.

- **schema_services**

  Number of schema service requests allowed.

- **pluggable_databases**

  Number of pluggable database service requests allowed.

- **memory**

  Amount of memory (GB) usage allowed.

- **storage**

  Amount of storage (GB) usage allowed.

Examples

```
emcli update_dbaas_quota
-role_name="My Role"
-databases="10"
-schema_services="10"
-pluggable_databases="10"
-memory="99"
-storage="99"
```

displays the following output:

```
Quota for "My Role" updated successfully.
```

update_dbaas_request_settings

Updates the database request settings.

Format

```
emcli update_dbaas_request_settings
        -future_reservation_length="<Future_Request_Scheduling_Period>"
        -maximum_archive_duration="<Request_Purging_Duration>"
        -default_retirement_period="<Default_Retention_Duration>"
```

Parameters

- **future_reservation_length**

  Amount of time in advance a self-service user can schedule a request.
  Example: "2 Months" for 2 Months, "10 Weeks" for 10 Weeks, and "No Reservation" for no restriction

- **maximum_archive_duration**

  Amount of time after which the ''Completed'' Self Service Create Requests will be purged from the Repository.
  Example: "2 Months" for 2 Months, "10 Weeks" for 10 Weeks, and "No Reservation" for no restriction

- **default_retirement_period**

  The maximum amount of time for which a self-service user can retain a service instance.
  Example: "2 Months" for 2 Months, "10 Weeks" for 10 Weeks, "No Reservation" for no restriction

Examples

```
emcli update_dbaas_request_settings
-future_reservation_length="2 Months"
-maximum_archive_duration="10 Weeks"
-default_retirement_period="No Reservation"
```

displays the following output:

```
Request settings updated successfully.
```

update_paas_zone

Updates a PaaS Infrastructure Zone definition.

Format

```
emcli update_paas_zone
        -name="<Name_of_PaaS_Zone>"
        [-description="<Description_of_PaaS_Zone>"]
        [-credential="<Global_Named_Credential>"]
        [-add_hosts="<Host1,Host2,Host3...>"]
        [-remove_hosts="<Host4,Host5...>"]
        [-add_ovm_zones="<OVMZone1,OVMZone2,OVMZone3...>"]
        [-remove_ovm_zones="<OVMZone4,OVMZone5...>"]
        [-add_roles="<SsaRole1,SsaRole2,..>"]
        [-remove_roles="<SsaRole3,SsaRole4,..>"]
        [-cpu_utilization="<Value_between_1_and_100>"]
        [-memory_utilization="<Value_between_1_and_100>"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the existing PaaS Infrastructure Zone.

- **description**

  Updated description of the PaaS Infrastructure Zone.

- **credential**

  Global named credentials to be updated. These will be used for provisioning in this PaaS Infrastructure Zone. The credentials should be the same for all hosts. A cloud administrator can only use the named credentials that they own.

- **add_hosts**

  Comma-separated list of the host targets to be added as members of this Paas Infrastructure Zone. The hosts must not be members of other PaaS Zones.

- **remove_hosts**

  Comma-separated list of the host targets to be removed as members from this Paas Infrastructure Zone. The hosts must not be associated with any Software Pool member.

- **add_ovm_hosts**

  Comma-separated list of the OVMZone targets to be added as members of this Paas Infrastructure Zone. The OVMZones to be added must not be already added to other existing PaaS Zones.

- **remove_ovm_hosts**

  Comma-separated list of the OVMZone targets to be removed as members from this Paas Infrastructure Zone.

- **add_roles**

  Comma-separated list of SSA roles to be added to the list of roles that can access this PaaS Infrastructure Zone. A PaaS infrastructure zone can be made available to a restricted set of users through the use of roles.

- **remove_roles**

  Comma-separated list of SSA roles to be removed from the list of roles that can access this PaaS Infrastructure Zone.

- **cpu_utilization**

  Placement policy constraints allow the cloud administrator to set maximum resource ceilings for any host in the PaaS Infrastructure Zone. This provides protection for the members of the PaaS Infrastructure Zone in terms of resource consumption. For example, a production PaaS Infrastructure Zone might limit CPU utilization to 80%, whereas a development PaaS Infrastructure Zone might allow up to 95%. The service instance will be provisioned on the first host that satisfies the placement constraints.

  The value entered must be between 1 and 100. If not provided, the default value is taken to be 80%.

- **memory_utilization**

  A Placement Policy constraint for memory used by the PaaS Infrastructure Zone.

  The value entered must be between 1 and 100. If not provided, the default value is taken to be 80 percent.

Examples

Example 1

```
emcli update_paas_zone
-name="My PaaS Zone"
-add_hosts="host3.mycompany.com"

PaaS Infrastructure Zone "My PaaS Zone" updated successfully.
```

Example 2

```
emcli update_paas_zone
-name="My PaaS Zone"
-cpu_utilization="65"

PaaS Infrastructure Zone "My PaaS Zone" updated successfully.
```

update_password

Updates passwords or other credentials for a given target.

Format

```
emcli update_password
        -target_type="ttype"
        -target_name="tname"
        -credential_type="cred_type"
        -key_column="column_name:column_value"
        -non_key_column="col:oldvalue:newvalue;..."
        [-input_file="tag1:file_path1;tag2:file_path2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target.

- target_name

  Name of the target.

- **credential_type**

  Credential type to use. The type must be a base type, not a derived type. A derived type contains the XML tag <CredentialTypeRef> within its definition.

- **key_column**

  Name and value of the key column for the credential type. Usually, the key column represents the user name. To get the key column for a target type, you can execute following EM CLI verbs:

  emcli show_credential_type_info — Displays key columns for all target types.

  emcli show_credential_type_info -target_type=<target_type> — Displays key columns for a specific target type.

- **non_key_column**

  Name, old value, and new value of the non-key column(s) to modify. Usually, this is the name of the password column. Alternatively, a tag from the -input_file argument can be used so that the credential values are not seen on the command line. You can specify this parameter more than once.

- **input_file**

  Path of the file that has non_key_column argument(s). This option is used to hide passwords. You must accompany each path with a tag that is referenced in the non_key_column argument. You can specify this option more than once.

  You can obtain the list of columns and the credential types they belong to by using the emcli show_credential_type_info command.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

Example 1

```
emcli update_password
          -target_type=oracle_database
          -target_name=myDB
          -credential_type=DBCreds
          -key_column="DBUserName:joe"
          -non_key_column="DBPassword:oldPass:newPass"
          -non_key_column="DBRole:normal:sysdba"
```

Example 2

In This example, FILE1 is a tag used to refer to the contents of passwordFile. The contents of the password file is:

```
DBPassword:oldPass:newPass;DBRole:normal:sysdba
```

Note that this example has the same effect as Example 1.

```
emcli update_password
          -target_type=oracle_database
          -target_name=myDB
          -credential_type=DBCreds
          -key_column="DBUserName:joe"
          -non_key_column="FILE1"
          -input_file="FILE1:passwordFile"
```

upload_jeeappcomp_file

Uploads one file to a Java EE Application component in the software library.

Format

```
emcli upload_jeeappcomp_file
        -entity_rev_id="entity_rev_id"
        -host="hostname"
        -filetype="filetype"
        -file="<absolute file path>[;<new file name>]"
        [-credential_set_name="setname"] | [-credential_name="name" -credential_
owner="owner"]
        [-upload_storage="<storage location name>;<storage type>"]
        [-use_latest_revision]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_rev_id**

  Identifier for the entity revision. You can view the entity ID by logging in to the Cloud Control console. The Software Library Home page exposes the identifier for folders and entities as a custom column called Internal ID. By default, this is hidden.

- **host**

  Target name of the host where the files are available.

- **filetype**

  The file type of the file specified with '-file'. Valid values include: "archive", "plan", "pre_deployment_script", "post_deployment_script", "target_execution_script", "additional_file", "zip".

- **file**

  Absolute path of the file to be uploaded. File name stored in the Software Library is overwritten with the name of the file being uploaded. Optionally, you can specify a different file name, separated by ";".

- **credential_set_name**

  The name of the preferred credential stored inside the Management Repository for the host target. It can be one of the following:

  HostCredsNormal - default unprivileged credential set.

  HostCredsPriv - privileged credential set.

- **credential_name**

  Name of a named credential stored in the Management Repository. This option must be specified along with the -credential_owner option.

- **credential_owner**

  Owner of a named credential stored in the Management Repository. This option must be specified along with the -credential_name option.

- **upload_storage**

Destination storage location and type of storage is passed as a parmeter for the upload, separated by ";". If no value is passsed, then the storage type and storage location are defaulted to the first upload location configured for Software Library. The storage type can be one of the following:

OmsShared (OMS Shared File System)

OmsAgent (OMS Agent File System)

The storage location specified must be in "active" status.

- **use_latest_revision**

  Flag for indicating that the upload should happen to the latest revision of the entity instead of the revision identified by entity_rev_id.

Examples

Example 1

Uploads the file '/u01/downloads/file1.jar' to the entity revision identified by entity_rev_id. The file present on the host 'example.com' should be accessible using the preferred credential set for the "HostCredsNormal" credential set, for the user logged in to EMCLI.

The host must be a managed host target in Enterprise Manager and the agent on this host must be up and running.

```
emcli upload_jeeappcomp_file
-entity_rev_id="oracle:defaultService:em:provisioning:1:cmp:COMP_Component:SUB_
JavaEEApplication:B1B1880C6A8C62AAE040548C42832D14:0.1"
-file="/u01/downloads/file1.jar"
-filetype="archive"
-host="example.com"
-credential_set_name="HostCredsNormal"
```

Example 2

Uploads the file newfile.xml to the entity revision identified by entity_rev_id. The file present on the host 'example.com' should be accessible using the credential named'"MyExampleCreds" owned by "EXAMPLE_USER". The entity revision specified should contain a file of filetype 'archive' before uploading any other filetype. File '/u01/downloads/file2.xml', after upload, will be associated with the entity revision as 'newfile.xml'. A new revision will be created from the latest revision of the entity.

```
emcli upload_jeeappcomp_file
-entity_rev_id="oracle:defaultService:em:provisioning:1:cmp:COMP_Component:SUB_
JavaEEApplication:B1B1880C6A8C62AAE040548C42832D14:0.1"
-file="/u01/downloads/file2.xml;newfile.xml"
-filetype="plan"
-host="example.com"
-credential_name="MyExampleCreds"
-credential_owner="EXAMPLE_USER"
-use_latest_revision
```

update_pool

Updates the details for a Software Pool.

Format

```
emcli update_pool
        -name="<Software_Pool_name>"
        -target_type="<Target_type_of_Software_Pool>"
        [-description="<Description_of_Software_Pool>"]
        [-add_members="<Member1, Member2...>"]
        [-remove_members="<Member4, Member5...>"]
        [-placement_constraints="<constraint1=value1,constraint2=value2...>"]
        [-properties="<property1=value1, property2=value2>"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of an existing Software Pool.

- **target_type**

  Target type of the Software Pool. Example: "mwaas_zone" for Middleware Pool, "oracle_cloud_zone" for Database Pool, and "schaas_pool" for Schema Pool.

- **description**

  Description of the Software Pool.

- **add_members**

  Comma-separated list of targets to be added as members of the Software Pool. The targets to be added must satisfy the membership constraints of the Software Pool.

- **remove_members**

  Member targets to be removed from the Software Pool.

- **placement_constraints**

  Comma-separated key-value pairs of the placement constraints that allow the self-service administrator to set maximum ceilings for resource utilization. This provides protection for the members of the Software Pool in terms of resource consumption.

- **properties**

  Comma-separated key value pairs for properties that need to be updated based on the pool target type.

Examples

```
emcli update_pool
-name="My Pool"
-target_type="mwaas_zone"
-add_members="MyMember2"
```

displays the following output:

```
Software Pool "My Pool" updated successfully.
```

update_procedure_input

Updates the configuration of a deployment procedure.

Format

```
emcli upate_procedure_input
        -name="name_of_procedure_configuration"
        [-input_file="file_path\file_name"]
        [-grants="users_and_access_levels"]
        [-schedule=
            start_time:yyyy/MM/dd HH:mm;
            tz:<java_timezone_ID>;
            grace_period:xxx;
        ]
        [-notification="procedure status"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the configuration for the procedure.

- **input_file**

  Input property file for the deployment procedure. The file_path should point to a file containing the data property file.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **grants**

  Specifies users and their corresponding access levels as a string of user:privilege pairs, each separated by a semi-colon (;). The user is an Enterprise Manager user name, and the privilege is either VIEW_JOB or FULL_JOB.

  See the example below.

- **schedule**

  Schedule for the deployment procedure. If not specified, the procedure is executed immediately.

  - **start_time** — When the procedure should start.
  - **tz** — Optional timezone ID.
  - **grace_period** — Optional grace period in minutes.

- **notification**

  Status of the procedure.

Example

```
emcli update_procedure_input
    -name=configProcedure
    -input_file=/home/data.properties -grants="user1:VIEW_JOB;user2:FULL_JOB"
    -schedule="start_time:2011/8/21 21:23;tz:America/New_York;grace_period:60"
    -notification="scheduled, action required, running"
```

update_service_template

Updates a Service Template.

Format

```
emcli update_service_template
      -name="<Service_Template_name>"
      -service_family="<Service_family_name>"
      -pool_target_type="<PoolTargetType>"
      [-add_software_pools="<SwPool1,SwPool2,SwPool3,...>"]
      [-remove_software_pools="RemovePool1,RemovePool2,RemovePool3,...>"]
      [-add_roles="<SsaRole1,SsaRole2,..>"]
      [-remove_roles="<RemoveSsaRole1,RemoveSsaRole2,..>"]
      [-description="<Updated_Description_of_Service_Template>"]
      [-input_file="data:<Name_of_Service_executable_MetaData_File>"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the Service Template.

- **service_family**

  Service family to which the Service Template belongs. Example: DBAAS for Database, and MWAAS for Middleware.

- **pool_target_type**

  Target type of Software Pools to be associated with the Service Template.

- **add_software_pools**

  Comma-separated list of the Software Pools to be associated with this Service Template.

- **remove_software_pools**

  Comma-separated list of the Software Pools to be removed from this Service Template.

- **add_roles**

  Comma-separated list of SSA roles to be added to this Service Template. The SSA roles must already be created before attempting to add them to the Service Template.

- **remove_roles**

  Comma-separated list of SSA roles to be removed from this Service Template.

- **description**

  Description of the Service Template.

- **input_file**

  File containing configuration and profile data that will be required for updating values of procedure configuration variables. Format the data in JSON format.

  For example `input_file='data:executable.json'`

Examples

```
emcli update_service_template
-name="Middleware service template August"
-service_family="MWAAS"
-add_roles="SSA_USER_ROLE_1"
-remove_roles="SSA_USER_ROLE_2"
-add_software_pools="mwPool3,mwPool4"
-description="Updated description. Large instance size Service Template."
-input_file="data:executable.json"
```

displays the following output:.

```
Service Template "Middleware service template August" updated successfully.
```

update_siebel

Updates the Siebel enterprise.

Format

```
emcli update_siebel
      -enterprise=<Siebel enterprise>
      [-server=<Siebel server>]
      [-updateAutoStartModeComponentsOnly]
      [-review_only]
      [-out_file='<fully qualified path of output_file>']
      [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **enterprise**

  Fully-qualified name of the Siebel enterprise in Enterprise Manager. For example, to update a Siebel enterprise '<Enterprise>' run the command `update_siebel -enterprise=<Enterprise>`.

- **server**

  Fully-qualified name of the Siebel server in Enterprise Manager.

- **updateAutoStartModeComponentsOnly**

  Indicates that the updateNow operation is performed for only the components with 'auto' mode on.

- **review_only**

  Indicates that the `updateNow` operation only displays the targets to be updated without actually saving them in the Enterprise Manager repository.

- **out_file**

  Fully-qualified path of `output_file`. The output of the command is redirected to this file.

- **debug**

  Executes the command in verbose mode and generates additional debug log messages in the output.

Example

This example updates the specified Siebel enterprise from Cloud Control, and the output of the command is redirected to file `update_output.txt`.

```
emcli update_siebel -enterprise=<Siebel enterprise> -out_file='c:\emcli\update_
output.txt' -debug
```

update_siteguard_configuration

Updates the Site Guard configuration to add additional standby systems. One primary system can be associated with one or more standby systems.

> **Note:** If you update the site configuration, you must also update the operation plan, as described in update_monitoring_creds_from_agent.

Format

```
emcli update_siteguard_configuration
      [-primary_system_name=<primary_system_name>]
      [-standby_system_name=<standby_system_name>]
```

Parameters

- **primary_system**

  Name of the primary system.

- **standby_system**

  Name of the standby system. You can specify this parameter more than once.

See Also

create_siteguard_configuration

delete_siteguard_configuration

Examples

```
emcli update_siteguard_configuartion
      -primary_system_name="BISystem1"
      -standby_system_name="BISystem2"
```

update_siteguard_credential_association

Updates the credential association.

Format

```
emcli update_siteguard_credential_association
      [-system_name=<system_name>]
      [-target_name=<target_name>]
      [-credential_type=<credential_type>]
      [-credential_name=<credential_name>]
      [-use_preferred_credential=true|false]
      [-credential_owner=<credential_owner>]
```

Parameters

- **system_name**

  Name of the system.

- **target_name**

  Optional name of the target.

- **credential_type**

  Type of credential, which can be HostNormal, HostPrivileged, WLSAdmin, or DatabaseSysdba.

- **credential_name**

  Name of the credential.

- **use_preferred_credential**

  Use a preferred credential instead of the named credential. You need to specify credential_name if this option is false.

- **credential_owner**

  Owner of the credential.

See Also

delete_siteguard_credential_association

create_siteguard_credential_association

Examples

Example 1

```
emcli update_siteguard_credential_association
          -system_name="austin-system"
          -credential_type="HostNormal"
          -credential_name="HOST-SGCRED"
          -credential_owner="sysman"
```

Example 2

```
emcli update_siteguard_credential_association
          -system_name="utah-system"
```

```
          -credential_type="HostPrivileged"
          -use_preferred_credential="true"
          -credential_owner="sysman"
```

Example 3

```
emcli update_siteguard_credential_association
          -system_name="austin-system"
          -target_name="austin-database-instance"
          -credential_type="DatabaseSysdba"
          -credential_name="HOST-DBCRED"
          -credential_owner="sysman"
```

update_siteguard_lag

Updates the limit for Apply lag and Transport lag for all databases or selected databases of the system.

Format

```
emcli update_siteguard_lag
        [-system_name="name_of_the_system"]
        [-target_name="name_of_the_target_database"]
        [-property_name=lag_type]
        [-value="lag_limit_in_seconds"]

[ ] indicates that the parameter is optional
```

Parameters

- **system_name**

  Name of the system whose lag limits you want to update.

- **target_name**

  Name of the database whose lag limits you want to update.

- **property_name**

  Name of the lag property. Valid values for this parameter are `ApplyLag` and `TransportLag`.

- **value**

  Time limit of the lag. Specify the values of this parameter in seconds.

Examples

Example 1

This example updates the Apply lag property with a lag limit of 1000 seconds on all of the databases configured on `austin-system`:

```
emcli update_siteguard_lag
        -system_name="austin-system"
        -property_name="ApplyLag"
        -value="1000"
```

Example 2

This example updates the Transport lag property with a lag limit of 2500 seconds on the `OID-db` database configured on `austin-system`:

```
emcli update_siteguard_lag
        -system_name="austin-system"
        -target_name="OID_db"
        -property_name="TransportLag"
        -value="2500"
```

update_siteguard_script

Updates the path and the all_hosts flag associated with any script.

Format

```
emcli update_siteguard_script
      -script_id=<script_ID>
      [-path=<script_path>]
      [-credential_type=<type_of_credential>]
      [-all_hosts=true|false]

[ ] indicates that the parameter is optional
```

Parameters

- **script_id**

  ID associated with the script.

- **path**

  Optional path to the script.

- **credential_type**

  Type of credential, which can be either HostNormal or HostPrivileged.

- **all_hosts**

  Enables the script to run on all the hosts in the system. For example: true or false.

See Also

create_siteguard_script

get_siteguard_scripts

Examples

```
emcli update_siteguard_script
      -script_id="10"
      -path="/tmp/newprescript"
      -all_hosts="true"

emcli update_siteguard_script -script_id="16"
          -path="/tmp/script"
          -credential_type="HostPrivileged"
```

update_swlib_entity

Modifies an entity in the software library. A new revision of the entity is created by default. Changing only the description or attribute values does not create a new revision, and such changes will be visible across all existing revisions of the entity.

Format

```
emcli update_swlib_entity
        -entity_rev_id="entity_rev_id"
        [-desc="entity_desc"]
        [-attr="<attr_name>:<attr_value>"]
        [-prop="<prop_name>:<prop_value>"]
        [-secret_prop="<secret_prop_name>:<secret_prop_value>"]
        [-note="note_text"]
        [-use_latest_revision]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_rev_id**

  Identifier of the entity revision. The software library home page exposes the identifier for folders and entities as a custom column (Internal ID) and is hidden by default.

- **desc**

  Description of the entity. The new description is visible to all existing revisions.

- **attr**

  An attribute and its value, separated by a colon (:). To specify values for multiple attributes, repeat this option. The new attribute value is visible to all existing revisions.

- **prop**

  Configuration property and its value, separated by a colon (:). To specify values for multiple attributes, repeat this option.

- **secret_prop**

  Configuration property and its secret value separated by a colon (:). It is recommended that the secret value not be specified on the command line. If omitted from the command line, the value is prompted for. To specify values for multiple properties, repeat this option.

- **note**

  Note on the entity. For multiple notes, repeat this option.

- **use_latest_revision**

  Indicates that the the latest revision of the entity should be updated instead of the revision identified by entity_rev_id.

Examples

This example modifies the entity revision identified by entity_rev_id. The entity revision identifier value can be found from the Software Library home page. The

software library home page exposes the identifier for folders and entities as a custom column, which is hidden by default.

A new description is specified. Values for the entity attributes (PRODUCT, PRODUCT_VERSION and VENDOR) are specified. The value for the DEFAULT_HOME configuration property is specified. A note on the entity is also specified.

A new revision is created for the modifications, but the specified entity revision (identified by entity_rev_id) remains unchanged. The identifier of the newly created entity is printed on the standard output.

```
entity_rev_id="oracle:defaultService:em:provisioning:1:cmp:COMP_Component:SUB_
Generic:B1B1880C6A8C62AAE040548C4D14:0.1"
       -entity_desc="myAcmeInstall description"
       -attr="PRODUCT:Acme"
       -attr="PRODUCT_VERSION:3.0"
       -attr="VENDOR:Acme Corp"
       -prop="DEFAULT_HOME:/u01/acme3/"
       -note="myAcmeInstall for test servers"
```

update_target_password

Updates the changed target password in the Enterprise Manager credential sub-system. For collection or monitoring credentials, the password change is also propagated to Enterprise Manager Management Agents.

Format

```
emcli update_target_password
        -target_type="ttype"
        -target_name="tname"
        -key_column="column_name:column_value"
        [-change_all_references="yes/no"]
        [-input_file="tag1:file_path1;tag2:file_path2;..."]

[ ] indicates that the parameter is optional
```

Note: When you execute this verb, you are prompted to enter the following values in non-echo mode:

-old_password
-new_password
-retype_new_password

Parameters

- **target_type**

  Type of target.

- **target_name**

  Name of the target.

- **key_column**

  Name and value of the key column for the credential type. The key column usually represents the user name.

  To obtain the key column for a target type, enter the following command:

  ```
  emcli show_credential_type_info -target_type=<target_type>"
  ```

  To obtain the key column for all target types, enter the following command:

  ```
  emcli show_credential_type_info
  ```

  To obtain the key column for a target type, enter the following command:

  ```
  emcli show_credential_type_info -target_type=<target_type>"
  ```

- **change_all_references**

  Specifies if the password must be changed for all references in Enterprise Manager for the given user.

  Possible values are:

  – Yes — Updates all references in Enterprise Manager for this password.

  – No — Updates the password for the current logged-in user. This is the default.

- **input_file**

File path that has old and new passwords. This **option** hides passwords. You must accompany each path with a tag referenced in the password **option**s. You can specify this **option** more than once.

For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

Examples

Example 1

This example asks the user to enter the values of the old and new passwords, then retype the new password to update the new password in Enterprise Manager for this target reference.

```
emcli update_target_password
      -target_type=host
      -target_name=myHost
      -key_column=HostUserName:Admin1
```

Example 2

This example asks the user to enter the values of the old and new passwords, then retype the new password to update the new password in Enterprise Manager for all users' credentials referenced with the mydb target name and Admin1 user name.

```
emcli update_target_password
      -target_type=oracle_database
      -target_name=mydb
      -key_column=DBUserName:Admin1
      -change_all_references=yes
```

update_ticket_status

Updates the ticket status and last modified time stamp in Enterprise Manager from the external ticketing system based on the ticket_guid and connector_guid.

Format

```
emcli update_ticket_status
    -ticket_guid="ticket guid"
    -connector_guid="connector guid"
    -status="Incident status"
    -last_updated_date="last modified date"
    -date_format=
```

Parameters

- **ticket_guid**

  Ticket ID for which the status is modified.

- **connector_guid**

  Ticketing Connector ID.

- **status**

  Modified status of an incident ticket.

- **last_updated_date**

  Specifies the last modified date of an incident ticket.

- **date_format**

  Specify a date format followed in the Ticketing System, as in "MM/dd/yyyy hh:mm:ss" if the date field in Incident management is "10/13/2009 5:38:24 AM".

Example

This example updates the ticket INC00000024 status as 'In Progress' in Enterprise Manager after the same ticket status was recently modified on the ticketing system.

```
emcli update_ticket_status
  -ticket_guid="INC21000024"
  -connector_guid="ccc1234"
  -status="2"
  -last_updated_date="05/28/2011 3:14:56PM"
  -date_format="MM/dd/yyyy hh:mm:ss"
```

upgrade_agents

Performs Agent upgrade prerequisites and submits the Agent upgrade job.

Format

```
emcli upgrade_agents
     -agents="full_agent_name"|-input_file="agents_file:location_of_output_file"
     [-validate_only]
     [-pre_script_loc]
     [-post_script_loc]
     [-pre_script_on_oms]
     [-post_script_on_oms]
     [-stage_location]
     [-job_name]
     [-override_credential]
     [-additional_parameters]

[ ] indicates that the parameter is optional
```

Parameters

> **Note:** Either the -agents or -input_file parameter is mandatory. If you provide both, the union of both are taken, prerequisites are performed on the Agents, and an Agent upgrade job is submitted.
>
> You can pass all of these parameters in a response file. Usage: -input_file="response_file:/scratch/response_file.txt" . A file name with the full path must be provided, and each parameter should be specified in each line. If a parameter/flag is passed both in the command line and in a response file, the command-line option is given precedence. A parameter should be specified as a name-value pair in the response file. For example: job_name=UPGRADE_AGT_121020

- **agents**

  Checks whether the specified Agents specified are upgradable, and submits an Agent upgrade job.

- **input_file**

  Checks whether the Agents specified in file are upgradable, and submits an Agent upgrade job.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **validate_only**

  Checks only whether Agents specified are upgradable. An Agent upgrade job will not be submitted.

- **pre_script_loc**

  Executes this script before upgrading the Agent.

- **post_script_loc**

  Executes this script after upgrading the Agent.

- **pre_script_on_oms**

  Use if pre-script is treated to be on OMS.

- **post_script_on_oms**

  Use if post-script is treated to be on OMS.

- **stage_location**

  Passes a custom staging location used by the Agent upgrade job.

- **job_name**

  Submits the job with this name.

- **override_credential**

  Preferred credential of the Oracle home of the Agent used to run root.sh. Use this parameter to override this and use a named Oracle home credential.

- **additional_parameters**

  Passes additional parameters to the Agent upgrade job.

Examples

Example 1

This example checks whether the Agents matching pattern abc% and xyz.domain.com:1243 are upgradable, then submits the Agent upgrade job.

```
emcli upgrade_agents -agents="abc%,xyz.domain.com:1243"
```

Example 2

This example checks whether Agents in the file are upgradable, then submits the Agent upgrade job.

```
emcli upgrade_agents -input_file="agents_file:/scratch/agents_file.txt"
```

Example 3

This example runs /scratch/pre_script before upgrading the Agent xyz.domain.com:1243 .

```
emcli upgrade_agents -agents="xyz.domain.com:1243" -pre_script_loc="/scratch/
pre_script"
```

Example 4

This example runs /scratch/post_script after upgrading the Agent xyz.domain.com:1243 .

```
emcli upgrade_agents -agents="xyz.domain.com:1243" -post_script_
loc="/scratch/post_script"
```

Example 5

This example checks whether Agent xyz.domain.com:1243 is upgradable, then submits UPGRADE_JOB123 .

```
emcli upgrade_agents -agents="xyz.domain.com:1243" -job_name="UPGRADE_JOB123"
```

Example 6

This example uses the NAMED_CRED123 credentials to run the root.sh after upgrading the Agent.

```
emcli upgrade_agents -override_credential="NAMED_CRED123"
```

Example 7

This example runs the Agent, upgrading with the additional parameters passed.

```
emcli upgrade_agents -additional_parameters="-ignorePrereqs -newParameter"
```

Example 8

This example uses the staging location $ENV_DIR on the Agent system.

```
emcli upgrade_agents -stage_location="%$ENV_DIR%"
```

upgrade_database

Upgrades a database.

Format

```
emcli upgrade_database
  -dbTargetName="target_to_be_upgraded"
  -dbTargetType="oracle_database|rac_database"
  -newOracleHome="directory_full_path"
  -hostCreds="named_credentials"
  -sysdbaCreds="named_credentials"
  [-precheck="YES|NO|ONLY"
  [-ignoreWarnings]
  [-diagnosticDest="diagnostic_destination"]
  [-disableArchiveLogMode]
  [-recompileInvalidObjects]
  [[-restoreSettingsOnly] | [-backupLocation="backup_location_full_path"]]
  [-listeners=<name:port[:NEW]>
  [-scriptsFromSoftwareLibrary "scripts_from_software_library"]
  [-beforeUpgradeCustomScript="custom_SQL_file_name"]
  -continueOnScriptError
  [-afterUpgradeCustomScript="Custom_SQL_file_name_upgrade"]
  [-noBlackout]

[ ] indicates that the parameter is optional
```

Parameters

- **dbTargetName**

  Enterprise Manager target name of the database to be upgraded. Versions 10.2.0.4 and above are supported for upgrade.

- **dbTargetType**

  Target type of the database — oracle_database for a single instance database, or rac_database for a cluster database.

- **newOracleHome**

  New Oracle Home directory full path. Upgrade to 11g Release 2 and later is supported. Does not support a database downgrade.

- **hostCreds**

  Named host credentials of the user who owns the Oracle Home installation. Should have necessary privileges on the database files to be upgraded.

- **sysdbaCreds**

  Named database credentials having SYSDBA privileges on the database to be upgraded.

- **precheck**

  Option to run prerequisite checks during the upgrade job. Valid values are:

  YES — Run prerequisite checks and proceed to the database upgrade if there are no errors during prerequisite checks.

  NO — Proceed to the database upgrade directly. Do not run prerequisite checks.

  ONLY — Run prerequisite checks only. Do not upgrade the database.

- **ignoreWarnings**

  Ignores any warnings during prerequisite checking and proceeds with the upgrade. Used only when pre-check is set to YES, otherwise ignored. Does not ignore errors.

- **diagnosticDest**

  Full directory path for Oracle trace and diagnostic files for the upgraded database. By default, ORACLE_BASE is used as the location.

- **disableArchiveLogMode**

  Disable archive logging during the database upgrade.

- **recompileInvalidObjects**

  The upgrade process may invalidate the objects in the database. You can choose to recompile invalid objects at the end of the upgrade. This increases the upgrade time, but minimizes subsequent latencies caused by on-demand automatic recompilation at run time.

- **restoreSettingsOnly**

  Reverts only the configuration changes made during the upgrade if upgrade fails. You can restore the database outside the upgrade using your custom restore strategy. Choose this **option** if you already have a custom backup and restore strategy for this database. In case of an upgrade failure, this setting will be used.

- **backupLocation**

  Full directory path to back up the database. Performs a full backup of the database. A script will be created to restore the database. All files are placed in the specified backup location. Reverts all the changes made during the upgrade if the upgrade fails.

- **listeners**

  Comma-separated list of the listener name and port (name1:port1,name2:port2) to register the upgraded database. Specify at least one listener in the case of a single-instance database target. These listeners should be configured in the new Oracle home or TNS_ADMIN location. Additionally, you can choose to create a new listener in the new Oracle home by specifying :NEW (name1:port1:NEW).

- **scriptsFromSoftwareLibrary**

  Specify the custom scripts from the software library components. The parameters 'beforeUpgradeCustomScript' and 'afterUpgradeCustomScript' are interpreted as entity URNs of the components that contain the scripts.

- **beforeUpgradeCustomScript**

  Full file path of the custom SQL script to be run before the database upgrade.

- **continueOnScripteError**

  Ignores a non-zero exit code when executing a custom SQL script and continues the upgrade job.

- **afterUpgradeCustomScript**

  Full file path of the custom SQL script to be run after the successful database upgrade.

- **noBlackout**

Suppresses a blackout of the database target. A blackout suspends monitoring of the database target from Enterprise Manager, which is the default behavior during a database upgrade.

Examples

```
emcli upgrade_database
   -dbTargetName=test1 -dbTargetType=oracle_database
   -newOracleHome=/u01/app/oracle/product/11.2.0/dbhome_2 -hostCreds=HOST_CREDS
   -sysdbaCreds=SYSDBA_CREDS -precheck=YES -ignoreWarnings -disableArchiveLogMode
   -beforeUpgradeCustomScript=/home/user1/sqlfiles/script1.sql
   -continueOnScriptError
   -afterUpgradeCustomScript=/home/user1/sqlfiles/script2.sql
   -diagnosticDest=/u01/app/oracle
   -recompileInvalidObjects -noBlackout
```

upload_ats_test_databank_file

Uploads a databank file for the specified ATS test.

Format

```
emcli upload_ats_test_databank_file
        -name=<target_name>
        -type=<target_type>
        -testname=<test_name>
        -testtype=<test_type>
        -databankAlias=<databank_alias>
        -input_file:databank=<databank_file>
        [-beaconName=<beacon_name>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the target.

- **type**

  Name of the target type.

- **testname**

  Name of the test.

- **testtype**

  Type of test.

- **databankAlias**

  Databank alias.

- **input_file**

  Databank file.

  For more information about the input_file parameter, see Section 5.2, "-input_file Syntax Guidelines".

- **beaconName**

  Beacon name.

Examples

Example 1

This example uploads the databank file corresponding to the specified test.

```
emcli upload_ats_test_databank_file
      -name="Service Name"
      -type="generic_service"
      -testname="Test Name"
      -testtype="OATS"
      -databankAlias="alias1"
      -input_file="databank:databankFile.csv"
```

Example 2

This example uploads the databank file corresponding to the specified test for the specified beacon.

```
emcli upload_ats_test_databank_file
      -name="Service Name"
      -type="generic_service"
      -testname="Test Name"
      -testtype="OATS"
      -databankAlias="alias1"
      -input_file="databank:databankFile.csv"
      -beaconName="Beacon Name"
```

Example 3

This example uploads the databank file corresponding to the specified test for all of the associated beacons. Any beacon-specific databank file present will be cleared.

```
emcli upload_ats_test_databank_file
      -name="Service Name"
      -type="generic_service"
      -testname="Test Name"
      -testtype="OATS"
      -databankAlias="alias1"
      -input_file="databank:databankFile.csv"
      -beaconName="ALL"
```

upload_patches

Uploads patches to the software library.

Format

```
emcli upload_patches
        -from_host="host_name"
        -patch_files="metadata_file_path;ZIP_file_path"
        [-cred_name="name" -cred_owner="owner"]

[ ] indicates that the parameter is optional
```

Parameters

- **from_host**

  Host from which to get the given patch files.

- **patch_files**

  List of patch file paths. A metadata file and a zip file must be provided.

- **cred_name**

  Named credential used to authenticate the host from which the given patch files are to be uploaded. If you do not provide this option, the normal preferred credential of the host from which the given patch files are to be uploaded will be used by default.

- **cred_owner**

  Owner of the named credentials used to authenticate the host from which the given patch files are to be uploaded.

Examples

```
emcli upload_patches -patch_files="/scratch/p13741363_112310_Linux-x86-64_
M.xml;/scratch/p13741363_112310_Linux-x86-64.zip" -from_host=h1.example.com

emcli upload_patches -patch_files="/scratch/p13741363_112310_Linux-x86-64_
M.xml;/scratch/p13741363_112310_Linux-x86-64.zip" -from_host=h1.example.com -cred_
name=AIMECRED -cred_owner=SYSMAN
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan

submit_patch_plan

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

upload_swlib_entity_files

Uploads one or more files to an entity revision in the software library.

Format

```
emcli upload_swlib_entity_files
        -entity_rev_id="entity_rev_id"
        -file="<abs_file_path>[;<new_file_name>]" | [-removefile="<existing_
          file_name>"]
        -host="hostname"
        [-credential_set_name="setname"] | [-credential_name="name"
          -credential_owner="owner"]
        [-upload_storage="<storage_location_name>;<storage_type>"]
        [-use_latest_revision]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_rev_id**

  Identifier of the entity revision. The software library home page exposes the identifier for folders and entities as a custom column (Internal ID), and is hidden by default.

- **file**

  Absolute path of the file to be uploaded. The file name stored in the software library on upload defaults to the name of the file being uploaded. You can optionally specify a different file name, separated by a semicolon (;).

- **removefile**

  Name of the file to be removed. This is an existing file carried forward from the specified entity revision.

  Alternatively, you can specify the following values:

  ALL — Remove all existing files (no carry forward).
  NONE — Retain all carried forward files.

  The default is NONE.

- **host**

  Target name of the host where the files are available.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target, which can be one of:

  HostCredsNormal — Default unprivileged credential set

  HostCredsPriv — Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. You must specify this option along with the credential_owner option.

- **credential_owner**

Owner of a named credential stored in the repository. You must specify this option along with the credential_name option.

- **upload_storage**

  Destination storage location and type for the upload, separated by a semicolon (;). The location specified must be in 'active' status. This defaults to the storage type and location of the first upload location configured for the software library.

  The storage type can be one of:

  OmsShared — OMS shared file system

  OmsAgent — OMS Agent file system

- **use_latest_revision**

  Flag indicating that the upload should occur to the latest revision of the entity instead of the revision identified by entity_rev_id.

Examples

Example 1

This example uploads the file '/u01/acme_downloads/file1.zip' to the entity revision identified. The file present on the host 'fs1.us.acme.com' should be accessible using the preferred credential set for the 'HostCredsNormal' credential set for the user logged in to EM CLI. The host must be a managed host target in Enterprise Manager, and the Agent on this host must be up and running.

```
emcli upload_swlib_entity_files
        -entity_rev_id="oracle:defaultService:em:provisioning:1:
          cmp:COMP_Component:SUB_Generic:
          B1B1880C6A8C62AAE040548C42832D14:0.1"
        -file="/u01/acme_downloads/file1.zip"
        -host="fs1.us.acme.com"
        -credential_set_name="HostCredsNormal"
```

Example 2

This example uploads the files to the specified entity revision. The file present on the host 'fs1.us.acme.com' should be accessible using the credential named 'MyAcmeCreds' owned by 'ACME_USER'. File '/u01/acme_downloads/file1.zip' after upload will be associated with the entity revision as 'newfile1.zip'. A new revision will be created from the latest revision of the entity.

```
emcli upload_swlib_entity_files
        -entity_rev_id="oracle:defaultService:em:provisioning:1:
           cmp:COMP_Component:SUB_Generic:
           B1B1880C6A8C62AAE040548C42832D14:0.1"
        -file="/u01/acme_downloads/file1.zip;newfile1.zip"
        -file="/u01/acme_downloads/file2.zip"
        -host="fs1.us.acme.com"
        -credential_name="MyAcmeCreds"
        -credential_owner="ACME_USER"
        -use_latest_revision
```

validate_server_generated_alerts

Compares and synchronizes database server-generated alert metric thresholds between Enterprise Manager and target database(s). You can run the verb by specifying one or more specific database target(s) or an Enterprise Manager Group. If you specify a group, compare and synchronize operations are run for all database targets in the group and all of its sub-groups.

This verb generates one row for each server-generated alert metric threshold that is out-of-sync. Each row contains details, such as metric name, object name, critical, and warning thresholds in Enterprise Manager and the target database. Each of the values are separated by a comma.

Format

```
emcli validate_server_generated_alerts
     [-group="group_name" [-fix] [-verbose]]
     [-targets="<target_name1:target_type1;target_name2:target_type2>"
          [-fix]
          [-verbose]]
     [-help]

[ ] indicates that the parameter is optional.
```

Parameters

- group

  Name of the Enterprise Manager group.

- targets

  Target Name, Target Type pair(s). This option is mutually exclusive with the -group option.

- fix

  Synchronizes server-generated alert metric thresholds between Enterprise Manager and the specified database target(s). Server -generated alert metric thresholds in Enterprise Manager are pushed to the target database(s) during the synchronize operation.

- verbose

  Provides a detailed report of out-of-sync thresholds.

- help

  Shows help messages.

Notes

- This verb may not push some metrics to the target database due to non-existent Object(s). For example: non-existent tablespace.
- This verb's default output format is comma-separated values (CSV).

Examples

Example 1

This example compares thresholds for all valid targets in 'ProdGroup' group.

```
emcli validate_server_generated_alerts -group="ProdGroup"
```

Example 2

This example compares and fixes thresholds for all valid targets in 'ProdGroup' group.

```
emcli validate_server_generated_alerts -group="ProdGroup" -fix
```

Example 3

This example compares thresholds between Enterprise Manager and 'ProdDb' (including RAC instances) and 'TestDb' individually.

```
emcli validate_server_generated_alerts
-targets="ProdDb:rac_database;TestDb:oracle_database"
```

Example 4

This example compares and fixes thresholds between Enterprise Manager and target databases 'ProdDb' and 'TestDb' individually.

```
emcli validate_server_generated_alerts
-targets="ProdDb:rac_database;TestDb:oracle_database" -fix
```

Example 5

This example shows this message.

```
emcli validate_server_generated_alerts -help
```

verify_adm

Submits a Verify Application Data Model job for the given Application Data Model and target.

Format

```
emcli verify_adm
     -adm_name=<application_data_model_name>
     -target_name=<target_name>
     -target_type=<target_type>
     [-job_name=<job_name>]
     [-db_cred_name=<database_named_credentials>]
     [-db_pref_creds_name=<database_preferred_credentials>]
     [-job_description=<job_description>]

[ ] indicates that the parameter is optional.
```

Parameters

- adm_name

  Application Data Model name for which the job will be submitted for verification.

- target_name

  Name of the target for which the verify ADM job will be submitted. This can either be the source database or any of the associated databases for the ADM.

- target_type

  Type of target for which the verify aDM job will be submitted for the Application Data Model.

- job_name

  Name of the job to be submitted.

- db_cred_name

  Name of named database credentials stored in the Enterprise Manager repository.

- db_pref_creds_name

  Name of preferred database credentials stored in the Enterprise Manager repository. The valid values for this parameter are:

  – DBCredsNormal — Default normal credential set for a database target.

  – DBCredsSYSDBA — SYSDBA credential set for a database target.

  You must provide a value of either db_pref_creds_name or db_cred_name for the successful submission of the job.

- job_description

  Description of the job to be submitted.

Output

Success/error messages.

Example

This example submits a job with name 'verify adm' on the target test_database for the application data model Sample_ADM.

```
emcli verify_adm
       -adm_name=Sample_ADM
       -target_name=test_database
       -target_type=oracle_pdb
       -job_name="verify adm"
       -db_cred_name=NC_testdb

emcli verify_adm
       -adm_name=Sample_ADM
       -target_name=test_database
       -target_type=oracle_pdb
       -job_name="verify adm"
       -db_cred_name=NC_testdb
       -job_description="verify adm job on test_database"

emcli verify_adm
       -adm_name=Sample_ADM
       -target_name=test_database
       -target_type=oracle_database
       -job_name="verify adm"
       -db_pref_creds_name=NC_testdb_pref
       -job_description="verify adm job on test_database"
```

verify_swlib

Verifies and reports the state of the Software Library.

Format

```
emcli verify_swlib
        [-report_type="storage|entity|uploadjobs|all"]

[ ] indicates that the parameter is optional.
```

Parameters

- **report_type**

  Type of report to be generated, which can be one of the following:

  - **storage** — Reports accessibility of upload storage locations.
  - **entity** — Reports sanity of entities w.r.t associated files in upload storage locations
  - **uploadjobs** — Reports active jobs uploading files to storage.
  - **all** — For all reports.

Example

This example generates the storage, upload jobs, and entities verification reports.

```
emcli verify_swlib
      -report_type="all"
```

verify_updates

Checks for archives that are missing in the software library, and prints steps to download them and re-import them to the software library.

Format

```
emcli verify_updates
```

Parameters

None.

version

Lists EM CLI verb versions or the EM CLI client version.

Format

```
emcli version
    [-verb_name=<verb_name_filter>]
    [-exact_match]
    [-noheader]
    [-script | -format=
               [name:"pretty|script|csv"];
               [column_separator:"column_sep_string"];
               [row_separator:"row_sep_string"];
    ]

[ ] indicates that the parameter is optional
```

Parameters

- **verb_name**

  Verb name filter. Selects matching EM CLI verb names. When you specify this , an output table shows the version for each verb whose name matches <verb_name_ filter>. The EM CLI client version is displayed when you do not specify this **option**.

  Verb filters use regular expression pattern matching unless you specify -exact_match. A zero-length filter matches everything.

  > **Note:** For Unix csh, use single quotes around a filter value containing '$' .

- **exact_match**

  Uses exact matching for filters.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format`=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

– row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Verb, Version (when -verb_name is specified)

Examples

Example 1

This example shows the version for all verbs:

```
emcli version -verb_name=
```

Example 2

This example shows the version for all verbs with names that exactly match the string "sync" :

```
emcli version -verb_name=sync -exact_match
```

Example 3

This example shows the version for all verbs with names starting with "log:"

```
emcli version -verb_name="^log"
```

Example 4

This example shows the version for all verbs with names that end with "in:"

```
emcli version -verb_name="in$"
```

Example 5

This example shows the version for all verbs with names that contain a substring matching "elp" or with names that begin with "ver" or "lo", contains "i", and ends with "n:"

```
emcli version -verb_name="elp|^(ver|lo).*i.*n$"
```

Example 6

This example shows the version for all verbs with names that exactly match the string "setup." Alternatively, you could use the -exact_match .

```
emcli version -verb_name="^setup$"
```

view_redundancy_group

Shows the present configuration of the redundancy group.

Format

```
emcli view_redundancy_group
      -redundancyGroupName="redGrpName"
```

Parameters

- **redundancyGroupName**

  You must specify a single redundancy group name. The target name should be the same as present in the repository, and it should be of target type= "generic_redundancy_group".

Examples

This example shows the details for the 'redGrp1' Redundancy Group.

```
emcli view_redundancy_group -redundancyGroupName='redGrp1'
```

6

Error Code Reference

This chapter documents errors and associated codes returned by EM CLI. You can use EM CLI return codes to manage the control flow in a workflow/scripting environment. EM CLI return codes for Verb errors are positive integers. A Verb returns either 0 (successful execution) or an error number.

The following sections provide reference tables for these types of errors:

- EM CLI infrastructure
- OMS connection
- File-fed option
- Built-in verb

6.1 EM CLI Infrastructure Errors

Any execution of the EM CLI client could result in the following errors.

Table 6–1 Infrastructure Errors

Error Code	Description
242	A Verb has encountered a problem with a dependency specific to the implementation of the Verb (INSIDE of its abstraction barrier) unrelated to the Verb's semantics.
248	Configuration files are corrupt or inaccessible.
253	The command name is not recognized.
254	Internal system error.

6.2 OMS Connection Errors

Verbs that execute at the OMS return these error codes as indicated in the listing for each applicable verb.

Table 6–2 OMS Connection Errors

Error Code	Description
243	License has not been accepted by the current user.
249	Cannot connect to the OMS.
250	Wrong credentials for log in to the OMS.

6.3 File-fed Option Errors

Verbs that allow for file-fed options (rather than options where the values are explicitly defined on the command line) can return the following error codes.

Table 6–3 File-Fed Option Errors

Error Code	Description
244	Cannot find an option value file.
245	Cannot read in an option value file.
246	An option value file is too big.

6.4 Built-in Verb Errors

The following error codes are returned by each verb (not including EM CLI infrastructure errors that apply to all verbs).

Table 6–4 Built-In Verb Errors

Verb	Error Code
add_beacon	0—Beacon added successfully. 129— Syntax Error. The displayed message indicates which argument is syntactically incorrect. 170—Service does not exist. 173—Beacon does not exist. 201—Beacon is already in the monitoring beacons list. 230—Insufficient privileges. 255—Back-end error. Verb failed.
add_group_to_mpa	2—I/O error occurred while writing to the MPA file. 3—The specified MP already exists in the MPA. 4—The group name is empty or not specified. 223—The supplied options are syntactically incorrect.
add_mp_to_mpa	1—File does not exist, is unreadable, or an I/O error occurred. 2—I/O error occurred while writing to the MPA file. 3—The specified MP already exists in the MPA. 4—The target-type definition file cannot be parsed. 5—The MPA filename is not between 1 and 255 characters. 6 —A file of a particular file type is required for another file. 223—The supplied options are syntactically incorrect.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
add_target	1—The supplied target type does not exist. Unable to retrieve target metadata from the specified host's Management Agent.
	2—Host does not exist.
	3—Agent does not exist.
	4—Group does not exist.
	5—No monitoring credentials set found for target in the repository.
	6—Target instance already exists in the repository.
	7—The supplied target properties are incomplete.
	8—One or more of the supplied target properties are invalid.
	15—Target deletion in progress.
	20—Unable to connect to the specified host's Agent.
	21—Unable to save the target instance to the specified host's Agent.
	22—Cannot add more than one Agent target for a single Agent URL.
	23—Unable to add an instance of an Agent target without a URL.
	219—Insufficient privileges to add the target to the group.
	223—Unable to parse command line correctly. Invalid argument value.
	File-Fed Option Errors—The errors associated with file-fed options.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
apply_privilege_delegation_setting	0—Setting successfully applied.
	2—Setting does not exist.
	3—All or some of the targets are invalid.
	129—Syntax error. The displayed message indicates which argument is syntactically incorrect.
apply_template_tests	1—Error processing input XML file.
	4—Insufficient privileges for apply template.
	6—Target does not exist.
	7—Incompatible template and target types during apply.
	8—Test(s) specified for overwriteExisting do not exist in the template.
	9—Key test(s) specified as disabled for apply.
	10—Stepgroup contains a step that does not exist in the file.
	11—Some text property in file does not conform to valid syntax.
	12—Some text property contains variable but variable value is missing.
	13—Some transaction property/threshold/collection setting does not conform to required restrictions.
	50—Generic error.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
argfile	Possible return error codes consist of the following list plus all of the errors returned by the Verb specified in the command line file for execution.
	244—The file does not exist.
	245—There is a problem reading in the file or it does not exist.
	246—The file ends inside a quoted token.
	247—The argfile options are specified incorrectly.
assign_test_to_target	0—Test assigned to target type successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	190—Test or target type invalid.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
change_service_system_assoc	0—Service system changed successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	171—System <system> does not exist.
	172—Key component does not exist.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
clear_credential	1—Target type does not exist.
	2—Target does not exist.
	3—Credential set does not exist.
	4—Insufficient privileges.
	5—Credential column does not exist.
create_aggregate_service	1—Target does not exist.
	2—Target exists.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
create_blackout	1—Blackout X already exists.
	2—Only Super Administrators are allowed to add a new reason (use get_ blackout_reasons).
	3—Agent targets cannot be directly blacked out.
	217—The blackout end_time cannot be in the past.
	The dates specified will never cause this blackout to take effect.
	The difference between the end_time and the start_time must be equal to the duration.
	The difference between the repeat interval and the duration must be at least X minutes.
	The duration must be -1 (for indefinite blackouts) or positive.
	The duration must be at least X minutes.
	219—Current user does not have OPERATOR privilege over all blackout targets.
	220—Target X does not exist.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
create_group	1—Group X already exists.
	2—Cannot add target X to typed group of base type Y.
	218—Group X is currently in the process of being deleted.
	219—Current user does not have privilege X over all member targets.
	220—Member target X does not exist.
	223—Unable to parse command line correctly.
	Invalid argument value.
	Group type is invalid.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
create_privilege_ delegation_setting	0—Setting successfully created.
	129—Syntax error. The displayed message indicates which argument is syntactically incorrect.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
create_red_group	0—Redundancy Group "<red_group_name>" created successfully.
	1—Redundancy Group "<red_group_name>" of target type <red_group_type> already exists.
	2—Cannot add target "<member_target_type>" to typed group of base type "<red_group_type>".
	3—Time Zone Region <timezone_region> does not exist.
	4—Redundancy Group Type "<red_group_type>" is invalid.
	218—Redundancy Group "<red_group_name>:<red_group_type>" is currently in the process of being deleted.
	220—Target "<member_target_name>:<member_target_type>" does not exist.
	223—Redundancy Group name "<red_group_name>" is not valid. It may contain only alphanumeric characters, multi-byte characters, a space, "-", "_", ".", ":", and have a maximum length of 256 characters.
	223—User name "<owner>" is not valid. It must begin with an alphabetic character, contain only alphanumeric characters, underscores (\"_\"), or periods (\".\"), and have a maximum length of 256 characters.
	223—Invalid value for parameter "add_targets": "<add_targets>". Reason: "<add_targets>" is not a name-value pair.
	223—Member Targets not of same type.
	223—"<generic_redundancy_group>" does not support member of type "<member_target_type>" .
create_role	1—Role by same name already exists.
	2—User with same name as role already exists.
	4—Privilege is invalid or nonexistent.
	5—Target specified in one of the privileges is invalid.
	6—The Super Administrator privilege cannot be granted to a role.
	7—Role does not exist.
	8—Group specified in one of the privileges is invalid.
	9—Job in privilege is invalid or nonexistent.
	10—Creating a role that you are assigning to the new role.
	11—The specified user does not exist.
	219—User is unauthorized to perform this action.
	223—Unable to parse command line correctly.
	Invalid argument value.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
create_service	0—Web application created successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	130—Missing key components.
	151—Test validation failed.
	171—System <system> does not exist.
	172—Key component does not exist.
	173—Beacon does not exist.
	181—No key tests defined.
	182—No key beacons defined.
	200—Service <target_name> already exists.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
create_system	0—System "<system_name:system_type>" created successfully.
	110—System "<system_name:system_type>" already exists.
	120—Member target "<member_target_name>:<member_target_type>" does not exist.
	122—Type "<system_type>" is not a valid System type.
	123—Time Zone Region "<timezone_region>" does not exist.
	130—Type meta version "<type_meta_ver>" is invalid.
	223—System name "<system_name>" is not valid. It must begin with an alphabetic char, contain only alphanumeric chars or any of "- _.:", and have a maximum length of 256 chars.
	223—Type meta version "<type_meta_ver>" is invalid. It must contain only numeric and "." characters, and have a maximum length of 8 chars.
	223—Timezone_region cannot be null or blank.
	223—Invalid value for parameter "add_members": "<add_members>". Reason: "<add_members>" is not a name-value pair.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
create_user	1—Target specified in one of the privileges is invalid.
	2—Group specified in one of the privileges is invalid.
	3—Job specified in one of the privileges is invalid.
	4—One of the specified privileges is invalid.
	5—Such user already exists.
	6—One or more roles to be granted to the new user does not exist.
	7—A role with the same name as the new user already exists.
	218—A delete is pending against this user until all blackouts and jobs submitted by this user are stopped.
	219—User has insufficient privileges to perform this operation.
	223—Unable to parse command line correctly:
	Invalid argument value.
	User name is somehow invalid.
	Supplied password does not have the proper format. Example: Password left empty.
	File-Fed Option Errors—The errors associated with file-fed options.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
delete_blackout	1—Blackout X created by user Y does not exist.
	2—Cannot delete a blackout that has not ended or was not stopped.
	219—You (X) do not have the SUPER_USER privilege needed to stop, delete, or modify blackout Y created by user Z.
	Only the blackout owner can stop, delete, or modify the blackout.
	Current user does not have OPERATOR privilege over all blackout targets.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
delete_group	1—Group X does not exist.
	218—Group X is currently in the process of being deleted.
	219—Current user does not have sufficient privileges to perform this action.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
delete_job	1—Specified job is invalid or non-existent.
	219—User has insufficient privileges to perform this operation.
	218—Some executions are not stopped when delete happens.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
delete_metric_promotion	0—SUCCESS
	223—SYNTAX_ERRNUM: Input is malformed.
	255—VERB_FAILED_ERRNUM: Back-end validation fails.
delete_privilege_ delegation_settings	0—Setting successfully deleted.
	2—All or some of the names are invalid.
	129—Syntax error. The displayed message indicates which argument is syntactically incorrect.
delete_role	1—Role does not exist.
	219—User is unauthorized to perform this action.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
delete_system	0—System "<system_name:system_type>" deleted successfully.
	121—System "<system_name:system_type>" does not exist.
	122—Type "<system_type>" is not a valid System type.
	219—Current user does not have sufficient privileges to perform this action.
	223—System name "<system_name>" is not valid. It must begin with an alphabetic character, contain only alphanumeric characters or any of "- _.:", and have a maximum length of 256 chars.
delete_target	15—Target deletion in progress.
	219—Insufficient privileges to delete specified target.
	220—Target does not exist.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
delete_test	0—Test deleted successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	174—Test does not exist.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
delete_user	1—Cannot delete the repository owner.
	2—Specified user does not exist.
	3—Cannot delete the current user.
	218—A delete is pending against this user until all blackouts and jobs submitted by this user are stopped.
	219—User has insufficient privileges to perform this operation.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
disable_audit	223—Syntax Error.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
disable_test	0—Test disabled successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	174—Test does not exist
	203—Test already disabled.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
enable_audit	223—Syntax Error.
enable_test	0—Test enabled successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	174—Test does not exist
	202—Test already enabled.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
execute_hostcmd	0—Command execution succeeded for all targets.
	2—Command execution failed for one or more targets. Detailed errors will be displayed for each failed target.
	3—Invalid or unknown targets in the targets list.
	4—Preferred credentials are missing for one or more targets.
	5—Invalid credential set name.
	223—Unable to parse the command line properly.
execute_sql	0—Command execution succeeded for all targets.
	2—Command execution failed for one or more targets. Detailed errors will be displayed for each failed target.
	3—Invalid or unknown targets in the targets list.
	4—Preferred credentials are missing for one or more targets.
	5—Invalid credential set name.
	223—Unable to parse the command line properly.
export_template	223—Unable to parse command line correctly, or an exception was thrown during SQL handling.
	245—There is a problem writing to the file.
extract_template_tests	2—Error serializing XML output.
	3—Insufficient privileges for extract template.
	5—Template does not exist in repository.
	50—Generic error.
get_aggregate_service_info	1—Target does not exist.
	2—Target exists.
get_aggregate_service_ members	1—Target does not exist.
	2—Target exists.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
get_blackout_details	1—Blackout X created by user Y does not exist. 223—Unable to parse command line correctly. OMS Connection Errors—The errors associated with connecting to the executing OMS.
get_blackout_reasons	OMS Connection Errors—The errors associated with connecting to the executing OMS.
get_blackout_targets	1—Host X does not exist. 223—Unable to parse command line correctly. 220—Target X does not exist.
get_blackouts	1—Host X does not exist. 220—Target X does not exist. 223—Unable to parse command line correctly. OMS Connection Errors—The errors associated with connecting to the executing OMS.
get_group_members	1—Group X does not exist. 223—Unable to parse command line correctly. OMS Connection Errors—The errors associated with connecting to the executing OMS.
get_groups	Other than the confirmation message, the get_groups verb only generates syntax errors. The SQL invoked by get_groups does not throw any exception. 0—All groups (TargetName, TargetType) in the repository are displayed. 223—Syntax Error: Argument -script cannot be specified with a value. 223—Syntax Error: -format argument "name" value must match one of these strings: "script | pretty | csv". 223—Syntax Error: Invalid value for parameter "format": "name:<format_name>;column_separator=<column_separator_char>". Reason: "column_separator=column_separator_char" is not a name-value pair. 223—Syntax Error: -format argument contains an unrecognized key name <key_name>
get_jobs	223—Unable to parse command line correctly. OMS Connection Errors—The errors associated with connecting to the executing OMS.
get_system_members	121—System "<system_name:system_type>" does not exist.
get_targets	223—Unable to parse command line correctly. OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
grant_privs	2—User does not exist.
	3—Invalid privilege.
	4—Invalid target privilege.
	5—Invalid globally unique identifier (GUID).
	6—One or more targets are not groups.
	7—Specified job does not exist.
	8—Privilege grant failed.
grant_roles	2—User does not exist.
	7—Role does not exist.
help	1—There is no help available.
	223—Unable to parse the command line correctly.
import_template	21—Occurs if one of the templates has an OMS version specified in it that does not match the version of the OMS you are importing it into, and there are no other errors.
	22—Occurs if one of the template files cannot be parsed, and there are no other errors.
	99—More than one of the templates to be imported had errors during processing.
	223—Unable to parse command line correctly, or an exception was thrown during SQL handling.
	245—There is a problem reading in the file, or it does not exist.
login	0—Verb success exit value.
	1—Cannot establish an OMS connection storage area, or a corrupt area already exists.
	2—A connection with the OMS cannot be established.
	3—The login with the credentials provided failed at the OMS.
	4— The Enterprise Manager license was not accepted by the current user.
	5—The user is already logged in Enterprise Manager.
	223—Command syntax error Verb exit value.
	241—Custom attribute error handling.
	255—Error code for browser-related errors.
logout	0—Verb success exit value.
	1—Cannot establish an OMS connection storage area, or a corrupt area already exists.
	2—A connection with the OMS cannot be established.
	3—The login with the credentials provided failed at the OMS.
	4— The Enterprise Manager license was not accepted by the current user.
	249—OMS connection error verb exit value.
	255—Error code for browser-related errors.
modify_aggregate_service	1—Target does not exist.
	2—Target exists.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
modify_group	1—Group X does not exist.
	2—Cannot add target X to typed group of base type Y.
	3—Group X contains itself as a sub-group at some level.
	219—Current user does not have sufficient privileges to perform this action:
	Current user does not have privilege X over all member targets. Current user does not have sufficient privileges on target X to add it to the group.
	220—Target X does not exist.
	223—Unable to parse command line correctly. Group type is invalid.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
modify_red_group	0—Redundancy Group ""<red_group_name>" modified successfully.
	1—Redundancy Group ""<red_group_name>:<red_group_type>" does not exist.
	2—Cannot add target "<member_target_type>" to typed group of base type "<red_group_type>".
	4—Redundancy Group Type "<red_group_type>" is invalid.
	218—Redundancy Group "<red_group_name>:<red_group_type>" is currently in the process of being deleted.
	220—Target "<member_target_name>:<member_target_type>" does not exist.
	223—Redundancy Group name "<red_group_name>" is not valid. It may contain only alphanumeric characters, multi-byte characters, a space, "-", "_", ".", ":", and have a maximum length of 256 characters.
	223—User name "<owner>" is not valid. It must begin with an alphabetic character, contain only alphanumeric characters, underscores (\"_\"), or periods (\".\"), and have a maximum length of 256 characters.
	223—Invalid value for parameter "add_targets": "<add_targets>". Reason: "<add_targets>" is not a name-value pair.
	223—Member Targets not of same type.
	223—"Generic redundancy group" does not support member of type "<member_target_type>" .

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
modify_role	4—Privilege is invalid or nonexistent.
	5—Target specified in one of the privileges is invalid.
	6—The Super Administrator privilege cannot be granted to a role.
	7—Role does not exist.
	8—Group specified in one of the privileges is invalid.
	9—Job in privilege is invalid or nonexistent.
	10—Cannot have a circular chain of role grants.
	11—The specified user does not exist.
	219—User is unauthorized to perform this action.
	223—Unable to parse command line correctly. Invalid argument value.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
modify_system	0—System "<system_name:system_type>" modified successfully.
	101—System <system_name:system_type> contains itself as a sub-system at some level.
	120—Member target "<member_target_name>:<member_target_type>" does not exist.
	121—System "<system_name:system_type>" does not exist.
	122—Type "<system_type>" is not a valid System type.
	219—Current user does not have sufficient privileges on target <member_target_name> to add it to the system.
	219—Current user does not have sufficient privileges to perform this action.
	223—Invalid value for parameter "add_members": "<add_members>". Reason: "<add_members>" is not a name-value pair.
modify_target	8—One or more of the supplied target properties are invalid.
	15—Target deletion in progress.
	219—Insufficient privileges to modify target.
	220—Target does not exist.
	223—Unable to parse command line correctly.
	File-Fed Option Errors—The errors associated with file-fed options.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
modify_user	1—Target specified in one of the privileges is invalid. 2—Group specified in one of the privileges is invalid. 3—Job specified in one of the privileges is invalid. 4—One of the specified privileges is invalid. 5—Specified user does not exist. 6—One or more roles to be granted to the new user does not exist. 218—A delete is pending against this user until all blackouts and jobs submitted by this user are stopped. 219—User has insufficient privileges to perform this operation. 223—Unable to parse command line correctly: Invalid argument value or user name is somehow invalid. File-Fed Option Errors—The errors associated with file-fed options. OMS Connection Errors—The errors associated with connecting to the executing OMS.
provision	1—An Internal error occurred. Could not get an Instance of the Assignment Manager. Exception occurred when getting URN from path. 2—Could not provision. Exception occurred either in getting editable ProvisioningAssigment object, or during call to Initiate Provisioning. 3—Could not get one or more URNs. Returned if any of imageUrn, bootServerUrn, stageServerUrn, networkProfileUrn, targetUrn retrieved is null. 4—Could not create assignment state. Failed to create an AssignmentState object. 5—Could not set assignment properties. Failed to set the assignment properties in the assignment state object. Since this verb uses the FileArgRemoteVerb, the following errors are also possible: ■ This Verb posts Verb.SYNTAX_ERRNUM if a specified option/file mapping on the command line is not properly formatted. ■ This Verb posts Verb.LOGIN_SYSTEM_ERRNUM if it cannot log in to the OMS. ■ This Verb posts Verb.OMS_CONNECTION_SYSTEM_ ERRNUM if it cannot connect to the OMS. ■ This Verb posts Verb.CONFIGURATION_SYSTEM_ ERRNUM if the configuration files are corrupt or inaccessible. ■ This Verb posts Verb.MISSING_FILE_SYSTEM_ERRNUM if it cannot find an option value file. ■ This Verb posts Verb.FILE_READ_SYSTEM_ERRNUM if it cannot read in an option value file. ■ This Verb posts Verb.FILE_SYNTAX_SYSTEM_ERRNUM.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
relocate_targets	0—Moved all targets from Source Agent to Destination Agent. 1—Target relocation has failed. The following errors are possible: ■ SQL exception when relocating targets : <Database-specific error message>. ■ Communication exception when relocating targets: < communication exception message >. ■ Verb usage error: `emcli relocate_targets -src_agent=<source agent target name> -dest_agent=<dest agent target name> {-target_name=<name of the target to be relocated> - target_type=<type of the target to be relocated>} | {-input_file=dupTargets:<complete path to file>} {-force=yes}; "` ■ Errors relocating targets from Source Agent to Destination Agent: < error message > < error message > ■ Exception in parsing targets from the command line argument <message>.
remove_beacon	0—Beacon removed successfully. 129—Syntax Error. The displayed message indicates which argument is syntactically incorrect. 170—Service does not exist. 173—Beacon does not exist. 225—Beacon not in monitoring beacons list. 230—Insufficient privileges. 255—Back-end error. Verb failed.
remove_service_system_assoc	0—System removed from service successfully. 129—Syntax Error. The displayed message indicates which argument is syntactically incorrect. 170—Service does not exist. 180—System does not exist. 230—Insufficient privileges. 255—Back-end error. Verb failed.
retry_job	1—Cannot restart job of a non-restartable type. 2—Specified job execution does not exist or has not failed. 3—The specified job execution has already been restarted and failed on restart. 219—User has insufficient privileges to perform this operation. 223—Unable to parse command line correctly. OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
revoke_roles	2—User does not exist.
	7—Role does not exist.
revoke_privs	2—User does not exist.
	3—Invalid privilege.
	4—One or more targets are invalid.
	5—Invalid globally unique identifier (GUID) privilege.
	6—One or more targets are not groups.
	7—Specified job does not exist.
	8—Privilege grant failed.
set_availability	0—Availability set successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	180—No system defined.
	181—No key tests defined.
	182—No key beacons defined.
	230—Insufficient privileges.
	231—Availability not changed.
	255—Back-end error. Verb failed.
set_credential	1—Target type does not exist.
	2—Target (of given target type) does not exist.
	3—Credential set does not exist.
	4—Insufficient privileges.
	5—Credential column does not exist.
	6—Credential column number mismatch.
set_key_beacons_tests	0—Key beacons and tests set successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	135—Must specify at least one key beacon and test.
	170—Service does not exist.
	173—Beacon does not exist.
	175—Beacon not in list of monitoring beacons.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
set_metric_promotion	0—SUCCESS
	223—SYNTAX_ERRNUM: Input is malformed.
	255—VERB_FAILED_ERRNUM: Back-end validation fails.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
set_properties	0—Properties set successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	132—Invalid property.
	133—Invalid property value.
	170—Service does not exist.
	173—Beacon does not exist.
	175—Beacon not in list of monitoring beacons.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
setup	1—The Verb cannot establish a configuration area, or a corrupt area already exists.
	2—A connection with the OMS cannot be established.
	3—The login with the provided credentials fails at the OMS.
	4—The supplied "url" option is malformed or is not http/https.
	5—The configuration directory is not local as determined by the user in non-trustall HTTPS mode.
	6—The Verb cannot collect the user password safely.
	7—License is not been accepted by the user.
	223—Unable to parse command line correctly.
stop_blackout	1—Blackout X created by user Y does not exist.
	2—The blackout has already ended or stopped.
	3—Agent-side blackouts cannot be edited or stopped.
	218—The start of the blackout is currently being processed.
	The blackout is already pending stop.
	The last set of edits to the blackout have not yet been committed.
	219—You (X) do not have the Super Administrator privilege needed to stop, delete, or modify blackout Y created by user Z.
	Only the blackout owner can stop, delete, or modify the blackout.
	Current user does not have OPERATOR privilege over all blackout targets.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
stop_job	1—Specified job is invalid or non-existent.
	219—User has insufficient privileges to perform this operation.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
submit_job	1—Supplied job type is invalid or non-existent.
	2—Job with the same name already exists.
	3—One or more specified targets are invalid.
	4—Missing job parameter.
	5—Invalid job parameters, possibly including the security parameters such as "pwd".
	217—Specified job schedule is invalid.
	219—User has insufficient privileges to perform this operation.
	223—Unable to parse command line correctly.
	Invalid argument value.
	File-Fed Option Errors—The errors associated with file-fed options.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
subscribeto_rule	1—Rule with name X and owner Y does not exist.
	2—EM user X does not exist.
	3—EM user X has no email addresses set up (see console tab Preferences->General).
	4—Outgoing Mail (SMTP) Server not set up (see console tab Setup->Notification Methods).
	219—You (X) do not have the SUPER_USER or MANAGE_ ANY_ USER privilege needed to add email addresses for user Y.
	You (X) do not have the SUPER_USER or MANAGE_ANY_ USER privilege needed to subscribe Y to the rule owned by Z.
	223—Unable to parse command line correctly.
	Invalid argument value.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
sync	1—The Verb cannot establish a configuration area or a corrupt area already exists.
	2—A connection with the OMS cannot be established.
	3—The login with the provided credentials fails at the OMS.
	4—The license has not been accepted by the current user.
	223—Unable to parse the command line correctly.
sync_beacon	0—Beacon synced successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	173—Beacon does not exist.
	175—Beacon not in list of monitoring beacons.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
update_audit_settings	223—Syntax error, which could be an invalid directory name or invalid audit settings.

Table 6–4 (Cont.) Built-In Verb Errors

Verb	Error Code
update_db_password	1—Invalid target. 2—Invalid key value parameter. 3—Invalid old password. 4—Invalid privilege. 223—Syntax error.
update_host_password	1—Invalid target. 2—Invalid key value parameter. 3—Invalid old password. 4—Invalid privilege. 223—Syntax error.
update_password	4—Target (of given target type) does not exist. 5—Credential type does not exist for given target. 6—Key value (that is, user name) does not exist. 7—Non-operator cannot change credentials. 8—Wrong value for old password. 9—Old and new passwords match. 10—No such non_key_column name.

A

Sample Scripts

This appendix provides the sample scripts that were discussed in Chapter 3 in a format that you can use to copy the desired lines into your Jython code.

Example A–1 Script That Retrieves All Targets and Prints Their Names

```
#emcli_get_targets.py

#Import all emcli verbs to current program
from emcli import *

def print_target_details(target):
   '''
   print the target name and target type given a target tuple.
   '''
   print target['Target Name'] + ' ' + target['Target Type']

#Set the OMS URL to connect to
set_client_property('EMCLI_OMS_URL','https://host1.example.com:1234/em')
#Accept all the certificates
set_client_property('EMCLI_TRUSTALL','true')

#Login to the OMS
login(username='adminuser')

#Invoke get_targets and loop over the targets array
targets_array = get_targets().out()['data']
for target in targets_array:
      #Call print_target_details function to print the target details
    print_target_details(target)
```

Example A–2 Script that Incorporates Functions in the get_targets Verb

```
#emcli_introspect_response.py

#Import all emcli verbs to current program
from emcli import *

#Set the OMS URL to connect to
set_client_property('EMCLI_OMS_URL','https://host1.example.com:1234/em')
#Accept all the certificates
set_client_property('EMCLI_TRUSTALL','true')

#Login to the OMS
login(username='sysman')

res = get_targets()
```

```
print 'Number of targets:'+str(len(res.out()['data']))
print 'Errors           :'+res.error()
print 'Exit code        :'+str(res.exit_code())
print 'IsJson           :'+str(res.isJson())
```

Example A–3 Script that Incorporates Custom SQL with the list() Function

```
#emcli_json_processing.py
#Import all EM CLI verbs to current program
from emcli import *
def format(str):
   '''
   Given a string argument returns it back or returns
   a blank string if it is of None type
   '''
   if str is None:
     return ""
   return str

def get_targets_with_props(p_prop_name, p_prop_val):
'''
Returns targets with given property name and its value. Uses list verb.
'''
l_sql = "select target_name, target_type, property_value " \
        "from mgmt$target_properties " \
        "where property_name = '" + p_prop_name + "' " + "   " \
        "and property_value like '" + p_prop_val + "'"
obj=list(sql=l_sql)
   return obj
#Set the OMS URL to connect to
set_client_property('EMCLI_OMS_URL','https://host1.example.com:1234/em')
#Accept all the certificates
set_client_property('EMCLI_TRUSTALL','true')
#Log in to the OMS
login(username='sysman')
#Find all the targets that have Version property set to release 12
l_targets = get_targets_with_props('Version', '12%')
for target in l_targets.out()['data']:
      tn =  target['TARGET_NAME']
      tt =  target['TARGET_TYPE']
      pv =  target['PROPERTY_VALUE']
      print "Name "+tn + " Type =" + tt + " value=" + pv
```

Example A–4 Script that Incorporates Exception Handling

```
#emcli_error_exception_handling.py

#import all emcli verbs to current program
from emcli import *
#import the verbexecutionerror
from emcli.exception import VerbExecutionError

#Set the OMS URL to connect to
set_client_property('EMCLI_OMS_URL','https://host1.example.com:1234/em')
#Accept all the certificates
set_client_property('EMCLI_TRUSTALL','true')

#Login to the OMS
login(username='sysman')
```

```
#Create a group
res = create_group(name='Jan_Doe_Group')

print res.out()

#Try to create the same group again
try:
  #This will trigger an exception as the group exist already
  create_group(name='Jan_Doe_Group')
except VerbExecutionError , e:
   print e.error()
   print 'Exit code:'+str(e.exit_code())
```

Example A–5 LifeCyclePropertyChange.py

```
#Disclaimer
#EXCEPT WHERE EXPRESSLY PROVIDED OTHERWISE, THE SITE, AND ALL CONTENT PROVIDED ON
#OR THROUGH THE SITE, ARE PROVIDED ON AN "AS IS" AND "AS AVAILABLE" BASIS. ORACLE
#EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED,
#INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
#FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT WITH RESPECT TO THE SITE AND ALL
#CONTENT PROVIDED ON OR THROUGH THE SITE. ORACLE MAKES NO WARRANTY THAT: (A) THE
#SITE OR CONTENT WILL MEET YOUR REQUIREMENTS; (B) THE SITE WILL BE AVAILABLE ON AN
#UNINTERRUPTED, TIMELY, SECURE,OR ERROR-FREE BASIS; (C) THE RESULTS THAT MAY BE
#OBTAINED FROM THE USE OF THE SITE OR ANY CONTENT PROVIDED ON OR THROUGH THE SITE
#WILL BE ACCURATE OR RELIABLE; OR (D) THE QUALITY OF ANY CONTENT PURCHASED OR
#OBTAINED BY YOU ON OR THROUGH THE SITE WILL MEET YOUR EXPECTATIONS.
#ANY CONTENT ACCESSED, DOWNLOADED OR OTHERWISE OBTAINED ON OR THROUGH THE USE OF
#THE SITE IS USED AT YOUR OWN DISCRETION AND RISK. ORACLE SHALL HAVE NO
#RESPONSIBILITY FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM OR LOSS OF DATA THAT
#RESULTS FROM THE DOWNLOAD OR USE OF CONTENT.
#ORACLE RESERVES THE RIGHT TO MAKE CHANGES OR UPDATES TO, AND MONITOR THE USE OF,
#THE SITE AND CONTENT PROVIDED ON OR THROUGH THE SITE AT ANY TIME WITHOUT NOTICE.
from emcli import *

search_list = ['PROPERTY_NAME=\'DBVersion\'','TARGET_TYPE=
 \'oracle_database\'','PROPERTY_VALUE LIKE \'11.2%\'']

if len(sys.argv) == 2:

       print login(username=sys.argv[0])
       l_prop_val_to_set = sys.argv[1]
       l_targets = list(resource="TargetProperties", search=search_list,
columns="TARGET_NAME,TARGET_TYPE,PROPERTY_NAME")
       for target in l_targets.out()['data']:
            t_pn =  'LifeCycle Status'
            print "INFO: Setting Property name " + t_pn + " to value " + l_prop_
val_to_set
            print set_target_property_value(property_records=target['TARGET_
NAME']+":"+target['TARGET_TYPE']+":"+t_pn+":"+l_prop_val_to_set)
 else: ]
       print "\n ERROR: Property value argument is missing"
       print "\n INFO: Format to run this file is filename.py <username> <Database
Target LifeCycle Status Property Value>"
```

Example A–6 dbPasswordChange.py

```
#Disclaimer
#EXCEPT WHERE EXPRESSLY PROVIDED OTHERWISE, THE SITE, AND ALL CONTENT PROVIDED ON
#OR THROUGH THE SITE, ARE PROVIDED ON AN "AS IS" AND "AS AVAILABLE" BASIS. ORACLE
```

```
#EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED,
#INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
#FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT WITH RESPECT TO THE SITE AND ALL
#CONTENT PROVIDED ON OR THROUGH THE SITE. ORACLE MAKES NO WARRANTY THAT: (A) THE
#SITE OR CONTENT WILL MEET YOUR REQUIREMENTS; (B) THE SITE WILL BE AVAILABLE ON AN
#UNINTERRUPTED, TIMELY, SECURE,OR ERROR-FREE BASIS; (C) THE RESULTS THAT MAY BE
#OBTAINED FROM THE USE OF THE SITE OR ANY CONTENT PROVIDED ON OR THROUGH THE SITE
#WILL BE ACCURATE OR RELIABLE; OR (D) THE QUALITY OF ANY CONTENT PURCHASED OR
#OBTAINED BY YOU ON OR THROUGH THE SITE WILL MEET YOUR EXPECTATIONS.
#ANY CONTENT ACCESSED, DOWNLOADED OR OTHERWISE OBTAINED ON OR THROUGH THE USE OF
#THE SITE IS USED AT YOUR OWN DISCRETION AND RISK. ORACLE SHALL HAVE NO
#RESPONSIBILITY FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM OR LOSS OF DATA THAT
#RESULTS FROM THE DOWNLOAD OR USE OF CONTENT.
#ORACLE RESERVES THE RIGHT TO MAKE CHANGES OR UPDATES TO, AND MONITOR THE USE OF,
#THE SITE AND CONTENT PROVIDED ON OR THROUGH THE SITE AT ANY TIME WITHOUT NOTICE.

from emcli  import *
from emcli.exception import VerbExecutionError
import sys
import time

def check_job_status(job):
  count=0
  while (count < 10):
   count = count + 1
   obj = emcli.get_jobs(job_id=job)
   #print obj.out()
   for entry in obj.out()['data']:
       l_status = entry['Status ID']
       l_exec_id = entry['Execution ID']
       #print entry['Status ID']
       if (l_status == '5'):
           print "Job completed successfully"
           count=100
       elif (l_status == '4'):
           l_resp = get_job_execution_detail(execution=l_exec_id, showOutput=True,
xml=True)
           print "Job failed, error details "
           print "Output " + str(l_resp.out())
           count=100
       else:
           time.sleep(2)

def update_db_pwd_for_target(p_target_name, p_target_type, p_old_password, p_new_
password):
   l_target_name = p_target_name
   l_target_type = p_target_type
   print "Changing the password for member : name = " + l_target_name + " type = "
+ l_target_type
   try :
       l_resp = update_db_password (target_name=l_target_name,
                                    target_type = l_target_type,
                                    change_at_target="yes",
                                    user_name="dbsnmp",
                                    old_password=p_old_password,
                                    new_password=p_new_password,
                                    retype_new_password=p_new_password)
       l_job_submitted = l_resp.out()['JobId']
       check_job_status(l_job_submitted)
   except emcli.exception.VerbExecutionError, e:
```

```
       print "ERROR : Change Password failed for name = " + l_target_name + " type
= " + l_target_type
       print "ERROR : " + e.error()

def update_db_pwd_for_group(p_group, p_old_password, p_new_password):
   print "Changing the password for group - " + p_group + " from " + p_old_
password + " to " + p_new_password
   members = get_group_members(name=p_group).out()['data']
   for member in members:
       l_target_name = member['Target Name']
       l_target_type = member['Target Type']
       update_db_pwd_for_target(l_target_name, l_target_type, p_old_password, p_
new_password)


#Set the OMS URL to connect to
set_client_property('EMCLI_OMS_URL','https://myoms.com/em')
#Accept all the certificates
set_client_property('EMCLI_TRUSTALL','true')

login(username=sys.argv[0])


l_grp_name = 'maurGroup'

l_group_members = ['db1:oracle_database','db2:oracle_database','db3:rac_database']



res = create_group(name = l_grp_name, add_targets = l_group_members)

print "Listing members for group " + l_grp_name

for member in get_group_members(name=l_grp_name).out()['data']:
   print member


y_n_input = raw_input('Now lets change the password for all the members in this
group(y/n)')
if y_n_input != 'y':
 exit(0)

l_tgt_username = "dbsnmp"
l_old_password = "secret1"
l_new_password = "secret2"

update_db_pwd_for_group(l_grp_name, l_old_password, l_new_password)
```

Example A–7 promote_discovered_dbs.py

```
#Disclaimer
#EXCEPT WHERE EXPRESSLY PROVIDED OTHERWISE, THE SITE, AND ALL CONTENT PROVIDED ON
#OR THROUGH THE SITE, ARE PROVIDED ON AN "AS IS" AND "AS AVAILABLE" BASIS. ORACLE
#EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED,
#INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
#FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT WITH RESPECT TO THE SITE AND ALL
#CONTENT PROVIDED ON OR THROUGH THE SITE. ORACLE MAKES NO WARRANTY THAT: (A) THE
#SITE OR CONTENT WILL MEET YOUR REQUIREMENTS; (B) THE SITE WILL BE AVAILABLE ON AN
#UNINTERRUPTED, TIMELY, SECURE,OR ERROR-FREE BASIS; (C) THE RESULTS THAT MAY BE
```

```
#OBTAINED FROM THE USE OF THE SITE OR ANY CONTENT PROVIDED ON OR THROUGH THE SITE
#WILL BE ACCURATE OR RELIABLE; OR (D) THE QUALITY OF ANY CONTENT PURCHASED OR
#OBTAINED BY YOU ON OR THROUGH THE SITE WILL MEET YOUR EXPECTATIONS.
#ANY CONTENT ACCESSED, DOWNLOADED OR OTHERWISE OBTAINED ON OR THROUGH THE USE OF
#THE SITE IS USED AT YOUR OWN DISCRETION AND RISK. ORACLE SHALL HAVE NO
#RESPONSIBILITY FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM OR LOSS OF DATA THAT
#RESULTS FROM THE DOWNLOAD OR USE OF CONTENT.
#ORACLE RESERVES THE RIGHT TO MAKE CHANGES OR UPDATES TO, AND MONITOR THE USE OF,
#THE SITE AND CONTENT PROVIDED ON OR THROUGH THE SITE AT ANY TIME WITHOUT NOTICE.

 from emcli.exception import VerbExecutionError
 import sys

 alltargets=False
 targetparms=0
 uname=''
 pword=''
 url=''
 monitor_pw=''

 def helpUsage():
    print 'Usage: promote_discovered_dbs.py [-help]'
    print '[-all] Add all discovered Single Instance DBs'
    print '[-targets <target1:target2:...] Add only targets listed'
    sys.exit()

 for i in range(len(sys.argv)):
    if sys.argv[i] in ("-help"):
            helpUsage()
    elif sys.argv[i] in ("-targets"):
            if i+1 < len(sys.argv):
                    targetparms = sys.argv[i+1]
    else:
                    print 'Usage: promote_discovered_dbs.py [-help]'
                    print '[-all] Add all discovered Single Instance DBs'
                    print '[-targets <target1:target2:...] Add only targets
listed'
                    sys.exit()
    elif sys.argv[i] in ("-url"):
            if i+1 < len(sys.argv):
                    url = sys.argv[i+1]
    elif sys.argv[i] in ("-username"):
            if i+1 < len(sys.argv):
                    uname = sys.argv[i+1]
    elif sys.argv[i] in ("-password"):
            if i+1 < len(sys.argv):
                    pword = sys.argv[i+1]
    elif sys.argv[i] in ("-monitor_pw"):
            if i+1 < len(sys.argv):
                    monitor_pw = sys.argv[i+1]
    elif sys.argv[i] in ("-all"):
                    alltargets = True

 # Make sure user did not specify target list and all targets.
 if alltargets<>0 and targetparms <>0:
    print 'Cannot specify target list and all switch'
    print 'Usage: promote_discovered_dbs.py -url <EM URL> -username <username>
-password <password> -monitor_pw <password>'
    print '[-all] Add all discovered SI Databses'
    print '[-targets <target1:target2:...] Add only list targets'
```

```
    print '[-help]'
    sys.exit()

 if len(uname)==0 or len(pword)==0 or len(url)==0:
         print 'Missing required arguments (-url, -username, -password)'
         print 'Usage: promote_discovered_dbs.py -url <EM URL> -username
<username> -password <password> -monitor_pw <password>'
         print '[-all] Add all discovered SI Databses'
         print '[-targets <target1:target2:...] Add only list targets'
         print '[-help]'
         sys.exit()

 # Set Connection properties and logon
 set_client_property('EMCLI_OMS_URL',url)
 set_client_property('EMCLI_TRUSTALL','true')
 login(username=uname,password=pword)

 cred_str = "UserName:dbsnmp;password:" + monitor_pw + ";Role:Normal"

 if targetparms <> 0:
    targetparms = targetparms.replace(":",":oracle_database;")+":oracle_database"
    target_array = get_
targets(unmanaged=True,properties=True,targets=targetparms).out()['data']
 elif alltargets:
    target_array = get_targets(targets="oracle_
database",unmanaged=True,properties=True ).out()['data']
 else:
    print 'Missing required arguments (-targets or -all)'
    helpUsage()

 if len(target_array) > 0:
    for target in target_array:
            print 'Adding target ' + target['Target Name'] + '...',

            for host in str.split(target['Host Info'],";"):
                if host.split(":")[0] == "host":]
                    print host.split(":")[1]
            try:
                res1 = add_target(type='oracle_database',name=target['Target
Name'],host=host.split(":")[1], credentials=cred_
str,properties=target['Properties'])
                print 'Succeeded'
            except VerbExecutionError, e:
                print 'Failed'
                print e.error()
                print 'Exit code:'+str(e.exit_code())
 else:
    print 'INFO: There are no targets to be promoted. Please verify the targets in
Enterprise Manager webpages.'
```

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W