Chapter 1: Introduction

This section contains the following topics:

Overview

CA Embedded Entitlements Manager (CA EEM) allows applications to share common access policy management, authentication, and authorization services.

Functions

CA EEM provides a set of security services. The following security services are available:

- Configuration services:
  - Registering and unregistering application instances
  - Administrative scoping of application administrators
  - Delegating administrative rights
  - Managing users and groups
- Administration security services:
  - Managing access, event, and obligation policies
  - Managing calendars
- Run-time security services:
  - Authenticating users
  - Authorizing access
  - Logging security events

Features

CA EEM consists of the following features:

General

- Policy isolation lets each registered application instance to use its own space for storing its application-specific data
- Run-time SDK available for Java, C++, and C#
- Administrative SDK available for Java, C++, and C#
- Command line interface support for administrative functions (insert/modify/remove objects):
  - XML export and import
  - Run-time checks
  - Migration tools
- Web interface support for standalone and launch-in-context access
- Secure HTTP communications
- Integrated with CA Security Command Center and CA Audit for security event management
- Integration with CA SiteMinder to retrieve user and group information from CA SiteMinder data store

Identity Management

- Shared global users and attributes for all applications
- Support for different modes for global users
  - Internal global users, complete with password policy management
  - External global users from LDAP directory servers
  - External global users from CA Identity Manager
- Integration with CA Identity Manager for role-based user provisioning and management
- Support for portable session export and import for single sign-on

Access Management

- Access management covers both Access Control Lists (ACLs) and business policies
- Policy language allows the use of user, session, environment, and resource attributes in making policy decisions
- Built-in administrative scoping of all objects
- Built-in support for delegated administration
- Built-in support for custom obligation checks requiring application-specific actions

- Local in-process evaluation of permission checks
- SDK and Web interface for defining access policies, ACLs, administrative scoping policies, and delegated authority

Client Access

You can access CA EEM Server through standard web and web services interfaces that provide third-party integration without requiring the client module. The interfaces are:

- HTML and iTechnology for configuration and administration
- iTechnology for delivering CA Audit events

iTechnology is a CA technology based on web standards such as HTTP, HTTPS, HTML, XML, and SSL. It provides a framework to create and deploy web services across the Internet.

Data Store Support

CA EEM supports identifying a single external user source, such as Microsoft Active Directory. CA EEM stores its configuration and policies in CA Directory regardless of where the user objects are stored.

Chapter 2: Installing on Windows

This section contains the following topics:

Installation Overview

The CA EEM installation on Windows operating environments consists of installing the following applications:

CA EEM Server

You can use CA EEM Server to define authorization policies on application resources using a web interface. The web-based administrative interface lets you manage identities and access policies. The existing security infrastructure is used to implement rules based on business logic, using resources and user attributes, defined in centralized user stores and other enterprise systems.

CA EEM Software Development Kit (SDK)

You can use the CA EEM SDK to embed identity-based security controls within applications. The SDK is comprised of libraries, java classes, header files, and a tutorial. You can use the SDK to implement CA EEM in any application. For more information on how to implement CA EEM using SDK, see the *Programming Guide*.

Each application installs separately and functions independently of the other.

Install Server

You can install CA EEM server using the installation wizard or the command line. Use the command line to install CA EEM in the silent mode and use the installation wizard for an interactive install.

JRE is no longer a minimum requirement to install and use CA EEM. You can install and use CA EEM with or without JRE. If you want to install CA EEM without JRE as a minimum requirement, you must skip the JRE selection path screen in the installation wizard. If you want to install CA EEM server in a silent mode without JRE, you must use the javahome parameter set to "none".

More Information

How to Skip Selecting JRE Path Screen in Installation Wizard (see page 7)
Wizard Setup Checklist (see page 6)
Install CA EEM Server in Silent Mode (see page 13)

Wizard Setup Checklist

While installing CA EEM server on Windows, you need the following information:

Note: You may have to restart your computer during the installation.

Field	Value
CA EEM Installation Path	Location on your computer where you intend to install CA EEM.
JRE Installation Path	Location of JRE installation on your computer. **Note**: If you want to install and use CA EEM without JRE, use the Skip button on the installation wizard.
EiamAdmin Password	The password associated with the CA EEM administrator EiamAdmin
Backup Directory	The location on your computer where you intend to back up the files from an earlier installation of CA EEM.

How to Skip Selecting JRE Path Screen in Installation Wizard

JRE is no longer a minimum requirement to install and use CA EEM. If you want to install CA EEM without JRE, you must do the following:

1. Set javahome parameter equal to "none".

   Note: If you set javahome parameter to "none", the installation wizard does not display the Java Path selection screen.

2. Install CA EEM using the installation wizard.

Set Javahome parameter

You must set the javahome parameter to the value "none" before you use the installation wizard to install CA EEM. Set the javahome parameter, from the command line as follows:

```
EEMServer_[version number]_win32.exe -s -a /z"javahome=None; "
```

Install Server Using Installation Wizard

The CA EEM Server installation wizard guides you through the installation process and provides you with options to define the installation parameters.

To install CA EEM Server

1. Do one of the following:

   - Open the Windows Explorer and double-click the install package EEMServer_<version number>_win32.exe on the target computer.
   - Enter the following command at the command prompt using installation parameters:

     ```
     EEMServer_8.4.231_win32.exe -s -a /z"eiampath=<Custom installation path for CA EEM>; etdirpath=<Custom installation path for CA Directory>; igpath=<Custom installation path for iGateway>; "
     ```

     You can provide a custom installation path using installation parameters. For more details about installation parameters, see Server Installation Parameters (see page 12).

     The installation wizard appears.

2. Follow the instructions on the installation wizard to complete the installation.

More Information:

How to Skip Selecting JRE Path Screen in Installation Wizard (see page 7)

Start Server

You must start the CA EEM Server to manage identities and access policies of registered applications.

To start using CA EEM Server

1. Do one of the following:
   - Enter the URL https://*hostname* or *ipaddress*:5250/spin/eiam in your browser. If you are on the CA EEM Server computer, specify http://l*ocalhost*:5250/spin/eiam.
   - Select Start, Programs, CA, Embedded Entitlements Manager, EEM UI on Windows operating environments.

     A login page appears.
2. Enter the following information in the login dialog:
   a. Select an application instance that you have registered from the Application drop-down. The default is <Global>. The default administrator username is EiamAdmin.

      Note: You can add other global users for login and set their usernames according to the preferences.

   b. Enter your password. This is the same password you specified during the installation of CA EEM Server for EiamAdmin.

   c. Select Remember my settings, if you want to log into the CA EEM Server with the same settings next time.
3. Click Log In.

   The CA EEM interface home page appears. For more information about how to use CA EEM Server, see the *Online Help*.

Activate Accessibility in CA EEM Server

The accessibility features in the CA EEM Server lets users, regardless of ability, to successfully use its products and supporting documentation to accomplish vital business tasks. When you activate accessibility, users can accomplish vital business tasks using only keyboard or using a screen reader.

To activate accessibility

1. Do one of the following:
   - Enter the URL https://*hostname* or *ipaddress*:5250/spin/eiam in your browser. If you are on the CA EEM Server computer, specify http://*localhost*:5250/spin/eiam.
   - Select Start, Programs, CA, Embedded Entitlements Manager, EEM UI on Windows operating environments.

     A login page appears.
2. Enter the following information in the login dialog:
   a. Select an application instance that you have registered from the Application drop-down. The default is <Global>. The default administrator username is EiamAdmin.

      Note: You can add other global users for login and set their usernames according to the preferences.

   b. Enter your password. This is the same password you specified during the installation of CA EEM Server for EiamAdmin.

   c. Select Remember my settings, if you want to log into the CA EEM Server with the same settings next time.
3. Click Activate Accessibility.

   Accessibility is activated in the CA EEM GUI.
4. Click Login.

   The CA EEM interface home page appears.

Upgrade Server

You can upgrade the existing installation of CA EEM Server to the current version.

To upgrade an existing installation of CA EEM Server

1. Run the EEMServer_<version number>_win32.exe on the target computer.
2. Depending on the version of CA EEM Server installed, one of the following occurs:
   - If the existing version of CA EEM Server is older than the version that is being installed, the installation wizard backs up the existing version and automatically upgrades to the newer version.
   - If the version that is being installed is older than the existing version, the installation wizard displays an error and quits the installation.
   - If the existing CA EEM Server version is same as the version that is being installed, the installer displays a message indicating that no upgrade is necessary. The installer exits after displaying the message.

Upgrading the CA EEM Server updates the following:

- CA EEM Server in \\CA\SharedComponents\iTechnology folder
- iGateway
- CA Directory

Also, all P12 certificates are migrated to PEM certificates.

More Information:

Wizard Setup Checklist (see page 6)

Remove Server

You can uninstall the CA EEM Server using the Add or Remove Programs of the Control Panel.

Note: You cannot remove CA EEM Server if applications are registered in CA EEM. You must unregister the applications before uninstalling CA EEM Server. For information about unregistering an application, see the *Online Help.*

Install SDK

The CA EEM SDK installation wizard guides you through the installation process.

To install the CA EEM SDK

1. Open the Windows Explorer and double-click the install package EEMSDK_<version number>_win32.exe, or run the installation file from the command prompt.

   The installation wizard appears.

2. Click I Agree to accept the Terms and Conditions.

   Note: The I Agree button is enabled only after you read, or scroll the text of the Terms and Conditions.

   The Choose Destination Location dialog appears. By default, the installation wizard installs the CA EEM SDK in the following location: C:\Program Files\CA\Embedded IAM SDK

3. Click Next.

   Or

   Click Browse and select a directory on your computer where the CA EEM SDK must be installed, and click Next.

   This starts installing the CA EEM SDK.

4. Click Finish.

   CA EEM SDK is installed.

   Note: An environmental variable %EIAM_SDK% is created during the installation to point to the installation path. Use this variable in the explorer path to open the installation folder.

Start SDK

To start CA EEM SDK click Start, Programs, CA, Embedded Entitlements Manager, EEM SDK.

The CA EEM SDK documentation window appears.

Remove SDK

You can uninstall the CA EEM SDK using the Add or Remove Programs of the Control Panel.

Server Installation Parameters

While installing CA EEM on Windows, you must collect information on the following command line parameters:

-eiampath

Specifies the path where the CA EEM Server will be installed. The default is C:\Program Files\CA\SharedComponents\Embedded IAM.

-etdirpath [path]

Specifies the path where CA Directory will be installed. The default is C:\Program Files\CA\Directory.

-igpath [path]

Specifies the path where iGateway will be installed. The default is C:\Program Files\CA\SharedComponents\iTechnology. An environmental variable %IGW_LOC% is created during the installation to point to the iGateway installation path.

-backupdir

Specifies the location where the data from the existing installation is backed up.

-capkiinstalldir

Specifies the installation folder path for the CAPKI module. The default is C:\Program Files\CA\SC\CAPKI.

-javahome [directory]

Sets the JAVA_HOME variable to [directory] when calling the iGateway installer. The CA EEM installer prompts you to set the variable even if it already set. This parameter does not have a default value.

Note: If you want to install CA EEM without java, you must set javahome to none.

CA EEM uses the following parameters during CA Directory installation. You can configure the parameters based on your needs.

Important! Before customizing the default port numbers ensure that no other services are configured to use the same ports.

-dsaport

Specifies the port that the dsa uses to listen to any requests directed at it.

Default: 509

-routerport

Specifies the port that the dsa uses to connect to the router dsa. A router DSA has no local data and no datastore. It can only route traffic to other DSAs.

Default: 1684

-dxdbsize

Specifies the maximum size of the datastore for CA EEM.

Default: 256 MB

-dxuser

Specifies a non-dsa user who can install, administer, and uninstall CA Directory. The dxuser can be a local system user or a network user.

Note: If you have installed CA Directory using a local system user as the dxuser, then during uninstallation that local system user may be deleted. So, if you are using a local system user as a dxuser to install CA Directory, ensure that the user is not setup to run any other programs.

Default: dsa

Note: On a computer with Microsoft Windows Server 2003, the maximum length of the string that you can use at the command prompt is 8,191 characters. With Microsoft Windows 2000, the maximum length of the string that you can use at the command prompt is 2,047 characters. For more information about InstallShield command length, see the *Release Notes*.

Install CA EEM Server in Silent Mode

Installing the CA EEM Server in silent mode requires two tasks:

1. Create the response file.
2. Run the command specifying the response file.

   A log file eiaminstall.log is created during the silent installation to record any installation errors.
3. Run the CA EEM installer after the restart to continue the installation.

Note: If you install CA EEM Server silently, you can also remove it silently.

Configure the Response File

CA EEM creates a Response File with default values. You can use the Response File with the default values or edit the Response File to silently install the CA EEM server.

Follow these steps:

1. Navigate to the location *EEMServer_home*\CA\SC\EmbeddedEntitlementManager.
2. Copy the response.properties file to the <install directory of target server>.
3. (Optional) Edit the response.properties file, and save the changes.

   The Response File is configured.

Run the Command Specifying the Response File

The following examples describe the commands to do a silent installation:

- To install the CA EEM Server in the silent mode, enter the following command at the command prompt:

  `EEMServer_<version number>_win32.exe -s -a /s /f1"pathname of response file"`

  Example:

  `EEMServer_8.4.231_win32.exe -s -a /s /f1"c:\resp.iss"`

- To create an installation log file during the silent installation, enter the following command at the command prompt:

  `EEMServer_<version number>_win32.exe -s -a /s /v"/qn /L*v <path to create log file>" /f1"pathname of response file"`

  Example:

  ```
  EEMServer_8.4.231_win32.exe -s -a /s /v"/qn /L*v c:\install.txt"
  /f1"c:\resp.iss"
  ```

Note: You can provide the install parameters with the install script. For more details about the parameters, see Server Installation Parameters (see page 12).

Remove CA EEM Server in Silent Mode

You must use a response file created from the same build of CA EEM Server to successfully remove the product. To remove CA EEM Server in the silent mode, enter the following command at the command prompt:

```
EEMServer_<version number>_win32.exe -s -a /s /f1"pathname of response file"
/z"uninstall"
```

Example:

```
EEMServer_8.4.231_win32.exe -s -a /s /f1"c:\resp.iss" /z"uninstall"
```

This removes CA EEM Server in silent mode.

Note: You cannot remove CA EEM Server if applications are registered in CA EEM. You must unregister the applications before uninstalling CA EEM Server. For information about unregistering an application, see the *Online Help.*

Chapter 3: Installing on Linux and UNIX

This section contains the following topics:

Installation Overview

The CA EEM installation on Linux and UNIX operating environments consists of installing the following applications:

CA EEM Server

You can use CA EEM Server to define authorization policies on application resources using a web interface. The web-based administrative interface lets you manage identities and access policies. The existing security infrastructure is used to implement rules based on business logic, using resources and user attributes, defined in centralized user stores and other enterprise systems.

CA EEM Software Development Kit (SDK)

You can use the CA EEM SDK to embed identity-based security controls within applications. The SDK is comprised of libraries, java classes, header files, and a tutorial. You can use the SDK to implement CA EEM in any application. For more information on how to implement CA EEM using SDK, see the *Programming Guide*.

Each application installs separately and functions independently of the other.

Install Server

CA EEM Server for Linux and UNIX uses a self-extracting shell script that guides you through the installation process. During the installation process, the script displays the license information and prompts for installation parameters. After the installation parameters are entered, the installation begins.

To install CA EEM Server for Linux and UNIX

1. Run the installation script EEMServer_<version number>_<operating system>.sh on the target computer.

   Example:

   ```
   sh EEMServer_8.4.231_sunos.sh
   ```

   The file is decompressed and the installation begins.

2. Enter Y to accept the Terms and Conditions of the license agreement (or N to decline and abort the installation).

   The script prompts for the installation parameters.

3. Enter the installation parameters.

   Note: For more information on the available installation parameters, see Server Installation Script Parameters (see page 21).

   Example:

   a. Enter the installation path for the CA EEM Server (or accept the default).

   A confirmation screen appears with the installation parameter values you entered.

4. Enter Y to continue the installation, if the information on the confirmation screen is correct (if you Enter N, the installer exits.).

5. Enter the EiamAdmin password.

   Note: The default administrator username is EiamAdmin.

   The installation procedure depends on the command line parameters and the type of CA EEM Server package being installed.

   The installer script completes the installation of CA EEM Server on your computer.

Upgrade Server

You can upgrade the existing installation of CA EEM Server to the current version.

To upgrade an existing installation of CA EEM Server

1. Run the EEMServer_<version number>_<operating system>.sh on the target computer.
2. Depending on the version of CA EEM Server installed, one of the following occurs:
   - If the existing version of CA EEM Server is older than the version that is being installed, the installation wizard backs up the existing version and automatically upgrades to the newer version.
   - If the version that is being installed is older than the existing version, the installation wizard displays an error and quits the installation.
   - If the existing CA EEM Server version is same as the version that is being installed, the installer displays a message indicating that no upgrade is necessary. The installer exits after displaying the message.

Upgrading the CA EEM Server updates the following:

- CA EEM Server in \\CA\SharedComponents\iTechnology folder
- iGateway
- CA Directory

Also, all P12 certificates are migrated to PEM certificates.

More Information:

Wizard Setup Checklist (see page 6)

Remove Server

To remove CA EEM Server, run the eiamuninstall.sh script from the installation directory.

Note: You cannot remove CA EEM Server if applications are registered in CA EEM. You must unregister the applications before uninstalling CA EEM Server. For information about unregistering an application, see the *Online Help.*

Install SDK

CA EEM SDK for Linux and UNIX uses a self-extracting shell script that guides you through the installation process. During the installation process, the script displays the license information and prompts for the installation parameters. After the installation parameters are entered, the installation begins.

To install CA EEM SDK for Linux and UNIX

1. Run the installation script EEMSDK_<version number>_<operating system>.sh on the target computer.

   Example:

   ```
   sh EEMSDK_8.4.231_sunos.sh
   ```

   The file is decompressed and installation begins.

2. Enter Y to accept the Terms and Conditions of the license agreement (or N to decline and abort the installation).
3. Enter the installation path for the CA EEM SDK (or accept the default).
4. Select install product.

   CA EEM SDK is installed on your computer.

   Note: An environmental variable $EIAM_SDK is created during the installation to point to the installation path. Use this variable in the explorer path to open the installation folder.

Start CA EEM SDK

To start CA EEM SDK, point your Web browser at $EIAM_SDK/Doc/Bookshelf/Bookshelf.html

Remove SDK

You can remove CA EEM SDK from Linux and UNIX operating systems.

To remove CA EEM SDK

1. Run the installation script EEMSDK_<version number>_<operating system>.sh on the target computer.

   Example:

   ```
   sh EEMSDK_8.4.231_sunos.sh
   ```

   The file is decompressed.

2. Select uninstall/remove product.

   The installation script removes CA EEM SDK on your computer.

Server Installation Script Parameters

While installing CA EEM, you must collect information on the following command line parameters that the script prompts for during the installation.

The script accepts the following command line parameters:

-backupdir

Specifies the location where the data from the existing installation is backed up.

-capkiinstalldir

Specifies the installation folder path for the CAPKI module.

Default: /opt/CA/SharedComponents/CAPKI

CA EEM uses the following parameters during CA Directory installation. You can configure the parameters based on your needs.

Important! Before customizing the default port numbers ensure that no other services are configured to use the same ports.

-dsaport

Specifies the port that the dsa uses to listen to any requests directed at it.

Default: 509

-routerport

Specifies the port that the dsa uses to connect to the router dsa. A router DSA has no local data and no datastore. It can only route traffic to other DSAs.

Default: 1684

-dxdbsize

Specifies the maximum size of the datastore for CA EEM.

Default: 256 MB

-dxuser

Specifies a non-dsa user who can install, administer, and uninstall CA Directory. The dxuser can be a local system user or a network user.

Note: If you have installed CA Directory using a local system user as the dxuser, then during uninstallation that local system user may be deleted. So, if you are using a local system user as a dxuser to install CA Directory, ensure that the user is not setup to run any other programs.

Default: dsa

-dxgroup

Specifies the group name of the dxuser.

Default: etrdir

-dsapw

Sets the password for the DSA account.

-eiamadminpw [password]

Sets the EiamAdmin password to [password]

-eiampath

Sets the path where the CA EEM Server is installed.

Default: /opt/CA/SharedComponents/EmbeddedIAM.

-etdirpath [path]

Sets the path where CA Directory is installed.

Default: /opt/CA/Directory

-igwuser

Specifies the user account under which the iGateway is installed. Only this user can administer and manage iGateway.

Default: root

-igwgroup

Specifies the group for the igwuser.

Default: root

-igpath [directory]

Sets the iGateway path. The path must be a fully qualified path, such as -iisystem. The default is /opt/CA/SharedComponents/iTechnology. An environmental variable $IGW_LOC is created during the installation to point to the installation path. Use this variable to open the installation folder.

-javahome [directory]

Sets the JAVA_HOME. This parameter defaults to the contents of the JAVA_HOME environment variable and is prompted for only if $JAVA_HOME is not set.

Note: If you want to install CA EEM without java, you must set javahome=none. This is not applicable to HP-UX.

-logfile [filename]

Causes the installer to write log information to [filename], defaults to /tmp/eiam-install.log.

-nocalm

Specifies that [set the CALM variable for your book] is not installed. If you specify this option, the Manage Reports tab is not available from the CA EEM admin GUI.

-silent

Runs the installation in silent mode. If a required parameter is not specified on the command line, the installation aborts and prints an appropriate message. It does not make any changes to the system unless all needed parameters are specified.

-tempdir [directory]

Specifies the directory to use for temp file storage. Default is /tmp/eiam_temp. This must be a fully-qualified path, and must be in its own subdirectory. This script uses rm -rf to remove the directory you specify hereupon script completion.

Install Server in Silent Mode

To install CA EEM Server in the silent mode on Linux or UNIX enter the following command at the command prompt:

```
sh EEMServer_<version number>_<operating system>.sh -silent -eiamadminpw password
-javahome directory
```

Example: The following command for Sun operating environments includes the minimum required parameters:

```
sh EEMServer_8.4.231_sunos.sh -silent -eiamadminpw password -javahome directory
```

You can specify additional installation parameters. Most install parameters have defaults. For more information about Script Parameters, see Server Installation Script Parameters (see page 21).

The file is decompressed and the installation begins.

Remove CA EEM Server in Silent Mode

To remove CA EEM Server, run eiamuninstall.sh -silent from the installation directory.

Note: You cannot remove CA EEM Server if any applications are registered. You must unregister all applications before the uninstallation successfully completes. For information about unregistering an application, see the *Online Help*.

Chapter 4: Configuring CA EEM SDK

This section contains the following topics:

CA EEM SDK Initialization

The following topics explain how to initialize CA EEM SDK. The eiam.config file controls the CA EEM SDK configuration.

More information:

About the eiam.config File

Use the eiam.config file to control CA EEM SDK configuration data such as:

- Cyclic buffer
- Logger configuration file
- SAF folder for storing audit files
- FIPS compatible mode
- SafeContext-related information

The eiam.config file consists of the following configurable parameters:

CyclicBuffer size

Specifies the number of log messages contained in a cyclic buffer. The cyclic buffer stores the specified number of latest log messages in the memory. As the buffer reaches the specified size, a new log message replaces the oldest log message in the buffer. If the application crashes, you can recover the latest log messages from the core.

Default: 500

Minimum: 0

Maximum: 1000

Note: This parameter is valid only for the CA EEM C++ SDK.

enable

Specifies if the cyclic buffer is enabled. If enabled is set to false, the cyclic buffer is disabled. So, you need not specify values of the parameters CyclicBuffer size, dump, and file.

Value: [*true*|*false*]

Default: true

Important! Cyclic buffer is enabled by default. If you enable the cyclic buffer, performance of CA EEM is affected.

dump

Specifies if the contents of cyclic buffer are written to a file if the eiam.config file is modified or updated.

Value: [*true*|*false*]

Default: false

file

Specifies filename of the dump file. If dump is set to false, the log messages are not written to a dump file. The file extension of file is .log.

LoggerConfiguration file

Specifies absolute path of the logger configuration files for CA EEM Java, C# SDK, and C++ SDKs. The CA EEM logging information is stored in the logger configuration files. eiam.log4cxx.config, eiam.log4net.config, and eiam.log4j.config are the logger configuration files for CA EEM C++ SDK, CA EEM C# SDK, and CA EEM Java SDK.

audit

SAF folder where audit files are stored for processing.

Note: For more information about SAF Directory, see the Reliable Event Delivery section in the *Programming Guide*.

Network sockettimeout

Specifies the socket timeout in milliseconds.

Default: 120000 (120 seconds)

Note: This parameter is valid only for the CA EEM C++ SDK and CA EEM Java SDK.

<SDK type ="C#">

Specifies the FIPS mode settings for C# SDK.

FIPSMode

Specifies the FIPS mode for CA EEM SDK. For FIPS-only mode, set the value to On.

Value: [Off|On]

Default: Off

digestAlgorithm

Specifies the cryptographic algorithm used to sign server requests. In FIPS mode, CA EEM C# SDK uses SHA1 as the digest algorithm by default. If FIPS mode is disabled, CA EEM C# SDK uses MD5 as the digest algorithm. MD5 is not supported in FIPS-only mode.

Value: [MD5|SHA1]

Default: MD5 for non-FIPS mode.

Note: In FIPS-only mode, CA EEM C# SDK supports only SHA1 as the digest algorithm.

<SDK type ="Java">

Specifies the FIPS mode settings for Java SDK.

FIPSMode

Specifies the FIPS mode for CA EEM SDK. For FIPS-only mode, set the value to On.

Value: [Off|On]

Default: Off

JCEProvider

Specify the Java Cryptography Extension (JCE) provider to use in the FIPS-only mode.

digestAlgorithm

Specifies the cryptographic algorithm used to sign server requests. For CA EEM SDK enabled in FIPS-only mode, use SHA1 as the digestAlgorithm. FIPS does not support MD5. If FIPS-only mode is disabled, the server requests are signed using MD5.

Value: MD5/SHA1/SHA256/SHA384/SHA512

Default: SHA1 for FIPS-only mode and MD5 for non-FIPs mode.

logLevel

Specifies the log level.

Value: [Error|Warning|Trace|Nolevel]

logToFile

Specifies if the log messages must be stored in a file.

Value: [True|False]

Default: False

logFile

Specifies the absolute path to the log file. This parameter is valid only if logToFile is set to True.

maxLogSize

Specifies the maximum size of the log file in MB.

<SDK type ="C++">

Specifies the FIPS mode settings for C++ SDK.

FIPSMode

Specifies the FIPS mode for CA EEM SDK. For FIPS-only mode, set the value to On.

Value: [Off|On]

Default: Off

etpkiCryptoLib

Specifies the installation path for the etpki libraries.

secureProtocol

Specifies the protocol that the CA EEM SDK uses to communicate with CA EEM Server.

Default: SSLV23

Values: SSLV23 / SSLV3 / TLSV1

Note: FIPS-only communication mode supports only TLSV1. Communication fails if you use SSLV2 or SSLV3 when FIPS mode is set to True.

digestAlgorithm

Specifies the cryptographic algorithm used to sign server requests. For CA EEM SDK enabled in FIPS mode, use SHA1 as the digestAlgorithm. FIPS does not support MD5. If FIPS mode is disabled, the server requests are signed using MD5.

Value: MD5/SHA1/SHA256/SHA384/SHA512

Default: MD5 for non-FIPS and SHA1 for FIPS-only mode. If the tag is empty, CA EEM uses the default values.

logLevel

Specifies the log level.

Value: [Error|Warning|Trace|Nolevel]

logToFile

Specifies if the log messages must stored in a file.

Value: [True|False]

Default: False

logFile

Specifies the absolute path to the log file. This parameter is valid only if logToFile is set to True.

maxLogSize

Specifies the maximum size of the log file in MB.

<SafeContext>

Specifies the information required to generate a SafeContext using the SafeContextFactory method.

Note: You can include more than one SafeContext tag in the eiam.config file. However, the refid must be unique for each SafeContext tag.

refid

Specifies the reference ID for a SafeContext tag. This ID must be unique. The SafeContextFactory uses the reference ID to pick the information required to generate a SafeContext.

Backend

Specifies the hostname of the CA EEM Server.

Application

Specifies the name of the application instance for which the SafeContext is generated. If the application name is not specified, SafeContextFactory attaches to the global application.

Locale

Specifies the locale.

Authentication Type

Specifies the authentication that the SafeContextFactory uses to attach to an application. The following are the supported authentication types:

- password-based authentication; use the UserAuth tags for this authentication
- PEM certificates; use the <Certificate type="pem"> for this authentication
- P12 certificates; use the <Certificate type="p12"> for this authentication
- PKCS#11 certificates; use the <Certificate type="p11"> for this authentication

Note: Use only one authentication type with a SafeContext tag.

UserAuth

Specify password-based authentication method.

Username

Specifies the username of the administrator needed to attach to an application instance or global instance.

Password

Specifies the munged password needed to authenticate the administrator.

PEM certificates

Specifies the details required for a PEM certificate-based authentication.

CertURI

Specifies the path including the certificate filename.

KeyURI

Specifies the path including the key file.

KeyPW

Specifies the munged password required to read the certificate file. This tag is valid only for the CA EEM C++ SDK. In FIPS-only mode, this tag must be blank.

PKCS#11 certificates

Specifies the details required for a P11 certificate-based authentication.

Provider

Specifies the path to the encryption libraries.

Userpin

Specifies the userpin to use with the PKCS#11 device.

ID

Specifies the ID of the PKCS#11 certificate.

P12 certificates

Specifies the details required for a P12 certificate-based authentication.

CertURI

Specifies the path including the P12 certificate filename.

KeyPW

Specifies the munged password to read or write to the certificate file.

Example of a eiam.config File for Java Application

The following is an example of the eiam.config file:

```
<EiamConfiguration>
  <!-- Absolute file path for logger configuration, For Java use:-
file="eiam.log4j.config" -->
  <LoggerConfiguration file="eiam.log4j.config"/>
  <!-- Absolute folder path for SAF folder where audit files will be stored for
processing-->
  <Saf directory="audit"/>
   <!-- Socket timeout in milli seconds. Default value is 2 mins -->
  <Network sockettimeout="120000"/>
<SDK type="Java">
        <iTechSDK>
                <FIPSMode>true</FIPSMode>
                <JCEProvider>JsafeJCE</JCEProvider>
                <Security>
                        <digestAlgorithm>SHA1</digestAlgorithm>
                </Security>
                <Debug>
                        <logLevel>trace</logLevel>
                </Debug>
        </iTechSDK>
  </SDK>
<!-- configuration to create SafeContext instance from SafeContextFactory -->
<SafeContext refid="RBC_Hospital" version="1.0">
        <!-- EEM server hostname -->
        <Backend>eiamServer</Backend>

        <!-- application instance to attach to (leave empty for global) -->
        <Application>RBC_Hospital</Application>

        <!-- locale -->
        <!-- java:format language-country-variant -->
        <Locale>en-us<\Locale>

        <!-- possible values for type are certificate/password/native -->
        <Authentication type="">
        <!-- input for certificate based authentication (PEM) keyPW is valid only for
C++
                <Certificate type="pem">
                        <CertURI>appcert.cer</CertURI>
                        <KeyURI>appcert.key</KeyURI>
                        <KeyPW></KeyPW/>
                </Certificate>
        </Authentication>
</SafeContext>
</EiamConfiguration>
```

Enable iTechnology SDK Logging

You can enable iTechnology SDK logging only for the CA EEM C++ SDK and the CA EEM Java SDK. For CA EEM C# SDK, use the logger configuration file.

To enable iTechnology SDK logging, open the eiam.config file and edit the following tags:

- logLevel
- logToFile
- logFile
- maxLogSize

For CA EEM Java SDK, edit the previously mentioned tags in the <SDK type ="Java"> section. For CA EEM C++ SDK, edit the tags mentioned in the <SDK type ="C++"> section.

More information:

About the eiam.config File (see page 26)

Before You Configure CA EEM Java SDK in FIPS-only Mode

To configure CA EEM Java SDK in FIPS-only mode, do the following tasks:

1. Configure JRE to use third-party Java Cryptography Extension (JCE) libraries.
2. Add the Crypto-J libraries as a JCE provider in the Java.security file.

   Note: For more information about how to configure JRE with JCE, see the respective JCE documentation.
3. Enable FIPS-only mode in the eiam.config file.

Configure CA EEM Java SDK in FIPS-only Mode

When you configure CA EEM SDK in FIPS-only mode, CA EEM uses FIPS 140-2 compliant cryptographic libraries to encrypt and decrypt sensitive data.

To configure CA EEM Java SDK in FIPS-only mode

1. Open eiam.config file and edit the following tags <SDK type="Java"> section:
   - FIPSMode
   - JCEProvider
   - digestAlgorithm
2. Save and close the eiam.config file.
3. Restart your application.

   CA EEM Java SDK is configured in FIPS-only mode.

More information:

About the eiam.config File (see page 26)

Configure CA EEM C++ SDK in FIPS-only Mode

When you configure CA EEM SDK in FIPS-only mode, CA EEM uses FIPS 140-2 compliant cryptographic libraries to encrypt and decrypt sensitive data.

To configure CA EEM C++ SDK in FIPS-only mode

1. Open eiam.config file and edit the following tags in the <SDK type="C++"> section.
   - FIPSMode
   - etpkiCryptoLib
   - secureProtocol
   - digestAlgorithm
2. Save and close the eiam.config file.
3. Restart your application.

   CA EEM C++ SDK is configured in FIPS-only mode.

More information:

About the eiam.config File (see page 26)

Configure CA EEM C# SDK in FIPS-only Mode

When you configure the CA EEM C# SDK in FIPS-only mode, CA EEM uses only FIPS 140-2 compliant cryptographic libraries to encrypt and decrypt sensitive data.

Note: CA EEM C# SDK does not support P11 certificates.

To configure CA EEM C# SDK in FIPS-only mode

1. Open eiam.config file and edit the following tags in the <SDK type="C#"> section:
   - FIPSMode
   - digestAlgorithm
2. Save and close the eiam.config file.
3. Restart your application.

   CA EEM C# SDK is configured in FIPS-only mode.

More information:

About the eiam.config File (see page 26)

Set SafeContext Information

The <SafeContext> tag in the eiam.config file contains information required to generate a SafeContext using the SafeContextFactory class. Each SafeContext tag in the eiam.config file is identified using a unique refID tag. To generate a SafeContext, you must pass this refID to the SafeContextFactory. Following are the benefits of specifying the SafeContext-related information in the eiam.config file:

To set SafeContext-related information:

1. Open eiam.config file and edit the <SafeContext> section to set the following tags:
   - refID
   - Backend
   - Application
   - Locale
   - Authentication Type
2. Save and close the eiam.config file.

Initialize CA EEM Java SDK

Configure CA EEM SDK using the SafeConfigurator class.

Note: You must configure eiam.config before you configure CA EEM SDK. If you do not configure the eiam.config file, the CA EEM SDK is initialized with the following default configuration:

- non-FIPS mode
- Logging is set to error and only console logging is enabled
- SAF location is disabled

To initialize CA EEM SDK, perform the following process:

1. Include the following API in your code to initialize CA EEM SDK during application startup:

   ```
   SafeConfigurator.getInstance().init(filename);
   ```

   Where

   filename

   Specifies the absolute path of the eiam.config file that you have defined for your application.

   Note: All the CA EEM SDK operations after this line are logged based on tracing levels of logging in the logger configuration.

2. Include the following API in your code during shutdown of your application:

   ```
   m_config.term();
   ```

   Note: For more information about the SafeConfigurator class, see the *Programming Guide*.

More information:

About the eiam.config File (see page 26)
About the Logger Configuration Files (see page 79)

Initialize CA EEM C++ SDK

Configure CA EEM SDK using the Safe::Configurator class.

Note: Configure the eiam.config before you configure CA EEM SDK. If you do not configure the eiam.config file, the CA EEM SDK is initialized with the following default configuration:

- non-FIPS mode
- Logging is set to error and only console logging is enabled
- SAF location is disabled
- CyclicBuffer is set to True

To initialize CA EEM SDK, perform the following process:

1. Include the following API in your code to initialize CA EEM SDK during application startup:

   ```
   Safe::Configurator::getInstance()->init(filename);
   ```

   Where

   filename

   Specifies the absolute path of the eiam.config file that you have defined for your application.

   Note: If you do not mention the filename, CA EEM SDK is initialized with the default values.

2. Include the following API in your code during shutdown of your application:

   ```
   Safe::Configurator::getInstance()->term();
   ```

   Note: For more information about the SafeConfigurator class, see the *Programming Guide*.

Important! If there are any invalid attributes in the eiam.config file, the CA EEM SDK aborts.

More information:

About the eiam.config File (see page 26)
About the Logger Configuration Files (see page 79)

Initialize CA EEM C# SDK

Configure CA EEM SDK using the SafeConfigurator class. To configure CA EEM SDK, perform the following process:

Note: Configure the eiam.config file before you configure CA EEM SDK. If you do not configure the eiam.config file, the CA EEM SDK is initialized with the following default configuration:

- non-FIPS mode
- Logging is set to error and only console logging is enabled
- SAF location is disabled

To initialize CA EEM SDK, perform the following process:

1. Include the following API in your code to initialize CA EEM SDK during application startup:

   ```
   SafeConfigurator.getInstance().Init(filename);
   ```

   Where

   filename

   Specifies the absolute path of the eiam.config file that you have defined for your application.

   Note: If you do not mention the filename, CA EEM SDK is initialized with the default values.

2. Include the following API in your code during shutdown of your application:

   ```
   SafeConfigurator.getInstance().term();
   ```

   Note: For more information about the SafeConfigurator class, see the *Programming Guide*.

More information:

About the eiam.config File (see page 26)
About the Logger Configuration Files (see page 79)

Package CA EEM Java SDK with Your Applications

To package CA EEM Java SDK, do the following:

1. Update the ClassPath with references to the required jar files from the following location:

   Windows: %EIAM_SDK%\jars

   UNIX and Linux: $EIAM_SDK/jars

2. Update your installer to package and deploy the required jar files and configuration files.

   Note: For more information about the required jars and configuration files, see the Required Files (see page 40) topic.

Required Files

Package the following files in your application:

- safe.jar
- servletapi-2.3.jar
- xml-apis.jar
- commons-codec-1.3.jar
- commons-logging-1.1.1.jar
- commons-logging-api-1.1.1.jar
- httpclient-4.0.jar
- httpcore-4.0.1.jar
- not-yet-commons-ssl-0.3.10.jar
- xercesImpl-2.9.1.jar
- xml-apis-2.9.1.jar

The following additional jars are required only if you use JDK 1.5:

- activation.jar
- jaxb-api.jar
- jaxb-impl.jar
- jsr173_1.0_api.jar

Package the following configuration files:

- eiam.config
- eiam.log4j.config

Package CA EEM C++ SDK with Your Applications

To package CA EEM C++ SDK, do the following:

1. Update your build script to search a directory for include files and lib path from the makeflags.mk file in the CA EEM SDK folder.
2. Update your installer to package and deploy the required dlls, include files, and configuration files.

   Note: For more information about the required dlls and configuration files, see the Required Files (see page 42) topic.
3. Include the CAPKI installer set up with your installer. Install the CAPKI installer with your product on the target system. The CAPKI installer is in following location:

   - **Windows**:%EIAM_SDK%\capki
   - **UNIX and Linux:** $EIAM_SDK/CAPKI

   Note: If you run Safex on a computer that is different from the computer where the CA EEM Server is installed, install CAPKI with Safex.

Required Files

Package the following dlls in your application:

Windows

Visual Studio 2003

%EIAM_SDK%\lib\win32 or $EIAM_SDK/lib/win32

Visual Studio 2005

%EIAM_SDK%\lib\win32_80 or $EIAM_SDK/lib/win32_80

Visual Studio 2008

%EIAM_SDK%\lib\win32_90 or $EIAM_SDK/lib/win32_90

UNIX platforms

- $EIAM_SDK/lib/<osname>

Package the following configuration files:

- eiam.config
- eiam.log4cxx.config

Include the following header files in your code when compiling your application:

Windows

%EIAM_SDK%\include

UNIX platforms

$EIAM_SDK/include

Package CA EEM C# SDK with Your Applications

To package CA EEM C# SDK, do the following:

1. Add the dlls from the following folder as referenced assemblies when building your application:

   ```
   %EIAM_SDK%\lib\csharp
   ```

2. Update your installer to package and deploy the referenced dlls, the eiam.config file, and the eiam.log4net.config file.

   Note: For more information about the reference dlls and configuration files, see the Required Files (see page 43) topic.

Required Files

Package and deploy the following referenced dlls in your application:

- log4net.dll
- CPoz.dll
- iclient.dll
- CsharpSDK.dll

Package and deploy the following configuration files:

- eiam.config
- eiam.log4net.config

Chapter 5: FIPS 140-2 Support

This section contains the following topics:

FIPS 140-2 Overview

The Federal Information Processing Standards (FIPS) 140-2 publication specifies the requirements for using cryptographic algorithms within a security system protecting sensitive, unclassified data. CA EEM Server embeds Crypto-C ME v2.0 cryptographic library from RSA, which has been validated as meeting the FIPS 140-2 *Security Requirements for Cryptographic Modules.* The validation certificate number for this module is 608.

CA EEM Java SDK uses a FIPS-compliant version of the BSAFE Crypto-J 4.0 cryptographic library from RSA. CA EEM C++ SDK embeds ETPKI 4.1.x, which uses RSA cryptography libraries.

CA EEM can operate in a non-FIPS mode or in a FIPS-only mode. The cryptographic boundaries, that is, the way CA EEM applies encryption, are the same in both modes, but the algorithms are different.

Computer products that use FIPS 140-2 accredited cryptographic modules in their FIPS-accredited mode can only use FIPS approved security functions such as AES (Advanced Encryption Algorithm), SHA-1 (Secure Hash Algorithm), and higher level protocols such as TLS v1.0 as explicitly allowed in the FIPS 140-2 standard and implementation guides.

In FIPS-only mode, CA EEM uses the following algorithms:

- SHA1 as the default digest algorithm to encrypt passwords and sign server requests. You can use any of the following algorithms in FIPS-only mode:
  - SHA1
  - SHA256
  - SHA384
  - SHA512
- TLS v1.0 for communication with external LDAP directories if the LDAP connection is over TLS.

Supported Security Modes in CA EEM

CA EEM supports the following modes of operation.

non-FIPS

This mode uses non-FIPS compliant techniques for cryptography. In this mode MD5 is the default algorithm that is used to encrypt and decrypt sensitive data. The CA EEM installer always install the CA EEM Server in a non-FIPS mode. In this mode, the CA EEM Server is backward compatible with the CA EEM clients. For example, you can use the CA EEM r8.4 SDK to connect to a CA EEM r8.4 SP3 Server.

FIPS-only

This mode uses only FIPS-compliant techniques for cryptography. This mode is not compatible with clients running in non-FIPS mode. You must use only CA EEM r8.4 SP3 SDK FIPS-only clients with CA EEM r8.4 SP3 Servers or later running in FIPS-only mode. In this mode, the CA EEM Server supports the following algorithms to encrypt and decrypt data:

- SHA1
- SHA256
- SHA384
- SHA512
- TLS v1.0 for communication with external LDAP directories if the LDAP connection is over TLS.

Note: SHA1 is the default algorithm.

How to Configure CA EEM Server in FIPS-only Mode

In FIPS-only mode, CA EEM must be configured to use FIPS-compliant algorithms. CA EEM Server and CA EEM SDK clients can communicate only if both are configured in the FIPS-only mode. Similarly, CA EEM Server in a FIPS-only mode can communicate only with an LDAP directory configured to use FIPS-compliant algorithms. To configure the CA EEM environment in a FIPS-only mode, do the following:

- Verify prerequisites for configuring CA EEM Server in FIPS-only mode
- Configure CA EEM Server in FIPS-only mode

More information:

Communication Between CA EEM Server and External LDAP Directories (see page 49)

Prerequisites for Configuring CA EEM Server in FIPS-only Mode

The following are the prerequisites for configuring the CA EEM Server in FIPS-only mode:

- Verify that the other CA products using iGateway such as CA ITM, CA ELM, and so on, are in FIPS-only mode. iGateway cannot be initialized both in FIPS-only mode and non-FIPS mode. When iGateway is initialized in FIPS-only mode, all products using iGateway must be in FIPS-only mode. Open the iGateway.conf file and verify the value for the following tag:

  `FIPSMode`

  If the value of this tag is set to False, it means that product using iGateway is in non-FIPS mode. Based on the existing configuration of iGateway decide appropriately if you want to enable CA EEM in FIPS-only mode.

- Verify the spindle versions used by the other CA products, open the spin.conf file and note the value for the following tags: <Spindle Name> and <version>. Verify with the respective product documentation if these versions are FIPS-compatible.

  Note: The iGateway.conf file and spin.conf files are stored at the following location:

  - **Windows**: %IGW_LOC%
  - **Linux and UNIX**: $IGW_LOC

Before Configuring CA EEM in FIPS-only Mode

Verify that your environment meets the minimum requirements before migrating the environment to use FIPS-only mode. Print the following to use as a checklist:

- ☐ Upgrade your CA EEM Sever to CA EEM r8.4 SP3 or later versions.
- ☐ Verify that the products that are integrated or connected with CA EEM are configured to use FIPS-only mode.

Configure CA EEM Server in FIPS-only Mode

When you configure CA EEM Server in FIPS-only mode, CA EEM uses only FIPS 140-2 compliant cryptographic libraries to encrypt and decrypt sensitive data.

Notes:

- In FIPS-only mode, use IE7 (or above) or Firefox 3.0 (or above) to view the CA EEM admin GUI. For more information about how to configure Firefox in FIPS 140-2 mode, see the Firefox support site.
- The following procedure is also valid for changing the security mode of the CA EEM Server from FIPS-only to non-FIPS or non-FIPS to FIPS -only.

To configure CA EEM in FIPS-only mode

1. Stop iGateway service.
2. Stop the CA Directory services using the following commands:

   Windows

   ```
   dxserver stop all
   ```

   Linux and UNIX

   ```
   su - dsa -c "dxserver stop all"
   ```

3. Open the iGateway.conf file and set the following tag to ON:

   ```
   <FIPSMode>ON<FIPSMode>
   ```

   Note: To change the mode from FIPS-only to non-FIPS, set FIPSMode tag to OFF.

4. Start the CA Directory services using the following commands:

   Windows

   ```
   dxserver start all
   ```

   Linux and UNIX

   ```
   su - dsa -c "dxserver start all"
   ```

5. Start iGateway service.

   CA EEM is configured in a FIPS-only mode.

Verify CA EEM Server is in FIPS-only Mode

To verify that the CA EEM Server is in FIPS-only mode, do the following:

1. Enter the URL https://*hostname* or IP *address*:5250/spin/eiam/about.csp in your browser.

   The About page opens.

2. Verify that the FIPS: label is set to Enabled.

   If the label is set to Enabled, it indicates that the CA EEM Server is in FIPS-only mode.

Communication Between CA EEM Server and External LDAP Directories

The communication between CA EEM Server and an external directory is dependent on the type of LDAP connection between the two: encrypted or nonencrypted. The following are the supported modes of operation of CA EEM Server and external directory based on encryption:

Encryption is enabled in CA EEM Server for LDAP communication

When CA EEM Server is configured to use encrypted channel of communication with an external LDAP directory, if CA EEM Server is in FIPS mode, the LDAP directory must also be configured to use FIPS-compatible mode.

Configure the CA EEM to use Server certificates in a PKCS#11 Device

To use nCipher PKCS#11 devices with the CA EEM Server or the CA EEM SDK, configure the nCipher device and set the following property is set as follows:

```
CKNFAST_OVERRIDE_SECURITY_ASSURANCES=all
```

Note: For more information about how to configure the nCipher device with a hard token, see the nCipher documentation.

To configure the CA EEM Server to use certificates stored in a PKCS#11 devices, do the following:

1. Stop the iGateway service.
2. Open the iGateway.conf file and edit the <Connector name="defaultport"> CA Portal5250</port> tags to set the following values:

   certType

   Defines the type of certificate to be used. Supported certificate types are p12, pem, and p11.

   Default: pem

   Type: Childnode

   Using P11 certificate

   <pkcs11Lib/>—Path to PKCS11 library provided by token

   <token/>—Token id

   <userpin/>—Munged user pin

   <id/>—Certificate and private key id

   <sensitive/>—Private key is sensitive. Sensitive keys are not converted as software keys and crypto operation are performed using the cryptopki hardware (nonsensitive key can be treated as sensitive, but sensitive keys cannot be converted or treated as nonsensitive key)

   Default: False
3. Save and close the iGateway.conf file.
4. Start the iGateway services.

Configure the CA EEM to Store Server Certificates in a PKCS#11 Device

To store the CA EEM certificates in a PKCS#11 device, do the following:

1. Stop the iGateway service.
2. Open the iGateway.conf file and edit the <CertificateManager> tags to set the following values:

   certType

   Defines the type of certificate to be used. Supported certificate types are p12, pem, and p11.

   Default: pem

   Type: Childnode

   Using P11 certificate

   <pkcs11Lib><pkcs11Lib/>—Path to PKCS11 library provided by token

   <token><token/>—Token id

   <userpin><userpin/>—Munged user pin

   <id><id/>—Certificate and private key id

   <sensitive><sensitive/>—Private key is sensitive. Sensitive keys are not converted as software keys and cryptographic operation are performed using the cryptoki hardware (nonsensitive key can also be treated as sensitive but sensitive keys cannot be converted/treated as nonsensitive key) – optional defaults to false
3. Save and close the iGateway.conf file.
4. Start the iGateway services.

Configure Your Application in FIPS-only Mode

To configure your application in FIPS-only mode, verify that the CA EEM SDK is in FIPS-only mode, CA EEM SDK uses only FIPS-compliant techniques for cryptography. The CA EEM SDK configuration file, eiam.config controls the secure mode of operation of the CA EEM SDK. Before configuring the CA EEM SDK in FIPS-only mode, verify the following:

- Verify that your CA EEM SDK is at version r8.4 SP3 or later.
- Migrate any existing P12 certificates used by CA EEM to PEM certificates.
- Initialize the CA EEM SDK in FIPS-only mode.

Migrate P12 certificates used by Your Application to PEM certificates.

CA EEM supports P12, PEM, and PKCS#11 certificates with the following considerations:

- P12 support is disabled (not available) under FIPS-only mode. As an alternative, in FIPS-only mode, PEM and PKCS#11 certificate support has been added.

Note: CA EEM C# SDK supports only PEM certificates in FIPS-only mode, P12 and PEM certificates in non-FIPS mode.

So, if you are using any P12 certificates, migrate these certificates to one of the supported certificate formats in the FIPS-only mode. Use the igwCertUtil utility to convert P12 certificates to pem certificates. The igwCertUtil is a utility to convert, create, or delete certificates. The igwCertUtil is located in the following folder:

Windows

%IGW_LOC%

UNIX and LINUX

$IGW_LOC

igwcertutil Utility—Create, Copy, Convert, and Delete Certificates

Valid on Windows, UNIX, and Linux

The create command has the following format:

```
igwCertUtil -version version -create -cert inputcert-params -issuer issuercert
-params [-debug] [-silent]
```

The convert command has the following format:

```
igwCertUtil -version version -conv -cert inputcert-params -target newcert-params
[-debug] [-silent]
```

The copy command has the following format:

```
igwCertUtil -version version -copy -cert inputcert-params -target newcert-params
[-debug] [-silent]
```

The delete command has the following format:

```
igwCertUtil -version version -delete -cert cert-params [-debug] [-silent]
```

-version *version*

Specifies the version of igwCertUtil used when creating, converting, copying, or deleting certificates. Version is used for backward compatibility. If igwCertUtil is modified, the version tag gets the old behavior.

-cert *inputcert*-parms

Specifies the certificate as an XML string when creating, converting, or copying certificates.

-issuer *issuercert*-parms

Specifies the certificate that is used to sign the newly generated certificate when creating a certificate. If no certificate is specified, a self-signed certificate is created.

-target *newcert*-parms

Specifies the configuration for the new certificate when converting (or copying) an existing certificate.

-cert *cert*-parms

-debug

(Optional) Turns on debugging for igwCertUtil.

-silent

(Optional) Turns on silent mode for igwCertUtil.

The following error codes are returned by igwCertUtil:

- CERTUTIL_ERROR_UNKNOWN (-1): unknown or undefined error happened
- CERTUTIL_SUCCESS (0): successful operation
- CERTUTIL_ERROR_USAGE (1): wrong command line arguments passed
- CERTUTIL_ERROR_READCERT (2): unable to read certificate
- CERTUTIL_ERROR_WRITECERT (3): unable to write certificate
- CERTUTIL_ERROR_DELETECERT (4): unable to delete certificate

Example: Convert P12 certificates to PEM certificates

The following example describes usage of converting a P12 certificate to a PEM certificate:

```
igwCertUtil -version 4.6.0.0 -conv  -cert
"<Certificate><certType>p12</certType><certURI>testCert.p12</certURI><certPW>pass
word</certPW></Certificate>" -target "<Certificate><certType>pem</certType>
<certURI>testCert.cer</certURI><keyURI>testCert.key</keyURI></Certificate>"
```

Example: Convert P12 Certificates to PKCS#11 certificate:

```
igwCertUtil -version 4.6.0.0 -conv  -cert
"<Certificate><certType>p12</certType><certURI>testCert.p12</certURI><certPW>pass
word</certPW></Certificate>" -target "<Certificate><certType>p11</certTyp
><pkcs11Lib>pathto-pkcs11Lib</pkcs11Lib><token>pkcs11token</token><userpin>user
in</userpin><id>certid</id></Certificate>"
```

Initialize the CA EEM SDK in FIPS-only Mode

The CA EEM SDK can be initialized in the FIPS-only mode by configuring the eiam.config file. To configure the eiam.config file, see the chapter, Configuring CA EEM SDK (see page 25).

More information:

Before You Configure CA EEM Java SDK in FIPS-only Mode (see page 34)
Configure CA EEM C++ SDK in FIPS-only Mode (see page 35)
Configure CA EEM C# SDK in FIPS-only Mode (see page 36)

Chapter 6: Back Up and Restore CA EEM Server

This section contains the following topics:

Overview

Backing up the following CA EEM data ensures that you can restore CA EEM Server installations if they are corrupted:

- Configuration files and folders
- CA EEM data stored in the internal datastore

File System Back Up

We recommend that you back up CA EEM servers regularly or whenever you have administered a change in the CA EEM servers environments. You can use the CA EEM server backups to restore your CA EEM server if it is corrupted.

You must back up the following CA EEM files and folders:

iTechnology folder

iPoz.conf

Defines the CA EEM Server configuration settings.

Eiam.conf

Defines the CA EEM Server session settings.

iPoz.map

Defines the custom mapped LDAP settings.

Spin.conf

Defines the CA EEM GUI launch settings.

logDepot.conf

Defines the event configuration settings.

logdepotdb

Defines the database that stores events.

calm_catalog folder

Defines the folder where the current events are stored.

calm_archive folder

Defines the folder where the archived events are stored.

CA Directory folder

iTechPoz.dxg

Defines the Data DSA and Router DSA file references. The CA EEM Server uses this information to read the configuration settings for its internal datastore.

iTechPoz-*.dxc

Defines the configuration settings for the Data DSA that manages the CA EEM Server data.

iTechPoz-*.pem

Defines the certificate files used during a failover configuration.

Back up the following additional files on UNIX and Linux from the iTechnology folder:

- eiam-type
- Sponsorfiles

Restore Procedures

You must restore your CA EEM data so that you can:

- Recover a CA EEM installation that is corrupted
- Recover a CA EEM server environment that is not working as desired

To recover CA EEM configuration files and data

1. Stop iGateway.
2. Rename all the backed up CA EEM .conf files to .conf.merge, and copy the renamed configuration files to the iTechnology folder. The .conf.merge files are required to merge the backed up configuration files with the new configuration files.
3. Restore CA EEM data.
4. Start iGateway.

Stop iGateway Service

Do the following commands to stop iGateway service:

Windows

1. Click Start, Run, and enter the following command:

   ```
   services.msc
   ```

   The Services panel appears.

2. Select CA iTechnology iGateway 4.6 service in the right pane of the Services panel, right-click, and select Stop.

   The iGateway service is stopped.

Linux and UNIX

$IGW_LOC/S99igateway stop

Start iGateway Service

Do the following commands to start iGateway service:

Windows

1. Click Start, Run, and enter the following command:

   ```
   services.msc
   ```

   The Services panel appears.

2. Select CA iTechnology iGateway 4.6 service in the right pane of the Services panel, right-click, and select Start.

   The iGateway service is started.

Linux and UNIX

$IGW_LOC/S99igateway start

Back up and Restore CA EEM Data

The CA EEM Server uses CA Directory as an internal data store to store the following data:

- Global users
- Global user groups
- Application users
- Application user groups
- Application policies

Note: If you have connected the CA EEM Server to an external LDAP directory, the global users are not stored in the internal data store.

The CA EEM Server data is stored in a directory namespace with the following DSA name:

```
iTechPoz-Hostname
```

Where, DSA is a process that manages the internal data store's namespace. The iTechPoz-Hostname DSA manages the CA EEM Server data.

The backup process involves dumping the data from this DSA to an LDIF file. And, the restore process involves loading the backed up LDIF file into the DSA.

How to Back Up CA EEM Data

Use one of the following procedures to back up CA EEM data:

- Create an online dump of the datastore and convert the dump to an LDIF file.
- Create an offline dump of the datastore.

Create a Backup from an Online Dump

You can take a consistent snapshot copy of the datastore of a running DSA (an online dump). The DSA completes any updates before carrying out the online dump and does not start any more updates until the copy is finished.

Note: Each dump overwrites the previous dump. If you want to save the online dump, copy it to another location before the next dump.

You can create a backup of an online dump using the following process:

1. Generate an online dump from the dsa console port on demand (see page 59).

   Or

   Schedule an online dump (see page 61).

2. Convert the online dump to an LDIF file (see page 62).

Generate an Online Dump from the DSA Console

To generate the online dump from the DSA console, you must do the following:

1. Set a local DSA console port.
2. Connect to a local DSA and dump the data store.

Set Local DSA Console Port

To set the local DSA console port

1. Stop the local DSA services using the following commands:

   Windows

   ```
   dxserver stop dsaname
   ```

   Linux and UNIX

   ```
   su - dsa -c "dxserver stop dsaname"
   ```

2. Open the configuration file (iTechPoz-*Hostname*.dxc) and add the local console port entry immediately after the snmp-port entry:

   ```
   console-port = 10510
   ```

   Note: The configuration file is located in the following folders:

   Windows

   ```
   %DXHOME%\config\knowledge\iTechPoz-Hostname.dxc
   ```

   Linux and UNIX

   ```
   $DXHOME/config/knowledge/iTechPoz-Hostname.dxc
   ```

3. Save the iTechPoz-Hostname.dxc file.

4. Start the local DSA services using the following commands:

   Windows

   ```
   dxserver start dsaname
   ```

   Linux and UNIX

   ```
   su - dsa -c "dxserver start dsaname"
   ```

Connect to a Local DSA Console and Dump the Data Store

To connect to a local DSA Console

1. Open a command prompt on the host on which the DSA is running.

2. Enter the following command:

   ```
   telnet localhost local-port-number
   ```

   local-port-number

   Specifies the console port number of the DSA to which you want to connect.

3. Enter the following command:

   ```
   dump dxgrid-db;
   ```

   An online dump *iTechPoz-Hostname.zdb* is created in the %DXHOME%\Data folder for Windows and $DXHOME/Data folder for UNIX platforms.

   Note: The online dump process is finished only when the .ZDB file size is equal to your data store (*.db) size.

Schedule an Online Dump

You can schedule the CA EEM Server to generate an online dump at a specifc time or to create the online dump at regular intervals.

Note: Each dump overwrites the previous backup file. Create a cron job on UNIX or a scheduled task on Windows to copy the backed up file to a safe location before the next dump.

To schedule an online dump

1. Stop the local DSA services using the following commands:

   Windows

   `dxserver stop dsaname`

   Linux and UNIX

   `su - dsa -c "dxserver stop dsaname"`

2. Open the configuration file (iTechPoz-*Hostname*.dxc) and add the following lines:

   `dump dxgrid-db period <start> <period>`

   where

   period *start period*

   (Optional) Specifies that the online dump is performed at regular intervals.

   start

   Defines the number of seconds from Sunday 00:00:00 a.m. GMT.

   Note: The start time is defined using GMT and not your local time.

   period

   Defines the number of seconds between online dumps.

   Note: The start time is relative to the period and should be lower than it. If you specify a start time that is greater than the period, the actual start is *start* - *period*. For example, if the *period* is 3600 seconds (one hour) and *start* is 3610 seconds, the online dump starts 10 seconds from midnight GMT and continues every hour from then on.

3. Save the iTechPoz-*Hostname*.dxc file.

4. Start the local DSA services using the following commands:

   Windows

   `dxserver start dsaname`

   Linux and UNIX

   `su - dsa -c "dxserver start dsaname"`

Example: Perform an Online Dump Every Hour

The following command takes a snapshot copy of the datastore every hour:

```
dump dxgrid-db period 0 3600
```

Example: Perform an Online Dump Every Night

The following command takes a snapshot copy of the datastore every night at 3 a.m. in a GMT+10:00 time zone:

```
dump dxgrid-db period 61200 86400
```

In this example, the start time is the number of seconds from Sunday midnight GMT to the first 3 a.m. slot, corrected by the time zone value, as follows:

```
(3 am – 10 time zone + 24 hours) * 60 minutes * 60 seconds = 61,200
```

The 24 hour period is calculated as follows:

```
24 hours * 60 minutes * 60 seconds = 86,400
```

Convert the Online Dump to an LDIF File

To create a backup of the data store from an online dump, you must convert the .ZDB files to an LDIF file.

To back up a directory to an LDIF file

1. Log in as the user dsa (on UNIX) or the administrator (on Windows).
2. Use the following command to back up the datastore to the LDIF file:

   dxdumpdb -f *filename* -z *dsaname*

   -f *filename*

   Specifies the file path and name of the LDIF file.

   -z

   Specifies that DXdumpdb dumps from the online dump.

   dsaname

   Specifies the name of the DSA.

   The LDIF file is created at the specified path.

Create an Offline Backup of the Data Store

An offline backup requires that you stop and start the DSA services every time you take a backup.

To create an offline backup

1. Login as the user dsa (on UNIX) or the administrator (on Windows).
2. Stop the local DSA services using the following commands:

   Windows

   `dxserver stop dsaname`

   Linux and UNIX

   `su - dsa -c "dxserver stop dsaname"`
3. Run the following command from the command line:

   `dxdumpdb -f filename  dsaname`

   -f *filename*

   Specifies the file path and name of the LDIF file.
4. Start the local DSA services using the following commands:

   Windows

   `dxserver start dsaname`

   Linux and UNIX

   `su - dsa -c "dxserver start dsaname"`

   The LDIF file is created at the specified path and the offline backup is successful.

How to Restore CA EEM Data

Use the following procedure to restore CA EEM data:

1. Stop the local DSA services using the following commands:

   Windows

   `dxserver stop dsaname`

   Linux and UNIX

   `su - dsa -c "dxserver stop dsaname"`

2. Use the DXloaddb to a datastore from an LDIF file.

   `dxloaddb dsaname ldif-file`

3. Start the local DSA services using the following commands:

   Windows

   `dxserver start dsaname`

   Linux and UNIX

   `su - dsa -c "dxserver start dsaname"`

Chapter 7: Configuring Failover

This section contains the following topics:

Failover

When a primary CA EEM Server is unavailable, the CA EEM client applications automatically switches to a secondary CA EEM Server configured in a failover setup.

To configure CA EEM failover, you must do the following:

- Data store failover (see page 65)
- Server failover (see page 70)

Application Data Store Failover

The CA EEM Server uses CA Directory as the application data store. This application data store provides built-in support for failover and recovery. Synchronize the following on all the servers in the failover configuration:

1. System time
2. Security mode (non-FIPS or FIPS-only)
3. Application data store
4. Ensure that the DNS lookup is proper

Important! Backup the application data stores before synchronizing. For more information about how to backup the data store, see Backing Up CA EEM Data Stored in CA Directory.

Configure the Application Data Store Failover

Note: Perform the steps in the following procedure on the primary server. Any steps to be performed on secondary servers are explicitly mentioned.

This procedure assumes that you have installed the CA EEM Server with the following default values:

- dsa user: dsa
- data dsa port: 509
- group membership: etrdir

If you customized any of these parameters to a custom value, replace the default values with the custom values.

To configure application data store failover

1. Stop the CA EEM services using the following commands on all the servers in the failover setup:

   Windows

   ```
   net stop igateway
   dxserver stop all
   ```

   Linux and UNIX

   ```
   $IGW_LOC/S99igateway stop
   su - dsa -c "dxserver stop all"
   ```

2. Copy the following files from each of the secondary CA EEM Servers to the primary server, say Server1, to the respective folders:

   Windows

   ```
   %DXHOME%\config\knowledge\iTechPoz-HostnameOfServerN.dxc
   %DXHOME%\config\knowledge\iTechPoz-HostnameOfServerN-Router.dxc
   %DXHOME%\config\ssld\personalities\iTechPoz-HostnameOfServerN.pem
   %DXHOME%\config\ssld\personalities\iTechPoz-HostnameOfServerN-Router.pem
   ```

   Linux and UNIX

   ```
   $DXHOME/config/knowledge/iTechPoz-HostnameOfServerN.dxc
   $DXHOME/config/knowledge/iTechPoz-HostnameOfServerN-Router.dxc
   $DXHOME/config/ssld/personalities/iTechPoz-HostnameOfServerN.pem
   $DXHOME/config/ssld/personalities/iTechPoz-HostnameOfServerN-Router.pem
   ```

3. Copy the following file from the secondary servers to a temporary folder on the primary server Server1:

   UNIX and Linux

   ```
   $DXHOME/config/ssld/iTechPoz-trusted.pem
   ```

   Windows

   ```
   %DXHOME%\config\ssld\iTechPoz-trusted.pem
   ```

4. Edit the configuration files (iTechPoz-*HostnameOfServerN*.dxc) of all the servers on Server1 as follows:

   Modify the following line:

   ```
   address      = tcp localhost port 509

   #address = tcp HostnameOfServerN port 509, tcp localhost port 509

   #dsa-flags  = multi-write
   ```

   To read:

   ```
   #address      = tcp localhost port 509

   address = tcp HostnameOfServerN port 509, tcp localhost port 509

   dsa-flags   = multi-write
   ```

   Notes:

   - CA EEM uses port number 509 as the default data dsa port. If you have configured the CA EEM Server to use a custom data dsa port, replace 509 with the custom port number.
   - To use IP addresses instead of hostname, enclose the IP address in double quotes (" ").

5. Edit the iTechPoz.dxg of Server1 to include the secondary server references.

   Example:

   ```
   # iTechPoz - iTechnology Repozitory
   # Source the knowledge files of the iTechPozRouter and iTechPoz DSAs.
   source "iTechPoz-HostnameofServer1-Router.dxc";
   source "iTechPoz-HostnameofServer1.dxc";
   source "iTechPoz-HostnameOfServer2-Router.dxc";
   source "iTechPoz-HostnameOfServer2.dxc";
   source "iTechPoz-ServerN-Router.dxc";
   source "iTechPoz-ServerN.dxc";
   ```

6. Create a new iTechPoz-trusted.pem file by concatenating the contents of iTechPoz-trusted.pem of a secondary server, Server2 with Server1.

   Windows

   ```
   type <absolute path to iTechPoz-trusted.pem of Server2> >> <absolute path to iTechPoz-trusted.pem of Server1>
   ```

   UNIX or Linux

   ```
   cat <absolute path to iTechPoz-trusted.pem of Server2> >> <absolute path to iTechPoz-trusted.pem of Server1>
   ```

   Example: type "C:\Program Files\CA\Directory\dxserver\config\ssld\iTechPoz-trusted_2.pem" >> "C:\Program Files\CA\Directory\dxserver\config\ssld\iTechPoz-trusted.pem

   The iTechPoz-trusted.pem of the secondary server is concatenated to the iTechPoz-trusted.pem of Server1.

7. Concatenate the iTechPoz-trusted.pem of each of the other secondary servers with the resultant iTechPoz-trusted.pem of the Server1 from Step 6.

8. Copy the following files from the primary server to the respective folders on all the secondary servers:

   Note: Back up the iTechPoz-trusted.pem and the data dsa and router files (iTechPoz*) of the secondary servers before performing the copying.

   UNIX and Linux

   ```
   $DXHOME/config/ssld/iTechPoz-trusted.pem
   $DXHOME/config/ssld/personalities/iTechPoz-*.pem
   $DXHOME/config/knowledge/iTechPoz*
   ```

   Windows

   ```
   %DXHOME%\config\ssld\iTechPoz-trusted.pem
   %DXHOME%\config\ssld\personalities\iTechPoz-*.pem
   %DXHOME%\config\knowledge\iTechPoz*
   ```

9. Edit the iTechPoz.dxg file on each of the secondary servers. The iTechPoz.dxg file must read:

   ```
   # iTechPoz - iTechnology Repozitory
   # Source the knowledge files of the iTechPozRouter and iTechPoz DSAs.
   source "iTechPoz-HostnameOfServerN-Router.dxc";
   source "iTechPoz-HostnameOfServerN.dxc";
   source "iTechPoz-HostnameOfServer1-Router.dxc";
   source "iTechPoz-HostnameOfServer1.dxc";
   source "iTechPoz-HostnameOfServer2-Router.dxc";
   source "iTechPoz-HostnameOfServer2.dxc";
   source "iTechPoz-ServerKRouter.dxc";
   source "iTechPoz-ServerK.dxc";
   ```

   Note: The entries for the localhost must appear before the entries for other servers.

10. Modify the ownership and group membership of the following files to dsa and etrdir respectively for all the CA EEM servers that are running on UNIX or Linux. Run the following commands:

    ```
    chown dsa:etrdir $DXHOME/config/ssld/iTechPoz-trusted.pem
    chown dsa:etrdir $DXHOME/config/knowledge/iTechPoz*
    chown dsa:etrdir $DXHOME/config/ssld/personalities/iTechPoz-*.pem
    ```

11. Start the CA EEM services using the following commands on all the servers:

    Windows

    ```
    dxserver start all
    net start igateway
    ```

    Linux and UNIX

    ```
    su - dsa -c "dxserver start all"
    $IGW_LOC/S99igateway start
    ```

    The application data store failover configuration is saved.

CA EEM Server Failover

Configure each CA EEM Server to trust certificates from all the other secondary servers in the failover configuration. Repeat the following procedure on all the servers in the failover configuration.

Note: CA EEM failover does not support storing certificates in PKCS#11 devices.

To configure the CA EEM Server for failover

1. Copy the rootcert.cer file from all the failover servers to the Server1.

   Note: The PEM certificate file (rootcert.cer) is located in the iTechnology directory of each failover server.

2. Enter the URL https://server1:5250/spin.

3. Select iTech Administrator, and click Go.

   A page appears displaying the dashboard for an iTech Administrator.

4. Click Login, Select iAuthority as the Option Type and login as EiamAdmin.

   The iTechnology Administrator GUI opens.

5. Click the Configure tab, add the hostname of failover server ServerN in the Trust another iAuthority pane and click Trust.

   An entry is added in iControl.conf file and Server1 starts trusting sessions from ServerN.

   Note: Repeat this step to add the hostnames of all other failover servers.

6. Click the iAuthority tab, and in the Add Trusted Root pane enter a Label to identify ServerN, browse to the location of PEM Certificate file for ServerN on the localhost, and click Add Trusted Root.

   An entry is added in iAuthority.conf and Server1 starts trusting certificates from ServerN.

   Note: Repeat this step to add certificate entries for all other servers in the failover setup.

Configure CA EEM Files

Configure a CA EEM Server to receive the list of available failover servers.

To configure CA EEM Server

1. Stop the iGateway service.
2. Open the iPoz.conf file and add the following tag for each of the servers in the failover setup:

   ```
   <BackboneMember>HostnameofServer2</BackboneMember>

   <BackboneMember>HostnameofServer3</BackboneMember>

   <BackboneMember>HostnameofServerN</BackboneMember>
   ```

   Note: The iPoz.conf file is in the following location:

   - **Windows**: %IGW_LOC%
   - **Linux and UNIX**: $IGW_LOC
3. Start the iGateway service.

Note: Repeat the preceding procedure on all the CA EEM Servers configured in the failover setup.

More information:

Start iGateway Service (see page 57)
Stop iGateway Service (see page 57)

Artifact Federation

Artifact federation is the ability to use artifacts across multiple CA EEM Servers in a failover setup. For example, an artifact generated on CA EEM Server1 can be used to authenticate against any CA EEM Server in the failover setup.

What are the prerequisites for artifact federation

Before configuring artifact federation, configure CA EEM Servers for data store failover and server failover.

How is artifact federation controlled

Artifact federation is controlled using the <ArtifactManager> tag in the iPoz.conf file.

Enable Artifact Federation

If you want to use artifacts federation, perform the following procedure on all CA EEM Servers in a failover setup.

Enable artifact federation

1. Stop the iGateway service.
2. Locate and open the iPoz.conf file from the following location:

   Windows

   %IGW_LOC%

   UNIX or Linux

   $IGW_LOC
3. Edit the following tag:

   ```
   <ArtifactManager SessionTimeout="10"
   RequestTimeout="30"ArtifactStore="local/federated"></ArtifactManager>
   ```

   Where

   SessionTimeOut

   Specifies the expiry time in minutes for an exported session.

   Default: 10 minutes

   RequestTimeOut

   Specifies the expiry time in minutes for a launch request.

   Default: 30 minutes

   Store

   Specifies the storage location of artifacts. If the value is mentioned as local, the artifacts from one CA EEM Server are not available on other CA EEM Servers in the failover setup. For artifacts to be available on all CA EEM Servers, set the value for this parameter as federated.

   Value: [local | federated]

   Default: local

   Note: The SessionTimeOut and RequestTimeOut parameters are also present in the eiam.conf file. If you specify these parameters in the eiam.conf file, the values from eiam.conf file take precedence.
4. Save and close the file.
5. Start the iGateway services.

   The artifact federation is enabled.

Chapter 8: Integrating With CA SiteMinder

This section contains the following topics:

CA SiteMinder

You can configure CA EEM to use the existing CA SiteMinder data store for retrieving user and group information. You can use CA EEM with CA SiteMinder to perform the following:

Retrieve user and group information from CA SiteMinder data store

Applications that embed CA EEM can access the user and group information from CA SiteMinder data store and define policies using CA EEM interface.

Authenticate users against CA SiteMinder user store

User with existing CA SiteMinder login credentials can log into any CA EEM enabled application.

Use the CA SiteMinder session information for user authentication

CA EEM supports single sign-on for CA SiteMinder users. If you have an existing CA SiteMinder session, you can access a CA EEM protected application without being prompted for authentication.

More Information:

CA SiteMinder Configuration Parameters (see page 74)
How Single Sign-on Works between CA SiteMinder and CA EEM (see page 75)
How Authentication Works Using CA SiteMinder Authentication Schemes (see page 75)

How You Integrate CA SiteMinder with CA EEM

To integrate CA SiteMinder with CA EEM, perform the following in CA SiteMinder Administrator:

- Create an agent in CA SiteMinder for communication between CA EEM and CA SiteMinder policy server. Ensure the agent supports 4.x agents.
- Create an administrator or use the existing default administrator "SiteMinder" with system level scope.
- Create a CA SiteMinder User Directory for authorization, which is used by CA EEM to retrieve LDAP attributes.
- Set the UniversalID field to uniquely identifies a user in the directory such as sAMAccountName or UID. You can set the UniversalID from the SiteMinder UI, User Directories, Properties, User attributes tab.
- Set the Password Attribute (RW) on the User Attribute tab to userPassword.
- Create a CA SiteMinder data store for authentication, which is used by CA EEM to authenticate users.

  Note: If the authentication and authorization user store is same, use the existing user store created for authorization.
- Create a Realm with the Resource Filter as “/iamt.html”.
- Create a CA SiteMinder domain and add the User Directories, administrator, and Realm to the domain.

For more information about CA SiteMinder, see the CA SiteMinder documentation.

CA SiteMinder Configuration Parameters

CA SiteMinder consists of two components, policy server and web agents.

Policy server

Policy server provides policy management, authentication, authorization and accounting.

Web agents

Web agents assist CA SiteMinder to access to Web applications and content according to defined security policies.

To enable the protocol between the agent and server, the agent must have a unique name and a shared secret key along with information to define a connection between the client application and the policy server.

For information on parameters to define between the client application and the policy server, see *Online Help*.

How Single Sign-on Works between CA SiteMinder and CA EEM

If you use an application that has an existing CA SiteMinder session to access an CA EEM enabled application, CA EEM recognizes the CA SiteMinder session ticket and creates an CA EEM session without re-authentication.

The following is the basic flow of events for application created using CA EEM with CA SiteMinder integration:

Example: Protecting a web application using CA SiteMinder

A web application using CA EEM with web server pages protected by CA SiteMinder is considered.

1. A user accesses a web application.
2. CA SiteMinder prompts for user authentication and the user submits credentials and is authenticated.
3. The user tries to access the original web application created using CA EEM.
4. Servlet code accesses the HTTPServletRequest context and sends the CA SiteMinder session token to the CA EEM using authenticateWithArtifact.
5. CA EEM Server validates the CA SiteMinder session against the CA SiteMinder Policy Server.
6. An CA EEM session is created and the user identity is loaded, if validation succeeds.

How Authentication Works Using CA SiteMinder Authentication Schemes

The following process describes how authentication is performed using CA SiteMinder APIs:

- A user calls the authenticateWithPassword method by providing the username and password.
- CA EEM sends this information to the CA EEM Server.
- Based on the information, the authentication is performed by calling the CA SiteMinder APIs.
- The group and user information is loaded for the authenticated user.

Note: When CA EEM is connected to a CA SiteMinder user directory, the search calls use the CA SiteMinder APIs instead of the CA EEM search calls.

Configure CA EEM Server-side Logging for CA SiteMinder Modules

To configure log level for CA SiteMinder integration

1. Create a file with the following contents, and save the file with the name sm_log.properties:

```
#filename: sm_log.properties
#set the default logging level for the root logger
.level = INFO
#set the default logging level for the logger name com.ca.eiam
com.ca.eiam.level = ALL
```

2. Modify the log level for the com.ca.eiam logger in the sm.properties file to any of the following values:

   SEVERE

   Specifies a level for messages indicating severe failure.

   WARNING

   Specifies a level for indicating warnings.

   INFO

   Specifies a level for informational messages.

   CONFIG

   Specifies a level for static configuration messages.

   FINE

   Specifies a level for tracing information.

   ALL

   Specifies that all levels of messages are logged.

3. Save the file at the following location:

   Windows

   %IGW_LOC%

   Linux and UNIX

   $IGW_LOC

4. Stop the iGateway service.

5. Open the iGateway.conf file from the location specified in Step 3, and add the following tags within the <JVMSettings></JVMSettings> tags:

```
<Properties name="eiam.sm">
<system-properties>java.util.logging.config.file=sm_log.properties</system-pr
operties>
</Properties>
```

6. Save and close the file.
7. Start the iGateway service.

Chapter 9: CA EEM SDK Logging

The CA EEM SDK logging lets you do the following:

- Application log levels can be changed at run time.
- Configure logging information such as filename, file size, number of backup log files, and so on.

The following files control the logging in the CA EEM SDK:

- eiam.log4cxx.config and eiam.config files for CA EEM C++ SDK
- eiam.log4net.config for CA EEM C# SDK
- eiam.log4j.config and eiam.config files for CA EEM Java SDK

The samples of these logger configurations are shipped with the CA EEM SDK package and are placed in Bin folder:

UNIX

$EIAM_SDK/bin

Windows

%EIAM_SDK%\bin

This section contains the following topics:

About the Logger Configuration Files (see page 79)

About the Logger Configuration Files

The logger configuration files, eiam.log4cxx.config, eiam.log4net.config, and eiam.log4j.config, are used to configure CA EEM SDK logging. These files contain the following major components:

- Appenders
- Loggers
- Root Logger

These components contain configurable parameters that let you customize the logging process based on your business requirements.

Appender

An appender contains parameters that control the logging of each logger. By default, the logger configuration files contains the following appenders:

SDK

Logs the SDK messages into a log file. Specifies the path including the file name of the log file.

Default: eiam.cppsdk.log for C++ SDK, EIAM.C#SDK.log for C#, and eiam.javasdk.log for Java SDK.

Note: If you are deploying your application under Tomcat server on Windows, verify that you use forward slash '/' in the path instead of the backward slash '\'. If you use backward slash, the log file is not created at the path you have specified; instead, the log file is created in the Apache Tomcat folder.

Network

Logs the network call related messages into a log file.

Default:eiam.network.cpp.log for C++ SDK, EIAM.NETWORK.C#SDK.log for C# SDK, and eiam.javasdk.log for Java SDK.

Performance

Logs the performance call related messages into a log file.

Default: eiam.performance.cpp.log for C++ SDK, EIAM.PERFORMANCE.C#SDK.log for C# SDK, and eiam.performance.java.log

Console

Displays the log messages on the console.

SDK appender is enabled by default. To enable other appenders, remove the comment strings (<!-- and -->) from their respective code.

An appender consists of the following configurable parameters:

file

Specifies the log filename of the appender.

append

Specifies if a set of log messages is appended to the log file. If the value is true, the set of log message is appended to the last log message in the log file.

Note: This parameter is named appendToFile in the eiam.log4net.config file.

BufferedIO

Specifies if the latest log message is buffered. If the value is true, latest few log messages are kept in memory before writing to log file. This option minimizes IO operation and is beneficial if the log level is higher.

Value: [*true*|*false*]

Default*:* false

Note: The default size of BufferedIO is 8 KB.

maxFileSize

Specifies the maximum size of the log file. If a log file exceeds the maximum size, a new log file filename log.1 is created and the contents of log file are transferred to log.1 file. The log file now contains latest log messages. If this file too exceeds the maximum size, a new log file filename log.2 is created, the contents of log.1 are transferred to log.2 file, and the contents of log file are transferred to log.1 file.

Default: 10 MB

Minimum: 10 KB

Maximum: 2 GB

Note: The minimum size of the maxFileSize must be greater than or equal to the size of BufferedIO. This parameter is named maximumFileSize in the eiam.log4net.config file.

maxBackupIndex

Specifies the maximum number of backup log files used for keeping old logs. If the number of log files exceeds the maximum backup index value, the file with the oldest log messages is deleted.

Default: 1

Minimum: 1

Maximum: 12

Note: This parameter is named maxSizeRollBackups in the eiam.log4net.config file.

rollingStyle

Specifies the criteria for creating log files. When this parameter is set to Size, if a log file exceeds the maximumFileSize, a new log file is created and the contents of the current log file are backed up.

Default*: Size*

ConversionPattern

Specifies the formatting of a log message. Configure the format modifiers and conversion characters to define the conversion pattern.

Note: For more information about conversion patterns, refer the topic log4j in www.apache.org.

Example: SDK Appender

```
<appender name="SDK" class="org.apache.log4j.RollingFileAppender">
    <!-- The active sdk log file -->
    <param name="file" value="eiam.cppsdk.log" />
    <param name="append" value="true" />
    <param name="BufferedIO" value="false"/>
    <param name="maxFileSize" value="10000KB" />
    <param name="maxBackupIndex" value="1" />
    <layout class="org.apache.log4j.PatternLayout">
    <!-- The log message pattern -->
    <param name="ConversionPattern" value="%5p %d{ISO8601} [%t] [%c] %m%n"/>
    </layout>
</appender>
```

Appender in eiam.log4net.config

An appender contains parameters that control the logging of each logger. By default, the logger configuration files contains the following appenders:

SDK

Logs the SDK messages into a log file. Specifies the path including the file name of the log file.

Default: EIAM.C#SDK.log

Note: If you are deploying your application under Tomcat server on Windows, ensure that you use forward slash '/' in the path instead of the backward slash '\'. If you use backward slash, the log file is not created at the path you have specified; instead, the log file is created in the Apache Tomcat folder.

Network

Logs the network call related messages into a log file.

Default:EIAM.NETWORK.C#SDK.log

Performance

Logs the performance call related messages into a log file.

Default: EIAM.PERFORMANCE.C#SDK.log

Console

Displays the log messages on the console.

SDK appender is enabled by default. To enable other appenders, remove the comment strings (<!-- and -->) from their respective code.

An appender consists of the following configurable parameters:

file

Specifies the log filename of the appender.

appendToFile

Specifies if a set of log messages is appended to the log file. If the value is true, the set of log message is appended to the last log message in the log file.

maxSizeRollBackups

Specifies the maximum number of backup log files used for keeping old logs. If the number of log files exceeds the maximum backup index value, the file with the oldest log messages is deleted.

Default: 1

Minimum: 1

Maximum: 12

rollingStyle

Specifies the criteria for creating log files. When this parameter is set to Size, if a log file exceeds the maximumFileSize, a new log file is created and the contents of the current log file are backed up.

Default*: Size*

maximumFileSize

Specifies the maximum size of the log file. If a log file exceeds the maximum size, a new log file filename log.1 is created and the contents of log file are transferred to log.1 file. The log file now contains latest log messages. If this file too exceeds the maximum size, a new log file filename log.2 is created, the contents of log.1 are transferred to log.2 file, and the contents of log file are transferred to log.1 file.

Default: 10 MB

Minimum: 10 KB

Maximum: 2 GB

Note: The minimum size of the maxFileSize must be greater than or equal to the size of rollingStyle.

ConversionPattern

Specifies the formatting of a log message. Configure the format modifiers and conversion characters to define the conversion pattern.

Note: For more information about conversion patterns, refer the topic log4net in www.apache.org.

Logger

Loggers let you control the log messages for CA EEM SDK. To enable a logger, remove the comment strings from their respective code.

A logger contains the following parameters:

logger name

Specifies the name of a logger.

additivity

Specifies if the log messages are duplicated in the SDK log file.

Value: [*true*|*false*]

Default: false

level value

Specifies log level of a logger.

Value: [*Trace*|*Debug*|*Info*|*Warn*|*Error*|*Fatal*|*Off*]

The following are the log levels, in the order of their precedence:

Note: Higher the log level, lesser is the performance of CA EEM.

Trace

Indicates low level debugging. It contains control flow and passes arguments.

Debug

Indicates messages used for problem diagnosis. It contains contextual information.

Info

Indicates contextual information that traces execution at a coarse-grained level in a production environment.

Warn

Indicates a potential problem in the system. For example, if the message category corresponds to security, a warning message must display if a dictionary attack is detected.

Error

Indicates a serious problem in the system. The problem is non-recoverable and requires manual intervention.

Fatal

Indicates a very sever error that may lead the application to abort.

Off

Indicates the absence of logging.

Note: The log level of the default SDK appender must be Error.

Example: Performance Logger

```
<logger name="Perform" additivity="false">
    <level value="trace"/>
    <appender-ref ref="Performance" />
</logger>
```

Root Logger

Root Logger controls the log level of all the appenders. However, if the log level of the referenced appender in root logger is different from the log level specified in the parent appender, the higher priority log level overrides the lower priority log level.

For example, if the log level of a the root logger is Error and the log level of the Network appender is Trace, the log level Trace overrides Error and the system considers log messages with log level Trace at the runtime.

Example: Root Logger

```
<root>
    <priority value="error" />
    <appender-ref ref="SDK" />
</root>
```

Configure the Logger Files

CA EEM lets you configure the log messages related to network, performance, console, and SDK classes.

To configure logger files

1. Open the logger configuration file, eiam.log4cxx.config, eiam.log4net.config, or eiam.log4j.config, in a text editor.
2. Enable loggers and appenders. Remove the comment strings from the logger and appender code to enable loggers and appenders.
3. Update the appender parameters.
4. Save the logger configuration file.

Example of a eiam.log4cxx.config File

The following is an example of the eiam.log4cxx.config file:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<!-- Note that this file is read by the sdk every 60 seconds -->

<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">

    <appender name="SDK" class="org.apache.log4j.RollingFileAppender">
        <!-- The active sdk log file -->
        <param name="file" value="eiam.cppsdk.log" />
        <param name="append" value="true" />
        <param name="BufferedIO" value="false"/>
        <param name="maxFileSize" value="10000KB" />
        <param name="maxBackupIndex" value="1" />
        <layout class="org.apache.log4j.PatternLayout">
        <!-- The log message pattern -->
        <param name="ConversionPattern" value="%5p %d{ISO8601} [%t] [%c] %m%n"/>
        </layout>
    </appender>

    <appender name="Network" class="org.apache.log4j.RollingFileAppender">
        <!-- The file to log Network calls -->
        <param name="file" value="eiam.network.cpp.log" />
        <param name="append" value="true" />
        <param name="BufferedIO" value="true"/>
        <param name="maxFileSize" value="10000KB" />
        <param name="maxBackupIndex" value="1" />
        <layout class="org.apache.log4j.PatternLayout">
        <!-- The log message pattern -->
        <param name="ConversionPattern" value="%5p %d{ISO8601} [%t] [%c] %m%n"/>
        </layout>
    </appender>

    <appender name="Performance" class="org.apache.log4j.RollingFileAppender">
        <!-- The file to log Performance calls -->
        <param name="file" value="eiam.performance.cpp.log" />
        <param name="append" value="true" />
        <param name="BufferedIO" value="true"/>
        <param name="maxFileSize" value="10000KB" />
        <param name="maxBackupIndex" value="1" />
        <layout class="org.apache.log4j.PatternLayout">
        <!-- The log message pattern -->
        <param name="ConversionPattern" value="%5p %d{ISO8601} [%t] [%c] %m%n"/>
        </layout>
    </appender>

    <appender name="Console" class="org.apache.log4j.ConsoleAppender">
        <!-- Logs to Console -->
        <layout class="org.apache.log4j.PatternLayout">
```

```
            <!-- The log message pattern -->
            <param name="ConversionPattern" value="%5p %d{ISO8601} [%t] [%c] %m%n"/>
            </layout>
        </appender>

        <!-- Remove comment to enable Performance Logging -->
        <!--
        <logger name="Perform" additivity="false">
            <level value="trace"/>
            <appender-ref ref="Performance" />
        </logger>
        -->

        <!-- Remove comment to enable Network Logging -->
        <!--
        <logger name="Network" additivity="false">
            <level value="trace"/>
            <appender-ref ref="Network" />
        </logger>
        -->

        <root>
            <priority value="error" />
            <appender-ref ref="SDK" />
            <!-- <appender-ref ref="Console" /> -->
        </root>
    </log4j:configuration>
```

Example of a eiam.log4net.config File

The following is an example of a eiam.log4net.config file:

```
<?xml version="1.0" encoding="utf-8" ?>

<log4net>
        <appender name="SDK" type="log4net.Appender.RollingFileAppender">
                <file value="EIAM.C#SDK.log" />
                <appendToFile value="true" />
                <maxSizeRollBackups value="1" />
                <maximumFileSize value="10000KB" />
                <rollingStyle value="Size" />
                <layout type="log4net.Layout.PatternLayout">
                        <conversionPattern value="%date [%thread] %-5level %logger
- %message%newline" />
                </layout>
        </appender>

        <appender name="Network" type="log4net.Appender.RollingFileAppender">
                <file value="EIAM.NETWORK.C#SDK.log" />
                <appendToFile value="true" />
                <maxSizeRollBackups value="1" />
                <maximumFileSize value="10000KB" />
                <rollingStyle value="Size" />
                <layout type="log4net.Layout.PatternLayout">
                        <conversionPattern value="%date [%thread] %-5level %logger
- %message%newline" />
                </layout>
        </appender>

        <appender name="Performance" type="log4net.Appender.RollingFileAppender">
                <file value="EIAM.PERFORMANCE.C#SDK.log" />
                <appendToFile value="true" />
                <maxSizeRollBackups value="1" />
                <maximumFileSize value="10000KB" />
                <rollingStyle value="Size" />
                <layout type="log4net.Layout.PatternLayout">
                        <conversionPattern value="%date [%thread] %-5level %logger
- %message%newline" />
                </layout>
        </appender>

        <appender name="ConsoleAppender" type="log4net.Appender.ConsoleAppender">
                <layout type="log4net.Layout.PatternLayout">
                        <conversionPattern value="%date [%thread] %-5level %logger
- %message%newline" />
                </layout>
        </appender>
```

```
        <!-- Uncomment to enable Performance Logging -->
        <!--
        <logger name="Perform" additivity="false">
                <level value="ERROR"/>
                <appender-ref ref="Performance" />
        </logger>-->

        <!-- Uncomment to enable Network Logging -->
        <!--<logger name="Network" additivity="false">
                <level value="ERROR"/>
                <appender-ref ref="Network" />
        </logger>-->

        <root>
                <level value="ERROR" />
                <appender-ref ref="SDK" />
                <!--    <appender-ref ref="ConsoleAppender" />   -->
        </root>
</log4net>
```

Example of a eiam.lo4j.config File

The following is an example of the eiam.log4cxx.config file:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">

<!-- Note that this file is read by the sdk every 60 seconds -->

<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">

        <appender name="SDK" class="com.ca.eiam.log4j.RollingFileAppender">
                <!-- The active sdk log file -->
                <param name="file" value="eiam.javasdk.log" />
                <param name="append" value="true" />
                <param name="BufferedIO" value="false"/>
                <param name="maxFileSize" value="10000KB" />
                <param name="maxBackupIndex" value="1" />
                <layout class="com.ca.eiam.log4j.PatternLayout">
                        <!-- The log message pattern -->
                        <param name="ConversionPattern" value="%5p %d{ISO8601}
[%t] [%c] %m%n"/>
                </layout>
        </appender>

        <appender name="Network" class="com.ca.eiam.log4j.RollingFileAppender">
                <!-- The file to log Network calls -->
                <param name="file" value="eiam.network.java.log" />
                <param name="append" value="true" />
                <param name="BufferedIO" value="false"/>
                <param name="maxFileSize" value="10000KB" />
                <param name="maxBackupIndex" value="1" />
                <layout class="com.ca.eiam.log4j.PatternLayout">
                        <!-- The log message pattern -->
                        <param name="ConversionPattern" value="%5p %d{ISO8601}
[%t] [%c] %m%n"/>
                </layout>
        </appender>

        <appender name="Performance"
class="com.ca.eiam.log4j.RollingFileAppender">
                <!-- The file to log Performance calls -->
                <param name="file" value="eiam.performance.java.log" />
                <param name="append" value="true" />
                <param name="BufferedIO" value="false"/>
                <param name="maxFileSize" value="10000KB" />
                <param name="maxBackupIndex" value="1" />
                <layout class="com.ca.eiam.log4j.PatternLayout">
                        <!-- The log message pattern -->
```

```
                        <param name="ConversionPattern" value="%5p %d{ISO8601}
[%t] [%c] %m%n"/>
                </layout>
        </appender>

        <appender name="Console" class="com.ca.eiam.log4j.ConsoleAppender">
                <!-- Logs to Console -->
                <layout class="com.ca.eiam.log4j.PatternLayout">
                        <!-- The log message pattern -->
                        <param name="ConversionPattern" value="%5p %d{ISO8601}
[%t] [%c] %m%n"/>
                </layout>
        </appender>

        <!-- Uncomment to enable Performance Logging -->
        <!--
        <logger name="Perform" additivity="false">
                <level value="trace"/>
                <appender-ref ref="Performance" />
        </logger>
        -->

        <!-- Uncomment to enable Network Logging -->
        <!--
        <logger name="Network" additivity="false">
                <level value="trace"/>
                <appender-ref ref="Network" />
        </logger>
        -->

        <root>
                <priority value="error" />
                <appender-ref ref="SDK" />
                <!-- <appender-ref ref="Console" /> -->
        </root>

</log4j:configuration>
```

Chapter 10: Configuring External Directory Server Support

This section contains the following topics:

Configure an External Directory with CA EEM

If you are using different external directory stores for authentication and authorization, configure CA EEM as follows:

- Use the iPoz.conf file to configure the external authentication directory with CA EEM
- Use the CA EEM admin GUI to configure the CA EEM Server with the external authorization directory.

  Note: For more information about how to configure references to external directory, see the Online Help.

To configure the CA EEM Server to use an external directory for authentication, configure the following options in the iPoz.conf file located in $IGW_LOC for UNIX and %IGW_LOC% for Windows folder after the installation.

Note: Stop iGateway before modifying the iPoz.conf file and restart it afterwards.

UseExternalAuthDirectory

Specifies whether you want to use a different external directory for Authentication. Type True to use a different external directory. The default is False.

ExternalAuthDirType

Specifies the type of external directory. The following table lists the supported external directories and the corresponding values you must specify for this tag:

Directory	Value
Active Directory	ADS
Sun One Directory	SunONE
Novel eDirectory	eDirectory

Directory	Value
Novel eDirectory CN	eDirectoryCN
Custom Mapped Directory	Map

ExternalAuthDirUserDn

Specifies the UserDn for the type of external directory specified.

ExternalAuthDirPassword

Specifies the user password in the encrypted format.

Note: Munge the password using the following command and paste it in the ipoz.conf file.

`$IGW_LOC/safex -munge <clear text password>`

ExternalAuthDirHost

Specifies the host name on which the external directory has been configured.

ExternalAuthDirPort

Specifies the port to which the external directory listens.

ExternalAuthDirUserSearchPreFilter

Specifies the pre-search filter per the external directory. You can search for any object class such as, users.

ExternalAuthDirUserSearchPostFilter

Specifies the post-search filter per the external directory. You can search for any object class such as, users.

ExternalDirCacheFolder

Specifies if the CA EEM Server must cache the external directory folders. If this tag is set to True, CA EEM Server caches the external folders and you can access these folders using the CA EEM admin GUI. If this tag is set to False, CA EEM does not display the external directory folders in the CA EEM admin GUI.

Value: [True|False]

Default: **True**

ExternalDirEscapeSlash

Specifies if CA EEM must handle the forward slash '/' in the DN returned by external directories. Set this tag to True if CA EEM has to escape the forward slash.

Value**:** [True|False]

Default: False

Configure External Directory Failover Support

You can extend the capability of CA EEM to fall back on another external directory server that is a replicated version of the server. This can be accomplished by providing the mapping in the iPoz.conf.

Before configuring CA EEM for external directory failover support, that the secondary external directories are an exact replica of the primary external directory.

Note: Stop iGateway before modifying the iPoz.conf file and restart it afterwards.

ExternalDirHostBackup

Specifies the host name of the replicated external directory server.

ExternalAuthDirHostBackup

Specifies the host name of the different external directory server to be used for user authentication.

Connect to LDAP Servers over TLS

To establish a TLS connection to LDAP server, you must configure the LDAP server to accept anonymous certificates. To configure EEM to connect to LDAP over TLS, do the following:

To configure CA EEM to connect to LDAP over TLS

1. Login to the CA EEM GUI.
2. Click Configure, EEM Server.
3. Click Global Users/ Global Groups.

   The EEM Server Configuration pane appears.
4. Select the Reference from an external directory option.
5. Enter the configuration details.

   Note: For more information about the configuration details, see the Online Help.
6. Select Use Transport Layer Security (TLS) option.
7. Click Save.

Connecting to LDAP Servers over SSL

To establish an SSL connection to LDAP servers you must have the following certificates:

Certificate Authority Certificate

You can obtain this certificate from a Certificate Authority, for example Verisign or Thwate. This certificate indicates that certificates issued by this Certificate Authority are valid and can be trusted.

LDAP Server Certificate

You must obtain this certificate from a trusted Certificate Authority. This certificate contains information about the LDAP server and identifies the LDAP server with the client.

Note: CA EEM supports only .pem certificates for SSL connections.

How CA EEM Connects to LDAP Server Over SSL

The following process explains how the CA EEM server and the LDAP server communicate over SSL.

1. The CA EEM server connects to the LDAP server using a Certificate Authority certificate.
2. The LDAP server verifies the Certificate Authority certificate, and if the certificate is valid establishes a handshake with the CA EEM server.
3. The LDAP server sends its public key to the CA EEM server during the handshake. The public key is used to encrypt data that is sent to the LDAP server.
4. The CA EEM server uses the public key to encrypt data and sends the data to the LDAP server.
5. The CA EEM server send username and password to authenticate against LDAP server.

How to Configure the SSL Connections

You must follow the following process to configure SSL communication between the LDAP server and the CA EEM server:

1. Configure the LDAP server to use certificates
2. Configure the CA EEM server to communicate over SSL

Configure the LDAP Server to Use SSL Certificates

To configure the LDAP server to use SSL, you must do the following steps:

1. Obtain a Certificate Authority certificate and install the certificate in the trusted certificate store on your LDAP server.
2. Obtain a server certificate from the Certificate Authority and install the certificate in the server certificate store of your LDAP server.
3. Enable the LDAP server to accept SSL connections.

Enable SSL in CA EEM Server

To enable SSL in the server

1. Copy the certificate of Certificate Authority from the LDAP server and save it to the computer where CA EEM server is running.
2. Stop the iGateway service.
3. Open the ipoz.conf file and edit the following tags:

   <ExternalDirSSL>

   Specifies if SSL communication is enabled or disabled. You must set this tag to "true" to enable SSL communication.

   <ExternalDirCACertPath>

   Specifies the path including the certificate file name for Certificate Authority that is stored on the computer where CA EEM server is running.
4. Start the iGateway service.

Chapter 11: Configuring Support for Large Numbers of Policies

This section contains the following topics:

Support for Large Number of Policies

Note: CA EEM provides support for large number of policies only in C++ SDK enabled client environment.

You must configure the CA EEM Server and clients before registering applications that use a large number of policies.

Note: CA EEM supports up to 20,000 policies on the HP-UX platform.

Configure Additional Settings for CA EEM Server on AIX

You must perform the following additional steps to configure the CA EEM Server to support the use of a large number of policies on AIX.

To configure CA EEM Server on AIX

1. Modify the network settings using the following command at the AIX command prompt:

   ```
   no -o tcp_nodelayack=1
   ```

2. Increase the process limit using the following command at the AIX command prompt:

   ```
   ulimit -d unlimited
   ulimit -f unlimited
   ```

Configure Client for all Operating Systems

To support the deployment of a large number of policies, you must configure clients for all operating systems:

- Increase the application cache update time to avoid cache updates during registering applications using Safex.

  For more information about cache update, see the *Programming Guide.*

  Note: We recommend setting the cache update time to 3600 seconds during registration to avoid cache updates during registration. After registration, change the cache update time to 30 seconds, which is the default setting.

- Enable Reliable Event Delivery.

  For more information about Reliable Event Delivery, see the *Programming Guide*.

Chapter 12: Archiving Events

This section contains the following topics:

Overview

You can generate and manage reports for events generated by the CA EEM Server. The archiving system organizes archived files into the following two states:

Warm db Files

Refers to the archive files created once the number of events exceeds Maximum Rows in an event database. Warm archived files are available for querying and reporting from CA EEM server. No data can be inserted into a warm db file. Warm db files are available in CA EEM server only for the number of days specified as Max Archive Days in Event Log Settings. The warm db files are stored in the %IGW_LOC%\calm_archive.

Cold db Files

Refers to the archive files in warm state that are backed up manually to another location. You must back up the calm_archive folder before the Max Archive Days. You cannot query or create reports from a cold db file. The cold db files need to be defrosted before they can be used for querying or reporting.

To change Event Log Settings

1. Log in to CA EEM.

   The CA EEM home page appears.

2. Click Manage Reports, Configuration, Services, Event Log Settings.

   The event log settings appear.

Note: For more information about configuring services to manage reports, see the *Online Help*.

Utility to Defrost Cold DB Files

CA EEM provides a utility to defrost cold db files. You must both restore and defrost the files from cold to warm state before you can run queries against the files and view live reports. The sem utility provides this functionality. The sem utility is found at the following location on the CA EEM Server:

- **Windows**: %IGW_LOC%
- **Linux and UNIX**: $IGW_LOC

To set up sem utility

1. Set environment variables based on your operating system:

   Linux or Solaris

   `export LD_LIBRARY_PATH = <sem_Extraction_Folder>:$LD_LIBRARY_PATH`

   AIX

   `export LIBPATH = <sem_Extraction_Folder>:$LIBPATH`

   HP-UX

   `export SHLIB_PATH= <sem_Extraction_Folder>:$SHLIB_PATH`

2. Run the sem utility.

SEM Utility Syntax

The sem utility has the following syntax:

sem -h <hostname> -u <user> -p <password> -listcolddb | -defrost <archive>

-h

Specifies the hostname of the computer where the cold db files are stored.

-u

Specifies the user name used to authenticate with CA EEM server.

-p

Specifies the password for a user name used to authenticate with CA EEM server.

-listcolddb

Lists all the cold db files stored on the host computer.

-defrost <archive>

Defrosts the specified archive file.

-fips

Specifies that the sem utility uses FIPS-compliant algorithms.

Note: The sem utility must be used with -fips option if the CA EEM Server is configured in a FIPS-only mode.

The following table explains return values of the sem utility:

Return Value	Description
0	Success
1	Invalid arguments
2	Invalid username
3	Authentication Failed
4	Failed to list cold db files
5	Failed to defrost a cold db file
6	Initialization error

Defrost Cold DB Files

You must both restore and defrost the files from cold to warm state before you can run queries against the files and view live reports.

To restore and defrost cold db files

1. Copy the cold db files to the current calm_archive folder.
2. Run the sem utility from the command line to retrieve a list of all cold db files.

   sem -h <*hostname*> -u <*username*> -p <*password*> -listcolddb

   CA EEM Server in FIPS-only mode

   sem -h <*hostname*> -u <*username*> -p <*password*> -fips -listcolddb
3. Run the sem utility to defrost cold db files.

   sem -h <*hostname*> -u <*username*> -p <*password*> -defrost <*archive*>

   CA EEM Server in FIPS-only mode

   sem -h <*hostname*> -u <*username*> -p <*password*> -fips -defrost <*archive*>

   The cold db files are restored and defrosted.

More Information:

Overview (see page 101)