Oracle® Communications Billing and Revenue Management

Elastic Charging Engine 11.3 System Administrator's Guide

Release 7.5

E70769-05

July 2018

Oracle Communications Billing and Revenue Management Elastic Charging Engine 11.3 System Administrator's Guide, Release 7.5

E70769-05

Contents

Preface

This guide provides instructions for administrating Oracle Communications Billing and Revenue Management (BRM) Elastic Charging Engine (ECE).

Audience

This document is intended for system administrators, system integrators, database administrators (DBA), and other individuals who are responsible for managing ECE charging servers and ensuring that the software is operating in the manner required for your business. This guide assumes that users have a working knowledge of Linux, Solaris, Oracle Coherence, and Oracle NoSQL Database.

Documentation Accessibility

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

Access to Oracle Support

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

Accessing Oracle Communications Documentation

ECE documentation and additional Oracle documentation; such as Oracle Database documentation, is available from Oracle Help Center:

- `http://docs.oracle.com`

Additional Oracle Communications documentation is available from the Oracle software delivery Web site:

- `https://edelivery.oracle.com`

Document Revision History

The following table lists the revision history for this book.

Version	Date	Description
E70769-01	April 2016	Initial release.
E70769-02	September 2016	Documentation updates for ECE 11.3 Patch Set 1. ■ Added the following section: Removing Excess Failed Updates in the BRM Gateway Suspense Queue ■ Updated the following information: Updated the header for Troubleshooting Failed Update Requests from BRM All procedures involving MBeans
E70769-03	December 2016	Documentation updates for ECE 11.3 Patch Set 2. ■ Added the following section: Configuring Default System Currency
E70769-04	August 2017	Documentation updates for ECE 11.3 Patch Set 4. ■ Added the following chapter: Backing Up and Restoring ECE ■ Updated the following section: Restoring the ECE System
E70769-05	July 2018	Documentation updates for ECE 11.3 Patch Set 8. ■ Updated the following section: Restoring a Complete System Backup

1

ECE System Administration Overview

This chapter provides an overview of system administration tasks you perform in Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) and the tools you use to perform them.

Overview of System Administration Tasks

To administer ECE, you perform the following tasks:

- Deploy ECE software nodes onto hosts
- Monitor ECE
  - Monitor physical hosts running ECE nodes
  - Monitor ECE clusters
  - Monitor ECE nodes
  - Monitor ECE services
- Start and stop ECE
  - Start ECE charging servers in one or more ECE nodes
  - Stop ECE charging servers in one or more ECE nodes
  - Start and stop Rated Event Formatter processes
- Deploy and manage Oracle NoSQL database data store instances
- Purge ECE rated events from the Oracle NoSQL database

Overview of ECE System Administration Tools

To administer ECE, you use the following tools:

- Elastic Charging Controller (ECC)

  Use ECC for managing ECE nodes in the cluster.

  See "Using the Elastic Charging Controller to Manage Nodes".
- Rated Event Formatter

  Configure the Rated Event Formatter to purge unwanted rated events from Oracle NoSQL Database.

  See "Managing Persisted Data in the Oracle NoSQL Database".

Overview of Third-Party System Administration Tools

To administer ECE, you use the following third-party tools:

- Java Monitoring and Management Console (JConsole) or any JMX client

  Use JConsole or another JMX client for setting ECE configuration parameters.

  See the discussion of using the configuration service in "Configuring the ECE System".

- Oracle Enterprise Manager Cloud Control (by way of Oracle Application Management Pack for Oracle Communications)

  Use Oracle Enterprise Manager Cloud Control to monitor ECE nodes.

  You can use Oracle Application Management Pack for Oracle Communications to monitor ECE nodes and clusters. Oracle Application Management Pack for Oracle Communications provides management capabilities for Oracle Communications Billing and Revenue Management (BRM) and other supported Oracle Communications applications. For detailed information about the management capabilities provided by Oracle Application Management Pack for Oracle Communications, see *Oracle Application Management Pack for Oracle Communications System Administrator's Guide*.

Directory Placeholders Used in This Guide

The following placeholders are used in this guide to refer to the directories that contain ECE system components. For example, *ECE_Home* is where the ECE Server software is installed.

Table 1–1 Directory Placeholders

Placeholder	Directory
ECE_Home	The directory in which ECE server software is installed. This directory contains the ECE Server directory (**/oceceserver**) and the ECE SDK directory (**/ocecesdk**) and various installation-related files.
PDC_Home	The directory in which PDC software is installed.

2 Setting Up and Managing Elastic Charging Engine Security

This chapter discusses how to implement security during system administration of Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE).

See *BRM Elastic Charging Engine Security Guide* for an overall discussion of how to install, configure, and use ECE securely as well as information about security prerequisites.

Setting Up User Accounts and User Groups

Access to ECE files is controlled by creating user accounts and granting specific permissions in the Elastic Charging Controller (ECC). After you have created user groups and set permissions, users can log in through ECC and manage ECE files.

Create the following user accounts to manage ECE files:

- UNIX accounts:
  - An administrator to run and manage ECE processes
  - If needed, users to manage the rated event files
  - If needed, users to manage ECE file systems
- Non-UNIX accounts:
  - Users created in the **keystore.jks** file exclusively for securing ECE processes from unauthorized access. See "About the Keystores" for more details.

You must also create a UNIX user group that includes all users, including the administrator.

For information about the permissions that you should grant these users and groups, see "Managing ECE Permissions".

Creating UNIX User Accounts and Groups

Create UNIX accounts and groups using UNIX commands in ECC. To create user accounts, use the **useradd** command. To create groups, use the **groupadd** command.

Managing ECE Permissions

This section describes the permissions that you should restrict for a secure ECE installation. Strict governance of file permissions prevents accidental overwriting and misuse of the system.

About Permission Types

Access to ECE files is controlled by creating user accounts and granting specific permissions in ECC. For information about which user accounts to create, see "Setting Up User Accounts and User Groups". After you have created user groups and set permissions, users can log in through ECC and manage ECE files.

ECE uses the traditional UNIX read, write, and execute permission types. In ECC, use the **chmod** command to set file permissions and the **chown** command to change file owners.

The **chmod** command can change the permissions of files and directories that have been created. You use the **setfacl** command to set permissions for files that have not been created at the time of installation, such as log files in the *ECE_Home*/**oceceserver/logs** directory.

Restrict permissions as much as possible. You may choose to have either a single administrative user with all permissions who runs ECE core processes and manages the rated event files and other directories, or to create multiple users with specific permissions to carry out these tasks.

Configuring Specific File Permissions

To ensure the integrity of the system, access to certain files should be tightly regulated.

Restrict access to the following files:

- **jmxremote.password**: ensure this file has read permission for the user and no permission to either the group or the world (Unix 400).
- **keystore.jks**: ensure only you have read permission.
- **server.jks**: ensure only you have read permission.
- **charging-cache-config.xml**: ensure this file has only read permission for the user.
- **charging-coherence-override-*XXX*.xml**: ensure this file has only read permission for the user.

To restrict access to these files:

1. Log in to ECC.
2. Navigate to the *ECE_Home*/**oceceserver/config** directory.
3. Enter the following commands:

```
chmod 400 jmxremote.password
chmod 400 keystore.jks
chmod 400 server.jks
chmod 400 charging-cache-config.xml
chmod 400 charging-coherence-override-mode.xml
```

Granting ECE User Permissions

Follow these guidelines when granting user permissions:

- Grant the administrator user all permissions.

- Grant execute permissions to only the user assigned to start processes.

> **Note:** On Linux, all directories must have an execute permission to the user.

- If you created users to manage the rated event files, grant them read and write permissions for the rated event directory.
- If you created users to manage file systems, grant them read and write permissions to the directories that they will manage.

To grant permissions to users, use the **chmod** command in ECC. To change file owners, use the **chown** command in ECC. To change the permissions for files, use the **setfacl** command.

The UNIX group containing all of the users should have the following directory permissions:

- Read and write permission to *ECE_Home*/**oceceserver/config** and *ECE_Home*/**oceceserver/logs**
- Read permission to *ECE_Home*/**oceceserver/lib**
- Read and execute permission to *ECE_Home*/**oceceserver/bin**

About the Default umask for ECE Users

You control permissions for new files by adjusting the default **umask**. You check your current **umask** by entering the **umask** command with no additional parameters. Users can also change their **umask** if they wish.

The default **umask** is set to **007** to exclude all users that are not in the same group.

If no default **umask** is set, any new files will be created with read-write-execute permissions granted to all users. Instead of having a **umask** that changes the default permissions globally, run the **setfacl** command to set permissions for files. For example, to set all logs in the logs directory to **600** for the logs that are created and logs that will be created in the future:

```
setfacl -m user::rw-,group::---,other::--- ECE_HOME/logs/*.log
```

About External Permissions

Permissions for non-UNIX accounts are defined within the **permissions.xml** file. When you install ECE, you create a principal administrator alias and password for Oracle Coherence cluster security, which is granted all permissions. The following example shows the content of **permissions.xml** after installation:

```
<permissions>
  <grant>
    <principal>
      <class>javax.security.auth.x500.X500Principal</class>
      <name>CN=Administrator,OU=DN</name>
    </principal>
    <permission>
      <target>*</target>
      <action>all</action>
    </permission>
  </grant>
</permissions>
```

where *Administrator* and *DN* are the DName credentials that were entered in the KeyStore Credentials screen at installation.

For more information about initial configuration of these fields, see *BRM Elastic Charging Engine Installation Guide*.

Managing Passwords in ECE

Access to ECE files is controlled by creating user accounts and granting specific permissions in ECC. For information about which user accounts to create, see "Setting Up User Accounts and User Groups". For information about granting permissions, see "Managing ECE Permissions".

Each UNIX-based user account is password protected and can be managed through ECC. These passwords do not expire by default. You can choose to set up password expiration using the **chage** command.

In addition to UNIX-based user accounts, you also create a non-UNIX account for access to external applications. ECE connects to Oracle Communications Billing and Revenue Management (BRM) and Pricing Design Center (PDC) to load and store the pricing and customer data required to charge for network events. For secure communication between ECE and these other applications, credentials are encrypted and stored in the keystore. See "About Managing External Application Passwords".

The passwords from ECE used to integrate with external systems do not expire by default. Change the secret keys used to encrypt them regularly.

About Managing External Application Passwords

When installing ECE, the passwords that you enter for Oracle Coherence security and for connecting to BRM or PDC are automatically encrypted. The encryption keys are stored in keystores and the password to the keys are stored in the **jmxremote.password** file. You can use the **encrypt** utility from ECC to update the encrypted BRM and PDC passwords.

About the Keystores

The keystores are file-based credential stores that store the encryption keys used by ECE for cluster security.

ECE uses two keystores:

- **keystore.jks**: used for secure communication between ECE and BRM or PDC
- **server.jks**: used for Oracle Coherence cluster security and enabling SSL across the cluster nodes

Each key in a keystore has an alias name and is password protected. The keystores themselves are also password protected.

The Installer prompts you for the following key credentials when you install ECE:

- The alias name and key password for Oracle Coherence cluster security.
- The key password for connecting to BRM and PDC. The password and alias name are the same for all BRM and PDC instances. The alias name cannot be changed.
- The password for the keystores. The Installer automatically uses the same password for **keystore.jks** and **server.jks**.

For more information about the initial configuration of the key credentials, see *BRM Elastic Charging Engine Installation Guide*.

The **keystore.jks** and **server.jks** files are created in *ECE_Home*/**oceceserver/config** when you install ECE.

About keystore.jks

ECE uses **keystore.jks** to ensure secure interaction between ECE and the BRM and PDC updaters. ECE communicates with BRM and PDC using the following clients and XML files:

- Pricing Updater: encrypted password stored in **JMSConfiguration.xml**
- BRM Gateway: encrypted password stored in **JMSConfiguration.xml**, encrypted connect string stored in **BRMConnectionConfiguration.xml**
- Customer Updater: encrypted password stored in **QueueConfiguration.xml**

When one of these clients tries to connect to BRM or PDC, it gets an encrypted password from the configuration file. To decrypt this password, the client uses the key password from **jmxremote.password** to access the key from **keystore.jks**. The client then uses this key to decrypt the encrypted password and connect to BRM or PDC.

Encrypting New Passwords

If the password used to connect to BRM or PDC is changed or if a new BRM or PDC instance is added, you must encrypt the new password and manually update the encrypted password in the configuration files. Encrypting the password ensures secure communication between ECE and BRM or PDC.

To encrypt a new password for BRM or PDC:

1. Log in to ECC.
2. Enter the following command:

   ```
   encrypt string  keystore_pw
   ```

   where *string* is the new password to be encrypted and *keystore_pw* is the password to the keystore.

   The **encrypt** utility returns a message containing the encrypted value for the new password.

To update the encrypted password in the configuration files:

1. Do one of the following:
   - If you are updating a BRM password:
     a. Open the *ECE_ Home*/**oceceserver/config/BRMConnectionConfiguration.xml** file.
     b. Search for the following text:

        ```
        <BRMConfigurations>
          <BRMConnectionConfiguration>
            <ConnectString>password</ConnectString>
          </BRMConnectionConfiguration>
        </BRMConfigurations>
        ```

     c. Change the value of *password* to the new encrypted password returned by the **encrypt** utility.

d. Open the *ECE_Home*/**oceceserver/config/QueueConfiguration.xml** file.

e. Search for the following text:

```
<QueueConfigurations>
  <GatewayEntry>
    <QueueConfiguration>
      <HostName>host</HostName>
      <Port>port</Port>
      <sid>sid</sid>
      <UserName>user</UserName>
      <Password>password</Password>
      <QueueName>queuename</QueueName>
      <BatchSize>size</BatchSize>
    </QueueConfiguration>
  </GatewayEntry>
</QueueConfigurations>
```

The values for *host*, *port*, *sid*, *user*, *queuename*, and *size* are already populated. Do not change them.

f. Change the value of *password* to the new encrypted password returned by the **encrypt** utility

- If you are updating a PDC password:

a. Open the *ECE_Home*/**oceceserver/config/JMSConfiguration.xml** file.

b. Search for the following text:

```
<MessagesConfigurations>
  <JMSDestination>
    <Password>password</Password>
  </JMS Destination>
</MessagesConfigurations>
```

c. Change the value of *password* to the new encrypted password returned by the **encrypt** utility.

2. Save and close the updated files.

3. Restart the node. For information about restarting nodes, see "Starting and Stopping ECE".

Checking Keystore Validity

To check keystore validity:

1. Log in to ECC.

2. Enter the following commands:

```
keytool -list -v -keystore server.jks
password: storepassword
keytool -list -v -storetype jceks -keystore keystore.jks
password: storepassword
```

where *storepassword* is the password to the keystore.

ECC returns the contents of **server.jks** and **keystore.jks**, indicating that the keystore is valid.

If ECC denies permission, ensure that you have set the permissions to **server.jks** and **keystore.jks** as described in "Configuring Specific File Permissions".

If ECC does not return the contents of the JKS files, it could be for the following reasons:

- The keystore JKS file was corrupted.

  A system administrator must recreate the keystore.
- The user forgot the password.

  Access will be denied until the ECE user goes through the company approved process for recreating the keystore with appropriate accounts.

The ECE installer creates a self signed certificate. Your company's authorized personnel can create a certificate and get it signed by a signing authority or self sign it. Deployment of all keystores must be validated for authenticity.

Setting Up Cluster Security

This section explains how to perform tasks related to ECE cluster security. To restrict access to the ECE cluster (an Oracle Coherence cluster), you must set up an authorized hosts list. You can optionally enable SSL for intra-cluster communication, in which case you must also enable Well Known Addresses (WKA).

Adding Trusted Hosts

The trusted host list is set up at installation in the Coherence Grid Security screen of the Installer. This list allows only specified hosts to connect to the cluster. You can add more hosts to the list in the charging override file.

The override file contains a section **<authorized-hosts-list>** that contains host address of each physical machine participating in the cluster. The address could be a server host name or an IP address. IP address is preferred if the hosts are multi-homed. Only the IP address that the network is binding to should be given. Using IP addresses also allows you to set a range of IP addresses which is not possible when using host names.

To add trusted hosts:

1. Open the *ECE_Home***/oceceserver/config/charging-coherence-override-secure-prod.xml** file in a text editor.
2. In the trusted hosts list, add new trusted hosts below those specified at installation by adding the following text:

   ```
   <cluster-config>
     <authorized-hosts>
       <host-address>host_ip1</host-address>
       <host-address>new_host_ip</host-address>
       <host-range>
         <from-address>ip_range_start1</from-address>
         <to-address>ip_range_end1</to-address>
       </host-range>
       <host-range>
         <from-address>new_ip_range_start</from-address>
         <to-address>new_ip_range_end</to-address>
       </host-range>
     </authorized-hosts>
   </cluster-config>
   ```

   where:

   - *host_ip1* is the pre-existing IP address of a host specified at installation.

- *new_host_ip* is a specific host that you are adding.
- *ip_range_start1* is the beginning of a pre-existing range of IP addresses specified at installation.
- *ip_range_end1* is the end of the pre-existing range of IP addresses specified at installation.
- *new_ip_range_start* is the beginning of the new range of IP addresses.
- *new_ip_range_end* is the end of the new range of IP addresses.

You can choose to enter *new_host_ip*, both *new_ip_range_start* and *new_ip_range_end*, or all three.

3. Save and close the file.

Securing Intra-Cluster Communication

Intra-cluster communication can be strengthened by enabling SSL. ECE supports SSL versions 2 and 3 and TLS version 1. For an overview of common SSL concepts, see *Oracle Coherence Security Guide*.

You should expect enabling SSL to have a negative impact on ECE performance. As such, SSL should only be enabled if required. Consider the implications of the performance degradation before enabling intra-cluster SSL.

By default, SSL is not enabled. You can select to enable it when you run the ECE installer. You can also enable SSL after installing ECE. See *BRM Elastic Charging Engine Security Guide* for more information about SSL.

Enabling SSL Within the ECE Cluster

To enable SSL:

1. Enable WKA. See "Enabling Well Known Addresses".
2. Open the *ECE_Home*/**oceceserver/config/ece.properties** file.
3. Search for the following system property:

```
tangosol.coherence.override=
```

4. Add **charging-coherence-override-secure-prod.xml** as follows:

```
tangosol.coherence.override=charging-coherence-override-secure-prod.xml
```

5. Open the *ECE_Home*/**oceceserver/config/defaultTuningProfile.properties** file.
6. Search for the following system properties:

```
tangosol.coherence.ssl.storepassword=
tangosol.coherence.ssl.keypassword=
```

7. Set the properties as follows so that two-way SSL can use **server.jks** for encrypted data:

```
tangosol.coherence.ssl.storepassword=keystorepassword
tangosol.coherence.ssl.keypassword=coherencepassword
```

where:

- *keystorepassword* is the password to **server.jks**
- *coherencepassword* is the password for Oracle Coherence grid security alias that was entered in the KeyStore Credentials screen at installation

8. Save and close the files.

To generate an SSL self-signed certificate:

1. Log in to the ECC.
2. Run the following command:

   ```
   keytool -genkeypair -dname "cn=Administrator, ou=DN" -alias admin -keypass
   password -keystore ECE_Home/oceceserver/config -storepass storepassword
   ```

   where:

   - *Administrator* and *DN* are the DName credentials that were entered in the KeyStore Credentials screen at installation
   - *password* is the key password for the admin alias
   - *storepassword* is the password for the keystore

   You can choose to set certificate expiry using the **-validity** option.

You can optionally have the certificate signed by a signing authority, but for ECE functionality a self-signed certificate is sufficient.

Enabling Well Known Addresses

The Well Known Addresses (WKA) mechanism allows cluster members to discover and join a cluster using unicast instead of multicast. You may need to enable WKA if your data center policy prohibits multicast. If you enable SSL for intra-cluster communication, you must enable Well Known Addresses (WKA).

To enable WKA:

1. Open the *ECE_ Home***/oceceserver/config/charging-coherence-override-secure-prod.xml** file.
2. Specify cluster members by adding text as follows:

   ```
   <cluster-config>
     <unicast-listener>
       <well-known-addresses>
         <socket-address id="id">
           <address>ip_address</address>
           <port>port</port>
         </socket-address>
         ...
       </well-known-addresses>
     </unicast-listener>
   </cluster-config>
   ```

   where:

   - *id* is the ID for a particular cluster member
   - *ip_address* is the IP address of the cluster member
   - *port* is the value specified in the member's unicast listener port

Ensure that the list of WKA members is the same on every cluster member so that no cluster member operates independently from the rest of the cluster.

For more information about setting up a WKA host list, see the discussion of Well Known Addresses in *Oracle Coherence Developer's Guide*.

Securing Inter-Cluster Communication

Inter-cluster communication between the ECE cluster and the BRM system can be strengthened by enabling SSL.

BRM Gateway can connect to the Connection Manager (CM) by using SSL. See "Enabling SSL Communication between BRM Gateway and the CM" for information.

Enabling SSL Communication between BRM Gateway and the CM

If you use SSL to secure connections between Java PCM clients and the CM, you must enable SSL communication between BRM Gateway and the CM.

For information about enabling SSL for Java PCM clients in BRM, see the discussion of implementing system security in *BRM System Administrator's Guide*.

To enable SSL communication between BRM Gateway and the CM:

1. Do one of the following, where *BRM_Home* is the directory in which you installed BRM:
   - If you enabled one-way SSL server and client authentication in the CM, verify that the *BRM_Home***/wallet/client/cwallet.sso** file was copied into your ECE installation (in the directory of your choice).
   - If you enabled two-way SSL server and client authentication between the CM and its PCM clients, verify that the *BRM_Home***/wallet/server/cwallet.sso** file was copied into your ECE installation (in the directory of your choice).

     For information about enabling two-way SSL server and client authentication between the CM and its PCM clients, see the discussion of enabling SSL in the CM in *BRM System Administrator's Guide*.
2. If the file listed in the previous step is not in your ECE installation, copy it from BRM to the ECE directory of your choice.
3. Copy the following JAR files to the *ECE_Home***/oceceserver/lib** directory:
   - *BRM_Home***/jars/osdt_cert.jar**
   - *BRM_Home***/jars/osdt_core.jar**
   - *BRM_Home***/jars/oraclepki.jar**
4. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home***/oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.
5. Expand **charging.connectionConfigurations.brmConnection**.
6. Expand **Attributes**.
7. Specify values for the following attributes:
   - **sslEnabled**: Change the value from **0** to **1**.

- **wallet**: Enter the path to the directory on your ECE installation where you copied your Oracle wallet from your BRM environment (**cwallet.sso**).

8. Stop and restart BRM Gateway.

   See "Starting and Stopping BRM Gateway" for information.

SSL communication between BRM Gateway and the CM is now enabled.

Setting Up Password-less SSH Between the Driver and Servers

You must set up bi-directional password-less Secure Shell (SSH) logins between the driver machine and each server machine for ECC to work. Password-less SSH allows servers to connect to the driver and synchronize ECE files.

To set up password-less SSH:

1. Log in to the ECC.
2. Run the following commands:

   ```
   ssh-keygen -t dsa
   ssh-copy-id -i ~/.ssh/id_dsa.pub user@host
   ```

   where:

   - *user* is the user name set for all host machines at installation in the ECE Cluster Details screen. For more information about this screen, see *BRM Elastic Charging Engine Installation Guide*.
   - *host* is the name of the server for which the password-less SSH is being established.

To test passwordless SSH:

1. Log in to ECC.
2. Run the following command:

   ```
   ssh user@host
   ```

   where:

   - *user* is the user name set for all host machines at installation in the ECE Cluster Details screen. For more information about this screen, see *BRM Elastic Charging Engine Installation Guide*.
   - *host* is the name of the server for which the password-less SSH is being established

   If a password is requested, password-less SSH setup has failed. Ensure that you have followed the steps for setting up password-less SSH correctly and run the test again.

3 Starting and Stopping ECE

This chapter provides instructions for starting and stopping Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) nodes.

Caution: Restarts of the ECE system are not expected. Stopping *all* charging server nodes and restarting them results in loss of data. Do not stop all charging server nodes unless you want all data to be removed from Coherence caches. See "Restoring the ECE System" for more information.

About Starting and Stopping ECE

Starting and stopping ECE involves starting and stopping ECE nodes. An ECE node is a Java Virtual Machine (JVM) process running Elastic Charging Engine software. Most ECE nodes are members of the Coherence cluster. Nodes for ECE software processes that are not Coherence members typically need to be restarted after you make configuration changes to the software processes on a running system.

You use Elastic Charging Controller (ECC) to start and stop ECE nodes based on the role name assigned to them in the ECE topology file (*ECE_home*/**oceceserver/config/eceTopology.conf** where *ECE_home* is the directory in which ECE is installed). For example, charging server nodes have the role **server**; the command **start server** starts all charging server nodes in the cluster.

ECE must be in a usage-processing state to be able to process usage requests sent from the Elastic Charging Client. When you are initially starting the ECE system, you place ECE in a usage-processing state by running ECE nodes in a specific order:

1. Start the charging server nodes (**start server**).
2. Start the **configLoader** utility (**start configLoader**).
3. Start Pricing Updater (**start pricingUpdater**).
4. Start Customer Updater (**start customerUpdater**).

For example, when you first load your data into ECE, you do the following:

1. Start ECC.

   ```
   ./ecc
   ```

2. Run the following commands in the listed order.

   These commands start the charging server nodes and load data into the ECE caches in the required order.

```
start
start configLoader
start pricingUpdater
start customerUpdater
```

Stopping all charging server nodes will result in loss of cache data. You can stop a specific charging server node with no loss of data in Coherence caches. See "Stopping Charging Server Nodes".

Non-charging server nodes can be stopped and restarted with no loss of cache data. See "Stopping Non-Charging Server Nodes".

See "Using the Elastic Charging Controller to Manage Nodes" for information about how to start and stop nodes using ECC.

Starting an ECE Standalone System

Start ECE nodes in the following order when starting an ECE standalone system:

1. Start the charging server nodes (**start server**).
2. Start the **configLoader** utility (**start configLoader**).
3. Start Pricing Updater (**start pricingLoader**).
4. Start Customer Updater (**start customerLoader**).

Starting and Stopping Diameter Gateway

Note: This procedure assumes you have done the following:

- Created Diameter Gateway instances by using the **charging.diameterGatewayConfigurations** MBean in a JMX editor.
- Defined each instance in the *ECE_home*/**oceceserver/config/eceTopology.conf** file so that it can be started and stopped using ECC.

For information about creating Diameter Gateway instances and defining Diameter Gateway instances in your topology, see the post-installation tasks in *BRM Elastic Charging Engine Installation Guide*.

To start Diameter Gateway:

1. Verify that the Elastic Charging Server is running with all required data.

   All ECE nodes used for Diameter credit-control processing must be running: charging server nodes, Pricing Updater, Customer Updater, External Manager (EM) Gateway, BRM Gateway, and Rated Event Formatter.

   You must have at least one charging server node running.

   All data, Oracle Communications Billing and Revenue Management (BRM) data and Pricing Design Center (PDC) pricing data, must be loaded into the ECE data grid.

2. Verify that the ECE nodes are in a usage-processing state:

   a. Start the ECE charging servers (if they are not started).
   b. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   c. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   d. In the editor's MBean hierarchy, expand the **ECE State Machine** node.
   e. Expand **StateManager**.
   f. Expand **Attributes**.
   g. Make sure the **stateName** attribute is set to **UsageProcessing**.
3. On the driver machine, change to the *ECE_home*/**oceceserver/bin** directory.
4. Start ECC:

   ```
   ./ecc
   ```

5. Do one of the following:
   - To start all Diameter Gateway instances, run the following command:

     ```
     start diameterGateway
     ```

   - To start a specific Diameter Gateway instance, run the following command:

     ```
     start Instance_Name
     ```

     where *Instance_Name* is the name of the Diameter Gateway instance as you defined it in the node-name column of the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

   At startup, Diameter Gateway connects to the network interface and port on which it listens for Diameter messages.

To stop Diameter Gateway, run one of the following commands from ECC:

- To stop all Diameter Gateway instances, run the following command:

  ```
  stop diameterGateway
  ```

- To stop a specific Diameter Gateway instance, run the following command:

  ```
  stop Instance_Name
  ```

For information about configuring Diameter Gateway, see the post-installation tasks in *BRM Elastic Charging Engine Installation Guide*.

Starting and Stopping RADIUS Gateway

Note: This procedure assumes you have added RADIUS Gateway instances and configured them in your topology. See "Adding RADIUS Gateway Nodes for Authentication and Accounting" and "Configuring RADIUS Gateway Nodes" for more information.

To start RADIUS Gateway:

1. Verify that the Elastic Charging Server is running with all required data.
2. Do one of the following:
   - To start all RADIUS Gateway instances, run the following command:

     ```
     start radiusGateway
     ```

   - To start a specific RADIUS Gateway instance, run the following command:

     ```
     start Instance_Name
     ```

     where *Instance_Name* is the name of the RADIUS Gateway instance as you defined it in the node-name column of the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

   At startup, RADIUS Gateway connects to the network interface and port on which it listens for RADIUS requests.

To stop RADIUS Gateway, run one of the following commands from ECC:

- To stop all RADIUS Gateway instances, run the following command:

  ```
  stop radiusGateway
  ```

- To stop a specific RADIUS Gateway instance, run the following command:

  ```
  stop Instance_Name
  ```

Starting and Stopping Rated Event Formatter

To start Rated Event Formatter:

1. Verify that the Oracle NoSQL database is running.
2. On the driver machine, change to the *ECE_home*/**oceceserver/bin** directory.
3. Start ECC:

   ```
   ./ecc
   ```

4. Do one of the following:
   - To start all Rated Event Formatter instances, run the following command:

     ```
     start ratedEventFormatter
     ```

   - To start a specific Rated Event Formatter instance, run the following command:

     ```
     start Instance_Name
     ```

     where *Instance_Name* is the name of the Rated Event Formatter instance as you defined it in the node-name column of the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

     When a custom plug-in is configured for a non-BRM system, the Rated Event Formatter instance must start at the same time as the custom plug-in instance.

     If the BrmCdrPluginDirect Plug-in is already running, stop the Rated Event Formatter instance associated with that plug-in and start both the instances at the same time.

   At startup, Rated Event Formatter connects to the Oracle NoSQL database and reads the rated event information from the Oracle NoSQL data store.

To stop Rated Event Formatter, run one of the following commands from ECC:

- To stop all Rated Event Formatter instances, run the following command:

  ```
  stop ratedEventFormatter
  ```

- To stop a specific Rated Event Formatter instance, run the following command:

  ```
  stop Instance_Name
  ```

  The instances associated with the BrmCdrPluginDirect Plug-in and custom plug-in can be stopped and restarted at any time. When the instances are restarted, they continue processing the rated events, including any rated events generated during the downtime.

For information about configuring Rated Event Formatter, see the discussion of implementing ECE with BRM in *BRM Elastic Charging Engine Implementation Guide*.

Starting and Stopping External Manager Gateway

The Elastic Charging Server must be in a usage-processing state before you can start EM Gateway.

ECE uses EM Gateway during the rerating process and to synchronize BRM and ECE customer data. To handle all customer data update requests, EM Gateway must run continuously in a production environment.

EM Gateway and BRM Gateway must be running before rerating is run.

To start EM Gateway:

1. On the driver machine, change to the *ECE_home*/**oceceserver/bin** directory.
2. Start ECC:

   ```
   ./ecc
   ```

3. Run the following command:

   ```
   start emGateway
   ```

To stop EM Gateway, run the following command:

```
stop emGateway
```

Starting Charging Server Nodes

To start charging server nodes:

1. On the driver machine, change to the *ECE_home*/**oceceserver/bin** directory.
2. Start the EEC:

   ```
   ./ecc
   ```

3. Do one of the following:
   - To start all of the charging server nodes in your topology, run the following command:

     ```
     start [server]
     ```

   - To start a specific charging server node, run the following command:

     ```
     start nodeName
     ```

where *nodeName* is the name of the node specified in the node-name column of the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

Stopping Charging Server Nodes

Caution: Stopping all charging server nodes will result in loss of data on a standalone system.

Server redundancy is a minimum requirement of ECE installations to enable built-in high availability and fault tolerance features that prevent all charging server nodes from stopping.

If you stop all charging server nodes, you will lose data in all Coherence caches (such as the data in the Customer cache, Chargeoffering cache, Serviceeventmap cache, and so on). The charging server nodes are not intended to be stopped.

See "Restoring the ECE System" for more information.

To stop a specific charging server node, run the following command:

```
stop nodeName
```

where *nodeName* is the name of the node specified in the node-name column of the **ECE_Home/oceceserver/config/eceTopology.conf** file.

Stopping Non-Charging Server Nodes

Non-charging server nodes can be stopped and restarted with no loss of cache data. For example, stopping the following ECE nodes disrupts temporarily real-time data updates coming from other applications, but the updates are processed when the ECE system is restarted:

- Customer Updater (**stop customerUpdater**)
- Pricing Updater (**stop PricingUpdater**)
- BRM Gateway (**stop brmGateway**)
- Rated Event Formatter (**stop ratedEventFormatter**)
- EM Gateway (**stop emGateway**)

Starting and Stopping BRM Gateway

Elastic Charging Server must be in a usage-processing state before you can start BRM Gateway.

BRM Gateway, and EM Gateway, must be running before rerating is run.

To start the BRM Gateway:

1. On the driver machine, change to the *ECE_home*/**oceceserver/bin** directory.
2. Start ECC:

   ```
   ./ecc
   ```

3. Run the following command:

```
start brmGateway
```

To stop BRM Gateway, run the following command:

```
stop brmGateway
```

Starting and Stopping Data-Loading Utility Nodes

The data-loading utility nodes (configLoader, pricingLoader, customerLoader) start as Coherence nodes and then stop when they are done with their processing. You do not need to stop them.

Restoring the ECE System

Restarts of the ECE system are not expected because ECE is built to always be running. ECE has built-in high availability and fault tolerance when server redundancy is employed. Server redundancy is a minimum requirement of ECE installations.

If you stop all charging server nodes on a standalone system, you will lose data in Coherence caches.

Patching ECE to upgrade it to a new version does not require restarting the system when you perform a rolling upgrade. A rolling upgrade applies the patch while the system is running. See the discussion about upgrading to new releases in *BRM Elastic Charging Engine Installation Guide* for more information about rolling upgrades.

If a restart is required after all ECE nodes are stopped, restore the ECE system by doing the following:

1. In PDC, publish all the PDC pricing data (the metadata, setup, pricing, and profile data) from the PDC database to ECE by running the following command:

   ```
   ImportExportPricing -publish -metadata -config -pricing -profile -target [ece]
   ```

2. In ECE do the following:

   a. On the driver machine, change to the *ECE_home*/**oceceserver/bin directory**.

   b. Start ECC.

      ```
      ./ecc
      ```

   c. Start ECE processes and gateways in the following order:

      Note: Depending on your installation, you can start Diameter Gateway, RADIUS Gateway, both Diameter Gateway and RADIUS Gateway, or none.

      ```
      start server
      start configLoader
      start pricingUpdater
      start customerUpdater
      start emGateway
      start brmGateway
      start ratedEventFormatter
      start diameterGateway
      start radiusGateway
      ```

      All data is now back in the ECE data grid.

Real-time data updates (coming from BRM) that had been temporarily disrupted due to the shutdown are processed upon restart.

Diameter Gateway processing that had been temporarily disrupted due to the shutdown is resumed upon restart. For example, the translation of Diameter requests into ECE Java API requests and the processing of push notifications from Elastic Charging Server (the processing of ECE JMS notifications) is resumed.

Troubleshooting Starting ECE

This section describes troubleshooting tips for starting ECE nodes.

Start Command Fails When Trying to Start Nodes in a Large Cluster

If you try to start nodes in a large cluster, and the **start** command fails, the problem may be due to the limitation on how many simultaneous SSH connections the machines in your environment can make from another machine or to another machine. The **start** command, by default, attempts to start 10 nodes simultaneously using 10 different threads. If the maximum number of open sessions permitted per network connection is less than 10 on the driver machine, the **start** command will fail. Likewise, if the maximum number of simultaneous connections to the SSH daemon on the server machine is less than the number of threads attempting to make connections on the server machine, the **start** command will fail.

To resolve the issue, change the number of nodes the **start** command will start simultaneously, or (if your runtime environment allows) change the limit of simultaneous SSH connections your environment can make from or to a machine.

For Linux systems, to set the limit for how many simultaneous SSH connections can be made from the driver machine in your environment:

1. On the driver machine, stop the SSH daemon.
2. Open the **/etc/ssh/sshd_config** file.
3. Set the **MaxSessions** property.

   MaxSessions specifies the maximum number of open sessions permitted per network connection. The default is **10**.
4. Save and close the file.
5. Start the SSH daemon.

To set the limit for how many simultaneous SSH connections can be made to the server machine in your environment:

1. On server machine, stop the SSH daemon.
2. Open the **/etc/ssh/sshd_config** file.
3. Set the **MaxStartups** property to be equal to or more than the number of nodes the **start** command starts at one time.

   MaxStartups specifies the maximum number of concurrent unauthenticated connections to the SSH daemon on the server machine. Additional connections are dropped until authentication succeeds or the LoginGraceTime expires for a connection. The default is **10**.
4. Save and close the file.
5. Start the SSH daemon.

To set the number of nodes the **start** command will start simultaneously:

1. On the driver machine, open the *ECE_home*/**oceceserver/ece.properties** file.
2. Add the following line to the file:

   ```
   numberOfStartCommandsExecutorThreads=number_of_threads
   ```

   where *number_of_threads* is the number of threads the **start** command uses to start the (same) number of nodes simultaneously.

   number_of_threads must be equal to or less than the value of the **MaxSessions** property set for the given environment.
3. Save and close the file.

4 Configuring the ECE System

This chapter describes how to configure the underlying system on which Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) runs and describes the client-side configurations that control how requests are sent to ECE.

See *BRM Elastic Charging Engine Implementation Guide* for information about configuring charging business rules at the server level that influence how usage requests are charged, including taxation configurations.

About ECE Configuration

You configure ECE configuration parameters by doing one of the following:

- Prior to starting the ECE charging servers, by editing the XML files located in the *ECE_home*/**oceceserver/config/management** directory, where *ECE_home* is the directory in which ECE is installed.
- After starting the ECE charging servers, by using the Java Management Extensions (JMX) MBean editor of your choice.

You configure usage-charging business parameters that control how ECE charges offline and online usage requests as well as system configuration parameters that influence how ECE charging servers operate. You can configure standard JVM tuning parameters for ECE nodes that are running JVMs in the cluster. See "Configuring JVM Tuning Parameters".

Several system configuration parameters are initially configured during ECE installation when you enter values for fields as requested by the Oracle Universal Installer GUI installation processes. These parameters can be modified after installation if needed.

Other system configuration parameters are set in a configuration file and properties file by using a text editor immediately after ECE installation.

The configuration parameters of configuration files in the *ECE_home*/**oceceserver/config/management** directory can be edited through JConsole on a running system. For example, usage-charging business parameters can be edited by using the ECE configuration service that exposes ECE configuration parameters as MBeans. See the discussion about accessing and editing ECE MBean parameters in *BRM Elastic Charging Engine Implementation Guide* for more information.

You can also edit configuration parameters by editing the configuration XML files directly using a text editor, but these edits need to be made *before* you start the charging servers. After ECE is running, you must use the JConsole to edit configuration parameters so that your edits can impact the running system (through the MBean API.)

For information about the configurable parameters in ECE and how you configure them, see the following topics:

- Initial Configuration
- System-Level Configuration
- Usage-Charging Configuration
- ECE Configuration File Reference

When you are within the network (inside the firewall), you can configure ECE configuration parameters remotely by logging in to the Coherence management JMX server using the host name and JMX port.

About Centralized Configuration

ECE offers centralized configuration. See the discussion about accessing and editing ECE MBean parameters in *BRM Elastic Charging Engine Implementation Guide* for more information.

For centralized configuration to work, two-way password-less SSH must be configured between client and server machines. See "Setting Up Password-less SSH Between the Driver and Servers" for instructions.

Initial Configuration

You typically install ECE on a machine that is meant to be used to administer the ECE system and is referred to as the *driver machine*. After installing ECE on the driver machine, you perform an initial configuration on the driver machine as follows:

- Specify the machines/hosts for each Coherence node in the *ECE_home*/**oceceserver/config/eceTopology.conf** file. See "Configuring ECE Topology" for more information.
- Specify required properties in the *ECE_home*/**oceceserver/config/ece.properties** file, including the ECE root directory, the ECE user name, and the IP address of the driver machine (driverIp).

All configuration settings configured on the driver machine are saved to XML files in the driver machine *ECE_home*/**oceceserver/config/management** directory.

After you complete the initial configuration on the driver machine, if your ECE cluster includes multiple machines, you use the Elastic Charging Controller (ECC) application to deploy ECE from the driver machine to all machines specified in your topology (using the **sync** command). See the discussion of ECE post-installation tasks in *BRM Elastic Charging Engine Installation Guide* for more information.

Important: You must provision the other machines on which you will deploy ECE. The **sync** command does not provision the other machines for you. See the discussion of ECE pre-installation tasks in *BRM Elastic Charging Engine Installation Guide* for information about provisioning machines for an ECE integrated.

Note: ECC commands are meant to be run from the driver machine for administering the ECE system (managing the nodes of the cluster).

System-Level Configuration

System-level configuration files are located in *ECE_home*/**oceceserver/config/*.xml**. See "ECE Configuration File Reference" for a summary of system-level configuration files. See the comments in each file for descriptions of the file parameters, including default values and accepted range of values.

For further information about ECE system-level configuration, see the following topics:

- Configuring JVM Tuning Parameters
- Configuring Trusted Hosts of the Cluster
- Configuring Coherence

Usage-Charging Configuration

See *BRM Elastic Charging Engine Implementation Guide* for information about configuring usage charging business rules (setting parameters that control usage-charging behavior).

You can configure usage-charging business rules in the following two ways:

- Before starting ECE by directly editing the *ECE_home*/**oceceserver/config/management/*.xml** files on the driver machine.
- After starting ECE by using the configuration service. See the discussion about accessing and editing ECE MBean parameters in *BRM Elastic Charging Engine Implementation Guide* for more information.

Configuring ECE Topology

Each node you define in the topology file must have a role associated with it. The role identifies the application that ECC starts when you enter commands for managing nodes. For example, when you enter ECC command **start configLoader**, ECC starts the node with the role **configLoader.** When you enter ECC command **start server**, ECC starts all nodes with role **server**.

To configure ECE topology:

Important: The topology file is pre-configured with several nodes that are required by ECE. Do not delete existing rows in this file.

1. Open the *ECE_home*/**oceceserver/config/eceTopology.conf** file.
2. Add a row for *each* node on each physical server machine in the Coherence cluster.

   For example, if you have three physical server machines and each physical server machine has three nodes, add nine rows.
3. For each row, enter the following information:
   - Name of the JVM process for that node.

You can assign an arbitrary name. This name is used to distinguish processes that have the same role.

- Role of the JVM process for that node.

  Each node in the ECE cluster plays a certain role.

- Host name of the physical server machine on which the node resides.

  For a standalone system, use **localhost**.

  A standalone system means that all ECE-related processes are running on a single physical server machine.

- (For multihomed hosts) IP address of the server machine on which the node resides.

  For those hosts that have multiple IP addresses, specify the IP address so that Coherence can be pointed to a port.

- Whether you want the node to be JMX-management enabled.

  See "Enabling a Charging Server Node for JMX Management".

- The tuning profile for that node. Each node is associated with a JVM tuning file. See "Configuring JVM Tuning Parameters".

4. (For SDK sample programs) To run the SDK sample programs by using **sdkCustomerLoader**, uncomment the line that defines the **sdkCustomerLoader** node.
5. Save the file.

Enabling a Charging Server Node for JMX Management

For each unique IP address in your physical topology, you must enable one charging server node for JMX management. When the JMX-enabled node starts, it provides a JMX management service on the specified topology's host and port. The service exposes ECE MBeans so that you can edit the MBean attributes by using a JMX editor, such as JConsole. The service also enables ECC to verify the status of nodes as it enables the Coherence management framework.

Note: Any ECE node can be enabled for JMX management, but you must enable the charging-server nodes for JMX management for central configuration of ECE to work. Charging-server nodes are always running, and enabling them for JMX management exposes MBeans for all ECE node processes (such as simulators and data loaders).

To enable a charging server node for JMX management:

1. Open the *ECE_home*/**oceceserver/config/eceTopology.conf** file.
2. In the row for *one* charging server node (node with role **server**), for *each* physical server machine or unique IP address in the cluster, provide the following information:

   - JMX port of the JVM process for that node.

     Enter any free port, such as **9999**, for the charging server node to be the JMX-management enabled node.

Choose a port number that is not in use by another application.

The default port number is **9999**.

- Specify if you want the node to be JMX-management enabled by entering **true** in the **start CohMgt** column.

  For charging server nodes (nodes with the role **server**), always enable JMX-management when a JMX port is supplied.

  Only one charging server node per physical server should be JMX-management enabled.

  Because multiple charging server nodes are running on a single physical machine, you set **CohMgt=true** for only one charging server node on each physical machine. Each machine must have one charging server node with **CohMgt=true** for centralized configuration of ECE to work.

3. Save the file.

Configuring JVM Tuning Parameters

Configure JVM tuning parameters for garbage collection and heap size tuning.

Each row in the topology file represents an ECE component that is a running JVM in the cluster. In the topology file, you can specify a JVM tuning profile file for each node defined. This allows you to provide specific tuning settings for each node in the cluster. Multiple nodes can point to the same tuning profile.

To configure JVM tuning parameters:

1. Open the *ECE_home*/**oceceserver/config/defaultTuningProfile.properties** file.

   You can create your own JVM tuning file and save it in this directory. You can name the file what you want.

2. Set the parameters as needed.
3. Save the file.
4. In the topology file, ensure your JVM tuning file is associated with the node to which you want the parameters to apply.

   The JVM tuning file is referenced by name in the topology file.

   See "Configuring ECE Topology" for information about the topology file.

Deploying JVM Tuning Parameter Updates onto a Running System

After configuring JVM tuning parameters, you can deploy JVM tuning parameter updates onto a running ECE system.

To deploy JVM tuning parameter updates onto a running system:

1. Log on to the driver machine.
2. Change directory to the *ECE_home*/**oceceserver/bin** directory.
3. Start the Elastic Charging Controller.

   ```
   ./ecc
   ```

4. Run the **sync** command to deploy the ECE installation onto server machines:

   ```
   sync
   ```

The **sync** command copies the relevant files of the ECE installation (which includes your JVM tuning parameter changes) onto the server machines you have defined to be part of the ECE cluster.

5. Open the *ECE_home*/**oceceserver/config/eceTopology.conf** file and define the following nodes at the top of the file (before the charging server nodes):

   - Updater nodes
   - Gateway nodes
   - Formatter nodes

   These processes must be restarted before restarting the charging server nodes.

6. Run the **rollingUpgrade** command to perform a rolling restart of ECE nodes that are currently running on your topology.

   ```
   rollingUpgrade
   ```

   One by one, each currently running node listed in the topology file is brought down and joined back to the cluster.

   When you run the **rollingUpgrade** command with no parameters specified, all running nodes are upgraded (charging server nodes, data-loading utility nodes, data updater nodes, and so on) except for simulator nodes (nodes that have the role **simulator**).

Configuring Trusted Hosts of the Cluster

As part of initial configuration, configure the trusted hosts for the cluster. Trusted hosts are the machines or processes that are allowed from a security perspective to be part of the cluster. Because the cluster contains private customer data, ensure that only trusted hosts can access this data. You enter trusted host information during the ECE installation process. See "Adding Trusted Hosts" for information about adding or modifying trusted host information for your ECE system.

See *BRM Elastic Charging Engine Security Guide* for overall information about installing and administering a secure system.

Configuring ECE System-Level Settings

See the following topics for information about configuring ECE system-level parameters:

- Configuring Coherence
- Configuring Logging
- Configuring Notifications

Configuring Coherence

The ECE configuration files related to configuring Coherence are as follows:

Note: In an ECE standalone system, the default values in these files typically do not need to be modified.

- *ECE_home*/**oceceserver/config/charging-cache-config.xml**

- *ECE_home*/**oceceserver/config/charging-coherence-override-dev.xml**
- *ECE_home*/**oceceserver/config/charging-coherence-override-prod.xml**
- *ECE_home*/**oceceserver/config/charging-coherence-override-secure-prod.xml**
- *ECE_home*/**oceceserver/config/charging-pof-config.xml**

Refer to the comments in each file for information about configuring their parameters.

See the Oracle Coherence documentation for general information about configuring Coherence.

Configuring Logging

Configure logging parameters so that you have the log levels set to the granularity you want. You can configure logging for each node in the cluster. The *ECE_home*/**oceceserver/logs** directory contains the log files for each node on each machine of your topology.

You can configure log levels and control logging in the following ways:

- Controlling log level for each node or for the entire grid by way of JMX (using JConsole).

  A grid-level log level means that the log level is applied to all Elastic Charging Server nodes that run the Coherence work manager agent.

  See "Setting Log Levels by Editing MBeans".

- Controlling log level for all nodes in the cluster.

  You set a global log level for the entire cluster by configuring the *ECE_home*/**oceceserver/config/log4j.properties** file. You must edit this file before starting ECE charging servers.

- Turning logging on or off using ECC.

  Nodes that are started using ECC produce a log under the logs directory using the filename format *node_name***.log**.

  To look up the logs produced by the nodes under the logs directory, use ECC **cat**, **ls**, and **tail** commands.

For collecting diagnostic information, Oracle recommends that you turn on ECC feedback mode which produces extra information when running commands. For example:

```
set feedback true
```

The feedback mode setting is saved in your local profile so you do not need to set it every time you start the Elastic Charging Controller.

Setting Log Levels by Editing MBeans

You can set the log level of the ECE module or modules for which you are setting the log level by ECE functional domain. Use this method to turn on debugging for all ECE modules that are used in the flow of an ECE functional domain that is associated with your debugging scenario. For example, if you are debugging a problem in which events are not being rerated properly, use this method to turn on debugging for all ECE modules used in the functional domain of *rerating*.

To set log levels for ECE functional domains:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ECE Logging** node.
2. Expand **Configuration**.
3. Expand **Attributes**.
4. Select **FunctionalDomains**.
5. Double-click in the attribute’s **Value** field.

   A list of ECE functional domains for which you can turn on debugging appears in the field.
6. In the list, copy the name of the ECE functional domain relevant to your debugging scenario (for example, **Policy**).
7. Under **Attributes**, select **LoggerLevels**.
8. Double-click in the attribute **Value** field.

   A list of log levels appears in the field.
9. Scroll through the list to determine which log level you want to use for the functional domain you chose in step 6 (for example, **DEBUG**).
10. Under **Configuration**, expand **Operations**.
11. Select one of the following operations:
    - To set the log level for one ECE node, select **setLogLevelForFunctionalDomain**.
    - To set the log level for the grid, select **setGridLogLevelForFunctionalDomain**.

      This applies the log level to all Elastic Charging Server nodes that run the Coherence work manager agent.
12. Specify values for the following operation parameters:
    - **p0**: Enter the name of the ECE functional domain you copied in step 6.

      Enter the name exactly as it appears in the list, including brackets, capitalization, and so on.
    - **p1**: Enter the log level you chose in step 9.
13. Click the operation button.

Configuring Charging-Server Health Thresholds

This section describes how to configure a charging-server health threshold.

About the Charging-Server Health Threshold

To mitigate charging server node failures that might threaten your system’s ability to handle your customer base, you can configure a charging-server health threshold. A charging-server health threshold is the minimum number of charging server nodes needed for your customer base. If the number of charging server nodes running on your system goes below the threshold, ECE stops processing usage requests and issues a **SystemHealthException**. For example, if you require six charging server nodes to process requests for your customer base, you set the threshold to **6**. But, if the number of available charging server nodes falls to 5, ECE stops processing usage requests and issues the exception. ECE continues to process update, management, query, top-up, debit, and refund requests.

When setting a charging-server health threshold, note the following:

- If a threshold is *N*, you need to run at least *n+ 1* nodes to have uninterrupted usage processing during a rolling upgrade.
- For an integrated system, have a minimum of two charging server nodes per machine (provided the total number of charging server nodes can handle the normal expected throughput for your system).
- For a standalone system for design or test environment, note the following guidelines:
  - Although you can use one charging server node in a design or test environment, setting a charging-server health threshold of **1** is not a valid configuration for deploying into a runtime environment.
  - The minimum configuration for an ECE standalone system is **3**, which accounts for two charging server nodes plus an additional node if both charging server nodes fail. In this case, you would set a charging-server health threshold to **2**.

See "Configuring the Charging-Server Health Threshold" for information about configuring the charging-server health threshold.

Configuring the Charging-Server Health Threshold

To configure a charging-server health threshold:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.
2. Expand **chargingServer**.
3. Expand **Attributes**
4. Set the **degradedModeThreshold** attribute to the minimum number of charging server nodes needed for your customer base (the number that can handle the normal expected throughput for your system). The default is **0**.

Checking the Ongoing Health of ECE Charging Client Nodes

You configure overload protection as a client-side configuration.

To check the ongoing health of ECE charging client nodes:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the appropriate ECE charging client node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ChargingClient** node.
2. Expand **BatchRequestService**.
3. Expand **Attributes**.
4. Check the value of the **SystemHealth** attribute:
   - **HEALTHY**: Charging client nodes are functioning.
   - **DEGRADED**: Charging client nodes are unavailable.

Configuring System Overload Protection

Because ECE is always on, it needs to handle system overload. The following scenarios might cause an overloaded system:

- An undersized ECE deployment or lack of infrastructure for usage growth
- Large batches of offline records
- Bulk customer updates that trigger numerous update requests

Though ECE has dynamic scalability to enable you to adjust sizing for peak times, overload protection is intended for an exceptionally overloaded system. ECE protects the system infrastructure from meltdown by controlling the number of submitted requests for processing. System overload causes the ECE executing requests to become stuck or busy. To ensure ECE does not get overloaded by a high volume of requests, it can be monitored and controlled in real time. When the maximum throughput is exceeded or there are spikes in the number of usage requests, ECE does not suffer from performance degradation if system overload protection measures are in place.

About Overload Protection Infrastructure

Overload protection uses thread pools to accept and process requests submitted to the system. Thread pools improve performance when executing large numbers of updates because of the reduced per-update overhead. They also provide a means of bounding and managing the resources, including requests.

When the request throughput from ECE is too large for the system to handle, ECE reduces its throughput until it reaches a sustainable error-free level. It informs the client of any submitted requests that are not able to be processed.

As with setting the charging-server health threshold, when overload protection is enabled, only usage requests are impacted. Usage requests continue to be accepted until the capacity reaches the configurable pending count.

The value of the pending count should generally be at least equal to the number of thread counts. Select this value carefully, based on the expected throughput of the ECE instance and the expected latency of each request as revealed by your performance testing results.

Update, management, query, top-ups and debit refunds requests are always accepted, even when the system is overloaded.

Configuring Overload Protection

You configure overload protection as a client-side configuration.

To configure overload protection:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.
2. Expand **ChargingClient**.
3. Expand **BatchRequestService**.
4. Set the **OverloadProtection** attribute to **true**.

About MBean Attributes Used to Configure Overload

The **BatchRequestService** MBean contains the attributes to configure the system for an unexpected throughput. Table 4–1 describes the attributes that can be configured for overload protection.

Note: When you configure the following MBean attributes, they are not saved. If the client is restarted, attributes are reset to their default values.

Table 4–1 MBean Attributes to Configure Overload Protection

MBean Attribute	Description
AcceptablePendingCount	Number of requests that are accepted and queued to be processed. ECE rejects these requests if the number of pending requests in the queue exceeds this value. After performance testing and monitoring the **ThreadPendingCount** attribute, this value can be determined. It is typically larger than the ECE thread count.
AcceptedTaskCount	Number of requests that have been accepted for processing since the start of the ECE instance. This attribute is read only.

Table 4–1 (Cont.) MBean Attributes to Configure Overload Protection

MBean Attribute	Description
BatchSize	Size of the ECE batch when it is submitted for processing.
BatchTimeOut	Amount of time ECE waits before the batch is submitted for processing, irrespective of how much of the batch is filled.
OverloadProtection	Flag that enables overload protection that is disabled by default (set to **false**).
RejectedTaskCount	Number of requests rejected since the start of the ECE instance. **RejectedTaskCount** plus **AcceptedTaskCount** should be equal to the total number of submitted requests on a single ECE instance. This attribute is read only.
ThreadPendingCount	Number of requests that are pending in the queue. Monitor this attribute before setting a value of **AcceptablePendingCount**. This attribute is read-only.

About Request Prioritization

When prioritizing requests, you can control when requests are sent through ECE.

New client-side ECE request queues can be set up by modifying the **brs-client-config.xml** file in the Elastic Charging Client JAR file. For more information about creating new queues, refer to "Configuring Client-Side ECE Request Queues".

In addition, in a scenario where both online network mediation and offline network mediation clients are sending requests through ECE and the throughput is at its limit, the online request is given priority over the offline request.

Configuring Client-Side ECE Request Queues

You can configure the request queues that your charging clients use to submit requests to ECE. The Elastic Charging Client, which is installed on the charging client (such as the network mediation client application), can use queues for sending requests to the Elastic Charging Server charging server nodes. For each queue, you can set the local thread pool for submitting requests for processing and for handling responses from the charging server nodes. You can also set the batch size and batch time out of each queue.

To configure client-side ECE request queues:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home***/oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.
2. Expand **ChargingClient**.
3. Expand **BatchRequestService**.

4. Set the thread pool size, batch time out, and batch size attributes for the request queues.

   For descriptions of each attribute, see the documentation for the BRSStatMXBean for **oracle.communication.brm.charging.brs** in *ECE Java API Reference*.

Configuring Default System Currency

You can configure ECE to use a default system currency for charging subscribers. During rating, ECE uses the subscriber's primary currency or the secondary currency to charge subscribers. If the currency used in the rate plans does not match the subscriber's primary or secondary currency, ECE uses the default system currency, US dollars.

To configure a default system currency:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.
2. Expand **charging.server**.
3. Expand **Attributes** and select **systemCurrencyNumericCode**.
4. Set the numeric code of the currency for the system.

   For descriptions of each attribute, see the documentation for the BRSStatMXBean for **oracle.communication.brm.charging.brs** in *ECE Java API Reference*.

Configuring Housekeeping: Expired Object Clean Up

Housekeeping tasks can efficiently manage ECE server memory. One such housekeeping task is the clean up of *expired objects*, historic data that is no longer used for processing customer updates and usage information.

ECE utilizes existing architecture to clean up expired objects with update and usage request processes. Cleaning up expired objects during the update and usage request process ensures regular system upkeep and avoids processing from occurring during peak processing times, potentially impacting performance.

About Cleaning Up Expired Objects in Update Requests

With update requests, the retention time of the expired object is checked for expiration. When a customer has an update request and an expired object for that customer has exceeded its retention time, it is removed during the update request process. You can configure the retention time of the expired object. See "Configuring Expired Object Clean Up in Update Requests" to configure the retention time of expired objects in update requests.

The following expired objects are processed in a customer's update request:

- Purchased charge offers
- Purchased alteration (discount) offers
- Balance items
- Expired audit data
  - Purchased charge offers
  - Purchased alteration (discount) offers
  - Products (Services)
  - Used alteration agreements
  - Used distribution agreements

The clean up time of a customer is performed as part of an update request if the last clean up has been more than one day. A timestamp at the customer level indicates the last clean up time. For example, a customer has balance updates at 10:00, 11:00, 12:00, and 13:00 on day 1 and another update at 10:30 on day 2. The first clean up is done at 10:00 on day 1, and clean ups are skipped at 11:00, 12:00, and 13:00. The next clean up occurs at 10:30 on day 2 as part of the update request.

Configuring Expired Object Clean Up in Update Requests

To configure the retention time (in days) of an expired object in an update request:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.
2. Expand **charging.expirationConfiguration**.
3. Expand **Attributes**.
4. Specify values for the expiration configuration attributes listed in Table 4–2.

About MBean Attributes Used to Configure Expired Object Retention Time

Table 4–2 lists MBean attributes and their default values that are used to configure expired object retention time. The maximum allowed retention time is 180 days and the minimum allowed retention time is 0 days.

Table 4–2 MBean Attributes to Configure Expired Object Retention Time

MBean Attribute	Default Retention Time (in Days)
expiredAuditRetentionIntervalInDays	60
expiredPurchasedProductRetentionIntervalInDays	30
expiredPurchasedAlterationRetentionIntervalInDays	30

Table 4–2 (Cont.) MBean Attributes to Configure Expired Object Retention Time

MBean Attribute	Default Retention Time (in Days)
expiredRatingProfileRetentionIntervalInDays	30
defaultExpirationRetentionIntervalInDays	30
defaultExpiredBalanceItemRetentionIntervalInDays	30

Expired audit objects share a single common retention time, whereas other expired objects have individual retention times.

Expired balance items are configured at the balance element level. If there is no configuration for a given balance element, **defaultExpiredBalanceItemRetentionIntervalInDays** is used. Table 4–3 lists the expired balance elements and example retention times.

Table 4–3 Expired Balance Element Retention Times

Expired Balance Element	Retention Time (in Days)
FREE_MIN	60
BONUS_POINTS	15

About Cleaning Up Expired Objects in Usage Requests

When the **TERMINATE** or **CANCEL** operation type in a customer's usage request is processed, the following objects are checked for expiration and are removed if the object has expired.

- Active sessions
- Balance reservations

Note: Expired active sessions and balance reservations are considered for removal immediately. You cannot configure a retention time for these objects.

Expired active sessions and corresponding expired balance reservations are cleaned up so that reserved balances are made available for future usage requests. All expired active sessions are terminated with no allowances for exceptions. Only used units are considered for terminating expired active sessions. In most instances, clean up is performed for the product for which the **TERMINATE** or **CANCEL** operation type is being processed but not for all the products for the given customer. One exception is if products share the same balance object as the original product that is being cleaned up for a given customer, those products will be cleaned up as well.

The requests from the active sessions that have expired are converted to **TERMINATE** requests and processed.

Setting Eviction Policies for the Identity Cache

The identity cache is part of the Elastic Charging Client and stores the public user identity information of customers; this cache is populated as requests are processed. Each time requests come into the system for new customers, their identity information is newly created in the cache. You can set eviction policies for the identity cache to remove entries (or units) from it when a maximum number of units (high-units parameter) is reached.

The identity cache is configured as a near cache with its front local scheme being a size-limited local cache. The local cache uses a HYBRID eviction policy that constitutes LRU (Least Recently Used) and LFU (Least Frequently Used) policies. If the entries from the front scheme are to be evicted (when the configured high-units number is reached), then those entries that have been least recently or least frequently used will be evicted.

By default, the identity cache is configured with a **HYBRID** eviction policy and a high-units of **20,500,000**.

For more information about the hybrid eviction policy for Coherence caches, refer to the Oracle Coherence documentation.

Configuring Notifications

You can configure ECE to send notifications, either in-session notifications that are part of the usage response or external notifications that are JMS messages. ECE can generate notifications for the following:

- **Notifications for BRM**. Notifications that are used by Oracle Communications Billing and Revenue Management (BRM) that are enabled as a best practice when integrating with the BRM system. When you use ECE as a charging engine for BRM, you can trigger notifications for sending information (updates) to BRM from ECE. For more information, see the chapter on implementing ECE with BRM in *BRM Elastic Charging Engine Implementation Guide*.
- **Notifications for online network mediation**. Notifications that are used by online network mediation software programs. These notifications are also used by Diameter Gateway. Typically, these notifications are used to send information to the customer. For more information, see the chapter on sending requests from Diameter Gateway to charging servers in *BRM Elastic Charging Engine Implementation Guide*.
- **Notifications for policy and charging rules functions (PCRF)**. Notifications that are used by PCRFs for policy and control. When you use ECE as a customer profile repository (SPR) for a PCRF, you can trigger notifications for sending information to the PRCF from ECE. For more information, see the chapter on implementing ECE with a PCRF in *BRM Elastic Charging Engine Implementation Guide*.

For ECE to publish external notifications, configure the JMS credentials for the JMS server on which the notification queue (JMS topic) resides. See the discussion of implementing ECE with BRM in *BRM Elastic Charging Engine Implementation Guide* for instructions.

Configuring ECE Data-Loading Utilities and Data Updaters

When you install ECE, you provide information for configuring the following data-loading utilities, which are used for loading data into ECE and updating that data:

- Data-loading utilities
  - **configLoader**
- Data-loading utilities used only for ECE standalone systems
  - **pricingLoader**

   - **customerLoader**

- Data updaters
   - Pricing Updater: Keeps ECE synchronized with Pricing Design Center (PDC)
   - Customer Updater: Keeps ECE synchronized with BRM asynchronously (not in real time)
   - External Manager (EM) Gateway: Keeps ECE synchronized with BRM in real time

To change configurations:

- For data-loading utilities, see the discussion of data-loading utilities in *BRM Elastic Charging Engine Implementation Guide*.
- For Customer Updater, see the discussion of implementing ECE with BRM in *BRM Elastic Charging Engine Implementation Guide*.
- For Pricing Updater, see the discussion of implementing ECE with PDC in *BRM Elastic Charging Engine Implementation Guide*.
- For EM Gateway, see the discussion of configuring EM Gateway in *BRM Elastic Charging Engine Implementation Guide*.

For information about asynchronous and synchronous data updates, see the discussion about synchronizing BRM and ECE customer data in *BRM Elastic Charging Engine Concepts*.

Configuring Usage-Charging Settings

See *BRM Elastic Charging Engine Implementation Guide* for information about configuring settings that control how Elastic Charging Server processes usage requests.

Updating Subscriber Lifecycle States for BRM

ECE supports the BRM subscriber lifecycle state feature. When new lifecycle states are added in BRM, you must update the ECE lifecycle state configuration so that the BRM information and ECE information remain synchronized. See the discussion of implementing ECE with BRM in *BRM Elastic Charging Engine Implementation Guide* for information about updating subscriber lifecycle states in ECE.

Adding Diameter Gateway Nodes for Online Charging

The ECE installer process creates a single instance of a Diameter Gateway node (**diameterGateway1**) that is added to your topology (added to your *ECE_home***/oceceserver/config/eceTopology.conf** file). By default, this single node listens to all network interfaces for Diameter messages.

For a standalone system, a single node is sufficient for basic testing directly after installation; for example, to test if the Diameter client can send a Diameter request to the Diameter Gateway node. Add additional Diameter Gateway nodes to your topology, configure them to listen on the different network interfaces in your environment, and perform performance testing to determine the minimum number of Diameter Gateway nodes needed for your customer base (the number that can handle the normal expected throughput for your system).

When adding Diameter Gateway nodes, note the following:

- In an ECE integrated system, have a minimum of two charging server nodes per machine (provided the total number of charging server nodes can handle the normal expected throughput for your system). The guideline is to have two Diameter Gateway node instances to allow for failover and additional nodes as needed to handle the expected throughput for your system.
- In a standalone system, the minimum configuration is three charging server nodes and two Diameter Gateway node instances to allow for failover. Server redundancy is a minimum requirement of ECE installations.

To add Diameter Gateway nodes:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home***/oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.
2. Expand **charging.diameterGatewayConfigurations**.
3. Expand **Operations**.
4. Select **addDiameterGatewayConfiguration**.
5. In the **name** parameter, enter a name for the Diameter Gateway node.
6. Click the **addDiameterGatewayConfiguration** button.

   You have created a Diameter Gateway node.

   (Optional) To fully configure the Diameter Gateway node now, specify values for all Diameter Gateway node configuration properties. See "Specifying Diameter Gateway Node Properties". Alternatively, first create multiple nodes, and later configure them.
7. To add a peer to the Diameter Gateway node, expand **charging.diameterGatewayPeerConfigurations**.
8. Expand **Operations** and select **addPeer**.
9. Specify the values for the following parameter:
   - **peerName.** Enter the name of the Diameter peer.
10. Click the **addPeer** button.

    The peer is added to Diameter Gateway.
11. Expand **charging.diameterGatewayPeerConfigurations.***Peer_Name*, where *Peer_ Name* is the name of the Diameter peer.
12. Expand **Attributes**.
13. For each peer connected to the Diameter Gateway, configure alternative peers by specifying values for the following attribute:
    - **alternatePeerNames.** Enter the name of the alternative peer for the specified Diameter peer. You can specify two alternative peers for each Diameter peer. If

the peer connected to Diameter Gateway fails or if it is unavailable, Diameter Gateway routes the notifications to the alternate peers configured.

14. Open the *ECE_home*/**oceceserver/config/eceTopology.conf** file.
15. Add a row for the Diameter Gateway node instance.
16. For that row, enter the following:
    - The name of the JVM process that you used when you created the Diameter Gateway node instance in the JMX editor.
    - The role of the JVM process for that node, **diameterGateway**.
    - The host name of the physical server machine on which the Diameter Gateway node resides.
    - The JVM tuning file that contains the tuning profile for the Diameter Gateway node.
17. Save the file.

Specifying Diameter Gateway Node Properties

For each Diameter Gateway node, you must specify node properties for configuring the node to communicate with your network as well as for tuning the node for optimal performance.

To specify Diameter Gateway node properties:

1. Access the ECE MBeans:
    a. Log on to the driver machine.
    b. Start the ECE charging servers (if they are not started).
    c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
    d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

       The **eceTopology.conf** file also contains the host name and port number for the node.
    e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.
2. Expand **charging.diameterGatewayConfigurations.***Instance_Name*, where *Instance_Name* is the name of the instance to configure.
3. Expand **Attributes**.
4. Specify values for all the attributes required to configure the instance.

   See Table 4–4 for attribute descriptions and default values.
5. Change directory to the *ECE_home*/**oceceserver/bin** directory.
6. Start the Elastic Charging Controller:

   ```
   ./ecc
   ```
7. Do one of the following:
    - If the Diameter Gateway instance is *not* running, start it.

      The instance reads its configuration information by name at startup.
    - If the Diameter Gateway instance is running, stop and restart it.

For information about stopping and starting Diameter Gateway instances, see the discussion about starting and stopping ECE in *BRM Elastic Charging Engine System Administrator's Guide*.

Table 4–4 Diameter Gateway Node Configuration Parameters

Name	Default	Description
name	"diameterGateway1"	The name of the Diameter Gateway instance. Name Diameter Gateway node instances consistently and uniquely (for example, **diameterGateway1**, **diameterGateway2**, and so on). If you want to use the same name for the Diameter Gateway instances, for example, for the disaster recovery configuration, ensure that the cluster name is unique for each of these instances. ECE uses both **name** and **clusterName** to identify the Diameter Gateway instance. The name you specify must match the name for this Diameter Gateway instance in the node-name column of the *ECE_home*/**oceceserver/config/eceTopology.conf** file. If you change the name of an existing instance by using the JMX editor, you must update the name of the instance in the topology file.

clusterName	" "	The cluster name of the Diameter Gateway instance.
		
		Name Diameter Gateway node instances consistently and uniquely (for example, **cluster1**, **cluster2**, and so on).
		
		Specify a unique cluster name if you are configuring Diameter Gateway nodes with the same name, which is required for the disaster recovery configuration. For example, you can configure two Diameter Gateway nodes as follows:
		
		Diameter Gateway 1
		
		`name='diameterGateway1'`
		`clusterName="cluster1"`
		
		Diameter Gateway 2
		
		`name='diameterGateway1'`
		`clusterName="cluster2"`
		
		The cluster name you specify must match the cluster name for this Diameter Gateway instance in the *ECE_home*/**oceceserver/config/charging-coherence-override-secure-prod.xml** file. If you change the cluster name of an existing instance by using the JMX editor, you must update the name of the instance in the Coherence override file.

diameterTrafficPort	"3868"	The port (on the physical host computer that is running the Diameter Gateway node instance) the Diameter Gateway instance listens on for handling Diameter messages. When adding new Diameter Gateway instances, choose a port number that is not in use by another application. When multiple Diameter Gateway instances run on the same physical host computer, each instance must use a different port number. The value set here is used by the Diameter Gateway instance to determine which port to bind to on the server where the Diameter Gateway node instance is running.

Table 4–4 (Cont.) Diameter Gateway Node Configuration Parameters

Name	Default	Description
diameterTrafficHost	""	The network interface (on the physical or virtual host computer that is running the Diameter Gateway node instance) that the Diameter Gateway node binds to and listens on for Diameter messages (sent from Diameter clients). The Diameter Gateway instance uses this value to determine which network interface to bind to on the server where the Diameter Gateway node instance is running. The value can be either an IP address or a host name. The value can also be an empty string. ■ If the value is an IP address or a host name, the Diameter Gateway instance listens for Diameter messages only on that one network interface. ■ If the value is an empty string (default), the Diameter Gateway instance listens for Diameter messages on all network interfaces available on the server.
diameterTrafficHostSctp	""	When SCTP is used, the network interface (on the physical or virtual host computer that is running the Diameter Gateway node instance) that the Diameter Gateway node binds to and listens on for Diameter messages (sent from Diameter clients). The Diameter Gateway instance uses this value to determine which network interface to bind to on the server where the Diameter Gateway node instance is running. The value can be either an SCTP IP address or host name or multiple SCTP IP addresses or host names. The value can also be an empty string. For a multihoming system, multiple IP addresses can be specified separated with a comma (,). For example: `10.240.179.147,10.240.182.149` ■ If the value is an SCTP IP address(es) or a host name(s), SCTP transport is enabled and the Diameter Gateway instance supports Diameter messages that use SCTP transport on those network interfaces. ■ If the value is an empty string (default), SCTP transport is disabled. To use this configuration, your operating system must have SCTP support. Verify that your operating system has SCTP support. If not, install the SCTP system package for your operating system version.

Table 4–4 (Cont.) Diameter Gateway Node Configuration Parameters

Name	Default	Description
originHost	n/a **Note:** Setting a value for this field is mandatory.	Enter the value for the Origin-Host attribute-value pair (AVP) to be sent in the Diameter request. This is a unique identifier that you assign your Diameter Gateway server on its host. It can be any string value. The value set here is used by the Diameter client to identify your Diameter Gateway server (at the application layer) as the connecting Diameter peer that is the source of the Diameter message.
originRealm	n/a **Note:** Setting a value for this field is mandatory.	Enter the value for the Origin-Realm AVP to be sent by the Diameter Gateway in outgoing Diameter requests. This is the signaling realm (domain) that you assign your Diameter Gateway server. You must set the same the origin realm value for all Diameter Gateway instances in the same ECE topology. The value set here is used by Diameter clients to identify your Diameter Gateway server as the source of the Diameter message.
loopback	"false"	Specifies the loopback setting for performance testing. Valid values are: ■ **True.** specifies that the Diameter Gateway instance does not send the credit-control request to ECE. Instead, the Diameter Gateway instance returns the success result code to the network element. ■ **False.** specifies that the Diameter Gateway instance sends the credit-control request to ECE.
ioThreadPoolSize	"10"	The number of threads used by the network I/O thread pool. The network I/O thread that the Diameter Gateway node instance uses for sending and receiving Diameter requests over a network socket using TCP. Valid values are greater than zero and up to any number the system resources allow.
responseTimeout	"10"	The maximum duration in seconds that the Diameter Gateway instance waits for a response from the Diameter client for a notification message the Diameter Gateway has sent to it. If the Diameter Gateway instance does not receive a response from the Diameter client within the specified duration, the Diameter Gateway instance stops waiting for a response and removes the notification from the JMS queue. Valid values are greater than zero and up to any number the system resources allow. Tune this value to the expected workload in the deployed environment.
requestProcessorThreadPool Size	"10"	The number of threads used by the request-processor thread pool. The request-processor thread pool is a Diameter Gateway thread pool that is dedicated to processing Diameter requests handed off to it from the I/O thread pool. Valid values are greater than zero and up to any number the system resources allow. Tune this value to the expected workload in the deployed environment.

Table 4–4 (Cont.) Diameter Gateway Node Configuration Parameters

Name	Default	Description
requestProcessorBatchSize	"10"	The batch size of the Diameter requests handed off by the network I/O thread pool to the request-processor thread pool. Valid values are greater than zero and up to any number the system resources allow. Tune this value to the expected workload in the deployed environment.
watchDogInterval	"30"	The duration in seconds that the Diameter Gateway instance waits before it issues a Device-Watchdog-Request message (DWR).
notificationThreadPoolSize	"10"	The number of threads used by the Diameter Gateway instance to process notification messages. Valid values are greater than zero and up to any number the system resources allow. Tune this value to the expected workload in the deployed environment.
maxNotificationCommitSize	"100"	The maximum number of dequeued notification messages from the JMS topic that can remain uncommitted. If the number of dequeued notification messages from the JMS topic exceeds this number, the Diameter Gateway instance stops reading messages until the read messages are committed.

Table 4–4 (Cont.) Diameter Gateway Node Configuration Parameters

Name	Default	Description
ccFailover	"FAILOVER_ SUPPORTED"	Indicates if the Diameter Gateway instance is operating in a cluster that supports failover. Valid values are: ■ "FAILOVER_SUPPORTED" ■ "FAILOVER_NOT_SUPPORTED" The value set here is the value the Diameter Gateway instance sends for the CC-Session-Failover AVP in all credit-control answers (CCAs) that the instance produces. For more information, see Diameter Credit-Control Application standard at: https://tools.ietf.org/html/rfc4006#section-8.4
creditControlFailureHandling	"RETRY_AND_ TERMINATE"	Indicates how the Diameter client should proceed if a CCA is not received prior to the Tx timeout. Valid values are: ■ "TERMINATE" ■ "CONTINUE" ■ "RETRY_AND_TERMINATE" The value set here is the value the Diameter Gateway instance sends for the Credit-Control-Failure-Handling AVP in all CCAs that the instance produces. For more information, see Diameter Credit-Control Application standard at: https://tools.ietf.org/html/rfc4006#section-8.14
directDebitingFailureHandling	"TERMINATE_OR_ BUFFER"	Indicates how the Diameter client should proceed if a Direct Debit CCA is not received prior to the Tx timeout. Valid values are: ■ "TERMINATE_OR_BUFFER" ■ "CONTINUE" The value set here is the value the Diameter Gateway instance sends for the Direct-Debiting-Failure-Handling AVP in all credit-control answers (CCAs) that it produces. For more information, see Diameter Credit-Control Application standard at: https://tools.ietf.org/html/rfc4006#section-8.15

Adding RADIUS Gateway Nodes for Authentication and Accounting

During ECE installation, if you specified that RADIUS Gateway must be started when ECE is started, the ECE installer process creates a single instance (node) of a RADIUS Gateway (**radiusGateway1**) that is added to your topology (added to your *ECE_ home*/**oceceserver/config/eceTopology.conf** file). By default, this single node listens to RADIUS messages.

For a standalone system, a single node is sufficient for basic testing directly after installation; for example, to test if the RADIUS client can send a RADIUS request to the RADIUS Gateway node. Add additional RADIUS Gateway nodes to your topology, configure them to listen on the different network interfaces in your environment, and test performance to determine the minimum number of RADIUS Gateway nodes needed for your customer base.

When adding RADIUS Gateway nodes, note the following:

- In an ECE integrated system, have two RADIUS Gateway nodes to allow for failover and additional nodes as needed to handle the expected throughput for your system.
- For a standalone system, the minimum configuration is two RADIUS Gateway nodes to allow for failover.

To add RADIUS Gateway nodes:

1. Log on to the driver machine.
2. Change directory to *ECE_home*/**oceceserver/bin**.
3. Start the Elastic Charging Controller:

   ```
   ./ecc
   ```

4. Access the ECE MBeans:
   a. Start the ECE charging servers (if they are not started).
   b. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   c. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   d. In the editor's MBean hierarchy, expand the **ECE Configuration** node.
5. Expand **charging.radiusGatewayConfigurations**.
6. Expand **Operations**.
7. Select **addRadiusGatewayConfiguration**.
8. In the **name** parameter, enter a name for the RADIUS Gateway node.
9. Click the **addRadiusGatewayConfiguration** button.

   You have created a RADIUS Gateway node.
10. Open the *ECE_home*/**oceceserver/config/eceTopology.conf** file.
11. Add a row for the RADIUS Gateway node instance.
12. For that row, enter the following information:
    - Name of the JVM process for the node instance.

      Enter the name used when the node instance was created in the JMX editor.
    - Role of the JVM process for the node instance.

      Enter the role **radiusGateway**.
    - Host name of the physical server machine on which the node resides.
    - JVM tuning file that contains the tuning profile for the node.
13. Save the file.

After adding the RADIUS Gateway nodes, you must specify the node properties for configuring the nodes to communicate with your network and for tuning the node for optimal performance. For information on the RADIUS Gateway configuration properties and default values, see "Configuring RADIUS Gateway Nodes".

Configuring RADIUS Gateway Nodes

You must configure each RADIUS Gateway node to communicate with your network and to perform optimally.

To configure RADIUS Gateway nodes:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).

      See "Starting and Stopping ECE" in *ECE System Administrator's Guide*.
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home***/oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.
2. Expand **charging.radiusGatewayConfigurations.**
3. Expand **Attributes**.
4. Specify values for the attributes listed in Table 4–5:

Note: Changing the values of base attributes affects all RADIUS Gateway node instances in your system.

Table 4–5 RADIUS Gateway Node Base Configuration Attributes

Name	Default	Description
avpName	"Service-Type"	The name of the attribute value pair (AVP) that is used to determine the product type during authentication. This is used in conjunction with **vendorId**.
timeToLive	"30000"	The expiry time (in milliseconds) for the RADIUS requests stored in the ECE cache.
wallet	"opt/wallet"	The path to the Oracle wallet file containing the SSL trusted certificates and the BRM root key for RADIUS Gateway. When RADIUS Gateway is started, the BRM root key in the Oracle wallet file is stored in the memory.
keyPass	"@KEY_PASS@"	The key password required for accessing certificates in the **keystore.jks** file. This password is stored in encrypted format.
queueSize	"8"	The number of incoming requests that can be simultaneously processed by the RADIUS server. Adjust the queue size to correspond to the number of threads.

Table 4–5 (Cont.) RADIUS Gateway Node Base Configuration Attributes

Name	Default	Description
keyStoreLocation	"@KEY_STORE_ LOCATION"	The path to the **keystore.jks** file that contains the certificates to support Extensible Authentication Protocol - Tunneled Transport Layer Security (EAP TTLS) authentication.
vendorId	"0"	The vendor ID of the AVP that you configured to determine the product type. This is used in conjunction with **avpName**.
enableRetransmissionChecks	"true"	The flag that is used to enable or disable the duplicate packet detection feature. This feature is enabled by default. RADIUS Gateway uses this feature to identify duplicate requests from RADIUS clients by validating it against the requests stored in the ECE cache. **Note:** You must restart RADIUS Gateway after enabling or disabling the duplicate packet detection feature.

5. Expand **charging.radiusGatewayConfigurations.***Instance_Name*, where *Instance_ Name* is the name of the RADIUS Gateway node to configure.
6. Expand **Attributes**.
7. Specify values for the attributes listed in Table 4–6:

Table 4–6 RADIUS Gateway Node Instance Configuration Attributes

Name	Default	Description
radiusTrafficPort	"1812"	The number assigned to the port on which RADIUS Gateway listens. Add one **radiusTrafficPort** entry for each port on which you want RADIUS Gateway to listen.
name	"radiusGateway1"	The name of the RADIUS Gateway instance. Name RADIUS Gateway node instances consistently and uniquely (for example, **radiusGateway1**, **radiusGateway2**, and so on). The name you specify must match the name for this RADIUS Gateway instance in the node-name column of the *ECE_home***/oceceserver/config/eceTopology.conf** file. If you change the name of an existing instance by using the JMX editor, you must update the name of the instance in the topology file.
noOfChallenges	"1"	The maximum number of challenges that can be sent to RADIUS clients when Challenge-Handshake Authentication Protocol (CHAP) is used for authentication. A random number within this value is chosen during authentication to carry out the number of challenges for a given authentication session. If the password is authenticated successfully, the challenge process begins and an Access-Challenge message is sent as reply to this request. If any of the challenge responses fail in authentication, an Access-Reject is sent. Upon all successful authentication, an Access-Accept message is sent.

Table 4–6 (Cont.) RADIUS Gateway Node Instance Configuration Attributes

Name	Default	Description
sharedSecret	"e59VPnxr1o5+FGW97w/aMA=="	The common password shared between RADIUS Gateway and Network Access Server (NAS). It is used by the RADIUS protocol for security. Each RADIUS Gateway instance must have a unique password in encrypted format.
ioThreadPoolSize	"16"	The number of selected threads that determines the maximum number of simultaneous processes that RADIUS Gateway can handle. You can increase the number of threads to increase the server performance and reduce the number of threads to reduce the throughput. There is no one criterion for setting the number of threads. Many factors impact the number of threads required, such as the cache size of each CPU, memory size, and swap size. Systems can handle as many as eight threads per CPU. On production systems, set these values higher.

8. Expand **charging.radiusGatewayEapPriorityConfiguration.**
9. Expand **Operations**.
10. Select **addEapType**.
11. Specify values for the parameters listed in Table 4–7:

Table 4–7 RADIUS Gateway Node Extensible Authentication Protocol (EAP) Parameters

Name	Default	Description
id	"21"	The unique identifier of the Extensible Authentication Protocol (EAP) type used for authentication.
name	"TTLS"	The name of the EAP type used for authentication. This is associated with the EAP ID. By default, the following EAP types are supported for authentication: TTLS and MD5.
priority	"1"	The priority set for the EAP type. 1 is the highest priority.

12. Click the **addEapType** button.
13. Change directory to the *ECE_home***/oceceserver/bin** directory.
14. Start the Elastic Charging Controller:

```
./ecc
```

15. Do one of the following:
    - If the RADIUS Gateway instance is *not* running, start it.

      The instance reads its configuration information by name at startup.
    - If the RADIUS Gateway instance is running, stop and restart it.

    For information about stopping and starting RADIUS Gateway instances, see "Starting and Stopping RADIUS Gateway".

Customizing the RADIUS Data Dictionary

This section covers customizing the RADIUS data dictionary.

About the RADIUS Data Dictionary

The data dictionary includes a list of AVPs that are used by RADIUS Gateway to perform authentication and accounting operations. The RADIUS data dictionary contains the standard AVPs that are prescribed in RADIUS Request for Comments (RFC) 2865, 2866, and 2869, and also some sample vendor-specific attributes. You can use the sample vendor-specific attributes as a template for adding custom vendor-specific attributes. The default location of the RADIUS data dictionary file is *ECE_home***/config/radius/radiusDictionary.xml**.

Important: Do not remove, rename, or move the RADIUS data dictionary file to a different location.

Creating a Custom Data Dictionary

You can create a custom data dictionary file by using the *ECE_home***/config/radius/radiusDictionary.xml** file as a template. The default location for your custom data dictionary file is *ECE_home***/config/radius/custom/***dictionary_file*, where *dictionary_file* is the name of your custom data dictionary file. You can add new vendor-specific attributes to your custom data dictionary file. See "Adding Custom Vendor-Specific Attributes".

Selecting a RADIUS Data Dictionary When Using Different NAS Vendors

If you must use NAS servers from multiple vendors, you have the following options:

- If your NAS is RFC 2865 compliant, you can use the RFC2865 data dictionary. This is the preferred solution. Update the dictionary file with any vendor-specific attributes associated with the NAS.
- If your NAS is not RFC 2865 compliant, you can use the RADIUS data dictionary files for adding vendor-specific attributes. See "Adding Custom Vendor-Specific Attributes" for more information.

Adding Custom Vendor-Specific Attributes

In special cases, where you are using NAS servers from multiple vendors, you must add the vendor attribute and code in your custom data dictionary file.

The syntax for adding a vendor-specific attribute is:

```
<?xml version="1.0" encoding="UTF-8"?>
    <dictionary schemaLocation= "radiusDictionary.xsd"
       <vendor value="vendor_ID"name="vendor_name"/>
       </attribute name="attribute_name" vendor="vendor_name" syntax="data_type"
code="attribute_ID"/>
     /dictionary>
```

Table 4–8 lists the vendor-specific attribute values and descriptions.

Table 4–8 Vendor-specific Attribute Values

Parameters	Description
vendor_ID	Number used to identify the NAS or gateway vendor. These numbers are assigned by the Internet Advisory Board (IAB). See your vendor's documentation for details. Some common vendor identification numbers are: ■ 9 (Cisco) ■ 10415 (3GPP) ■ 2636 (Juniper)
vendor_name	Name of the vendor.
attribute_name	Name of the attribute. This must be unique. **Important:** Do not use the same attribute name as used in the default RADIUS data dictionary file. Using the same attribute name in the custom data dictionary file overrides the attribute values in the default RADIUS data dictionary file.
attribute_ID	Identification number assigned to the attribute in the dictionary.
data_type	Any one of the following data types: ■ **UnsignedInt** 32-bit unsigned value in big endian order (high byte first). ■ **Integer** 32-bit value in big endian order (high octet first). ■ **String** 0-253 octets ■ **Ipaddr** 4 octets in network octet order ■ **Binary** 0-254 octets ■ **Password** (n * 16) (>= 16) octets. This field is encrypted according to the User-Password AVP in RFC 2865. ■ **Short** 16-bit value ■ **Octet** 8-bit value ■ **ifid** IPv6 interface ID ■ **ipv6addr** IPv6 address ■ **date** UNIX timestamp in seconds (since January 1, 1970 GMT)

Loading the RADIUS Mediation Specification Data

RADIUS Gateway uses the RADIUS mediation specification data to determine which product and event type combination and network mapping applies to an incoming request from the RADIUS client.

To load the RADIUS mediation specification data:

1. Create a mediation specification file or open the sample RADIUS mediation specification file.

   A sample mediation specification file (*ECE_home*/**oceceserver/sample_data/config_data/specifications/ece_simple**) is available.

   Important: Create only one RADIUS mediation specification file to represent the mediation specification for RADIUS Gateway.

   ---

2. Load the pricing data from PDC into ECE.

   For every event definition, which contains charging operation types (for example, **Initiate**) loaded into ECE from PDC, ECE generates network mapping files.

   See the discussion about load pricing data from PDC in *BRM Elastic Charging Engine Implementation Guide*.

3. Add a row (in the table) for each new product to be rated that specifies the following information:

   - **Service-Identifier AVP**

     A unique identifier of the service. The Service-Identifier AVP value is sent by the RADIUS request. "null" is valid if the field is not expected to be present in the request.

   - **ProductType**

     The product type that you have defined for the event in its associated request specification.

   - **EventType**

     The event type that you have defined for the event in its associated request specification.

   - **Version**

     The version number of the request specification that you want to apply to the event.

   - **ValidFrom**

     A future date and time when you want RADIUS Gateway to recognize a newly deployed request specification.

     To have requests processed according to a new specification, you would enter:

     yyyy-*mm*-*dd***T***hh*:*mm*:*ss* [*timezone*]

     If *timezone* is not specified, it defaults to **UTC**.

   - **Network-Mapping-FileName**

     The name of the network mapping file generated for the product and event combination.

   See Example 4–1 for a sample entry in the RADIUS mediation specification file.

4. Open the *ECE_home*/**oceceserver/config/management/migration-configuration.xml** file.

5. Search the **configObjectsDataDirectory** parameter and copy the value. For example:

   ```
   configObjectsDataDirectory = ECE_home/oceceserver/sample_data/config_
   data
   ```

6. Save the mediation specification file to that same directory.
7. Load the file into the ECE server by running the following command:

   ```
   start configLoader
   ```

   The utility loads the RADIUS mediation specification data to the ECE cluster. The **configLoader** utility uses the location in the **configdata** parameter for loading the data. As mediation specification files have same names, so any existing RADIUS mediation specification data in the ECE cluster is overwritten.

Example 4–1 Sample RADIUS Mediation Specification Entry

```
RadiusMediationTable {
Service-Identifier| ProductType | EventType | Version |  ValidFrom |  Network-Mapping-FileName|
 "1" | "TelcoGprs" | "EventDelayedSessionTelcoGprs" | 2.0 | "2010-12-31T12:01:01 PST" |
"EventDelayedSessionTelcoGprs_TelcoGprs.xml" |
}
```

When you load the RADIUS mediation specification data into the ECE cluster, RADIUS Gateway re-creates its in-memory usage-request builder map and uses the mapping definitions to send requests to ECE.

About Mapping RADIUS Network Attributes to Event Attributes

To process requests from RADIUS clients, you map network attributes from RADIUS clients to the corresponding event attributes in ECE. You do this by editing the network mapping file. When you load the pricing data from PDC into ECE, ECE generates the network mapping file for each product and event combination. Some default network mappings are already pre-configured in the files generated by ECE. You can update the default values in these files.

RADIUS Gateway uses this mapping in ECE to process requests by dynamically mapping the values of the network attributes in the RADIUS request to the corresponding event attributes in ECE.

Mapping RADIUS Network Attributes to Event Attributes

If you add or remove an event attribute from the event definition in PDC, you have to add or remove the corresponding network attributes in ECE. You do this by editing the network mapping file in ECE.

Before you map the attributes, load the RADIUS mediation specification file. See "Loading the RADIUS Mediation Specification Data" for more information.

To map network attributes to event attributes:

1. Load the pricing data from PDC into ECE.

   Mapping files will be automatically generated when the pricing data is published from PDC to ECE.

   See the discussion about load pricing data from PDC in *BRM Elastic Charging Engine Implementation Guide*

For every event definition, which contains charging operation types (for example, **Initiate**) loaded into ECE from PDC, ECE generates the network mapping files. The network mapping files are stored in the directory specified by the **configObjectsDataDirectory** parameter in the *ECE_Home*/**oceceserver/config/management/migration-configuration.xml** file

A sample network mapping file is available in the (*ECE_home*/**oceceserver/sample_data/config_data/specifications/ece_end2end/network_mapping**) directory. You can use this as a reference for mapping the attributes.

2. Open a network mapping file in a text editor.
3. Ensure that the ORIGIN_NETWORK event attribute is added as a top-level attribute in the network mapping file.
4. Map the network attributes to the event attributes by doing the following:
   a. Search for the event attribute that you want to map to the network attribute.
   b. Add the following entry:

   ```
   <networkField>NetworkAttribute</networkField>
   ```

   where *NetworkAttribute* is the attribute of the requests received from RADIUS clients.

   For example:

   ```
   <attributeMapping type="RadiusMediationEntries">
        <attribute>
          <name>TERMINATE_CAUSE</name>
          <networkField>Acct-Terminate-Cause</networkField>
        </attribute>
      </attributeMapping>
   ```

5. Save and close the file.

   Important: Verify that the name of this network mapping file is specified in the RADIUS mediation specification file.

6. Load the network mapping data by doing one of the following:
   - If RADIUS Gateway is running, run the following command:

     ```
     start configLoader loadNetworkMapping
     ```

   - If RADIUS Gateway is *not* running, run the following commands:

     ```
     start customerUpdater
     start radiusGateway
     ```

   The network mapping data is loaded into the ECE cluster. Any existing network mapping data available for the product and event specification in the ECE cluster is overwritten. ECE is now in a usage-processing state, where it can accept requests from RADIUS Gateway.

When you load the network mapping into the ECE cluster, RADIUS Gateway re-creates its in-memory usage-request builder map and begins using the latest mapping definitions to send requests to ECE.

5 Monitoring and Managing ECE

This chapter describes how to monitor and manage Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE).

Overview of Monitoring and Managing Tasks

The managing of ECE nodes is centralized through the use of a *driver machine*. Any machine on which you have established password-less SSH and installed the Groovy Shell (**groovysh**) and set Groovy in its PATH can be used as the driver machine. Typically, the machine on which ECE is initially installed becomes the driver machine.

You can configure ECE on the driver machine and then push out the configuration from that one machine to all server machines in the cluster. The primary applications used for managing ECE are the Elastic Charging Controller (ECC) and Oracle Application Management Pack for Oracle Communications (you install Application Management Pack for Oracle Communications as a plug-in on an existing Oracle Enterprise Manager Cloud Control instance). You install these applications on the driver machine.

You can monitor ECE nodes (server processes) by using Oracle Application Management Pack for Oracle Communications, by using JConsole, or by using ECC. Each tool affords different degrees of operation and monitoring capabilities.

A system administrator who wants to maintain, update and tune the ECE system will use ECC.

A systems operator who wants to monitor the ECE system and perform basic operational tasks will use Oracle Application Management Pack for Oracle Communications.

- Oracle Application Management Pack for Oracle Communications monitors several things including:
  - The nodes that are running
  - The nodes that are stopped
  - The performance of charging server nodes

  Oracle Application Management Pack for Oracle Communications provides management capabilities for Oracle Communications Billing and Revenue Management (BRM) and other supported Oracle Communications applications. For detailed information about the management capabilities provided by Oracle Application Management Pack for Oracle Communications, see *Oracle Application Management Pack for Oracle Communications System Administrator's Guide*.
- JConsole shows:

  - The nodes that are running

  When one of the charging server nodes is JMX enabled, you can obtain statistics about all charging server nodes in the cluster by using JConsole.

  JConsole also enables you to change ECE configurations that are exposed through JMX.

- ECC lists
  - The nodes that are running
  - The nodes that are stopped

  You can add and remove a node from your topology by using ECC as well as perform various other management task. You use ECC, a command line application, for day-to-day administration, managing and operating of the ECE system.

  You can also use ECC to synchronize the ECE installation on the driver machine to different machines. You first provision the different machines as required for having ECE installed and then you run the ECC sync command to synchronize the installation onto the machines.

  See "Using the Elastic Charging Controller to Manage Nodes" for more information.

Configuring Alerts for ECE Charging Servers

You can configure a charging-server health threshold so that you are alerted when Elastic Charging Server node failures threaten the ability of your system to handle your customer base. See "Configuring Charging-Server Health Thresholds" for more information.

You can set alerts for ECE when you use Oracle Application Management Pack for Oracle Communications (a plug-in for Oracle Enterprise Manager Cloud Control) for monitoring ECE nodes. For detailed information about the management capabilities provided by Oracle Application Management Pack for Oracle Communications, see *Oracle Application Management Pack for Oracle Communications System Administrator's Guide*.

Configuring and Reading Log Files

ECE error log files provide detailed information about system problems. If you have a problem with an ECE process or node, look in its corresponding log file in the *ECE_Home*/**oceceserver/logs** directory. This directory contains the log where log files are generated for each node on each machine of your topology.

You can use JConsole for configuring log levels on each node (to manage the node's logging levels). When using the configuration service, select the **MBeans** tab and expand the navigation tree of the **ECE Logging** MBean.

See "Configuring Logging" for information about configuring logging parameters.

Monitoring Rated Event Formatter Health

To monitor the health status of the Rated Event Formatter processes, you look at the Rated Event Formatter log files. If you use Oracle Application Management Pack for Oracle Communications, you can use Oracle Enterprise Manager Cloud Control to check the Rated Event Formatter log files.

6 Managing Persisted Data in the Oracle NoSQL Database

This chapter describes maintenance tasks for managing Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) data stored in the Oracle NoSQL Database.

About Persisting Data in the Oracle NoSQL Database

Online and offline usage-request processing results in rated events. ECE persists rated event data in the Oracle NoSQL Database for the following purposes:

- Temporary persistence

  ECE persists rated event data to the Oracle NoSQL Database temporarily until it is sent to another system for long-term persistence.

  For example, rated event data is persisted in the Oracle NoSQL Database for a short time before it is sent to Oracle Communications Billing and Revenue Management (BRM) for long-term persistence so that network operators can use the stored data for auditing purposes.

  You can configure the amount of time to store the rated event data in the Oracle NoSQL Database and when to purge it from the Oracle NoSQL Database when it is no longer needed. See "Purging Rated Events from the Oracle NoSQL Database" for information about managing ECE data in the Oracle NoSQL database.

- Performance

  For maintaining optimal performance, ECE charging servers use temporary data in memory when processing online and offline usage requests. The rated event data that results from online and offline usage-request processing is held for a very short period in a dedicated Coherence cache before it is persisted in the Oracle NoSQL Database and expired from the Coherence cache.

- Storage of rated event data for the generation of Rated Event (RE) Loader files

  ECE Rated Event Formatter formats the rated event data from the Oracle NoSQL Database as BRM CDR files. The BRM Rated Event (RE) Loader loads the BRM CDR files into the BRM database to update customer account balances.

- Duplicate usage request detection upon a partial ECE cluster failure

  Oracle NoSQL Database detects and deletes duplicate rated event information if sent by the ECE charging servers. For example, when an ECE charging server node fails after having sent rated event data to the Oracle NoSQL Database, the node will resend the rated event data to the Oracle NoSQL Database upon startup.

Oracle NoSQL Database will override the existing data to prevent duplicate data entries.

Data redundancy of rated event data is achieved in both ECE charging server nodes as well as in the Oracle NoSQL Database. To handle an ECE charging server node failure, you can configure multiple nodes on which to create a backup copy of the rated event data stored on the failed node; this kind of data redundancy can also be configured within Oracle NoSQL Database.

See the Oracle Coherence documentation for general information about Coherence and availability.

See the Oracle NoSQL Database documentation for information about the Oracle NoSQL database and availability.

See "Configuring Charging-Server Health Thresholds" for information about configuring ECE for availability in the cluster.

Purging Rated Events from the Oracle NoSQL Database

Rated events are purged from the Oracle NoSQL Database by Rated Event Formatter. For information about purging rated events from Oracle NoSQL Database, see the following topics:

- About Purging Rated Events from the Oracle NoSQL Database
- Configuring How Rated Events Are Purged from the Oracle NoSQL Database

About Purging Rated Events from the Oracle NoSQL Database

After the Rated Event Formatter is started, it continuously purges rated events from Oracle NoSQL Database based on how you configure it. By default, Rated Event Formatter purges the rated events immediately after it has processed them (a rated event is written as an RE Loader record before it can be purged).

Depending on your business needs, you can purge rated events from the Oracle NoSQL Database immediately after they are processed by Rated Event Formatter or retain them for a period of time after they are processed. For example, you can retain them for an hour if legal requirements mandate that, or retain them for thirty minutes to give another program time to get data from the rated events before they are purged.

Configuring How Rated Events Are Purged from the Oracle NoSQL Database

To configure how rated events are purged from Oracle NoSQL database:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home***/oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.

2. Expand **charging.ratedEventFormatters.***Instance_Name*, where *Instance_Name* is the name of the instance you want to configure.
3. Expand **Attributes**.
4. Set the **retainDuration** attribute to the number of seconds to wait before purging rated events.
5. Use Elastic Charging Controller to stop and restart the Rated Event Formatter instances that you configured.

   For information about stopping and starting Rated Event Formatter instances, see the discussion about starting and stopping ECE in *BRM Elastic Charging Engine System Administrator's Guide*.

The Rated Event Formatter instance purges the rated events and writes log information to the Rated Event Formatter log file.

For more information about configuring Rated Event Formatter, see the discussion of implementing ECE with BRM in *BRM Elastic Charging Engine Implementation Guide*.

7 Using the Elastic Charging Controller to Manage Nodes

This chapter describes how to use Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) Elastic Charging Controller (ECC) for the operational management of ECE.

You can use Oracle Application Management Pack for Oracle Communications to manage and monitor ECE nodes and clusters. Oracle Application Management Pack for Oracle Communications provides management capabilities for Oracle Communications Billing and Revenue Management (BRM) and other supported Oracle Communications applications. For detailed information about the management capabilities provided by Oracle Application Management Pack for Oracle Communications, see *Oracle Application Management Pack for Oracle Communications System Administrator's Guide*.

About Elastic Charging Controller (ECC)

ECC is used for managing ECE software processes which have a role defined in the ECE topology file (*ECE_Home*/**oceceserver/config/eceTopology.conf**).

ECC can manage *nodes* in the Coherence cluster, such as charging server nodes and data-loading utility nodes, as well as ECE utilities such as the **pricingLoader** utility.

ECC reads the ECE topology file to know where ECE nodes and software are located in your hardware topology.

ECC is an extension of the Groovy Shell (**groovysh**) for ECE. ECC adds ECE-specific commands to the default Groovy Shell commands. For information about the default Groovy Shell commands and command syntax, see the Groovy Web site at:

`http://groovy-lang.org/groovysh.html#GroovyShell-Commands`

ECC is installed on the driver machine and can be used to push out configuration changes from the driver machine to all server machines across which the cluster is deployed. ECC uses its **sync** command for pushing out configuration changes across the cluster. See "About Using ECC on the Driver Machine" for information about the driver machine.

See *BRM Elastic Charging Engine Installation Guide* for information about setting up the topology file and setting up the driver machine and the server machines.

The following are some of the tasks you can perform by using ECC:

- Start all nodes defined in your topology.

For example, start charging server node instances, data-loading utility nodes, data updater nodes, simulator nodes, and so on.

- Stop all nodes defined in your topology.
- Start the charging server node instances.
- Stop the running charging server node instances.
- Start the simulator.
- Check the status of a running node.
- Add a node to your topology.
- Remove a node from your topology.

About Using ECC on the Driver Machine

The driver machine, the server that acts as the root of your ECE installation, is where you use ECC. You run ECC from the driver machine and use it to push the ECE installation out to the various server machines on the cluster. The driver machine contains your topology file, ECE properties file, JVM tuning file and so on. The driver machine contains the *ECE_Home*/**oceceserver/management/** configuration settings that apply to all servers currently running on the grid and will apply to those that will be added later.

Prerequisites for Using ECC

Before using ECC, perform the following prerequisite tasks:

1. Install ECE server software on the driver machine and specify the ECE root directory, ECE user name, and the IP address of the driver machine in the *ECE_Home*/**oceceserver/config/ece.properties** file.

   ECC is included in the ECE server software package and must be run on the server that is the driver machine.

   Ensure you have the correct values set in the **ece.properties** file before you use the **sync** command.

2. Verify that all the nodes in your cluster are defined in the *ECE_Home*/**oceceserver/config/eceTopology.conf** file.

   ECC reads the topology you define in this file to locate all the nodes it can manage.

   For example, after you define all charging-server node instances in the **eceTopology.conf** file, you can use ECC commands to start, stop, and initialize these nodes.

3. Verify that two-way password-less SSH is configured between the driver machine and all server machines in the cluster.

4. Verify that the proper permissions are granted in ECC.

   Access to ECE files is controlled by creating user accounts and granting specific permissions in ECC. After you have created user groups and set permissions, users can log in through ECC and manage ECE files. For information, see "Setting Up User Accounts and User Groups".

Using ECC

To use ECC:

1. Log on to the driver machine.
2. Change directory to the *ECE_Home*/**oceceserver/bin** directory.
3. Enter the following command:

   ```
   ./ecc
   ```

4. From the command prompt, run the command you would like to run.

   See "Running ECC Commands".

   For information about each command you can use ECC Help. See "Getting Help Using ECC".

ECC Command Output

ECC command output provides the following information:

- The success or failure of the command (status of true or false)
- The tasks the command executed
- For a failed task, the error message explaining the failure

All ECC commands return a number of events depending on the number of tasks the command performs. ECC command output includes the events and their associated attributes, along with the overall command status.

Depending on individual command results, the following fields are populated and printed on the ECC command output for an event returned by a command:

- **Request**: A string only printed in an error case.
- **Response**: A string returned if the command or task returns a value or includes a confirmation message. For example, the **encrypt** command returns the encrypted password.

  If there is an error performing a task, an error message is stored in the **Response** field.
- **Status**: A boolean indicating that the task succeeded or failed.
- **Node**: A string of the node name.
- **Details**: A map with further details like the node process ID.

Here is an example of the ECC command output for the **start** command when starting all charging server nodes for an ECE topology that contains only two charging server nodes (with node names **ecs1** and **ecs2**):

```
ecc:000> start server
-- No JMX Enabled Nodes are available.
-- Starting one JMX-Enabled Node.
-- Node 'ecs1' started with PID 1331
-- ecs1 (JMX-Enabled) is running and has joined the cluster.
-- Node 'ecs2' started with PID 2081
-- ecs2 is running and has joined the cluster.
===> status: true
[
Status: true
Node: ecs1
details: [pid:1331, state:running]
----------
Status: true
Node: ecs2
```

```
details: [pid:2081, state:running]
----------
]
```

Here is an example of the ECC command output for the **encrypt** command when encrypting the password **aStrongPassword**.

```
ecc:000> encrypt aStrongPassword storepassword
Encrypted value for "aStrongPassword" is

===> status: true
[
Status: true
response: acMTrZo4mXEJa7yWUKClJQ==
]
```

Running ECC Commands

You run ECC commands to manage nodes (ECE processes) in the cluster. This section provides information about each command. For information about running each command you can also use the ECC **help** command. See "Getting Help Using ECC".

For a list of all ECC commands, see Appendix B, "ECC Commands".

Running the start Command

The following is the syntax and examples for the **start** command:

Syntax:

start [*role*] -or- [*nodeName*]

where:

- *role* is the name of the role specified for that node in the **role** column of the ECE topology file; this indicates the type of node such as whether it is a charging server node (**server**) or a simulator node (**simulator**).
- *nodeName* is the name of the node specified for that node in the **node-name** column of the ECE topology file.

For example, to start nodes by role, you can do the following:

- Start all nodes of type **server**:

  ```
  start server
  ```

- Start all nodes of type simulator:

  ```
  start simulator
  ```

For example, to start nodes by name, you can do the following:

- Start a charging server named ecs1:

  ```
  start ecs1
  ```

If you run only the **start** command without anything passed; for example:

```
start
```

this is equivalent to **start server**.

Running the stop Command

Caution: Stopping charging server nodes removes all data from memory. For example, if you run **stop server**, you will lose all data in Coherence caches. See "Restoring the ECE System" for more information.

The following is the syntax and examples for the **stop** command:

Syntax:

stop [*role*] -or- [*node1* [*node2*] [...]]

For example, to stop all nodes that have the role server:

```
stop server
```

For example, to stop nodes by name:

```
stop ecs1
```

To get help on the **stop** command:

```
help stop
```

Running the addNode Command

The following is the syntax for the **addNode** command:

Syntax:

addNode [*node name*] [*role*] [*host name*] [*host ip*] [*JMX port*] [*start CohMgt*] [*JVM Tuning File*]

Run the command **help addNode** for information about the mandatory and optional fields for this command.

When you run the **addNode** command, you add nodes to the cluster.

The **addNode** command runs the **sync** command which installs ECE on the host machine of the node.

addNode adds an entry for the node in the topology file.

addNode does not start the node after it adds the node to the cluster. You must use the **start** command to start the node.

Running the removeNode Command

When you run the **removeNode** command, you remove nodes from the cluster.

removeNode removes the entry for the node from the topology file.

You cannot use the **removeNode** command when nodes are running. You are required to stop a node before you remove it by using the **removeNode** command. The **removeNode** command reports a warning if you try to run it for a node in a running state.

Running the infoCollector Command

You use the **infoCollector** command to collect log files so that you can send them to Oracle technical support when troubleshooting problems with ECE. The command collects ECE configuration files from various directories and puts them in a central location. The command provides options to TAR the files and also enables you to include your custom files (apart from ECE configuration files).

See "Collecting Log Files for Sending to Oracle Support" for more information.

For information about **infoCollector** command syntax, run the following ECC command:

```
help infoCollector
```

Running the rollingUpgrade Command

You use the **rollingUpgrade** command to deploy updates onto ECE software while maintaining operation of the ECE syste"Performing a Rolling Upgrade"m (without stopping ECE). For more information, see .

The rollingUpgrade command can be used for the following purposes:

- Deploying JVM tuning parameter updates onto a running ECE system. See "Deploying JVM Tuning Parameter Updates onto a Running System" for more information.
- Deploying JMS configuration setting updates onto a running ECE system. See the discussion of configuring notifications for charging in *BRM Elastic Charging Engine Implementation Guide* for more information.

The following is the syntax and examples for the **rollingUpgrade** command:

Syntax:

rollingUpgrade [*role*]

For example, to upgrade all running nodes in the topology except the simulator roles, run it with no arguments:

```
rollingUpgrade
```

For example, to upgrade all running charging server nodes (nodes that have the role **server**):

```
rollingUpgrade server
```

The order in which the nodes are restarted adheres to the order in which the nodes are listed in the *ECE_Home*/**oceceserver/config/eceTopology.conf** file.

You can choose to upgrade nodes (bring them down, upgrade them, and join them back to the cluster) by node *role*. For example, to first upgrade all the nodes of role **ratedEventFormatter**, followed by all the nodes of role **server**, followed by all the nodes of role **updater**, and then followed lastly by all the nodes of role **diametergateway**, you would enter the following commands:

```
rollingUpgrade ratedEventFormatter
rollingUpgrade server
rollingUpgrade updater
rollingUpgrade diametergateway
```

You require a full cluster restart (when a rolling upgrade cannot be performed) under the following conditions:

- Non-backward compatible upgrades
- Adding or removing a property

Request specification data once loaded into the cluster are not reversed with a roll back.

The **rollingUpgrade** command can take as input the user name and password for secured JMX environments:

```
rollingUpgrade username=username password=Password
```

Getting Help Using ECC

Get help using ECC by:

- Using the help Command
- Using Completion Support

See Appendix B, "ECC Commands" for a list of all ECC commands.

Using the help Command

Use the ECC **help** command for information about using all commands:

```
help
```

To get help on a particular command:

```
help start
usage: start [role]

start all the nodes of type role.
```

ECC provides a description about the command.

Using Completion Support

ECC offers completion support for commands, command options, and command-option parameters. For example:

- Type **st**, then press the **space bar** and **Tab key** for command completion:

  ```
  ecc:000>st

  start stop]
  ```

  ECE lists all command that begin with **st**.

- Type **start**, then press the **space bar** and **Tab key** for command-option completion:

  ```
  ecc:000>start

  gateway  loader  server  simulator]
  ```

  ECE lists all command options available for the **start** command. In the preceding example, ECE lists all nodes you can start.

- Type **start server**, and then press the **space bar** and **Tab key** for command-option parameter completion:

```
ecc:000> start server

ebugPort=portNumber  extraClassPath=path  extraJavaProperties=pros...]
```

Performing a Rolling Upgrade

Important: Rolling upgrades may not be supported for upgrading to specific versions of ECE. Refer to *BRM Elastic Charging Engine Release Notes* for information about rolling upgrade support in ECE releases.

Rolling upgrades can be used for the following:

- Deploying JVM tuning parameter updates onto a running ECE system. See "Deploying JVM Tuning Parameter Updates onto a Running System" for more information.
- Deploying JMS configuration setting updates onto a running ECE system. See the discussion of configuring notifications for charging in *BRM Elastic Charging Engine Implementation Guide* for more information.
- Upgrading from an old version of ECE to a new version of ECE. See the chapters about upgrading ECE in *BRM Elastic Charging Engine Installation Guide* for information about performing a rolling upgrade when upgrading to a new version of ECE.

You perform a rolling upgrade to upgrade ECE software while maintaining operation of the ECE system (without stopping ECE). You would perform a rolling upgrade, for example, when you need to install a patch recommended by Oracle.

The **rollingUpgrade** command performs an upgrade of all nodes of a particular role while ensuring high availability of the system. For example, if a patch fix only applies to the functionality of charging servers, you can apply the command only to nodes with the role **server**.

One by one, the command will stop the older versions of nodes and start current versions of these nodes.

Prior to moving to the next node to upgrade, the command ensures that the last upgraded node has correctly started and joined the cluster. If the last upgraded node does not correctly start and join, the command interrupts the upgrade process.

For rolling back the upgrade, or downgrade to the older version, run this command from the old installation location.

If no role is provided, all running nodes except simulators are upgraded.

To perform a rolling upgrade:

Note: Before performing the rolling upgrade, a new installation of ECE must be performed in a different directory. After launching the Elastic Charging Controller using the new installation, the rollingUpgrade is called to upgrade the system to the new version.

---

1. Perform the new installation of ECE and apply the required patch updates to the installation.

   For example, install a patch recommended by Oracle onto the new installation.

2. Verify that the topology you specify in your new installation (in the ECE topology file) is the same as the topology specified in your old installation.
3. Go to the new ECE installation's **config** directory.

   ```
   cd ECE_Home_new_installation/oceceserver/config
   ```

4. Open the **ece.properties** file and set the **numberOfPauseSecondsForRollingUpgrade** parameter

   The default is **30**.

   Set the amount of time in seconds you want ECC to pause between bringing down and upgrading each server node that is running on the old installation.

   Setting a time lapse allows for rebalancing of Coherence partitions (for example, for a server node being brought down to be upgraded, it allows time for its backup node to become a primary node). The time lapse also helps ensure that upgraded nodes have come up and joined the cluster individually before the next node is brought down for upgrade.
5. Go to the **bin** directory of the new ECE installation.

   ```
   cd  ECE_Home_new_installation/oceceserver/bin
   ```

6. Start ECC.

   ```
   ./ecc
   ```

7. Run the following command to start the rolling upgrade from the new installation while ECE is still operating on the old installation:

   ```
   rollingUpgrade
   ```

   One by one, each node on the old location is brought down, upgraded, and joined back to the cluster.

   When you run the **rollingUpgrade** command with no parameters specified, all running nodes are upgraded (charging server nodes, data-loading utility nodes, data updater nodes, and so on) except for simulator nodes (nodes that have the role **simulator**).

   The order in which the nodes are restarted adheres to the order in which the nodes are listed in the *ECE_Home*/**oceceserver/config/eceTopology.conf** file.

   You can choose to upgrade nodes (bring them down, upgrade them, and join them back to the cluster) by node *role*. For example, to first upgrade all the nodes of role **ratedEventFormatter**, followed by all the nodes of role **server**, followed by all the nodes of role **updater**, and then followed lastly by all the nodes of role **diametergateway**, you would enter the following commands:

   ```
   rollingUpgrade ratedEventFormatter
   rollingUpgrade server
   rollingUpgrade updater
   rollingUpgrade diametergateway
   ```

After the upgrade has completed, the new version of ECE is used and you can decide what you would like to do with the old directory installation.

Roll-Back

To roll back an upgrade:

1. Go to the old ECE installation's bin directory.

   ```
   cd ECE_Home_old_version/oceceserver/bin
   ```

2. Start ECC.

   ```
   ./ecc
   ```

3. Run the following command to start rolling back the upgrade:

   ```
   ecc:000> rollingUpgrade
   ```

   One by one, each server node is rolled back.

8 Backing Up and Restoring ECE

This chapter describes the tasks that you perform to back up and restore Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE).

About Backing Up and Restoring ECE

After installing or upgrading an ECE system, perform one of the following backups as a safety measure:

- Standard configuration backup. See "About Standard Configuration Backup".
- Complete system backup. See "About Complete System Backup".

Repeat the backup process whenever you make any changes in the data or configuration files. If you do not back up the ECE system regularly, you need to reinstall and reconfigure ECE if the system is corrupted due to operational or system errors. Reinstalling and reconfiguring eliminates any chance of recovering and reprocessing data processed by the ECE system at the time of the error.

About Standard Configuration Backup

The standard configuration backup is a backup of the ECE configuration files and ECE installation area (the ECE installation directory and its content: *ECE_home*). You can perform this backup as soon as you install and configure ECE. In particular, make sure you back up all customized files. For example, you need to perform this procedure if you customized any settings in the *ECE_home*/**oceceserver/config/management/charging-settings.xml** file.

Repeat the backup process whenever you customize or update ECE configuration files. For instructions on backing up the standard configuration, see "Backing Up Standard Configuration".

When you complete a standard configuration backup, you can use the backup configuration directory and installation area to restore your system when needed. For instructions on restoring the standard configuration, see "Restoring a Standard System Configuration".

Important: Store this backup in a safe location. The data in these files are necessary if you encounter any operational or system issues.

If you require ECE technical support, email a copy of your backup configuration directory to your Oracle Global Support representative. If you want to send a copy of your configuration directory, use the tar command to create an archived version of that directory.

About Complete System Backup

The complete system backup is a backup of the complete ECE system. You can perform the complete system backup by creating a complete offline backup of the following:

- ECE configuration files and ECE installation area. See "Backing Up Standard Configuration".
- Oracle NoSQL database configuration and Oracle NoSQL database installation area. See "About Backing Up Oracle NoSQL Installation and Configuration".

Important: If you have multiple ECE charging server nodes configured in your system, perform the backup on the driver machine. Make sure that you repeat the backup process regularly.

Store this backup in a safe location. The data in these files are necessary if you encounter any operational or system issues.

Typically, ECE files generated at run time (such as Rated Event (RE) Loader files (rated event data), RE loader control files, log files, and Coherence-related files) are automatically recovered when you restore your ECE system. For example, if the RE loader control files are not available in your restored ECE system, Rated Event Formatter automatically generates the RE loader control files based on the event specification data loaded into ECE from PDC. Therefore, you need not create an offline backup of the ECE files generated at run time.

However, if you want to avoid the risk of losing any data, you can create a backup of RE loader files. See the discussion about backing up RE Loader files in *BRM Configuring Pipeline Rating and Discounting*.

For instructions on restoring your complete ECE system backup, see "Restoring a Complete System Backup".

About Backing Up Oracle NoSQL Installation and Configuration

ECE uses Oracle NoSQL Database to temporarily store rated event information. Specifically, ECE uses the multiple-node Oracle NoSQL data store configuration in production environment for storing the ECE data. Make sure you create a backup of Oracle NoSQL installation area (the Oracle NoSQL installation directory and its content) and the multiple-node Oracle NoSQL data store configuration. You can perform this backup as soon as you install and configure the Oracle NoSQL database. Repeat the backup process whenever you customize or update the Oracle NoSQL data store configuration.

For instructions on backing up the Oracle NoSQL Database installation, see "Backing Up the Oracle NoSQL Database Installation".

You can also make a complete offline backup of Oracle NoSQL Database using the appropriate backup tools for your database version and ensure that the backup is completely valid and usable. The backup must contain both the database definition and all the database contents. See the Oracle NoSQL Database documentation for more information on performing full database backups.

You can make a complete offline backup of the multiple-node Oracle NoSQL data store configuration by taking snapshots of the nodes. See the discussion about backing up the Oracle NoSQL data store in the Oracle NoSQL database documentation for instructions on backing up an Oracle NoSQL Database data store.

Backing Up Standard Configuration

To back up standard configuration:

1. On the driver machine, go to the *ECE_home* directory.
2. Copy the content of the *ECE_home* directory to a new directory:

   ```
   cp -R ECE_home NewName
   ```

   where *NewName* is the name for the new directory.

   > **Note:** You can remove the ECE log files and Coherence-related files from the backup directory.

3. Create an archive of the entire directory:

   ```
   tar cvf NewName.tar.gz NewName
   ```

4. Store the backup copy in a location outside of the ECE system:

   ```
   mv NewName.tar.gz New_Directory
   ```

   where *New_Directory* is the new location that is outside of the ECE system.

   A compressed TAR file, of all copied files, is created with the extension **tar** in the new location specified (for example, ece_backup/ece_home.tar.gz).

Backing Up the Oracle NoSQL Database Installation

To back up a Oracle NoSQL database installation:

1. On the machine in which the Oracle NoSQL database is installed, go to the *Oracle_NoSQL_DB_Home* directory.
2. Copy the content of the *Oracle_NoSQL_DB_Home* directory to a new directory:

   ```
   cp -R Oracle_NoSQL_DB_Home NewName
   ```

   where *NewName* is the name for the new directory.
3. Create an archive of the entire directory:

   ```
   tar cvf NewName.tar.gz NewName
   ```

4. Store the backup copy in a location outside of the ECE system:

   ```
   mv NewName.tar.gz New_Directory
   ```

   where *New_Directory* is the new location that is outside of the ECE system.

   A compressed TAR file, of all copied files, is created with the extension **tar** in the new location specified (for example, Oracle_noSQL_db_backup/noSQL_db_bkp.tar.gz).

Restoring a Standard System Configuration

This section outlines how to restore an ECE system backup.

To restore an ECE system backup:

1. In PDC, publish all the PDC pricing data (the metadata, setup, pricing, and profile data) from the PDC database to ECE by running the following command:

   ```
   ImportExportPricing -publish -metadata -config -pricing -profile -target [ece]
   ```

2. Delete or rename the damaged *ECE_home* directory:
   - To delete, run the following command:

     ```
     rm -R ECE_home
     ```

   - To rename, run the following command:

     ```
     mv ECE_home New_Name
     ```

3. Retrieve the backup tar file of the *ECE_home* directory.
4. Extract from the tar file the backup copy of the directory:

   ```
   tar xvf ECE_home.tar.gz
   ```

   The command recreates the copy of the *ECE_home* directory.
5. On the driver machine, go to the *ECE_home*/**bin** directory.
6. Start ECC:

   ```
   ./ecc
   ```

7. Enable real-time synchronization of BRM and ECE customer data updates. See the discussion about configuring ECE for synchronizing BRM and ECE customer data in real time in *BRM Elastic Charging Engine Implementation Guide* for more information.
8. Start ECE processes and gateways in the following order:

   Important: Depending on your installation, you can start Diameter Gateway, RADIUS Gateway, both Diameter Gateway and RADIUS Gateway, or none.

   ```
   start server
   start configLoader
   start pricingUpdater
   start customerUpdater
   start emGateway
   start brmGateway
   start ratedEventFormatter
   start diameterGateway
   start radiusGateway
   ```

   All data is now back in the ECE data grid.

   Real-time-data updates, which had been temporarily disrupted due to the shutdown, are processed upon restart.

Restoring a Complete System Backup

To restore a complete ECE system backup:

1. In PDC, publish all the PDC pricing data (the metadata, setup, pricing, and profile data) from the PDC database to ECE by running the following command:

```
ImportExportPricing -publish -metadata -config -pricing -profile -target [ece]
```

2. Delete or rename all the damaged directories; for example, *ECE_home* and *Oracle_NoSQL_DB_Home*:

   - To delete, run the following command:

     ```
     rm -R Directory_Name
     ```

   - To rename, run the following command:

     ```
     mv Directory_Name New_Name
     ```

3. Retrieve all the backup tar files. For example, ece_home.tar.gz and noSQL_db_bkp.tar.gz.
4. Extract from the tar files the backup copy of the directories:

   ```
   tar xvf Directory_Name.tar.gz
   ```

   The command recreates the directories in your restored installation directory.
5. Repeat steps 2 to 4 to restore all the damaged directories.
6. On the driver machine, go to the *ECE_home*/**bin** directory.
7. Start ECC:

   ```
   ./ecc
   ```

8. Enable real-time synchronization of BRM and ECE customer data updates. See the discussion about configuring ECE for synchronizing BRM and ECE customer data in real time in *BRM Elastic Charging Engine Implementation Guide* for more information.
9. Start ECE processes and gateways in the following order:

   Important: Depending on your installation, you can start Diameter Gateway, RADIUS Gateway, both Diameter Gateway and RADIUS Gateway, or none.

   ```
   start server
   start configLoader
   start pricingUpdater
   start customerUpdater
   start emGateway
   start brmGateway
   start ratedEventFormatter
   start diameterGateway
   start radiusGateway
   ```

   All data is now back in the ECE data grid.

   Real-time-data updates, which had been temporarily disrupted due to the shutdown, are processed upon restart.

9 Configuring ECE for Disaster Recovery

This chapter provides an overview of the Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) disaster recovery architecture. In addition, it describes how to configure ECE for disaster recovery and how the ECE system handles failover.

For information about the standard ECE system architecture, see the discussion about ECE system architecture in *BRM Elastic Charging Engine Concepts*.

About Disaster Recovery

Disaster recovery is a backup and recovery process that provides continuity in realtime processing of prepaid services in case of system failure. Implementing disaster recovery involves configuring multiple ECE systems that run at geographically remote locations. Typically, you configure a production system and one or more remote backup systems that take over when the production system fails. However, you can also configure duplicate remote production systems.

An ECE backup system at a remote location is called a *backup site* and an active ECE production system is called a *production site*.

When configuring ECE for disaster recovery, consider the following objectives:

- **Recovery time objective (RTO).** Specifies the time within which the services must be resumed after a system failure.
- **Recovery point objective (RPO).** Specifies the time within which the ECE data must be recovered from the production site before the quantity of data lost during the recovery period exceeds the maximum allowable threshold.

ECE supports the following types of disaster recovery systems:

- **Active-cold standby.** A disaster recovery system that consists of an active production site and one or more idle backup sites. This system requires starting the backup site manually when the production site goes down. This might cause a delay in bringing up the backup site to full operational capability. As a result, RTO and RPO are relatively high in an active-cold standby system.
- **Active-hot standby.** A disaster recovery system that consists of an active production site and one or more active backup sites. The ECE data is asynchronously replicated from the production site to the backup sites. When the production site goes down, the ECE requests are diverted from the production site to the backup sites. The RTO and RPO for an active-hot standby system are less than the active-cold standby system.
- **Segmented active-active.** A disaster recovery system that consists of two or more active production sites at remote locations (one primary production site and one or

more remote production sites), which concurrently processes ECE requests for a different set of customers. The pricing, customer, and ECE cache data are asynchronously replicated from the primary production site to remote production sites. When one of the production sites goes down, the requests from that site are diverted to the other sites. The RTO and RPO for a segmented active-active system are less than active-cold standby and active-hot standby systems. ECE requests are routed across the production sites based on your load balancing configuration. See "About Load Balancing in a Segmented Active-Active System" for more information.

Overview of Disaster Recovery Architecture

A disaster recovery system contains a primary production site and one or more backup sites or remote production sites.

In an active-cold standby system, each site contains Oracle Communications Billing and Revenue Management (BRM), Pricing Design Center (PDC), and ECE with the following core components:

- ECE Charging Server
- Customer Updater
- Pricing Updater
- BRM Gateway
- External Manager (EM) Gateway
- Rated Event Formatter
- Rated Event Publisher
- Diameter Gateway
- RADIUS Gateway

In an active-hot standby system or a segmented active-active system, each site contains BRM, PDC, and ECE. However, Customer Updater, Pricing Updater, and EM Gateway are configured only in the primary production site.

For more information about these components, see the discussion about ECE system architecture in *BRM Elastic Charging Engine Concepts*.

Figure 9–1 shows a basic architecture of an ECE disaster recovery system.

Figure 9–1 Basic Architecture of Disaster Recovery System

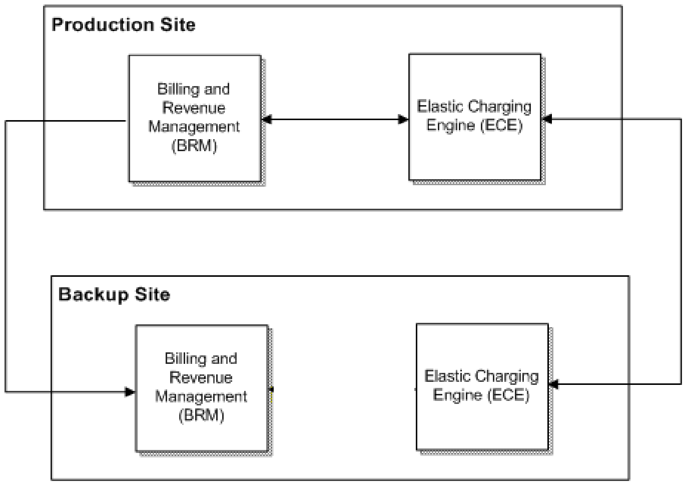

Production Site
Billing and Revenue Management (BRM)
Elastic Charging Engine (ECE)
Backup Site
Billing and Revenue Management (BRM)
Elastic Charging Engine (ECE)

Figure 9–1 contains a production site and a backup site. Table 9–1 describes the status of each site depending on the type of disaster recovery system.

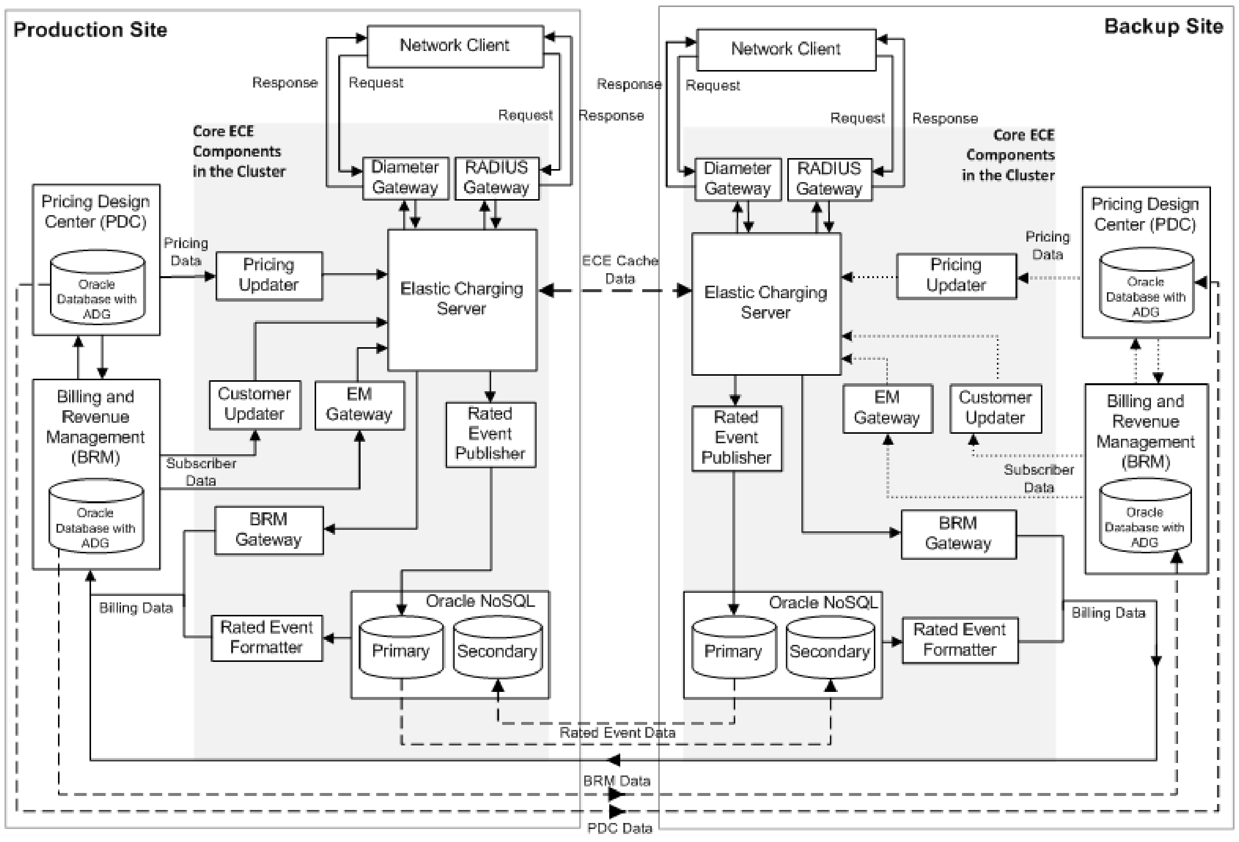
Table 9–1 Status of Disaster Recovery Systems

Disaster Recovery System	Production Site	Backup Site
Active-cold standby	Active	Idle standby
Active-hot standby	Active	Active standby
Segmented active-active	Active	Active

Figure 9–2 shows the architecture of an active-hot standby system. The solid line depicts the data flow, the dashed line depicts the data replication flow, and the dotted line depicts the components that are available but are not configured to load data.

Note: For simplicity, Figure 9–2 shows the architecture of an active-hot standby system with one production site and one active backup site. In general, an active-hot standby system can contain more than one active backup site to allow failover.

Figure 9–2 Architecture of an Active-Hot Standby Disaster Recovery System



Some of the key aspects of the architecture shown in Figure 9–2 are:

- The architecture has an active production site and an active backup site.
- ECE components are deployed and are active in both sites, but only the production site is processing requests.
- PDC and BRM databases are available in both sites, but they are active only in the production site. The PDC and BRM data are replicated from the PDC and BRM databases in the production site to the PDC and BRM databases in the backup site asynchronously using Oracle Active Data Guard.
- The ECE data in the Oracle NoSQL database data store is replicated from the production site to the backup site asynchronously using primary and secondary Oracle NoSQL database data store nodes.
- The ECE cache data is replicated from the production site to the backup site asynchronously using the Oracle Coherence federated caching feature.
- ECE requests are routed only to the production site. When the production site fails, the backup site takes over the role of the production site, and the ECE requests are diverted to the backup site.

About Replicating Data in a Disaster Recovery System

You replicate the following data from the production site to the backup or remote production sites:

- **Data in the BRM and PDC databases.** You use Oracle Active Data Guard to replicate data from the BRM and PDC databases in the production site to the BRM and PDC databases in the backup or remote production sites asynchronously. When the BRM and PDC databases in the production site fail, manual or automatic failover is performed to switch the BRM and PDC databases in the backup or remote production sites into the primary role based on your Oracle Active Data Guard configuration. For more information about configuring Oracle Active Data Guard and performing a manual or automatic failover, see the discussion about data guards in the Oracle Database documentation.

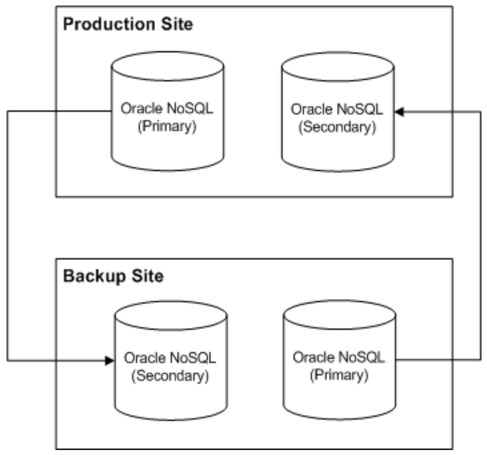
- **Data in the ECE cache.** In an active-hot standby system or a segmented active-active system, you use the Oracle Coherence federated caching feature to synchronize the ECE data grid and replicate the ECE cache data between participant sites (primary production site and backup sites or remote production sites) asynchronously. For more information about federated caching, see the Oracle Coherence documentation. You use the **gridSync** utility to start the federation service to replicate data to the participant sites asynchronously and also replicate all the existing ECE cache data to the participant sites. See "Replicating ECE Cache Data" for more information.
- **Data in the Oracle NoSQL database data store.** In an active-hot standby system or a segmented active-active system, you use the primary and secondary storage nodes of the Oracle NoSQL database data store to replicate the rated event data between participant sites. You configure the primary and secondary nodes in all the participant sites to replicate the data. See "About Configuring Oracle NoSQL Database Data Store Nodes" for more information.
- **Call detail records (CDRs) generated by Rated Event Formatter.** You use the Secure Shell (SSH) File Transfer Protocol (SFTP) utility to replicate CDR files generated by Rated Event Formatter between participant sites.

About Configuring Oracle NoSQL Database Data Store Nodes

In an active-hot standby system or a segmented active-active system, you configure primary and secondary storage nodes in the Oracle NoSQL database data store for each participant site to store and replicate the rated event data. The secondary storage nodes in one site act as a backup for the primary storage nodes of the other site. For example, when rated events are stored in the primary storage nodes of the production site, the information is replicated in the secondary storage nodes of the backup site. Similarly, the information that is stored in the primary storage nodes of the backup site is replicated in the secondary storage nodes of the production site.

Figure 9–3 shows the primary and secondary Oracle NoSQL database storage nodes configured in a disaster recovery system.

Figure 9–3 Primary and Secondary Oracle NoSQL Database Storage Nodes



Oracle recommends that you configure a minimum of three storage nodes in each site: two storage nodes to store rated events and one storage node to allow for failover. You can add additional storage nodes as needed to handle the expected throughput for your system. For information about adding and configuring storage nodes in the Oracle NoSQL database data store, see the Oracle NoSQL Database documentation.

About Load Balancing in a Segmented Active-Active System

In a segmented active-active system, ECE requests are routed across the production sites based on your load balancing configuration. To ensure proper load balancing on your system, you can use a combination of global and local load balancers. The local load balancer routes the connection requests across the full range of available Diameter Gateway and RADIUS Gateway nodes. The global load balancer routes the connection requests to the Diameter Gateway and RADIUS Gateway nodes in only one site unless it detects that site is busy or if the local load balancer signals that it cannot reach ECE. You can set up your own load balancing configuration based on your requirements.

Configuring a Disaster Recovery System

This section describes how to configure a disaster recovery system.

Configuring an Active-Cold Standby System

To configure an active-cold standby system:

1. In the production site and backup sites, do the following:
   - Install ECE and other components required for the ECE integrated system.

     See the discussion about installing all ECE components in *BRM Elastic Charging Engine Installation Guide* for instructions.
   - Ensure that any customizations to configuration files and extension implementation files in your production site are reapplied to the corresponding default files in the backup sites.
2. In the production site, start ECE. See "About Starting and Stopping ECE" for more information.

Configuring an Active-Hot Standby System or a Segmented Active-Active System

To configure an active-hot standby system or a segmented active-active system:

1. In the primary production site and backup or remote production sites, install ECE and other components required for the ECE integrated system.

   See the discussion about installing all ECE components in *BRM Elastic Charging Engine Installation Guide* for instructions.

2. In the primary production site, do the following:

   a. Configure primary and secondary Oracle NoSQL database data store nodes. See "About Configuring Oracle NoSQL Database Data Store Nodes".

   b. On the machine on which the Oracle WebLogic server is installed, verify that the Java Message Service (JMS) queues have been created for loading pricing data and for sending event notification and that JMS credentials have been configured correctly.

      See the discussion about creating JMS queues and configuring credentials in *BRM Elastic Charging Engine Installation Guide* for more information.

   c. Ensure that the connection details for all the instances of the following ECE core components are available in the *ECE_home*/**oceceserver/config/management/charging-settings.xml** file, where *ECE_home* is the directory in which ECE is installed:

      Customer Updater

      BRM Gateway

      EM Gateway

      Rated Event Formatter

      Rated Event Publisher

      Diameter Gateway

      RADIUS Gateway

      Important: If you are configuring a segmented active-active system, ensure the following:

      - The name of Diameter Gateway, RADIUS Gateway, Rated Event Formatter, and Rated Event Publisher for each site is unique.
      - A minimum of two instances of Rated Event Formatter are configured to allow for failover.

      For information on the configuration parameters for Customer Updater, Rated Event Formatter, Rated Event Publisher, BRM Gateway, and EM Gateway, see the discussion about implementing ECE with BRM in *BRM Elastic Charging Engine Implementation Guide*.

      For information on the configuration parameters for Diameter Gateway, see Table 4–4, " Diameter Gateway Node Configuration Parameters".

      For information on the configuration parameters for RADIUS Gateway, see Table 4–5, " RADIUS Gateway Node Base Configuration Attributes" and Table 4–6, " RADIUS Gateway Node Instance Configuration Attributes".

d. Verify that the JMS queues' details for Pricing Updater are specified in the **PdcEceQueue** and **PDCResultQueue** sections of the *ECE_home*/**config/management/migration-configuration.xml** file.

   See the discussion about configuring the Pricing Updater in *BRM Elastic Charging Engine Implementation Guide* for more information.

e. Add all the participant sites' details in the federation-config section of the ECE Coherence override file (for example, *ECE_home*/**oceceserver/config/charging-coherence-override-prod.xml**).

   To confirm which ECE Coherence override file is used, see the **tangosol.coherence.override** value in the *ECE_home*/**oceceserver/config/ece.properties** file.

   Table 9–2 provides the federation configuration parameter descriptions and default values.

Table 9–2 Federation Configuration Parameters

Name	Description
name	The name of the participant site. **Note:** The name of the participant site must match the name of the cluster in the participant site.
address	The IP address of the participant site.
port	The port number assigned to the Coherence clusterport of the participant site.
initial-action	Specifies whether the federation service has to be started for replicating data to the participant sites. Valid values are: ▪ **start.** Specifies that the federation service has to be started and the data must be automatically replicated to the participant sites. ▪ **stop.** Specifies that the federation service has to be stopped and the data must not be automatically replicated to the participant sites. **Note:** Ensure that this parameter is set to **stop** for all the participant sites except for the current site. For example, if you are adding the backup or remote production sites details in the primary production site, this parameter must be set to **stop** for all the backup or remote production sites.

f. Start ECE. See "About Starting and Stopping ECE" for more information.

g. Start the following ECE processes and gateways:

```
start emGateway
start ratedEventFormatter
start diameterGateway
start radiusGateway
```

   ECE in the primary production site is now in a usage-processing state to process requests sent from the network.

3. In the backup or remote production sites, do the following:

a. Configure primary and secondary Oracle NoSQL database data store nodes. See "About Configuring Oracle NoSQL Database Data Store Nodes" for more information.

b. On the machine on which the Oracle WebLogic server is installed, verify that the JMS queues have been created for event notification and the JMS credentials have been configured correctly. See the discussion about creating JMS queues and configuring credentials in *BRM Elastic Charging Engine Installation Guide* for more information.

c. Set the following parameter in the *ECE_home*/**oceceserver/config/ece.properties** file to **false**:

```
loadConfigSettings = false
```

The application-configuration data is not loaded into memory when you start the charging server nodes.

d. Add all the participant sites details in the federation-config section of the ECE Coherence override file (for example, *ECE_home*/**oceceserver/config/charging-coherence-override-prod.xml**).

To confirm which ECE Coherence override file is used, see the **tangosol.coherence.override** value in the *ECE_Home*/**oceceserver/config/ece.properties** file. Table 9–2 provides the federation configuration parameter descriptions and default values.

e. Go to the *ECE_home*/**oceceserver/bin** directory.

f. Start Elastic Charging Controller (ECC):

```
./ecc
```

g. Start the charging server nodes:

```
start server
```

4. In the primary production site, run the following commands:

```
gridSync start
gridSync replicate
```

The federation service is started and all the existing data is replicated to the backup or remote production sites.

5. In the backup or remote production sites, do the following:

a. Verify that the same number of entries as in the primary production site are available in the customer, balance, configuration, and pricing caches in the backup or remote production sites by using the **query.sh** utility.

See the discussion about using the **query.sh** in *BRM Elastic Charging Engine Implementation Guide* for more information.

b. Verify that the charging server nodes in the backup or remote production sites are in the same state as the charging server nodes in the primary production site.

c. Configure the following ECE core components and the Oracle NoSQL database connection details by using a JMX editor:

Rated Event Formatter

Rated Event Publisher

Diameter Gateway

RADIUS Gateway

Important: If you are configuring a segmented active-active system, ensure the following:

- The name of Diameter Gateway, RADIUS Gateway, Rated Event Formatter, and Rated Event Publisher for each site is unique.
- Minimum two instances of Rated Event Formatter are configured to allow for failover.

See the discussion about implementing ECE with BRM in *BRM Elastic Charging Engine Implementation Guide* for information on configuring Rated Event Formatter and Rated Event Publisher by using a JMX editor.

See "Specifying Diameter Gateway Node Properties" and "Configuring RADIUS Gateway Nodes" for information on configuring Diameter Gateway and RADIUS Gateway by using a JMX editor.

d. If you are configuring a segmented active-active system, start the following ECE processes and gateways:

```
start brmGateway
start ratedEventFormatter
start diameterGateway
start radiusGateway
```

The remote production sites are up and running with all required data.

e. If you are configuring a segmented active-active system, run the following command:

```
gridSync start
```

The federation service is started to replicate the data from the backup or remote production sites to the primary production site.

After starting Rated Event Formatter in the remote production sites, ensure that you copy the CDR files generated by Rated Event Formatter from the remote production sites to the primary production site by using the SFTP utility.

Replicating ECE Cache Data

To replicate the ECE cache data:

1. Go to the *ECE_home*/**oceceserver/bin** directory.
2. Start ECC:

   ```
   ./ecc
   ```

3. Do one of the following:

   - To start replicating data to a specific participant site asynchronously and also replicate all the existing ECE cache data to a specific participant site, run the following commands:

     ```
     gridSync start [remoteClusterName]
     gridSync replicate [remoteClusterName]
     ```

where *remoteClusterName* is the name of the cluster in a participant site.

- To start replicating data to all the participant sites asynchronously and also replicate all the existing ECE cache data to all the participant sites, run the following commands:

  ```
  gridSync start
  gridSync replicate
  ```

See "gridSync" for more information on the **gridSync** utility.

About Disaster Recovery Operations

When the production site fails, perform the following disaster recovery operations:

- **Switchover.** In an active-cold standby system, enable a backup site to take over the role of the production site. See "Switching Over to a BackUp Site" for more information.
- **Failover.** In an active-hot standby system or a segmented active-active system, enable a backup or remote production site to take over the role of the primary production site. See "Failing Over to a Backup Site or a Remote Production Site" for more information.
- **Switchback.** After the original production site is fixed, you can return the sites to their original state by switching the workload back to the original production site. See "Switching Back to the Original Production Site" for more information.

Switching Over to a BackUp Site

To switch over to a backup site:

1. In the backup site, start ECE. See "About Starting and Stopping ECE" for more information.
2. Verify that ECE is connected to BRM and PDC in the backup site by doing the following:

   Important: If only ECE in the production site failed and BRM and PDC in the production site are still running, you must change the BRM, PDC, and Customer Updater connection details in the backup site to connect to BRM and PDC in the production site.

   See the discussions about configuring BRM Gateway and Pricing Updater in *BRM Elastic Charging Engine Implementation Guide* for information on changing the connection details by using a JMX editor.

   - Verify that the JMS queues' details for Pricing Updater in the backup site are specified in the **PdcEceQueue** and **PDCResultQueue** sections of the *ECE_home*/**config/management/migration-configuration.xml** file.
   - Verify that the connection details of BRM in the backup site are provided in the **oracle.communication.brm.charging.appconfiguration.beans.connection.BRMConnectionConfiguration** section of the *ECE_home*/**oceceserver/config/management/charging-settings.xml** file.

3. Load the pricing and customer data into ECE. See the discussions about loading pricing data and customer data into ECE in *BRM Elastic Charging Engine Implementation Guide*.
4. Ensure that the network clients route all requests to the backup site.

> **Important:** Information such as the balance, configuration, and rated event data that was still in the ECE cache when the production site failed is lost.

The former backup site is now the new production site. When you recover the original production site, after performing the planned tasks in the original production site, you can use it again as either the production or backup site. See "Switching Back to the Original Production Site" for more information.

Failing Over to a Backup Site or a Remote Production Site

To fail over to a backup site or a remote production site:

1. In a backup or remote production site, stop replicating the ECE cache data to the primary production site by running the following command:

   ```
   gridSync stop [PrimaryProductionClusterName]
   ```

   where *PrimaryProductionClusterName* is the name of the cluster in the primary production site.
2. In a backup or remote production site, do the following:
   a. In a backup or remote production site, change the BRM, PDC, and Customer Updater connection details to connect to BRM and PDC in the backup or remote production site by using a JMX editor.

   > **Important:** If only ECE in the primary production site failed and BRM and PDC in the primary production site are still running, you need not change the BRM and PDC connection details in the backup or remote production site.

   See the discussions about configuring BRM Gateway and Pricing Updater in *BRM Elastic Charging Engine Implementation Guide* for information on changing the connection details by using a JMX editor.

   b. Start BRM and PDC.

   See the discussion about starting and stopping the BRM system in *BRM System Administrator's Guide* and the discussion about starting and stopping PDC in *PDC Installation and System Administration Guide* for more information.
3. Recover the data in the Oracle NoSQL database data store of the primary production site by doing the following:
   a. Convert the secondary Oracle NoSQL database data store node of the primary production site to the primary Oracle NoSQL database data store node by performing a failover operation in the Oracle NoSQL database data store.

   See the discussion about performing a failover in the Oracle NoSQL Database documentation for more information.

The secondary Oracle NoSQL database data store node of the primary production site is now the primary Oracle NoSQL database data store node of the primary production site.

   b. In a backup or remote production site, convert the rated events from the Oracle NoSQL database data store node that you converted into the primary node in step 3a into CDR files by starting Rated Event Formatter.

      See "Starting and Stopping Rated Event Formatter" for more information.

   c. In a backup or remote production site, load the CDR files that you generated in step 3b into BRM by using Rated Event (RE) Loader.

      See the discussion about setting up RE Loader for ECE in *BRM Elastic Charging Engine Implementation Guide* for more information.

   d. Shut down the Oracle NoSQL database data store node that you converted into the primary node in step 3a.

      See the discussion about shutting down an Oracle NoSQL data store node in the Oracle NoSQL Database documentation for more information.

   e. Stop Rated Event Formatter that you started in step 3b.

      See "Starting and Stopping Rated Event Formatter" for more information.

4. In a backup or remote production site, start Pricing Updater, Customer Updater, and EM Gateway by running the following commands:

   ```
   start pricingUpdater
   start customerUpdater
   start emGateway
   ```

   All the pricing and customer data is now back into the ECE grid in the backup or remote production site.

5. Stop and restart BRM Gateway. See "Starting and Stopping BRM Gateway" for more information.

6. Migrate internal BRM notifications from the primary production site to a backup or remote production site. See "Migrating ECE Notifications" for more information.

   Important: If the expiry duration is configured for these notifications, ensure that you migrate the notifications before they expire. For the expiry duration, see the **expiry-delay** entry for the ServiceContext module in the *ECE_home***/oceceserver/config/charging-cache-config.xml** file.

7. Ensure that the network clients route all requests to the backup or remote production site.

The former backup site or one of the remote production sites is now the new primary production site. You can now recover the original production site and use it again as either the production or backup site.

Switching Back to the Original Production Site

When the original production site is recovered, you can switch back the new production site and original production site to the roles they had prior to the failover operation.

Important: If you switch back to the original production site in an active-cold standby system, any data in memory will be lost. Oracle does not recommend switching back to the original production site in an active-cold standby system.

In an active-hot standby or a segmented active-active system, before you switch back to the original primary production site, you must restart the original primary production site.

To restart the original primary production site:

1. Install ECE and other required components in the original primary production site.

   Important: If only ECE in the original primary production site failed and BRM and PDC in the original primary production site are still running, install only ECE and provide the connection details of BRM and PDC in the original primary production site during ECE installation.

   See the discussion about installing all ECE components in *BRM Elastic Charging Engine Installation Guide* for instructions on ECE integrated installation.

2. Configure primary and secondary Oracle NoSQL data store nodes. See "About Configuring Oracle NoSQL Database Data Store Nodes" for more information.

3. On the machine on which the Oracle WebLogic server is installed, verify that the JMS queues have been created for loading pricing data and for sending event notification and that JMS credentials have been configured correctly.

   See the discussion about creating JMS queues and configuring credentials in *BRM Elastic Charging Engine Installation Guide* for more information.

4. Set the following parameter in the *ECE_home*/**oceceserver/config/ece.properties** file to **false**:

   ```
   loadConfigSettings = false
   ```

   The configuration data is not loaded into memory.

5. Add all the participant sites details in the federation-config section of the ECE Coherence override file (for example, *ECE_home*/**oceceserver/config/charging-coherence-override-prod.xml**).

   To confirm which ECE Coherence override file is used, see the **tangosol.coherence.override** value in the *ECE_home*/**oceceserver/config/ece.properties** file. Table 9–2 provides the federation configuration parameter descriptions and default values.

6. Go to the *ECE_home*/**oceceserver/bin** directory.

7. Start ECC:

   ```
   ./ecc
   ```

8. Start the charging server nodes:

   ```
   start server
   ```

9. Replicate the ECE cache data to the original production site by using the **gridSync** utility. For more information, see "Replicating ECE Cache Data".

10. Start the following ECE processes and gateways:

```
start brmGateway
start ratedEventFormatter
start diameterGateway
start radiusGateway
```

To switch back to the original primary production site in an active-hot standby or a segmented active-active system:

1. Verify that the same number of entries as in the new production site are available in the customer, balance, configuration, and pricing caches in the original production site by using the **query.sh** utility.

   See the discussion about using the **query.sh** in *BRM Elastic Charging Engine Implementation Guide* for more information.

2. Stop Pricing Updater, Customer Updater, and EM Gateway in the new primary production site and then start them in the original primary production site.

3. Migrate internal BRM notifications from the new primary production site to the original primary production site. See "Migrating ECE Notifications" for more information.

4. If you are switching back to BRM and PDC in the original primary production site, do the following:

   a. In the new primary production site, change BRM Gateway, Customer Updater, and Pricing Updater connection details to connect to BRM and PDC in the original primary production site by using a JMX editor.

      See the discussions about configuring BRM Gateway and Pricing Updater in *BRM Elastic Charging Engine Implementation Guide* for information on changing the connection details by using a JMX editor.

   b. Stop RE Loader in the new primary production site and then start it in the original primary production site.

      See the discussion about setting up RE Loader for ECE in *BRM Elastic Charging Engine Implementation Guide* for more information.

   c. Stop and restart BRM Gateway in both the new primary production site and the original primary production site. See "Starting and Stopping BRM Gateway" for more information.

The roles of the sites are now reversed to the original roles.

Migrating ECE Notifications

To migrate ECE notifications:

1. Access the ECE MBeans:
   a. Do one of the following:

      If you are migrating notifications from the primary production site to a backup or remote production site, log on to the driver machine in the backup or remote production site.

      If you are migrating notifications from the new primary production site to the original primary production site, log on to the driver machine in the original primary production site.

   b. Start the ECE charging servers (if they are not started).

   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.

   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home***/oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.

   e. In the editor's MBean hierarchy, expand the **ECE Configuration** node.

2. Expand **systemAdmin**.
3. Expand **Operations**.
4. Select **triggerFailedClusterServiceContextEventMigration**.
5. In the **failedClusterName** parameter, enter the name of the failed site's cluster.
6. Click the **triggerFailedClusterServiceContextEventMigration** button.

All the internal BRM notifications are migrated to the destination site.

10 Troubleshooting ECE

This chapter describes ways to troubleshoot Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE).

Overview of Troubleshooting ECE

The following are the primary components used for diagnosing ECE system problems:

- JConsole

  When ECE is running, you can access Java Management Extensions (JMX) information provided by the Java Virtual Machine (JVM) by using JConsole. You can edit ECE MBean attributes using JConsole. To do so, you must enable one charging server node for JMX management for each unique IP address in your physical topology. See "Enabling a Charging Server Node for JMX Management" for more information.

  For information about JConsole, see the discussion of using JConsole in the JDK product documentation.

- Oracle Enterprise Manager Cloud Control

  If you have Oracle Application Management Pack for Oracle Communications, Oracle Enterprise Manager Cloud Control can be used for providing information about a problem condition in ECE.

  See *Oracle Application Management Pack for Oracle Communications System Administrator's Guide* for more information.

- Log files

  Each ECE node has a log file for logging errors during processing.

  See "Using Error Logs to Troubleshoot ECE" for more information.

  See "Configuring Logging" for information about setting log levels.

- Elastic Charging Controller (ECC)

  ECC has a status command that returns information about nodes in the topology.

  See the ECC help command for information about the status command.

For troubleshooting JVM issues, note the following:

- You can refer to the Java Help Center Web site:

  `http://www.java.com/en/download/help/index.xml`

- You define Java VM arguments and properties for ECE processes in your own JVM tuning files.

For information about configuring your JVM tuning files in ECE, see "Configuring JVM Tuning Parameters".

Troubleshooting JVM and Coherence

ECE is based on JVM technologies and Coherence. When troubleshooting ECE, you may need to troubleshoot JVM and Coherence technologies as well. See "Troubleshooting JVM and Coherence" for more information.

For troubleshooting JVM issues, note the following:

- You can refer to the Java Help Center Web site:

  `http://www.java.com/en/download/help/index.xml`

- JVM arguments and properties are defined in the *ECE_home*/**config/ece.properties** file, where *ECE_home* is the directory in which ECE is installed.
- Java tuning profiles for nodes are defined in the *ECE_home*/**oceceserver/config/defaultTuningProfile.properties** file.
- When ECE is running, you can access JMX information provided by the JVM by using JConsole. For more information, see the Web site at:

  `http://docs.oracle.com/javase/7/docs/technotes/guides/management/jconsole.html`

For troubleshooting Coherence issues, note the following:

- You can access runtime information about Coherence through JMX, specifically through the Coherence MBeans (which are used to manage and monitor different parts of Coherence).

  See the Coherence MBean documentation for the meaning of Coherence JMX metrics as they relate to ECE.

  See the Coherence MBean reference in *Oracle Coherence Developer's Guide Release 3.6.1* documentation at:

  `http://docs.oracle.com/cd/E15357_01/coh.360/e15723/appendix_mbean.htm`

- You can access the Oracle Coherence Community Resources Web site at:

  `http://www.oracle.com/technetwork/middleware/coherence/community/index.html`

- See "Configuring Coherence" for information about the ECE configuration files related to configuring Coherence.

Troubleshooting Checklists

For information about troubleshooting checklists, refer to the following topics:

- ECE Troubleshooting Checklist When Offline
- ECE Troubleshooting Checklist When Online
- ECE General Troubleshooting Checklist

ECE Troubleshooting Checklist When Offline

You can use the following troubleshooting checklist when ECE is not connected to external applications in the charging system:

- Verify the installation layout and the VERSION file.
- Verify ECE configuration files (under the *ECE_home*/**oceceserver/config** directory). See "Configuring the ECE System" for information about configuration options.

  In particular, check the topology, the tuning profiles (if any), and the **ece.properties** file.
- Verify Coherence configuration files (under the *ECE_home*/**oceceserver/config** directory).
- Verify bi-directional password-less SSH between the driver machine and the server machines.

  Servers should be accessible from the driver machine by way of SSH without entering a password, and the driver machine should be accessible from each server machine by way of SSH without a password.
- Verify the security configuration.

  Some examples of security errors are as follows:

  – If the server fails with the error **password file read access must be restricted**, the cause may be that the **jmxremote.password** file has write access. The solution is to remove write access to this file (chmod 400 *ECE_home*/**oceceserver/config/jmxremote.password**) or disable the JMX security setting if it is not needed.

  – If the server fails with the error **cannot find security credentials**, the cause may be that the Coherence security settings are incorrect. The solution is to correct the Coherence security settings properties in the following files:

    ECE_home/**oceceserver/config/defaultTuningProfile.properties**

    ECE_home/**oceceserver/config/ece.properties**
- Verify ECE deployment in remote machines. Issues can arise from out-of-synchronization installations on various remote machines. Running the ECC **sync** command can ensure that all remote machines are in synchronization.

ECE Troubleshooting Checklist When Online

You can use the following troubleshooting checklist when ECE is connected to external applications in the charging system:

- Verify nodes are running through JMX such as by using JConsole.
- Verify the machine state of nodes.

  In order to start processing requests, ECE goes through different states, such as startup, configuration, update, and processing and so on. State Manager controls the state transition. A node that is in a state other than USAGE_PROCESSING will not process usage requests.
- Verify the charging-server health threshold.

  The number of running nodes may have gone below the threshold. See "Configuring Charging-Server Health Thresholds" for information.
- Verify ECE management settings.

  All management configuration files are located under *ECE_home*/**oceceserver/config/management**. See "About ECE Configuration" for more information.

ECE General Troubleshooting Checklist

When any problems occur, do some troubleshooting before you contact Oracle support. You know your installation better than Oracle support does. You know if anything in the system has been changed, so you are more likely to know where to look first. If you have a problem with your ECE system, ask yourself the following questions first, because Oracle support will ask them of you:

- What exactly is the problem? Can you isolate it? For example, if it is a problem with an application, does it occur on one instance of the application, or all instances?

  Oracle technical support needs a clear and concise description of the problem, including when it began to occur.
- What do the log files say?

  Oracle technical support asks for this information first. Check the error log for the ECE component you are having problems with.
- Read through the ECE troubleshooting checklist. Look through the list of common problems and their solutions.
- Has anything changed in the system? Did you install any new hardware or new software? Did the network change in any way?
- Have you read the Release Notes?

  The Release Notes include information about known problems and workarounds.
- Does the problem resemble another one you had previously?
- Has your system usage recently jumped significantly?
- Is the system otherwise operating normally?
- Has response time or the level of system resources changed?
- Are users complaining about additional or different problems?
- Can you run clients successfully?
- Are any other processes on the system hardware functioning normally?

If you still cannot resolve the problem, contact Oracle technical support as described in "Getting Help for ECE Problems".

Using Error Logs to Troubleshoot ECE

ECE error log files provide detailed information about system problems. If you have a problem with an ECE process or node, look in its corresponding log file in the *ECE_home***/oceceserver/logs** directory. This directory is where log files are generated for each node on each machine of your topology.

Log files include errors that need to be managed, as well as errors that do not need immediate attention (for example, invalid login attempts). To manage log files, make a list of the important errors for your system to discern them from errors that do not require immediate attention.

See "Configuring Logging" for information about setting log levels.

Understanding Error-Message Syntax

ECE error messages use the following syntax:

```
log4j.appender.ECE_LOG.layout.ConversionPattern=%d
```

```
{yyyy-MM-dd HH:mm:ss.SSS zzz} %5p -
%X{correlationId} - %X{requestId} - %X{customerId} - %m%n
```

For example:

```
2012-09-11 22:27:09.565 PDT DEBUG -
324984520132531235 - 49de072b-33ae-4f6c-816d-35c25b9ade78 - Cust#6500000587 -
 Successfully retrieved tariffPolicy from customer; candidate balance ids ::
 [Bal#6500000587\]
```

Collecting Diagnostic Information

For collecting diagnostic information, turn on the ECC feedback mode, which produces extra information when running commands. For example:

```
ecc:000>set feedback true
```

The feedback mode setting is saved in your local profile so you do not need to set it every time you start ECC.

See "Configuring Logging" for information about how to control ECE logging and log levels.

Collecting Log Files for Sending to Oracle Support

You can use the **infoCollector** ECC command to collect log files from your ECE system. The command copies files that are relevant to diagnosing problems from all server machines and the driver machine and puts them in a well known location. The command also creates a compressed TAR file of all collected files, which you can then send to Oracle support.

The **infoCollector** command does not collect files from systems on which the Elastic Charging Client is running (for example, it does not collect files from the network mediation system).

To collect log files from your ECE system:

1. (Optional) Create a directory to put the collected log files.

   If you do not specify a directory, the command puts the collected log files in the user home directory (the directory to which your platform $HOME environment variable points).

2. Log on to the driver machine.

3. Change directory to the *ECE_home*/**oceceserver/bin** directory.

4. Start ECC:

   ```
   ./ecc
   ```

5. Run the following command:

   ```
   infoCollector
   ```

   See "infoCollector Syntax" for information about syntax and arguments for running the command.

   The following occurs:

   - From each server machine in your topology, the following file and directories are copied to the following location on the driver machine (where *user_home* is the user home directory of the driver machine and *server_host* is the IP address or host name of the server machine as it is defined in the *ECE_home*/**oceceserver/config/eceTopology.conf** file):

  – *user_home*/*server_host*/**oceceserver/VERSION**
  – *user_home*/*server_host*/**oceceserver/logs**
  – *user_home*/*server_host*/**oceceserver/config**
  – *user_home*/*server_host*/**oceceserver/brm_config**
  – *user_home*/*server_host*/**oceceserver/odi_transformation**

- From the driver machine, the following file and directories are copied to the following location on the driver machine (where *user_home* is the user home directory of the driver machine and *driver_host* is the IP address or host name of the driver machine as it is defined in the *ECE_home*/**oceceserver/config/eceTopology.conf** file):
  – *user_home*/*driver_host*/**oceceserver/VERSION**
  – *user_home*/*driver_host*/**oceceserver/config**
  – *user_home*/*driver_host*/**oceceserver/brm_config**
- A compressed TAR file, of all copied files, is created with the extension **tar.gz** in the user home directory of the driver machine (for example, *user_home*/**info_collector.tar.gz**).

infoCollector Syntax

```
infoCollector [-v] [-nc] [-l] [-d dir] [-s] [-e "FileFilter", "FileFilter2",
"..."]
```

where:

- **-v** turns on verbose mode that outputs information pertaining to the collected files; this option provides status of what is being copied as the command collects files.
- **-nc** does *not* compress the resulting directory into a compressed TAR file.
- **-l** includes all log files into the collection; if **-l** is omitted, the command collects only those log files that match the node-name with the **.log** suffix that you specify.
- **-d** *dir* uses the directory you specify to hold all collected data where *dir* is the path of the directory.
- **-s** includes all files from the *ECE_home*/**oceceserver/sample_data** directory.
- **-e "***FileFilter***"** is the path name or path name pattern of custom directories or files to collect and include in the compressed TAR file.

  Separate multiple filters with a comma.

  For example, assume in your *ECE_home* directory you have the files **notes.txt**, **comments_1.txt**, **comments_2.txt**, and **comments_3.txt**, and you also have a directory named **observations** that contains the files **observation1.txt**, **observation2.txt**, and **observation3.txt**.

  The path name pattern can be either an explicit file name such as **notes.txt**, an entire directory tree such as **observations**, or a wildcard * such as shown for the files that begin with **comments** in the following example:

  ```
  infoCollector -e "/home/example/ECE_11.2.0.2/notes.txt",
  "/home/example/ECE_11.2.0.2/observations", "/home/example/ECE_
  ```

```
11.2.0.2/comments*"
```

The preceding **infoCollector** command would put all custom files and directories into a directory named **extra_files** and the resulting collection of files would have the following directory structure:

```
infoCollector_2014-05-07T11:02:10
  localhost-DRIVER
    extra_files
      observations
        observation3.txt
        observation2.txt
        observation1.txt
      notes.txt
      comments_3.txt
      comments_2.txt
      comments_1.txt
    config
    brm_config
    VERSION
```

Displaying Data in the Coherence Caches When Troubleshooting

You can use the **query.sh** script for querying Coherence caches and displaying the data that you want.

See the discussion of the query tool in *BRM Elastic Charging Engine Implementation Guide* for more information.

Troubleshooting Failed Diameter-Message Processing in Diameter Gateway

If you suspect a problem with how Diameter Gateway nodes are processing Diameter messages, look in the *ECE_home*/**oceceserver/logs/***Instance_Name***.log** files for errors, where *Instance_Name* is the name of the Diameter Gateway-node instance (a name you defined in the ECE topology file) that you need to troubleshoot. For example, look in **diameterGateway1.log**. The log file contains information about any errors during Diameter-message processing.

To set log levels for Diameter Gateway nodes, obtain the Diameter Gateway module names in the *ECE_home*/**oceceserver/config/log4j.properties** file, and then set the log levels by module as described in "Setting Log Levels by Editing MBeans".

Diameter Gateway returns all Diameter result codes (Result-Codes) as part of the Credit Control Answer (CCA) message. When an error occurs, the error ID and name are returned in the result code. For example, if the CCR was missing an Event-Timestamp AVP, the error would be:

```
DiameterTalk Answers =[
Diameter Message: CCA
Version: 1
Msg Length: 144
Cmd Flags: PXY
Cmd Code: 272
App-Id: 4
Hop-By-Hop-Id: 1497412149
End-To-End-Id: 734750287
  Session-Id (263,M,l=11) = 111
```

```
Result-Code (268,M,l=12) = DIAMETER_MISSING_AVP (5005)
Origin-Host (264,M,l=24) = dgw1.example.com
Origin-Realm (296,M,l=19) = example.com
Auth-Application-Id (258,M,l=12) = 4
CC-Request-Type (416,M,l=12) = INITIAL_REQUEST (1)
CC-Request-Number (415,M,l=12) = 0
Failed-AVP (279,M,l=20) =
  Event-Timestamp (55,M,l=12) = 3627391363 (Fri Dec 12 08:42:43 PST 2014)
```

For the error, an error message is written to the **diametergateway***Instance_Name***.log** file that indicates the nature and stack trace for the error.

For information about Diameter Gateway result codes, see "Diameter Gateway Result Codes".

Diameter Gateway nodes must be started after the customer data is loaded into the ECE grid; otherwise, they cannot process Diameter requests. For information about starting Diameter Gateway, see "Starting and Stopping ECE".

Troubleshooting Failed RADIUS-Message Processing in RADIUS Gateway

If you suspect a problem with how RADIUS Gateway nodes are processing RADIUS messages, see the following files for errors that you need to troubleshoot:

- *ECE_home***/oceceserver/logs/***Instance_Name***.log** files, where *Instance_Name* is the name of the RADIUS Gateway-node instance (a name you defined in the ECE topology file); for example, *ECE_home***/oceceserver/logs/radiusGateway1.log**
- Charging-server node log files; for example, *ECE_home***/oceceserver/logs/ecs1.log**.

To set log levels for RADIUS Gateway nodes, obtain the RADIUS Gateway module names in the *ECE_home***/oceceserver/config/log4j.properties** file, and then set the log levels by module as described in "Setting Log Levels by Editing MBeans".

RADIUS Gateway returns all the results as part of the reply-message attribute-value pair (AVP) in the RADIUS response. For example, if the user password in the authentication request is incorrect, the following error message is returned in the RADIUS response:

```
Session_Timeout AVP after Deletion : null
2016-03-07 23:37:58.896 PST DEBUG -  -  -  - ECE Radius server - Sending the
response to client
 Code: Access-Reject(3)
 Identifier: 0
 Length: 20
 Authenticator: 0x00000000000000000000000000000000
 Reply-Message: RadiusGatewayMessagesBundle-31015: Incorrect password from User
 User-Name: 0049100033
```

Troubleshooting Failed Update Requests from BRM

Update requests pass event information from Oracle Communications Billing and Revenue Management (BRM) to ECE; occasionally, update-request events fail. The following are examples of when update requests from BRM might fail:

- A delay in receiving Pricing Design Center (PDC) pricing data

  ECE does not receive a charge or discount offer from PDC but does receive the offer's external ID in an update request from BRM. Because the offer is not loaded into ECE due to the PDC delay, the external ID update does not find the offer in the cache. Consequently, the offer update fails.

- Configuration errors in customer data.
- Errors in management requests associated with the rerating process.

  Management-type requests for the rerating process, such as PrepareToRerate and RerateCompleted, may fail. For information on troubleshooting rerating errors, see "Troubleshooting Problems with Rerating".

The preceding types of failed update requests are placed in a suspense queue. During the post-installation phase of an integrated ECE system, you create the suspense queue, and you create a queue table called **ifw_sync_sus**. See the discussion about creating the suspense queue in *BRM Elastic Charging Engine Installation Guide.*

About Customer Updater Error Log Files

After failed requests are placed in the suspense queue, you view and manually reprocess them. When failed events are added to the suspense queue, error messages are recorded in the *ECE_home*/**oceceserver/logs/customerUpdater.log** files.

Viewing Failed Events in the Suspense Queue

To view failed events in the suspense queue:

1. Map the suspense queue to the **ifw_sync_sus** table.
2. Run the **select user_data from ifw_sync_sus** query to view the failed events after connecting to the SQL database.

For more information, see the discussion about installing account synchronization in *BRM Installation Guide*.

Removing Excess Failed Updates in the BRM Gateway Suspense Queue

If the failed updates in the BRM suspense queue are not processed, they stay in the queue and an error is logged in the file *ECE_Home*/**oceceserver/logs/Instance_Name.log**.

You can check the log file as part of the daily health status monitoring, and in case of an error in the BRM Gateway suspense queue, you can manually move or delete failed updates from the queue and restart the WebLogic server.

To manually move or delete a failed update:

1. Log on to the WebLogic server on which the BRM Gateway suspense queue resides.
2. Log in to WebLogic Server Administration Console.
3. In the **Services** section, select **JMS Modules.**
4. In the **JMS Modules**, click **ECE Module**.
5. In **Summary of Resources**, click **Suspense Queue**.
6. Click **Monitoring**, and select **ECE!Suspense Queue**.
7. Click **Show Messages**.

   The Summary of **JMS Messages** appears.
8. In the **JMS Messages** table, select a message.
9. Do one of the following:
   - To move a message:

a. Click **Move**.

b. In the **NotificationServer** field, select **JMS Server**.

c. In the **DestinationServer** field, select **Suspense Queue.**

d. Click **Finish**.

   The failed update is sent back to the BRM Gateway suspense queue for reprocessing.

- To delete a message, click **Delete**.

  The failed update is deleted from the BRM Gateway suspense queue.

Propagating Events from the Suspense Queue to the Account Synchronization DM Database Queue

After you view the events in the suspense queue and identify the cause of failure, you propagate the events from the suspense queue to the Account Synchronization Data Manager (DM) database queue with the **events_propagate_utility** script. Customer Updater reprocesses the events. For more information, see "events_propagate_utility.pl".

Troubleshooting Problems with Coherence

When loading the pricing data from PDC into ECE, if you get the "No space left on device" error, configure the Coherence flash journal resources manager, which manages temporary journal-based files, by doing the following:

Important: This configuration may affect the performance of your system.

1. Open the ECE Coherence override file your ECE system uses (for example, *ECE_home*/**oceceserver/config/charging-coherence-override-prod.xml**).

   To confirm which ECE Coherence override file is used, refer to the **tangosol.coherence.override** parameter of the *ECE_home*/**oceceserver/config/ece.properties** file.

2. Add the following section:

```
<journaling-config>
        <flashjournal-manager>
                       <directory>/logs/oracle/flashjournal</directory>
                       <tmp-purge-delay>15m</tmp-purge-delay>
                       <maximum-file-size>100MB</maximum-file-size>
                       <high-journal-size
system-property="coherence.flashjournal.highjournalsize">10GB</high-journal-si
ze>
     </flashjournal-manager>
</journaling-config>
```

3. Save and close the file.

Troubleshooting Problems with Rerating

Rerating errors can occur at different stages of the rerating process. If ECE cannot rerate events for a customer, rerating errors are handled as follows for each stage:

- **In the prepare-to-rerate stage**

  Errors in this stage are logged in the **CustomerUpdater.log** file with appropriate reasons. No acknowledgement is sent so the acknowledgement queue will be empty.

- **During rerating**

  Errors are logged in the **emGateway.log** with appropriate reasons.

- **In the rerate-complete stage**

  Errors are logged in the **CustomerUpdater.log** file. The acknowledgement will be sent back to BRM. ECE sends a notification to BRM using BRM Gateway to create a new rerate job. BRM uses the information in the notification for creating a new rerate job for that customer.

Troubleshooting Performance Issues by Using Coherence JMX Metrics

ECE provides Coherence metrics that can help you troubleshoot performance problems and performance tuning, and isolate hardware issues.

To troubleshoot performance issues by using Coherence JMX metrics:

1. Access the ECE MBeans:
   a. Log on to the driver machine.
   b. Start the ECE charging servers (if they are not started).
   c. Start a JMX editor, such as JConsole, that enables you to edit MBean attributes.
   d. Connect to the ECE charging server node set to **start CohMgt = true** in the *ECE_home*/**oceceserver/config/eceTopology.conf** file.

      The **eceTopology.conf** file also contains the host name and port number for the node.
   e. In the editor's MBean hierarchy, expand the **Coherence** node.
2. Expand **Service**.
3. View the Coherence JMX metrics that apply to your troubleshooting scenario. For example:
   - Expand **InvocationService**, select the appropriate node, and expand **Attributes** .
     – The **TaskCount** attribute specifies the number of request batches received by the node.
     – The **TaskAverageDuration** attribute specifies the average request batch latency for the node.
   - Expand **BRMDistributedCache**, select the appropriate node, and expand **Attributes**.
     – The **RequestTotalCount** attribute specifies the following, depending on the request type:

       For Initiate, Update, Terminate, and Cancel requests, it specifies the number of entry processor invocations.

       For Auth Query requests, it specifies the number of get() operations.
     – The **RequestAverageDuration** attribute specifies the following, depending on the request type:

       For Initiate, Update, and Terminate requests, it specifies entry processor latency.

       For Auth Query requests, it specifies get latency.

Note: To reset the attribute values for a **Service** subnode, expand the subnode’s **Operations** node, select the **resetStatistics** operation, and click the **resetStatistics** button.

Getting Help for ECE Problems

If you cannot resolve your ECE problem, contact Oracle support.

Before You Contact Oracle Support

Problems can often be fixed simply by stopping and restarting a node. To stop and restart ECE nodes, see "Starting and Stopping ECE".

If that does not solve the problem, look at the error log for the application or process that reported the problem. See "Using Error Logs to Troubleshoot ECE". Be sure to observe the checklist for resolving problems with ECE before reporting the problem to Oracle technical support. See "Troubleshooting Checklists".

Reporting Problems

If the checklist for resolving problems with ECE does not help you to resolve the problem, write down the pertinent information:

- A clear and concise description of the problem, including when it began to occur.
- Relevant portions of the relevant log files.
- Relevant configuration files.
- Recent changes in your system, even if you think they are not relevant.
- List of all ECE components and patches installed on your system.

When you are ready, report the problem to Oracle support.

11 ECE System Administration Utilities

This chapter provides reference information for Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) utilities that are used for system administration.

encrypt.sh

Use the **encrypt.sh** utility to encrypt passwords that ECE uses for connecting to Oracle Communications Billing and Revenue Management (BRM) and Pricing Design Center (PDC).

For more information, see "About Managing External Application Passwords".

Location

ECE_Home/**oceceserver/bin**

Syntax

```
encrypt.sh string storepassword
```

where:

- *string* is the new password to be encrypted
- *storepassword* is the password to the keystore

Parameters

-help
Displays the syntax for this utility.

Results

The encrypt utility returns a message containing the encrypted value for the new password.

When you run the script, information is logged in the *ECE_Home*/**logs/encrypt.log** file.

events_propagate_utility.pl

Use the **events_propagate_utility.pl** utility to propagate events on update requests from the suspense queue to the BRM Account Synchronization DM database queue.

For more information, see "Troubleshooting Failed Update Requests from BRM".

Location

ECE_Home/**brm_utils/**

Copy **event_propagate.sql** and **events_propagate_utility.pl** from the ECE installation directory (*ECE_Home*/**brm_utils/**) to your BRM machine (for example, *BRM_Home*/**sys/test/**).

Syntax Overview

The following actions are supported for **events_propagate_utility**:

- Syntax for Moving Events from the Suspense Queue
- Syntax for Listing All Queues
- Syntax for Loading Data into the Database
- Syntax for Purging Data from a Queue
- Syntax for Getting Help

Syntax for Moving Events from the Suspense Queue

Moves events from the suspense queue to the BRM Account Synchronization DM database queue.

```
perl events_propagate_utility.pl move

-s queue_name
-d queue_name
[-l user_id/password@oracle_sid]
-n number_of_events
```

Parameters for Moving Events from the Suspense Queue

The following parameters are used to move events from the suspense queue to the BRM Account Synchronization DM database queue:

- **-s** *queue_name*

  Specifies the suspense queue name.

- **-d** *queue_name*

  Specifies the BRM Account Synchronization DM database queue name.

- **-l** *user_id/password@oracle_sid*

  Specifies the user ID and password to connect to the database.

- **-n** *number_of_events*

  Specifies the number of events being moved from the suspense queue.

Syntax for Listing All Queues

Lists all queues on a BRM machine.

```
perl events_propagate_utility.pl list

[-l user_id/password@oracle_sid]
```

Parameters for Listing All Queues

The following parameters are used to list all queues on a BRM machine:

- **-l** *user_id/password@oracle_sid*

  Specifies the user ID and password to connect to the database.

Syntax for Loading Data into the Database

Loads data from a queue into the database.

```
perl events_propagate_utility.pl load

[-l user_id/password@oracle_sid]
```

Parameters for Loading Data into the Database

The following parameters are used to load data into the database:

- **-l** *user_id/password@oracle_sid*

  Specifies the user ID and password to connect to the database.

Syntax for Purging Data from a Queue

Purges data from the suspense queue or the BRM Account Synchronization DM database queue.

```
perl events_propagate_utility.pl purge

[-l user_id/password@oracle_sid]
-q queue_name
```

Parameters for Purging Data from Queues

The following parameters are used to purge data from the suspense queue and the BRM Account Synchronization DM database queue:

- **-l** *user_id/password@oracle_sid*

  Specifies the user ID and password to connect to the database.

- **-q** *queue_name*

  Specifies the queue name from which data is purged.

Syntax for Getting Help

Displays the syntax for the **events_propagate_utility** script.

```
perl events_propagate_utility.pl help
```

Results

The **events_propagate_utility** script notifies you when it runs successfully. Otherwise, look in the logs.

gridSync

Use the **gridSync** utility to start the federation service for replicating the ECE cache data to the participant sites asynchronously and also replicate the existing ECE cache data to the participant sites.

For more information, see "Replicating ECE Cache Data".

Location

ECE_Home/**oceceserver/bin**

Syntax

```
[-help] gridSync {start|stop|pause|replicate}[remoteClusterNames]
```

Parameters

help

Displays the syntax for the **gridSync** utility.

start

Starts the federation service to replicate the ECE cache data to the backup or remote production sites.

stop

Stops replicating ECE cache data to the backup or remote production sites.

pause

Pauses replication of ECE cache data to the backup or remote production sites.

replicate

Replicates all the existing ECE cache data to the backup or remote production sites.

remoteClusterNames

Specify the name of the Coherence cluster in the backup site.

Results

When you run the **gridSync** utility, information is logged in the *ECE_Home*/**logs/encrypt.log** file.

A

ECE Directory Structure and Contents

This appendix describes the Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) directory structure and directory contents.

ECE Directory Structure and Contents

The tables that follow summarize the directory structure and directory contents of the following:

- ECE Server Software
- ECE SDK
- ECE BRM Integration Pack
- ECE Oracle Management Server Integration Pack

ECE Server Software

Table A–1 shows the ECE server software directory structure: *ECE_home* is the directory where ECE is installed.

Table A–1 Elastic Charging Engine Server Software Directory Structure and Contents

Directory	Description
ECE_home/**oceceserver/bin**	Scripts.
ECE_home/**oceceserver/config**	Configuration files.
ECE_home/**oceceserver/config/management**	Configuration files for configuring usage-charging business rules, data migration preferences, and developer tool settings. All of the configurations defined in these files can be viewed and changed through JMX.
ECE_home/**oceceserver/lib**	Libraries such as JAR files and binaries.
ECE_home/**oceceserver/logs**	Log files generated for each node on each machine.

Table A–1 (Cont.) Elastic Charging Engine Server Software Directory Structure and

Directory	Description
ECE_home/**oceceserver/sample_data**	Sample data used by ECE for an initial standalone installation, which includes sample data for integrating with Oracle Communications Billing and Revenue Management (BRM) and Pricing Design Center (PDC).
ECE_home/**oceceserver/sample_data/config_data**	Sample data used by ECE for an initial standalone installation.
ECE_home/**oceceserver/sample_data/config_ data/specifications/ece_end2end**	Sample ECE request specification files. **Note:** Sample request specification files are not intended for production systems.
ECE_home/**oceceserver/sample_data/crossref_data**	Sample data used by ECE for an initial standalone installation.
ECE_home/**oceceserver/sample_data/customer_data**	Sample data used by ECE for an initial standalone installation.
ECE_home/**oceceserver/sample_data/policy_data**	Sample data that can be used for testing the integration of policy clients with ECE. Includes sample data files for configuration data, pricing data, product cross-reference data, and customer data.
ECE_home/**oceceserver/sample_data/pricing_data**	Sample data used by ECE for an initial standalone installation.

ECE SDK

Table A–2 shows the ECE SDK software directory structure: *ECE_home* is the directory in which ECE is installed.

Table A–2 Elastic Charging Engine SDK Directory Structure and Contents

Directory	Description
ECE_home/**ocecesdk/bin**	Directories that contain shell scripts for compiling and running various types of sample programs.
ECE_home/**ocecesdk/bin/extensions**	Shell scripts for extension-implementation sample programs.
ECE_home/**ocecesdk/bin/notification**	Shell scripts for notification sample programs.
ECE_home/**ocecesdk/bin/plugin**	Shell scripts for BrmCdrPluginDirect plug-in sample programs.
ECE_home/**ocecesdk/bin/policy**	Shell scripts for policy sample programs.
ECE_home/**ocecesdk/bin/query**	Shell scripts for query sample programs.

Table A–2 (Cont.) Elastic Charging Engine SDK Directory Structure and Contents

Directory	Description
ECE_home/**ocecesdk/bin/update**	Shell scripts for update sample programs.
ECE_home/**ocecesdk/bin/usage**	Shell scripts for usage sample programs.
ECE_home/**ocecesdk/config**	Configuration files common to all sample programs.
ECE_home/**ocecesdk/config/extensions**	Configuration files for extension-implementation sample programs.
ECE_home/**ocecesdk/source** **Note:** The full path showing the Java project directory structure to the sample programs is: *ECE_home*/**ocecesdk/source/oracle/communication/brm/charging/sdk**	All Java sample programs.
ECE_home/**ocecesdk/source/.../sdk/extensions**	Source files for extension-implementation sample programs (for pre-request processing and post-request processing). Includes the data loader used for extensions.
ECE_home/**ocecesdk/source/.../sdk/notification**	Source files for notification sample programs.
ECE_home/**ocecesdk/source/.../sdk/plugin**	Source files for BrmCdrPluginDirect plug-in sample programs.
ECE_home/**ocecesdk/source/.../sdk/policy**	Source files for policy sample programs.
ECE_home/**ocecesdk/source/.../sdk/query**	Source files for query sample programs.
ECE_home/**ocecesdk/source/.../sdk/update**	Source files for update sample programs.
ECE_home/**ocecesdk/source/.../sdk/usage**	Source files for usage sample programs.

ECE BRM Integration Pack

Table A–3 shows the ECE BRM Integration Pack software directory structure: *ECE_home* is the directory in which ECE is installed.

Table A–3 Elastic Charging Engine BRM Integration Pack Directory Structure and Contents

Directory	Description
ECE_home/**oceceserver/brm_config**	Configuration files to apply to your BRM installation when integrating ECE with BRM.
ECE_home/**oceceserver/odi_transformation**	Oracle Data Integrator configuration files and scripts used for extracting data from BRM.

ECE Oracle Management Server Integration Pack

Table A–4 shows the ECE Oracle Management Server Integration Pack software directory structure: *ECE_home* is the directory in which ECE is installed.

Table A–4 ECE Oracle Management Server Integration Pack Directory Structure and Contents

Directory	Description
ECE_home/**oceceem**	Configuration files for the ECE Oracle Management Server Integration Pack.

B ECC Commands

This appendix describes the Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) Elastic Charging Controller (ECC) commands.

See "Using the Elastic Charging Controller to Manage Nodes" for information about ECC.

ECC Commands

Table B–1 shows the ECC commands and their descriptions.

Table B–1 Elastic Charging Controller (ECC) Commands

Command	Description
addNode	Add an ECE node. addNode runs the sync command which installs ECE on the host machine of the node. addNode adds an entry for the node in the topology file. addNode does not start the node. You must use the start command to start the node.
cat	Concatenate and print files.
encrypt	Encrypt the given string.
:exit	Exit the shell.
:help	Get help for commands.
:history	Get history.
infoCollector	Collect log files.
init	(For simulator only) Initialize a node by role.
jps	Launch the Java Virtual Machine Process Status Tool.
kill	Same as bash kill.
ls	Same as bash ls.
removeNode	Remove an ECE node. You must stop the node before running removeNode. removeNode reports a warning if the node is in a running state.
rollingUpgrade	Perform a rolling upgrade of ECE.

Table B–1 (Cont.) Elastic Charging Controller (ECC) Commands

Command	Description
shutdown	(For simulator only) Gracefully terminate simulator by name or by role.
simulate	(For simulator only) Start simulating request on node by role or name.
status	Display cluster information or status. **status** goes through each node defined in the topology to query if the node is started or stopped.
start	Start nodes by name or by role.
stop	Stop nodes by name or by role.
sync	Install ECE on all remote hosts as specified in the topology file. See "Configuring ECE Topology".
tail	Display the last part of a file.

C

ECE Configuration File Reference

This appendix describes Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) configuration files.

System Configuration Files

Table C–1 summarizes configuration files relevant to tuning the ECE system, such as tuning Oracle Coherence and JVM parameters. All system configuration files are in *ECE_Home*/**oceceserver/config**.

See "Configuring the ECE System" for information about configuring system-level parameters.

Table C–1 ECE System Configuration File Reference

File Name	Description
charging-cache-config.xml	Used for configuring Coherence. Do not modify.
charging-coherence-override-dev.xml	Used for configuring Coherence. Used for development.
charging-coherence-override-prod.xml	Used for configuring Coherence. Used for production.
charging-coherence-override-secure-prod.xml	Used for configuring Coherence. Used for production. When you enable SSL, the key and store passwords must be set in this file.
charging-context.xml	System configuration file. Do not modify.
charging-pof-config.xml	System configuration file. Do not modify.
coherence-jaas.config	Used for Java security. Do not modify.
defaultTuningProfile.properties	Used for setting JVM tuning parameters, such as those for garbage collection and heap size. Typically, the default values do not need to be modified. See "Configuring JVM Tuning Parameters".
ece.properties	System configuration file.

Table C–1 (Cont.) ECE System Configuration File Reference

File Name	Description
eceTopology.conf	Used for specifying the physical server machines for each Coherence node in the cluster. See "Configuring ECE Topology".
groovysh.profile	System configuration file. Do not modify.
JMSConfiguration.xml	Used for configuring the JMS credentials for the JMS server on which the notification queue (JMS topic) resides. See "Configuring Notifications".
jmxremote.password	Used for JXM security. See the discussion of performing a secure ECE installation in *BRM Elastic Charging Engine Security Guide*.
jrds.groovy	System configuration file. Do not modify.
keystore.jks	Used for authentication of applications outside of the cluster, such as Pricing Design Center (PDC) and Oracle Communications Billing and Revenue Management (BRM) (Coherence cluster security).
log4j.properties	Used for configuring logging for each node in the cluster. This file is used only initially before the ECE system is running. After ECE is running, you use JConsole to set logging.
Notification.xsd	XSD schema file for XML notification messages generated by ECE.
permissions.xml	Used for authorization of nodes in the Coherence cluster. See the discussion of performing a secure ECE installation in *BRM Elastic Charging Engine Security Guide*.
sdkTuningProfile.properties	Used for configuring tuning parameters for the sample programs included in the ECE SDK. Typically, the default values do not need to be modified.
server.jks	Used for authentication of nodes in the Coherence cluster (Coherence cluster security). Do not modify. See the discussion of setting up and managing ECE security in "Setting Up and Managing Elastic Charging Engine Security".

Business Configuration Files

Table C–2 summarizes configuration files relevant to setting ECE charging business parameters and runtime configurations. All business configuration files are in *ECE_Home***/oceceserver/config/management**.

For information about configuring usage-charging parameters, see the discussion of implementing charging business rules in *BRM Elastic Charging Engine Implementation Guide*.

Table C–2 ECE Business Configuration File Reference

File Name	Description
charging-settings.xml	Used for configuring charging business rules and runtime configurations. This file is also used for configuring the following ECE components: ▪ Used for configuring the BRM Gateway. See the discussion of implementing ECE with Oracle Communications Billing and Revenue Management in *BRM Elastic Charging Engine Implementation Guide*. ▪ Used for configuring Rated Event Formatter. See the discussion of implementing ECE with BRM in *BRM Elastic Charging Engine Implementation Guide*. ▪ Used for configuring Customer Updater so that Customer Updater can dequeue update requests sent from BRM. See the discussion of implementing ECE with BRM in *BRM Elastic Charging Engine Implementation Guide*.
migration-configuration.xml	Used for specifying the directories from which you want the data-loading utilities (configuration loader, pricing loader, and customer loader) to load data.
test-tools.xml	Used for specifying developer tool settings.

D

Diameter Gateway Result Codes

This appendix describes Oracle Communications Billing and Revenue Management Elastic Charging Engine (ECE) Diameter Gateway result codes. For more information, see the Result-Code AVP section in the IETF web site at:

`https://tools.ietf.org/html/rfc3588#section-7.1`

Supported Result Codes

Diameter Gateway supports the following categories of result codes:

- Success Result Codes
- Protocol-Error Result Codes
- Transient-Failure Result Codes
- Permanent-Failure Result Codes

Success Result Codes

Success result codes are returned when the request is successfully completed.

Table D–1 lists the success result codes.

Table D–1 Success Result Codes

Result Code	Code	Notes
DIAMETER_SUCCESS	2001	The diameter request was executed successfully.

Protocol-Error Result Codes

Protocol-error result codes are returned at the base protocol level. The diameter proxy agents may attempt to correct these errors.

Table D–2 lists the protocol-error result codes.

Table D–2 Protocol-Error Result Codes

Result Code	Code	Notes
DIAMETER_COMMAND_ UNSUPPORTED	3001	The command-code in the diameter request is not recognized or supported by the diameter node. Verify that the command-code sent by the peer conforms to Gy, Sh, or Sy protocol specifications.
DIAMETER_UNABLE_TO_ DELIVER	3002	The origin and destination information in the diameter message header is incorrect. Verify that the targeted diameter peer node is run with the correct realm and Origin-Host names.
DIAMETER_REALM_NOT_ SERVED	3003	The destination realm in the diameter message header is not recognized. Verify that the message realm name matches the realm configuration in Diameter Gateway.
DIAMETER_TOO_BUSY	3004	The request cannot be processed within the timeout period of Elastic Charging Server. Elastic Charging Server (the charging server nodes) may be overloaded. Verify that the charging server nodes have reached the point of saturation. Check the CPU usage on the charging server nodes.
DIAMETER_APPLICATION_ UNSUPPORTED	3007	The Diameter Gateway server received the diameter request with the Auth-Application-Id other than Gy, Sy, or Sh. Verify that the Auth-Application-Id in the diameter request is either Gy (4), Sy (16777302), or Sh attribute-value pair (16777217).
DIAMETER_INVALID_HDR_ BITS	3008	The diameter message header contains bits of unexpected byte size, or it is corrupted. Verify the diameter message header.
DIAMETER_INVALID_AVP_ BITS	3009	The diameter message contains the attribute-value pair (AVP) bits of unexpected byte size, or it is corrupted. Verify the diameter message AVP stream.

Transient-Failure Result Codes

Transient-failure result codes are returned to indicate that the request could not be processed at the time it was received. These requests may be processed in the future.

Table D–3 lists the transient-failure result codes.

Table D–3 Transient-Failure Result Codes

Result Code	Code	Notes
DIAMETER_END_USER_ SERVICE_DENIED	4010	The rating function failed with one of the following errors: ▪ MISSING_SYSTEM_ALTERATION_ FOR_DEBIT ▪ INCORRECT_IMPACTS_FOR_DEBIT & LIFECYCLE_VALIDATION_FAILED Check the **ecs.log** file for the rating function failure details.
DIAMETER_CREDIT_LIMIT_ REACHED	4012	The subscriber's credit floor or ceiling limit is breached. Update the subscriber's credit.

Permanent-Failure Result Codes

Permanent-failure result codes are returned to indicate that the requests will not be processed and that further requests from the diameter client will not be accepted for processing.

Table D–4 lists the permanent-failure result codes.

Table D–4 Permanent-Failure Result Codes

Result Code	Code	Notes
DIAMETER_AVP_ UNSUPPORTED	5001	The Diameter Gateway server received a message with an unknown AVP. Verify that the AVP is present in the **dictionary_main.xml** file.
DIAMETER_UNKNOWN_ SESSION_ID	5002	The Sy request contains an unknown Session-Id. Verify that the Session-Id is same as in the initial request.
DIAMETER_ AUTHORIZATION_REJECTED	5003	The Diameter Gateway authorization failed with the data in the request message AVP. Check the log files for more information on the authorization failure.
DIAMETER_INVALID_AVP_ VALUE	5004	The request AVP data cannot be converted to an expected PayloadItem type. Verify that the AVP type in the request is compatible with the PayloadItem type in ECE.
DIAMETER_MISSING_AVP	5005	The Diameter Gateway server received a request without the Gy, Sy, or Sh AVP. Verify that the request contains the required AVPs as per the 3GPP specifications.

Table D–4 (Cont.) Permanent-Failure Result Codes

Result Code	Code	Notes
DIAMETER_AVP_NOT_ ALLOWED	5008	A diameter message was received with an AVP that is not allowed as per the **dictionary_main.xml** file. Check for any AVP cardinality rules violations as defined in the **dictionary_ main.xml** file.
DIAMETER_AVP_OCCURS_ TOO_MANY_TIMES	5009	The AVP-to-ECE PayloadItem mapping violated the cardinality mapping rules per the request specification. Check for cardinality rule violations in the AVP-to-PayloadItem mapping.
DIAMETER_NO_COMMON_ APPLICATION	5010	The Diameter Gateway server received a Capabilities-Exchange-Request (CER) message requesting applications other than Gy, Sy, or Sh. Check the Diameter Gateway server connection.
DIAMETER_UNSUPPORTED_ VERSION	5011	The Diameter Gateway server received a request message with a version number that is not supported. Verify that the diameter client version is compatible with the Diameter Gateway server version.
DIAMETER_UNABLE_TO_ COMPLY	5012	An unexpected system error occurred on the Online Charging System (OCS). Check the ECE and Diameter Gateway log files for details about the error.
DIAMETER_INVALID_AVP_ LENGTH	5014	The AVP in the request message contains more or fewer bytes than expected by the Diameter Gateway server. Verify that the AVP data types are configured with the correct byte size.
DIAMETER_NO_COMMON_ SECURITY	5017	A CER message was processed without security mechanism. Verify that security is enabled in Diameter Gateway Communications Layer.
DIAMETER_USER_UNKNOWN	5030	ECE was not able to locate the subscriber ID in the diameter request. Verify that the subscriber exists in ECE.
DIAMETER_RATING_FAILED	5031	The Online Charging System rating function failed because it received one of the following messages: ■ INSUFFICIENT_RATED_QUANTITY ■ NO_QUALIFIED_CHARGE_OFFERS ■ NO_RATED_QUANTITY ■ ZERO_RUM_QUANTITY Check the **esc.log** file for the rating function failure.

Table D–4 (Cont.) Permanent-Failure Result Codes

Result Code	Code	Notes
DIAMETER_ERROR_ UNKNOWN_POLICY_ COUNTERS	5570	The Online Charging System does not recognize one or more policy counters specified in the diameter request. Verify that the ECE server is configured for tracking the policy counters requested by the diameter message.
DIAMETER_ERROR_SUBS_ DATA_ABSENT	5106	ECE received a request for some profile data that is unknown. Verify that ECE is configured to return the corresponding profile data for the diameter request.
DIAMETER_ERROR_USER_ UNKNOWN	5001	The policy request user is unknown to ECE. Verify that the subscriber exists in ECE.