Oracle® Beehive

Integration Guide

Release 2 (2.0.1.8)

E16650-06

August 2013

Documentation for administrators who are responsible for integrating and managing Oracle and third-party, server-based components with Oracle Beehive.

ORACLE®

Oracle Beehive Integration Guide, Release 2 (2.0.1.8)

E16650-06

Primary Authors: Patricia Huey, Roza Leyderman, Sheela Vasudevan

Contributing Authors: Raymond Gallardo, Joe Paradise, Joshua Stanley

Contributors: Vaishnavi Arcot, Preeti Bhoj, Manish Bisarya, Henrik Blixt, Mario Bonin, Warren Briese, Tarun Chauhan, Vimal Chopra, Jason Davis, Manon Delisle, Vikas Dhamija, Jeff Doering, Ramesh Dommeti, Bikash Ghosh, Richard Hall, Marc-Andre Houle, Indira Iyer, William Jacobs, Duane Jensen, Ravi Jupudy, Daniel Kapaya, John Klinke, Archana Kokkula , Fidel Lee, Louise Luo, Tait McCarthy, Arup Mohanty, Paul Nock, Mark Paterson, Francois Perrault, Mohammed Qureshi, Jamie Rancourt, Reza Rokni, John Sawa, Akash Shah, Costa Siourbas, Vilas Varghese, Venkata Naga Vedulai, Jennifer Waywell, Anthony Ye

Contents

3 Integrating Microsoft Exchange Server with Oracle Beehive

4 Integrating an External User Directory with Oracle Beehive

6 Integrating Oracle Universal Content Management with Oracle Beehive

7 Integrating Oracle Universal Records Management with Oracle Beehive

11 Integrating Cisco Voice Gateway with Oracle Beehive Voicemail and Fax

A Example of a Classification Plug-in Collection File

B Oracle Auto-Attendant Markup Language (AAML) Reference

List of Examples

List of Tables

List of Figures

Preface

This guide describes the concepts, considerations, options, and steps associated with integrating and managing Oracle and third-party, server-based components with Oracle Beehive.

This preface contains the following topics:

Audience

This guide is intended for anyone who is responsible for integrating and managing Oracle and third-party, server-based components with Oracle Beehive. This guide is particularly useful to administrators and other resources who plan to deploy and manage Oracle Beehive and any integrated components.

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 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 documents in the Oracle Beehive Release 2 (2.0) documentation library:

Administration Guides

- *Oracle® Beehive Administrator's Guide*

- *Oracle® Beehive Administrator's Reference Guide*
- *Oracle® Beekeeper Online Help (Integrated UA)*

Application Development

- *Oracle® Beehive Application Developer's Guide*
- *Oracle® Beehive Business Views*
- *Oracle® Beehive Java Content Repository Java API Reference*
- *Oracle® Beehive RESTful Web Services API Reference*
- *Oracle® Beehive SOAP Web Services API Reference*

Installation Guides

- *Oracle® Beehive Installation Guide for Linux*
- *Oracle® Beehive Installation Guide for Microsoft Windows*
- *Oracle® Beehive Installation Guide for Solaris Operating System*
- *Oracle® Beehive Installation Help (Integrated UA)*

Online Helps

- *Oracle® Beehive Central*
- *Oracle® Beehive Webmail*
- *Oracle® Beehive Standards-based Clients*
- *Oracle® Beehive Team Collaboration*
- *Oracle® Beehive Conferencing*
- *Oracle® Beehive Extensions for Outlook Supplemental Help and Release Notes*
- *Oracle® Beehive Extensions for Explorer Supplemental Help and Release Notes*
- *Oracle® Beehive Extensions for Explorer (OBEE) (Integrated UA)*
- *Oracle® Beehive Extensions for Outlook (OBEO) (Integrated UA)*

Mobile Devices

- *Oracle® Beehive Using Windows Mobile Device*
- *Oracle® Beehive Using iPhone or iPad*
- *Oracle® Beehive Using BlackBerry*
- *Oracle® Beehive Registering and Configuring Mobile Devices*

Planning Guides

- *Oracle® Beehive Concepts*
- *Oracle® Beehive Deployment Guide*
- *Oracle® Beehive Licensing Information*

Release Notes

- *Oracle® Beehive Release Notes*

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.

1

Integrating Supported Components with Oracle Beehive

Oracle Beehive supports integration with a wide range of optional Oracle and third-party components. This provides organizations flexibility in the deployment options for organizations, especially for those who want to leverage existing or specialized component investments.

This guide describes the concepts, considerations, options, and steps associated with integrating and managing supported components with Oracle Beehive, and includes the following modules:

- "Integrating IBM Lotus Domino with Oracle Beehive"
- "Integrating Microsoft Exchange Server with Oracle Beehive"
- "Integrating an External User Directory with Oracle Beehive"
- "Integrating Oracle Information Rights Management (Oracle IRM) with Oracle Beehive"
- "Integrating Oracle Universal Content Management with Oracle Beehive"
- "Integrating Oracle Universal Records Management with Oracle Beehive"
- "Integrating Oracle Enterprise Manager Grid Control with Oracle Beehive"
- "Integrating Oracle Secure Enterprise Search 10g with Oracle Beehive"
- "Integrating Symantec Scan Engine with Oracle Beehive"
- "Integrating Cisco Voice Gateway with Oracle Beehive Voicemail and Fax"

2

Integrating IBM Lotus Domino with Oracle Beehive

This module describes integrating IBM Lotus Domino with Oracle Beehive. This module contains the following sections:

Overview of Integrating IBM Lotus Domino with Oracle Beehive

Oracle Collaboration Coexistence Gateway is an Oracle proprietary solution that allows the Oracle Beehive users to collaborate with the IBM Lotus Domino users. Additionally, this solution allows the IBM Lotus Domino users to use Oracle Beehive features such as Team Collaboration, and Synchronous Collaboration without having to migrate from IBM Lotus Domino.

Note: The IBM Lotus Domino coexistence solution supports the `Meeting` data type only.

The Oracle Coexistence Gateway must be deployed on a dedicated Domino server or an existing Domino server.

Oracle Collaboration Coexistence Gateway includes the following services:

- Oracle Connector for Domino
- Oracle Free Busy Client
- Coexistence Connector Domino Databases
- Oracle Beehive Coexistence Service

The first three services run on the computer where Oracle Connector for Domino is installed. The last service is on Oracle Beehive. These services update and propagate data between the two coexisting systems.

You can centrally manage Oracle Beehive Coexistence using the `beectl` command and Oracle Beekeeper. This includes managing components of Collaboration Coexistence Gateway Coexistence that are installed on the remote servers.

Note: Prior to integrating IBM Lotus Domino with Oracle Beehive, Oracle recommends that you become familiar with the concepts associated with this integration. To learn more about the associated concepts, including the Oracle Collaboration Coexistence Gateway, see "Oracle Beehive Integration with IBM Lotus Domino Server" in *Oracle Beehive Concepts*.

Procedure for Integrating IBM Lotus Domino with Oracle Beehive

This section describes the procedure for integrating IBM Lotus Domino with Oracle Beehive. This section contains the following topics:

Installing Oracle Collaboration Coexistence Connector for IBM Lotus Domino Server

The *Oracle Beehive Installation Guide for Microsoft Windows* consists of the following information that you will need to install the components of the Oracle Collaboration Coexistence Gateway solution:

- For information about what you will require before beginning the installation, see "Oracle Collaboration Coexistence Gateway" in Module 3 of the *Oracle Beehive Installation Guide for Microsoft Windows*.
- For information about the installation sequence, and installation of the Oracle Collaboration Coexistence Gateway, see "Oracle Collaboration Coexistence Gateway Install Help" in Module 6 of the *Oracle Beehive Installation Guide for Microsoft Windows*.
- For information about the installation sequence for Oracle Coexistence Connector for IBM Lotus Domino Server, see "Oracle Coexistence Connector for IBM Lotus Domino Server Sequence of Screens" in Module 6 of the *Oracle Beehive Installation Guide for Microsoft Windows*.

Configuring the Oracle Collaboration Coexistence Gateway

This section includes information that is necessary to configure the Oracle Collaboration Coexistence Gateway. This section contains the following topics:

Performing Post Installation Configuration Tasks

After installing Oracle Beehive and Oracle Connector for IBM Lotus Domino, and configuring a coexistence system on Oracle Beehive, you must perform additional post installation configuration tasks before the Oracle Collaboration Coexistence Gateway is fully functional.

To run the Oracle Collaboration Coexistence Gateway:

1. If your IBM Lotus Domino deployment does not allow users to receive e-mails from external addresses, the IBM Lotus Domino administrator must reconfigure IBM Lotus Domino to allow incoming external e-mails.
2. Configure a relay server on Oracle Beehive. For additional information about this task, see "Setting Up E-mail Relay Routing" in the *Oracle Beehive Administrator's Guide*.
3. Ensure that your local domains are configured in the Oracle Beehive E-mail Service. You can do this by using Oracle Beekeeper or by running the following `beectl` command:

   ```
   append_value --component _EmailService --name LocalEmailPatterns --value
   "*@<yourdomain>"
   ```

   Where `<yourdomain>` represents your local domain.

   For the configuration changes to take effect, run the `beectl` command, `activate_configuration`.
4. If your IBM Lotus Domino and Oracle Beehive deployments have local users with the same e-mail domain, set the `AcceptInvalidLocalUsers` E-mail Service property to `TRUE`. You can do this using Oracle Beekeeper or by using the following `beectl` command:

   ```
   modify_property --component _EmailService:TransportProperties --name
   AcceptInvalidLocalUsers --value TRUE
   ```

   For the configuration changes to take effect, run the `beectl` command `activate_configuration`.

Rule for Rewriting E-mail Domains

After configuring pre-resolution rules, and applying the changes in Oracle Beehive, run the following commands for each Oracle Beehive primary e-mail domain on each Coexistence Connector system:

```
$ORACLE_HOME\beehive\collabcoex_domino_connector\coexctl_domino.exe set_property
--file
$ORACLE_HOME\beehive\collabcoex_domino_connector\conf\OGLOConfiguration.xml
--section smtpRedirection --property <example.com> --value <special.example.com>
```

Note that you must also configure the pre-resolution rules Oracle Beehive for the rules to take effect. For information about configuring rules in Oracle Beehive, see "Configuring Complex Rule-Based E-mail Parameters using Oracle Beekeeper" in the "Managing Oracle Beehive E-mail" module of the *Oracle Beehive Administrator's Guide*.

Configuring a Coexistence System on Oracle Beehive

This section includes information about adding a coexistence system, by using the `add_coexistence_system` command.

The `add_coexistence_system` command takes either the `--url` parameter or both `--host_name` and `-port` parameters. Use the `--url` option if you need to customize the URI portion of the URL. Otherwise, you can use either of the methods of specifying the host.

Optionally, the `--use_https` parameter can be used along with the `--host_name` and `--port` parameters to specify that the HTTPS protocol is used.

The following examples show how to specify the host by using the different methods described in the preceding paragraphs:

Example 1:

```
beectl> add_coexistence_system --name COEX --product_name  DOMINO --host_name
example.com --port 80 --partnership_key
```

The preceding command generates a URL of the following type:

```
http://example.com:80/coexConnector/CoexMLPort
```

Example 2:

```
beectl> add_coexistence_system --name COEX --product_name  DOMINO --host_name
example.com --port 80 --use_https --partnership_key
```

The command in Example 2 generates a URL of the following type:

```
https://example.com:80/coexConnector/CoexMLPort
```

Example 3:

```
beectl> add_coexistence_system --name COEX --product_name  DOMINO --url
http://example.com -- partnership_key
```

The command in Example 3 generates a URL of the following type:

```
http://example.com:80/coexConnector/CoexMLPort
```

To add a coexistence system to your Oracle Beehive deployment:

1. Determine the fully qualified domain name and port of your Oracle Connector for the IBM Lotus Domino installation.
2. Navigate to the `ORACLE_HOME/beehive/bin` directory on the system hosting the Oracle Beehive deployment.
3. Run the following command:

   ```
   beectl> add_coexistence_system --name <name> --product_name <product_name>
   {--url <url>|--host_name <host_name> --port <port_number> {--use_http|--use_
   https}} --partnership_key [--timezone_alias_namespace <timezone_alias_
   namespace>]
   ```

> **Note:** The argument passed to the `-partnership_key` option is the password defined during the installation of the Oracle Connector for IBM Lotus Domino.

The argument passed to the `--url` option is displayed in the summary screen of the Oracle Connector for Domino installation, where:

- `<host_name>` represents the fully qualified domain name of the system hosting Oracle Coexistence gateway for IBM Lotus Domino.
- `<port>` represents the HTTP port of your Oracle Coexistence Gateway for the IBM Lotus Domino installation.
- `<name>` represents the name that you want to give to the Remote Coexistence System.
- `<partnership_key>` represents the password defined during the installation of the Oracle Coexistence Gateway for IBM Lotus Domino.

- <timezone_alias_namespace> is a string that corresponds to a time zone namespace in Oracle Beehive and is used to map time zones from IBM Lotus Domino to Oracle Beehive, and vice versa. By default this is set to JAVA. Do not change this value, unless advised to do so by Oracle support.

Example 1: Configuring a Coexistence Remote System

In this example, you configure a Coexistence Remote System with the name test against a IBM Lotus Domino residing on a server host named, servertest.ca.oracle.com, by using the following command:

```
beectl>  add_coexistence_system --name test --product_name domino --use_http
--host_name servertest.ca.oracle.com --port 7777 --partnership_key Welcome1
```

This command produces the following result:

```
Coexisting remote system has been successfully created.
--------------------------------------------------------------------------------
General
Name                : TEST
Product Name        : DOMINO
URL                 :
http://servertest.ca.oracle.com:7777/coexConnector/CoexMLPort
Timezone alias namespace: JAVA
```

Explaining Behaviors

This section describes some of the behaviors that you may encounter while installing the Oracle Beehive Collaboration Coexistence Gateway.

This section includes the following topics:

- Event Recurrence Behaviors
- Using E-Mail for Unsupported Flows
- Day Events
- Free/Busy
- Attachments
- Holidays
- Preventive Behavior

Event Recurrence Behaviors

Both Oracle Beehive and IBM Lotus Domino support the concept of repeating meetings with no recurrence rules, which makes it possible to maintain a link between instances even if the particular system does not understand the recurrence rule provided.

However, Lotus Notes does not allow a user to modify the parent information of a recurring meeting. Therefore, a user cannot change the recurrence pattern of a meeting once it has been created. If a recurrence has been modified in Oracle Beehive, then the Coexistence Connector may have to cancel the previous recurrence and invite the participants to a new recurring meeting, which would mean that their participation status is reset.

Additionally, Lotus Notes is unable to modify multiple recurring instances differently with a single message if the changes made are not identical for each instance. As such, the Coexistence Connector will have to send an e-mail for each modified instance to the invited Domino participants. However, Lotus Notes is capable of detecting a set of

related e-mail invitations and ask the users whether they wish to automatically apply the changes to all related meeting e-mails, which improves the usability.

Also note that the IBM Lotus Domino server and the Notes client do not allow unbounded recurrences. However, using the IBM Lotus Domino Outlook Connector, a user can create unbounded recurrences, which the IBM Lotus Domino server would limit to a certain number of instances. For more information, see your IBM Lotus Domino documentation.

Additionally, when modifications are made to a recurring meeting in IBM Lotus Domino, in Oracle Beehive the meetings show every occurrence as an exception.

Using E-Mail for Unsupported Flows

The following e-mail messages are sent to Oracle Beehive with an ICS attachment, where applicable:

- Tasks assignments to Oracle Beehive are sent as e-mail messages.
- Meetings that Domino attendees delegate to Oracle Beehive attendees are sent as e-mail messages, because this feature is not supported by Oracle Beehive.
- Comments or requests for comments when responding to meetings that are accepted are sent to Oracle Beehive as e-mail messages.
- When an Oracle Beehive user proposes a new meeting date, time, or location, the participant Status of the user is set to **Tentative** in Oracle Beehive and the meeting organizer receives the counter proposal through e-mail.

Day Events

IBM Lotus Domino only supports personal day events (there is no support for Day Event invitations.) Therefore, as day events are handled differently in IBM Lotus Domino and Oracle Beehive, the following behaviors are expected:

- Day Events Created in IBM Lotus Domino
- Day Events Created in Oracle Beehive

Day Events Created in IBM Lotus Domino

Day events that are created in IBM Lotus Domino appear in Oracle Beehive as meetings that occur from 4 a.m to 8 p.m. For more information about importing data from IBM Lotus Domino, see "Manually Importing User Data".

Day events created in IBM Lotus Domino by the Outlook Connector appear in Oracle Beehive as meetings that occur from 4 a.m. to 8 p.m.

As a day event is a personal event in IBM Lotus Domino, no free/busy information is sent to Oracle Beehive, and the user's time appears as free for that specific day in Oracle Beehive.

Day Events Created in Oracle Beehive

When a Coexistence Oracle Beehive user invites a Coexistence Remote User to a day event, the day event is displayed in the Coexistence Remote User's agenda on IBM Lotus Domino as a blocking meeting that occurs from 4 a.m. to 8 p.m.

The Coexistence Domino user should then select the **Mark As Available** option while accepting the event so that the user's free/busy status remains unchanged.

Free/Busy

It is recommended to include `coexlotusfb` as part of the IBM Lotus Domino Server Task, so when coexistence starts, the free/busy process stars automatically. To do this, include `coexlotusfb` under the `ServerTask` parameter list of the IBM Lotus Domino configuration file `notes.ini`.

When free/busy information is sent from IBM Lotus Domino to Oracle Beehive, it is always for a 3 months range, starting from the current time, in 15 min interval. Therefore, one hour long meeting created in Lotus Domino occupies four fifteen minute interval, and a five minute meeting occupies an entire interval of fifteen minutes in Oracle Beehive.

Attachments

Oracle recommends that the maximum attachment size (configured by default to be 2 MB) not be increased beyond 10 MB. Any cross-scheduling messages greater than 10 MB will not be transferred from Oracle Connector for Exchange to Oracle Beehive.

Note: The same restrictions apply to files attached to meetings created in Oracle Beehive. You must modify the value of the maximum size for attachments in Oracle Beehive so that the configuration changes are automatically sent to the Oracle Connector for Domino. If the configuration changes are made on Oracle Connector for Domino first, then the same modifications must be made in Oracle Beehive as the changes will not be propagated.

Holidays

Holidays imported within Domino by end users are not imported to Oracle Beehive while provisioning a user as an Coexistence Remote User or as an Coexistence Beehive user. These holidays are not imported by the `import_coexistence_data` option, either.

Preventive Behavior

As Oracle Beehive does not support counter proposals on meeting or delegation on the basis of each meeting, both these options are enforced by default when a Coexistence Beehive User invites a Coexistence Remote user.

Troubleshooting Coexistence

This section describes how to troubleshoot issues that you might encounter while installing and configuring Oracle Collaboration Coexistence Gateway.

Enabling Coexistence Service logging

You can, at any time, modify the logging properties for the Coexistence Service from `beectl`. This will include additional trace in the BEEAPP log itself along with all the other services. Note that only the Coexistence Service logging will then be included in the `log.xml` file. There is no need to bounce the BEEAPP container, changes will take effect immediately. For additional logging, contact support.

Example 7 Enabling CoexistenceService logging

Find your BEEAPP instance name using:

```
beectl> status
```

The preceding command returns a result similar to the following:

```
BEEAPP_site1.stamm01.us.oracle.com
```

Modify the Coexistence Service logging using the following command:

```
beectl>  modify_property --component <beeapp_name>:LoggingProperties --name
ModuleLogLevel --value "oracle.ocs.collabcoex.service.coexistence.task:FINER"
--activate_configuration
```

For example:

```
beectl>  modify_property --component BEEAPP_
site1.stamm01.us.oracle.com:LoggingProperties --name ModuleLogLevel --value
"oracle.ocs.collabcoex.service.coexistence.task:FINER" --activate_configuration
```

Enabling Coexistence Gateway Logs

The Coexistence Gateway generates different log files based on the tasks executed, including:

- Generic communication service log
- Requests coming in from Oracle Beehive
- Requests going out to Oracle Beehive
- Free/busy requests
- Import requests

All log files can be found in the following location (assuming the default installation path):

```
ORACLE_HOME\beehive\collabcoex_domino_connector\logs
```

To modify the log level, navigate to `conf` and edit the `OGLOConfiguration.xml` file and then add the following fragment:

```
<Section Name="loglevel">
<Property Name="default" Value="FINEST" />
</Section>
```

Administering Your IBM Lotus Domino and Oracle Beehive Integration

This section describes how to perform the administrative tasks related to Oracle Collaboration Coexistence Gateway. This section includes the following topics:

- Oracle Beehive Coexistence Service Administrative Tasks
- Oracle Connector for IBM Lotus Domino Administrative Tasks

Oracle Beehive Coexistence Service Administrative Tasks

This section describes how to perform the administrative tasks related to the Oracle Beehive Coexistence Service. The instructions in this section are based on `beectl` shell.

For more information about the `beectl` commands used in this section, see "Oracle Beehive Command-Line Utility" in the *Oracle Beehive Administrator's Reference Guide*.

This section includes the following topics:

Listing Oracle Beehive Remote Coexistence Systems

Oracle Beehive administrators may require details about the Oracle Beehive Remote Coexistence Systems configured on their deployment.

This section includes information about listing all configured Remote Coexistence Systems, by using the `list_coexistence_systems` command. Run the following command to list the configured Remote Coexistence Systems on your Oracle Beehive deployment:

```
beectl> list_coexistence_systems
```

This command produces the following result:

```
--------------------------------------------------------------------------------
Coexisting remote systems:
--------------------------------------------------------------------------------
General
------------------------
Name                    : TEST
Product Name            : DOMINO
URL                     :
http://servertest.ca.oracle.com:7777/coexConnector/CoexMLPort
Timezone alias namespace: JAVA
```

To view additional details, use the `--show_more` option, as shown in the following command:

```
beectl> list_coexistence_systems --show more
```

This command produces the following result:

```
--------------------------------------------------------------------------------
Coexisting remote systems:
--------------------------------------------------------------------------------
General
------------------------
Name                    : TEST
Product Name       : DOMINO
URL                       :
http://servertest.ca.oracle.com:7777/coexConnector/CoexMLPort
Timezone alias namespace: JAVA
Configuration
------------------------
State                   : IN-SYNC
Modified On        : 8/19/09 10:05 AM

Capabilities
------------------------
State                        : IN-SYNC
```

```
Expertise                 : None
Cross-scheduling    : Meeting
```

In addition to the information provided by the `list_coexistence_systems` command, you can use the `beectl` command `list_coexistence_connectors` to display information about individual coexistence connectors within a Remote Coexistence System, as shown in the following example:

```
beectl> list_coexistence_connectors --name TEST
```

This command produces the following result:

```
--------------------------------------------------------------------------------
List of coexistence connectors for a coexisting remote system.
--------------------------------------------------------------------------------
Connector Name          : Beehive
Routing List                    : null
Relay Address Type     : null
--------------------------------------------------------------------------------
```

Modifying Oracle Beehive Remote Coexistence Systems

Oracle Beehive administrators may need to modify certain properties of an Oracle Beehive Remote Coexistence System.

This section includes information about modifying a Remote Coexistence System, by using the `modify_coexistence_system` command. Different options are used with this command depending on the properties that must be modified.

In the following example, the Remote Coexistence System listed in the "Listing Oracle Beehive Remote Coexistence Systems" section, named TEST, is assigned a new port:

```
beectl> modify_coexistence_system --select_by_name TEST --port 80
```

You can also update the `partnership_key` option of the Remote Coexistence System.

For more information about the `modify_coexistence_system` command, including available options and syntax, use the following command:

```
beectl modify_coexistence_system --help
```

Provisioning Users for Coexistence

Before users can take advantage of the Oracle Collaboration Coexistence Gateway, Oracle Beehive administrators must provision users for coexistence with a Remote Coexistence System.

This section includes information about provisioning users for coexistence with a Remote Coexistence System, by using the `modify_coexistence_profile` command.

The following are the prerequisites for provisioning users for coexistence with a Remote Coexistence System using the `modify_coexistence_profile` command:

- The user account must exist on Oracle Beehive.
- Oracle recommends that Oracle Beehive and IBM Lotus Domino users have the same primary e-mail address. Ensure that the IBM Lotus Domino SMTP address property of every user is set to the Oracle Beehive e-mail address. This address should be set as primary.

Example 2: Provisioning a User for Coexistence

In this example, `buser7` is being provisioned for coexistence with Remote Coexistence System TEST.

Use the BEEHIVE argument for the `--accessible_system` option to indicate an Oracle Beehive user who is coexisting. In addition, Oracle Beehive deployment supports multiple Oracle Coexistence Systems. The `--system_affinity` parameter is mandatory while provisioning Oracle Beehive users, as shown in the following example:

```
beectl>  modify_coexistence_profile --accessible_system BEEHIVE --system_affinity
TEST --user loginid=buser7
```

The command produces the following result:

```
Coexistence profile has been created and processing is initiated.
======================================================================
User Name: beehive7

Affinity with system: TEST
Accessible systems: BEEHIVE
Last Modification: 8/20/09 8:19:51 AM
State: CREATED

Request Status: No Request Submitted
Next Attempt After: 8/20/09 8:19:51 AM
======================================================================
```

Example 3: Provisioning a Domino User for Coexistence

In this example, `duser8` is provisioned for coexistence with Remote Coexistence System TEST.

Use the Coexistence system name (as defined when creating the Coexistence Remote System) argument for the `--accessible_system` option to indicate that the user is a Coexisting Domino User, as shown in the following example:

```
beectl>  modify_coexistence_profile --accessible_system TEST --user loginid=duser8
```

This command produces the following result:

```
Coexistence profile has been created and processing is initiated.
======================================================================
User Name: domino8

Affinity with system: TEST
Accessible systems: TEST
Last Modification: 8/20/09 9:47:08 AM
State: CREATED

Request Status: No Request Submitted
Next Attempt After: 8/20/09 9:47:08 AM
======================================================================
```

Using Gather Stats() after Bulk Provisioning User Accounts

The first time you bulk load users, during the upload, the database statistics will quickly become out of date. This can lead to poor system performance, both during and after the upload process. If you are loading a large number of users, consider using the `gather stats()` function in the Oracle Beehive database.

To improve performance, you can use the following SQL*plus command (as either the SYS or BEE_DATA user) to force the database to refresh statistics:

```
exec dbms_stats.gather_schema_stats('BEE_DATA',GRANULARITY=>'ALL');
```

Consider gathering statistics in one of the following ways:

- Split the load into two files, with one small batch and one large. Run the small batch, then use the function `gather stats()`, and then do the larger batch
- Run `gather stats()` while doing the bulk load all at once. This option will slow performance of the bulk load while `gather stats()` is running.

In either case, if you are loading a large amount of users, then this should provide a significant performance improvement after the database command completes.

Post-Provisioning Notes

After running the `modify_coexistence_profile` command, the registration process begins. To verify if the registration process has completed, refer to the "Viewing the Registration State of Coexisting Users" section.

Manually Importing User Data

To avoid or delay the data import process of coexistence users (which normally occurs automatically), run the `modify_coexistence_profile` command with the `--no_data_import` option. You can then use the `import_coexistence_data` command to manually trigger user data import.

The following are the reasons for manually importing user data:

- To customize date ranges and data types for a subset of users.
- To trigger the import at a later time, and not immediately when users are registered for coexistence.
- To reattempt the import manually if the automated data import has failed or partially succeeded.
- To extend the date ranges to get more data in the past.

To register a coexistence user without importing the user data into Oracle Beehive, use the `--no_data_import` option, as shown in the following example:

```
beectl>  modify_coexistence_profile --accessible_system BEEHIVE --no_data_import
--system_affinity TEST --user loginid=buser7
```

To manually import coexistence user data into Oracle Beehive, use the `beectl` command `import_coexistence_data`, as shown in the following example:

```
beectl>  import_coexistence_data {--user <user_cen> | --email <user_email> }
--data_type <type> [--data_type <type> ...] [--timerange_start <start>]
[--relative_timerange_start <relative_start>] [--timerange_end <end>] [--relative_
timerange_end <relative_end>]
```

To specify one or more types of user data to import, use the `--data_type` option. The valid types of user are: `MEETING`, `TASK`, `CONTACT` or `MESSAGE`.

> **Note:** For coexistence with IBM Lotus Domino, Oracle Beehive currently supports the `MEETING` data type only.

To specify an absolute range of time for which all data must be imported, use the `--timerange_start` and `--timerange_end` options. You can use a timestamp, for example, 2007-10-01T12:00:00Z.

You can also use the keywords `NOW` (to specify a time range beginning or ending with the current time) or `INFINITY` (to specify that there should be no limit).

To specify a relative range of time (in days) for which all data must be imported, use the `--relative_timerange_start` and `--relative_timerange_end` options. Specify an integer for each option. A negative integer represents the number of days prior to the current date, while a positive integer represents the number of days from the current date to the value of the positive integer.

For more information about the `import_coexistence_data` command, including available options and syntax, see "import_coexistence_data" in the *Oracle Beehive Administrator's Reference Guide*.

Example 4: Manually Importing User Data

In this example, the `MEETING` data type is imported for user `duser3` from the Remote Coexistence System TEST using the following command.

```
beectl> import_coexistence_data --data_type MEETING --user user=duser3
--timerange_start=-INFINITY --timerange_end INFINITY
```

The command produces the following result:

```
The following data import requests were created:
-------------+----------------------+-----------------------------------------
 Data type   |  Time range start    |  Time range end
-------------+----------------------+-----------------------------------------
MEETING      | -INFINITY            | INFINITY
-------------+----------------------+-----------------------------------------
```

Viewing the Registration State of Coexisting Users

After a user has been provisioned for coexistence, Oracle Beehive attempts to register the user. The user must be registered for coexistence before being able to coexist. Although this occurs automatically, it may take some time to take effect. To view the registration state of coexisting users, use the `list_coexistence_profiles` command, as shown in the following example:

```
beectl> list_coexistence_profiles
```

This command produces the following result:

```
----------+---------------------+--------------------+-----------+-------------
User Name | Affinity with system | Accessible systems | State      | Import state
----------+---------------------+--------------------+-----------+-------------
domino2   | TEST                | TEST               | COMPLETED  | NONE
----------+---------------------+--------------------+-----------+-------------
beehive2  | TEST                | BEEHIVE            | COMPLETED  | NONE
------------+---------------------+--------------------+-----------+------------
```

The following are the five possible values that can appear in the State column:

- `CREATED`: Indicates that the registration request has been created.
- `IN-PROGRESS`: Indicates that the registration request has been sent to the Oracle Beehive Remote Coexistence System, however, the Oracle Beehive Coexistence service is waiting for a response.

- COMPLETED: Indicates that the registration request acknowledgement has been received from the Oracle Beehive Remote Coexistence System and the provisioning, and registration process has completed.
- DELETED: Indicates that the user was once provisioned for coexistence, but no longer is.
- REJECTED: Indicates that the specified systems are not available for a given user. For example, if you specify Coexistence Remote System name (TEST in our example) as the system for a user that does not exist in the IBM Lotus Domino server, then the registration process will fail and the profile will be set to REJECTED.

Note that the value in the State column must be COMPLETED before a user is fully provisioned for coexistence.

To view the additional information about the registration state of coexisting users, use the list_coexistence_profiles command with the -show more option, as shown in the following example:

```
beectl> list_coexistence_profiles -show more
```

The following columns will be displayed when you run the preceding command:

- User Name
- Accessible systems
- State
- Last Modification
- Import State
- Last import

To view the complete information about coexisting users, use the list_coexistence_profiles command with the --show_all option, as shown in the following example:

```
beectl> list_coexistence_profiles --show all
```

A result similar to one of the following examples is displayed for each user:

```
User Name: domino1

Affinity with system: TEST
Accessible systems: TEST
Last Modification: 8/19/09 10:54:09 AM
State: COMPLETED
=====================================================================
User Name: domino3

Affinity with system: TEST
Accessible systems: BEEHIVE
Last Modification: 8/20/09 10:15:24 AM
State: COMPLETED

Data Type: MEETING
GUID: guid-71758457AEB9DF61E040578CF7142BEF0000000188DF
State: CREATED
Latest process time: NOT-STARTED
Received entity: 0
```

```
Expected entity: 0
Time range start: -INFINITY
Time range end: INFINITY
```

Viewing the Import State of Coexisting Users

After a user data is imported from the Coexistence Remote System, Oracle Beehive attempts to import the data type specified for the user.

Although the user data is imported automatically, it may sometimes be delayed. To view the Import state of coexisting users, use the `list_coexistence_profiles` command, as shown in the following example:

```
beectl> list_coexistence_profiles
```

This command produces the following result:

```
----------+---------------------+--------------------+-----------+-------------
User Name | Affinity with system | Accessible systems | State     |Import state
----------+---------------------+--------------------+-----------+-------------
domino3   | TEST                | BEEHIVE            | COMPLETED | CREATED
----------+---------------------+--------------------+-----------+-------------
```

The following four possible values can appear in the Import State column:

- NONE: Indicates that no import request is raised.
- CREATED: Indicates that the import request is created.
- IN-PROGRESS: Indicates that the import request is sent to the Oracle Beehive Remote Coexistence System, however, the Oracle Beehive Coexistence service is waiting for a response.
- COMPLETED: Indicates that the import request acknowledgement has been received from the Oracle Beehive Remote Coexistence System and the data import process has completed.

Deprovisioning Users from an Oracle Beehive Remote Coexistence System

If a user has been provisioned for coexistence, and coexistence is no longer required thereafter, then the user can be deprovisioned.

This section includes information about provisioning users for coexistence with a Remote Coexistence System, by using the `delete_coexistence_profile` command.

Example 5: Deprovisioning a User for Coexistence

In this example, buser1 is deprovisioned for coexistence from Remote Coexistence System TEST, by using the following command:

```
beectl> delete_coexistence_profile --user user=buser1
```

The following result is displayed:

```
Coexistence profile has been deleted and processing is started
```

For more information about the `delete_coexistence_profile` command, see "delete_coexistence_profile" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

After running the `delete_coexistence_profile` command, the deprovisioning process begins.

To view the state of the coexistence profile, refer to "Viewing the Registration State of Coexisting Users".

Oracle Connector for IBM Lotus Domino Administrative Tasks

This section explains how to stop and start Oracle Connector for IBM Lotus Domino. The section includes the following topics:

- Best Practices for Stopping and Starting Oracle Connector for IBM Lotus Domino
- Stopping the Oracle Connector for IBM Lotus Domino
- Starting the Oracle Connector for IBM Lotus Domino
- Stopping the Oracle Free Busy Client
- Starting the Oracle Free Busy Client

Best Practices for Stopping and Starting Oracle Connector for IBM Lotus Domino

Oracle Connector for IBM Lotus Domino works in tandem with the `BeehiveConnectorForDomino` OC4J instance. Both are fully integrated and installed on the computer where Oracle Connector for IBM Lotus Domino was deployed.

Stopping the Oracle Connector for IBM Lotus Domino

The `BeehiveConnectorForDomino` OC4J instance works in tandem with the Oracle Connector for IBM Lotus Domino. It is installed on the same computer as Oracle Connector for IBM Lotus Domino.

To stop the Oracle Connector for IBM Lotus Domino:

1. Click the **Start** button on the task bar in the computer hosting Oracle Connector for IBM Lotus Domino Server.
2. Select **Programs**.
3. Select **Oracle - domino**.
4. Click **Stop SOA suite**.

Starting the Oracle Connector for IBM Lotus Domino

The `BeehiveConnectorForDomino` OC4J instance works in tandem with the Oracle Connector for IBM Lotus Domino. The `BeehiveConnectorForDomino` OC4J instance is installed on the same computer as Oracle Connector for IBM Lotus Domino.

To start the Oracle Connector for IBM Lotus Domino:

1. Click the **Start** button on the task bar in the computer hosting Oracle Connector for IBM Lotus Domino Server.
2. Select **Programs**.
3. Select **Oracle - domino**.
4. Click **Start SOA suite**.

Stopping the Oracle Free Busy Client

The Oracle Free Busy Client works in tandem with the Oracle Connector for IBM Lotus Domino. It is installed on the same computer as Oracle Connector for IBM Lotus Domino.

To stop the Oracle Free Busy Client:

1. From the Domino Lotus Console, run the following command:

   ```
   tell coexlotusfb quit
   ```

2. To verify the Oracle Free Busy Client status, run the `show tasks` command and look for the Oracle FB Client tasks.

Starting the Oracle Free Busy Client

The Oracle Free Busy Client works in tandem with the Oracle Connector for IBM Lotus Domino. It is installed on the same computer as Oracle Connector for IBM Lotus Domino.

To stop the Oracle Free Busy Client:

1. From the IBM Lotus Domino Console, run the `load coexlotusfb` command.
2. To verify the Oracle free Busy Client status, run the `show tasks` command and then look for the Oracle FB Client tasks.

You can also configure the Oracle Free Busy Client to start automatically by adding the task in the `notes.ini` file. To do so, include `coexlotusfb` under the `ServerTask` parameter list of the IBM Lotus Domino configuration file `notes.ini`.

3

Integrating Microsoft Exchange Server with Oracle Beehive

This module describes how to integrate Microsoft Exchange Server with Oracle Beehive.

This module contains the following topics:

Overview of Integrating Microsoft Exchange Server with Oracle Beehive

Oracle Beehive supports integration with Microsoft Exchange Server 2003 and Microsft Exchange Server 2007 through the Oracle Collaboration Coexistence Gateway.

Oracle Collaboration Coexistence Gateway is an Oracle proprietary solution that allows Oracle Beehive users to collaborate with the Microsoft Exchange users. Additionally, the solution allows the Microsoft Exchange users to make use of Oracle Beehive features such as Team Collaboration, and Synchronous Collaboration without necessarily being migrated to Oracle Beehive for Enterprise Messaging.

Oracle Collaboration Coexistence Gateway can be broken down to the following services:

- Oracle Connector for Exchange
- Oracle Change Notification Service for Exchange
- Oracle Communication Service
- Oracle Beehive Coexistence Service

The first three services run on the computer or computers where Oracle Connector for Exchange and Oracle Change Notification Service for Exchange were installed. The last service is on Oracle Beehive Installation.

These services update and propagate data between the coexisting systems. Depending on the deployment, the information being updated and propagated may include events, tasks, and e-mails.

You can centrally manage Oracle Beehive Coexistence from the `beectl` command line and from Oracle Beekeeper. This includes managing components of Collaboration Coexistence Gateway that are installed on remote servers.

Prior to integrating Microsoft Exchange Server 2003 or Microsoft Exchange Server 2007 with Oracle Beehive, Oracle recommends that you become familiar with the concepts associated with this integration. To learn more about the associated concepts, including the Oracle Collaboration Coexistence Gateway, see "Oracle Beehive Integration with Microsoft Exchange Server" in *Oracle Beehive Concepts*.

Note: Version-specific information in this module is highlighted where applicable. For example, for information that is specific to Microsoft Exchange Server 2003, the product name Microsoft Exchange Server 2003 is used. In cases where information applies to Microsoft Exchange Server 2003 and 2007, the generic form Microsoft Exchange Server is used.

Prerequisites for Integrating Microsoft Exchange Server with Oracle Beehive

This section contains the prerequisites for integrating Microsoft Exchange Server with Oracle Beehive. This section contains the following topic:

Load Balancer Requirements for Deploying Oracle Beehive with Microsoft Exchange Server

Load Balancer Requirements for Deploying Oracle Beehive with Microsoft Exchange Server

Depending on the message type, the Oracle Collaboration Coexistence Gateway sends messages between Oracle Beehive and Microsoft Exchange Server over HTTP/HTTPS or SMTP. Therefore, you will need one or more load balancers if your integration plans include more than one Oracle Beehive Application Tier instance or more than one Oracle Coexistence Connector instance. In these cases, load balancers are required to manage the HTTP/HTTPS traffic and, potentially, the SMTP traffic between Oracle Beehive and Microsoft Exchange Server. This requirement is due to the fact that the Coexistence Service (which resides on all Oracle Beehive server instances) and the Oracle Coexistence Connector (which must reside on at least one Microsoft Exchange Server instance) accept only a single URL representing their connection endpoints. Oracle recommends that you configure load balancers for SMTP-based coexistence traffic.

Procedure for Integrating Microsoft Exchange Server with Oracle Beehive

This section contains the procedure for integrating Microsoft Exchange Server with Oracle Beehive.

This section contains the following section:

Installing and Configuring the Oracle Collaboration Coexistence Gateway

Installing and Configuring the Oracle Collaboration Coexistence Gateway

This section includes information and cross-references that are necessary to install and configure the Oracle Collaboration Coexistence Gateway solution. This section contains the following topics:

- Installation
- Post-Installation Configuration
- Post-Installation Tasks

Installation

The *Oracle Beehive Installation Guide for Microsoft Windows* contains all of the information that you will need related to preparing for and installing the components of the Oracle Collaboration Coexistence Gateway solution:

- For information about what you will require before beginning the installation, see "Oracle Collaboration Coexistence Gateway" in Module 3 of the *Oracle Beehive Installation Guide for Microsoft Windows*.
- For information about the installation sequence, and installation of the Oracle Collaboration Coexistence Gateway, see "Oracle Collaboration Coexistence Gateway Install Help" in Module 6 of the *Oracle Beehive Installation Guide for Microsoft Windows*.
- For information about the installation sequence for Oracle Coexistence Connector for Microsoft Exchange Server, see "Oracle Coexistence Connector for Microsoft Exchange Server Sequence of Screens" in Module 6 of the *Oracle Beehive Installation Guide for Microsoft Windows*.
- For information about the installation sequence for Oracle Change Notification Service for Microsoft Exchange Server, see "Oracle Change Notification Service for Microsoft Exchange Server Sequence of Screens" in Module 6 of the *Oracle Beehive Installation Guide for Microsoft Windows*.

Note: After installing the components of the Oracle Collaboration Coexistence Gateway solution, a Remote Coexistence System must be configured in the Oracle Beehive deployment.

Follow the steps outlined in the "Configuring a Coexistence System on Oracle Beehive" section to complete the installation.

Follow the steps outlined in the "Post-Installation Configuration" section to complete the configuration.

Post-Installation Configuration

After installing Oracle Beehive, Oracle Connector for Exchange, and Oracle Change Notification Service for Exchange, and "Configuring a Coexistence System on Oracle Beehive", additional post-installation configuration must be completed before the Oracle Collaboration Coexistence Gateway is fully functional.

Complete all the tasks in this list to get the Oracle Collaboration Coexistence Gateway up and running:

1. Adjust quota restrictions on Microsoft Exchange. For additional information about this task, see "Adjusting Quota Restrictions for the Oracle Connector for Exchange Mailbox".

2. If your Microsoft Exchange deployment does not allow users to receive e-mails from external addresses, the Microsoft Exchange administrator must reconfigure Microsoft Exchange to allow incoming external e-mails.
3. Configure a relay server on Oracle Beehive. For additional information about this task, see "Setting Up E-mail Relay Routing" in Chapter 8, "Managing Oracle Beehive E-mail" of the *Oracle Beehive Administrator's Guide*.
4. Ensure your local domains are configured in the E-mail Service. You can do this with Oracle Beekeeper, or by running the following `beectl` command:

   ```
   beectl> append_value --component _EmailService --name LocalEmailPatterns
   --value "*@<yourdomain>"
   ```

   Where <*yourdomain*> represents your local domain.

   Note: For the configuration changes to take effect, you must run the `activate_configuration` beectl command.

5. Set the `CoexUserLocal` E-mail Service property to `FALSE` using the following beectl command:

   ```
   beectl> modify_property --component _EmailService --name CoexUserLocal --value
   FALSE
   ```

   Note: For the configuration changes to take effect, you must run the `activate_configuration beectl` command.

6. If your Microsoft Exchange and Oracle Beehive deployments have local users with the same e-mail domain, then set the `AcceptInvalidLocalUsers` E-mail Service property to `TRUE`. You can do this using Oracle Beekeeper, or by using the following `beectl` command:

   ```
   beectl> modify_property --component _EmailService:TransportProperties --name
   AcceptInvalidLocalUsers --value TRUE
   ```

Post-Installation Tasks

This section is a compilation of information related to post-installation administration tasks that may or may not be required after installing Oracle Connector for Exchange and Oracle Change Notification Service for Exchange. Read each preamble carefully because not all deployment scenarios will require the post-installation tasks described in this section. For a list or required post-installation tasks, refer to "Post-Installation Configuration".

This section includes the following topics:

Adjusting Quota Restrictions for the Oracle Connector for Exchange Mailbox

During the installation of Oracle Connector for Exchange, a mailbox is created for the service. This mailbox will be given the following name: `Oracle-coexConnExch`

<*host*>. This folder should not be subject to default mail quota policies imposed on regular user folders.

For Oracle Connector for Exchange to function as it was intended, the Microsoft Exchange administrator must remove all mailbox quota policies that may be enforced on this folder before attempting a coexistence deployment.

Installing the Oracle Change Notification Service for Exchange on the Same Host as Microsoft Exchange

While installing Oracle Connector for Exchange on the same computer as Microsoft Exchange, a prompt appears giving you the **Option to Configure Oracle Change Notification Service**. If you select **No** to this option and complete the installation, then you will not be able to install Oracle Change Notification Service for Exchange on that computer using the Oracle Universal Installer.

Note: Oracle does not recommend installing the Oracle Connector for Exchange on the same server that hosts user mailboxes. However, if you choose to install the Oracle Connector for Exchange on a server that hosts user mailboxes, then you will need to install the Oracle Change Notification Service for Exchange on that server too.

If after choosing not to install Oracle Change Notification Service for Exchange on the same computer as Microsoft Exchange and Oracle Connector for Exchange, then you want to install Oracle Change Notification Service for Exchange on that same computer, follow these instructions:

1. On the computer hosting the Oracle Connector for Exchange and Microsoft Exchange, navigate to the `C:\oracle\product\<version>\exconnector_1\beehive\collabcoex_connector`.

   Where <*version*> represents the folder with the version number of Oracle Connector for Exchange.

2. Run the following command:

   ```
   coexctl.exe install_eventsink --install_directory
   C:\oracle\product\<version>\exconnector_1\beehive\collabcoex_connector --admin_
   account <WindowsUserAccount> --admin_password <psw>
   ```

   Where:

   - <*version*> represents the folder with the version number of Oracle Connector for Exchange
   - <*WindowsUserAccount*> represents a Windows user account, with the rights and privileges outlined in the "Windows User Account" section of the *Oracle Beehive Installation Guide for Microsoft Windows*.
   - <*psw*> represents the <*WindowsUserAccount*> user's password

   If the installation is successful, then the following message is returned:

   ```
   EventSink registered successfully.
   ```

3. Start the Oracle Change Notification Service for Exchange. For instructions on starting the Oracle Change Notification Service for Exchange, see "Starting Oracle Change Notification Service for Exchange".

Note: If this method is used to install Oracle Change Notification Service for Exchange, then the Oracle Universal Installer cannot be used to uninstall it. To uninstall Oracle Change Notification Service for Exchange using the command-line tool, run the following command:

```
coexctl.exe uninstall_eventsink
```

Redirecting E-mail Back to Microsoft Exchange for Delivery to Oracle Beehive

By default, Microsoft Exchange redirects all messages directed towards coexisting Oracle Beehive users to the Coexistence Connector, which is then responsible for delivering those messages to Oracle Beehive. However, the following limitations apply when the Coexistence Connector sends the e-mails to Oracle Beehive itself:

- SMTP SSL/TLS is not supported
- SMTP Authentication is not supported
- Receipt notifications are not supported (although delivery and non-delivery reports are supported)
- Retries are not supported. That is, if the Coexistence Connector cannot send the e-mail to Oracle Beehive (through SMTP) on its first attempt, then the Coexistence Connector will not retry.

Beginning with Oracle Beehive Release 1 (1.5), Oracle recommends configuring the Coexistence Connector to redirect all e-mail messages addressed to Oracle Beehive users sent from Microsoft Exchange, back to Microsoft Exchange for delivery.

By having Microsoft Exchange deliver the messages, MIME conversion will be performed by Microsoft Exchange and not by the Coexistence Connector.

If you do not make this configuration change, then listed limitations apply, and you will have to configure e-mail sending options in two locations. Having Microsoft Exchange deliver e-mails enables you to configure e-mail sending, retry intervals, and other Microsoft Exchange features through the usual Microsoft Exchange UI.

Example Redirection Configuration

Suppose that one of the domains of your Oracle Beehive deployment is `example.com`. Configure the system in the following way:

1. The Coexistence Connector is configured to replace all `example.com` addresses in the e-mail envelope with `special.example.com`, and then redirect it to Exchange for delivery.
2. Microsoft Exchange is configured to send messages addressed to `special.example.com` to Oracle Beehive. The message forwarded from the Coexistence Connector will therefore be delivered by Exchange directly to Oracle Beehive.
3. The Oracle Beehive E-mail Service is configured with a pre-resolution rule, such that upon receipt of the message, Oracle Beehive will rewrite `special.example.com` addresses back to `example.com` before delivering the message to the Oracle Beehive user.

To enable this feature, you must add one or more pre-resolution rule for incoming e-mails in Oracle Beehive to rewrite addresses in the e-mail envelopes. A simple rule that rewrites `special.example.com` to `example.com` is listed in Example 3–1, "Simple Rule for Rewriting E-mail Domains". The rule is implemented using a new (in

Oracle Beehive Release 1 (1.5)) Mail Service feature, `RewriteDomainAction`, which can perform regular expressions on the e-mail domain.

> **Caution:** Rewrite actions are run in sequence. This means rewrite result can be matched by subsequent actions by mistake. You should carefully craft the matching patterns to avoid this.

Example 3–1 Simple Rule for Rewriting E-mail Domains

```
<?xml version="1.0" encoding="UTF-8"?>
<java version="1.5.0_16" class="java.beans.XMLDecoder">
 <object class="oracle.ocs.mail.service.ruleengine.RuleEngine">
  <void property="name">
   <string>5e8f2f1b59e7de1f:-bd35894:11d64fcfda8:-8000</string>
  </void>
  <void property="rules">
   <array class="oracle.ocs.mail.service.ruleengine.Rule" length="2">
    <void index="0">
     <object
class="oracle.ocs.mail.service.ruleengine.mail.actions.RewriteDomainAction">
      <void property="name">
       <string>rewrite domain rule</string>
      </void>
      <void property="pattern">
         <string>(special.)(example.com)</string>
      </void>
      <void property="replaceString">
       <object class="java.util.ArrayList">
        <void method="add">
         <string>\2</string>
        </void>
       </object>
      </void>
     </object>
    </void>
    <void index="1">
     <object class="oracle.ocs.mail.service.ruleengine.TopLevelAction">
      <void property="argumentType">
       <string>oracle.ocs.mail.service.mom.MsgContentAndEnvelope</string>
      </void>
      <void property="group">
       <string>THE_RULE</string>
      </void>
      <void property="name">
       <string>THE_RULE</string>
      </void>
      <void property="rootRule">
       <string>rewrite domain rule</string>
      </void>
     </object>
    </void>
   </array>
  </void>
 </object>
</java>
```

If you need to rewrite multiple domains (because your Oracle Beehive users have more than one valid e-mail domain), then you can write a more complex rule. An example of a composite rule that rewrites multiple domains is shown in Example 3–2,

"Composite Rule for Rewriting Multiple E-mail Domains". In this example, `special.example.com` is rewritten to `example.com`, and `special.example.anotherdomain.com` is rewritten to `example.anotherdomain.com`.

Example 3–2 Composite Rule for Rewriting Multiple E-mail Domains

```
<?xml version="1.0" encoding="UTF-8"?>
<java version="1.5.0_16" class="java.beans.XMLDecoder">
 <object class="oracle.ocs.mail.service.ruleengine.RuleEngine">
  <void property="name">
   <string>5e8f2f1b59e7de1f:62dfa7e3:11d6f346721:-8000</string>
  </void>
  <void property="rules">
   <array class="oracle.ocs.mail.service.ruleengine.Rule" length="4">
    <void index="0">
     <object
class="oracle.ocs.mail.service.ruleengine.mail.actions.RewriteDomainAction">
      <void property="name">
       <string>rewrite domain rule</string>
      </void>
      <void property="pattern">
         <string>(special.)(example.com)</string>
      </void>
      <void property="replaceString">
       <object class="java.util.ArrayList">
        <void method="add">
         <string>\2</string>
        </void>
       </object>
      </void>
     </object>
    </void>
    <void index="1">
     <object
class="oracle.ocs.mail.service.ruleengine.mail.actions.RewriteDomainAction">
      <void property="name">
       <string>rewrite domain rule 2</string>
      </void>
      <void property="pattern">
         <string>(special.)(example.)(anotherdomain.com)</string>
      </void>
      <void property="replaceString">
       <object class="java.util.ArrayList">
        <void method="add">
         <string>\2</string>
        </void>
        <void method="add">
         <string>com</string>
        </void>
       </object>
      </void>
     </object>
    </void>
    <void index="2">
       <object class="oracle.ocs.mail.service.ruleengine.CompositeRule">
      <void property="children">
       <array class="java.lang.String" length="2">
        <void index="0">
         <string>rewrite domain rule</string>
```

```
      </void>
      <void index="1">
       <string>rewrite domain rule 2</string>
      </void>
     </array>
    </void>
    <void property="mode">
     <string>SEQUENCE</string>
    </void>
    <void property="name">
     <string>rewrite domains</string>
    </void>
   </object>
  </void>
  <void index="3">
   <object class="oracle.ocs.mail.service.ruleengine.TopLevelAction">
    <void property="argumentType">
     <string>oracle.ocs.mail.service.mom.MsgContentAndEnvelope</string>
    </void>
    <void property="group">
     <string>THE_RULE</string>
    </void>
    <void property="name">
     <string>THE_RULE</string>
    </void>
    <void property="rootRule">
     <string>rewrite domains</string>
    </void>
   </object>
  </void>
 </array>
</void>
</object>
</java>
```

Once you have written the XML to configure your Oracle Beehive E-mail Service pre-resolution rule, perform the following steps to apply it on your Oracle Beehive instance:

> **Note:** In the following procedure, if you have already configured pre-resolution rules, then they will be overwritten with the new rules created in your XML file. You can use the `beectl list_properties` command to see the current value of the `PreResolutionRules` property. If it contains a configuration that you need to preserve, then contact your Oracle Support representative for assistance.

1. Use the `beectl list_components` command to find the identifier of your E-mail Service Transport Properties component:

   ```
   beectl> list_components
   ```

   Search the results for the `EmailService.TransportProperties` component, and copy the corresponding ID. For example:

   ```
   ------------------------------+-----------------------------------------------
   -
   EmailService.TransportPropert | e7ffb5e7-757c-46a1-961d-b3b12fd4ad14
   ```

```
ies                          |
-----------------------------+-----------------------------------------------
-
```

2. Use the `beectl modify_property` command to upload the XML file:

```
beectl> modify_property --component <Transport_properties_ID> --name
PreResolutionRules --file <XML_File>
```

3. Validate and activate your proposed configuration using the `beectl activate_configuration` command:

```
beectl> activate_configuration
```

Once the changes are applied in Oracle Beehive, run the following for each Oracle Beehive primary e-mail domain, on each Coexistence Connector machine:

```
$ORACLE_HOME\beehive\collabcoex_connector\coexctl.exe set_property --file $ORACLE_
HOME\beehive\collabcoex_connector\conf\ConnectorConfiguration.xml --section
smtpRedirection --property <example.com> --value <special.example.com>
```

Substitute the correct domain names for the `--property` and `--value` values.

Administering Your Microsoft Exchange Server and Oracle Beehive Integration

This section explains how to perform various administrative tasks related to Oracle Collaboration Coexistence Gateway, and includes the following topics:

- Oracle Beehive Coexistence Service Administrative Tasks
- Oracle Connector for Exchange Administrative Tasks
- Oracle Change Notification Service for Exchange Administrative Tasks

Oracle Beehive Coexistence Service Administrative Tasks

This section describes how to perform various administrative tasks related to the Oracle Beehive Coexistence Service. The instructions in this section assume that the `beectl` shell is being used.

> **See Also:** For more information about the `beectl` commands used in this section, see "Oracle Beehive Command-Line Utility" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

This section includes the following topics:

- Configuring a Coexistence System on Oracle Beehive
- Listing Oracle Beehive Remote Coexistence Systems
- Modifying Oracle Beehive Remote Coexistence Systems
- Provisioning Users for Cross-Scheduling Coexistence
- Viewing the Registration State of Coexisting Users
- Deprovisioning Users from an Oracle Beehive Remote Coexistence System

Configuring a Coexistence System on Oracle Beehive

This section includes information about adding a coexistence system using the `add_coexistence_system` command.

The `add_coexistence_system` command takes either a `--url`, or both a `--host_name` and `--port`. You must use the `--url` option if you need to customize the URI portion of the URL. Otherwise, you can use either method of specifying the host.

Optionally, the `--use_https` parameter can be used along with the `--host_name` and `--port` parameters to specify that the HTTPS protocol will be used.

The following examples show how to format input using each method:

```
beectl add_coexistence_system --host_name example.com --port 80 --partnership_key
```

This command generates a URL of `http://example.com:80/coexConnector/CoexMLPort`

```
beectl add_coexistence_system --host_name example.com --port 80 --use_https
--partnership_key
```

This command generates a URL of `https://example.com:80/coexConnector/CoexMLPort`

```
beectl add_coexistence_system --url http://example.com --partnership_key
```

This command generates a URL of `http://example.com:80/coexConnector/CoexMLPort`

```
beectl add_coexistence_system --url https://example.com/uri --partnership_key
```

This command generates a URL of `https://example.com:80/uri`

Follow these steps to add a coexistence system to your Oracle Beehive deployment:

1. Determine the fully qualified domain name and port of your Oracle Connector for Exchange installation.
2. Navigate to the `ORACLE_HOME/beehive/bin` directory on the system hosting the Oracle Beehive deployment.
3. Run the following command:

   ```
   beectl> add_coexistence_system --name <name> --product_name <product_name>
   {--url <url>|--host_name <host_name> --port <port_number> {--use_http|--use_
   https}} --partnership_key [--timezone_alias_namespace <timezone_alias_
   namespace>]
   ```

   Note: The argument passed to the `-partnership_key` option is the password defined during the installation of the Oracle Connector for Exchange or Oracle Connector for Lotus Domino. For more information about the Partnership Key value, see "Oracle Collaboration Coexistence Gateway Install Help" in Module 6 of the *Oracle Beehive Installation Guide for Microsoft Windows*.

   The argument passed to the `--url` option is displayed in the summary screen of the Oracle Connector for Exchange installation, where:

   - *<OracleConnectorForExchangeHost>* represents the fully qualified domain name of the system hosting Oracle Connector for Exchange

Note: Specify the fully qualified domain name of the load balancer in the case of a deployment with multiple routing groups.

- <*port*> represents the HTTP port of your Oracle Connector for Exchange installation
- <*coexistence_system_name*> represents the name that you want to give to the Remote Coexistence System
- <*timezone_alias_namespace*> is a string that corresponds to a time zone namespace in Oracle Beehive and is used to map time zones from Exchange to Oracle Beehive and vice versa. By default this is set to `MSFT`. Unless advised to do so by Oracle support, do not change this value.

Listing Oracle Beehive Remote Coexistence Systems

Oracle Beehive administrators may require details about the Oracle Beehive Remote Coexistence Systems configured on their deployment. This section includes information about listing all configured Remote Coexistence Systems using the `list_coexistence_systems` command.

Run the following command to list the configured Remote Coexistence Systems on your Oracle Beehive deployment:

```
beectl> list_coexistence_systems
```

The command will return output similar to the following example:

```
--------------------------------------------------------------------------------
Coexistence remote collaboration system details.
--------------------------------------------------------------------------------
Name                    : Exchange
URL                     : http://10.156.42.99:7777/coexConnector/CoexMLPort
Timezone alias namespace: MSFT
--------------------------------------------------------------------------------
```

For additional details, use the `--show_more` option:

```
beectl> list_coexistence_systems --show more

--------------------------------------------------------------------------------
Coexistence remote collaboration system details.
--------------------------------------------------------------------------------
Id                      :
121A:2A16:cors:56F748517E90371EE040578CF71854800000000186A6
Name                    : Exchange
URL                     : http://10.156.42.99:7777/coexConnector/CoexMLPort
Timezone alias namespace: MSFT
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Configuration properties status.
--------------------------------------------------------------------------------
Modified On             : 9/19/08 10:17 AM
Last synchronization    : 9/19/08 10:17 AM
Propagation state       : IN-SYNC
```

In addition to the information provided by the `list_coexistence_systems` command, you can use the `beectl list_coexistence_connectors` command to

show information about individual coexistence connectors within a Remote Coexistence System:

```
beectl> list_coexistence_connectors

--------------------------------------------------------------------------------
List of coexistence connectors for remote collaboration systems.
--------------------------------------------------------------------------------
Connector Name          : Oracle-coexConnExch (tmcoex1)
Routing List            : RGC426AE:*;1;OCS:*;1;
Relay Address Type      : RGC426AE
--------------------------------------------------------------------------------
```

Modifying Oracle Beehive Remote Coexistence Systems

Oracle Beehive administrators may need to modify certain properties of an Oracle Beehive Remote Coexistence System. This section includes information about modifying a Remote Coexistence System using the `modify_coexistence_system` command. Depending on what properties require modification, different options are required.

In the following example, the Remote Coexistence System listed in the "Listing Oracle Beehive Remote Coexistence Systems" section, named `TMCOEX2`, is being assigned a new URL:

```
beectl> modify_coexistence_system --select_by_name TMCOEX2 --url
http://new.example.com
```

You can also update the Remote Coexistence System's `partnership_key` attribute.

> **See Also:** For more information about the `modify_coexistence_system` command, including available options and syntax, see "modify_coexistence_system" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Provisioning Users for Cross-Scheduling Coexistence

Before users can take advantage of the Oracle Collaboration Coexistence Gateway, Oracle Beehive administrators must provision users for coexistence with a Remote Coexistence System. Note that while provisioning users for coexistence, you must specify the connector as there can be multiple connectors defined in Oracle Beehive.

This section includes information about provisioning users for coexistence with a Remote Coexistence System using the `modify_coexistence_profile` command.

> **Note:** Oracle recommends that the maximum attachment size (configured by default to be 2MB) not be increased beyond 20MB. Any cross-scheduling messages greater than 20MB will not be transferred from Oracle Connector for Exchange to Oracle Beehive.

Prerequisites

Before provisioning users for coexistence with a Remote Coexistence System using the `modify_coexistence_profile` command the following conditions must be met:

- The user account must exist on Oracle Beehive. For information about provisioning users on Oracle Beehive, see "Provisioning User Accounts Using beectl" in the *Oracle Beehive Administrator's Guide*.
- Oracle recommends that Oracle Beehive and Microsoft Exchange users have the same primary e-mail address. Ensure that the every user's Microsoft Active Directory SMTP address property is set to the Oracle Beehive e-mail address. This address should be set as primary.

Example 3–3 Provisioning a User for Coexistence

In this example, `user2` is being provisioned for coexistence with Remote Coexistence System `TMCOEX2`.

Using the `BEEHIVE` argument for the `--accessible_system` option indicates an Oracle Beehive user who will be coexisting:

```
beectl> modify_coexistence_profile --user user=user2 --accessible_system BEEHIVE

Coexistence profile has been created and processing is initiated.
```

Using the `EXCHANGE` argument for the `--accessible_system` option indicates that the user is a Coexisting Third Party User (the user can access Microsoft Exchange only):

```
beectl> modify_coexistence_profile --user user=user2 --accessible_system EXCHANGE
```

See Also: For more information about the `modify_coexistence_profile` command, including available options and syntax, see "modify_coexistence_profile" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Using Gather Stats after Bulk Provisioning User Accounts

The first time you bulk load users, during the upload, the database statistics will quickly become outdated. This can lead to poor system performance, both during and after the upload process. If you are loading a large number of users, then consider using the `gather stats()` function in the Oracle Beehive database.

To improve performance, you can use the following SQL*plus command (as either the `SYS` or `BEE_DATA` user) to force the database to refresh statistics:

```
exec dbms_stats.gather_schema_stats('BEE_DATA',GRANULARITY=>'ALL');
```

Consider gathering statistics in one of the following ways:

- Split the load into two files, with one small batch and one large. Run the small batch, then `gather stats()`, and then do the larger batch
- Run the `gather stats ()` while doing the bulk load all at once. This option will slow performance of the bulk load while `gather stats()` is running.

In either case, if you are loading a large amount of users, then this should provide a significant performance improvement after the database command completes.

Post-Provisioning Notes

After running the `modify_coexistence_profile` command, the registration process will begin. To view whether the registration process has completed, see "Viewing the Registration State of Coexisting Users".

Manually Importing User Data

To avoid or delay the registration process of coexistence users (which normally occurs automatically), you can run the `beectl modify_coexistence_profile` command with the `--no_data_import` option. You can then use the `beectl import_coexistence_data` command to manually trigger user data import.

You might wish to do this for any of the following reasons:

- You want to customize date ranges and data types for a subset of users.
- You want to trigger the import at a later time, and not immediately when users are registered for coexistence.
- If the automated data import failed or partially succeeded, you can reattempt the import manually.
- You want to extend the date ranges to get more data in the past.

To register a coexistence user without importing that user's data into Oracle Beehive, use the `--no_data_import` option, such as in the following example:

```
beectl> modify_coexistence_profile --user user=<user_login_id> --accessible_system
BEEHIVE --no_data_import
```

To manually import coexistence user data into Oracle Beehive, use the `beectl import_coexistence_data` command:

```
beectl> import_coexistence_data {--user <user_cen> | --email <user_email> }
[--data_type <type>] [--timerange_start <start>] [--relative_timerange_start
<relative_start>] [--timerange_end <end>] [--relative_timerange_end <relative_
end>]
```

Use the `--data_type` option to specify one or more particular types of user data to import. Valid types are `MEETING` and `TASK`.

Use the `--timerange_start` and `--timerange_end` options to specify an absolute range of time for which all data should be imported. You can use a timestamp, for example `2007-10-01T12:00:00Z`. You can also use the keywords `NOW` (to specify a time range beginning or ending with the current time) or `INFINITY` (to specify that there should be no limit).

Use the `--relative_timerange_start` and `--relative_timerange_end` options to specify a relative range of time (in days) for which all data should be imported. Specify an integer for each option. A negative integer represents a number of days prior to today, while a positive integer represents a number of days in the future (from today).

See Also: For more information about the `import_coexistence_data` command, including available options and syntax, see "import_coexistence_data" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Viewing the Registration State of Coexisting Users

After a user has been provisioned for coexistence using the steps outlined in "Provisioning Users for Cross-Scheduling Coexistence", Oracle Beehive attempts to register the user. The user must be registered for coexistence before being able to coexist. Although this occurs automatically, it may take some time.

To view the registration state of coexisting users use the `list_coexistence_profiles` command:

```
beectl> list_coexistence_profiles
```

Output similar to the following will be returned:

```
--------------+--------------------+-----------+------------------------------
User Name     | Accessible systems  | State     | Import state
--------------+--------------------+-----------+------------------------------
user1         | BEEHIVE             | COMPLETED | NONE
--------------+--------------------+-----------+------------------------------
user3         | EXCHANGE            | COMPLETED | COMPLETED
--------------+--------------------+-----------+------------------------------
```

Five possible values appear in the `State` column:

- `CREATED` indicates that the registration request has been created.
- `IN-PROGRESS` indicates that the registration request has been sent to the Oracle Beehive Remote Coexistence System, but the Oracle Beehive Coexistence service is waiting for a response.
- `COMPLETED` indicates that the registration request acknowledgement has been received from the Oracle Beehive Remote Coexistence System and the provisioning, and registration process has completed.
- `DELETED` indicates that the user is not provisioned for coexistence anymore.
- `REJECTED` indicates that the specified systems were not possible for a given user. For example, if you specify `EXCHANGE` as one of the systems for a user that does not exist in the Microsoft Exchange Server, the registration process will fail and the profile will be set to `REJECTED`.

The value in the `State` column must be `COMPLETED` before a user has been fully provisioned for coexistence.

To view the additional information about the registration state of coexisting users, use the `list_coexistence_profiles` command with the `--show_more` option:

```
beectl> list_coexistence_profiles --show_more
```

The following columns are displayed:

- User Name
- Accessible systems
- State
- Last Modification
- Import State
- Last import

To view complete information about coexisting users, use the `list_coexistence_profiles` command with the `--show_all` option:

```
beectl> list_coexistence_profiles --show_all
```

The output similar to the following is displayed for each user:

```
================================================================
User:user2
Accessible system: BEEHIVE,EXCHANGE
Profile process state: COMPLETED
Profile modifiedon: 8/5/08 7:34:07 AM
```

```
Data Type: TASK
GUID: guid-F14819DF1B5F4DA683E1C9393DFF106E000000000001
State: CREATED
Latest process time: NOT-PROCESSED
Received entity: 0
Expected entity: 0
Time range start: 7/6/08 2:28:08 PM UTC
Time range end: INFINITY

Data Type: MEETING
GUID: guid-F14819DF1B5F4DA683E1C9393DFF106E000000000000
State: CREATED
Latest process time: NOT-PROCESSED
Received entity: 0
Expected entity: 0
Time range start: 7/6/08 2:28:06 PM UTC
Time range end: INFINITY
================================================================
```

Deprovisioning Users from an Oracle Beehive Remote Coexistence System

If a user has been provisioned for coexistence, and coexistence is no longer needed thereafter, then the user can be deprovisioned. This section includes information about provisioning users for coexistence with a Remote Coexistence System using the `delete_coexistence_profile` command.

Example 3–4 Deprovisioning a User for Coexistence

In this example, `user2` is being deprovisioned for coexistence from Remote Coexistence System `TMCOEX2`:

```
beectl> delete_coexistence_profile --user user=user2

Coexistence profile has been deleted and processing is started.
```

> **See Also:** For more information about the `delete_coexistence_profile` command, including available options and syntax, see "delete_coexistence_profile" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Post-Deprovisioning Notes

After running the `delete_coexistence_profile` command, the deprovisioning process will begin. To view the state of the coexistence profile, see "Viewing the Registration State of Coexisting Users".

Oracle Connector for Exchange Administrative Tasks

This section explains how to stop and start Oracle Connector for Exchange. The following topics are covered in this section:

- Best Practices for Stopping and Starting Oracle Connector for Exchange
- Stopping Oracle Connector for Exchange
- Starting Oracle Connector for Exchange
- Stopping the BEECONNECTOR OC4J Instance

- Starting the BEECONNECTOR OC4J Instance
- Configuring Oracle Connector for Exchange to use HTTPS

When installed in the default location, the command line tool to stop and start Oracle Connector for Exchange is located in the `C:\oracle\product\<version>\exconnector_1\beehive\collabcoex_connector` folder. This section assumes that you have navigated to the above mentioned directory from the Windows command line prompt.

Best Practices for Stopping and Starting Oracle Connector for Exchange

Oracle Connector for Exchange works in tandem with the `BEECONNECTOR OC4J` instance. Both are installed on the computer where Oracle Connector for Exchange was deployed.

Due to the interdependency between Oracle Connector for Exchange and the `BEECONNECTOR OC4J` instance, Oracle recommends stopping and starting the service and instance in a specific order.

Stopping When stopping the Oracle Connector for Exchange, Oracle recommends the following steps:

1. Stop the `BEECONNECTOR OC4J` instance. For more information on stopping the `BEECONNECTOR OC4J` instance, see "Stopping the BEECONNECTOR OC4J Instance".
2. Stop Oracle Connector for Exchange. For more information about stopping Oracle Connector for Exchange, see "Stopping Oracle Connector for Exchange".

Starting When starting the Oracle Connector for Exchange, Oracle recommends the following steps:

1. Start Oracle Connector for Exchange. For more information on starting Oracle Connector for Exchange, refer to "Starting Oracle Connector for Exchange".
2. Start the `BEECONNECTOR OC4J` instance. For details about starting the `BEECONNECTOR OC4J` instance, see "Starting the BEECONNECTOR OC4J Instance".

Stopping Oracle Connector for Exchange

Before stopping the Oracle Connector for Exchange, the `BEECONNECTOR OC4J` instance must be stopped. For details about stopping the `BEECONNECTOR OC4J` instance, see "Stopping the BEECONNECTOR OC4J Instance".

To stop Oracle Connector for Exchange, run the following command:

```
coexctl.exe stop_connector
```

> **Note:** Oracle Connector for Exchange can also be stopped in the Windows Services panel, by selecting and stopping Oracle Coexistence Connector for Exchange and Oracle Coexistence Administration Service.

Starting Oracle Connector for Exchange

To start Oracle Connector for Exchange, run the following command:

```
coexctl.exe start_connector
```

> **Note:** Oracle Connector for Exchange can also be started in the Windows Services panel, by selecting and starting Oracle Coexistence Connector for Exchange and Oracle Coexistence Administration Service.

Stopping the BEECONNECTOR OC4J Instance

The `BEECONNECTOR OC4J` instance works in tandem with the Oracle Connector for Exchange. It is installed on the same computer as Oracle Connector for Exchange, and can be stopped using the following instructions:

1. Click the **Start** button on the Windows computer hosting Oracle Connector for Exchange
2. Select **Programs**
3. Select **Oracle - coex**
4. Click **Stop SOA suite**

Starting the BEECONNECTOR OC4J Instance

The `BEECONNECTOR OC4J` instance works in tandem with the Oracle Connector for Exchange. The `BEECONNECTOR OC4J` instance is installed on the same computer as Oracle Connector for Exchange, and can be started using the following instructions:

> **Note:** The Oracle Connector for Exchange should be started before starting the BEECONNECTOR OC4J instance. For details about starting Oracle Connector for Exchange, refer to "Starting Oracle Connector for Exchange".

1. Click the **Start** button on the Windows computer hosting Oracle Connector for Exchange
2. Select **Programs**
3. Select **Oracle - coex**
4. Click **Start SOA suite**

Configuring Oracle Connector for Exchange to use HTTPS

If Oracle Beehive was configured to use HTTPS, then the Oracle Connector for Exchange must be configured to use the Oracle Beehive HTTPS URL.

Follow these steps to configure Oracle Connector for Exchange to use the Oracle Beehive HTTPS URL:

1. Stop the `BEECONNECTOR OC4J` Instance. For information about stopping the `BEECONNECTOR OC4J` Instance, see "Stopping the BEECONNECTOR OC4J Instance".
2. Using the command prompt on the computer hosting Oracle Connector for Exchange, navigate to the `C:\oracle\product\<version>\exconnector_1\beehive\collabcoex_connector` directory.

   Where <*version*> represents the installed version of Oracle Connector for Exchange.

3. Run the following command:

```
coexctl.exe set_property --file C:\oracle\product\<version>\beehive\collabcoex_
connector\conf\OGWEConfiguration.xml --section communicationservice --property
endpoint --value
"https://<OracleBeehiveHost>:<OracleBeehivePort>/coexService/CoexMLPort"
```

   Where:

   - <*OracleBeehiveHost*> represents your Oracle Beehive hostname
   - <*OracleBeehivePort*> represents the HTTPS listening port of your Oracle Beehive installation

4. Start the `BEECONNECTOR OC4J` Instance. For information about starting the `BEECONNECTOR OC4J` Instance, refer to "Starting the BEECONNECTOR OC4J Instance".

Oracle Change Notification Service for Exchange Administrative Tasks

This section explains how to stop and start the Oracle Change Notification Service for Exchange from the command line.

When installed in the default location, the command line tool to stop and start the Oracle Change Notification Service for Exchange is located in the `c:\oracle\product\<version>\exconnector_1\beehive\collabcoex_connector` folder. This section assumes that you have navigated to the above mentioned directory from the Windows command line prompt.

Stopping Oracle Change Notification Service for Exchange

To stop the Oracle Change Notification Service for Exchange, run the following command:

```
coexctl.exe stop_eventsink
```

> **Note:** You can stop Oracle Change Notification Service for Exchange in the Windows Services panel, by selecting and stopping Oracle Change Notification Service for Exchange.

Starting Oracle Change Notification Service for Exchange

To start the Oracle Change Notification Service for Exchange, run the following command:

```
coexctl.exe start_eventsink
```

> **Note:** You can start Oracle Change Notification Service for Exchange in the Windows Services panel, by selecting and starting Oracle Change Notification Service for Exchange.

4

Integrating an External User Directory with Oracle Beehive

This module provides an overview of integrating an external User Directory Service (UDS) with Oracle Beehive. This module describes how to integrate and synchronize UDS with an external LDAP-based directory, such as Oracle Internet Directory, so that all user data is mastered by the LDAP-based directory.

This module includes the following sections:

Overview of Integrating an External User Directory with Oracle Beehive

Oracle Beehive provides flexible user account management and provisioning by supporting both native and system-external user directory options. With Oracle Beehive, administrators can manage user account data either natively in Oracle Beehive or externally through integration with a supported LDAP-based user directory server. Oracle Beehive provides this flexibility for user account management through the User Directory Service.

Currently, Oracle Beehive supports the following user directory servers:

- Oracle Internet Directory
- IBM Tivoli Directory Server
- Microsoft Active Directory Server
- OpenLDAP Directory Server
- Oracle Directory Server Enterprise edition (formerly Sun Directory Server)

Organizations can also leverage external user directories to manage user authentication attributes, such as usernames and passwords. This option requires Oracle Beehive administrators to configure the Authentication Service after installation using the `beectl` command-line utility. For more information regarding this option, see *Oracle Beehive Administrator's Guide*.

Oracle Beehive user data may be mastered by the Oracle Beehive User Directory Service (UDS) or an external LDAP-based directory. An LDAP-mastered user is authenticated against the LDAP server. Some identity attributes of this user are stored on the LDAP server, and the remaining attributes can be stored in Oracle Beehive. For

example, if a user's `givenname` and `familyname` attributes are stored on the LDAP server, then they are only for display in Oracle Beehive. The `phonenumber` and `middlename` attributes of the same user may be stored on Oracle Beehive and can be modified in Beehive.

If UDS is synchronized with an external LDAP server, then it will contact the LDAP server at regular intervals for all records that were changed. UDS will update its records accordingly. You may change the frequency that UDS contacts the LDAP server.

If you make a change in UDS, then it will not update the LDAP server with which it is synchronized.

> **Note:** It is not necessary to master all user account attributes in an LDAP server. Some attributes may be mastered in LDAP and others in UDS. However, all users that need to authenticate or login must be mastered in the same place.

If you choose to integrate Oracle Beehive with a supported external user directory, then you should consider the limitations of this option as well as the steps to prepare your deployment for this option. This topic is covered in the following section:

Limitations of Deploying Oracle Beehive with an External User Directory

Limitations of Deploying Oracle Beehive with an External User Directory

Despite the multiple options and benefits provided by Oracle Beehive's support of external user directories, there are certain limitations as follows:

- Oracle Beehive does not support integration with multiple, unique user directories. Oracle Beehive can only integrate with a single user directory instance on a specific server.
- User accounts can only be mastered in one place, either natively in Oracle Beehive or in an external user directory. Despite this requirement, administrators may choose to master most user account attributes in one directory even if the user accounts themselves are mastered in Oracle Beehive or in an external user directory. However, certain user account attributes that are specific to Oracle Beehive, such as voicemail passwords and instant messaging user names, can only be mastered in Oracle Beehive.
- To manage user account attributes that are mastered in external LDAP directories, administrators can only use the tools provided with or for those directories. Administrators cannot use the tools provided with Oracle Beehive, such as `beectl`, to manage user account attributes that are mastered in external LDAP directories.
- The synchronization process between the external user directory servers and Oracle Beehive is unidirectional. The changes in an external user directory are imported into Oracle Beehive only. Oracle Beehive does not promote the user account attributes that it manages to external user directories.

Prerequisites for Integrating an External User Directory with Oracle Beehive

This section describes the requirements for integrating an external user directory with Oracle Beehive. This section includes the following topics:

Creating XML Files for Integrating an LDAP server with UDS

After installing Oracle Beehive but before deploying the system with an external user directory, you must import user account data from your external user directory server instance to UDS. This process involves creating the following XML files:

- **LDAP mapping profile**: An XML file that contains external user directory server settings and specifies how to convert those entries to Oracle Beehive users and groups. This involves specifying attribute mappings between those defined in an external directory server and those used by Oracle Beehive.
- **User file**: An XML file that represents, in a format specified by Oracle Beehive, all the user accounts that need to be synchronized. To create this, administrators use the LDAP mapping profile and the `beectl download_ldap_user_data` command.

OpenLDAP Directory Requirements

If you choose to integrate Oracle Beehive with OpenLDAP, ensure that you have configured OpenLDAP Directory as follows to synchronize it with Oracle Beehive:

- Do not use the option `lastmod off` in your `slapd.conf` configuration file. Either omit the option or use `lastmod on` instead.
- Ensure that all OpenLDAP users have the following attributes set to a value other than null:
  - entryUUID
  - `modifiersName`
  - ModifyTimestamp

  Verify that these attributes are set with the `ldapsearch` command:

  ```
  ldapsearch -h <hostname> -p <port> -b "<user base>" <attribute names>
  ```

  For example, if the host and port of your OpenLDAP Directory is `myldap.example.com:8888`, and the base DN of your users is `cn=Users,dc=example,dc=bee`, then call the following command:

  ```
  ldapsearch -h myldap.example.com -p 8888 -b "cn=Users,dc-example,dc=bee"
  entryUUID modifiersName ModifyTimestamp
  ```

- Ensure that you have configured the system time setting correctly on the OpenLDAP server. If the system time (as retrieved from the date command) is not configured correctly on the OpenLDAP server, then synchronization may not work.

  It is required that the time settings on OpenLDAP server must be the same as that of the Oracle Beehive database or must be ahead of the Oracle Beehive database time. Note that the time settings on OpenLDAP server must not be behind the Beehive database time settings.

For more information, including the steps to create these files, refer to "Step 1: Creating an LDAP Mapping Profile".

Procedure for Integrating an External User Directory with Oracle Beehive

Integrating LDAP with UDS consists of the following steps:

- Step 1: Creating an LDAP Mapping Profile
- Step 2: Loading Users and Groups
- Step 3: Enabling Synchronization

Notes: You will need the user name and password of a user of your LDAP server who has access to the following:

- Attributes in the Directory Information Tree (DIT)
- Change logs

This user does not need write access to your LDAP server. However, if you later integrate Microsoft Exchange or IBM Lotus Domino Coexistence Solution, then you would require write access to the LDAP server.

The steps in this module will use the user `cn=orcladmin`.

These steps use Oracle Internet Directory as the LDAP server to synchronize.

This section also covers these topics:

- Controlling How Often UDS Contacts the LDAP Server
- Retrieving Information About the LDAP Server

Step 1: Creating an LDAP Mapping Profile

The LDAP mapping profile is an XML file that tells UDS the following information:

- Which LDAP entries should be synchronized
- How to treat entries with specific attributes or domain names (DN) (for example, whether to map them as `ENTERPRISE_USER`, `EXTENDED_ENTERPRISE_USER`, or `EXTERNAL_PERSON`)
- How to map the attributes of each user type to Oracle Beehive attributes.

Creating the LDAP mapping profile consists of the following steps:

- Step A: Creating an LDAP Mapping Profile from a Template
- Step B: Renaming the Profile
- Step C: Specifying LDAP Server Settings
- Step D: Providing Mapping Details for Each User Type and Static Group
- Step E: Providing Scope and Membership Mapping Information
- Step F: Providing Attribute Mapping for Each User Type and Static Group
- Step G: Adding Profile to Oracle Beehive

Step A: Creating an LDAP Mapping Profile from a Template

Navigate to the directory `ORACLE_HOME/beehive/templates/uds/`. It contains LDAP mapping profile templates for the following LDAP servers. These templates

must be edited and customized depending on how your LDAP directory is configured and structured:

Table 4–1 LDAP Mapping Profile Templates

File Name	Directory Type
adprofile_template.xml	Microsoft Active Directory
ibmprofile_template.xml	IBM Tivoli Directory Server
oidprofile_template.xml	Oracle Internet Directory
sunprofile_template.xml	Sun Directory Server
openldap_profile_template.xml	OpenLDAP Directory

Depending on your LDAP server, copy one of these files to another location, such as your home directory. Edit this file to create your LDAP mapping profile.

These steps use `oidprofile_template.xml`.

Step B: Renaming the Profile

Rename the profile in the `<profile_name>` tag. The following is an excerpt from an LDAP mapping profile:

```
<profile>
  <profile_name>my_profile</profile_name>
  <!-- ... -->
</profile>
```

Step C: Specifying LDAP Server Settings

Enter the LDAP server's host and port, administrator's user name and password (which must be obfuscated), and the users and groups base search. The following is an excerpt from an LDAP mapping profile:

```
<ldap_server>
  <host>www.ldapserver.com</host>
  <port>389</port>
  <!-- <ssl_port>636</ssl_port> -->
  <connection_timeout>
    120
    <!-- This is the default value, in seconds -->
  </connection_timeout>
  <ldap_user_name>cn=orcladmin</ldap_user_name>
  <!-- obfuscated password -->
  <ldap_user_password>
    fCgF4UPWg+Vm7IkSBSY07NOSkJ2XXTYRwGynrIM0mx/CHQF4W58Mab0izRX6Bxb6
  </ldap_user_password>
  <user_search_base>dc=oracle,dc=com</user_search_base>
  <group_search_base>
    cn=groups,dc=us,dc=oracle,dc=com
  </group_search_base>
  <primary_authentication_attribute>uid</primary_authentication_attribute>
  <!-- The primary authentication attribute is required only
       for DEFAULT profile -->
  <digest_authentication>
    <!-- Corresponds to the DigestAuthentication
         property of the component _CURRENT_SITE:LdapServer.
         This property can have multiple digest_authentication_attribute
```

```
        values -->
    <digest_authentication_attribute>
      <!-- An attribute from the user object (in the LDAP directory)
           that is required for digest authentication -->
    </digest_authentication_attribute>
  </digest_authentication>
</ldap_server>
```

In this excerpt, only users under `dc=oracle,dc=com` will be mapped to Oracle Beehive users. Similarly, only groups under `cn=groups,dc=us,dc=oracle,dc=com` will be mapped to Oracle Beehive groups.

Note: After configuring the LDAP Server configuration, set `AuthStoreType` on the `_AuthenticationService` configuration to `ldap`. Once this property is set, run `activate_configuration` followed by `modify_local_configuration_files`. If you do not perform these steps, then the LDAP entries would not be present in the `system-jazn-data.xml` file.

The `<connection_timeout>` element is used by UDS to establish a connection to an external directory. If UDS cannot establish a connection within the number of seconds specified in this element, it aborts the connection attempt. The default value is 120 seconds. This element is available for Oracle Beehive Release 1 (1.3) and later.

For more information about the `DigestAuthentication` property, see "Configuring Digest Authentication".

Notes: The LDAP user specified in `<ldap_user_name>` must have access to the change logs. If you later update the profile with a different LDAP user, then UDS will be synchronized with the state of the LDAP server corresponding to the latest change log number.

To obfuscate the LDAP administrator's password, use the `beectl obfuscate` command:

```
beectl obfuscate --expiration_time_in_minutes 0
Enter value for password:
Confirm value of password:
Successfully obfuscated the string.
fCgF4UPWg+Vm7IkSBSY07NOSkJ2XXTYRwGynrIM0mx/CHQF4W58Mab0izRX6Bxb6
```

Other `beectl` commands require obfuscated passwords. Use the same command to obfuscate them.

If your LDAP server is Microsoft Active Directory, then the user specified in `<ldap_user_name>` must have the following privileges:

- Read access on the directory objects being synchronized.
- Viewing rights to the deleted objects container in Microsoft Active Directory. For more information, refer to "How to let non-administrators view the Active Directory deleted objects container in Windows Server 2003 and in Windows 2000 Server" at `http://support.microsoft.com/kb/892806`.

Step D: Providing Mapping Details for Each User Type and Static Group

Provide the mapping details for each user type and static group. The following is an excerpt from an LDAP mapping profile:

```
<user_type_map>
  <user_type_map_entry>
    <source_field_type>DN</source_field_type>
    <source_field_value>
      cn=users,dc=partners,dc=oracle,dc=com
    </source_field_value>
    <user_type>EXTENDED_ENTERPRISE_USER</user_type>
  </user_type_map_entry>
  <user_type_map_entry>
    <source_field_type>DN</source_field_type>
    <source_field_value>
      cn=users,dc=us,dc=oracle,dc=com
    </source_field_value>
    <user_type>ENTERPRISE_USER</user_type>
  </user_type_map_entry>
</user_type_map>

<group_type_map>
  <group_type_map_entry>
    <source_field_type>DN</source_field_type>
    <source_field_value>
      cn=groups,dc=us,dc=oracle,dc=com
    </source_field_value>
    <group_type>STATIC_GROUP</group_type>
  </group_type_map_entry>
</group_type_map>
```

> **Note:** In case of multiple `user_type_map rules`, ensure that you put the specific rule first, and then the generic rule. In this way, if the first rule does not get satisfied, then the subsequent rules are evaluated.

This excerpt maps the following entries:

- A user that is under the DN specified in `<user_search_base>` (in this example, it is `dc=oracle,dc=com`) and whose DN contains `cn=users,dc=partners,dc=oracle,dc=com` will be mapped to `EXTENDED_ENTERPRISE_USER`.
- An entry that is under the DN specified in `<user_search_base>` and whose DN contains `cn=users,dc=us,dc=oracle,dc=com` will be mapped to `ENTERPRISE_USER`.

> **Note:** Users of type `EXTENDED_ENTERPRISE_USER` must be specified before users of type `ENTERPRISE_USER`.
>
> Once the users have been created in Oracle Beehive, they cannot be converted or updated to another type of user. For example, a user of type `ENTERPRISE_USER` cannot be converted to a user of type `EXTENDED_ENTERPRISE_USER` (and the other way around). Similarly, an `EXTERNAL_PERSON` cannot be converted to an `ENTERPRISE_USER` or `EXTENDED_ENTERPRISE_USER`.

Exclusion and Inclusion Consider the following example:

```
<user_type_map>
  <user_type_map_entry>
    <source_field_name>UserStatus</source_field_name>
    <source_field_type>ATTRIBUTE</source_field_type>
    <source_field_value>true</source_field_value>
    <user_type>ENTERPRISE_USER</user_type>
  </user_type_map_entry>
</user_type_map>
```

In this example, a user (created in your LDAP directory) whose `UserStatus` attribute is set to `true` will be mapped to `ENTERPRISE_USER`.

However, if `UserStatus` is changed to any value other than `true` or nullified, then UDS synchronization will set the user's status as `DISABLED` in Oracle Beehive because the user no longer satisfies the condition specified in this `<user_type_map>`.

If `UserStatus` is changed back to `true`, then UDS synchronization will set the user's status as `ENABLED` in Oracle Beehive.

If a user in LDAP is deleted, then UDS synchronization will set the user's status as `MARKED_FOR_DELETE` in Oracle Beehive.

> **Note:** The attribute specified in `<source_field_name>` (in the previous example, this would be `UserStatus`) must be of a string data type in your LDAP directory. UDS synchronization does not support any other LDAP data types. In addition, the values specified in `<source_field_name>` and `<source_field_value>` must exactly match an attribute and corresponding value (respectively) in your LDAP directory.

Step E: Providing Scope and Membership Mapping Information

Provide community mapping information. Enter this information in an `<scope_type_map>` element. Users specified in a `<scope_type_map>` will be added to, or scoped within, the community (organization or enterprise) specified in the same element. A user may only be scoped within a single community.

You may optionally specify a `<membership_type_map>` element. Users specified in this element will be scoped within the community (organization or enterprise) specified by `<scope_type_map>`. In addition, users will become a member of the community specified in the `<membership_type_map>` element. A user may be a member of zero or more communities.

The following is an excerpt from an LDAP mapping profile.

```
<scope_type_map>

  <scope_type_map_entry>
    <source_field_type>DN</source_field_type>
    <source_field_value>dc=us,dc=example,dc=com</source_field_value>
    <scope>
      <name>Entr1</name>
      <identifier>enpr=Oracle</identifier>
    </scope>
<membership_type_map>
      <membership_type_map_entry>
        <source_field_type>DN</source_field_type>
        <source_field_value>
```

```
        dc=external,dc=us,dc=example,dc=com</source_field_value>
      <name>My_Organization</name>
      <identifier>orgn=My_Organization,enpr=My_Enterprise</identifier>
    </membership_type_map_entry>
  </membership_type_map>
 </scope_type_map_entry>
 <scope_type_map_entry>
</scope_type_map>
```

This excerpt maps the following entry:

- A user that is under the DN `dc=us,dc=oracle,dc=com` will be scoped within the enterprise `My_Enterprise`.
- A user that is under the DN `dc=external,dc=us,dc=example,dc=com` will be scoped within the same enterprise (`My_Enterprise`). The same user will be a member of the organization `My_Organization`.

  Tips: To retrieve the identifier for an enterprise, call the following `beectl` command:

  ```
  beectl list_enterprises
  ----------------------------------------------
  | Enterprise Name      | Identifier          |
  ----------------------------------------------
  | MyEnterprise         | enpr=My_Enterprise  |
  ----------------------------------------------
  ```

  To retrieve the identifier for an organization, call the following `beectl` command:

  ```
  beectl list_organizations --scope enpr=My_Enterprose

  Organization name:    My_Organization
  Description:          Unknown
  Identifier:           orgn=My_Organization,enpr=My_Enterprise
  ...
  ```

Step F: Providing Attribute Mapping for Each User Type and Static Group

Provide the attribute mapping for each user type and static group. The following is an excerpt from an LDAP mapping profile:

```
<directory_attribute_map>
  <directory_attribute_map_entry>
    <source_object>ENTERPRISE_USER</source_object>
    <AttributeMap>
      <Field>
        <source_attribute>givenname</source_attribute>
        <target_attribute>GIVENNAME</target_attribute>
        <target_attribute_type>ATTRIBUTE</target_attribute_type>
      </Field>
      <Field>
        <source_attribute>sn</source_attribute>
        <target_attribute>FAMILYNAME</target_attribute>
        <target_attribute_type>ATTRIBUTE</target_attribute_type>
      </Field>
    </AttributeMap>
  </directory_attribute_map_entry>

  <directory_attribute_map_entry>
    <source_object>EXTENDED_ENTERPRISE_USER</source_object>
```

```
    <AttributeMap>
      <Field>
        <source_attribute>givenname</source_attribute>
        <target_attribute>GIVENNAME</target_attribute>
        <target_attribute_type>ATTRIBUTE</target_attribute_type>
      </Field>
      <Field>
        <source_attribute>sn</source_attribute>
        <target_attribute>FAMILYNAME</target_attribute>
        <target_attribute_type>ATTRIBUTE</target_attribute_type>
      </Field>
    </AttributeMap>
  </directory_attribute_map_entry>

  <directory_attribute_map_entry>
    <source_object>STATIC_GROUP</source_object>
    <AttributeMap>
      <Field>
        <source_attribute>displayname</source_attribute>
        <target_attribute>NAME</target_attribute>
        <target_attribute_type>ATTRIBUTE</target_attribute_type>
      </Field>
    </AttributeMap>
  </directory_attribute_map_entry>
</directory_attribute_map>
```

In this excerpt, for each `ENTERPRISE_USER`, the `givenname` LDAP attribute will be mapped to the `GIVENNAME` attribute in Oracle Beehive. Similarly, for each `STATIC_GROUP`, the `displayname` LDAP attribute will be mapped to the `NAME` attribute in Oracle Beehive.

Mapping Postal Addresses You may use the `ORAPOSTAL` user account address field scheme to map the postal address attributes of the users in your LDAP directory.

The `ORAPOSTAL` scheme contains the following fields:

- `l1`: Address line 1
- `l2`: Address line 2
- `box`: Post box number
- `cy`: City
- `st`: State
- `code`: Postal code
- `c`: Country

Map your LDAP postal address attributes to these fields. The following excerpt demonstrates how to map to the fields of the `ORAPOSTAL` scheme:

```
<Field>
  <source_attribute>l1=street?cy=l?st=st?code=postalcode?c=c</source_attribute>
  <target_attribute>BUSINESS</target_attribute>
  <target_extended_attribute>ORAPOSTAL</target_extended_attribute>
  <target_attribute_type>ADDRESS</target_attribute_type>
</Field>
```

In this excerpt, LDAP postal address attributes will be mapped to `ORAPOSTAL` fields as follows:

- street maps to l1
- l maps to cy
- st maps to st
- postalcode maps to code
- c maps to c

The entire postal address will be mapped to an address user account field (as specified by <target_attribute_type>) of type business (as specified by <target_attribute>).

Mapping Active Directory Proxy Addresses An Active Directory user's entry contains an attribute named proxyAddresses that holds all the e-mail addresses of a particular user.

The following is an example of a proxyAddresses attribute:

```
proxyAddresses: smtp:rholmes@example.com ;
SMTP:Robert.Holmes@example.com ; MBX:0 ; X400:c=US ; p=Example ;
s=Holmes ; g=Robert ; RFAX: Holmes, Robert @
```

Consider the following example:

```
<Field>
  <source_attribute>proxyAddresses</source_attribute>
  <target_attribute>PROXY</target_attribute>
  <target_extended_attribute>MAILTO</target_extended_attribute>
  <target_attribute_type>ADDRESS</target_attribute_type>
  <source_special_handling>PROXY</source_special_handling>
</Field>
```

If the <source_special_handling> element is omitted, then UDS synchronization will map the Active Directory proxyAddresses value of a user to the Oracle Beehive address type PROXY with the scheme MAILTO. This creates the following:

```
proxy1:mailto:smtp:rholmes@example.com
proxy2:mailto:smtp:robert.holmes@example.com
...
proxy8:mailto:RFAX: Holmes, Robert
```

If the <source_special_handling> element is included, then Oracle Beehive will only synchronize values that start with smtp: and remove the text smtp:. As a result, the actual values in Oracle Beehive become as follows:

```
proxy1:mailto:rholmes@example.com
proxy2:mailto:robert.holmes@example.com
```

Consequently, the <source_special_handling> element properly formats Active Directory e-mail addresses for Oracle Beehive and ignores values that start with MBX, RFAX, and other protocols not used by Oracle Beehive. This kind of mapping enables you to incorporate your legacy e-mail addresses into Oracle Beehive by synchronizing it with Active Directory's method of inbound mail lookup resolution.

In addition, this kind of mapping maps the user's DEFAULT_ADDRESS_BY_SCHEME element for scheme MAILTO: to be that of the value of the proxyAddresses value marked with the uppercase SMTP:. For example, if the proxyAddresses value in Active Directory was:
smtp:rholmes@example.com;SMTP:robert.holmes@example.com;smtp:rob.holmes@example.com, then the default_address_by_scheme for scheme

MAILTO: (that is, the default e-mail address used in clients such as Webmail) will be robert.holmes@example.com as it was prefixed by the text SMTP:.

Mapping primary_principal Attribute Ensure that you map primary_principal to the attribute your LDAP server is configured for authentication. Modify and map primary_principal to sAMAccountName for Active Directory or uid (by default) for Oracle Internet Directory, otherwise authentication will fail.

Consider the following example for Active Directory:

```
<Field>
<source_attribute>sAMAccountName</source_attribute>
<target_attribute>PRIMARY</target_attribute>
<target_attribute_type>PRINCIPAL</target_attribute_type>
</Field>
```

Mapping external_inbox Attribute You may synchronize the value of the UDS attribute external_inbox with an arbitrary attribute in your LDAP directory. If the value of external_inbox is true, then that user's e-mail messages will be routed somewhere other than his or her Personal Workspace Inbox.

Suppose you have created the attribute orclBeehiveUserStatus in your LDAP directory. You have configured orclBeehiveUserStatus to have a value of either external-inbox or local-inbox. You want external_inbox to be true if orclBeehiveUserStatus has a value of external-inbox and external_inbox to be false if orclBeehiveUserStatus has a value of local-inbox or smtp. You would use the following XML excerpt to represent this mapping:

```
<Field>
  <source_attribute>orclBeehiveUserStatus</source_attribute>
  <target_attribute>is_external_inbox</target_attribute>
  <target_attribute_type>ATTRIBUTE</target_attribute_type>
  <source_target_value_mapping>
    <value_mapping>
      <source_field_value>external-inbox</source_field_value>
      <target_field_value>true</target_field_value>
    </value_mapping>

    <value_mapping>
      <source_field_value>local-inbox</source_field_value>
      <target_field_value>false</target_field_value>
    </value_mapping>

  </source_target_value_mapping>
</Field>
```

> **Note:** The <source_target_value_mapping> mechanism can also be used for mapping any LDAP attribute to any other Beehive attribute, except Beehive attributes such as LOCALE and TIMEZONE.

If is_external_inbox is false for a particular user (which is the default), any e-mail message addressed to that user that originates from Oracle Beehive's inbound virtual mail server (VMS) will be delivered to that user's inbox. Conversely, if is_external_inbox is true, then any e-mail messages addressed to that user will be redirected to Oracle Beehive's outbound VMS. See "Managing Oracle Beehive E-mail" in *Oracle Beehive Administrator's Guide* for more information about Oracle Beehive

VMSes. See "Managing and Provisioning Oracle Beehive Users" for more information about `is_external_inbox` and other user properties.

Step G: Adding Profile to Oracle Beehive

A default directory profile is the one used by both authentication, and UDS. This profile stores and reads LDAP server information from the site. For non-default directory profiles, the LdapServer object is stored within the profile itself. Most customers will only require a single profile.

For a default directory profile, the value of the `<profile_flag>` element (found in the `<profile>` element) is `DEFAULT`. For a non-default profile, the value is `NON_DEFAULT`.

Follow these steps to add a non-default directory profile:

1. Add the profile with the following `beectl` command:

   ```
   beectl add_directory_profile --file oidprofile_template.xml
   ```

   The utility may return an error similar to the following:

   ```
   Failed to add directory profiles. See the log file.
   ```

   The log file is `ORACLE_HOME/logs`.

2. Activate the configuration:

   ```
   beectl activate_configuration
   ```

To add a default directory profile, follow these steps:

1. Add the profile with the following `beectl` command:

   ```
   beectl add_directory_profile --file oidprofile_template.xml
   ```

2. Modify the `AuthStoreType` property of the Authentication Service to `ldap` with the `beectl modify_property` command. Refer to "Configuring Authentication Service to Use LDAP Server" for more information. (This is required so that the `beectl modify_local_configuration_files` works properly after you call `beectl add_directory_profile` to add the default profile.)

3. Activate the configuration and commit changes:

   ```
   beectl activate_configuration
   beectl modify_local_configuration_files
   ```

Changing Directory Profile to Default Profile or Non-Default Profile

> **Note:** An ENABLEd profile will synchronize, whereas a DISABLEd profile will not synchronize. Only a DEFAULT profile is used for authentication and not a NON-DEFAULT profile.

.Follow these steps to change a directory profile to a default or non-default profile:

Changing Default Profile to Non-Default Profile

1. Download the existing LDAP mapping profile with the `beectl list_directory_profiles` command. The following example downloads the existing LDAP profile and saves it in the file `existing_profile.xml`:

   ```
   beectl list_directory_profiles --file existing_profile.xml
   ```

These steps will use existing_profile.xml as the file name of your existing LDAP mapping profile.

2. Edit the file existing_profile.xml and change the value of <profile_flag> from DEFAULT to NON_DEFAULT.
3. Upload the LDAP mapping profile with the beectl modify_directory_profile command:

```
beectl modify_directory_profile --file existing_profile.xml
```

4. Modify the AuthStoreType property of the Authentication Service from ldap to db with the beectl modify_property command:

```
beectl modify_property --component _AuthenticationService --name AuthStoreType
--value db
```

5. Activate the configuration and commit changes:

```
beectl activate_configuration
beectl modify_local_configuration_files
```

Changing Non-Default Profile to Default

1. Download the existing LDAP mapping profile with the beectl list_directory_profiles command. The following example downloads the existing LDAP profile and saves it in the file existing_profile.xml:

```
beectl list_directory_profiles --file existing_profile.xml
```

These steps will use existing_profile.xml as the file name of your existing LDAP mapping profile.

2. Edit the file existing_profile.xml and change the value of <profile_flag> from NON_DEFAULT to DEFAULT.
3. Upload the LDAP mapping profile with the beectl modify_directory_profile command:

```
beectl modify_directory_profile --file existing_profile.xml
```

4. Modify the AuthStoreType property of the Authentication Service from db to ldap with the beectl modify_property command:

```
beectl modify_property --component _AuthenticationService --name AuthStoreType
--value ldap
```

5. Activate the configuration:

```
beectl activate_configuration
```

Directory Profile Validation

When you add a directory profile, Oracle Beehive validates the following in the XML file:

1. LDAP credentials
2. <poll_interval>, <profile_flag>, and <directory_type>
3. The existence of <user_search_base> and <group_search_base> in your LDAP server

4. For <scope_type_map> and <membership_type_map>, the following are validated:
   a. <source_field_type> (either DN or ATTRIBUTE)
   b. <source_field_value>: If <source_file_type> is DN (if <source_field_type> is ATTRIBUTE, then this validation is skipped)
   c. Values defined in <identifier> are validated for their existence; if you have specified an invalid enterprise or organization identifier, then an appropriate error message is returned
5. For <user_type_map>, the following are validated:
   a. <source_field_type> (either DN or ATTRIBUTE)
   b. <source_field_value>: If <source_file_type> is DN (if <source_field_type> is ATTRIBUTE, then this validation is skipped)
   c. <user_type> (either ENTERPRISE_USER, EXTENDED_ENTERPRISE_USER, or EXTERNAL_PERSON)
6. For <group_type_map> the following are validated:
   a. <source_field_type> (either DN or ATTRIBUTE)
   b. <source_field_value>: If <source_file_type> is DN (if <source_field_type> is ATTRIBUTE, then this validation is skipped)
   c. <group_type> (only valid value is STATIC_GROUP)
7. For <directory_attribute_map> the following are validated:
   a. <target_attribute>: If <source_object> is ENTERPRISE_USER or EXTENDED_ENTERPRISE_USER, attribute mappings for the PRINCIPAL and FAMILYNAME target attributes must exist. If <source_object> is STATIC_GROUP, attribute mappings for the NAME target attribute must exist. If <source_object> is EXTERNAL_PERSON, attribute mappings for the FAMILYNAME target attribute must exist.
   b. <target_attribute_type>
   c. <target_extended_attribute>

Step 2: Loading Users and Groups

The following steps describe how to load all users and groups from the LDAP server to UDS:

1. Generate an XML file from the LDAP server based on the mapping profile you loaded into Oracle Beehive in the previous step. The following command will create a file named UsersFromLDAP.xml in your home directory based on the profile named oidldapdirectoryprofile:

```
beectl download_ldap_user_data
  --file UsersFromLDAP.xml
  --profile oidldapdirectoryprofile
```

Note: You do not need administrator privileges to the LDAP server in order to extract data from it. Therefore a normal user may run the `beectl download_ldap_user_data` command.

However, LDAP directories may impose a search limit for non-administrator users.

For example, if your OID server has 500 records, OID may impose a search limit of 200 records for non-administrator users. If you are a normal user, the `beectl download_ldap_user_data` command will return only 200 records. As a result, you will not be able to synchronize all your users.

Check your LDAP server documentation for maximum returned result limitations and how to manage them.

---

2. In a text editor, open the file you generated (`UsersFromLDAP.xml`), and check for the following:
   - `primary_principal` is mapped to the attribute your LDAP server is configured for authentication, for example, `sAMAccountName` for Active Directory or `uid` (by default) for Oracle Internet Directory, otherwise authentication will fail.
   - Enterprise and organization identifiers are correct for your Oracle Beehive deployment and all the organizations already exist in Oracle Beehive
   - The element `familyname` is defined and contains a value for each user

   ---

   Notes: If you receive many errors or inconsistencies in the generated XML file, then delete it, correct the LDAP mapping profile, and recreate the generated XML file.

   ---

3. Add the users in the generated XML file to Oracle Beehive with the `beectl add_user` command:

```
beectl add_user --file UsersFromLDAP.xml
  --ldapbootstrap
```

Note:

- If users are created using the `beectl add_user` command with the `file` option by running the command multiple times with a different batch of users in each run, then the manager or assistant attribute of the users is set in Oracle Beehive only if one of the following conditions is met:
  - The manager or assistant is already present in Oracle Beehive.
  - The manager or assistant is being created in the same XML as the user.

  If the manager or assistant is created in a different XML at a later time, then it is not set by the `add_user` command. To resolve this issue, you must reconcile users created in multiple batches by running the `validate_directory_command` with the `reference` option.
- The `-ldapbootstrap` parameter indicates to Oracle Beehive that the users to be added are mastered/synched from the LDAP server and will authenticate to the LDAP server.

4. Ensure that the users were added successfully with the `beectl list_users` command:

```
beectl list_users
```

Step 3: Enabling Synchronization

Enable the synchronization profile with the following commands: These commands enable a profile named `oidldapdirectoryprofile`:

```
beectl list_properties --component oidldapdirectoryprofile
beectl modify_property --component oidldapdirectoryprofile
                       --name ProfileState --value ENABLE
beectl activate_configuration
```

Note: Your users will not be able to log in yet, even though they are provisioned. Your users will able to log in once you have completed the step described in "Configuring Authentication Service to Use LDAP Server".

Controlling How Often UDS Contacts the LDAP Server

By default, UDS contacts the LDAP server's change log every 30 seconds for updates. You may change this interval in either of the following ways:

- In your LDAP mapping profile, change the value in the `<poll_interval>` tag.

  The following is an excerpt from an LDAP mapping profile with an interval set to fifteen seconds:

```
<profile>
  <profile_name>oidldapdirectoryprofile</profile_name>
  <poll_interval>15</poll_interval>
  <profile_state>DISABLE</profile_state>
  <profile_flag>DEFAULT</profile_flag>
```

```
  <directory_type>ORACLE_INTERNET_DIRECTORY</directory_type>
  <ldap_server>
  <!-- ... -->
</profile>
```

If you make any changes to your LDAP mapping profile, then use the command `modify_directory_profile` to update the existing profile.

Notes: The value of the `<profile_flag>` element may be `DEFAULT` or `NON_DEFAULT`.

A `DEFAULT` profile is the one used by both authentication and UDS. This profile stores and reads LDAP server information from the site.

For `NON_DEFAULT` profiles, the `LdapServer` object is stored within the profile itself.

When adding a default profile, modify the `AuthStoreType` property of the Authentication Service to `ldap` with the `beectl modify_property` command. Refer to "Configuring Authentication Service to Use LDAP Server" and "Changing Directory Profile to Default Profile or Non-Default Profile" for more information. (This is required so that the `beectl modify_local_configuration_files` works properly after you call `beectl add_directory_profile` to add the default profile.)

- Use the `beectl modify_property` command. The following commands set the value of the property `PollInterval` to fifteen seconds:

```
beectl list_properties --component oidldapdirectoryprofile
beectl modify_property
  --component oidldapdirectoryprofile
  --name PollInterval
  --value 15
beectl activate_configuration
```

Note: If the LDAP server's change log is cleaned up or purged more frequently than the UDS update frequency, then the data might be lost.

Retrieving Information About the LDAP Server

When you create a profile, Oracle Beehive creates an `LdapServer` configuration object. Use the `beectl list_properties` to get information about it:

```
beectl list_properties --component _CURRENT_SITE:LdapServer
-------------------------------------------------------------------------
| Property Name                  | Property Value                       |
-------------------------------------------------------------------------
| LdapServerHostName             | ldapserver.com                       |
| LdapServerPort                 | 389                                  |
| LdapServerSslPort              | 636                                  |
| SslEnabled                     | false                                |
| LdapServerUser                 | cn=orcladmin                         |
| LdapServerPassword             | [Protected Value]                    |
| SSLMode                        | 0                                    |
| UserSearchBase                 | cn=users,dc=us,dc=oracle,dc=com      |
| UserSearchBaseForSync          |                                      |
```

```
| GroupSearchBase                  | cn=groups,dc=us,dc=oracle,dc=com |
| UserObjectClass                  |                                  |
| GroupObjectClass                 |                                  |
| PrimaryAuthenticationAttribute   | uid                              |
| PrimaryAuthenticationCredential  | not applicable                   |
| ProtocolAuthenticationAttribute  | not applicable                   |
| ProtocolAuthenticationCredential | not applicable                   |
| VoiceAuthenticationAttribute     | not applicable                   |
| VoiceAuthenticationCredential    | not applicable                   |
| DirectoryType                    | ORACLE_INTERNET_DIRECTORY        |
| Alias                            | OracleInternetDirectory          |
-----------------------------------------------------------------------
```

Notes: You may have multiple `LdapServer` objects in an Oracle Beehive deployment if you have configured more than one LDAP mapping profile. The Authentication Service uses the `LdapServer` object set at the site level, which is created by the UDS mapping profile.

The site level `LdapServer` object may not have all required properties for authentication.

The following table describes the properties of the `LdapServer` object:

Table 4–2 LdapServer Properties

Property Name	Required	Description
LdapServerHostName	Required	LDAP server host name
LdapServerPort	Required	LDAP server port for non-SSL connections
LdapServerSslPort	Required	LDAP server port for SSL connections
SslEnabled	Required	If set to true, only SSL connections are used
LdapServerUser	Required	LDAP server user with bind and search privileges. This user must be able to look up attributes for all LDAP users provisioned to use Oracle Beehive
LdapServerPassword	Required	Password for LdapServerUser
SSLMode	Not used	Ignore this property
UserSearchBase	Required	User search base dn. The search scope is always subtree (recursive search).
GroupSearchBase	Required	Group search base dn. Search Scope is always subtree (recursive search).
UserObjectClass	Optional	Name of the user object class in the directory. This attribute is used to construct a search filter for the users. If this value is not specified, a default value is used, described in "Default UserObjectClass and GroupObjectClass Values".

Table 4–2 (Cont.) LdapServer Properties

Property Name	Required	Description
GroupObjectClass	Optional	Name of the group object class in the directory. This attribute is used to construct a search filter for the groups. If this value is not specified, a default value is used, described in "Default UserObjectClass and GroupObjectClass Values".
PrimaryAuthenticationAttribute	Required	The name of the attribute the LDAP server uses to authenticate a user. For example, set this to `uid` for Oracle Internet Directory, or `sAMAccountName` for Active Directory.
PrimaryAuthenticationCredential	Not used	Ignore this property
ProtocolAuthenticationAttribute	Not used	Ignore this property
ProtocolAuthenticationCredential	Not used	Ignore this property
VoiceAuthenticationAttribute	Not used	Ignore this property
VoiceAuthenticationCredential	Not used	Ignore this property

DirectoryType	Required	Indicates which specific LDAP directory is being configured. Valid values are the following:
		`ORACLE_INTERNET_DIRECTORY`
		`MICROSOFT_ACTIVE_DIRECTORY`
		`IBM_TIVOLI_DIRECTORY`
		`SUN_ONE_DIRECTORY`
		`OPENLDAP_DIRECTORY`

Alias	Optional	Alias for this LdapServer configuration object. Use this alias to refer to this LdapServer configuration object from `beectl`.

Default and Custom Object Classes for Users and Groups

UDS uses the values specified in the properties `UserObjectClass` and `GroupObjectClass` to determine whether an entity in your LDAP directory is a user or a group, respectively. The default schema of your LDAP directory uses a particular object class for users and another for groups. Oracle Beehive automatically sets `UserObjectClass` and `GroupObjectClass` to the name of the default object class used by the users and groups of your LDAP directory. Refer to "Default UserObjectClass and GroupObjectClass Values" for these default values.

However, if you are using a custom schema in your LDAP directory and have defined your users and groups with object classes other than the default ones for your directory, follow the steps described in "Specifying Non-Default User and Group Object Classes"

Default UserObjectClass and GroupObjectClass Values Depending on the LDAP directory type, the values of `UserObjectClass` and `GroupObjectClass` are set to one of the values specified in the following table, if those properties have not been explicitly set:

Table 4–3 Default UserObjectClass and GroupObjectClass Values

Directory/Property	UserObjectClass	GroupObjectClass
IBM_TIVOLI_DIRECTORY	inetOrgPerson	groupOfNames
MICROSOFT_ACTIVE_DIRECTORY	user	group
ORACLE_INTERNET_DIRECTORY	orclUserV2	orclGroup
SUN_ONE_DIRECTORY	inetOrgPerson	groupOfUniqueNames
OPENLDAP_DIRECTORY	inetOrgPerson	groupOfNames

Specifying Non-Default User and Group Object Classes Modify your LDAP mapping profile as described in "Modifying Directory Profile". In your LDAP mapping profile, include the elements `<user_objectclass>` and `<group_objectclass>` in the `<ldapserver>` section of your LDAP mapping profile.

For example, if the schema of your LDAP directory defines its users with the object class `BeehivePerson` and its groups with `BeehiveGroup`, you would modify your LDAP mapping profile as follows:

```
<ldap_server>
  <!-- ... -->
  <user_objectclass>BeehivePerson</user_objectclass>
  <group_objectclass>BeehiveGroup</group_objectclass>
</ldap_server>
```

Administering Your External User Directory and Oracle Beehive Integration

This section describes how to administer your external user directory and Oracle Beehive integration. This section includes the following topics:

- Configuring Authentication Service to Use LDAP Server
- Configuring Digest Authentication
- Changing LDAP Administrator's Password
- Oracle Internet Directory Considerations
- Modifying Directory Profile
- Active Directory Considerations
- Troubleshooting General LDAP Synchronization Issues

Configuring Authentication Service to Use LDAP Server

The following steps describe how to configure the Authentication Service so that it uses your LDAP server. These steps assume that you have already enabled a synchronization profile for your LDAP server.

Notes: If you are using Active Directory as your LDAP server, see "Active Directory Considerations" before proceeding with these steps.

Before configuring the Authentication Service to use Active Directory, you must ensure that host names returned in an LDAP referral by Active Directory can be resolved by the Domain Name System (DNS) on the host on which you installed Oracle Beehive.

If you installed Oracle Beehive Integration for Zimbra in a separate Oracle home other than Oracle Beehive, then after configuring the Authentication Service to use your LDAP server, run the `beectl modify_local_configuration_files` command on both your Oracle Beehive Integration for Zimbra and Oracle Beehive homes.

1. Modify the `AuthStoreType` property of the Authentication Service to `ldap` with the `beectl modify_property` command:

```
beectl list_properties --component _AuthenticationService
-------------------------------------------------------------------------
| Property Name                  | Property Value                       |
-------------------------------------------------------------------------
| Alias                          | _AuthenticationService               |
-------------------------------------------------------------------------
| AuthStoreType                  | db                                   |
-------------------------------------------------------------------------
.........................................................................
.........................................................................
.........................................................................
beectl modify_property --component _AuthenticationService
                       --name AuthStoreType --value ldap

beectl activate_configuration

beectl modify_local_configuration_files
```

Note: The `beectl modify_local_configuration_files` command will ask you to run this command on all your other instances. **Do not run this command on all your other instances at this time.** For each instance, make your desired changes to the `AuthStoreType` property and run `beectl activate_configuration` before running the `beectl modify_local_configuration_files` command.

2. To test the Authentication Service, log in with any user:

```
beectl login
  --authuser newuser
  --authpassword <Password of newuser, obfuscated. To use non-obfuscated
  passwords, run beectl in shell mode>
User newuser is successfully authenticated and logged in.
```

To test connectivity with the LDAP server use either the commands `ldapbind` or `ldapsearch`. Refer to the documentation of your LDAP server for more information about these commands.

3. Grant administration privileges to another LDAP user or group in your system.

Note: Once you have changed `AuthStoreType` to `ldap`, you will not be able to login with the `beeadmin` account to administer your Oracle Beehive deployment.

Therefore, you should assign administration privileges to another LDAP user or group or assign the appropriate privileges to particular users or groups depending on the security policy of your site.

For more information about the `beeadmin` account, see the section "About Special and System-Reserved Accounts" in "Managing and Provisioning Oracle Beehive Users" in *Oracle Beehive Administrator's Guide*.

Configuring Digest Authentication

Digest authentication is an authentication method that involves using some known secrets (or passwords) from both the client and server to calculate a hash value. This hash value is transmitted instead of the actual secret (or password). One of the major benefits of digest authentication is that the password is not exposed while being transmitted.

To use digest authentication with a particular LDAP directory, the directory must be able to do one of the following:

- Store the user password in clear text or reversible encrypted form
- Store an A1 hash value of the password. An A1 hash value is an intermediate value used for the calculation of the authentication methods HTTP digest and SASL digest-MD5. The A1 hash value is created from a user's password, principal name (`userid`) and realm.

Configuring digest authentication, using an LDAP directory as the authentication repository, involves the following steps:

- Step A: Configure SSL for Oracle Beehive and LDAP Directory
- Step B: Determine Digest Mechanism Depending on LDAP Directory
- Step C: Configure Oracle Beehive

For OpenLDAP Directory, see "Configuring Digest Authentication for OpenLDAP Directory".

Step A: Configure SSL for Oracle Beehive and LDAP Directory

Digest authentication requires that your Oracle Beehive instance and the corresponding LDAP server to be configured in SSL mode. See "Configuring SSL" in the *Oracle Beehive Installation Guide for Microsoft Windows* to configure your Oracle Beehive instance for SSL. Refer to the documentation of your LDAP directory to configure it in SSL mode.

Note: To log in to an LDAP directory in SSL mode, the `jps-config.xml` file must have `ldaps` instead of `ldap` in the URL value in both the following locations:

```
$BKPR_HOME/j2ee/home/application-deployments/javasso
<property name="oracle.security.jaas.ldap.provider.url"
value="ldaps://user1.us.oracle.com:6060"/>

$BKPR_HOME/j2ee/home/application-deployments/beekeeper
<property name="oracle.security.jaas.ldap.provider.url"
value="ldaps://user1.us.oracle.com:6060"/>
```

Step B: Determine Digest Mechanism Depending on LDAP Directory

Determine the digest mechanism Oracle Beehive will use by referring to the following table. The digest mechanism used depends on the availability of a clear-text user password, reversible encrypted password, or a secure A1 hash value of the password. You may have to configure your LDAP directory to use certain digest mechanisms.

In the following table, each cell in the **Digest Mechanism** column specifies a digest mechanism and a list of password formats; the specified digest mechanism requires that your LDAP server is configured to store only **one** of these password formats.

Table 4–4 Supported Digest Authentication Mechanisms

Digest Mechanism	Active Directory	IBM Tivoli Directory	Oracle Internet Directory	Sun One Directory
SASL CRAM-MD5 ■ Plain text ■ Reversible encryption	Not supported; Active directory does not allow any passwords to be read; the password attribute is write-only	Supported	Supported	Supported
SASL Digest-MD5 ■ Plain text ■ Reversible encryption ■ A1 hash value	Supported; extend the schema to store the A1 hash value	Supported	Supported	Supported; extend the schema to store the A1 hash value
HTTP Digest ■ Plain text ■ Reversible encryption ■ A1 hash value	Supported; extend the schema to store the A1 hash value	Supported	Supported	Supported; extend the schema to store the A1 hash value
SyncML v1.0 Digest ■ Plain text ■ Reversible encryption	Not supported; Active directory does not allow any passwords to be read; the password attribute is write-only	Supported	Supported	Supported
Sync ML v1.1 Digest ■ Plain text ■ Reversible encryption	Not supported; Active directory does not allow any passwords to be read; the password attribute is write-only	Supported	Supported	Supported

Step C: Configure Oracle Beehive

These steps assume that you have already configured Oracle Beehive to authenticate with an LDAP directory.

1. Set the properties `UseSecureHash` and `AuthenticationRealm` in the `AuthenticationService` component:

   - `UseSecureHash`: Set this property to `true` to use the A1 hash value. Set this property to `false` to use the plain or reversible encrypted password. This depends on which digest authentication method you are going to use. Refer to Table 4–4, " Supported Digest Authentication Mechanisms".
   - `AuthenticationRealm`: Default authentication realm. This value is returned to clients when digest authentication is initiated. For example, HTTP and SASL digest authentication requires that the realm value to be sent with the authentication challenge.

   The following `beectl` commands set `UseSecureHash` to `true` and `AuthenticationRealm` to `myrealm@example.com`:

```
beectl modify_property
  --component _AuthenticationService
  --name UseSecureHash
  --value true

beectl modify_property
  --component _AuthenticationService
  --name AuthenticationRealm
  --value myrealm@example.com
```

2. Set the `DigestAuthenticationAttribute` property in the `LdapServer` configuration object. See "Retrieving Information About the LDAP Server" for more information about this object.

   The `DigestAuthentication` property specifies which attributes from the user object (in the LDAP directory) are required for the digest authentication. This property can have multiple values. The format of the property value is *<Mechanism Type>*:*<Attribute Name>*. The following table lists the possible values for *<Mechanism Type>* and the type of attribute with which it is associated:

Table 4–5 Valid Digest Mechanism Type Values

***<Mechanism Type>* Value**	**Type of Attribute to Specify**
DEFAULT	Attribute name for reversible encrypted or plain password
SASL.DIGEST_MD5	Attribute name for SASL digest authentication
HTTP.DIGEST	Attribute name for HTTP digest authentication

   For example, suppose you are using Oracle Internet Directory as your LDAP server. Oracle Internet Directory stores the user password in reversible encrypted format in the attribute `orclrevpwd`. If you have also specified that Oracle Internet Directory stores the A1 hash value in the attribute `authpassword;beehive`, set `DigestAuthenticationAttribute` as follows:

```
beectl modify_property
  --component <ID of LdapServer object>
  --name DigestAuthenticationAttribute
  --value DEFAULT:orclrevpwd
  --value SASL.DIGEST_MD5:authpassword;beehive
  --value HTTP.DIGEST:authpassword;beehive
```

> **Note:** To compute and set the A1 hash value, refer to the documentation of your LDAP server. You may also use a third-party tool to create this value or create this value yourself by using the information in RFC 2617, *HTTP Authentication: Basic and Digest Access Authentication*.

3. Activate the configuration and commit changes.

```
beectl activate_configuration
beectl modify_local_configuration_files
```

Configuring Digest Authentication for OpenLDAP Directory

If you are using OpenLDAP Directory, then follow these steps to configure digest authentication:

1. Ensure your directory type is `OPENLDAP_DIRECTORY`, your Oracle Beehive instance is configured for SSL, and OpenLDAP Directory is in SSL mode.
2. Add the value `OPENLDAP_DIRECTORY:oracle.ocs.csi.authentication.handlers.impl.jaas.callback.impl.OcsLdapPasswordAccessor` to the list of values in the `PwdAccessorPlugin` property of the `_AuthenticationService` component:

```
beectl append_value
  --component _AuthenticationService
  --name PwdAccessorPlugin
  --value OPENLDAP_DIRECTORY:oracle.ocs.csi.authentication.handlers.
           impl.jaas.callback.impl.OcsLdapPasswordAccessor
```

3. Ensure that you are storing user passwords as clear text (plain) in OpenLDAP. Simply use the `ldapmodify` command to add or modify a user's password.
4. Activate the configuration and commit changes.

```
beectl activate_configuration
beectl modify_local_configuration_files
```

Changing LDAP Administrator's Password

To change the administrator's password of an LDAP server synchronized with Oracle Beehive, follow these steps:

1. Stop Oracle Beehive with the `beectl stop --all` command.
2. Change the password of the LDAP administrator in your LDAP server.
3. Start Oracle Beehive with the `beectl start` command.
4. Change the password for the LDAP administrator in Oracle Beehive with the following command (obfuscate the password with the `beectl obfuscate` command.)

```
beectl modify_secure_property
  --component <LdapServer of the profile>
  --name LdapServerPassword
  --value <Password of LDAP administrator, obfuscated>
  --obfuscated
```

5. Run the `beectl modify_local_configuration_files` command.

Oracle Internet Directory Considerations

This section covers the following topics:

- Synchronizing with Directory Replication Group
- Migrating Oracle Internet Directory from One Server to Another
- Troubleshooting Synchronization between Oracle Beehive and Oracle Internet Directory

Synchronizing with Directory Replication Group

A Directory Replication Group (DRG) consists of the directory servers that participate in the replication of a given naming context. If you have synchronized Oracle Beehive with an Oracle Internet Directory server that belongs to a multimaster DRG, then ensure that the attribute `orclDIPRepository` is set to `true`.

This ensures changes made to any server in the multimaster DRG are synchronized with Oracle Beehive.

For more information about directory replication groups, see Chapter 29, "Oracle Internet Directory Replication Concepts" in *Oracle Internet Directory Replication Concepts*.

Migrating Oracle Internet Directory from One Server to Another

If you migrate an Oracle Internet Directory server (that is `synchronized` with Oracle Beehive) to another Oracle Internet Directory server, then modify the `LdapServer` property in the `_CURRENT_SITE` component with the name of the new Oracle Internet Directory server:

```
beectl list_properties --component _CURRENT_SITE

...

| EventListenerDatabase    |                                      |
| SearchDatabase           |                                      |
| BusinessDatabase         |                                      |
| VirusScanEngineCluster   |                                      |
| SiteId                   | 17378                                |
| LdapServer               | OLD_OID_server_example.com           |
| LanguagePack             | byte array of size 656902            |
| ClusteringEnabled        | true                                 |
| DiagnosabilityProperties | 1f90e-7b46-427c-a6ef-636bcbb88f89 |
| DebugProperties          | 19804e92-9028-49b3-af36-be4ac4abb4f5 |
| BtiGlobalConfiguration   |                                      |
| Name                     | R1                                   |

beectl modify_property --component _CURRENT_SITE
  --name LdapServer
  --value <new Oracle Internet Directory server>
beectl modify_change_number
  --profile <name of your LDAP profile>
  --number <Changelog number of your new Oracle Internet Directory server>
beectl activate_configuration
beectl modify_local_configuration_files
```

Troubleshooting Synchronization between Oracle Beehive and Oracle Internet Directory

- Check the files `oidldapd.log` and `oidrepld.log` from Oracle Internet Directory.
- To retrieve change log information, the Oracle Directory Integration Server Control tool (`odisrv`) must be up and running. Refer to Chapter 2, "odisrv" in *Oracle Identity Management User Reference*.
- Ensure that the `orcldiprepository` parameter is set to true. Refer to Chapter 9, "Oracle Identity Management Attribute Reference in *Oracle Identity Management User Reference*.
- Ensure that the value of `changeid` from the `bee_data.uds_sync_profile` table is updated with the `chg_no` value from `ods.ods_chg_log` (which is a table from the Oracle Internet Directory schema). For more information about this table, see the section "LDAP-Based Replication" in Chapter 29, "Oracle Internet Directory Replication Concepts" in *Oracle Internet Directory Administrator's Guide*.

  Note: Take the `chg_no` value from Oracle Internet Directory used with the cloned system and not the information from the LDAP server used by the production system.

- In Oracle Internet Directory, ensure that the attributes `krbaPrincipalName` and `orclUserApplnProvStatus` exist, otherwise create them. The `bulkload` bulk management tool might fail if these aren't defined.

  For more information about Oracle Internet Directory attributes, see Chapter 9, "Oracle Identity Management Attribute Reference in *Oracle Identity Management User Reference*.

  For more information about `orclUserApplnProvStatus`, see the section "Provisioning Status in Oracle Internet Directory" in Chapter 12, "Oracle Directory Integration Platform Service Concepts" in *Oracle Identity Management Integration Guide*.

  For more information about bulkload, see the section "bulkload" in Chapter 9, "Using Bulk Tools" in *Oracle Internet Directory Administrator's Guide*.
- For more information about troubleshooting Oracle Internet Database, see Appendix J, "Troubleshooting Oracle Internet Directory" in *Oracle Internet Directory Administrator's Guide*.

Modifying Directory Profile

To modify a directory profile, follow these general steps:

1. Make changes to the existing LDAP mapping profile, then modify the directory profile with the command `beectl modify_directory_profile --file <file name of modified LDAP mapping profile>`.

   Alternatively, you may modify the directory profile through Oracle Beekeeper or with the `beectl modify_property` command.
2. Activate the modified profile with the command `beectl activate_configuration`.
3. Run the command `beectl validate_directory_entry` to validate all your users and groups already synchronized with UDS:

```
beectl validate_directory_entry --all --profile exampleProfile --delete
```

After modifying the directory profile, you do not have to restart the LDAP server.

Notes: You may not delete a profile with which you have already loaded users and groups from your LDAP server to UDS.

Do not change the name of the profile. Oracle Beehive treats a renamed profile as a new profile.

For `Validate_directory_entry` to reconcile with `--all`, `--all_users`, and `--all_groups`, the attributes in the rule map of directory profile should be indexed on the LDAP.

Active Directory Considerations

This section covers the following topics:

- Active Directory Administrator's Account
- LDAP Referrals

Active Directory Administrator's Account

Ensure that the user that you specified as the Active Directory administrator is not used by other applications. This may affect Oracle Beehive authentication.

In particular, if you notice a significant number of invalid credential errors in your Active Directory log files, ensure that the Active Directory administrator's account is not locked out or disabled.

Note: You must have the Active Directory profile validation check permissions to ensure that the user account is successfully deleted in the UDS when deletions occur in Active Directory.

LDAP Referrals

During authentication, the Oracle Beehive Authentication Service may request Active Directory to perform an operation, such as a search. When Active Directory has referrals enabled, instead of the search results, Active Directory may respond with an LDAP referral. This referral may point to an Active Directory instance on another host.

Ensure that the host names returned in the referral can be resolved by the DNS on the host on which you installed Oracle Beehive.

For more information about LDAP referrals, see the Active Directory documentation.

Troubleshooting General LDAP Synchronization Issues

Call the command `beectl validate_directory_entry` to reconcile any LDAP directory entries that are not synchronized with Oracle Beehive.

Note: To retrieve the LDAP DN of a user, use the `ldapsearch` command or other similar utility from your LDAP directory. You cannot use `beectl validate_directory_entry` to retrieve the DN of a synchronized user. In particular, if you change the DN of a synchronized user, you must use the new DN to validate that user with `beectl validate_directory_entry`. Oracle Beehive does not synchronize a user's DN.

Oracle Beehive uniquely identifies users (and all other entities) by their Beehive Object Distinguished Name (BODN), which has no relation to a user's LDAP DN. When you run the command `beectl list_users`, a user's BODN is specified by User Identifier.

5

Integrating Oracle Information Rights Management (Oracle IRM) with Oracle Beehive

This module describes how to integrate Oracle Information Rights Management (Oracle IRM) 11*g* with Oracle Beehive. This module assumes the user has a strong understanding of the concepts and procedures related to managing Oracle IRM and developing applications for it.

This module includes the following topics:

- Overview of Oracle Information Rights Management (Oracle IRM)
- Overview of Integrating Oracle IRM with Oracle Beehive
- Prerequisites for Integrating Oracle IRM with Oracle Beehive
- Procedures for Integrating Oracle Beehive with Oracle IRM
- Using Documents Sealed by Oracle IRM

For detailed information about Oracle IRM, see the following guides:

- *Oracle Fusion Middleware Administrator's Guide for Oracle Information Rights Management Server*
- *Oracle Fusion Middleware Developer's Guide for Oracle Information Rights Management Server*
- *Oracle Fusion Middleware User's Guide for Oracle Information Rights Management Desktop*
- *Oracle Fusion Middleware External User Support Guide for Oracle Information Rights Management Desktop*

Overview of Oracle Information Rights Management (Oracle IRM)

Oracle Information Rights Management (Oracle IRM) enables users to manage and access sealed documents. When integrated with Oracle Beehive, users can seal documents inside Oracle Beehive workspaces. Document owners (that is, participants who add documents to workspaces) control access to their sealed document and any subsequent copies, including those saved by users on their computers or sent as e-mail attachments.

For example, consider a workspace called `Patents`, to which an Oracle Beehive administrator configures and applies an Oracle IRM policy that automatically seals all documents within the workspace. A workspace participant (Joe), uploads a document called `beehive-patent.doc` to the workspace. Joe's action automatically seals

`beehive-patent.doc` as `beehive-patent.sdoc`, according to the specifications of the previously mentioned policy.

> **Note:** When a document is sealed, its extension is modified with the addition of a leading .s*. For example, Microsoft Word documents have an extension `.sdoc`. Microsoft Excel documents have an extension `.sxls`. Oracle IRM handles the conversion to the new extensions.

Joe then forwards a copy of the sealed document in an e-mail to another `Patents` workspace participant (Jennifer) who has `READ` privileges only. By using Oracle IRM Desktop, which she previously installed, Jennifer can save a copy of the sealed document on her computer. However Jennifer can only open and read it. She cannot modify it.

Futhermore, if Jennifer's membership in the workspace is modified, or if Joe changes the permissions associated with `beehive-patent.sdoc`, those changes apply to any copies of the document on Jennifer's computer. For example, copies on Jennifer's computer become inaccessible to her if any of the following occur:

- Jennifer is removed from the workspace
- The document is deleted from the workspace
- The document is moved to a folder where Jennifer does not have access
- Jennifer's Oracle Beehive account is suspended or removed (say, if she leaves the organization or company)

Overview of Integrating Oracle IRM with Oracle Beehive

The process of integrating Oracle IRM and Oracle Beehive consists of the following steps:

1. Generate a universal unique identifier (UUID) for the Oracle IRM classification system that Oracle Beehive will use.
2. Use `beectl` or Oracle Beekeeper to configure and start the Oracle Beehive Information Rights Management (IRM) Service. Configuring the IRM Service includes specifying the Oracle IRM server and other settings.
3. Create an XML-based classification plug-in collection file and deploy it in a Java archive (`.jar`) on the Oracle WebLogic application server that hosts Oracle IRM.
4. Use Oracle Beekeeper to configure one or more Oracle IRM policies and apply them to Oracle Beehive workspaces.
5. Verify that the user accounts in Oracle Beehive match the user accounts in Oracle IRM server, and resolve any differences.

From the end-user perspective, the following steps are necessary:

1. Install Oracle IRM Desktop on users' computers.
2. Configure each user copy of Oracle IRM Desktop so that it points to the Oracle IRM server instance that was previously integrated with Oracle Beehive.

After completing these steps, Oracle Beehive users can upload documents to workspaces where they are sealed automatically, and then check out the sealed documents and manage them according to their permissions.

Prerequisites for Integrating Oracle IRM with Oracle Beehive

Oracle Beehive supports integration with Oracle IRM 11*g*. Integrating Oracle IRM with Oracle Beehive requires a strong understanding of the concepts and procedures related to managing Oracle IRM and developing applications for it.

Before integrating Oracle IRM with Oracle Beehive, verify your deployment meets the following prerequisites:

- Both the Oracle IRM server and Oracle Beehive are installed and running
- Oracle IRM Desktop is installed on users' computers

> **Note:** Oracle IRM Desktop is required to work with sealed documents, but it is not required for *sealing* documents. Oracle IRM automatically seals documents for users when they upload documents to Oracle Beehive workspaces where IRM policies have been applied.

Procedures for Integrating Oracle Beehive with Oracle IRM

The section describes the procedures for integrating Oracle Beehive with Oracle IRM, and includes the following topics:

Step 1: Generate a Universal Unique Identifier (UUID)

Generate a universal unique identifier (UUID). The UUID will be used to identify the classification plug-in collection file that you create in the steps that follow. For more information, refer to *Oracle Fusion Middleware Developer's Guide for Oracle Information Rights Management Server*.

Step 2: Configure and Start the Information Rights Management (IRM) Service

By default, the Information Rights Management (IRM) Service is disabled. To enable integration with Oracle IRM, configure and start the IRM Service using `beectl` or Oracle Beekeeper (recommended). The procedure in this section refers to Oracle Beekeeper, although you can use beectl if you're familiar with the equivalent commands and options.

To configure and start the IRM Service with Oracle Beekeeper:

1. In the Services pane of Oracle Beekeeper (bottom left-hand side), click **IrmService**.
2. Click the **Configuration** tab.
3. Click **Edit**.
4. Click **Advanced**.
5. Modify the values for the following properties according to your deployment:

Note: For more information on IRM Service properties, see "IrmService" in *Oracle Beehive Administrator's Reference Guide*.

- ClassificationId
- ClassificationLabelDescription
- ClassificationLabelName
- ClassificationLocale
- ClassificationSystemId
- ContentLabelDescription
- ContentLabelLocale
- ContentLabelName
- DesktopSyncURI
- DesktopURI
- IrmDesktopServiceEndPoint
- IrmSealingServiceEndPoint
- IrmUserName
- IrmUserPassword
- KeysetUUID
- LicenseExpirationTime
- SystemLabelDescription
- SystemLabelLocale
- SystemLabelName

6. From the status drop-down list, click **Enable**.
7. Click **Apply** to apply the changes to the proposed configuration and leave the window open, or click **Save & Close** to apply the change and close the window.
8. Activate your configuration. In the System panel, select **Configuration**. The Configuration pane appears on the right-side of the screen. The version number is displayed in the Active Configuration section.

Step 3: Create and Deploy a Classification Plug-in Collection File

After configuring and starting the IRM Service, create a classification plug-in collection file (`.xml`), bundle it in a Java archive (`.jar`), and deploy the archive on the Oracle WebLogic application server that hosts Oracle IRM.

For more information on context classification systems and collections, see the following guides:

- *Oracle Fusion Middleware Administrator's Guide for Oracle Information Rights Management Server*
- *Oracle Fusion Middleware Developer's Guide for Oracle Information Rights Management Server*

To create and deploy a classification plug-in collection file:

1. Create a classification plug-in collection file based on the sample provided in Appendix A, "Example of a Classification Plug-in Collection File."
2. In the classification plug-in collection file, specify the following settings based on your deployment:
   - **uuid:** The UUID that you generated in Step 1: Generate a Universal Unique Identifier (UUID) and set in Step 2: Configure and Start the Information Rights Management (IRM) Service.
   - **endpointAddress:** The URL for your Oracle Beehive server.
3. Save the classification plug-in collection file as `oracle_irm_classification_plugins.xml`.
4. Create a new Java archive (`.jar`) and include `oracle_irm_classification_plugins.xml` in it at the following location:

   `<archive_name>/META-INF/`
5. Deploy the Java archive on the application server that hosts Oracle IRM and in the same domain as Oracle IRM. In other words, move the `.jar` file to the following directory:

   `$WLS_HOME/<irm_domain>/lib`

   where `WLS_HOME` is the Oracle WebLogic server home directory where Oracle IRM is running and `<irm_domain>` is the name of the Oracle WebLogic domain into which the Oracle IRM server was installed.
6. Restart the application server that hosts Oracle IRM.

Step 4: Configure an Oracle IRM Policy

This section describes how to create and configure an Oracle IRM policy that seals documents in Oracle Beehive. The prerequisites are as follows:

- An Oracle Beehive instance that is already configured to work with an Oracle IRM server
- Administrative privileges for Oracle Beekeeper

For more information about Oracle Beehive policies, refer to *Oracle Beehive Administrator's Guide*.

To create and configure an Oracle IRM policy:

1. Log into Beekeeper by using an account with `ADMIN` privileges.
2. On the Enterprises pane, select **Policies**.
3. Near the top of the Policies pane, select the workspace name.

   This controls the scope of the policy.
4. Under the Policies tab, click **New**.
5. In the New Policy window, select **General** tab.

   Provide the name and description of the policy.
6. Select the **Rules** tab.

   Using the Condition Builder, specify the condition for the new policy.

   You may use such attributes as `DOCUMENT_CREATED`, `DOCUMENT_COPIED`, `DOCUMENT_MOVED`, and `DOCUMENT_UPDATED`.

Note that Oracle supports only document events.

For example, to seal all Microsoft Word documents, build the following condition:

```
custom_attributes.document_name LIKE '%doc'
```

7. Select the action **Seal Document**.
8. Click **Save**.
9. Click **Close**.

Step 5: Install and Configure Oracle IRM Desktop

This section contains the procedure for installing and configuring Oracle IRM Desktop on Microsoft Windows XP only. For more information about installing Oracle IRM Desktop, see *Oracle Fusion Middleware User's Guide for Oracle Information Rights Management Desktop*.

To install and configure Oracle IRM Desktop:

1. Download the Oracle IRM Desktop installation program and save it on your computer. If you do not know the location of the Oracle IRM Desktop installation program for your deployment, contact your system administrator.
2. Save all your work and close all Microsoft Office applications.
3. Start the installation program.
4. On the Welcome page, click **Next**.

   Click **Next** again.
5. On the Oracle IRM Desktop options screen, select **General** tab.

   Ensure that **Select all Preferences** is checked.
6. Select the **Servers** tab.

   Click **New**.
7. On the Add Server window, enter the server URL.

   For example:

   ```
   https://irm.mycompany.com:8002/irm_desktop
   ```
8. Click **Validate**.

   You may have a self-signed certificate, which you must validate or import.
9. In the Certificate Import Wizard window, click **View Certificate**.

   Click **Install Certificate**.
10. Click **Finish**.

    The desktop client prompts for your Oracle IRM Server credentials.
11. In the Oracle IRM Server Credentials window, enter your username (your e-mail address) and password.
12. Click **OK**.
13. When the authetication completes, you can see the new server in the Oracle IRM Desktop options window in the **Servers** tab.

Using Documents Sealed by Oracle IRM

This section contains instructions for a subset of the tasks that you can perform with sealed documents using Oracle IRM Desktop. For more information, refer to *Oracle Fusion Middleware User's Guide for Oracle Information Rights Management Desktop*.

Sealing an Oracle Beehive Document

You may choose to configure a workspace-level policy to seal newly uploaded documents based on a particular condition. You may also specify folder options as a condition of a workspace policy.

For example to seal all documents that use the word *patent* in their name, build the following policy using the instructions in "Step 4: Configure an Oracle IRM Policy":

```
custom_attributes.document_name LIKE '%patent%'
```

Orace IRM can be used to seal all Microsoft Office format documents (with extensions `.doc`, `.xls`, `.ppt`, and so on), Acrobat documents (with extension `.pdf`), and image files (with extensions `.jpg`, `.gif`, and so on). For a complete list of supported file types and extensions, see *Oracle Fusion Middleware User's Guide for Oracle Information Rights Management Desktop*.

After an appropriate policy is implemented, upload a document to a folder or workspace with a defined sealing policy, and it is automatically sealed.

After a document is sealed, it replaces the original document by overwriting it.

Note that the sealing process is asynchronous and may take a few seconds to minutes depending on the size of the document and load on the system.

Note also that Oracle IRM supports manual sealing of documents, however the integration between Oracle Beehive and IRM does not because the Oracle IRM sealing capability is not exposed in the client. Therefore, documents can only be sealed using policies.

Opening a Sealed Document

After you install Oracle IRM Desktop, you may access a sealed document just like any other document, such as by double-clicking on the document icon. When you open a sealed document for the first time, you are prompted for your username and password. Enter your e-mail address, such as `my.name@my.company.com`, and the corresponding password. After you are authenticated, you may work with the document for sixty minutes.

The Oracle Beehive IRM Service prevents users from accessing local copies of documents that are deleted from the server. As this check occurs only once every sixty minutes (to avoid overloading the server with check requests), you may be able to access a sealed document for a short time even if the server copy is deleted.

To verify if a document exists in Oracle Beehive:

1. In the system tray, right-click the Oracle IRM Desktop icon and select Checkin.
2. Open the document.

   You may be prompted for your username and password.

   If the document is accessible, you can see the document content. Otherwise, you receive an `Access Denied` message with an explanation of why you are unable to access the document content.

Note that a sealed document is protected by Oracle Beehive workspace permissions. Therefore, if you do not have WRITE access to the document on the server, you will not be able to edit the local sealed copy. Similarly, you cannot copy the document content into another document if you do not have WRITE permissions for the document.

To summarize actions allowed based on Oracle Beehive permissions:

- Oracle Beehive READ permission allows the actions OPEN and PRINT.
- Oracle Beehive WRITE permission allows the actions OPEN, PRINT, EDIT, and COPY

Updating a Sealed Document

After opening a sealed document, you may edit it as you would a regular document provided you have Oracle Beehive WRITE permissions. If you have READ permissions only, then you cannot copy text from a sealed document and paste it onto another document.

After editing a document, you may save it as a sealed document. You may then upload the updated sealed document into the workspace, overwriting the older copy. A background process reseals the document with updates.

Note that if your workspace is set up with manual version management (check-in and check-out), you cannot open stale local copies of the document. You must obtain the most recent copy from the workspace to open or edit a sealed document.

6

Integrating Oracle Universal Content Management with Oracle Beehive

This module provides an overview of Oracle Universal Content Management (Oracle UCM) integration with Oracle Beehive.

This module includes the following topics:

- Overview of Oracle Universal Content Management
- Prerequisites for Integration with Oracle UCM
- Procedures for Integration with Oracle UCM
- Administering an Integrated Oracle UCM Environment

Overview of Oracle Universal Content Management

This section describes the benefits and limitations of integrating Oracle UCM with Oracle Beehive, provides an architectural overview and deployment models of this integration, and discusses network considerations.

For definition of terms used in this section, see the Glossary in *Oracle Beehive Concepts*.

This section contains the following topics:

Benefits of Integrating Oracle UCM with Oracle Beehive

Oracle UCM provides a central repository for Web site content management and other application content management.

Integrating Oracle Beehive with Oracle UCM enables users to access published content directly form their daily working environment in the context of team workspaces. Therefore, they collaborate in the Oracle Beehive team workspaces to browse a remote repository, read documents, update them, produce new content, and create shortcuts to remote documents and folders.

Limitations of Integrating Oracle UCM with Oracle Beehive

The limitations of integrating Oracle UCM with Oracle Beehive include the following:

- Content that is hosted in Oracle UCM is accessible to Oracle Beehive as read-only.
- Publishing to Oracle UCM is not supported in Oracle Beehive Release 2 (2.0).
  - You can manually check content back into Oracle UCM directly, using Oracle UCM client tools.

Architectural Overview of Oracle UCM Integration and Oracle Beehive

Beehive integration with Oracle UCM implements access to remote repositories.

The remote repository model of integration supports read-only access in Oracle UCM. It enables users to collaborate in Oracle Beehive Team Collaboration, to browse the remote repository, read documents, and create shortcuts to remote documents and folders. If content must be updated, it may be copied from a content repository to an Oracle Beehive team workspace.

Remember the following points:

- Both the Oracle Beehive instance and the Oracle UCM instance must use the same user repository base.
- The Oracle Beehive instance is aware of the Oracle UCM instance. Oracle UCM is not aware of Oracle Beehive. This ensures very minimal or no configuration changes to existing Oracle UCM deployments in the organization where Oracle Beehive is deployed.
- Content is typically not duplicated or replicated between Oracle Beehive and Oracle UCM, and no copies of content from Oracle UCM are stored in Oracle Beehive. (Note, duplicate copies of content will exist if users manually copy content from Oracle UCM and paste it in Oracle Beehive workspaces.) This eliminates content management overhead, and ensures that Beehive users who access remote content always receive the most current content in Oracle UCM.
- The access control and security applied to the content in Oracle UCM instances is maintained while users access content through Oracle Beehive.

  For example, if a user does not have access to specific content in an Oracle UCM instance, he will not have access to the same content through Oracle Beehive.
- Oracle Beehive users can access Oracle Beehive content, such as documents, Wiki pages, and forums, and also the remote content from a single client, the Oracle Beehive Team Collaboration. Users can manage this remote content and documents as a Oracle Beehive documents for collaboration purposes, creating shortcuts, applying tags and categories, and so on.

Network Considerations of Integrating Oracle UCM with Oracle Beehive

When Oracle UCM and Oracle Beehive are on the same network without a firewall, the steps described in this module work as expected.

In your organization, Oracle Beehive, Oracle UCM, and your user directory may be running on different secured networks. In that case, additional steps must be completed to ensure that the integration works.

- If the LDAP is on a protected network, then Oracle Beehive and Oracle UCM servers must be granted access for user authentication.

- If Oracle UCM is on a protected network, then access must be granted to the Oracle Beehive servers. There are two common configurations for this network architecture:
  - Oracle Beehive is in DMZ, and Oracle UCM is in the secure network. Then the firewall must be configured to allow Oracle Beehive servers (or Oracle Beehive server network) to initiate connections to Oracle UCM servers.
  - Oracle Beehive and Oracle UCM are on separate networks, separated by different NAT firewalls. In this scenario, integration may not be possible. Opening a tunnel to Oracle UCM enables all connections from the Oracle Beehive network to have administrative access to Oracle UCM.

Note that other network topologies may have different requirements for allowing connections between Oracle UCM and Oracle Beehive. For more information, contact Oracle Support.

Prerequisites for Integration with Oracle UCM

Oracle Beehive supports integration with Oracle UCM 10*g* Release 3 or higher. At minimum, the user bases for Oracle UCM and Oracle Beehive must match. To meet this requirement, Oracle recommends that you configure Oracle UCM and Oracle Beehive to use the same user directory through LDAP.

To complete an integration with Oracle UCM, you must have the following privileges:

- Oracle UCM Administrator privileges to configure an LDAP provider, and to configure roles, groups, and credential maps.
- Command line administrator access to a server that hosts the Oracle UCM instance, to update configuration files and to restart the Oracle UCM instance.
- Beehive System Administrator (Beekeeper) privileges for creating, enabling, disabling, or deleting remote repositories.
- For each workspace, Oracle Beehive workspace-coordinator privileges (through Oracle Beehive Team Collaboration) for enabling remote content access for the workspace, and for creating additional remote mounts if needed. Remote repositories can refer to remote folders or saved UCM queries, while remote mounts can refer to remote repositories only.

  In practice, if a remote repository is already defined, any workspace coordinator can configure remote mount points in a workspace they own. Because every user can create team workspaces, no additional privileges are required.

Procedures for Integration with Oracle UCM

To integrate Oracle Beehive with Oracle UCM, you must prepare the Oracle UCM instance for integration, and also ensure that both a remote repository and a remote mount exists.

This section contains the following topics:

Configuring the Oracle UCM Instance

To prepare an Oracle UCM instance for integration with Oracle Beehive, you must complete the following tasks:

Configuring Host-Based Authentication

Oracle UCM must be configured to allow administrative access to the middle tiers of the Oracle Beehive instance used in the integration. This means that the integration is made through an LDAP instance that is configured for use with the appropriate Oracle Beehive instance.

To configure host-based authentication:

1. Log in to the server that hosts the Oracle UCM instance.
2. Change to the *UCM_HOME*/config directory.

   UCM_HOME is the directory where Oracle UCM is installed.
3. Edit the file config.conf in one of the following ways:
   - Set the SocketHostAddressSecurityFilter property to enable all hosts to connect:

     ```
     SocketHostAddressSecurityFilter=*.*.*.*
     ```

   - Set the SocketHostAddressSecurityFilter property to enable only one hosts to connect:

     ```
     SocketHostAddressSecurityFilter=1.2.3.4
     ```

   - Set the SocketHostAddressSecurityFilter property to enable several specified hosts to connect:

     ```
     SocketHostAddressSecurityFilter=1.2.3.4 1.2.3.5,1.2.3.6,1.2.3.7
     ```

4. Restart the Oracle UCM instance:

   UCM_HOME/etc/idcserver_restart

Creating and Configuring an LDAP Provider

Oracle UCM and Oracle Beehive must leverage the same user base. Oracle recommends that you create and configure an LDAP provider to prepare the Oracle UCM instance for integration with Oracle Beehive.

To create and configure an LDAP provider:

1. Log in to the Oracle UCM interface using a sysadmin account.
2. In the navigation bar, click **Administration**.
3. Click **Providers**.

   The Providers page displays.

4. On the Providers page, select **Add a new LDAPUSER provider**.
5. Enter values for the following fields:
   - Provider Name
   - Source Path
   - LDAP Server
   - LDAP Port
   - Credential Map
   - Default Network Roles
   - LDAP Admin DN

Creating Login Mappings

To authenticate users, you must map logins into Oracle UCM to the correct user fields in LDAP.

To create login mappings:

1. Log in to the server that hosts the Oracle UCM instance.
2. Change to the *UCM_HOME*/data/providers/*provider_name* directory.
   - *UCM_HOME* is the directory where Oracle UCM is installed.
   - *provider_name* is the directory for the LDAP provider created in section "Creating and Configuring an LDAP Provider" on page 6-4.
3. Edit the file `provider.had` by adding the following line:

   ```
   LdapUserSearchFilter=(&objectclass=person)(mail=user))
   ```

4. Restart the Oracle UCM instance:

   ```
   UCM_HOME/etc/idcserver_restart
   ```

Creating Security Roles, Groups, and Permissions

In the security protocol for Oracle UCM, users are assigned roles and content is assigned to security groups. In this context, *roles* have *permissions* for *groups*.

You must create roles and security groups in Oracle UCM to prepare it for integration with Oracle Beehive.

Examples of security groups could be of the type `OUR_PUBLIC_GROUP`. Examples of security roles could be of the following types:

- All users: `OUR_USERS` with `READ`, `WRITE`, `DELETE` permissions for `OUR_PUBLIC_GROUP`
- Administrators: `OUR_ADMINS` with `READ`, `WRITE`, `DELETE`, `ADMIN` permissions for `OUR_PUBLIC_GROUP`

To create a security role:

1. Log in to the Oracle UCM interface using a `sysadmin` account.
2. In the navigation bar, click **Administration**.
3. Click **Admin Applets**.
4. Select **User Admin**.

A User Admin window appears.

5. In the User Admin window, select the **Security** menu, and then select **Permissions by Role**.
6. In the new window, click **Add New Role**.
7. In the new window, enter the **name** of the new role you are creating.
8. Click **OK**.
9. Close the **Permission by Role** window.

To create a security group:

1. Log in to the Oracle UCM interface using a `sysadmin` account.
2. In the navigation bar, click **Administration**.
3. Click **Admin Applets**.
4. Select **User Admin**.

   A User Admin window appears.
5. In the User Admin window, select the **Security** menu, and then select **Permissions by Group**.
6. In the new window, click **Add Group**.
7. In the new window, enter the **name** and **description** of the new group you are creating.
8. Click **OK**.
9. Close the **Permission by Role** window.

To assign permissions:

1. Log in to the Oracle UCM interface using a `sysadmin` account.
2. In the navigation bar, click **Administration**.
3. Click **Admin Applets**.
4. Select **User Admin**.

   A User Admin window appears.
5. In the User Admin window, select the **Security** menu, and then select **Permissions by Group**.
6. Select an existing security group.

   A list of available roles appears.
7. Select a role and click **Edit Permissions**.
8. Select the permissions for that role in the security group.
9. Click **OK**.
10. Close the **Permission by Role** window.

Registering a Credential Map

You must register the credential map that is specified in the section "Creating and Configuring an LDAP Provider" on page 6-4.

To register a credential map:

1. Log in to the Oracle UCM interface using a `sysadmin` account.
2. In the navigation bar, click **Administration**.
3. Click **Credential Map**.
4. Enter the name of the credential map that you specified when creating the LDAP provider, in step 5 of "Creating and Configuring an LDAP Provider" on page 6-4.
5. In the text field, add the following entry:

```
|#all|,                %%
|#all|,                OUR_USERS
&<login_id>,           OUR_ADMINS
```

   This uses the example from "Creating Security Roles, Groups, and Permissions" on page 6-5
6. Click **OK**.

Creating and Configuring a Remote Repository

This section briefly discusses remote repositories, demonstrates how to create a one through the `beectl` command line, and how to configure it by enabling it in Oracle Beehive Team Collaboration.

Note that you must create a remote repository based on a seeded `RemoteRepositoryDefinition`. If an appropriate remote repository exists, you may use it instead. Also, remote repositories can be either be path-based or query-based.

This section contains the following topics:

About Oracle UCM Remote Repositories

When a user is connected to a remote repository, the remote mounts have read-only access.

Users can do the following:

- Browse remote mounts, including drill down folder trees and read documents
- Create shortcuts to a remote folder or document
- Copy documents and folders locally to the Oracle Beehive workspace

Users cannot do the following:

- Upload a document directly to a remote mount
- Create, update, or delete folders or documents

Creating Remote Repositories

You can create and configure a remote repository using Oracle Beekeeper. For more information, see "Managing Remote Repositories" in *Oracle Beekeeper Online Help*.

You can also create and configure a remote repository using the `beectl add_remote_repository` command.

To create and configure a remote repository using beectl:

1. Decide on the repository definition you want to use.

   At this time, Oracle Beehive supports only Oracle UCM.

2. Create a remote repository configuration file.

   The following code is an example of a configuration file.

```
<?xml version="1.0" encoding="UTF-8" ?>
<RemoteRepositoryInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xml.oracle.com/beehive/remotecontent/
    remote_repository_template.xsd"
  xmlns="http://xml.oracle.com/beehive/remotecontent">
  <name>computer_name</name>
  <remote_repository></remote_repository>
  <description>repository_with_UCM_instantiated</description>
  <definitionname>Oracle UCM</definitionname>
  <scope>enpr=oracle</scope>
  <Attributes>
    <attribute>
      <name>host</name>
      <defaultValue>default_value_of_machine</defaultValue>
      <final>true</final>
    </attribute>
  </Attributes>
</RemoteRepositoryInfo>
```

3. Add the remote repository to the workspace using the following command:

```
beectl add_remote_repository --file /path/remote_repository_file.xml
```

Enabling a Remote Repository in Oracle Beehive Team Collaboration

You enable remote repositories for workspaces using Oracle Beehive Team Collaboration. To enable a remote repository, you must have workspace coordinator privileges in the workspace where you want to enable the remote repository.

For more information, including the steps to enable a remote repository using Oracle Beehive Team Collaboration, see the *Oracle Beehive Team Collaboration Help* at the following location:

http://www.oracle.com/technology/products/beehive/beehive_users/2_0/teamcollab.htm

Creating a Remote Mount

You can create a remote mount in a workspace either through Oracle Beehive Team Collaboration or using the `beectl add_remote_share` command. In either case, ensure that you have workspace coordinator privileges in the workspace.

For the steps to enable a remote mount using Oracle Beehive Team Collaboration, see *Oracle Beehive Team Collaboration Help* at the following location:

http://www.oracle.com/technology/products/beehive/beehive_users/2_0/teamcollab.htm

For the steps to enable a remote mount using the `beectl add_remote_share` command, see "Creating a Remote Mount Using beectl" on page 6-9.

Creating a Remote Mount Using beectl

You can create a remote repository using the `beectl add_remote_share` command. Also, you may create a remote mount in a workspace even without an enabled remote repository in that workspace. The mount will remain hidden until the remote repository is enabled in the workspace.

Note that the terminology remote mount in Oracle Beekeeper is equivalent to remote share in the XML files used by the `beectl` command environment.

To create a remote mount using beectl:

1. Decide on the remote repository to use as the basis for the mount or share.
2. Create a remote mount configuration file. See `folder_mount.xml` and `query_share.xml` for examples.

   The following code is an example of a folder-based mount configuration file.

```
<?xml version="1.0" encoding="UTF-8" ?>
<RemoteShareInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xml.oracle.com/beehive/remotecontent/
    remote_share.xsd"
  xmlns="http://xml.oracle.com/beehive/remotecontent">
  <name>computer_name_folders_only</name>
  <remote_share></remote_share>
  <description>Beehive_Documents_Folder_on_Oracle_UCM</description>
  <repository_scope>enpr=oracle</repository_scope>
  <repository_name>UCM_repository_name</repository_name>
  <scope>wksp=Beehive Team,enpr=oracle</scope>
  <Attributes>
    <attribute>
      <name>rootPath</name>
      <value>/value_of_root_path</value>
    </attribute>
  </Attributes>
</RemoteShareInfo>
```

   The following code is an example of a query-based mount configuration file.

```
<?xml version="1.0" encoding="UTF-8" ?>
<RemoteShareInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xml.oracle.com/beehive/remotecontent/remote_share.
  xmlns="http://xml.oracle.com/beehive/remotecontent">
  <name>computer_name_search_only</name>
  <remote_share></remote_share>
  <description>Beehive Search on Oracle UCM.</description>
  <repository_scope>enpr=oracle</repository_scope>
  <repository_name>UCM_repository_name</repository_name>
  <scope>wksp=Beehive Team,enpr=oracle</scope>
  <Attributes>
    <attribute>
      <name>searchQuery</name>
      <value><![CDATA[dDocTitle <matches> `*RCS*`]]></value>
    </attribute>
  </Attributes>
</RemoteShareInfo>
```

3. Add the remote mount to the workspace using the following command:

```
beectl add_remote_share -file /path/remote_share_file.xml
```

Administering an Integrated Oracle UCM Environment

The system administrator can disable select repositories in all workspaces, by using the Oracle Beekeeper interface.

After the repository or mount are defined and configured, the workspace coordinator performs the following tasks:

- The workspace coordinator must enable the specific repository to make it available in that workspace. Each workspace can have its own set of enabled repositories.
- After a repository is enabled, a default mount to that repository is created.
- The workspace coordinator may also create additional mounts to the same repository, based either on a folder or on a saved query.
- The workspace coordinator may disable previously enabled repositories. This action does not delete any mounts, but it only removes them from the view of the user.

7

Integrating Oracle Universal Records Management with Oracle Beehive

This module provides an overview and steps for integrating Oracle Universal Records Management (Oracle URM) with Oracle Beehive.

This module includes the following topics:

Overview of Integrating Oracle URM with Oracle Beehive

Oracle URM enables organizations to manage their records and retention policies, disposition processes, and litigation holds or freezes in a central repository. Organizations can then apply those policies, dispositions, and holds to content stored in other systems, such as Oracle Beehive. Oracle Beehive integrates with Oracle URM through a records management service.

Records management is an optional service. It is enabled by installing and configuring Oracle Beehive with Oracle URM, which provides life cycle and disposition management of records-managed Oracle Beehive artifacts. After Oracle URM is installed and configured with Oracle Beehive, you may start the records management service and begin filing records for documents and e-mails.

Benefits of Integrating Oracle URM with Oracle Beehive

Oracle Beehive manages records organized through Oracle URM. Oracle Beehive retains artifacts for which you have filed records within the Oracle Beehive database, but treats them differently from other artifacts.

When an artifact becomes records-managed in-place, Oracle Beehive ensures that the content is never truly altered or deleted from the system unless an Oracle URM action is issued. From an end-user perspective, users are able to delete e-mails or documents that are records-managed, and empty them from the Workspace Trash.

Those records-managed artifacts are stored in a special Records Management container in the Oracle Beehive database. The Oracle URM application can still query for, and perform operations on these stored artifacts.

Artifacts may be placed under retention, which means that they can be treated as regular artifacts by Oracle Beehive.

Artifacts may also become records-managed without being placed under retention. These types of records are called non-records (as a short form of non-retained records). Artifacts with non-records may be treated as regular artifacts, including being modified or deleted, by Oracle Beehive. Oracle URM sends an instruction to Oracle Beehive to handle non-record artifacts if they still exist in the system.

In Oracle Beehive, you can create records of any of the following:

- Any document stored in a workspace in Oracle Beehive
- Any e-mail in an Oracle Beehive e-mail Inbox or subfolder of an Inbox
- Any e-mail sent from an Oracle Beehive e-mail user, if you turn on this feature by setting a property of the E-mail Service

As an administrator, you may manually file records for these artifacts using a `beectl` command, or by using Oracle Beekeeper. Additionally, you may create policies to automatically file records for documents and e-mails, based on triggering criteria.

Note that Oracle Beehive's built-in auditing function automatically audits events related to Records Management. Such auditing is not configurable by the administrator. You may, however, create an audit trail to review record management activity. For more information about auditing in Oracle Beehive, see "Managing Audit Policies" in *Oracle Beehive Administrator's Guide*.

Whenever a document is filed as a record, metadata about the artifact is sent to Oracle URM. This metadata describes the artifact and its original context. The artifact itself continues to be stored in Oracle Beehive. This ensures that the content is immutable from a system perspective (unless it is a non-record).

Table 7–1 lists the document metadata sent to Oracle URM with documents. Table 7–2 lists the metadata sent to Oracle URM when e-mail messages are filed as records.

Table 7–1 Artifact Metadata Sent to Oracle URM with All Artifacts

Name	Type
CollabID	`String`
Creator Name	`String`
Artifact URL	`String`
Media Type	`String`
Creation Date	Date in "`MM/DD/YY`" format

Table 7–2 Artifact Metadata Sent to Oracle URM with E-Mail Messages

Name	Type
Sender	`String`
Sent Date	Date in "`MM/DD/YY`" format
Subject	`String`
Hidden Addresses	`String`, a comma-delimited list of addresses
Primary Addresses	`String`, a comma-delimited list of addresses
Secondary Addresses	String, comma delimited list of addresses

Prerequisites for Integrating Oracle URM with Oracle Beehive

Deploying Oracle Beehive with Oracle URM requires that you have both an Oracle Beehive instance and an Oracle URM 10*g* R3 (or later) instance installed and configured.

Procedures for Integrating Oracle URM with Oracle Beehive

To integrate Oracle URM with Oracle Beehive, you perform certain tasks in Oracle Beehive and other tasks in Oracle URM, as follows:

1. In Oracle URM, ensure that the Records Management administrator has all the relevant roles and privileges. For details of this process, see "Step 1: Granting the Necessary Roles to the Records Management Administrator".
2. In Oracle Beehive, register Oracle URM and configuring Oracle Beehive. This includes the following:
   - Specifying the Oracle URM instance in Oracle Beehive
   - Enabling the Records Management Service
   - Optionally, enabling record filing of sent e-mails in Oracle Beehive.
   - Updating the Oracle Beehive configuration
   - Restarting the BEEAPP managed component

   For details of this process, see "Step 2: Registering Oracle URM and Configuring Oracle Beehive" on page 7-4.
3. In Oracle URM, create retention categories and record folders. This includes the following:
   - Creating retention categories in Oracle URM
   - Creating record folders under a retention category
   - Optionally, viewing the retention categories and folders in Oracle Beehive

   For details of this process, see "Step 3: Creating Retention Categories and Record Folders in Oracle URM" on page 7-7.
4. In Oracle URM, set up disposition rules on retention categories. For details of this process, see "Step 4: Setting Up Disposition Rules in Oracle URM" on page 7-7.

For detailed instructions on installing and configuring Oracle URM see the *Oracle Universal Records Manager Installation Guide*.

Note that a special administrator account with super user privileges on all data may be created in Oracle Beehive. This account may be used for queries in Oracle URM, to enable access to the records that Oracle URM is pulling from Oracle Beehive.

The following sections describe the steps for integrating Oracle URM with Oracle Beehive. Both Oracle Beehive and Oracle URM should be installed before you perform these tasks. Additionally, you must have administrator privileges.

Step 1: Granting the Necessary Roles to the Records Management Administrator

The Oracle Beehive Records Management Service uses the Oracle URM administrator account and Web Services to connect to Oracle URM. To enable this connection, you must grant the necessary roles to the Oracle URM administrator account.

To grant necessary roles to the Records Management administrator:

1. Log in to Oracle URM.
2. Click **Admin Applets**.
3. Click **User Admin**.
4. Select the Admin user (`sysadmin`).

   Click **Edit**.
5. Add the following roles: `rma`, `rmaadmin`, `admin`, `ermadmin`, `ermrequestor`, `rmaprivileged`, and `sysmanager`.

   Note: If the values for these roles change in Oracle URM, corresponding changes must made in Oracle Beehive, and the Records Management Service must be restarted. You may use the `beectl modify_property` command to update these values in Oracle Beehive.
6. Click **Save**.
7. Exit the interaction.

Step 2: Registering Oracle URM and Configuring Oracle Beehive

To register Oracle URM and configure Oracle Beehive, complete the following tasks:

- Step 2A: Specifying the Oracle URM Instance
- Step 2B: Enabling the Records Management Service
- Step 2C: Enabling Record Filing of Sent E-mails
- Step 2D: Updating the Oracle Beehive Configuration
- Step 2E: Restarting the BEEAPP Managed Component

Step 2A: Specifying the Oracle URM Instance

To specify the Oracle URM instance in Oracle Beehive, you can use the `beectl` command-line utility or Oracle Beekeeper.

To specify the Oracle URM instance using the beectl utility:

1. On the operating system command line, issue the `beectl add_urm` command to configure an agent for an Oracle URM instance, as follows:

```
beectl add_urm
  --rm_admin_name URM_Inst_RM_Administrator_Name
  --rm_admin_password URM_Inst_RM_Administrator_Password
  --urm_url URM_Inst_URL
  --rm_email_admin RM_Administrator_Email_Id
 [--disposition_loader_interval URM_Inst_Disposition_Loader_Interval]
 [--disposition_processor_interval URM_Inst_Disposition_Processor_ Interval]
 [--agent_name URM_Inst_Agent_Name]
```

   where:

   - `rm_admin_name` is the Records Management administrator name for the Oracle URM instance. Permitted type is `String`. This option is mandatory.

- `rm_admin_password` is the Records Management administrator password for the Oracle URM instance. Permitted type is a secure string. A prompted value (with re-confirmation) or an obfuscated value (along with `--obfuscated` option) may be provided on the command line. In shell mode, clear text may also be provided. This option is mandatory.
- `urm_url` is the URL for the Oracle URM instance. Permitted value is `String`. This option is mandatory.

  This URL, which is required by the adapter to communicate with the Oracle URM server, depends on the Oracle URM instance name and server name specified during installation. This URL may be obtained from any URL to the Oracle URM server on the Web interface. For example, a service requests on the Web interface may be of the following form. Discard everything after the question mark. The service URL is the first part of any service URL string. In this case, it is the text in bold face.

  `http://urm.yourcompany.com/xpedio/idcplg``?IdcService=GET_DOC_PAGE&Action=GetTemplatePage&Page=HOME_PAGE&Auth=Internet`

- `rm_email_admin` is the e-mail ID of the Records Management administrator. Permitted type is `String`. This option is mandatory.

  If you use the `rm_email_admin` option to enable filing records of sent e-mail messages, the supplied user account must exist in Oracle Beehive. After activating the configuration, you cannot easily change this account value, so select it carefully. If you must change this account, contact Oracle Support.
- `disposition_loader_interval` is the disposition loader interval for the Oracle URM instance. Permitted type is `Long`.
- `disposition_processor_interval` is the disposition processor interval for the Oracle URM instance. Permitted type is `Long`.
- `agent_name` is the agent name for the Oralce URM instance. Permitted type is `String`.

  The agent name, specified by the `agent_name` option, may be any meaningful name that indicates that this agent is for the Oracle Beehive Records Management Service. It may contain up to `30` characters. Possible examples may be `BeeAdapter` or `BeehiveRMAdapter`.

2. Issue the `beectl activate_configuration` command:

```
beectl activate_configuration
```

To specify an Oracle URM instance using Oracle Beekeeper:

For the steps for specifying an Oracle URM instance using Oracle Beekeeper, refer to "Records" in the *Oracle Beekeeper Online Help*.

Step 2B: Enabling the Records Management Service

To enable the Records Management Service in Oracle Beehive, you can use the `beectl` command-line utility or Oracle Beekeeper.

To enable the Records Management Service using beectl:

1. Enable the Records Management Service by using the `beectl modify_property` command.

```
beectl>modify_property --component <RecordsManagementServiceInstance_id> --name
Status --value ENABLED
```

Note that the records management service is disabled by default.

Note also that you can get the component identifier for the Records Management Service instance by using the `beectl list_components` command.

To enable the Records Management Service using Oracle Beekeeper:

For the steps for enabling the Records Management Service using Oracle Beekeeper, see "Managing Services" in the *Oracle Beekeeper Online Help*.

Step 2C: Enabling Record Filing of Sent E-mails

By default, you can file records of e-mail messages from any Oracle Beehive user's Inbox or subfolder of Inbox. However, to file records of e-mail messages sent by Oracle Beehive users, you have to modify a property of the Transport Properties subcomponent of the E-mail Service. To do this, run the `beectl modify_property` command.

Note: Enabling record filing of sent e-mails is optional.

To enable record filing of sent e-mails:

1. At the command prompt, issue the `beectl modify_property` command:

```
beectl modify_property
  --component _EmailService:TransportProperties
  --name SentEmailPluginEnabled
  --value true
```

   where:

   - `component` is the Email Service, _EmailService:TransportProperties
   - `name` is the name of the property to modify, `SentEmailPluginEnabled`
   - `value` is the Boolean value for the named property, `true`

2. Issue the `beectl activate_configuration` command:

```
beectl activate_configuration
```

For more information about enabling record filing of sent e-mails, see "Configuring Sent E-mail Plugins" in *Oracle Beehive Administrator's Guide*.

Step 2D: Updating the Oracle Beehive Configuration

To update the proposed configuration in Oracle Beehive:

1. Use the `beectl activate_configuration` command to update Oracle Beehive with this proposed configuration:

```
beectl> activate_configuration
```

Step 2E: Restarting the BEEAPP Managed Component

BEEAPP is a managed component that contains a variety of Beehive services, such as the E-mail Service, the Presence Service, and the Records Management Service. You must restart it to use the new configurations and to restart the Records Management Service.

To restart the BEEAPP managed component:

1. Restart the `BEEAPP` OC4J managed component using the `beectl restart` command with the `--component` option:

   ```
   beectl> restart --component <your OC4J BEEAPP component identifier>
   ```

   You can use the `beectl status` command to list the managed components and their identifiers for your deployment.

Step 3: Creating Retention Categories and Record Folders in Oracle URM

After completing the steps in "Step 2: Registering Oracle URM and Configuring Oracle Beehive", log in to Oracle URM using a Records Management administrator account. Create the various Retention Categories and Record Folders as required by your organization. You may create retention categories either under Record Series or under File plan in the Oracle URM user interface. You may create Record Folders either under a retention category or under another record folder (nested record folders).

To create retention categories and record folders:

1. Click **Create** and select **Retention Category** to create a retention category under a file plan.

   Click a record series, then click **Create** and select **Retention Category** to create a retention category under a record series.

2. For each retention category, provide values required on the creation page. The retention category ID must be unique, but names can be duplicated. If the **Allow Non-records** is checked, then the retention category will allow the check-in of non-records, artifacts that respect disposition rules but have no content modification or deletion restrictions.

3. Click a retention category, click **Create**, and select **Record Folder** to create a record folder under a retention category.

   Open a record folder, click **Create**, and select **Record Folder** to create a record folder within another record folder.

4. Fill in the values required on the creation page and provide a unique record folder ID for each new record folder.

5. This is an optional step. To review the new retention categories and record folders in Oracle URM, view them in Oracle Beehive, in the `beectl` shell, using the `list_file_plan` command:

   ```
   beectl> list_file_plan
   ```

Step 4: Setting Up Disposition Rules in Oracle URM

Disposition rules are set on Retention Categories only. In Oracle Beehive, the Records Management Service supports only the `DESTROY` disposition action.

Disposition rules are set within the Oracle URM user interface.

To set up disposition rules for testing:

1. On the retention category page, select the **Information** drop-down list and click **Disposition Information**.

2. On the Disposition Instruction page, click the **Add** link.

   A **Disposition Rule** panel displays.

3. On the Disposition Rule panel, select a triggering event.

   For Oracle Beehive Records Management disposition tests, choose the **Cancel** event, which can be triggered from the Oracle URM administrator UI.

4. Specify a retention period.

   For Oracle Beehive Records Management disposition tests, set the retention period to zero weeks.

5. Choose Disposition Action **Destroy**.

6. Leave the **Destination Location** and **Destination Container** blank.

7. Set values for **Apply to Records Folder** and **Notification Reviewer**.

   By default, the Notification Reviewer is `sysadmin`.

8. Click **Actions**, select **Trigger Dates**, and select **Cancel** on the record from Oracle URM.

   This is a mandatory action that triggers a cancel action and starts disposition processing in Oracle URM.

Administering the Oracle URM and Oracle Beehive Integration

This section describes how to administer your Oracle URM and Oracle Beehive integration. This section includes the following topics:

- Filing Records of Artifacts in Oracle Beehive
- Removing Records Management of Artifacts in Oracle Beehive
- Troubleshooting Records Management Service Operations

Filing Records of Artifacts in Oracle Beehive

You can manually file records of artifacts (documents and e-mails) using `beectl` commands. You can also file records automatically, using policies.

Each of these procedures is detailed in the following sections:

- Filing Records of Artifacts Using beectl
- Filing Records of Artifacts Using Policies

These tasks are comprehensive and fully explain the use of records in Oracle Beehive. However, Oracle recommends that in most cases you can use Oracle Beekeeper to manage records. For more information, see "Records" *Oracle Beekeeper Online Help*.

Filing Records of Artifacts Using beectl

The command-line based record filing is provided primarily as a tool for system administrators to manually file records that were not filed using a policy action, using the `beectl add_record` command.

To manually file records using beectl:

1. At the command prompt, issue the `beectl add_record` command:

```
beectl add_record
  --artifact artifact_to_file_identifier
{ --retention_category Oracle_URM_retention_category_identifier |
  --record_folder Oracle_URM_record_folder_identifier }
[ --no_retention Boolean_specifying_if_record_or_non-record ]
```

where:

- `artifact` is the identifier of the artifact that must be filed
- `retention_category` is the identifier of the retention category for the document, which is defined in Oracle URM; use either this parameter, or `retention_folder`
- `record_folder` is the identifier of the record folder in Oracle URM; use this parameter or `retention_category`
- `no_retention` is the Boolean value that specifies `true` if a record is of a non-retention type (non-record)

2. This is an optional step. To view a list of record categories and record folders, along with their identifiers, issue the `beectl_list_file_plan` command:

```
beectl> list_file_plan
```

Filing Records of Artifacts Using Policies

You can make use of the Oracle Beehive policy framework to create records management policies. Records management policies automate the process of filing records for artifacts. You specify a policy condition for record filing, and a destination Retention Category. Oracle Beehive automatically files records for artifacts that meet the policy condition.

To set up a records management policy:

1. Select a Retention Category for this policy. All records that meet this policy's condition will be filed into that Retention Category. You can list available Retention Categories using the `beectl list_file_plan` command:

```
beectl> list_file_plan
```

2. Create an Oracle Beehive policy XML file, using the special Records Management policy action `File a Record`, and setting the `actionPreferenceInfo`, as in the following example:

```
...
      <ActionInfo>
        <name>File a Record</name>
      </ActionInfo>
       <ActionPreferenceInfos>
            <actionPreferenceInfo>
               <key>category_id</key>
               <value>Your Category ID</value>
            </actionPreferenceInfo>
           <actionPreferenceInfo>
               <key>is_record</key>
               <value>true</value>
            </actionPreferenceInfo>
         </ActionPreferenceInfos>
...
```

In this example, you must replace **Your Category ID** with the actual Retention Category ID you determined in step 1

3. Complete the XML file by specifying conditions to trigger the policy. Any condition which is valid for an Oracle Beehive policy may be used.

See Also: For detailed instructions on how to create Policy XML files, refer to "Creating and Managing Custom Policies" in the *Oracle Beehive Administrator's Guide*.

4. Create the policy by issuing the `beectl add_policy` command:

```
beectl> add_policy --file full_path_to_policy_xml_file
```

The following example demonstrates a typical Records Management policy. Here, a record is filed on any file uploaded to a specified folder, with the Retention Category `LC`.

```
<?xml version="1.0" encoding="UTF-8" ?>
<PolicyInfo isExtensible="true">
  <policy></policy>
  <scope></scope>
  <template></template>
  <name>Records Management Document Policy</name>
  <description>This policy files all documents uploaded to container identified
              by eid 4A92592959297602769797962 under retention category with
              id LC</description>
  <attributes>
    <attributeDefId></attributeDefId>
    <type></type>
    <value></value>
  </attributes>
  <RuleInfos>
    <RuleInfo priority="1">
      <name>RULE ONE</name>
      <description>Rule One</description>
      <eventTypeName>DOCUMENT_CREATED</eventTypeName>
      <ruleId></ruleId>
       <toRemove>false</toRemove>
       <templateRuleIds>
        <templateRuleId></templateRuleId>
      </templateRuleIds>
      <ConditionInfo>
         <Simple>
           <leftSide>common_attributes.container.eid</leftSide>
           <operator>=</operator>
           <rightSide>'4A92592959297602769797962'</rightSide>
        </Simple>
      </ConditionInfo>
      <ActionInfo>
        <name>File a Record</name>
      </ActionInfo>
       <ActionPreferenceInfos>
            <actionPreferenceInfo>
               <key>category_id</key>
               <value>LC</value>
            </actionPreferenceInfo>
         </ActionPreferenceInfos>
</RuleInfo>
  </RuleInfos>
</PolicyInfo>
```

Removing Records Management of Artifacts in Oracle Beehive

Filing an artifact as a record makes it immutable when it is under records management control. Conversely, this means that no Oracle Beehive command or action may delete the artifact.

To release an existing artifact from records management control, you must use the `beectl delete_record` command. You must have the `RECORDS_MGR` privilege.

Note that the `beectl delete_record` command cannot be reversed or undone. It also does not delete the artifact from Oracle Beehive. Instead, it removes the designation of that artifact as a record, enabling other Oracle Beehive operations to handle it normally (including deleting).

Note also that if the artifact is in the Record Store and its records management is removed or released, that artifact is deleted from the system. Only deleted and purged records-managed artifacts are placed in the Record Store. When the record for the artifact is deleted, the earlier action of deleting the artifact completes.

To delete a record from Oracle Beehive:

Issue the `beectl delete_record` command:

```
beectl> delete_record --artifact artifact_to_delete_identifier
```

where:

- `artifact` is the identifier of the artifact that must be deleted

Troubleshooting Records Management Service Operations

This section describes actions you can take to help troubleshoot issues with Records Management in Oracle Beehive. If you cannot diagnose your problem using the following steps, then contact your Oracle support representative.

This section contains the following topics:

Record Filing Failed

Use the following steps as guidelines for troubleshooting cases where the filing of a record for an artifact failed to occur.

To troubleshoot a failed record filing:

1. If a policy-based record filing failed, then check the following:
   - Check to see if the policy was successfully registered, using the `beectl list_policies` command. If your policy is not listed, then it has not been created in Oracle Beehive.
   - Check that the policy was created at the correct scope and for the correct operation (event). Events include `DOCUMENT_CREATED`, `ES_MSG_SENT` for filing sent e-mail messages as records, and `ES_MSG_DELIVERED` for filing

received e-mail messages as records. Review the policy XML file for the scope and event attributes, ensuring there are no errors.

- Attempt to create a policy with the same triggering conditions and scope, but with a general (non record-filing) action.
- Test to see if this policy successfully runs when the triggering event occurs.

  Note that when writing policies for filing records of e-mail messages, you should be aware that a separate e-mail event is raised for each recipient of an e-mail. The event payload of each event contains the `recipient_eid` of the user.

  Note also that if you create a policy that depends on `recipient_eid`s with two or more different values, it will never evaluate to `true` because the policy only evaluates events that containing one recipient.

2. Ensure that the Records Management Service is running.

   Use the `beectl status --all_services` command, and check the status of the Records Management Service.

   Alternatively, run the `beectl list_file_plan` command. It completes successfully only if the Records Management Service is running.

3. Ensure that your Oracle Universal Records Management Server is running. You can verify this by logging in to Oracle URM as `sysadmin`.

4. Review the log files for the `BEEAP`P component, to see if the Records Management Event Action is triggered. Event processing is asynchronous, so there is a delay between the service raising an event (such as `document_created`) and the event service sending the event to the interested service (in this case the Records Management Service). If the log file does not show the Records Management Event Action, then it may be that the event was not dispatched.

   The log file is located at *`$ORACLE_HOME`*`/beehive/logs/oc4j/BEEAPP/log.txt`. You may have rotated logs of the format `log.txt.<number>`.

5. In the Oracle Beehive database, check the `ECA_FAILED_ACTION_DETAILS` in the `bee_code` schema for any exception message stored for event processing failures.

6. Log in to Oracle URM as `sysadmin`. Select Browse Content and Search for Records filed under the name of the Adapter that was used when you registered Oracle URM with Oracle Beehive. Try to locate the `CollabID` for the document or e-mail for which you are trying to file a record, in the list of returned records.

7. The Records Management Service audits all possible actions and errors. You can query the `AUDIT_RECORDS` table with the event names that should have triggered the records management policy, to see if there are any results.

8. Check the `ocs_logs` table in the `bee_code` schema. Messages are stored here in the event that auditing fails.

Removing Records Failed

Typically, deleting a record of an artifact is a synchronous manual operation, so the exception message returned by `beectl` should tell you what went wrong.

Use the following steps as guidelines for troubleshooting cases where the removal of an artifact's record failed to occur:

- Ensure that the Records Management Service is running. You can use the `beectl status --all_services` command, and check the status of the Records

Management Service. Alternatively, try running the `beectl list_file_plan` command: this command only completes successfully if the Records Management Service is running.

- Ensure that your Oracle Universal Records Management Server is running. You can verify this by logging in to Oracle URM as `sysadmin`.

Disposition Not Processed

Disposition processing is an asynchronous automatic processing performed by the Records Management Service. By default, Oracle Beehive loads and processes dispositions from Oracle URM once every hour.

Use the following steps as guidelines for troubleshooting cases where a record disposition failed to occur:

- Ensure that the Records Management Service is running. You can use the `beectl status --all_services` command, and check the status of the Records Management Service. Alternatively, try running the `beectl list_file_plan` command. This command completes successfully only if the Records Management Service is running.
- Ensure that your Oracle Universal Records Management Server is running. You can verify this by logging in to Oracle URM as `sysadmin`.
- Review the log files for the `BEEAPP` component and the `ocs_logs` table in the `bee_code` schema, to see if there are any messages that indicate dispositions have been fetched from Oracle URM and loaded in the Oracle Beehive database.

  If there are any dispositions listed, check the `AUDIT_RECORDS` table and `ocs_logs` table for disposition errors. If a disposition action fails, its status will be changed to `ERR`. The `disposition_exception` column indicates the cause of the failure, along with a full stack trace.

Oracle URM Login, Password, or URL Incorrect or Changed

During the procedure outlined in "Step 2: Registering Oracle URM and Configuring Oracle Beehive", if the `beectl add_urm` command was specified with incorrect `rm_admin_name`, `rm_admin_password`, or `urm_url` properties, or if these values change (such as if you change the Oracle URM admin password), you can use the `beectl modify_property` command to change the values:

```
beectl> modify_property --component component_id_URM_connector_from_add_urm
                        --name property_name
                        --value new_value
```

You can find the component ID by using the `beectl list_components` command:

```
beectl> list_components --type your_company_URM
```

You can then list the properties of the component using the `beectl list_properties` command:

```
beectl> list_properties --component component_identifier
```

After making changes to component properties, you must run the `beectl activate_configuration` command to validate and activate your proposed configuration:

```
beectl> activate_configuration
```

Configuring Oracle URM for Dispositions Testing

Oracle URM processes dispositions in batches. The impact of this is that disposition tasks are not necessarily available at any given time. For convenient testing of dispositions in Oracle Beehive, you can make the following configuration changes:

1. From the Oracle URM administrator UI, click **Administration**, select **Configure Records Management**, choose the **Audit** tab, and select **Checked-in Audit Entries**.
2. On the option screen that opens, click the link for **Default Metadata for Checked-in Audit Entries**.
3. A check-in screen will open. Enter a value in the required **Title** field for the title of the checked in Audit logs and click the **Submit** button.
4. Select a retention category, and the system will list records checked in under this retention category by a given agent.
5. Choose a few documents, click **Actions**, select **trigger dates**, and select **Cancel**.
6. Wait for one to two minutes for the Oracle URM table to get updated. Then from **Administration**, select **Configure Records Management**, select **Batch Services**, and select **Run All**.
7. Click **My Content Server** and select **My Records Assignments**. All due dispositions appear on the list.
8. Use the actions icon to the right of a particular disposition to approve it. After, the disposition is in a pending completion state.

   To see the disposition, click **Pending Completion** tab and choose **My Completed Option** for the external source. This approves the disposition and shows it as a pending disposition for the Oracle Beehive adapter.

8

Integrating Oracle Enterprise Manager Grid Control with Oracle Beehive

This module describes how to integrate Oracle Enterprise Manager Grid Control with Oracle Beehive.

This module contains the following topics:

Overview of Integrating Grid Control with Oracle Beehive

Oracle Beehive supports the following scenarios for integrating with Grid Control:

- Installing Oracle Beehive through Grid Control. See "Installing Oracle Beehive Through Grid Control" on page 8-2.
- Installing Oracle Beehive separately from Grid Control. See "Installing Oracle Beehive Separately" on page 8-2.

In either scenario, after you complete the installation, you can monitor Oracle Beehive through Grid Control. See "Administering Your Oracle Beehive Integration with Grid Control" on page 8-2.

Prerequisites for Integrating Grid Control with Oracle Beehive

For Oracle Beehive installations using Grid Control, ensure that you have the following prerequisites in place before you begin the integration:

- You must have Oracle Enterprise Manager 10*g* Release 5 Grid (10.2.0.5.0) or later.
- If you want to install Oracle Beehive through Grid Control, you must have Oracle Beehive Provisioning Application. This application enables you to install and configure Oracle Beehive products from Oracle Enterprise Manager Grid Control. See "Installing Oracle Beehive Through Grid Control" on page 8-2 for more information.
- You can deploy the following Oracle Beehive products with the Oracle Beehive Enterprise Deployment Procedure:
  - Oracle Beehive
  - Oracle Beehive for DMZ

- Oracle Beekeeper
- Oracle Coexistence Connector for Microsoft Exchange Server
- Oracle Coexistence Connector for Lotus Domino Server

Procedures for Integrating Grid Control with Oracle Beehive

This section contains:

Installing Oracle Beehive Through Grid Control

You can install, and then afterwards monitor, Oracle Beehive through Grid Control. To accomplish this, you use Oracle Beehive Provisioning Application. See "Installing Oracle Beehive Provisioning Application to Allow Oracle Beehive to Be Provisioned Through Oracle Enterprise Manager Grid Control" in the following guides for more information:

- *Oracle Beehive Installation Guide for Linux*
- *Oracle Beehive Installation Guide for Microsoft Windows*
- *Oracle Beehive Installation Guide for Solaris Operating System (SPARC 64-Bit)*

Installing Oracle Beehive Separately

To install Oracle Beehive separately, see the *Oracle Beehive Installation Guide* for your operating system. After you have installed Oracle Beehive separately, you can monitor it through Grid Control.

Administering Your Oracle Beehive Integration with Grid Control

To monitor Oracle Beehive through Grid Control, such as using Grid Control to monitor Oracle Beehive targets, see Chapter 12, "Oracle Beehive Management," in *Oracle Enterprise Manager Concepts*.

9

Integrating Oracle Secure Enterprise Search 10*g* with Oracle Beehive

This module describes how to integrate Oracle Secure Enterprise Search Release 10*g* with Oracle Beehive.

This module contains the following topics:

About Integrating Oracle Secure Enterprise Search with Oracle Beehive

Oracle Beehive maintains an optimized search index that enables users to perform comprehensive searches across all Oracle Beehive artifacts. At the enterprise level, however, other information repositories might exist and contain information that users need. For example, depending on their roles, knowledge workers might need to find expense reports or purchase requisitions stored outside of Oracle Beehive. This level of search across all enterprise information repositories is provided by Oracle Secure Enterprise Search.

Oracle Secure Enterprise Search has been designed as a stand-alone enterprise search solution. It incorporates best-in-class indexing, crawling, and security capabilities to create a reliable and comprehensive search solution for any organization.

To use this powerful option requires the completion of tasks (post-installation) that are specific to Oracle Beehive and others that are specific to Oracle Secure Enterprise Search 10*g*. For example, your Oracle Beehive instance must be configured in Oracle Secure Enterprise Search 10*g* as a federated data source.

You use the `beectl` utility to complete the tasks that are specific to Oracle Beehive, which include:

- Creating a special user account that has administrator rights to the content managed by Oracle Beehive
- Configuring the Secure Enterprise Search service by entering the host and port number of your Oracle Beehive instance, and by enabling the service
- Activating these changes using the `activate_configuration` command

For detailed information about using the `beectl` utility, see "Managing Oracle Beehive Using beectl" in *Oracle Beehive Administrator's Guide*.

You use the Oracle Secure Enterprise Search Administration Tool to complete the tasks that are specific to Oracle Secure Enterprise Search 10*g*, which include:

- Activating the identity plug-in on the Global Settings - Identity Management Setup page.
- Specifying your Oracle Beehive database instance as a federated source.
- Specifying the Uniform Resource Identifier (URI) of the Oracle Beehive Search Service.
- Entering the credentials of the special user account that you created in Oracle Beehive.

For more information about the tasks that are specific to Oracle Secure Enterprise Search 10*g*, see "Setting Up Federated Sources" in the *Oracle Secure Enterprise Search 10g Administrator's Guide*.

Note: additional integration details are mentioned in the section "Configuring Oracle Secure Enterprise Search (SES) Integration" in the *Readme* of the patchsets.

Prerequisites for Integrating Oracle Secure Enterprise Search

Oracle Beehive supports integration with Oracle Secure Enterprise Search or later.

Procedure for Integrating Oracle Secure Enterprise Search

This section contains:

- Step 1: Add Oracle Secure Enterprise Search to Oracle Beehive
- Step 2: Configure Oracle Secure Enterprise Search for Oracle Beehive

Step 1: Add Oracle Secure Enterprise Search to Oracle Beehive

1. Optionally, if you are not in shell mode, from the `$ORACLE_HOME/beehive/bin` directory, obfuscate the password that you will enter in Step 2.

   For example:

   ```
   beectl obfuscate <expiration_time_in_minutes>
   ```

   The default for `expiration_time_in_minutes` is `30`. If you do not want the obfuscation to expire, then set this value to `0`.

2. Create and register a trusted identity account.

   This account will have administrative privileges for the content managed by Oracle Beehive.

   If you are not in shell mode, then enter the `--obfuscated` option after the password entry. If you are in shell mode, then omit the `--obfuscated` option but use the same `add_trusted_identity` syntax as shown in the following example.

   For example:

   ```
   beectl add_trusted_identity
     --is_service false
     --service_name sessearch
   ```

```
  --type SES
  --name BeehiveSesTrustedEntity
  --password <password> --obfuscated
```

Enter the arguments all on one line. The examples in this guide are formatted as shown here for easier readability.

For the `password` argument, enter any valid password. Remember the trusted identity account name and password, because you will need them in "Step 2: Configure Oracle Secure Enterprise Search for Oracle Beehive".

3. Configure the host and port number of your Oracle Beehive instance, as follows:

```
beectl modify_property
  --component _SesEndptService
  --name Host
  --value <Host name of your Oracle Beehive instance>

beectl modify_property
  --component _SesEndptService
  --name Port
  --value <Oracle Beehive HTTP listening port>

beectl modify_property
  --component _SesEndptService
  --name SesEndptServiceEnabled
  --value true
```

To determine the HTTP listening port, run the `beectl list_ports` command and then search for the property name `HttpListenPort`.

4. Activate and commit changes:

```
beectl> activate_configuration
beectl> modify_local_configuration_files
```

Step 2: Configure Oracle Secure Enterprise Search for Oracle Beehive

1. Log in to the Oracle Secure Enterprise Search administrator page.

   The URL typically has the following form:

   ```
   http://<Oracle SES host name>:<HTTP listening port>/search/admin
   ```

2. Click the **Sources** tab.
3. From the **Source type** drop-down list, select **Federated**.
4. Click the **Create** button.
5. Enter the following values to define the source:
   - **Source Name**: Enter any name to identity your source.
   - **Web Services URL**: Enter the Web Services URL in the following format:

     ```
     http://<Oracle Beehive host name>:<Oracle Beehive HTTP listening
     port>/ses-endpt/OracleSearch
     ```

   - **Remote Entity Name**: Enter the entity name that you created in Step 2 under "Step 1: Add Oracle Secure Enterprise Search to Oracle Beehive".

     ```
     BeehiveSesTrustedEntity
     ```

- **Remote Entity Password**: Enter the password for the BeehiveSesTrustedEntity user that you created in Step 2 under "Step 1: Add Oracle Secure Enterprise Search to Oracle Beehive".
- **Search User Attribute**: Enter a value for this field only if Oracle Secure Enterprise Search uses a different authentication attribute than Oracle Beehive. Otherwise, leave this field blank.

10

Integrating Symantec Scan Engine with Oracle Beehive

This module describes how to integrate Symantec Scan Engine with Oracle Beehive.

This module contains the following topics:

About Integrating Symantec Scan Engine with Oracle Beehive

You can integrate Symantec Scan Engine version 5.1.2 or later with Oracle Beehive. This integration enables your organization to use existing Symantec Scan Engine instances for anti-virus features beyond those that Oracle Beehive provides. After you complete the integration, your Oracle Beehive system can use the Symantec Scan Engine scan types, modes, artifact, and filtering capabilities.

The integration automatically includes Device Management Service (DMS), which which uses Symantec Scan Engine to scan uploaded client application zip files or downloaded client application binary files. During the upload of a client application zip file, if a virus is found, then DMS cancels the upload operation. During the download of a client application binary file, if a virus is found, then DMS performs one of the following actions:

- If the scan policy is set to scan and repair, then DMS attempts to remove the virus from the downloaded file. If it cannot repair the file, then DMS cancels the download operation.
- If the scan policy is set to scan only, then DMS cancels the download operation.

To manage the Oracle Beehive Virus Scanner, you use the `beectl` utility and Oracle Beekeeper. (For detailed information about using the `beectl` utility, see in *Oracle Beehive Administrator's Guide*.) To manage Symantec Scan Engine, use the Symantec Scan Engine tools.

The procedure for integrating Symantec Scan Engine with Oracle Beehive is to first add Symantec Scan Engine to Oracle Beehive, and then to enable the virus scanning functionality. You can perform the integration while Symantec Scan Engine is running virus scans, but be aware that the configuration process will add to the load of the scan engine, and may affect performance.

Prerequisites for Integrating Symantec Scan Engine with Oracle Beehive

Before you integrate Symantec Scan Engine with Oracle Beehive, ensure that the Oracle Beehive Server can communicate with the Symantec Scan Engine server, through the ICAP listening port of the scan engine. The Symantec Scan Engine administrative ports (typically 8004 and 8005) must be open on the scan engine server in order to allow access to the Symantec administrative console, which is used for configuring, administering, and running reports on the scan engine. This enables the Scan server to communicate with the Symantec Scan Engine site so that it can download the definitions.

Procedure for Integrating Symantec Scan Engine with Oracle Beehive

This section contains:

Step 1: Adding a Symantec Scan Engine to Oracle Beehive

To add Symantec Scan Engine to Oracle Beehive:

1. Ensure that Symantec Scan Engine version 5.1.2 or later is installed.

   You can find the version from the Symantec Scan Engine administrative console. Alternatively, run the following command:

   ```
   <SYMANTEC_INSTALL_LOCATION>/bin/symcscan.sh version
   ```

2. Add the first Symantec Scan Engine instance to Oracle Beehive.

   For example:

   ```
   beectl> add_virus_scan_engine
     --hostname my_symantec_server.example.com
     --port 6002
     --validate_connection true
   ```

   Enter the arguments all on one line. The examples in this guide are formatted as shown here for easier readability.

   In this example:

   - `hostname`: Name of the host computer where Symantec Scan Engine is installed. Enter the fully qualified name (for example, `my_symantec_server.example.com`).
   - `port`: Port number of the ICAP port used by the Symantec Scan Engine host computer.
   - `validate_connection`: Checks the ICAP port to ensure that the connection is valid. The addition of the scan engine object to the configuration system will succeed even if the validation for connectivity fails. If you have validated the connectivity earlier or through other means, if you are doing an Oracle Beehive configuration before the Symantec Scan Engine is installed or running, or if you plan to validate at a later date, then you can omit this option.

     Enter `true` to validate the connection; otherwise, enter `false`.

3. Repeat Step 2 for each Symantec Scan Engine instance that you want to configure with Oracle Beehive.
4. Specify a virus scan policy for Oracle Beehive.

   For example:

   ```
   beectl> modify_virus_scan_policy --scanpolicy scan_and_repair
   ```

   The following are valid values for the `scanpolicy` setting:

   - `NO_SCAN_OR_REPAIR`. Disables virus scanning. This setting is the default.
   - `SCAN_ONLY`: Oracle Beehive e-mail uses the `SCAN_ONLY` setting, regardless of the setting you enter here. Beehive e-mail has a built-in repair function that removes the infected portions and attachments from e-mail messages.
   - `SCAN_AND_REPAIR`: This setting has a larger performance impact than the `SCAN` setting. However, of the two Oracle Beehive services that currently use Symantec scanning, only DeviceManagementService uses the full repair feature for downloaded or uploaded client application modules that have been infected with viruses.
5. Validate and activate the configuration.

   ```
   beectl> activate_configuration
   ```

6. Enable the virus scanning and/or attachment blocking for the Symantec Scan Engine-Oracle Beehive configuration.

   See "Step 2: Enabling the Symantec Scan Engine Virus Scanning or Attachment Blocking", next.

Step 2: Enabling the Symantec Scan Engine Virus Scanning or Attachment Blocking

This section contains:

About Enabling the Symantec Scan Engine Virus Scanning or Attachment Blocking

After you have added Symantec Scan Engine to Oracle Beehive, then you are ready to perform one or both of the following tasks:

- Enable virus scanning, which makes the message body and attachments of your Oracle Beehive e-mail system available for virus scans.
- Enable attachment blocking, which prevents certain types of files, such as those with the extension `.zip` or `.exe`, from being attached to e-mails.

If you have enabled virus scanning, attachment blocking, or both, then you can customize the notification that is sent to e-mail recipients when a virus scan is performed. The procedures in this section describe how to perform this customization.

Enabling the Integrated Symantec Scan Engine Virus Scanning

To enable virus scanning:

1. Ensure that you have added Symantec Scan Engine to Oracle Beehive, as described in "Step 1: Adding a Symantec Scan Engine to Oracle Beehive" on page 10-2.

If you enable virus scanning but do not have Symantec Scan Engine configured, then Oracle Beehive may prevent delivery of e-mail messages while it stores them in a queue and waits for Symantec Scan Engine to respond. This behavior is intended to prevent delivery of unscanned messages in the event an external virus scan engine becomes nonresponsive.

2. Log in to Oracle Beekeeper.
3. In the Services box, select **Email**.
4. In the Email pane, select the **Configuration** tab, and then click the **Edit** button.

   A separate edit window appears.
5. In the edit window, select the **Transport Properties** tab.
6. Under **Post Resolution Rules**, expand the **Virus Scanning** section.
7. Select the **Activate virus scanning** checkbox.

   After the **Virus Scanning** region expands to include the following options, make the appropriate selections, as follows:

   - **Notify local senders about virus** check box
   - **Notify remote senders about virus** check box
   - **Notified administrators** check box
   - **Notifier Email** field, in which you enter the e-mail address of the person sending the e-mail notification
   - **Subject** field, in which you enter a subject header
   - **Message**, in which you enter a message letting users know that Oracle Beehive detected a virus in their e-mails
8. Click **Apply** to apply the proposed configuration without closing the configuration window, or click **Save & Close** to apply the proposed configuration and close the window.
9. To activate the configuration, in the System box, select **Configuration Control**, and then click **Activate**.

Administering Your Symantec Scan Engine Integration

This section contains:

Validating the Symantec Scan Engine Connectivity

At any time, you can validate the Symantec Scan Engine connectivity by using the `beectl validate_virus_scan_engine_connectivity` command. The syntax is as follows:

```
beectl> validate_virus_scan_engine_connectivity
[ --hostname <scanengine_hostname>]
[ --port <scanengine_port> ]
```

For example:

```
beectl> validate_virus_scan_engine_connectivity
 --hostname my_symantec_server.example.com
 --port 6002
```

If you omit the `hostname` and `port` arguments, then the `validate_virus_scan_engine_connectivity` command restricts the search to the local computer, with the assumption that `hostname` is `localhost` and `port` is `1344`.

Creating a Symantec Scan Engine Cluster Configuration

A cluster configuration is a set of Symantec Scan Engine engines of the same type, that is, Symantec, that run on different `server:port` combinations. Oracle Beehive can connect to any of these engines, with preference given to the one running on the local computer. Virus scanning policies are defined at the cluster level. When you add the first Symantec Scan Engine, Oracle Beehive creates the cluster. A Site can have only one cluster.

To create a Symantec Scan Engine cluster configuration:

1. Ensure that you have completed the virus scanning configuration as described in "Step 2: Enabling the Symantec Scan Engine Virus Scanning or Attachment Blocking" on page 10-3.
2. Log in to Oracle Beekeeper.
3. In the System box, select **Topology**.

   By default, the Topology tab is displayed. If you want a more granular selection, then select the **By Service** tab.
4. In the Topology pane, expand the target hierarchy until all of the Oracle Beehive instances appear.
5. Select the Site level node, and then from the list of target hierarchies, select the site that you want.
6. From the **View** menu, select **Configuration**.

   The Topology pane changes to indicate the root hierarchy item you selected.
7. Select the **Virus Scan Engine Cluster** tab, and then select the **Edit** button.

   An edit window appears.
8. In the edit window, select the **Virus Scan Engine Cluster** tab, and then click the **Create Virus Scan Engine Cluster** button.
9. Enter the following settings:
   - **Alias**: Enter an alias for this group of scan engines.
   - **Virus Scan Policy**: Select from the following options:

- **NO_SCAN_OR_REPAIR** (default)
- **SCAN_ONLY**
- **SCAN_AND_REPAIR**

10. To access additional advanced parameters, click the **Advanced** link.
11. In the **ScanEngines** section, click the plus icon to add one or more scan engines. For each scan engine, enter the **Scan Engine Host Name** and **Scan Engine Client Comm Port**, and optionally, enter an **Alias**.
12. Click **Apply** to apply the proposed configuration without closing the configuration window, or click **Save & Close** to apply the proposed configuration and close the window.
13. To activate the configuration, in the System box, select **Configuration Control**, and then click **Activate**.
14. Click **Apply**. Alternatively, click **Save & Close**.

Enabling the E-Mail Attachment Blocking

To enable attachment blocking:

1. Log in to Oracle Beekeeper.
2. In the Services box, select **Email**.
3. In the Email pane, select the **Configuration** tab, and then click the **Edit** button.

   A separate edit window appears.
4. In the edit window, select the **Transport Properties** tab.
5. Under **Post Resolution Rules**, expand the **Attachment Blocking** section.
6. Select the **Activate attachment blocking** checkbox.

   After the **Attachment Blocking** region expands to include the following options, make the appropriate selections, as follows:

   - **Process only emails from**, from which you select from the **Origin** list
   - **Remove attachments of Type**, from which you select from the **Extension** list
   - **Notify local senders about bad attachments** check box
   - **Notify remote senders about bad attachments** check box
   - **Notifier Email** field, in which you enter the e-mail address of the person sending the e-mail notification
   - **Subject** field, in which you enter a subject header
   - **Message**, in which you enter a message letting users know that Oracle Beehive detected a virus in their e-mails
7. Click **Apply** to apply the proposed configuration without closing the configuration window, or click **Save & Close** to apply the proposed configuration and close the window.
8. To activate the configuration, in the System box, select **Configuration Control**, and then click **Activate**.

See Also: "Customizing E-Mail Notifications That Have Blocked Attachments or Viruses" on page 10-7

Customizing E-Mail Notifications That Have Blocked Attachments or Viruses

To customize the notification to addressees (intended recipients) of messages with blocked attachments or viruses:

1. Log in to Oracle Beekeeper.
2. In the Services box, select **Email**.
3. In the Email pane, select the **Configuration** tab.
4. Click the **Edit** button.

   An edit window appears.
5. In the edit window, select the **Transport Properties** tab.
6. Under Post Resolution Rules, expand the **Virus & Bad Attachment Notification** section.
7. Under Notification Message, complete the **Subject** and **Notification** fields.
8. Click **Apply** to apply the proposed configuration without closing the configuration window, or click **Save & Close** to apply the proposed configuration and close the window.
9. To activate the configuration, in the System box, select **Configuration Control**, and then click **Activate**.

Reviewing Virus Scan Results

The virus scan captures the following types of infection details:

- Virus name
- Virus ID
- Entity Identifier (Entity ID, Entity Type, CollabID)
- Date of the scan
- Status of whether the virus was removed or not
- Number of attempts to repair the scanned entity
- Component within a container (if the scanned entity was a container or multi-part mime message)

You can use the `beectl list_virus_scan_results` command to review the results of virus scans.

The syntax is as follows:

```
beectl> list_virus_scan_results
[ --scandate <scandate> ]
[ --scandate_from <scandate_from> ]
[ --scandate_to <scandate_to> ]
[ --virus_name <virus_name> ]
[ --virus_id <virus_id> ]
[ --obsolete_only <true/false> ]
[ --entity_type <entity_type> ]
[ --maximum_results <max_results> ]
[ --display_columns <display_columnns> ]
[ --count_only <true/false> ]
```

If you omit the arguments, then Oracle Beehive lists up to 2000 of the scan results available in the Beehive database.

In this specification:

- `--scandate`: Specifies an exact date-time for the scan. Optional. For example, the following setting specifies one second before midnight on December 12, 2009:

  ```
  --scandate "2009-12-10T23:59:59"
  ```

  Enclose the date in double quotation marks. Permitted formats are as follows:

  - `YYYY-MM-dd'T'HH:MM:SS.SS'Z'`
  - `YYYY-MM-dd'T'HH:MM:SS.SS`
  - `YYYY-MM-dd'T'HH:MM:SS'Z'`
  - `YYYY-MM-dd'T'HH:MM:SS`
  - `YYYY-MM-dd'Z', YYYY-MM-DD`

  If you want the scan to cover a period of time, then use the `scandate_from` and `scandate_to` arguments instead of `scandate`.

- `--scandate_from` and `--scandate_to`: Specify a time range for the results set, using the same formats that the `scandate` argument uses. Only the results of scans conducted on the specified dates will be shown. For example, to scan the entire day of December 12, 2009, enter the following settings:

  ```
  --scandate_from "2009-12-02T00:00:00" --scandate_to "2009-12-02T23:59:59"
  ```

  Use the same time formats as the `scandate` option.

- `--virus_name`, `--virus_id`: Return the results for the exact specified virus name or ID. Enclose these settings in double quotation marks. For the `virus_name` setting, the name typically ends in a semi-colon (`;`).

- `--entity_type`: Returns the results for the given type of entity scanned. Enter one of the following values:

  - `emsg`: E-mail messages
  - `capm`: Client application module

- `--obsolete_only`. Specifies whether to return the results where the entity scanned (for example, an e-mail message) has been removed from the system (deleted). Enter `true` or `false`.

- `--maximum_results`: Specifies a maximum number of results to be returned by the query.

- `--display_columns`: Limits the information to be displayed about each result. Enter a comma-delimited list of values, and enclose the group in a set of double quotation marks. Valid choices are as follows:

  - `virus_id`
  - `virus_name`
  - `component_name`
  - `scan_date`
  - `repair_attempts`
  - `entity_id`
  - `entity_type`
  - `repaired`

  - `aux_data`
  - `collab_id`

- `--count_only`: A setting of `true` returns a count of the results that match; otherwise, enter `false`. If you set `count_only` to `true`, then do not use the `maximum_results` and `display_columns` arguments.

For example:

```
beectl> list_virus_scan_results
 --scandate_from "2009-12-02T00:00:00"
 --scandate_to "2009-12-02T23:59:59"
 --virus_name "Encrypted container deleted;"
 --virus_id "\-9"
 --obsolete_only true
 --entity_type capm
 --maximum_results 100
 --display_columns "virus_id, virus_name,component_name, repaired"
 --count_only false
```

Deleting a Symantec Scan Engine Configuration

To delete a Symantic Scan Engine configuration:

1. If Oracle Beehive e-mail is configured to have the virus scan rule enabled, then de-activate this virus scan rule.

   If you delete a scan engine configuration when the virus scan rule and e-mail service configuration settings are enabled, then e-mail deliveries can fail.

   Run the following command to de-activate the virus scan rule:

   ```
   beectl> modify_scan_policy
    --scanpolicy NO_SCAN_OR_REPAIR
   ```

2. Disable virus scanning in the e-mail service configuration.
   a. Log in to Oracle Beekeeper.
   b. In the Services box, select **Email**.

      The Email window appears.
   c. Select the **Configuration** tab.
   d. Select the **Transport Properties** tab.
   e. Expand the **Post Resolution Rules** list. (It should be expanded by default.)
   f. Expand the **Virus Scanning** list.
   g. Under the **Configuration** tab, click the **Edit** button.

      A secondary window appears, showing the **Transport Properties** tab.
   h. Clear the **Activate virus scanning** checkbox.
   i. Click **Apply**. Alternatively, click **Save & Close**.
3. From the `beectl` utility, delete the scan engine configuration.

   For example:

   ```
   beectl> delete_virus_scan_engine
     --hostname my_symantec_server.example.com
     --port 6002
   ```

4. From Oracle Beekeeper, delete the cluster that was associated with the scan engine configuration:
   a. In the System box, select **Topology**.
   b. In the Topology pane, expand the target hierarchy until all of the Oracle Beehive instances appear.
   c. Select the Site level node, and then from the list of target hierarchies, select the site that you want.
   d. From the **View** menu, select **Configuration**.
   e. Select the **Virus Scan Engine Cluster** tab, and then select the **Edit** button.
   f. Click the **Remove Virus Scan Engine cluster** button.
   g. Click the **Apply** button to save the configuration, or click **Save & Apply** to save the configuration and close the window.
   h. From the System box, select **Configuration Control**.
   i. Click the **Activate** button.

Deleting Virus Scan Results

You can delete stored results from a virus scan. You should periodically delete results to avoid consuming an inordinate amount of space in the Oracle Beehive data store, which could affect performance. To delete virus scan results, use the `beectl delete_virus_scan_results` command.

Use the following syntax:

```
beectl> delete_virus_scan_results
[ --scandate <scandate> ]
[ --scandate_from <scandate_from> ]
[ --scandate_to <scandate_to> ]
[ --virus_name <virus_name> ]
[ --virus_id <virus_id> ]
[ --entity_type <entity_type> ]
[ --obsolete_only <true/false> ]
```

In this example:

- `--scandate`: Specifies an exact date-time for the scan to be deleted. Optional. For example, the following setting specifies one second before midnight on December 12, 2009:

  ```
  --scandate 2009-12-10T23:59:59
  ```

  Permitted formats are as follows:

  - `YYYY-MM-dd'T'HH:MM:SS.SS'Z'`
  - `YYYY-MM-dd'T'HH:MM:SS.SS`
  - `YYYY-MM-dd'T'HH:MM:SS'Z'`
  - `YYYY-MM-dd'T'HH:MM:SS`
  - `YYYY-MM-dd'Z', YYYY-MM-DD`

- `--scandate_from` and `--scandate_to`: Specify a time range for the scans to be deleted, using the same formats that the `scandate` argument uses. Only the results of scans conducted on the specified dates will be shown. For example, to scan the entire day of December 12, 2009, enter the following settings:

```
--scandate_from 2009-12-02T00:00:00 --scandate_to 2009-12-02T23:59:59
```

Use the same formats as the `scandate` option.

- `--virus_name`, `--virus_id`: Delete the results for the exact specified virus name or ID.
- `--entity_type`: Deletes the results for the given type of entity scanned. For e-mail messages, the entity type is `emsg`.
- `--obsolete_only`: Specifies whether to delete the results where the entity scanned (for example, an e-mail message) has been removed from the system (deleted).

11

Integrating Cisco Voice Gateway with Oracle Beehive Voicemail and Fax

Note: Enabling Oracle Beehive Voicemail functionality requires advanced configuration. Some necessary Cisco Voice Gateway configuration is not fully documented in this guide. Please contact your Oracle support representative for important configuration and security information related to deploying Oracle Beehive Voicemail.

Oracle Beehive Voicemail is provided by the Voice Message Service through integration with Cisco Voice Gateway. It enables a variety of functionality for Oracle Beehive users, including the ability to access and manage voice messages from a telephone or as audio files in the e-mail Inbox. Oracle Beehive Fax is provided by the Fax Message Service. This module describes how to set up and configure the necessary software components to enable the voicemail functionality. It contains the following sections:

- Introduction to Managing Oracle Beehive Voicemail
- Configuring Oracle Beehive Voice Message Service
- Configuring Oracle Beehive Fax

See Also:

- For more information on configuring the Voice Message Service, see "Managing the Voice Message Service" in the *Oracle Beehive Administrator's Guide*.
- For more information on configuring the Fax Service, see "Managing the Fax Message Service" in the *Oracle Beehive Administrator's Guide*.

Introduction to Managing Oracle Beehive Voicemail

This section includes the following topics:

- About Facilities
- About Auto Attendants
- About Voicemail Infrastructure

About Facilities

A Facility is an Oracle Beehive group, defined for a physical location that is connected to a common telecommunication infrastructure. This concept allows all users at that facility to have common attributes and be treated in a common way. For example, a facility can set the default language for the voicemail users of that facility. Users may also have their own unique preference properties defined which will override the facility properties.

Facilities are optional. You can set preferences and properties at the enterprise level, and those properties not specified at the facility level will default to the properties defined in the enterprise preference set. Facilities allow you the flexibility of having different settings at different physical locations, all grouped together in a logical configuration within the Oracle Beehive server. This provides the flexibility to define different telecommunication infrastructures associated with those physical locations.

About Auto Attendants

An Auto Attendant is an automated call-answering and routing server, which presents a collection of menus that are created for individual phone numbers supported at a facility. The menus allow callers to be routed to the correct department or extension, or expose additional recorded information (such as street directions or operation hours), and the corporate directory.

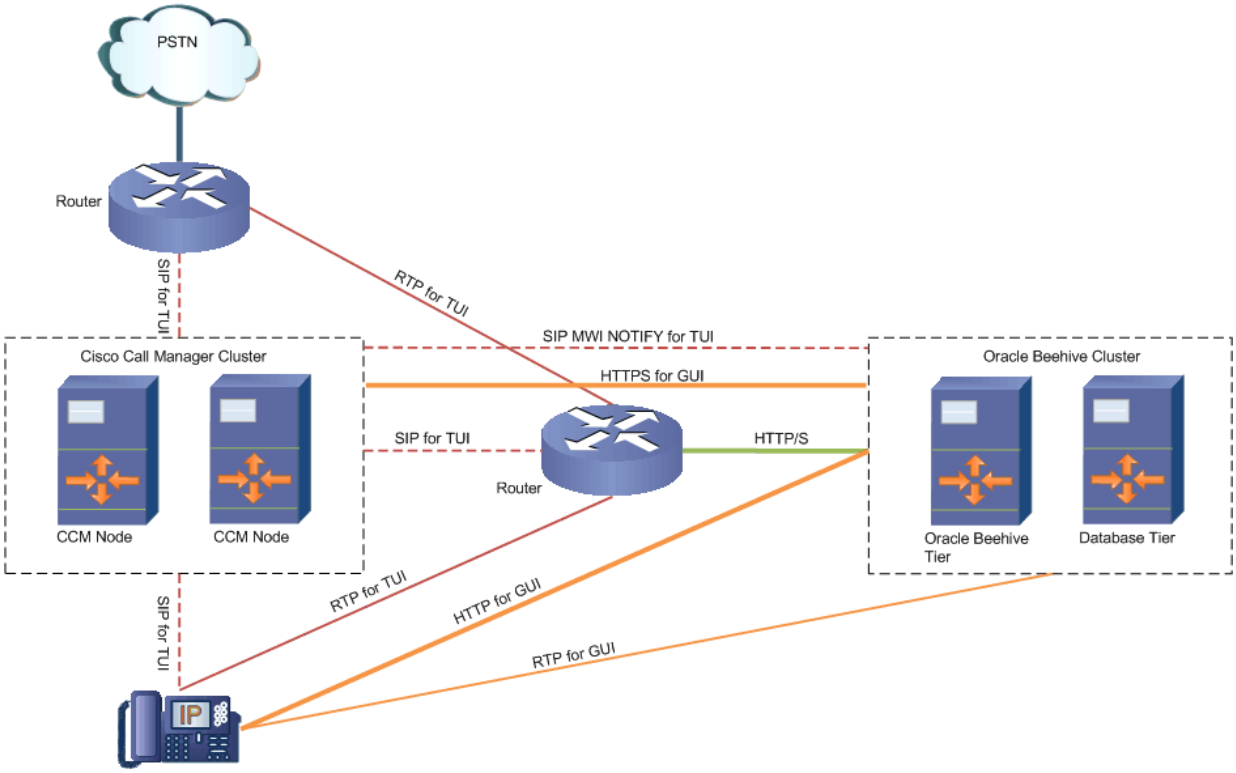
Auto Attendants are optional. You can deploy Oracle Beehive voicemail functionality with or without the use of Auto Attendants. You can have one Auto Attendant for each facility, or one for the whole enterprise.

About Voicemail Infrastructure

Oracle Beehive voicemail and automated attendant (AA) telephone user interface (TUI) use VoiceXML to present the TUI to the caller via a Cisco VoiceXML capable router. The Cisco router has a VoiceXML browser imbedded in the Cisco IOS operating system. VoiceXML is a W3C standards-based approach for voice applications and services, which leverages a Web-based development and deployment model instead of a proprietary telephony hardware and software approach. Since VoiceXML uses a Web based model, all the Oracle Beehive voice applications are executed on the server and only the VoiceXML and audio `.wav` files are served to the Cisco VoiceXML browser via HTTP or HTTPS.

Figure 11–1 shows an example deployment architecture linking Cisco Call Manager (CCM) hardware to Oracle Beehive.

Figure 11–1 Oracle Beehive Voicemail Centralized Deployment



This section contains the following topics:

- Cisco Dependencies and Requirements
- Voicemail UDS Requirements
- Voicemail Preference Properties

Cisco Dependencies and Requirements

To deploy Oracle Beehive Voicemail, Cisco VoiceXML-capable hardware and Cisco IOS VXML software is required. The supported Cisco VXML routers are the 2800 and 3800 Series Internet Service Routers (ISR), and the AS5350XM/AS5400XM Universal gateways. These routers offer a VoiceXML feature set in the Cisco IOS operating system version 12.4(15)T5 or higher, to execute the Oracle Beehive voice applications.

Please refer to your Cisco account representative to determine the VXML browser software licensing required for your environment and hardware. Cisco Unified Call Manager 5.0 or greater is the only IP PBX that has been tested with Oracle Beehive voicemail redirection.

Voicemail UDS Requirements

For an Oracle Beehive user to be active for voicemail, the user's UDS record must contain `voice_principal`, `voice_pin`, and `tel:` (telephone scheme) address attributes. The voice principal and telephone scheme address must be numbers containing no special characters or spaces. For complete documentation on configuring the appropriate user record attributes, see "Managing and Provisioning Oracle Beehive Users" in the *Oracle Beehive Administrator's Guide*. You can use the `beectl` command-line tool to add and modify preference profiles, users' voice properties, groups, and group properties.

Voicemail Preference Properties

Voicemail configuration properties follow the Oracle Beehive model of property inheritance: Enterprise, Facility, and User. This allows you to group configuration properties. Enterprise properties apply to all voicemail facilities and users. Facility properties apply to the users defined in the matching facility for that user. User properties are defined at the user level inside the user's UDS Voicemail Preference Profile. For example, the user property PreferedLocale will override the locale defined at the user's facility.

There are three types of preference properties:

- Enterprise Preference Properties
- Facility Preference Properties
- User Preference Properties

Enterprise Preference Properties Voicemail stores application configuration options in Oracle Beehive Preference Property Profiles. The enterprise properties for voicemail are stored in a Preference Profile called `VoiceEnterprisePrefs`. A default preference profile of VoiceEnterprisePrefs is created by default upon installation. Oracle recommends that you view and change these preference properties specific to your installation. You can view the properties within the preference set by using the `beectl list_preference_properties --set prfs=VoiceEnterprisePrefs,enpr=Oracle` command, for example. To add or modify a preference property, see the `beectl` command `add_preference_property` or `delete_preference_property`. Oracle recommends that you verify that the voicemail `RecordStreamURIs` and `RecordPlaybackURIs` enterprise properties are configured properly for the site installation prior to using the Voicemail Service, otherwise voicemail message recording and playback will fail if these properties are not set correctly for the site installation.

Facility Preference Properties Voicemail has the ability to define multiple facilities in order to decentralize the deployment of voicemail router resources offices or to define a logical grouping of preference properties for a physical location without having to run or administer multiple voicemail applications. A voicemail facility is a group of users and properties that share the same physical location (or logical grouping) and configuration information. You can create voicemail facilities by using the `beectl add_voice_facility` command. This command creates the voicemail user's phone number mapping to a UDS user group and facility properties preference set associated with the newly created facility. The default installation has a single facility configured against the `ALL_USERS` group, by matching all incoming phone phones to that facility. It is possible to use that default facility for voicemail, but directory lookup will not be available. Oracle recommends that you create a more specific phone number matching facility and either a static or dynamic user group mapped to that facility.

User Preference Properties User properties are defined inside the user's voicemail preference set. These properties can define the user's `PreferredLocale` and `ActiveGreetingType`.

To list a user's voicemail preference properties, use the `beectl list_preference_properties` command:

```
beectl> list_preference_properties --set prfs=VoiceMail,user=<userID>
```

Then, use the `beectl add_preference_property` command to set the `PreferredLocale` or `ActiveGreetingType` as needed. For example, to set the PreferredLocale to `en-us`:

```
beectl add_preference_property --set prfs=VoiceMail,user=<userID> --name
PreferredLocale --type string --value en-us
```

Note: You can set the number of secondary languages by using the `/.beectl add_preference_property` command. The maximum number of secondary languages that Oracle Beehive Voice Mail supports is four.

Configuring Oracle Beehive Voice Message Service

You must perform some configuration before Oracle Beehive voicemail functionality is fully enabled. At a minimum, you must configure the enterprise, and you optionally may configure one or more facilities. You must also ensure that all users who will use Oracle Beehive voicemail have required values for the relevant attributes in their user accounts.

If you want to enable the Message Waiting Indicator (MWI) and graphical user interface features (GUI) of your Cisco IP small-screen telephones, you must configure the correct voicemail enterprise and facility properties. If you want to use an auto attendant to answer and forward calls, then you must configure the auto attendant and associated voicemail facility properties.

This section contains the following topics:

- Configuring the Enterprise
- Creating Voicemail Users
- Managing Facilities
- Configuring the Voicemail Touch-tone User Interface (TUI)
- Enabling HTTPS for Cisco VXML Enabled Device Access to Oracle Beehive
- Configuring Cisco IP Phone Voicemail GUI Application
- Configuring the Voicemail GUI and Message Waiting Indicator
- Cisco Router Configuration
- Cisco Unified Call Manager Configuration
- Configuring the Auto Attendant

Voicemail Properties

Table 11–1, " Voicemail Properties" lists all the possible properties used by the Voicemail Service either at the Enterprise and Facility level. When configuring the voicemail preferences properties of an enterprise or facility, refer to this table for a description and examples.

Inheritance Rules

In general, the hierarchy is Enterprise -> Facility -> User, such that property values set at the lowest level are used first, and if no value is set, then the property value at the next-higher level is used. This general flow is not followed in some cases.

The following rules are applied for determining inheritance of property values:

- `UNDEFINED VALUES`

When the Enterprise value is not defined, then the default value is used. When the default value is not defined, then the Facility's value, if defined, is used. When neither value is defined, then the application determines a value, which may cause undesirable results.

- `BOOLEAN`

  When the Enterprise value is `true`, then the Facility's value, if set, is used. If not set, then `true` is used. If the Enterprise value is `false`, then all Facilities will use the value `false`, *unless it has been overridden by being specified at the Facility level*.

- `INTEGER`

  When the Enterprise value is `-1`, then the Facility's value is used.

- `STRING` and `STRING ARRAY`

  Both of these types behave as `UNDEFINED VALUES`

Table 11–1 Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
AxlConfigA	STRING	https://<CCM_HOST>:8443/axl/,https://<CCM_HOST>:8443/realtimeservice/services/RisPort,,,bhvmgui,password,f,f	none	The comma separated fields, in order, are: 1. The AXL URL 2. The realtime info URL 3. The proxy host 4. The proxy port 5. The user provisioned for AXL access (read only) 6. The password for the AXL user 7. Whether certificate validation is enabled for SSL connections to CCM 8. Whether hostname verification is enabled for SSL connections to CCM

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
AxlConfigB	STRING	https://192.188.175.105:8443/axl/,https://192.188.175.105:8443/realtimeservice/services/RisPort,,,bhvmgui,password,f,f STRING_LIST		This field's contents are the same as for AxlConfigA, and are used for backup AXL services running on another Cisco Call Manager Node
BusinessHourAttendantName	STRING			The name of the attendant to be used when the BusinessHours are matched.
BusinessHours	STRING			Defines the business hours used for the facility auto attendant. Represented as a list of seven comma-separated time intervals showing the open hours for each day of the week starting with Monday. Time intervals are in the format HHMM-HHMM in 24-hour notation. For example, ???0900-1700,0900-1700,0900-1700,0900-1700,,??? represent 9:00 AM to 5:00 PM, Monday through Friday.

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
CallBackSenderRule	INTEGER	4	0	Controls whether or not the voicemail's sender can be called back or not. 0 = No/Off 1 = Call backs to Beehive users only 2 = Call backs to phone numbers in Allowed Numbers 3 = Call backs to either Beehive or Allowed Numbers 4 = Always / On (no restrictions)
ChooseLanguageFirst	BOOLEAN	true	false	If this value is true, then certain dialogs will present the caller with "For English press 1, For French press 2, ..." rather than a UI that simply presents the prompts in both languages: "[English] Please enter your mailbox number. [French] Please enter your mailbox number..."
DefaultAttendantName	STRING			The name of the auto attendant to be used when none of the BusinessHours, Holidays, or SpecialDates are matched.
ExclusiveAudioContent URIs	STRING ARRAY		audio	The default value is a relative audio path

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
ExtensionTranslationRules	STRING ARRAY	`13125550???|1650506????`		The ExtensionTranslationRules is a pipe-delimited (|) String containing simplistic rules for defining how extensions are expanded. A caller may enter 3-5 digits to indicate an extension. The extension is matched against the patterns provided and if a match is found, then the full set of digits is returned. For example, the extension 61234 would return 16505061234. While 789 would return 13125550789.
HolidayAttendantName	STRING			The name of the attendant to be used when the HolidayHours are matched.
HolidayHours	STRING			Defines the holidays for the facility auto attendant. Represented as a comma-separated list of dates and time intervals representing holidays. For example, ???112707:0000-2359,123107:1400-2359??? represents all day on Thanksgiving Day and after 2:00 PM on New Years Day 2007.

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
IpPhoneHttpProxyHost	STRING			This only needs to be set if the Oracle Beehive tier needs to communicate through a proxy to reach the IP phones' HTTP server
IpPhoneHttpProxyPort	INTEGER			This only needs to be set if the Oracle Beehive tier needs to communicate through a proxy to reach the IP phones' HTTP server
IpPhonePassword	STRING	<bhvmgui_ password>		Password of the account defined in Cisco Call Manager which has device control over the users telephones for audio playback and return call functionality in Voicemail GUI
IpPhonePasswordAlgorithm	STRING			This only needs to be set if local IP phone authentication is used. Oracle recommends you use Cisco Call Manager Administrative XML Layer (AXL) configuration for IP authentication
IpPhonePasswordNumBits	INTEGER			This only needs to be set if local IP phone authentication is used. Oracle recommends you use Cisco Call Manager Administrative XML Layer (AXL) configuration for IP authentication

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
IpPhonePasswordSeed	STRING			This only needs to be set if local IP phone authentication is used. Oracle recommends you use Cisco Call Manager Administrative XML Layer (AXL) configuration for IP authentication
IpPhoneRtpMaxPort	INTEGER			The maximum port in the range defined for the Cisco IP Phone.
IpPhoneRtpMinPort	INTEGER			The minimum port in the range defined for the Cisco IP Phone.
IpPhoneUserName	STRING	bhvmgui		Username of the account defined in Cisco Call Manager which has device control over the users' telephones for audio playback and return call functionality in the Voicemail GUI
isExtensionDialingEnabled	BOOLEAN		true	Enables extension dialing in auto attendant applications
isGlobalLookupFallbackEnabled	BOOLEAN	true	true	This value provides the directory lookup module the ability to find users in the global directory if no match is found in the local facility lookup.

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
isMWIEnabled	BOOLEAN	true	true	Enables or disables users' Message Waiting Indicator (MWI)
isOperatorConfigured	BOOLEAN	true	true	Determines whether a caller can transfer to a live operator from the Touchtone User Interface (TUI)
isPhoneNumberPresentable	BOOLEAN	true	true	This value enables or disables the playing of phone numbers of incoming voicemail messages when the voicemail header is read inside the TUI.
isRetrievalEnabled	BOOLEAN	true	true	Determines whether users are allowed to retrieve their messages
MaxRecordingDuration	INTEGER		-1	Specifies the maximum duration of a recording, in seconds. -1 means that the Facility's value will be used. If not set at the Facility level, then 180 is used.
MwiAlgorithm	STRING	SHA1PRNG		Defines the algorithm used to create the SIP NOTIFY messages for MWI which must be set
MwiCcmHost	STRING			The IP address of the Cisco Call Manager SIP trunk configured to accept SIP NOTIFY messages

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
MwiCcmPort	INTEGER		5060	The port of the Cisco Call Manager SIP trunk configured to accept SIP NOTIFY messages
MwiMaxLocalPort	INTEGER	5080		Highest port in the range that Oracle Beehive will use to send and receive SIP messages
MwiMinLocalPort	INTEGER	5061		Lowest port in the range that Oracle Beehive will use to send and receive SIP messages
MwiSipProxyHost	STRING			SIP Proxy Host that is authorized to send SIP messages to Cisco Call Manager SIP MWI trunk
MwiSipProxyPort	INTEGER			SIP Proxy Port that is authorized to send SIP messages to Cisco Call Manager SIP MWI trunk
MwiSourcePhone	STRING	16505551234		Voicemail DNIS phone number that is presented in the SIP NOTIFY message

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
OperatorTransferNumber.	STRING	sip:16505551234@192.168.1.10 or tel:+16505551212 NOTE: These values are dependent upon the Cisco router configuration and CUCM dialplan implementation specific to the site.		Determines the Operator Phone Number to be used when a caller presses 0 to be transfer to an Operator. This can be a full SIP URI or tel:+ URI
RecordPlaybackURIs	STRING ARRAY		http://<host>:<port>/voice-servlet/vmail/data/shared/playback	The URI used inside the voicemail service to define the fully qualified URIs inside the VXML. If a load balancer is used, this must be defined to the load balancer hostname.
RecordPlaybackURIs	STRING ARRAY		http://<host>:<port>/voice-servlet/vmail/data/shared/playback	Cisco only supports audio streaming using HTTP. For performance reasons, it is recommended that you use HTTP and not HTTPS.
RecordStreamURIs	STRING ARRAY		http://<host>:<port>/voice-servlet/vmail/crs	The URI used inside the voicemail service to define the fully qualified URIs inside the VXML. If a load balancer is used, this must be defined to the load balancer hostname.

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
RecordStreamURIs	STRING ARRAY		http://<host>:<port>/voice-servlet/vmail/crs	Cisco only supports audio streaming using HTTP. For performance reasons, it is recommended that you use HTTP and not HTTPS.
RetrieveMsgMenuLocale	STRING	en-US		The locale to use when messages are being retrieved by a caller. This locale is always used at the login, but the user's locale, if specified, will be used after he or she is authenticated
RtpServerHost	STRING	beehive.example.com		The mid-tier host that runs the RTP Server used to stream audio to Cisco IP phones in the voicemail GUI service
RtpServerHttpProxyHost	STRING			This value must be set if the RtpServer has been deployed on a standalone host which can only be accessed by the Oracle Beehive tier through a proxy
RtpServerHttpProxyPort	INTEGER			This value must be set if the RtpServer has been deployed on a standalone host which can only be accessed by the Oracle Beehive tier through a proxy
RtpServerMaxPort	INTEGER		32768	Highest port in the range used by the RTP server to send RTP packets

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
RtpServerMinPort	INTEGER		20480	Lowest port in the range used by the RTP server to send RTP packets
RtpServerPacketSize	INTEGER		160	RTP Packet Size. Leave at default for Cisco IP phones
RTPServerPassword	STRING	RtpPassword		This value must be defined but the password is only used for internal communication
RTPServerURI	STRING	http://<BH_MT>:<PORT>/voice-servlet/rtp_server/RtpServer.jsp		The URI to access the RTP Server
RTPServerUserName	STRING	RtpClient		This value must be defined, but the user name is only used for internal communication
SecondaryLanguages	STRING ARRAY			This is an array of locales used in multi-lingual deployments. The service first presents the facilities preferred locales, but, if this array has values, then these locales are presented as well giving callers the ability to interact with the service in multiple languages
SharedAudioContentURIs	STRING ARRAY		../shared-audio	The default value is a relative audio path
SpecialAttendantName	STRING			The name of the attendant to be used when the SpecialDates are matched.

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
SpecialDates	STRING			Defines the special dates (similar to Holidays) for the facility auto attendant. Represented as a comma-separated list of dates and time intervals representing special dates. For example, ???112707:0000-2359,123107:1400-2359??? represent all day Thanksgiving Day and after 2:00 PM on New Years Day 2007.
StoreMsgMenuLocale	STRING	en-US		The locale to use when messages are being left for a user
TelephoneAnsweringAddress	STRING ARRAY	vm-no-reply@example.com	TelephoneAnsweringService@example.com	This is the e-mail from: address that is used for voicemail messages when an Oracle Beehive user is not the originator
TransferPrefix	STRING	tel:+ or sip:	tel:+	This is the value attached to the beginning of the voicemail sender's phone number or the User defined OperatorNumber. This is specific to the deployment.

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
TransferSuffix	STRING	@nnn.nnn.nnn.nnn		This is the value attached to the end of the voicemail sender's phone number or the User defined OperatorNumber. This is specific to the deployment.
UserOperatorRule	INTEGER	4	0	Controls whether or not the user's specified operator number can be used. 0 = No/Off 1 = Call backs to Beehive users only 2 = Call backs to phone numbers in Allowed Numbers 3 = Call backs to either Beehive or Allowed Numbers 4 = Always / On (no restrictions)

Configuring the Enterprise

This section describes how to configure enterprise-level preferences for Oracle Beehive voicemail functionality.

> **Note:** The Enterprise is created during Oracle Beehive installation. The only optional steps are to configure the settings for Voicemail and Auto Attendant. The values are stored in enterprise preferences.

This section contains the following topic:

- Configuring Enterprise Preferences

Configuring Enterprise Preferences

Use the following commands to set any of the Enterprise Preference properties listed in Table 11–1, " Voicemail Properties".

1. First, get the identifier for your enterprise by using the `beectl list_enterprises` command:

```
beectl> list_enterprises
```

2. Then, use the enterprise identifier with the `beectl list_preference_properties` command to get the Voicemail Enterprise Preferences:

```
beectl> list_preference_properties prfs=VoiceEnterprisePrefs,enpr=<Enterprise identifier>
```

3. Use the `beectl add_preference_property` command to set or modify preference properties:

```
beectl> add_preference_property --set
prfs=VoiceEnterprisePrefs,enpr=<Enterprise identifier> --name <name> --value
<value> --type <type>
```

See Table 11–1, " Voicemail Properties" for the values of the `--name`, `--value`, and `--type` attributes. Repeat this step for each enterprise property.

Creating Voicemail Users

Generally, you can follow the instructions in "Managing and Provisioning Oracle Beehive Users" in the *Oracle Beehive Administrator's Guide* to create users with access to Oracle Beehive voicemail functions. You must provide values for any voicemail user for the following specific user attributes, in addition to the required user attributes:

- --voice_principal <phone number>
- --voice_pin <PIN>
- --address <type>:tel:<phone number>

The voice_principal and tel: address must be integers containing no special characters or spaces. The voice_principal is used for user authentication because it is associated to the voice_pin. The tel: address attribute is used to associate the redirected DNIS passed by the PBX to the user account or user's e-mail Inbox.

Note: It is possible to define multiple phone numbers that map to the same user account, but only the voice principal defined for that account can be used to authenticate.

Managing Facilities

This section describes how to create and configure a Facility. Facilities allow you to deploy more than one voicemail system (such as, at different physical locations), each with its own properties. Those properties not set at the facility level, will default to their enterprise-level values.

This section contains the following topics:

- Creating a Facility
- Sample Facility XML File

Creating a Facility

To create a Facility, you must create both a UDS user group and a voice facility. The group's values (enterprise identifier, name, properties, and so on) are defined in an XML file. The values of these properties are shown in Table 11–1, " Voicemail Properties". For an example of facility XML file, see Example 11–1, "Sample Facility XML File". It is also possible to use Oracle Beekeeper to define the group. In this case,

you will need to add groups using the XML, but it would still be necessary to look up the group `collabID` entity identifier for that group created in Oracle Beekeeper so the proper group `collabID` could be input into the voice facility command.

Perform the following steps to create a Facility:

1. Create a group by using the `beectl add_group` command:

   ```
   beectl> add_group --file <path to Group XML file>
   ```

   See Example 11–1, "Sample Facility XML File" on page 11-21 for an example Facility XML file.

2. Add a voice facility using the `beectl add_voice_facility` command:

   ```
   beectl> beectl> add_voice_facility --group_collabid <1234:grup:1234:5678>
   [--include <phone rules>] [--exclude <phone rules>]]
   ```

   Note: This command initializes the voicemail component configuration properties which maps the voicemail phone numbers assigned to the UDS group (facility) and creates the `VoiceFacilityPrefs` preference property set. This mapping is contained within a table which is used for phone number look-up, defined with the inclusion and exclusion numbers assigned during the `add_voice_facility` command execution.

   Using the optional --include and --exclude options, you can specify a range of phone numbers to be associated with this voice facility. Use a question mark (?) as a wildcard. Multiple include and exclude ranges can be specified on the command line by delimiting them with a pipe (|) symbol in quotes due to the command shell limitations. For example:

   ```
   beectl> add_voice_facility --name --group_collabid <1234:grup:1234:5678>
   --include "1866612????|1866264????|4730"
   ```

   This example associates all phone numbers in the range "18666120000-18666129999 and 18662640000-99999 and 4730" number ranges with this facility. The phone number lookup is based upon ANI, then RDNIS (redirect number), and then DNIS (dialed number), in that order, to make the association.

   Note: It is also possible to add full SIP URIs to the facility if needed depending upon the router configuration and deployment. If the router has SIP header parsing enabled, then it may be necessary to use a full SIP URI. For example: `--include "sip:1866612????@192.168.1.1"`

3. Add facility preference properties to the voice facility by adding or deleting properties in the VoiceFacilityPrefs preference set assigned to the group which was used during the `add_voice_facility` command. The following commands demonstrate how to list and modify properties in the preference set. The following example uses the default ALL_USERS group, but that can be substitute by the group that was used to create your own custom UDS group facility.

   ```
   beectl> list_preference_profiles --consumer agrp=ALL_USERS --entity_format id
   ```

Look for the VoiceFacilityPrefs Identifier and use that identifier in the next command to add or modify properties:

```
  Name : VoiceFacilityPrefs
  Description :
  Identifier : 721D:6EBE:prfs:77A70C3093BF9C34E040578C36151005000000001DA3

beectl> list_preference_properties --set
721D:6EBE:prfs:77A70C3093BF9C34E040578C36151005000000001DA3

beectl> add_preference_property --set
721D:6EBE:prfs:77A70C3093BF9C34E040578C36151005000000001DA3 --name isMWIEnabled
--value false --type boolean
```

4. To modify or add properties to the `VoiceFacilityPrefs` preference set, repeat the previous steps using the `add_preference_property` command.

You can look up with which facility or facilities a given phone number is assigned and selected based upon weight (strength of number match) by using the `beectl list_voice_facilities` command:

```
beectl> list_voice_facilities --phone <user or voicemail DNIS number>
```

Sample Facility XML File

Example 11–1, "Sample Facility XML File" is a sample XML-formatted file for creating the Group when setting up a Facility. In this example, a static group is used, but it possible to use dynamic groups as well. Be sure to replace the name, description, and scope values with the correct ones for the Facility you are creating.

Note: To find the CollabID of the enterprise or a user, use the global option `--entity_format id` with the appropriate `beectl` list command. For example:

```
beectl list_users --user user=exampleuser --show ALL --entity_
format id
```

The user's CollabID will be shown on the Identifier line of the output.

Example 11–1 Sample Facility XML File

```
<?xml version="1.0" encoding="utf-8"?>
<groups>
  <group type="grup">
    <name>18665552020</name>
    <description>18665552020 Voicemail Facility</description>
    <scope>
<!-- Define Enterpripse CollabID here -->
      <cen>178B:5E25:enpr:360B9A7289F63579E040578C05156389000000018845</cen>
    </scope>
    <members>
      <add>
        <actor>
          <item>
            <!-- User 1 -->
<!-- Define a User's CollabID here -->
            <cen>178B:5E25:user:63386283615A46D59306642C37BF3D07000000000000</cen>
          </item>
        </actor>
```

```
      </add>
     </members>
  </group>
</groups>
```

Example 11–2, "Sample Modifying Facility XML File" shows the XML for modifying a facility. Note that when modifying a group, you do not provide the enterprise identifier, so you should remove the <scope> element tags from a file you previously used to create the group. The facility group's identifier is used. In this example, a second user (User 2) is added to the group.

Example 11–2 Sample Modifying Facility XML File

```
<?xml version="1.0" encoding="utf-8"?>
<groups>
  <group type="grup"
cen="178B:5E25:grup:63386283615A46D59306642C37BF3D0700000000003F">
    <name>18665552020</name>
    <description>18665552020 Voicemail Facility</description>
    <members>
      <remove>
        <actor>
            <!-- User 1 -->
            <cen>178B:5E25:user:63386283615A46D59306642C37BF3D07000000000000</cen>
          </item>
          <item>
            <!-- User 2 -->
            <cen>178B:5E25:user:63386283615A46D59306642C37BF3D07000000000009</cen>
          </item>
        </actor>
      </remove>
     </members>
  </group>
</groups>
```

Configuring the Voicemail Touch-tone User Interface (TUI)

For the Voicemail TUI to function the only required action is to create users with the required attribute values, as described in "Creating Voicemail Users" on page 11-19. Once a user is created you can call into the voice service, and leave and listen to voice messages.

Enabling HTTPS for Cisco VXML Enabled Device Access to Oracle Beehive

To enable the Cisco VXML router to access Oracle Beehive via HTTPS, the Oracle Beehive system must be configured for HTTPS, and the Voicemail Component Properties must be modified to map the RecordPlaybackURIs and RecordStreamURIs properties to the HTTPS URIs.

The Cisco VXML device also needs to import the Oracle Beehive application server's CA certificate, to enable access to Oracle Beehive using HTTPS.

Perform the following steps:

1. Configure HTTPS for this Oracle Beehive instance

   For instructions, see "Changing HTTP Port" in Chapter 12, "Oracle Beehive Post-Installation Procedures" in the *Oracle Beehive Installation Guide* for your platform (Linux or Solaris only).

2. Modify the voicemail facility object's properties, using the following commands, replacing the items in angle-brackets (<>) with the appropriate values:

```
beectl> modify_property --component <voice enterprise alias or object ID>
--name RecordPlaybackURIs --value
https://<host>:<port>/voice-servlet/vmail/data/shared/playback

beectl> modify_property --component <voice enterprise alias or object ID>
--name RecordStreamURIs --value https://<host>:<port>/voice-servlet/vmail/crs

beectl> activate_configuration
```

To secure HTTPS between Cisco VXML-enabled Routers and Oracle Beehive, you need to import the Oracle Beehive certificate into the IOS device during device configuration. Configure your Cisco VXML router for HTTPS application access using the following steps:

1. From Internet Explorer, access the Oracle Beehive Application Server with [https://<ServerIP>:<port>/

   Use the server and HTTP port for the computer hosting the Oracle Beehive tier.]

   The **Security Alert** dialog box displays

2. Click **View Certificate**

   The **Certificate** dialog box displays

3. Select the **Details** tab

   <All> will be highlighted in the **Show** drop-down list

4. Click **Copy to File**

   The **Certificate Export Wizard** dialog appears

5. Click **Base-64 encoded X.509 (.CER)** and then click **Next**

6. Specify a file name in the **File to Export** dialog box and then click **Next**

7. Click **Finish**

   An **Export was Successful** message displays.

8. Click **OK** and close the **Security Alert** dialog box.

9. Open the exported file in a text editor and copy the text that appears between the `---BEGINCERTIFICATE--` and `--END CERTIFICATE--` tags.

   You are now ready to copy the Oracle Beehive Application Server certificate information to the IOS device

10. Access the IOS device in privileged EXEC mode

> **Note:** For more information about managing the IOS device, refer to the Cisco IOS Command-Line Interface documentation

11. Access global configuration mode by entering the configuration terminal

12. Create and enroll a `trustpoint` by entering the following commands:

```
crypto pki trustpoint xxxx
en terminal
revocation-check none
exit
```

Where xxxx is a `trustpoint` name

The IOS device exits configuration terminal mode and returns to privileged EXEC mode

13. To copy the certificate exported to the text file to the IOS device, perform the following steps:
    a. Enter:

    ```
    crypto pki auth xxxx
    ```

    Where xxxx is the `trustpoint` name specified in step 12
    b. Paste in the certificate you copied from the text file in Step 9
    c. Enter:

    ```
    quit
    ```

    A message displays describing the certificate attributes, and a confirmation prompt appears
14. Enter:

    ```
    Yes
    ```

    A message reports that the certificate was successfully imported
15. Associate the imported certificate with the http client by entering the following command:

    ```
    Enter http client secure-trustpoint xxxx
    ```

    Where xxxx is the `trustpoint` name specified in the previous steps

You have finished importing the certificate.

Configuring Cisco IP Phone Voicemail GUI Application

The IP Phone GUI is a Cisco phoneXML application that is served from Oracle Beehive to the Cisco IP Phones, for use on the Cisco IP Phone's graphical display. This application is supported by Cisco Hard Phone 7970 Series as well as Cisco IP communicator, which is a software phone. In order for the voicemail GUI advanced features (play audio and return call) to function properly, the internal Cisco IP phone's Web server must be enabled. These Web servers are enabled by default, but some deployments disable them. Also, the Cisco IP Phones do not support HTTPS, so in order for the IP phones to access the XML application, the Oracle Beehive application server must allow HTTP access from the phone to the URI `http://<beehive server>:<port>/voice-servlet/cisco-ip-phones`.

> **Note:** Cisco IP Communicator version 7.0.1 is supported. Other versions may not be fully compatible.

The configuration for the voicemail GUI is determined by the network topology and how Cisco Unified Call Manager is deployed.

The following are the points of communication:

- HTTP and RTP from the Oracle Beehive application tier to the IP Phones

- HTTPS from the Oracle Beehive application tier to the Cisco Call Manager AXL interface
- Standard client traffic flow of HTTP from the Cisco IP phones to the Oracle Beehive application tier

The communication needed from the Oracle Beehive application tier to the IP Phones is necessary because Oracle Beehive will push requests to the IP Phones via HTTP utilizing the internal Web server running on the Cisco IP phones. Also, to play back voicemail audio files, RTP communication needs to be enabled from the Oracle Beehive application tier to the IP phones. The Oracle Beehive application tier needs to communicate to the Cisco Call Manager AXL interface to look up the IP Phone's registered IP address, to push HTTP commands and RTP streams for audio playback.

To configure the voicemail GUI the properties shown in Table 11–2, " Cisco IP Phone Recommended Deployment Properties" need to be set, depending on your network topology and Cisco Call Manager Configuration. These properties can be set in the Voice Enterprise preference set or in the facility group file. The values in **bold** are the recommended values to define.

> **Note:** To make the voicemail GUI available from the Cisco IP Phones, you must also set certain properties, as described in "Configuring the Voicemail GUI and Message Waiting Indicator" on page 11-28.

Table 11–2 Cisco IP Phone Recommended Deployment Properties

Preference Name	Type	Example Value	Default Value	Description
IpPhoneUserName	STRING	bhvmgui	none	Defines the account name provisioned in Cisco Call Manager that has device control over the user's device
IpPhonePassword	STRING	password	none	Defines the account password provisioned in Cisco Call Manager that has device control over the user's device

Table 11–2 (Cont.) Cisco IP Phone Recommended Deployment Properties

Preference Name	Type	Example Value	Default Value	Description
AxlConfigA	STRING	https://<CCM HOST>:8443/axl/,https://<CCM HOST>:8443/realtimeservice/services/RisPort,,,bhvmgui,password,f,f	none	The comma separated fields, in order, are: 1. The AXL URL 2. The realtime info URL 3. The proxy host 4. The proxy port 5. The user provisioned for AXL access (read only) 6. The password for the AXL user 7. Whether certificate validation is enabled for SSL connections to CCM 8. Whether hostname verification is enabled for SSL connections to CCM
RTPServerURI	STRING	http://bigip-beehive.example.com/voice-servlet/rtp_server/RtpServer.jsp	http://<beehive_middle_tier>:<PORT>/voice-servlet/rtp_server/RtpServer.jsp	The URI to access the RTP Server
RTPServerUserName	STRING	RtpClient	none	This value must be defined but the username and password is only used for internal communication
RTPServerPassword	STRING	RtpPassword	none	This value must be defined but the username and password is only used for internal communication

Table 11–2 (Cont.) Cisco IP Phone Recommended Deployment Properties

Preference Name	Type	Example Value	Default Value	Description
MwiCcmHost	String	callmanger.example.com	none	The Cisco Call Manager IP address where the MWI SIP trunk is defined
MwiCcmPort	INTEGER	5060	none	The Cisco Call Manager port of the MWI SIP trunk
IpPhoneRtpMinPort	INTEGER	20480	none	Minimum value for allowed RTP port range to IP phones
IpPhoneRtpMaxPort	INTEGER	32768	none	Maximum value for allowed RTP port range to IP phones
IpPhoneHttpProxyHost	STRING	internal-proxy.example.com	none	Defines the proxy needed for the Oracle Beehive tier to access the internal IP Phones
IpPhoneHttpProxyPort	INTEGER	80	none	Defines the proxy port needed for the Oracle Beehive tier to access the internal IP Phones
MwiAlgorithm	STRING	SHA1PRNG	none	Defines the algorithm used to created the SIP NOTIFY messages for MWI
MwiSipProxyHost	STRING	sip-proxy.example.com	none	SIP Proxy Host that is authorized to send SIP messages to Cisco Call Manager SIP MWI trunk
MwiSipProxyPort	INTEGER	5060	none	SIP Proxy Port that is authorized to send SIP messages to Cisco Call Manager SIP MWI trunk

Table 11–2 (Cont.) Cisco IP Phone Recommended Deployment Properties

Preference Name	Type	Example Value	Default Value	Description
MwiMinLocalPort	INTEGER	5060	none	Lowest port in the range that Oracle Beehive will use to send and receive SIP messages
MwiMaxLocalPort	INTEGER	5080	none	Highest port in the range that Oracle Beehive will use to send and receive SIP messages
MwiSourcePhone	STRING	18665551234	none	Voicemail DNIS phone number that is presented in the SIP NOTIFY message
RtpServerHost	STRING	bigip-beehive.example.com	none	Host where the RTP Server is running
RtpServerMinPort	INTEGER	20480	20480	Minimum port range used by the RTP server to send RTP packets
RtpServerMaxPort	INTEGER	32768	32768	Maximum port range used by the RTP server to send RTP packets
RtpServerPacketSize	INTEGER	160	160	RTP Packet Size. Leave at default for Cisco IP phones

Configuring the Voicemail GUI and Message Waiting Indicator

Telephones within your deployment may have a Message Waiting Indicator (MWI), which lights up when the phone number has received one or more voicemail messages. Additionally, sophisticated phones may have a Graphical User Interface (GUI), which presents a menu of choices to the user.

In order to enable the use of voicemail features through an IP phone's GUI, or to enable MWI operation, you must set the properties identified in Table 11–3, " Voicemail Properties". These are a subset of the total set of properties, which are listed in Table 11–1, " Voicemail Properties" on page 11-6. You must set these properties at the Enterprise level of scope, but you can also set them at a Facility level; at a given Facility, the Facility-level properties override the global Enterprise-level properties.

The properties listed are for configuring using local IP Phone authentication.

To use local IP Phone configuration every device in Cisco Call Manager needs to be configured with the authentication server URL: `http://<Beehive_HOST>:<port>/voice-servlet/cisco-ip-phones/authenticate.jsp`

"Configuring Enterprise Preferences" on page 11-18 describes how to set the voicemail properties.

Table 11–3 Voicemail Properties

Property	Value	Type
IpPhonePasswordSeed		STRING
IpPhonePasswordAlgorithm	AES	STRING
IpPhonePasswordNumBits	128	INTEGER
IpPhoneRtpMinPort	20480	INTEGER
IpPhoneRtpMaxPort	32768	INTEGER
RtpServerHost	<hostname> (of the computer on which Beehive is installed	STRING
RTPServerUserName	RTPClient	STRING
RTPServerPassword	RTPpwd	STRING
RTPServerURI		STRING
MwiAlgorithm	SHA1PRNG	STRING
MwiCcmHost		STRING
MwiCcmPort	5060	INTEGER
MwiSipProxyHost		STRING
MwiSipProxyPort	15060 (optional property)	INTEGER
MwiMinLocalPort	5060	INTEGER
MwiMaxLocalPort	5080	INTEGER
MwiSourcePhone		STRING

Cisco Router Configuration

This section assumes you have Cisco IOS and Cisco Call Manager Administration configuration experience. Cisco 2800/3800 Series or AS5400XM with IOS version 12.4(11T) or greater with VXML feature set, is required for Oracle Beehive voicemail.

To configure your Cisco router hardware for use with Oracle Beehive, perform the following steps:

1. Configure Translation Rule
2. Configure Global VXML Configuration Options
3. Configure Voicemail VXML Application
4. Configure Transcoding (Optional)

Each step is described in its own section.

Configure Translation Rule

The router must be configured for full E.164 phone numbers, which map to the phone numbers defined in users' UDS record `voice_principal` and `tel:` address attributes.

These rules will change, depending on the incoming DNIS delivery method provided by your PRI.

The following example shows how 5-digit DNIS delivery is expanded:

```
voice translation-rule 10
 rule 2 /\(^627..$\)/ /170332\1/
!
!
voice translation-profile FULL_E164_IN
 translate called 10
!
voice-port 0/3/0:23
 translation-profile incoming FULL_E164_IN
```

Configure Global VXML Configuration Options

Make the following configurations:

- http client cache memory pool 8192

  Note: If you record prompts with file sizes larger than the http client cache, the prompt will not be cached. Latency between the gateway and the Oracle Beehive instance could cause a timeout while Oracle Beehive waits for the gateway to send audio files. The end-user may experience long pauses while using the TUI, and a timeout could abruptly end the call without any error messages.

  If you experience this issue, increase the http client cache memory pool size to a size larger than your largest prompt file.

- http client cache memory file 200
- http client cache refresh 300
- http client response timeout 30
- ivr prompt memory 4096
- ivr prompt streamed http

  Note: If you are using `https`, then ivr prompt streamed must be set to none.

- ivr record memory system 48000
- ivr record memory session 1500
- vxml tree memory 100000
- vxml version 2.0

The following settings are necessary only if fax is to be supported on the PSTN side of the ingress gateway:

- fax receive called-subscriber d
- fax interface-type fax-mail
- mta send server bh-midtier port 5025

Note: This is the SMTP service running on your Oracle Beehive server.

- mta send with-subject both
- mta send mail-from hostname example.com
- mta send mail-from username s

Configure Voicemail VXML Application

Use the following configuration settings:

```
application
  service vm_bh http://beehive.example.com:7777/voice-servlet/vmail/start.vxml
!
dial-peer voice 500 voip
  description Voicemail Pilot Number
  huntstop
  service vm_bh
  session protocol sipv2
  incoming called-number 18665551234
  dtmf-relay rtp-nte sip-notify
  codec g711ulaw
  no vad
```

Use the following settings to configure Fax functionality (if you will be setting up fax service with Oracle Beehive).

```
!
dial-peer voice 310 mmoip
 service fax_on_vfc_onramp_app out-bound
 destination-pattern 1866.......
 information-type fax
 session target mailto:faxservice@example.com
 image encoding MH
```

Note: To complete the configuration of fax functionality with Oracle Beehive, follow the steps in "Configuring Oracle Beehive Fax" on page 11-35.

Configure Transcoding (Optional)

Cisco only supports uncompressed audio on IVR application call legs. Depending on the Call Manager deployment configuration using compressed audio, it is possible to configure transcoding on the local router in order to support multiple codecs.

Use the following configuration settings:

```
sccp local Loopback0
sccp ccm 192.188.175.105 identifier 1 priority 1 version 5.0.1
sccp
!
sccp ccm group 1
description Reston Lab transcoding for IPC
bind interface Loopback0
associate ccm 1 priority 1
associate profile 1 register restontvg2
```

```
 !
dspfarm profile 1 transcode
codec g711ulaw
codec g711alaw
codec g729ar8
codec g729abr8
codec gsmfr
codec g729r8
codec pass-through
maximum sessions 12
associate application SCCP
```

Cisco Unified Call Manager Configuration

To configure the Touch-tone User Interface (TUI), you must do the following:

1. Create SIP Trunk
2. Create Voicemail Pilot Number
3. Create Voicemail Profile
4. Assign Voicemail Profile to users' Directory Phone Number

To configure Voicemail GUI, you must do the following:

1. Create CCM User that is associated to all users' phone devices
2. Create a Read Only user with AXL Access
3. Define IP Phone Services in CCM

Note: You can also configure the language availability for Oracle Beehive voicemail users from Cisco Call Manager user interface. The default location for the CCM User interface is `https://<CCM_HOST>:8443/ccmuser/`.

For locale support on Cisco Call Manager, the appropriate locale packs must be installed. Once the locales are installed in Cisco Call Manager, Oracle Beehive voicemail users can select the locale using the Cisco Call Manager User interface.

Configuring the Auto Attendant

This section contains the following topics:

Example AAML-based Auto Attendant File

The following AAML file is an example with the commands used to upload the AAML file and prompts using `beectl`. For more information, refer to Appendix B, "Oracle Auto-Attendant Markup Language (AAML) Reference."

```
<!-- The attendant name is used in the Voice Facility Property to associate the AA
to the facility -->
<attendant name="GIT">
  <languages>
    <audio>
       Thank you for calling Oracle Corparation. To proceed in English press
       one, to proceed in Spanish press two.
    </audio>
    <language>en_US</language>
    <language>es</language>
  </languages>
  <menu>
    <audio-set>
      <audio lang="en_US">
        To reach an employee at this office press one.
        To leave a general message for the operator press two.
        Alternatively you can try to reach a live operator if available by
        pressing three.
        To hear these options again press nine.
      </audio>
      <audio lang="es">
      </audio>
    </audio-set>
    <option ord="1">
<!-- The directory facility is the UDS group name to associate the users that will
be looked up in the local directory -->
      <directory facility="OVL_DYN"/>
    </option>
    <option ord="2">
<!-- The voicemail mailbox is used to send the caller directly to this voicemail
users voicemail box -->
      <voicemail mailbox="18665552005"/>
    </option>
    <option ord="3">
<!-- The transfer destination is what is used in the VXML transfer tag executed on
the Cisco VXML router -->
      <transfer destination="sip:18665552005@192.168.1.10"/>
    </option>
    <option ord="9">
      <reprompt/>
    </option>
  </menu>
</attendant>

beectl> add_attendant --file attendant.aaml
beectl> upload_attendant_prompt --name GIT --type language --file
LanguagePrompt.wav
beectl> upload_attendant_prompt --name GIT --type menu --language en_US --file
EnglishMenu.wav
beectl> upload_attendant_prompt --name GIT --type menu --language es --file
SpanishMenu.wav
```

Installing an Auto Attendant

To install an Auto Attendant, create an Auto Attendant Markup Language (AAML) document for your attendant, and record your voice prompts, making sure that they are 8000Hz mono u-law WAVE files. Make the files accessible from the computer on which Oracle Beehive is installed. For more information on creating AAML files, refer to Appendix B, "Oracle Auto-Attendant Markup Language (AAML) Reference."

Use the `beectl add_attendant` command to add the auto attendant:

```
beectl> add_attendant --file <AAML file>
```

Next, use the `beectl upload_attendant_prompt` command to upload your recorded audio prompts:

```
beectl> upload_attendant_prompt --name <attendant name> --language <language>
--type <type> --file <prompt file>
```

You can review the VoiceXML at `http://<host>:<port>/voice-servlet/aa/view/<attendant name>.do`.

Associating an Auto Attendant with a Facility

To associate an Auto Attendant to a facility, first create one or more facilities by following the directions in "Managing Facilities" on page 11-19. When you create a Facility, you create both a UDS group (static or dynamic) and a voice facility. The group's values, such as identifier, Name, Members, and so on, are defined in an XML file. See Example 11–1, "Sample Facility XML File" for an example Facility XML file.

See "Managing Facilities" on page 11-19 for instructions on creating a Facility.

After the voicemail facility has been created, it is just a matter of adding an additional preference property to the `VoiceFacilityPrefs` preference set as demonstrated in "Creating a Facility" as well as mapping the Auto Attendant DNIS in that facility lookup and configuring the Auto Attendant application in the Cisco router.

```
beectl> add_preference_property --set
721D:6EBE:prfs:77A70C3093BF9C34E040578C36151005000000001DA3 --name
DefaultAttendantName --value <Attendant Name in AAML> --type string
```

Use the `beectl activate_configuration` command to validate and activate the configuration changes:

```
beectl> activate_configuration
```

For more information on administration commands for the auto attendant, see "Auto Attendant Administration Commands" on page 11-35.

Configure an Auto Attendant VXML Application in Cisco IOS

To configure an Auto Attendant VXML application in Cisco IOS, use the following configuration settings:

```
application
  service aa_bh http://beehive.example.com:7777/voice-servlet/aa/start.vxml
!
dial-peer voice 500 voip
  description AA Number
  huntstop
  service aa_bh
  session protocol sipv2
  incoming called-number> 18664441234
  dtmf-relay rtp-nte sip-notify
  codec g711ulaw
  no vad
```

For more information on administration commands for the auto attendant, see "Auto Attendant Administration Commands".

Auto Attendant Administration Commands

This section lists the various `beectl` commands used for managing the Auto Attendant, along with descriptions.

- **add_attendant**

  Adds an auto attendant using the information from an AAML document. The name of the attendant will be taken from the name attribute of the root element, attendant. It should not contain any white space.

- **delete_attendant**

  Deletes an auto attendant by name.

- **modify_attendant**

  Updates an existing auto attendant with a new AAML file. You can also use this command to rename an existing attendant.

- **upload_attendant_prompt**

  Uploads a prompt for an attendant.

- **delete_attendant_prompt**

  Deletes a prompt.

- **list_attendant_prompts**

  List all prompts that have been uploaded for an attendant.

- **list_attendant_aaml**

  Prints the Auto Attendant Markup Language document for an auto attendant.

- **modify_ip_phone_password_seed**

  Resets the seed of the password generator for IP phones. To send commands to an IP phone the requestor must be authenticated. The password is generated based on information in the phone but must be seeded to ensure security.

Configuring Oracle Beehive Fax

Oracle Beehive fax functionality is enabled using the same Cisco infrastructure as voicemail. Once you have configured Cisco Call Manager for voicemail, you must perform additional configuration to enable fax.

In Oracle Beehive, the Fax Message Service provides configuration options. You must also create a special Fax User, and set up a business event notification for that user. The Cisco Call Manager will send all faxes to that user, and then the notification will trigger a process that forwards the fax to the intended Oracle Beehive user.

To configure Cisco Call Manager for fax, see "Configure Voicemail VXML Application" on page 11-31

Perform the following procedure to create the special Oracle Beehive fax account and set up the notification:

1. Add a user using the `beectl add_user` command. You can use any name for the account; in this example, `FaxUser` is used. Give the user an e-mail address, such as faxuser@<yourcompany.com>:

   ```
   beectl> add_user --family_name FaxUser --scope <your enterprise identifier>
   --login_id faxuser --login_password <password> --address BUSINESS_
   1:mailto:faxuser@example.com
   ```

2. After the fax service user is provisioned, get the Entity ID (EID) of that user by using the `beectl list_users` command with the `--entity_format` option:

   ```
   beectl> list_users --user user=faxuser --show ALL --entity_format id
   ```

   The EID is the portion of the user's CollabID following the `:user:` segment. For example, if the `list_users` command produced the following output:

   ```
   User Identifier: 05C1:7403:user:9AE5E38909BE41C181BAD42CE1B88F5300000000000E
   ```

   Then the EID is `9AE5E38909BE41C181BAD42CE1B88F5300000000000E`

3. Use the XML file shown in Example 11–3 to create a subscription, by using the `beectl add_event_subscription` command:

   ```
   beectl> add_event_subscription --file <name of XML file>
   ```

   When you have completed this step, Oracle Beehive is ready to receive fax messages from the Cisco Call Manager.

Example 11–3 Sample Fax User Event Subscription

In this example, replace the bolded EID with the EID of your Oracle Beehive fax user.

```
<?xml version="1.0" encoding="UTF-8" ?>
<eventSubscription xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:noNamespaceSchemaLocation="eventSubscription.xsd">
<subscriberId></subscriberId>
<name>ES_MSG_DELIVERED_EVENT_FAX_LISTENER</name>
<description>Subscription to sync Fax repository for incoming fax </description>
<eventName>ES_MSG_DELIVERED</eventName>
<Condition>
  <simple>
    <leftSide>RAWTOHEX(custom_attributes.recipient_eid)</leftSide>
    <operator>=</operator>
    <rightSide>'E603E73114BB4944AF5A6E5014D520E10000000003C1'</rightSide>
  </simple>
  <!--
  <conjunction>
  </conjunction>-->
  <!--
  <disjunction>
  </disjunction>-->
</Condition>
<Action>
  <isPLSQLAction>F</isPLSQLAction>
  <actionString>oracle.ocs.management.model.FaxMessageService:ES_MSG_
DELIVERED</actionString>
  <ActionPreferenceInfos>
    <actionPreferenceInfo>
      <key></key>
      <value></value>
    </actionPreferenceInfo>
  </ActionPreferenceInfos>
</Action>
</eventSubscription>
```

A

Example of a Classification Plug-in Collection File

This appendix contains an example of a classification plug-in collection file for Oracle Information Rights Management (Oracle IRM). This example can be used as a basis for the file that is required to integrate Oracle IRM with Oracle Beehive, as described in Chapter 5, "Integrating Oracle Information Rights Management (Oracle IRM) with Oracle Beehive." Values that must be changed to match the settings of your deployment appear in **bold**.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Copyright (c) 2007, 2009, Oracle and/or its affiliates.
All rights reserved. -->
<core:ClassificationPluginCollection xmlns:core="http://xmlns.oracle.com/irm/core"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:system="http://xmlns.oracle.com/irm/system"
 xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <classificationPlugin>
    <!-- Define the classification system that this plugin is for -->
    <system>
        <uuid>2de4af00-3b14-11de-8a39-0800200c9a66</uuid>
    </system>
    <!-- Define a human readable label for the classification system
         for diagnostic purposes -->
    <label>
            <locale>en</locale>
            <name>Beehive</name>
            <description>Beehive classification system</description>
    </label>
    <!-- Should this classification system be enabled? -->
    <enabled>true</enabled>
    <!-- The definition describes how the classification system is implemented -->
      <definition>
        <interfaceClass>
           oracle.irm.engine.core.classification.ClassificationServices
        </interfaceClass>
        <resourcePath>
/oracle/irm/engine/core/classification/aspects/classes/ClassificationServices.xml
        </resourcePath>
        <!-- The base class section pulls in standard classification system
             functionality -->
        <baseClass>
          <interfaceClass>
             oracle.irm.engine.core.classification.ClassificationServices
          </interfaceClass>
```

```
          <resourcePath>
/oracle/irm/engine/core/classification/aspects/classes/ClassificationServices.xml
          </resourcePath>
        </baseClass>
            <!-- Here we pull in all classification system functionality required
                 by this system -->
            <slices>
                <!-- The ClassificationServicesResponse aspect allows us to
                     respond to calls on ClassificationServices with a hard coded
                     response -->
                <implementationClass>
oracle.irm.engine.core.classification.aspects.response.ClassificationServicesRespo
nse
                </implementationClass>
                <!-- The methods state which methods we wish to hard code the
                     response to -->
                <methods>
                   <!-- Construct is required for initialization and should always
                        be included -->
                   <methodClass>
oracle.irm.engine.core.classification.ClassificationServicesSlice
                   </methodClass>
                   <name>construct</name>
                </methods>
                <methods>
                  <methodClass>
oracle.irm.engine.core.classification.ClassificationServicesSlice
                  </methodClass>
                  <name>listContentAttributes</name>
                </methods>
                <!-- InjectionData provides the configuration information
                     containing the hard coded response data -->
                <injectionData>
                    <field>
                       <fieldClass>
oracle.irm.engine.core.classification.aspects.response.ClassificationServicesRespo
nse
                        </fieldClass>
                        <name>responses</name>
                    </field>
            <data xsi:type="system:MethodResponseCollection"
             xmlns:system="http://xmlns.oracle.com/irm/system">
            <!-- There should be a methodResponse for each hard coded response -->
            <methodResponse>
                <method>
                    <methodClass>
oracle.irm.engine.core.classification.ClassificationServices
                    </methodClass>
                    <name>listContentAttributes</name>
                </method>
              <!-- An empty response is equivalent to null -->
                <response xsi:type="core:ClassificationServices_
ListContentAttributesResponse"/>
           </methodResponse>
         </data>
       </injectionData>
     </slices>
     <!-- The ClassificationServicesProxy aspect allows us to respond
        to calls on ClassificationServices by calling out to web service -->
     <slices>
```

```
        <implementationClass>
oracle.irm.j2ee.jws.core.classification.ClassificationServicesProxy
        </implementationClass>
        <!-- By not specifying which methods we are implementing here we are
             saying all methods. Anything not implemented by the response aspect
             will go over the web service. -->
        <!-- InjectionData provides the configuration information
             containing the web service details -->
        <injectionData>
            <field>
               <fieldClass>
oracle.irm.j2ee.jws.core.classification.ClassificationServicesProxy
               </fieldClass>
               <name>webServiceProxyContext</name>
             </field>
             <system:WebServiceProxyContext>
                 <!-- Web service endpoint URI -->
                 <endpointAddress>
                     http://beehive.yourcompany.com/bhirm/ClassificationServices
                 </endpointAddress>
                 <!-- Authentication to use with the endpoint -->
                 <authenticationType>NONE</authenticationType>
                 <!-- How long (in seconds) should we wait to connect/for a
                      response. -1 means infinite wait -->
                 <timeout>-1</timeout>
             </system:WebServiceProxyContext>
           </injectionData>
      </slices>
    </definition>
  </classificationPlugin>
</core:ClassificationPluginCollection>
```

B

Oracle Auto-Attendant Markup Language (AAML) Reference

The appendix provides information on Oracle Auto-Attendant Markup Language (AAML) elements, and contains the following sections:

- Overview of Automated Attendants
- Overview of AAML Documents
- Element Definitions
- Example AAML Document

Overview of Automated Attendants

An automated attendant (auto-attendant) is a telephony application that acts as a virtual receptionist for an office. The application answers the telephone and enables callers to search for and contact individuals at a particular facility. The application also enable callers to obtain general information about a specific facility or enterprise.

The options and behaviors of auto-attendants are defined in AAML (`.aaml`) documents. For more information, see "Overview of AAML Documents".

Overview of AAML Documents

An AAML document defines the options and behaviors of an auto-attendant, and consists of the following sections:

- Languages Section
- Greeting Section
- Extension-Dialing Section
- Menu Section
- Event Handler Section

These sections are represented by the following elements: languages, greeting, extension-dialing, menu, and handlers, respectively. Also, the execution of an auto-attendant starts with the Languages Section, followed by the Greeting Section, and then the optional Extension-Dialing Section, Menu Section, and Event Handler Section.

Languages Section

The Languages section declares the languages the caller can select in the auto-attendant. It contains an optional prompt and a list of languages. When a caller hears the language prompt, the caller can use the keypad to select the language to use for the rest of the call. The prompt is specified by an audio element. The list of languages is specified by a sequence of language elements.

The order of the languages in the document determines the corresponding digit to select, starting with the one key (1). The language section is required even if only one language will be used. However, in that case, a prompt is not needed (nor will one be played even if specified). This is the only time an audio element is used without an lang attribute, since the message will be played before the desired language is known. In the single language case, the execution will jump straight to the greeting section.

For example, the following language section defines a prompt that declares English and Spanish as choices for the caller:

Example B–1 Language Section with Two Languages

```
<languages>
  <audio>For English press one, para Espaniol...</audio>
  <language>en-us</language>
  <language>es-us</language>
</languages>
```

However, the following Language section only specifies English, so no prompt will be displayed:

Example B–2 Language Section with One Language

```
<languages>
   <language>en-us</language>
</languages>
```

Greeting Section

As the name implies, the Greeting section defines the attendant's greeting. This will be played after the language has been selected (if applicable) and before the main menu. This is the only time the greeting will be played.

The Greeting section allows for prompts in multiple languages which is specified by an audio-set element. The audio set should contains one audio element per language for each language declared in the languages section. The audio-set element is required even in the single-language case.

Once the greeting has played execution moves on to the menu section. If the caller presses any keys while the greeting plays, the attendant will behave as if the key press was made during the menu prompt, allowing users familiar with an attendant to skip the greeting. This feature can be disabled by setting the bargein attribute to `false` in the greeting tag, thus forcing callers to hear the entire greeting.

Example B–3 Greeting Section

```
<greeting>
  <audio-set>
     <audio lang="en-us">English greeting goes here</audio>
     <audio lang="es-us">Spanish greeting goes here</audio>
   </audio-set>
</greeting>
```

Extension-Dialing Section

The optional Extension-Dialing section can be used to enable callers to call a person directly by dialing the person's extension. This is achieved through an extension-dialing element declaration.

Menu Section

The heart of the application is the Menu section, which contains a menu element. The menu element consists of a prompt and up to nine options. An option can have a prompt defined that will be played upon selection. After an option's prompt is played, an action will be taken.

The system supports the following actions, each of which has a corresponding element with the same name:

- **directory:** The directory action invokes the directory module allowing a the caller to lookup people. The first prompt of the directory module is `To search for a...`. If the caller elects to cancel the directory lookup, then the caller will hear the menu again. If the caller finds someone and elects to call the person or leave a voicemail, then the caller will not return to the attendant.
- **disconnect:** The disconnect action terminates the call after the prompt is played.
- **reprompt:** The reprompt action plays the menu prompt and allows the caller to choose a different option.
- **submenu:** The submenu action transfers control of the call to another attendant; callers will be able to return to this attendant by pressing the start key (*). The return option is implicitly added in the sub-menu. In other words, it is not specified in the AAML of the sub-menu. If the currently selected language is also declared in the sub-attendant, then the execution begins with the greeting prompt, otherwise execution starts with the language selection. If the sub-attendant only has one language, then it is used regardless of the current language.
- **transfer:** The transfer action redirects the call to a given phone number. There is no way to return once the call has transferred.

Event Handler Section

The event handlers determine what message should be played if the caller presses an invalid key or does not press anything within a timeout period (this is system dependent). This is known as throwing a nomatch or a noinput event, respectively. There are two sub-types of event handlers for each main type (nomatch and noinput), `warning` and `disconnect`. The `disconnect` handler is invoked when a maximum number of events of one type is reached; warning in every other case. The message played and the maximum are specified separately for each event type.

Element Definitions

This section contains the definitions and details for the following elements:

attendant

The root element of an AAML document.

Parents

greeting, languages, menu, noinput, nomatch

Children

none

Attributes

name

The name of the auto-attendant. It must start with an letter and can contain only letters, numbers, and underscores. It should be the same name of the file (without the `.aaml` suffix).

audio

Represents a prompt in a given language. There can be only one prompt per language in a given audio-set.

Parent
audio-set, languages

Children
none

Attribute

lang
The language code of the recorded file.

uri
The URI of a recording of the contained text. It is not necessary if you upload prompts with `beectl`.

Remarks
The contained character data is a text of the message, while the URI points to a recording of the text. The lang attribute is not necessary when tag used as part of a languages element.

audio-set

A set of messages, one for each declared language. There should one audio tag in a set for each language declared in the languages element.

Parents
greeting, menu, option, noinput, nomatch

Children
audio

Attributes
none

Remarks
Represents a message in all languages declared for this auto-attendant. This element must containing one audio element for each language declared for this document.

directory

Jumps to the directory module. The first prompt of the directory module is `Spell the person's first name, then press star....` Any prompt played in an option containing this element should be made to flow well with this prompt.

Parents
option

Children
none

Attributes

facility

The name of the facility to be used as the search context when looking up users in the directory.

Remarks

Indicates that the parent option should go to the directory module.

disconnect

Terminates the phone call.

Parents

option

Children

none

Attributes

none

Remarks

Indicates that the parent option should end the phone call.

extension-dialing

Declares the callers should be allowed to call a person directly by dialing their extension. The length of extensions and the prefix to prepend to them are defined in the facility properties. If this element is present the option to dial by extension will be presented after the greeting prompt. The caller can continue on the main menu instead by pressing the pound key (#). Pre-recorded prompts will be provided by the system. An administrator may choose to re-record them to maintain consistency with other prompts. If so, then see the call flow to ensure that the wording is consistent with the behavior of the application. There is no need to record the no-input prompts as they are the same as the ones used in the rest of the application.

Parents

attendant

Children

none

Attributes

none

greeting

Defines the greeting message heard after selecting the language and before the main menu.

Parents
attendant

Children
audio-set

Attributes

bargein
If `true`, allows the caller to enter a menu option while listening to the greeting. If `false`, all input will be ignored and the greeting will continue to play. Default is `true`.

goto

A menu option that will trigger the IVR gateway to fetch a VoiceXML document and begin executing it. The document can be any VoiceXML document that can be understood by the gateway.

Parents
option

Children
none

Attributes

url
The URL of a VoiceXML document.

handlers

Contains the noinput and nomatch handlers for the attendant. This element can contain at most two noinputs and nomatches, one of each type (warning or disconnect).

Parents
attendant

Children
noinput, nomatch

Attributes
none

language

Declares a language for the auto-attendant.

Parents
languages

Children
none

Attributes
none

Remarks
Must contain an ISO language code of the desired language. An audio element with this code in the lang attribute must then be given in all audio-sets in the rest of the document.

languages

Declares the set of languages used in the auto-attendant.

Parents
attendant

Children
audio, language

Attributes
none

Remarks
The order of the child language elements determines the DTMF key that selects the given language. For example, given the following:

```
<languages>
  <audio uri="...">For English press one, pour Francais tirez deux.</audio>
  <language>en</language>
  <language>fr</language>
</languages>
```

Pressing 2 selects French as the language for the duration of the call (as the prompt suggests).

menu

Defines the main menu of the auto-attendant.

Parents
attendant

Children
audio-set, option

Attributes

none

Remarks

Contains the main menu prompt instructing the caller on the available options and the definition of those options.

noinput

Defines what to tell the caller if they do not provide a response before the timeout period (system dependent). If a disconnect handler is not given, the application will disconnect after three noinputs. If the type is `warning`, then the menu will be replayed after the element's prompt. If the type is `terminal`, then the call is terminated. An explicit disconnect or reprompt element is not necessary.

Parents

attendant

Children

audio-set

Attributes

type

The type of handler. Can be either warning or terminal. Default is warning.

count

The number of no inputs before disconnecting. Valid only if type is terminal. Default is `3`.

nomatch

Defines what to tell the caller if he or she presses a key that is not defined as a valid input. If a disconnect handler is not given, then the application disconnects after three nomatches. If the type is warning, then the menu will be replayed after this element's prompt. If the type is terminal, then the call will be terminated. An explicit disconnect or reprompt element is not necessary.

Parents

attendant

Children

audio-set

Attributes

type

The type of handler. Can be either warning or terminal. Default is warning.

count

The number of nomatches before disconnecting. Valid only if type is terminal. Default is 3.

option

Defines a menu option.

Parents

menu

Children

audio-set, directory, disconnect, reprompt, submenu, transfer

Attributes

ord

The DTMF key to select the option. Can be a single digit from 1 to 9.

Remarks

Since the ord attribute defines the key to select the option, it must be distinct from any other ords in the menu. An option can only contain one of directory, disconnect, reprompt, submenu, or transfer. The audio-set defines what message (if any) should be played after a caller selects the option and before the defined action is carried out.

reprompt

Instructs an option to go back to the main menu after playing its prompt.

Parents

option

Children

none

Attributes

none

submenu

Jump to another attendant as a sub-menu.

Parents

option

Children

none

Attributes

name

The name of an attendant to which control should be transferred.

Remarks

After the call enters the attendant named by the name attribute, the behavior will be as if that attendant was called directly, with the following exceptions:

- If the languages element defined in the sub-attendant includes the currently selected language, then it will be used without presenting the caller with the language selection menu.
- If the currently selected language is not defined and two or more other languages are defined, then the user will be presented with the language menu.
- If the sub-attendant has only one language but if differs from the one currently selected, then the language of the sub-attendant will be used.

There will be an additional menu option of * available, which returns the caller to this attendant. The caller will hear the menu prompt upon return.

transfer

Instructs an option to transfer the call to a given phone number.

Parents

option

Children

none

Attributes

destination

The phone number to which the call should be transferred.

Remarks

The destination must be an RFC-3966 compliant phone number URL.

voicemail

Transfers the call directly to voicemail. The call jumps directly to the user's greeting message without ringing the phone.

Parents

option

Children

none

Attributes

mailbox

The phone number of the mailbox to which the call is transferred.

Example AAML Document

```
<attendant name="chicago">
  <languages>
    <audio>For English press one, pour Francais tirez deux.</audio>
    <language>en</language>
    <language>fr</language>
  </languages>
  <greeting>
    <audio-set>
      <audio lang="en">
        Fake Corp, making stuff happen.
      </audio>
      <audio lang="fr">
        Fake Corp, nous faisons des choses se produire.
      </audio>
    </audio-set>
  </greeting>
  <extension-dialing/>
  <menu>
    <audio-set>
      <audio lang="en">
        To look up an employee press one. To be connected to campus
        security press two. For the message of the day press four. If
        you are done press four or just hang up.
      </audio>
      <audio lang="fr">
        Pour cherche person composez l'une. S'il vous pouvez parlerr
        avec le securite tirez deux...
      </audio>
    </audio-set>
    <option ord="1">
      <audio-set>
      </audio-set>
```

```
    <directory facility="chicago.us.oracle"/>
  </option
  <option ord="2">
    <audio-set>
      <audio lang="en">
        Transferring to the campus security. One moment please.
      </audio>
      <audio lang="fr">
        Une moment.
      </audio
    </audio-set>
    <transfer destination="tel:+13126518315"/>
  </option>
  <option ord="3">
    <audio-set
      <audio lang="en">
        Coming from the north side, take Lake Shore Drive to Jackson,
        turn West...
      </audio>
      <audio lang="fr">
        ...
      </audio>
    </audio-set>
    <reprompt/>
  </option>
  <option ord="4">
    <audio-set>
      <audio lang="en">
        Goodbye.
      </audio>
      <audio lang="fr">
        Au revoir.
      </audio>
    </audio-set>
    <disconnect/>
  </option>
</menu>
```

```
    <handlers>
      <noinput type="warning">
        <audio-set>
          <audio lang="en">
            Awaiting response.
          </audio>
          <audio lang="fr">
            ...
          </audio>
        </audio-set>
      </noinput>
      <noinput type="disconnect">
        <audio-set>
          <audio lang="en">
            Thank you for calling, goodbye.
          </audio>
          <audio lang="fr">
            ...
          </audio>
        </audio-set>
      </noinput>
      <nomatch>
        <audio-set>
          <audio lang="en">
            You have entered an incorrect option.
          </audio>
          <audio lang="fr">
            Ce n'est pas une chois correcte.
          </audio>
        </audio-set>
      </nomatch>
    </handlers>
  <attendant>
```

Index

I

L

M

O

P

Q

R

S

T

U

V