Red Hat Enterprise Linux 8

Configuring and managing Identity Management

Configuring, managing and maintaining Identity Management in Red Hat Enterprise Linux 8
Abstract

This documentation collection provides instructions on how to effectively configure, manage and maintain Identity Management on Red Hat Enterprise Linux 8.
# Table of Contents

**Providing Feedback on Red Hat Documentation** .............................................. 7

**Chapter 1. Logging Into Identity Management from the Command Line** ............ 8

1.1. Using Kinit to Log In to IDM Manually ................................. 8
   **Procedure** ........................................ 8

1.2. Destroying a User’s Active Kerberos Ticket ............................ 9
   **Procedure** ........................................ 9

1.3. Configuring an External System for Kerberos Authentication .......... 9
   **Prerequisites** ....................................... 9
   **Procedure** ........................................ 10
   **Additional resources** .................................. 10

**Chapter 2. Viewing, Starting and Stopping the Identity Management Services** ........ 11

2.1. Viewing the Status of IDM Services ........................................ 11

2.2. Starting and Stopping the Entire Identity Management Server: The Ipactl Utility ........ 12
   **Ipactl commands** ..................................... 12

2.3. Starting and Stopping an Individual Identity Management Service: The SystemCtl Utility 12
   **Useful systemctl commands** ................................ 13

**Chapter 3. Introduction to the IDM Command-Line Utilities** ...................... 14

   **Prerequisites** ........................................ 14

3.1. What is the IPA Command Line Interface .................................. 14

3.2. What is the IPA Help ........................................... 14

3.3. Using IPA Help Topics ......................................... 15
   **Procedure** ........................................ 15

3.4. Using IPA Help Commands ........................................ 15
   **Procedure** ........................................ 16
   **Additional resources** ...................................... 16

3.5. Structure of IPA Commands ............................................ 16

3.6. Using an IPA Command to Add a User Account to IDM ................. 17
   **Prerequisites** ........................................ 17

3.7. Using an IPA Command to Modify a User Account in IDM ............... 18
   **Prerequisites** ........................................ 18

3.8. How to Supply a List of Values to the IDM Utilities ................... 19

3.9. How to Use Special Characters with the IDM Utilities ................. 20

**Chapter 4. Searching Identity Management Entries from the Command Line** ........ 21

4.1. Overview of Listing IDM Entries ....................................... 21

4.2. Showing Details for a Particular Entry .................................. 21
   **Procedure** ........................................ 22

4.3. Adjusting the Search Size and Time Limit ................................ 22
   4.3.1. Adjusting the search size and time limit in the command line ....... 22
      **Procedure** ........................................ 22

4.3.2. Adjusting the search size and time limit in the Web UI ............... 23
      **Procedure** ........................................ 23

**Chapter 5. Accessing the IDM Web UI in a Web Browser** ............................ 25

5.1. What is the IDM Web UI ............................................. 25
5.2. WEB BROWSERS SUPPORTED FOR ACCESSING THE WEB UI

5.3. ACCESSING THE WEB UI

Procedure

CHAPTER 6. LOGGING IN TO IDM IN THE WEB UI: USING A KERBEROS TICKET

6.1. PREREQUISITES

6.2. KERBEROS AUTHENTICATION IN IDENTITY MANAGEMENT

6.3. USING KINIT TO LOG IN TO IDM MANUALLY

Procedure

6.4. CONFIGURING THE BROWSER FOR KERBEROS AUTHENTICATION

Procedure

6.5. LOGGING IN TO THE WEB UI USING A KERBEROS TICKET

Procedure

6.6. CONFIGURING AN EXTERNAL SYSTEM FOR KERBEROS AUTHENTICATION

Procedure

6.7. WEB UI LOGIN FOR ACTIVE DIRECTORY USERS

CHAPTER 7. LOGGING IN TO THE IDENTITY MANAGEMENT WEB UI USING ONE TIME PASSWORDS

7.1. PREREQUISITES

7.2. ONE TIME PASSWORD (OTP) AUTHENTICATION IN IDENTITY MANAGEMENT

7.3. ENABLING THE ONE TIME PASSWORD IN THE WEB UI

Procedure

7.4. ADDING OTP TOKENS IN THE WEB UI

Procedure

7.5. LOGGING INTO THE WEB UI WITH A ONE TIME PASSWORD

Procedure

7.6. SYNCHRONIZING OTP TOKENS USING THE WEB UI

Procedure

7.7. CHANGING EXPIRED PASSWORDS

Procedure

CHAPTER 8. PUBLIC KEY CERTIFICATES IN IDENTITY MANAGEMENT

8.1. CERTIFICATE AUTHORITIES IN IDM

8.2. COMPARISON OF CERTIFICATES AND KERBEROS

8.3. THE PROS AND CONS OF USING CERTIFICATES TO AUTHENTICATE USERS IN IDM

CHAPTER 9. CONVERTING CERTIFICATE FORMATS TO WORK WITH IDM

9.1. CERTIFICATE FORMATS AND ENCODINGS IN IDM

System configuration

Certificate encodings

User authentication

Useful certificate commands

9.2. CONVERTING AN EXTERNAL CERTIFICATE TO LOAD INTO AN IDM USER ACCOUNT

Procedure

9.2.1. Converting an external certificate in the IdM CLI and loading it into an IdM user account

Procedure

9.2.2. Converting an external certificate in the IdM web UI for loading into an IdM user account:

Procedure
9.3. PREPARING TO LOAD A CERTIFICATE INTO THE BROWSER
9.3.1. Exporting a certificate and private key from an NSS database into a PKCS #12 file
   Procedure
9.3.2. Combining certificate and private key PEM files into a PKCS #12 file
   Procedure

9.4. CERTIFICATE-RELATED COMMANDS AND FORMATS IN IDM

CHAPTER 10. CONFIGURING IDENTITY MANAGEMENT FOR SMART CARD AUTHENTICATION

10.1. CONFIGURING THE IDM SERVER FOR SMART CARD AUTHENTICATION
   Prerequisites
   Procedure
10.2. CONFIGURING THE IDM CLIENT FOR SMART CARD AUTHENTICATION
   Prerequisites
   Procedure
10.3. ADDING A CERTIFICATE TO A USER ENTRY IN IDM
   Prerequisites
   10.3.1. Adding a certificate to a user entry in the IdM Web UI
   10.3.2. Adding a certificate to a user entry in the IdM CLI
10.4. CONFIGURING THE BROWSER FOR SMART CARD AUTHENTICATION
   Prerequisites
   Procedure
10.5. LOGGING IN TO IDM WITH SMART CARDS
   Prerequisites
   Procedure

CHAPTER 11. CONFIGURING AUTHENTICATION WITH A CERTIFICATE STORED ON THE DESKTOP OF AN IDM CLIENT

11.1. CONFIGURING THE IDENTITY MANAGEMENT SERVER FOR CERTIFICATE AUTHENTICATION IN THE WEB UI
   Procedure
11.2. REQUESTING A NEW USER CERTIFICATE AND EXPORTING IT TO THE CLIENT
   Procedure
11.3. MAKING SURE THE CERTIFICATE AND USER ARE LINKED TOGETHER
11.4. CONFIGURING A BROWSER TO ENABLE CERTIFICATE AUTHENTICATION
   Procedure
11.5. AUTHENTICATING TO THE IDENTITY MANAGEMENT WEB UI WITH A CERTIFICATE AS AN IDENTITY MANAGEMENT USER
   Procedure
   Additional Resources
11.6. CONFIGURING AN IDM CLIENT TO ENABLE AUTHENTICATING TO THE CLI USING A CERTIFICATE
   Procedure

CHAPTER 12. CONFIGURING CERTIFICATE MAPPING RULES IN IDENTITY MANAGEMENT

12.1. CERTIFICATE MAPPING RULES FOR CONFIGURING AUTHENTICATION ON SMART CARDS
   12.1.1. Certificate mapping rules for trusts with Active Directory domains
   12.1.2. Components of an identity mapping rule in IdM
   12.1.3. Obtaining the issuer from a certificate for use in a matching rule
      Prerequisites
      Procedure
      Additional information
12.2. CONFIGURING CERTIFICATE MAPPING FOR USERS STORED IN IDM
   Prerequisites
   12.2.1. Adding a certificate mapping rule in IdM
   12.2.1.1. Adding a certificate mapping rule in the IdM web UI
12.2.1.2. Adding a certificate mapping rule in the IdM CLI
12.2.2. Adding certificate mapping data to a user entry in IdM
12.2.2.1. Adding certificate mapping data to a user entry in the IdM web UI
12.2.2.2. Adding certificate mapping data to a user entry in the IdM CLI

12.3. CONFIGURING CERTIFICATE MAPPING FOR USERS WHOSE AD USER ENTRY CONTAINS THE WHOLE CERTIFICATE
Prerequisites
12.3.1. Adding a certificate mapping rule for users whose AD entry contains whole certificates
12.3.1.1. Adding a certificate mapping rule in the IdM web UI
12.3.1.2. Adding a certificate mapping rule in the IdM CLI

12.4. CONFIGURING CERTIFICATE MAPPING IF AD IS CONFIGURED TO MAP USER CERTIFICATES TO USER ACCOUNTS
Prerequisites
12.4.1. Adding a certificate mapping rule if the trusted AD domain is configured to map user certificates
12.4.1.1. Adding a certificate mapping rule in the IdM web UI
12.4.1.2. Adding a certificate mapping rule in the IdM CLI
12.4.2. Checking certificate mapping data on the AD side

12.5. CONFIGURING CERTIFICATE MAPPING IF AD USER ENTRY CONTAINS NO CERTIFICATE OR MAPPING DATA
Prerequisites
12.5.1. Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data
12.5.1.1. Adding a certificate mapping rule in the IdM web UI
12.5.1.2. Adding a certificate mapping rule in the IdM CLI
12.5.2. Adding a certificate to an AD user’s ID override if the user entry in AD contains no certificate or mapping data
12.5.2.1. Adding a certificate to an AD user’s ID override in the IdM web UI
12.5.2.2. Adding a certificate to an AD user’s ID override in the IdM CLI

12.6. COMBINING SEVERAL IDENTITY MAPPING RULES INTO ONE

CHAPTER 13. USING IDM CA RENEWAL MASTER
13.1. IDM CA RENEWAL MASTER OVERVIEW
The role of the CA renewal master server
The role of certmonger on CA replicas
The correct functioning of IdM CA renewal master
13.2. CHANGING AND RESETTNG IDM CA RENEWAL MASTER
Prerequisites
Procedure
13.3. SWITCHING FROM AN EXTERNALLY TO SELF-SIGNED CA IN IDM
Prerequisites
Procedure
13.4. RENEWING THE IDM CA RENEWAL MASTER WITH AN EXTERNALLY-SIGNED CERTIFICATE
Prerequisites
Procedure

CHAPTER 14. OBTAINING AN IDM CERTIFICATE FOR A SERVICE USING CERTMONGER
14.1. CERTMONGER OVERVIEW
What certmonger does
Types of certificates certmonger tracks
Certmonger components
14.2. OBTAINING AN IDM CERTIFICATE FOR A SERVICE USING CERTMONGER
Prerequisites
Procedure
14.3. COMMUNICATION FLOW FOR CERTMONGER REQUESTING A SERVICE CERTIFICATE
14.4. VIEWING THE DETAILS OF A CERTIFICATE REQUEST TRACKED BY CERTMONGER
14.5. STARTING AND STOPPING CERTIFICATE TRACKING
Procedure
Additional information

14.6. RENEWING A CERTIFICATE MANUALLY
Procedure

14.7. MAKING CERTMONGER RESUME TRACKING OF IDM CERTIFICATES ON A CA REPLICA
Prerequisites
Procedure
Additional resources

CHAPTER 15. ENABLING AD USERS TO ADMINISTER IDM
15.1. ID OVERRIDES FOR AD USERS
Prerequisites
Procedure

15.2. USING ID OVERRIDES TO ENABLE AD USERS TO ADMINISTER IDM
Prerequisites
Procedure

15.3. MANAGING IDM COMMAND-LINE INTERFACE (CLI) AS AN AD USER

CHAPTER 16. USING CANONICALIZED DNS HOST NAMES IN IDM
16.1. ADDING AN ALIAS TO A HOST PRINCIPAL
Prerequisites
Procedure

16.2. ENABLING CANONICALIZATION OF HOST NAMES IN SERVICE PRINCIPALS ON CLIENTS
Prerequisites
Procedure

16.3. OPTIONS FOR USING HOST NAMES WITH DNS HOST NAME CANONICALIZATION ENABLED
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Please let us know how we could make it better. To do so:

- For simple comments on specific passages, make sure you are viewing the documentation in the Multi-page HTML format. Highlight the part of text that you want to comment on. Then, click the Add Feedback pop-up that appears below the highlighted text, and follow the displayed instructions.

- For submitting more complex feedback, create a Bugzilla ticket:
  1. Go to the Bugzilla website.
  2. As the Component, use Documentation.
  3. Fill in the Description field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.
  4. Click Submit Bug.
CHAPTER 1. LOGGING IN TO IDENTITY MANAGEMENT FROM THE COMMAND LINE

Identity Management (IdM) uses the Kerberos protocol to support single sign-on. Single sign-on means that the user enters the correct user name and password only once, and then accesses IdM services without the system prompting for the credentials again.

**IMPORTANT**

In IdM, the System Security Services Daemon (SSSD) automatically obtains a ticket-granting ticket (TGT) for a user after the user successfully logs in to the desktop environment on an IdM client machine with the corresponding Kerberos principal name. This means that after logging in, the user is not required to use the `kinit` utility to access IdM resources.

If you have cleared your Kerberos credential cache or your Kerberos TGT has expired, you need to request a Kerberos ticket manually to access IdM resources. The following sections present basic user operations when using Kerberos in IdM.

1.1. USING KINIT TO LOG IN TO IDM MANUALLY

This procedure describes using the `kinit` utility to authenticate to an Identity Management (IdM) environment manually. The `kinit` utility obtains and caches a Kerberos ticket-granting ticket (TGT) on behalf of an IdM user.

**NOTE**

Only use this procedure if you have destroyed your initial Kerberos TGT or if it has expired. As an IdM user, when logging onto your local machine you are also automatically logging in to IdM. This means that after logging in, you are not required to use the `kinit` utility to access IdM resources.

**Procedure**

1. To log in to IdM

   - under the user name of the user who is currently logged in on the local system, use `kinit` without specifying a user name. For example, if you are logged in as `example_user` on the local system:

     ```bash
     [example_user@server ~]$ kinit
     Password for example_user@EXAMPLE.COM:
     [example_user@server ~]$ kinit
     ``

     If the user name of the local user does not match any user entry in IdM, the authentication attempt fails:

     ```bash
     [example_user@server ~]$ kinit
     kinit: Client 'example_user@EXAMPLE.COM' not found in Kerberos database while getting initial credentials
     ``

   - using a Kerberos principal that does not correspond to your local user name, pass the required user name to the `kinit` utility. For example, to log in as the `admin` user:

     ```bash
     [example_user@server ~]$ kinit admin@EXAMPLE.COM
     Password for admin@EXAMPLE.COM:
     [example_user@server ~]$ kinit admin@EXAMPLE.COM
     ```
2. Optionally, to verify that the login was successful, use the `klist` utility to display the cached TGT. In the following example, the cache contains a ticket for the `example_user` principal, which means that on this particular host, only `example_user` is currently allowed to access IdM services:

```
$ klist
Ticket cache: KEYRING:persistent:0:0
Default principal: example_user@EXAMPLE.COM
Valid starting      Expires             Service principal
11/10/2019 08:35:45   11/10/2019 18:35:45   krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

1.2. DESTROYING A USER’S ACTIVE KERBEROS TICKET

This section describes how to clear the credentials cache that contains the user’s active Kerberos ticket.

**Procedure**

1. To destroy your Kerberos ticket:

   ```bash
   [example_user@server ~]$ kdestroy
   ```

2. Optionally, to check that the Kerberos ticket has been destroyed:

   ```bash
   [example_user@server ~]$ klist
   klist: Credentials cache keyring ’persistent:0:0’ not found
   ```

1.3. CONFIGURING AN EXTERNAL SYSTEM FOR KERBEROS AUTHENTICATION

This section describes how to configure an external system so that Identity Management (IdM) users can log in to IdM from the external system using their Kerberos credentials.

Enabling Kerberos authentication on external systems is especially useful when your infrastructure includes multiple realms or overlapping domains. It is also useful if the system has not been enrolled into any IdM domain through `ipa-client-install`.

To enable Kerberos authentication to IdM from a system that is not a member of the IdM domain, define an IdM-specific Kerberos configuration file on the external system.

**Prerequisites**

- The `krb5-workstation` package is installed on the external system. To find out whether the package is installed, use the following CLI command:

  ```bash
  # yum list installed krb5-workstation
  Installed Packages
  krb5-workstation.x86_64  1.16.1-19.el8  @BaseOS
  ```
Procedure

1. Copy the `/etc/krb5.conf` file from the IdM server to the external system. For example:

```
# scp /etc/krb5.conf root@externalsystem.example.com:/etc/krb5_ipa.conf
```

**WARNING**

Do not overwrite the existing `krb5.conf` file on the external system.

2. On the external system, set the terminal session to use the copied IdM Kerberos configuration file:

```
$ export KRB5_CONFIG=/etc/krb5_ipa.conf
```

The `KRB5_CONFIG` variable exists only temporarily until you log out. To prevent this loss, export the variable with a different file name.

3. Copy the Kerberos configuration snippets from the `/etc/krb5.conf.d/` directory to the external system.

Users on the external system can now use the `kinit` utility to authenticate against the IdM server.

**Additional resources**

- For details on Kerberos, see the `krb5.conf(5)`, `kinit(1)`, `klist(1)`, and `kdestroy(1)` man pages.
CHAPTER 2. VIEWING, STARTING AND STOPPING THE IDENTITY MANAGEMENT SERVICES

Identity Management (IdM) servers are Red Hat Enterprise Linux systems that work as domain controllers (DCs). A number of different services are running on IdM servers, most notably the Directory Server, Certificate Authority (CA), DNS, and Kerberos.

2.1. VIEWING THE STATUS OF IDM SERVICES

To view the status of the IdM services that are configured on your IdM server:

```
[root@server ~]# ipactl status
Directory Service: RUNNING
krb5kdc Service: RUNNING
kadmin Service: RUNNING
named Service: RUNNING
httpd Service: RUNNING
ntpd Service: RUNNING
pki-tomcatd Service: RUNNING
smb Service: RUNNING
winbind Service: RUNNING
ipa-otpd Service: RUNNING
ipa-dnskeysyncd Service: RUNNING
ipa: INFO: The ipactl command was successful
```

In the output above:

- The Kerberos service is divided into two parts, \texttt{krb5kdc} and \texttt{kadmin}. The \texttt{krb5kdc} service is the Kerberos version 5 Authentication service and Key Distribution Center (KDC) deamon. The \texttt{kadmin} service is the Kerberos V5 database administration program.

- The \texttt{named} service refers to the Internet domain name service (DNS).

- \texttt{pki} is the Command-Line Interface for accessing Certificate System services. The \texttt{pki-tomcatd} program handles Identity Management operations related to certificates.

The output of the \texttt{ipactl status} command on your server depends on your IdM configuration. For example, if an IdM deployment does not include a DNS server, the \texttt{named} service is not present in the list.

![NOTE]

You cannot use the IdM web UI to view the status of all the IdM services running on a particular IdM server. Kerberized services running on different servers can be viewed in the \texttt{Identity → Services} tab of the IdM web UI.

You can start or stop the entire server, or an individual service only.

To start, stop, or restart the entire IdM server, see:

- Section 2.2, “Starting and stopping the entire Identity Management server: the \texttt{ipactl} utility”

To start, stop, or restart an individual IdM service, see:
2.2. STARTING AND STOPPING THE ENTIRE IDENTITY MANAGEMENT SERVER: THE \texttt{IPACTL} UTILITY

Use the \texttt{ipactl} utility to stop, start, or restart the entire IdM server along with all the installed services. Using the \texttt{ipactl} utility ensures all services are stopped, started, or restarted in the appropriate order. You do not need to have a valid Kerberos ticket to run the \texttt{ipactl} commands.

\textbf{ipactl commands}

To start the entire IdM server:

\texttt{# ipactl start}

To stop the entire IdM server:

\texttt{# ipactl stop}

To restart the entire IdM server:

\texttt{# ipactl restart}

To show the status of all the services that make up IdM:

\texttt{# ipactl status}

\textbf{IMPORTANT}

You cannot use the IdM web UI to perform the \texttt{ipactl} commands.

2.3. STARTING AND STOPPING AN INDIVIDUAL IDENTITY MANAGEMENT SERVICE: THE \texttt{SYSTEMCTL} UTILITY

Changing IdM configuration files manually is generally not recommended. However, certain situations require that an administrator performs a manual configuration of specific services. In such situations, use the \texttt{systemctl} utility to stop, start, or restart an individual IdM service.

For example, use \texttt{systemctl} after customizing the Directory Server behavior, without modifying the other IdM services:

\texttt{# systemctl restart dirsrv@REALM-NAME.service}

Also, when initially deploying an IdM trust with Active Directory, modify the \texttt{/etc/sssds/sssd.conf} file, adding:

- specific parameters to tune the timeout configuration options in an environment where remote servers have a high latency
- specific parameters to tune the Active Directory site affinity
- overrides for certain configuration options that are not provided by the global IdM settings
To apply the changes you have made in the `/etc/sssd/sssd.conf` file:

```bash
# systemctl restart sssd.service
```

Running `systemctl restart sssd.service` is required because the System Security Services Daemon (SSSD) does not automatically re-read or re-apply its configuration.

Note that for changes that affect IdM identity ranges, a complete server reboot is recommended.

**IMPORTANT**

To restart multiple IdM domain services, always use `ipactl`. Because of dependencies between the services installed with the IdM server, the order in which they are started and stopped is critical. The `ipactl` utility ensures that the services are started and stopped in the appropriate order.

**Useful systemctl commands**

To start a particular IdM service:

```bash
# systemctl start name.service
```

To stop a particular IdM service:

```bash
# systemctl stop name.service
```

To restart a particular IdM service:

```bash
# systemctl restart name.service
```

To view the status of a particular IdM service:

```bash
# systemctl status name.service
```

**IMPORTANT**

You cannot use the IdM web UI to start or stop the individual services running on IdM servers. You can only use the web UI to modify the settings of a Kerberized service by navigating to **Identity → Services** and selecting the service.
CHAPTER 3. INTRODUCTION TO THE IDM COMMAND-LINE UTILITIES

The following sections describe the basics of using the Identity Management (IdM) command-line utilities.

Prerequisites

- Installed and accessible IdM server.
  For details, see Installing Identity Management.

- To use the IPA command line interface, authenticate to IdM with a valid Kerberos ticket.
  For details about obtaining a valid Kerberos ticket, see Logging in to Identity Management from the command line.

3.1. WHAT IS THE IPA COMMAND LINE INTERFACE

The IPA command line interface (CLI) is the basic command-line interface for Identity Management (IdM) administration.

It supports a lot of subcommands that are used to manage IdM, such as the `ipa user-add` command to add a new user.

IPA CLI allows you to:

- Add, manage, or remove users, groups, hosts and other objects in the network.
- Manage certificates.
- Search entries.
- Display and list objects.
- Set access rights.
- Get help with the correct command syntax.

3.2. WHAT IS THE IPA HELP

The IPA help is a built-in documentation system for the IdM server.

IPA command line interface (CLI) generates available help topics from loaded IdM plugin modules. If you want to run the IPA help successfully, you need to:

- Have an IdM server installed and running.
- Be authenticated with a valid Kerberos ticket.

Executing the `ipa help` command without options displays information about basic help usage and the most common command examples.

Executing help with options has the following syntax:

```
$ ipa help [TOPIC | COMMAND | topics | commands]
```
• [] – Brackets mean that all parameters are optional and you can write just `ipa help` and the command will be executed.

• | – The pipe character means or. Therefore, you can use TOPIC or COMMAND or topics or commands with the basic `ipa help` command.

• topics – You can run the command `ipa help topics` and it will execute correctly. The command displays a list of topics that are covered by IPA help, for example, `user, cert, server` and many others.

• TOPIC – The TOPIC with capital letters means variable, therefore, you can use the particular topic, for example, `ipa help user`.

• commands – You can run the command `ipa help commands` and it will execute correctly. The command displays a list of commands which are covered by the IPA help, for example, `user-add, ca-enable, server-show` and many others.

• COMMAND – The COMMAND with capital letters means variable, therefore, you can use the particular command, for example, `ipa help user-add`.

### 3.3. USING IPA HELP TOPICS

The following procedure helps you to understand using the IPA help in the command line interface.

**Procedure**

1. Open terminal and connect to the IdM server.

2. Enter `ipa help topics` to display a list of topics covered by help.

   ```
   $ ipa help topics
   ```

3. Select one of the topics and create a command according to the following pattern: `ipa help [topic_name]`, instead of the `topic_name` string, add one of the topics you listed in the previous step. In the example, we use the following topic: `user`

   ```
   $ ipa help user
   ```

4. If the IPA help command is too long and you cannot see the whole text, use the following syntax:

   ```
   $ ipa help user | less
   ```

   You can then scroll down and read the whole help.

The IPA CLI displays a help page for the `user` topic. After reading the overview, you can see many examples with patterns for working with topic commands.

### 3.4. USING IPA HELP COMMANDS

The following procedure helps you to understand creating the IPA help commands in the command line interface.
Procedure

1. Open terminal and connect to the IdM server.

2. Enter `ipa help commands` to display a list of commands covered by help.

   ```bash
   $ ipa help commands
   ```

3. Select one of the commands and create a help command according to the following pattern: `ipa help <COMMAND>`, instead of the `<COMMAND>` string, add one of the commands you listed in the previous step.

   ```bash
   $ ipa help user-add
   ```

Additional resources

- For details, see `man ipa` page.

3.5. STRUCTURE OF IPA COMMANDS

The IPA CLI distinguishes the following types of commands:

- Built-in commands – Built-in commands are all available in the IdM server.
- Plug-in provided commands

Structure of IPA commands allows you to manage various types of objects. For example:

- Users,
- Hosts,
- DNS records,
- Certificates,

and many others.

For most of these objects, the IPA CLI includes commands to:

- Add (`add`)
- Modify (`mod`)
- Delete (`del`)
- Search (`find`)
- Display (`show`)

Commands have the following structure:

- `ipa user-add`, `ipa user-mod`, `ipa user-del`, `ipa user-find`, `ipa user-show`
- `ipa host-add`, `ipa host-mod`, `ipa host-del`, `ipa host-find`, `ipa host-show`
You can create a user with the `ipa user-add [options]`, where `[options]` are optional. If you use just the `ipa user-add` command, the script asks you for details one by one.

To change an existing object, you need to define the object, therefore the command includes also object: `ipa user-mod USER_NAME [options].`

### 3.6. USING AN IPA COMMAND TO ADD A USER ACCOUNT TO IDM

The following describes adding a new user to the Identity Management database using command line.

**Prerequisites**

- You need to have administrator privileges to add user accounts to the IdM server.

**Procedure**

1. Open terminal and connect to the IdM server.
2. Enter the command for adding a new user:

   ```
   $ ipa user-add
   ```

   The command runs a script where you can add basic data necessary for creating a user account.
3. In the **First name:** field, enter the first name of the new user and press the **Enter** key.
4. In the **Last name:** field, enter the last name of the new user and press the **Enter** key.
5. In the **User login [suggested user name]:** enter the user name or just press the **Enter** key if the suggested user name works for you.

   User name must be unique for the whole IdM database. If an error occurs, that the user already exists, you need to start from the beginning with the `ipa user-add` command and try a different user name.

After you successfully added the user name, the user account has been added to the IdM database and the IPA CLI prints on the output the following log:

```
----------------------
Added user "euser"
----------------------
User login: euser
First name: Example
Last name: User
Full name: Example User
Display name: Example User
Initials: EU
Home directory: /home/euser
GECOS: Example User
Login shell: /bin/sh
Principal name: euser@IDM.EXAMPLE.COM
Principal alias: euser@IDM.EXAMPLE.COM
Email address: euser@idm.example.com
UID: 427200006
GID: 427200006
```
As you can see, a user password is not set to the user account. If you want to add also password, use the *ipa user-add* command in the following syntax:

```
$ ipa user-add --first=Example --last=User --password
```

The IPA CLI then asks you for adding or confirming a user name and password.

If the user has been already created, you can add only the password with the `ipa user-mod` command.

**Additional resources**
For more information about parameters, enter the following help command to the command line:

```
$ ipa help user-add
```

### 3.7. USING AN IPA COMMAND TO MODIFY A USER ACCOUNT IN IDM

You can change many parameters for each user account. For example, you can add a new password to the user.

Basic command syntax is different from the *user-add* syntax because you need to define the existing user account for which you want to perform changes, for example, add a password.

**Prerequisites**
- You need to have administrator privileges to modify user accounts in the IdM server.

**Procedure**

1. Open terminal and connect to the IdM server.

2. Enter the command for adding a password:

   ```
   $ ipa user-mod euser --password
   ```

   The command runs a script where you can add the new password.

3. Enter the new password and press the **Enter** key.

After you successfully added the user name, the user account has been added to the IdM database and the IPA CLI prints on the output the following log:

```
----------------------
Modified user "euser"
----------------------
User login: euser
First name: Example
Last name: User
Home directory: /home/euser
Principal name: euser@IDM.EXAMPLE.COM
Principal alias: euser@IDM.EXAMPLE.COM
Email address: euser@idm.example.com
```
The user password is now set for the account and the user can log into IdM.

Additional resources
For more information about parameters, enter the following help command to the command line:

```bash
$ ipa help user-mod
```

### 3.8. HOW TO SUPPLY A LIST OF VALUES TO THE IDM UTILITIES

Identity Management (IdM) stores values for multi-valued attributes in lists.

IdM supports the following methods of supplying multi-valued lists:

- Using the same command-line argument multiple times within the same command invocation:

  ```bash
  $ ipa permission-add --right=read --permissions=write --permissions=delete ...
  ```

- Alternatively, you can enclose the list in curly braces, in which case the shell performs the expansion:

  ```bash
  $ ipa permission-add --right={read,write,delete} ...
  ```

Examples above show a command `permission-add` which adds permissions to an object. The object is not mentioned in the example. Instead of `...` you need to add the object for which you want to add permissions.

When you update such multi-valued attributes from the command line, IdM completely overwrites the previous list of values with a new list. Therefore, when updating a multi-valued attribute, you must specify the whole new list, not just a single value you want to add.

In the command above, the list of permissions includes reading, writing and deleting. When you decide to update the list with the `permission-mod` command, you must add all values, otherwise those not mentioned will be deleted.

Example 1:– The `ipa permission-mod` command updates all previously added permissions.

```bash
$ ipa permission-mod --right=read --right=write --right=delete ...
```

or

```bash
$ ipa permission-mod --right={read,write,delete} ...
```

Example 2– The `ipa permission-mod` command deletes the `--right=delete` argument because it is not included in the command:

```bash
$ ipa permission-mod --right=read --right=write ...
```
3.9. HOW TO USE SPECIAL CHARACTERS WITH THE IDM UTILITIES

When passing command-line arguments that include special characters to the `ipa` commands, escape these characters with a backslash (`\`). For example, common special characters include angle brackets (`<` and `>`, ampersand (`&`), asterisk (`*`), or vertical bar (`|`).

For example, to escape an asterisk (`*`):

```bash
ipa certprofile-show certificate_profile --out=exported\*profile.cfg
```

Commands containing unescaped special characters do not work as expected because the shell cannot properly parse such characters.
CHAPTER 4. SEARCHING IDENTITY MANAGEMENT ENTRIES FROM THE COMMAND LINE

The following sections describe how to use IPA commands, which helps you to find or show objects.

4.1. OVERVIEW OF LISTING IDM ENTRIES

This section describes the `ipa *-find` commands, which can help you to search for a particular type of IdM entries.

To list all the `find` commands, use the following `ipa help` command:

```
$ ipa help commands | grep find
```

You may need to check if a particular user is included in the IdM database. You can then list all users with the following command:

```
$ ipa user-find
```

To list user groups whose specified attributes contain a keyword:

```
$ ipa group-find keyword
```

For example the `ipa group-find admin` command lists all groups whose names or descriptions include string `admin`:

```
----------------
3 groups matched
----------------
Group name: admins
Description: Account administrators group
GID: 427200002

Group name: editors
Description: Limited admins who can edit other users
GID: 427200002

Group name: trust admins
Description: Trusts administrators group
```

When searching user groups, you can also limit the search results to groups that contain a particular user:

```
$ ipa group-find --user=user_name
```

To search for groups that do not contain a particular user:

```
$ ipa group-find --no-user=user_name
```

4.2. SHOWING DETAILS FOR A PARTICULAR ENTRY
Use the `ipa *-show` command to display details about a particular IdM entry.

**Procedure**

- To display details about a host named `server.example.com`:

  ```
  $ ipa host-show server.example.com
  Host name: server.example.com
  Principal name: host/server.example.com@EXAMPLE.COM
  ...
  ```

4.3. ADJUSTING THE SEARCH SIZE AND TIME LIMIT

Some queries, such as requesting a list of IdM users, can return a very large number of entries. By tuning these search operations, you can improve the overall server performance when running the `ipa *-find` commands, such as `ipa user-find`, and when displaying corresponding lists in the Web UI.

**Search size limit**

Defines the maximum number of entries returned for a request sent to the server from a client’s CLI or from a browser accessing the IdM Web UI.

Default: 100 entries.

**Search time limit**

Defines the maximum time (in seconds) that the server waits for searches to run. Once the search reaches this limit, the server stops the search and returns the entries discovered in that time.

Default: 2 seconds.

If you set the values to `-1`, IdM will not apply any limits when searching.

**IMPORTANT**

Setting search size or time limits too high can negatively affect server performance.

4.3.1. Adjusting the search size and time limit in the command line

The following text describes adjusting search size and time limits in the command line:

- Globally
- For a specific entry

**Procedure**

1. To display current search time and size limits in CLI, use the `ipa config-show` command:

   ```
   $ ipa config-show
   Search time limit: 2
   Search size limit: 100
   ```

2. To adjust the limits globally for all queries, use the `ipa config-mod` command and add the `--searchrecordslimit` and `--searchtimelimit` options. For example:
3. To adjust the limits only for a specific query, add the `--sizelimit` or `--timelimit` options to the command. For example:

```
$ ipa user-find --sizelimit=200 --timelimit=120
```

### 4.3.2. Adjusting the search size and time limit in the Web UI

The following text describes adjusting search size and time limits in the IdM Web UI:

- Globally
- For a specific entry

**Procedure**

To adjust the limits globally for all queries:

1. Log in to the IdM Web UI.
2. Click **IPA Server**.
3. On the **IPA Server** tab, click **Configuration**.
4. Set the required values in the **Search Options** area.
   Default values are:
   - Search size limit: 100 entries
   - Search time limit: 2 seconds
5. Click **Save** at the top of the page.
After saving the values, search an entry and verify the result.
THE FOLLOWING SECTIONS PROVIDE AN OVERVIEW OF THE IDM (IDENTITY MANAGEMENT) WEB UI AND DESCRIBE HOW TO ACCESS IT.

5.1. WHAT IS THE IDM WEB UI

The IdM (Identity Management) Web UI is a web application for IdM administration, a graphical alternative to the IdM command line tools.

You can access the IdM Web UI as:

- **IdM users**: A limited set of operations depending on permissions granted to the user in the IdM server. Basically, active IdM users can log in to the IdM server and configure their own account. They cannot change settings of other users or the IdM server settings.

- **Administrators**: Full access rights to the IdM server.

- **Active Directory users**: A limited set of operations depending on permissions granted to the user.
  Active Directory users cannot be administrators for Identity Management.

5.2. WEB BROWSERS SUPPORTED FOR ACCESSING THE WEB UI

IdM (Identity Management) supports the following browsers for connecting to the Web UI:

- Mozilla Firefox 38 and later
- Google Chrome 46 and later

5.3. ACCESSING THE WEB UI

The following procedure describes the first logging in to the IdM (Identity Management) Web UI with a password.

After the first login you can configure your IdM server to authenticate with:

- Kerberos ticket
  For details, see Section 6.2, “Kerberos authentication in Identity Management”.

- Smart card
  For details, see Section 10.1, “Configuring the IdM server for smart card authentication”.

- One time password (OTP) – this can be combined with password and Kerberos authentication.
  For details, see Section 7.2, “One time password (OTP) authentication in Identity Management”.

Procedure

1. Type an IdM server URL into the browser address bar. The name will look similarly to the following example:

   https://server.example.com
You just need to change `server.example.com` with a DNS name of your IdM server.

This opens the IdM Web UI login screen in your browser.

- If the server does not respond or the login screen does not open, check the DNS settings on the IdM server to which you are connecting.

- If you use a self-signed certificate, the browser issues a warning. Check the certificate and accept the security exception to proceed with the login.
  
  To avoid security exceptions, install a certificate signed by a certificate authority.

2. On the Web UI login screen, enter the administrator account credentials you added during the IdM server installation.

   For details, see Installing an Identity Management server: With integrated DNS, with an integrated CA.

   You can enter your personal account credentials as well if they are already entered in the IdM server.

3. Click Log in.

   After the successful login, you can start configuring the IdM server.
### CHAPTER 5. ACCESSING THE IDM WEB UI IN A WEB BROWSER

#### Active users

<table>
<thead>
<tr>
<th>User</th>
<th>First name</th>
<th>Last name</th>
<th>Status</th>
<th>UID</th>
<th>Email address</th>
<th>Telephone Number</th>
<th>Job Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin</td>
<td>Administrator</td>
<td></td>
<td>Enabled</td>
<td>427200000</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Showing 1 to 1 of 1 entries.
CHAPTER 6. LOGGING IN TO IDM IN THE WEB UI: USING A KERBEROS TICKET

The following sections describe the initial configuration of your environment to enable Kerberos login to the IdM Web UI and accessing IdM using Kerberos authentication.

6.1. PREREQUISITES

- Installed IdM server in your network environment
  For details, see Installing Identity Management in Red Hat Enterprise Linux 8

6.2. KERBEROS AUTHENTICATION IN IDENTITY MANAGEMENT

Identity Management (IdM) uses the Kerberos protocol to support single sign-on. Single sign-on authentication allows you to provide the correct user name and password only once, and you can then access Identity Management services without the system prompting for credentials again.

The IdM server provides Kerberos authentication immediately after the installation if the DNS and certificate settings have been configured properly. For details, see Installing Identity Management.

To use Kerberos authentication on hosts, install:

- the IdM client
  For details, see Preparing the system for Identity Management client installation.

- the krb5conf package

6.3. USING KINIT TO LOG IN TO IDM MANUALLY

This procedure describes using the kinit utility to authenticate to an Identity Management (IdM) environment manually. The kinit utility obtains and caches a Kerberos ticket-granting ticket (TGT) on behalf of an IdM user.

**NOTE**

Only use this procedure if you have destroyed your initial Kerberos TGT or if it has expired. As an IdM user, when logging onto your local machine you are also automatically logging in to IdM. This means that after logging in, you are not required to use the kinit utility to access IdM resources.

**Procedure**

1. To log in to IdM

   - under the user name of the user who is currently logged in on the local system, use kinit without specifying a user name. For example, if you are logged in as example_user on the local system:

      ```
      [example_user@server ~]$ kinit
      Password for example_user@EXAMPLE.COM:
      [example_user@server ~]$ 
      ```
If the user name of the local user does not match any user entry in IdM, the authentication attempt fails:

```bash
[example_user@server ~]$ kinit
kinit: Client 'example_user@EXAMPLE.COM' not found in Kerberos database while getting initial credentials
```

- using a Kerberos principal that does not correspond to your local user name, pass the required user name to the `kinit` utility. For example, to log in as the `admin` user:

```bash
[example_user@server ~]$ kinit admin
Password for admin@EXAMPLE.COM:
[example_user@server ~]$ 
```

2. Optionally, to verify that the login was successful, use the `klist` utility to display the cached TGT. In the following example, the cache contains a ticket for the `example_user` principal, which means that on this particular host, only `example_user` is currently allowed to access IdM services:

```bash
$ klist
Ticket cache: KEYRING:persistent:0:0
Default principal: example_user@EXAMPLE.COM
Valid starting      Expires             Service principal
11/10/2019 08:35:45   11/10/2019 18:35:45   krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

### 6.4. CONFIGURING THE BROWSER FOR KERBEROS AUTHENTICATION

To enable authentication with a Kerberos ticket, you may need a browser configuration. The following steps help you to support Kerberos negotiation for accessing the IdM domain.

Each browser supports Kerberos in a different way and needs different set up. The IdM Web UI includes guidelines for the following browsers:

- Firefox
- Chrome

**Procedure**

1. Open the IdM Web UI login dialog in your web browser.

2. Click the link for browser configuration on the Web UI login screen.
3. Follow the steps on the configuration page.

![Configuration Page]

After the setup, turn back to the IdM Web UI and click Log in.

6.5. LOGGING IN TO THE WEB UI USING A KERBEROS TICKET

This procedure describes logging in to the IdM Web UI using a Kerberos ticket-granting ticket (TGT).

The TGT expires at a predefined time. The default time interval is 24 hours and you can change it in the IdM Web UI.

After the time interval expires, you need to renew the ticket:

- Using the kinit command.
- Using IdM login credentials in the Web UI login dialog.

Procedure

- Open the IdM Web UI.
  If Kerberos authentication works correctly and you have a valid ticket, you will be automatically authenticated and the Web UI opens.

  If the ticket is expired, it is necessary to authenticate yourself with credentials first. However, next time the IdM Web UI will open automatically without opening the login dialog.
If you see an error message **Authentication with Kerberos failed**, verify that your browser is configured for Kerberos authentication. See Section 6.4, "Configuring the browser for Kerberos authentication".

### 6.6. CONFIGURING AN EXTERNAL SYSTEM FOR KERBEROS AUTHENTICATION

This section describes how to configure an external system so that Identity Management (IdM) users can log in to IdM from the external system using their Kerberos credentials.

Enabling Kerberos authentication on external systems is especially useful when your infrastructure includes multiple realms or overlapping domains. It is also useful if the system has not been enrolled into any IdM domain through `ipa-client-install`.

To enable Kerberos authentication to IdM from a system that is not a member of the IdM domain, define an IdM-specific Kerberos configuration file on the external system.

#### Prerequisites

- The `krb5-workstation` package is installed on the external system. To find out whether the package is installed, use the following CLI command:

```bash
# yum list installed krb5-workstation
```

```
Installed Packages
krb5-workstation.x86_64 1.16.1-19.el8 @BaseOS
```

#### Procedure

1. Copy the `/etc/krb5.conf` file from the IdM server to the external system. For example:

```bash
# scp /etc/krb5.conf root@externalsystem.example.com:/etc/krb5_ipa.conf
```
2. On the external system, set the terminal session to use the copied IdM Kerberos configuration file:

   $ export KRB5_CONFIG=/etc/krb5_ipa.conf

   The KRB5_CONFIG variable exists only temporarily until you log out. To prevent this loss, export the variable with a different file name.

3. Copy the Kerberos configuration snippets from the /etc/krb5.conf.d/ directory to the external system.

4. Configure the browser on the external system, as described in Section 6.4, “Configuring the browser for Kerberos authentication”.

Users on the external system can now use the kinit utility to authenticate against the IdM server.

6.7. WEB UI LOGIN FOR ACTIVE DIRECTORY USERS

To enable Web UI login for Active Directory users, define an ID override for each Active Directory user in the default trust view. For example:

   [admin@server ~]$ ipa idoverrideuser-add 'Default Trust View' ad_user@ad.example.com
CHAPTER 7. LOGGING IN TO THE IDENTITY MANAGEMENT WEB UI USING ONE TIME PASSWORDS

Access to IdM Web UI can be secured using several methods. The basic one is password authentication.

To increase the security of password authentication, you can add a second step and require automatically generated one-time passwords (OTPs). The most common usage is to combine password connected with the user account and a time limited one time password generated by a hardware or software token.

The following sections help you to:

- Understand how the OTP authentication works in IdM.
- Configure OTP authentication on the IdM server.
- Create OTP tokens and synchronize them with the FreeOTP app in your phone.
- Authenticate to the IdM Web UI with the combination of user password and one time password.
- Re-synchronize tokens in the Web UI.

7.1. PREREQUISITES

- Accessing the IdM Web UI in a web browser

7.2. ONE TIME PASSWORD (OTP) AUTHENTICATION IN IDENTITY MANAGEMENT

One-time passwords bring an additional step to your authentication security. The authentication uses your password + an automatically generated one time password.

To generate one time passwords, you can use a hardware or software token. IdM supports both software and hardware tokens.

Identity Management supports the following two standard OTP mechanisms:

- The HMAC-Based One-Time Password (HOTP) algorithm is based on a counter. HMAC stands for Hashed Message Authentication Code.
- The Time-Based One-Time Password (TOTP) algorithm is an extension of HOTP to support time-based moving factor.

IMPORTANT

IdM does not support OTP logins for Active Directory trust users.

7.3. ENABLING THE ONE TIME PASSWORD IN THE WEB UI

The IdM Web UI allows you to configure hardware or software device to generate one-time passwords.

The one time password is entered just after the usual password in the dedicated field in the login dialog.
Only administrators can enable OTP authentication in the user settings.

**Prerequisites**

- Administration privileges

**Procedure**

1. Log in to the IdM Web UI with your username and password.

2. Open the **Identity → Users → Active user** tab.

3. Click your username to open the user settings.

4. In the **User authentication types**, select **Two factor authentication (password + OTP)**.

5. Click **Save**.

At this point, the OTP authentication is enabled on the IdM server.

Now you or users themselves need to assign a new token ID to the user account.

**7.4. ADDING OTP TOKENS IN THE WEB UI**

The following section helps you to add token to the IdM Web UI and to your software token generator.

**Prerequisites**

- Active user account on the IdM server.
- Administrator has enabled OTP for the particular user account in the IdM Web UI.
- A software device generating OTP tokens, for example FreeOTP.

**Procedure**

1. Log in to the IdM Web UI with your username and password.

2. To create the token in your mobile phone, open the **Authentication → OTP Tokens** tab.

3. Click **Add**.
4. In the **Add OTP token** dialog box, leave everything unfilled and click **Add**. At this stage, the IdM server creates a token with default parameters at the server and opens a page with a QR code.

5. Copy the QR code into your mobile phone.

6. Click **OK** to close the QR code.

Now you can generate one time passwords and log in with them to the IdM Web UI.

7.5. LOGGING INTO THE WEB UI WITH A ONE TIME PASSWORD

This procedure describes the first login into the IdM Web UI using a one time password (OTP).
Prerequisites

- OTP configuration enabled on the Identity Management server for the user account you are using for the OTP authentication. Administrators as well as users themselves can enable OTP. To enable the OTP configuration, see Section 7.3, “Enabling the one time password in the Web UI”

- A hardware or software device generating OTP tokens configured.

Procedure

1. In the Identity Management login screen, enter your user name or a user name of the IdM server administrator account.

2. Add the password for the user name entered above.

3. Generate a one time password on your device.

4. Enter the one time password right after the password (without space).

5. Click Log in.
   If the authentication fails, synchronize OTP tokens.
   If your CA uses a self-signed certificate, the browser issues a warning. Check the certificate and accept the security exception to proceed with the login.
   If the IdM Web UI does not open, verify the DNS configuration of your Identity Management server.

After successful login, the IdM Web UI appears.

7.6. SYNCHRONIZING OTP TOKENS USING THE WEB UI

If the login with OTP (One Time Password) fails, OTP tokens are not synchronized correctly.

The following text describes token re-synchronization.

Prerequisites

- A login screen opened.
• A device generating OTP tokens configured.

Procedure

1. On the IdM Web UI login screen, click Sync OTP Token.

2. In the login screen, enter your username and the Identity Management password.

3. Generate one time password and enter it in the First OTP field.

4. Generate another one time password and enter it in the Second OTP field.

5. Optionally, enter the token ID.

6. Click Sync OTP Token

After the successful synchronization, you can log in to the IdM server.

7.7. CHANGING EXPIRED PASSWORDS
Administrators of Identity Management can enforce you having to change your password at the next login. It means that you cannot successfully log in to the IdM Web UI until you change the password.

Password expiration can happen during your first login to the Web UI.

If the expiration password dialog appears, follow the instructions in the procedure.

**Prerequisites**

- A login screen opened.
- Active account to the IdM server.

**Procedure**

1. In the password expiration login screen, enter the user name.
2. Add the password for the user name entered above.
3. In the OTP field, generate a one time password, if you use the one time password authentication.
   If you do not have enabled the OTP authentication, leave the field empty.
4. Enter the new password twice for verification.
5. Click **Reset Password**.

After the successful password change, the usual login dialog displays. Log in with the new password.
CHAPTER 8. PUBLIC KEY CERTIFICATES IN IDENTITY MANAGEMENT

This chapter introduces X.509 public key certificates, which are used to authenticate users, hosts and services in Identity Management (IdM). In addition to authentication, X.509 certificates also enable digital signing and encryption to provide privacy, integrity and non-repudiation.

A certificate contains information about

- the subject that the certificate authenticates
- who has signed (validated) the certificate, that is the issuer
- the start and end of the validity of the certificate
- the valid uses of the certificate
- the public key of the subject

A message encrypted by the public key can only be decrypted by a corresponding private key. Although a certificate and the public key it includes can be made freely available, a user, host or machine must keep their private key secret.

8.1. CERTIFICATE AUTHORITIES IN IDM

Certificate authorities operate in a hierarchy of trust. In an IdM environment with an internal Certificate Authority (CA), all the IdM hosts, users and services trust certificates that have been signed by the CA. Apart from this root CA, IdM supports sub-CAs to which the root CA has granted the ability to sign certificates in their turn. Frequently, the certificates that such sub-CAs are able to sign are certificates of a specific kind, for example VPN certificates.

From the certificate point of view, there is no difference between being signed by a self-signed IdM CA and being signed externally.

The role of the CA is the following:

- It issues and verifies digital certificates
- It signs the certificate to prove that the certificate belongs to the user, host or service that presents it
- In an IdM environment with an internal CA, the CA which is the Certificate Renewal Master and which maintains the Certificate Revocation List (CRL) is the highest authority

8.2. COMPARISON OF CERTIFICATES AND KERBEROS

Certificates perform a similar function to that performed by Kerberos tickets. Kerberos is a computer network authentication protocol that works on the basis of tickets to allow nodes communicating over a non-secure network to prove their identity to one another in a secure manner. The following table shows a comparison of Kerberos and X.509 certificates:

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Kerberos</th>
<th>X.509</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
By default, Kerberos in Identity Management only ensures the identity of the communicating parties.

8.3. THE PROS AND CONS OF USING CERTIFICATES TO AUTHENTICATE USERS IN IDM

The advantages of using certificates to authenticate users in IdM include:

- A PIN that protects the private key on a smart card is typically less complex and easier to remember than a regular password.

- Depending on the device, a private key stored on a smart card cannot be exported. This provides additional security.

- Smart cards can make logout automatic: IdM can be configured to log out users when they remove the smart card from the reader.

- Stealing the private key requires actual physical access to a smart card, making smart cards secure against hacking attacks.

- Smart card authentication is two-factor authentication: it requires both something you have (the card) and something you know (the PIN).

- Smart cards are more flexible than passwords because they provide the keys that can be used for other purposes, such as encrypting email.

- Using smart cards use on shared machines that are IdM clients does not typically pose additional configuration problems for system administrators. In fact, smart card authentication is an ideal choice for shared machines.

The disadvantages of using certificates to authenticate users in IdM include:

- Users might lose or forget to bring their smart card or certificate and be effectively locked out.

- Mistyping a PIN multiple times might result in a card becoming locked.

- There is generally an intermediate step between request and authorization by some sort of security officer or approver. In IdM, the security officer or administrator must run the `ipa cert-request` command.

- Smart cards and readers tend to be vendor and driver specific: although a lot of readers can be used for different cards, a smart card of a specific vendor might not work in the reader of another vendor or in the type of a reader for which it was not designed.
The learning curve to certificates and smart cards might seem daunting to administrators with no experience in the area.
CHAPTER 9. CONVERTING CERTIFICATE FORMATS TO WORK WITH IDM

This user story describes how to make sure that you as an IdM system administrator are using the correct format of a certificate with specific IdM commands. This is useful, for example, in the following situations:

- You are loading an external certificate into a user profile. For details, see Section 9.2, “Converting an external certificate to load into an IdM user account”.

- You are using an external CA certificate when configuring the IdM server for smart card authentication or configuring the IdM client for smart card authentication so that users can authenticate to IdM using smart cards with certificates on them that have been issued by the external certificate authority.

- You are exporting a certificate from an NSS database into a pkcs #12 format that includes both the certificate and the private key. For details, see Section 9.3.1, “Exporting a certificate and private key from an NSS database into a PKCS #12 file”.

9.1. CERTIFICATE FORMATS AND ENCODINGS IN IDM

Certificate authentication including smart card authentication in IdM proceeds by comparing the certificate that the user presents with the certificate, or certificate data, that are stored in the user’s IdM profile.

System configuration

What is stored in the IdM profile is only the certificate, not the corresponding private key. During authentication, the user must also show that he is in possession of the corresponding private key. The user does that by either presenting a PKCS #12 file that contains both the certificate and the private key or by presenting two files: one that contains the certificate and the other containing the private key.

Therefore, processes such as loading a certificate into a user profile only accept certificate files that do not contain the private key.

Similarly, when a system administrator provides you with an external CA certificate, he will provide only the public data: the certificate without the private key. The ipa-advise utility for configuring the IdM server or the IdM client for smart card authentication expects the input file to contain the certificate of the external CA but not the private key.

Certificate encodings

There are two common certificate encodings: Privacy-enhanced Electronic Mail (PEM) and Distinguished Encoding Rules (DER). The base64 format is almost identical to the PEM format but it does not contain the -----BEGIN CERTIFICATE-----/-----END CERTIFICATE----- header and footer.

A certificate that has been encoded using DER is a binary X509 digital certificate file. As a binary file, the certificate is not human-readable. DER files sometimes use the .der filename extension, but files with the .crt and .cer filename extensions also sometimes contain DER certificates. DER files containing keys can be named .key.

A certificate that has been encoded using PEM Base64 is a human-readable file. The file contains ASCII (Base64) armored data prefixed with a "-----BEGIN ..." line. PEM files sometimes use the .pem filename extension, but files with the .crt and .cer filename extensions also sometimes contain PEM certificates. PEM files containing keys can be named .key.
Different **ipa** commands have different limitations regarding the types of certificates that they accept. For example, the **ipa user-add-cert** command only accepts certificates encoded in the **base64** format but **ipa-server-certinstall** accepts **PEM**, **DER**, **PKCS #7**, **PKCS #8** and **PKCS #12** certificates.

### Table 9.1. Certificate encodings

<table>
<thead>
<tr>
<th>Encoding format</th>
<th>Human-readable</th>
<th>Common filename extensions</th>
<th>Sample IdM commands accepting the encoding format</th>
</tr>
</thead>
<tbody>
<tr>
<td>PEM/base64</td>
<td>Yes</td>
<td>.pem, .crt, .cer</td>
<td>ipa user-add-cert, ipa-server-certinstall, ...</td>
</tr>
<tr>
<td>DER</td>
<td>No</td>
<td>.der, .crt, .cer</td>
<td>ipa-server-certinstall, ...</td>
</tr>
</tbody>
</table>

**Section 9.4, “Certificate-related commands and formats in IdM”** lists further **ipa** commands with the certificate formats that the commands accept.

#### User authentication

When using the web UI to access IdM, the user proves that he is in possession of the private key corresponding to the certificate by having both stored in the browser’s database.

When using the CLI to access IdM, the user proves that he is in possession of the private key corresponding to the certificate by one of the following methods:

- The user adds, as the value of the **X509_user_identity** parameter of the **kinit -X** command, the path to the smart card module that is connected to the smart card that contains both the certificate and the key:

  ```
  $ kinit -X X509_user_identity='PKCS11:opensc-pkcs11.so' idm_user
  ```

- The user adds two files as the values of the **X509_user_identity** parameter of the **kinit -X** command, one containing the certificate and the other the private key:

  ```
  $ kinit -X X509_user_identity='FILE: `/path/to/cert.pem,/path/to/cert.key`' idm_user
  ```

### Useful certificate commands

To view the certificate data, such as the subject and the issuer:

```
$ openssl x509 -noout -text -in ca.pem
```

To compare in which lines two certificates differ:

```
$ diff cert1.crt cert2.crt
```

To compare in which lines two certificates differ with the output displayed in two columns:

```
$ diff cert1.crt cert2.crt -y
```

### 9.2. CONVERTING AN EXTERNAL CERTIFICATE TO LOAD INTO AN IDM USER ACCOUNT
This section describes how to make sure that an external certificate is correctly encoded and formatted before adding it to a user entry.

**Prerequisites**

- If your certificate was issued by an Active Directory certificate authority and uses the **PEM** encoding, make sure that the **PEM** file has been converted into the **UNIX** format. To convert a file, use the `dos2unix` utility provided by the eponymous package.

### 9.2.1. Converting an external certificate in the IdM CLI and loading it into an IdM user account

The **IdM CLI** only accepts a **PEM** certificate from which the first and last lines (`-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----`) have been removed.

**Procedure**

1. Convert the certificate to the **PEM** format:
   - If your certificate is in the **DER** format:
     ```
     $ openssl x509 -in cert.crt -inform der -outform pem -out cert.pem
     ```
   - If your file is in the **PKCS #12** format, whose common filename extensions are `.pfx` and `.p12`, and contains a certificate, a private key, and possibly other data, extract the certificate using the `openssl pkcs12` utility. When prompted, enter the password protecting the private key stored in the file:
     ```
     $ openssl pkcs12 -in cert_and_key.p12 -clcerts -nokeys -out cert.pem
     ```
     Enter Import Password:

2. Obtain the administrator's credentials:

   ```
   $ kinit admin
   ```

3. Add the certificate to the user account using the **IdM CLI** following one of the following methods:
   - Remove the first and last lines (`-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----`) of the **PEM** file using the `sed` utility before adding the string to the **ipa user-add-cert** command:
     ```
     $ ipa user-add-cert some_user --certificate="$(sed -e '/BEGIN CERTIFICATE/d;/END CERTIFICATE/d' cert.pem)"
     ```
   - Copy and paste the contents of the certificate file without the first and last lines (`-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----`) into the **ipa user-add-cert** command:
     ```
     $ ipa user-add-cert some_user --
     certificate=MIIDlzCCAn+gAwIBAgIBATANBgkqhki...
     ```
NOTE
You cannot pass a PEM file containing the certificate as input to the `ipa user-add-cert` command directly, without first removing the first and last lines (-----BEGIN CERTIFICATE----- and -----END CERTIFICATE-----):

```bash
$ ipa user-add-cert some_user --cert=some_user_cert.pem
```

This command results in the "ipa: ERROR: Base64 decoding failed: Incorrect padding" error message.

4. Optionally, to check if the certificate was accepted by the system:

```bash
[idm_user@r8server]$ ipa user-show some_user
```

9.2.2. Converting an external certificate in the IdM web UI for loading into an IdM user account:

**Procedure**

1. Using the CLI, convert the certificate to the PEM format:
   - If your certificate is in the DER format:
     ```bash
     $ openssl x509 -in cert.crt -inform der -outform pem -out cert.pem
     ```
   - If your file is in the PKCS #12 format, whose common filename extensions are .pfx and .p12, and contains a certificate, a private key, and possibly other data, extract the certificate using the `openssl pkcs12` utility. When prompted, enter the password protecting the private key stored in the file:
     ```bash
     $ openssl pkcs12 -in cert_and_key.p12 -clcerts -nokeys -out cert.pem
     Enter Import Password:
     ```

2. Open the certificate in an editor and copy the contents. You can include the "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" header and footer lines but you do not have to, as both the PEM and base64 formats are accepted by the IdM web UI.

3. In the IdM web UI, log in as security officer.

4. Go to **Identity → Users → some_user**.

5. Click **Add** next to **Certificates**.

6. Paste the PEM-formatted contents of the certificate into the window that opens.

7. Click **Add**.

If the certificate was accepted by the system, you can see it listed among the **Certificates** in the user profile.

**9.3. PREPARING TO LOAD A CERTIFICATE INTO THE BROWSER**
Before importing a user certificate into the browser, make sure that the certificate and the corresponding private key are in a **PKCS #12** format. There are two common situations requiring extra preparatory work:

- The certificate is located in an NSS database. For details how to proceed in this situation, see Section 9.3.1, “Exporting a certificate and private key from an NSS database into a PKCS #12 file”.

- The certificate and the private key are in two separate **PEM** files. For details how to proceed in this situation, see Section 9.3.2, “Combining certificate and private key PEM files into a PKCS #12 file”.

Afterwards, to import both the CA certificate in the **PEM** format and the user certificate in the **PKCS #12** format into the browser, follow the procedures in Section 11.4, “Configuring a browser to enable certificate authentication” and Section 11.5, “Authenticating to the Identity Management Web UI with a Certificate as an Identity Management User”.

### 9.3.1. Exporting a certificate and private key from an NSS database into a PKCS #12 file

**Procedure**

1. Use the `pk12util` command to export the certificate from the NSS database to the **PKCS12** format. For example, to export the certificate with the `some_user` nickname from the NSS database stored in the `~/certdb` directory into the `~/some_user.p12` file:

   ```
   $ pk12util -d ~/certdb -o ~/some_user.p12 -n some_user
   Enter Password or Pin for "NSS Certificate DB": 
   Enter password for PKCS12 file:
   Re-enter password:
   pk12util: PKCS12 EXPORT SUCCESSFUL
   ```

2. Set appropriate permissions for the `.p12` file:

   ```
   # chmod 600 ~/some_user.p12
   ```

   Because the **PKCS #12** file also contains the private key, it must be protected to prevent other users from using the file. Otherwise, they would be able to impersonate the user.

### 9.3.2. Combining certificate and private key PEM files into a PKCS #12 file

This section describes how to combine a certificate and the corresponding key stored in separate **PEM** files into a **PKCS #12** file.

**Procedure**

- To combine a certificate stored in `certfile.cer` and a key stored in `certfile.key` into a `certfile.p12` file that contains both the certificate and the key:

  ```
  $ openssl pkcs12 -export -in certfile.cer -inkey certfile.key -out certfile.p12
  ```

### 9.4. CERTIFICATE-RELATED COMMANDS AND FORMATS IN IDM
Table **IdM certificate commands and formats** displays certificate-related commands in IdM with acceptable formats.

**Table 9.2. IdM certificate commands and formats**

<table>
<thead>
<tr>
<th>Command</th>
<th>Acceptable formats</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ipa user-add-cert some_user --certificate</code></td>
<td>base64 PEM certificate</td>
<td></td>
</tr>
<tr>
<td><code>ipa-server-certinstall</code></td>
<td>PEM and DER certificate; PKCS#7 certificate chain; PKCS#8 and raw private key; PKCS#12 certificate and private key</td>
<td></td>
</tr>
<tr>
<td><code>ipa-cacert-manage install</code></td>
<td>DER; PEM; PKCS#7</td>
<td></td>
</tr>
<tr>
<td><code>ipa-cacert-manage renew --external-cert-file</code></td>
<td>PEM and DER certificate; PKCS#7 certificate chain</td>
<td></td>
</tr>
<tr>
<td><code>ipa-ca-install --external-cert-file</code></td>
<td>PEM and DER certificate; PKCS#7 certificate chain</td>
<td></td>
</tr>
<tr>
<td><code>ipa cert-show &lt;cert serial&gt; --certificate-out /path/to/file.pem</code></td>
<td>N/A</td>
<td>Creates the PEM-encoded file.pem file with the certificate having the <code>&lt;cert_serial&gt;</code> serial number.</td>
</tr>
<tr>
<td><code>ipa cert-show &lt;cert serial&gt; --certificate-out /path/to/file.pem</code></td>
<td>N/A</td>
<td>Creates the PEM-encoded file.pem file with the certificate having the <code>&lt;cert_serial&gt;</code> serial number. If the <code>--chain</code> option is used, the PEM file contains the certificate including the certificate chain.</td>
</tr>
<tr>
<td><code>ipa cert-request --certificate-out=FILE /path/to/req.csr</code></td>
<td>N/A</td>
<td>Creates the req.csr file in the PEM format with the new certificate.</td>
</tr>
<tr>
<td><code>ipa cert-request --certificate-out=FILE /path/to/req.csr</code></td>
<td>N/A</td>
<td>Creates the req.csr file in the PEM format with the new certificate. If the <code>--chain</code> option is used, the PEM file contains the certificate including the certificate chain.</td>
</tr>
</tbody>
</table>
CHAPTER 10. CONFIGURING IDENTITY MANAGEMENT FOR SMART CARD AUTHENTICATION

Authentication based on smart cards is an alternative to passwords. User credentials are stored on the smart card in the form of a private key and a certificate, and special software and hardware is used to access them. The user places the smart card into a reader or a USB socket and supplies the PIN code for the smart card instead of providing his login and password.

Identity Management (IdM) supports smart card authentication with:

- User certificates issued by the IdM certificate authority
- User certificates issued by an external certificate authority

This user story shows how to set up smart card authentication in IdM for both types of certificates. In the user story, the `smartcard_ca.pem` CA certificate is the file containing the certificate of a trusted external certificate authority.

The user story contains the following modules:

- Section 10.1, “Configuring the IdM server for smart card authentication”
- Section 10.2, “Configuring the IdM client for smart card authentication”
- Section 10.3, “Adding a certificate to a user entry in IdM”
- Section 10.4, “Configuring the browser for smart card authentication”
- Section 10.5, “Logging in to IdM with smart cards”

10.1. CONFIGURING THE IDM SERVER FOR SMART CARD AUTHENTICATION

If you want to enable smart card authentication for users whose certificates have been issued by the certificate authority of the `EXAMPLE.ORG` domain, whose LDAP distinguished name (DN) is `CN=Certificate Authority,DC=EXAMPLE,DC=ORG`, then you need to obtain the certificate of the authority so that you can run it with the script configuring the IdM server. You can, for example, download the certificate from a web page whose certificate has been issued by the authority. For details, see Steps 1 - 4a in Section 11.4, “Configuring a browser to enable certificate authentication”.

To enable smart card authentication for IdM users who have been issued a certificate by the IdM Certificate Authority, obtain the CA certificate from the `/etc/ipa/ca.crt` file on the IdM server on which the IdM CA is running.

This section describes how to configure an IdM server for smart card authentication. First, obtain files with the CA certificates in the PEM format, then run the in-built `ipa-advertise` script. Finally, reload the system configuration.

Prerequisites

- You have root access to the IdM server.

Procedure

1. Create a directory in which you will do the configuration:
2. Navigate to the directory:

```
[root@server]# cd ~/SmartCard/
```

3. Obtain the relevant CA certificates stored in files in the PEM format: `.pem`, `.crt` and `.cer`. The IdM Certificate Authority certificate is located in the `/etc/ipa/ca.crt` file. For convenience, copy the certificates to the directory in which you want to do the configuration:

```
[root@server SmartCard]# cp /etc/ipa/ca.crt ~/SmartCard/
[root@server SmartCard]# cp /tmp/smartcard_ca.pem ~/SmartCard/
```

4. Optionally, if you use certificates of external certificate authorities, use the `openssl x509` utility to view the contents of the files in the PEM format to check that the **Issuer** and **Subject** values are correct:

```
[root@server SmartCard]# openssl x509 -noout -text -in smartcard_ca.pem | more
```

5. Generate a configuration script with the in-built `ipa-advise` utility, using the administrator’s privileges:

```
[root@server SmartCard]# kinit admin
[root@server SmartCard]# sudo ipa-advise config-server-for-smart-card-auth > config-server-for-smart-card-auth.sh
```

The `config-server-for-smart-card-auth.sh` script performs the following actions:

- It configures the IdM Apache HTTP server.
- It enables Public Key Cryptography for Initial Authentication in Kerberos (PKINIT) on the Key Distribution Center (KDC).
- It configures the IdM Web UI to accept smart card authorization requests.

6. Execute the script, adding the PEM files containing the CA certificates as arguments:

```
[root@server SmartCard]# chmod +x config-server-for-smart-card-auth.sh
[root@server SmartCard]# ./config-server-for-smart-card-auth.sh smartcard_ca.pem ca.crt
```

```
Ticket cache:KEYRING:persistent:0:0
Default principal: admin@IDM.EXAMPLE.COM
[...] Systemwide CA database updated.
```

```
The ipa-certupdate command was successful
```

7. Optionally, if the certificate authority that issued the user certificate does not provide any Online Certificate Status Protocol (OCSP) responder, you may need to disable OCSP check for authentication to the IdM Web UI:

   a. Set the `SSLOCSPEnable` parameter to `off` in the `/etc/httpd/conf.d/ssl.conf` file:

```
SSLOCSPEnable off
```
b. Restart the Apache daemon (httpd) for the changes to take effect immediately:

```
[root@server SmartCard]# sudo systemctl restart httpd
```

**WARNING**

Do not disable the OCSP check if you only use user certificates issued by the IdM CA. OCSP responders are part of IdM.

For instructions on how to keep the OCSP check enabled, and yet prevent a user certificate from being rejected by the IdM server if it does not contain the information about the location at which the CA that issued the user certificate listens for OCSP service requests, see the `SSLOCSPDefaultResponder` directive in Apache mod_ssl configuration options.

The server is now configured for smart card authentication.

**NOTE**

To enable smart card authentication in the whole topology, run the procedure on each IdM server.

### 10.2. CONFIGURING THE IDM CLIENT FOR SMART CARD AUTHENTICATION

This section describes how to configure IdM clients for smart card authentication. The procedure needs to be run on each IdM system, a client or a server, to which you want to connect while using a smart card for authentication. For example, to enable an `ssh` connection from host A to host B, the script needs to be run on host B.

As an administrator, run this procedure to enable smart card authentication using

- the `ssh` protocol
- the console login
- the Gnome Display Manager (GDM)
- the `su` command

This procedure is not required for authenticating to the IdM Web UI. Authenticating to the IdM Web UI involves two hosts, neither of which needs to be an IdM client:

- the machine - possibly outside of the IdM domain - on which the browser is running
- the IdM server on which `httpd` is running

The following procedure assumes that you are configuring smart card authentication on an IdM client that is not also an IdM master. For this reason you need two computers: an IdM master to generate the configuration script, and the IdM client on which to run the script.
Prerequisites

- Your IdM server has been configured for smart card authentication, as described in Section 10.1, "Configuring the IdM server for smart card authentication".
- You have root access to the IdM server and the IdM client.

Procedure

1. On an IdM master, generate a configuration script with `ipa-advise` using the administrator’s privileges:

   ```
   [root@server SmartCard]# kinit admin
   [root@server SmartCard]# ipa-advise config-client-for-smart-card-auth > config-client-for-smart-card-auth.sh
   ```

   The `config-client-for-smart-card-auth.sh` script performs the following actions:
   - It configures the smart card daemon.
   - It sets the system-wide trust store.
   - It configures the System Security Services Daemon (SSSD) to allow smart card logins to the desktop.

2. From the IdM master, copy the script to a directory of your choice on the IdM client machine:

   ```
   [root@server SmartCard]# scp config-client-for-smart-card-auth.sh root@client.idm.example.com:/root/SmartCard/
   Password: config-client-for-smart-card-auth.sh 100%  2419 3.5MB/s  00:00
   ```

3. From the IdM master, copy the CA certificate files in the PEM format for convenience to the same directory on the IdM client machine as used in the previous step:

   ```
   [root@server SmartCard]# scp {smartcard_ca.pem,ca.crt} root@client.idm.example.com:/root/SmartCard/
   Password: smartcard_ca.pem 100% 1237  9.6KB/s  00:00
   ca.crt 100% 2514 19.6KB/s  00:00
   ```

4. On the client machine, execute the script, adding the PEM files containing the CA certificates as arguments:

   ```
   [root@client SmartCard]# kinit admin
   [root@client SmartCard]# chmod +x config-client-for-smart-card-auth.sh
   [root@client SmartCard]# .config-client-for-smart-card-auth.sh smartcard_ca.pem ca.crt
   Ticket cache:KEYRING:persistent:0:0
   Default principal: admin@IDM.EXAMPLE.COM
   [ ]
   Systemwide CA database updated.
   The ipa-certupdate command was successful
   ```

The client is now configured for smart card authentication.
10.3. ADDING A CERTIFICATE TO A USER ENTRY IN IDM

This procedure describes how to add an external certificate to a user entry in IdM.

Instead of uploading the whole certificate, it is also possible to upload certificate mapping data to a user entry in IdM. User entries containing either full certificates or certificate mapping data can be used in conjunction with corresponding certificate mapping rules to facilitate the configuration of smart card authentication for system administrators. For details, see Chapter 12, Configuring certificate mapping rules in Identity Management.

NOTE

If the user’s certificate has been issued by the IdM Certificate Authority, the certificate is already stored in the user entry, and you can skip this section.

Prerequisites

- You have the certificate that you want to add to the user entry at your disposal.

10.3.1. Adding a certificate to a user entry in the IdM Web UI

1. Log into the IdM Web UI as an administrator if you want to add a certificate to another user. For adding a certificate to your own profile, you do not need the administrator’s credentials.

2. Navigate to Users → Active users → sc_user.

3. Find the Certificate option and click Add.

4. In the Command-Line Interface, display the certificate using the cat utility or a text editor:

   ```
   [user@client SmartCard]$ cat sc_user_certificate.pem
   ```

5. Copy and paste the certificate from the CLI into the window that has opened in the Web UI.

6. Click Add.
10.3.2. Adding a certificate to a user entry in the IdM CLI

1. Log into the IdM Web UI as an administrator if you want to add a certificate to another user:

   [user@client SmartCard]$ kinit admin

For adding a certificate to your own profile, you do not need the administrator’s credentials:

   [user@client SmartCard]$ kinit sc_user

2. Create an environment variable containing the certificate with the header and footer removed and concatenated into a single line, which is the format expected by the `ipa user-add-cert` command:

   [user@client SmartCard]$ export CERT=`openssl x509 -outform der -in sc_user_certificate.pem | base64 -w0`

3. Add the certificate to the profile of sc_user using the `ipa user-add-cert` command:

   [user@client SmartCard]$ ipa user-add-cert sc_user --certificate=$CERT

The `sc_user` entry now contains an external certificate.
10.4. CONFIGURING THE BROWSER FOR SMART CARD AUTHENTICATION

This module describes how to configure the Firefox browser for smart card authentication.

Identity Management supports the following browsers for connecting to the web UI:

- Mozilla Firefox 38 and later
- Google Chrome 46 and later

The following procedure shows how to configure the Mozilla Firefox 57.0.1 browser.

Prerequisites

- Your IdM server has been configured for smart card authentication, as described in Section 10.1, “Configuring the IdM server for smart card authentication”.
- A smart card is inserted into the USB slot of the host on which you want to configure the browser for smart card authentication.
- On the smart card, both the certificate and the private key of the IdM user are stored. For details about importing the certificate and the key on to the smart card, please refer to your smart card vendor’s documentation.
- The user entry in IdM contains the certificate that is stored on the smart card. For details about uploading a certificate into an IdM user’s user entry, see Section 10.3, “Adding a certificate to a user entry in IdM”.

Procedure

1. Open Firefox, click on Preferences.

Figure 10.2. Firefox preferences

![Firefox preferences](image)
2. Navigate to **Privacy & Security**.

3. Click on **Security Devices**.

**Figure 10.3. Security devices**

4. In the new **Device Manager** dialogue window, click on **Load**.

**Figure 10.4. Loading a security device**

5. In the new **Load PKCS#11 Device Driver** dialogue window, enter the module name, for example **OpenSC**. Enter the module filename. The module for OpenSC is located in the `/usr/lib64/opensc-pkcs11.so` file.
6. Optionally, check that module can log in to Firefox.

a. Click on **PIV Card Holder pin (PIV_II)** in the left pane and click **Log In** in the right pane.

Figure 10.6. Logging in with the security device

<table>
<thead>
<tr>
<th>Security Modules and Devices</th>
<th>Details</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>= NSS internal PKCS #11 Module</td>
<td>Status</td>
<td>Not Logged In</td>
</tr>
<tr>
<td>= Generic Crypto Services</td>
<td>Description</td>
<td>Yubico Yubikey 4 OTP+U2F+CCID 00:00</td>
</tr>
<tr>
<td>= Software Security Device</td>
<td>Manufacturer</td>
<td>Yubico</td>
</tr>
<tr>
<td>= Built-in Roots Module</td>
<td>HW Version</td>
<td>4.49</td>
</tr>
<tr>
<td>= System Trust</td>
<td>FW Version</td>
<td>0.0</td>
</tr>
<tr>
<td>= Default Trust</td>
<td>Label</td>
<td>PIV Card Holder pin (PIV_II)</td>
</tr>
<tr>
<td>= OpenSC</td>
<td>Manufacturer</td>
<td>PIV_II</td>
</tr>
<tr>
<td>= PIV Card Holder pin (PIV_II)</td>
<td>Serial Number</td>
<td>00000000</td>
</tr>
<tr>
<td></td>
<td>HW Version</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>FW Version</td>
<td>0.0</td>
</tr>
</tbody>
</table>

b. Enter the PIN of the smart card and click **OK**.

Figure 10.7. Entering the smart card PIN

If successful, you will see the **Log In** button grayed out.
7. Click **OK**.

Now your browser is ready for smart card authentication using the loaded security device.

### 10.5. LOGGING IN TO IDM WITH SMART CARDS

This section provides information about using smart cards for logging in to IdM Web UI.

**Prerequisites**

- The web browser is configured for using smart card authentication.
- The IdM server has been configured for smart card authentication.
- The certificate installed on your smart card is known to the IdM server.
- You need the PIN for the certificate access.
- The smart card has been plugged to the reader.

**Procedure**

1. Open the IdM Web UI in the browser.

2. Click on **Log In Using Certificate**
3. If the Password Required dialog box opens, add the PIN to unlock the certificate and click the OK button. The User Identification Request dialog box opens.

   If the smart card contains more than one certificate, select the certificate you want to use for authentication in the drop down list below Choose a certificate to present as identification

4. Click the OK button.

Now you are successfully logged in to the IdM Web UI.
CHAPTER 11. CONFIGURING AUTHENTICATION WITH A CERTIFICATE STORED ON THE DESKTOP OF AN IDM CLIENT

By configuring Identity Management (IdM), IdM system administrators can enable users to authenticate to the IdM web UI and command-line interface (CLI) using a certificate that a Certificate Authority (CA) has issued to the users.

The web browser can run on a system that is not part of the IdM domain.

This user story provides instructions on how to effectively configure and test logging into Identity Management web UI and CLI with a certificate stored on the desktop of an IdM client. In following this user story,

- you can skip Section 11.2, “Requesting a new user certificate and exporting it to the client” if the user you want to authenticate using a certificate already has a certificate;
- you can skip Section 11.3, “Making sure the certificate and user are linked together” if the user’s certificate has been issued by the IdM CA.

NOTE

Only Identity Management users can log into the web UI using a certificate. Active Directory users can log in with their user name and password.

11.1. CONFIGURING THE IDENTITY MANAGEMENT SERVER FOR CERTIFICATE AUTHENTICATION IN THE WEB UI

As an Identity Management (IdM) administrator, you can allow users to use certificates to authenticate to your IdM environment.

Procedure

As the Identity Management administrator:

1. On an Identity Management server, obtain administrator privileges and create a shell script to configure the server.
   a. Run the `ipa-advise config-server-for-smart-card-auth` command, and save its output to a file, for example `server_certificate_script.sh`:

```
# kinit admin
# ipa-advise config-server-for-smart-card-auth > server_certificate_script.sh
```

   b. Add execute permissions to the file using the `chmod` utility:

```
# chmod +x server_certificate_script.sh
```

2. On all the servers in the Identity Management domain, run the `server_certificate_script.sh` script
   a. with the path of the IdM Certificate Authority certificate, `/etc/ipa/ca.crt`, as input if the IdM CA is the only certificate authority that has issued the certificates of the users you want to enable certificate authentication for:

```
# ./server_certificate_script.sh /etc/ipa/ca.crt
```
b. with the paths leading to the relevant CA certificates as input if different external CAs signed the certificates of the users who you want to enable certificate authentication for:

```
# ./server_certificate_script.sh /tmp/ca1.pem /tmp/ca2.pem
```

**NOTE**

Do not forget to run the script on each new replica that you add to the system in the future if you want to have certificate authentication for users enabled in the whole topology.

### 11.2. REQUESTING A NEW USER CERTIFICATE AND EXPORTING IT TO THE CLIENT

As an Identity Management (IdM) administrator, you can create certificates for users in your IdM environment and export them to the IdM clients on which you want to enable certificate authentication for users.

**NOTE**

You can skip this section if the user you want to authenticate using a certificate already has a certificate.

**Procedure**

1. Optionally, create a new directory, for example `~/certdb/`, and make it a temporary certificate database. When asked, create an NSS Certificate DB password to encrypt the keys to the certificate to be generated in a subsequent step:

   ```
   # mkdir ~/certdb/
   # certutil -N -d ~/certdb/
   Enter a password which will be used to encrypt your keys.
   The password should be at least 8 characters long,
   and should contain at least one non-alphabetic character.
   Enter new password:
   Re-enter password:
   ```

2. Create the certificate signing request (CSR) and redirect the output to a file. For example, to create a CSR with the name `certificate_request.csr` for a 4096 bit certificate for the `idm_user` user in the `IDM.EXAMPLE.COM` realm, setting the nickname of the certificate private keys to `idm_user` for easy findability, and setting the subject to `CN=idm_user,O=IDM.EXAMPLE.COM`:

   ```
   # certutil -R -d ~/certdb/ -a -g 4096 -n idm_user -s "CN=idm_user,O=IDM.EXAMPLE.COM"
   > certificate_request.csr
   ```

3. When prompted, enter the same password that you entered when using `certutil` to create the temporary database. Then continue typing randomly until told to stop:

   ```
   Enter Password or Pin for "NSS Certificate DB":
   ```
A random seed must be generated that will be used in the creation of your key. One of the easiest ways to create a random seed is to use the timing of keystrokes on a keyboard.

To begin, type keys on the keyboard until this progress meter is full. DO NOT USE THE AUTOREPEAT FUNCTION ON YOUR KEYBOARD!

Continue typing until the progress meter is full:

4. Submit the certificate request file to the server. Specify the Kerberos principal to associate with the newly-issued certificate, the output file to store the certificate, and optionally the certificate profile. For example, to obtain a certificate of the IECUserRoles profile, a profile with added user roles extension, for the idm_user@IDM.EXAMPLE.COM principal, and save it in the ~/idm_user.pem file:

```
# ipa cert-request certificate_request.csr --principal=idm_user@IDM.EXAMPLE.COM --profile-id=IECUserRoles --certificate-out=~/idm_user.pem
```

5. Add the certificate to the NSS database. Use the -n option to set the same nickname that you used when creating the CSR previously so that the certificate matches the private key in the NSS database. The -t option sets the trust level. For details, see the certutil(1) man page. The -i option specifies the input certificate file. For example, to add to the NSS database a certificate with the idm_user nickname that is stored in the ~/idm_user.pem file in the ~/certdb/ database:

```
# certutil -A -d ~/certdb/ -n idm_user -t "P:\" -i ~/idm_user.pem
```

6. Verify that the key in the NSS database does not show (orphan) as its nickname. For example, to verify that the certificate stored in the ~/certdb/ database is not orphaned:

```
# certutil -K -d ~/certdb/  
< 0> rsa 5ad14d41463b87a095b1896cf0068ccc467df395  NSS Certificate DB: [replaceable]idm_user
```

7. Use the pk12util command to export the certificate from the NSS database to the PKCS12 format. For example, to export the certificate with the idm_user nickname from the /root/certdb NSS database into the ~/idm_user.p12 file:

```
# pk12util -d ~/certdb/ -o ~/idm_user.p12 -n idm_user
Enter Password or Pin for "NSS Certificate DB":
Enter password for PKCS12 file:
Re-enter password:
pk12util: PKCS12 EXPORT SUCCESSFUL
```

8. Transfer the certificate to the host on which you want the certificate authentication for idm_user to be enabled:

```
# scp ~/idm_user.p12 idm_user@client.idm.example.com:/home/idm_user/
```

9. On the host to which the certificate has been transferred, make the directory in which the .pkcs12 file is stored inaccessible to the 'other' group for security reasons:

```
# chmod o-rwx /home/idm_user/
```
10. For security reasons, remove the temporary NSS database and the .pkcs12 file from the server:

```
# rm ~/certdb/
# rm ~/idm_user.p12
```

### 11.3. MAKING SURE THE CERTIFICATE AND USER ARE LINKED TOGETHER

**NOTE**

You can skip this section if the user’s certificate has been issued by the IdM CA.

For certificate authentication to work, you need to make sure that the certificate is linked to the user that will use it to authenticate to Identity Management (IdM).

- If the certificate is provided by a Certificate Authority that is not part of your Identity Management environment, link the user and the certificate following the procedure described in [Linking User Accounts to Certificates](#).

- If the certificate is provided by Identity Management CA, the certificate is already automatically added in the user entry and you do not have to link the certificate to the user account. For details on creating a new certificate in IdM, see Section 11.2, “Requesting a new user certificate and exporting it to the client”.

### 11.4. CONFIGURING A BROWSER TO ENABLE CERTIFICATE AUTHENTICATION

For certificate authentication to work in your Identity Management web UI, you need to import the user and Certificate Authority (CA) certificates into the Mozilla Firefox or Google Chrome browser running on the host on which you want to enable certificate authentication. The host itself does not have to be part of the IdM domain.

Identity Management supports the following browsers for connecting to the web UI:

- Mozilla Firefox 38 and later
- Google Chrome 46 and later

The following procedure shows how to configure the Mozilla Firefox 57.0.1 browser.

**Procedure**

1. Open Firefox, then navigate to **Preferences → Privacy & Security**.
2. Click **View Certificates**.
View Certificates in Privacy and Security

3. In the **Your Certificates** tab, click **Import**. Locate and open the certificate of the user in the PKCS12 format, then click **OK** and **OK**.

4. Make sure that the Identity Management Certificate Authority is recognized by Firefox as a trusted authority:
   a. Save the IdM CA certificate locally:
      - Navigate to the IdM web UI by writing the name of your IdM server in the Firefox address bar. Click **Advanced** on the Insecure Connection warning page.
      - Insecure Connection
      - **Add Exception**. Click **View**.
      - View the Details of a Certificate
In the Details tab, highlight the Certificate Authority fields.

Exporting the CA Certificate

Click Export. Save the CA certificate, for example as the CertificateAuthority.crt file, then click Close, and Cancel.

b. Import the IdM CA certificate to Firefox as a trusted certificate authority certificate:
Open Firefox, navigate to Preferences and click **Privacy & Security**.

Click **Privacy & Security** section in Preferences.

Click **View Certificates**.

In the **Authorities** tab, click **Import**. Locate and open the CA certificate that you saved in the previous step in the **CertificateAuthority.crt** file. Trust the certificate to identify websites, then click **OK** and **OK**.

In the browser, navigate to the Identity Management web UI at, for example, https://server.idm.example.com/ipa/ui.

Click **Login Using Certificate**.

5. Continue to **Authenticating to the Identity Management Web UI with a Certificate as an Identity Management User**.

### 11.5. AUTHENTICATING TO THE IDENTITY MANAGEMENT WEB UI WITH A CERTIFICATE AS AN IDENTITY MANAGEMENT USER

This procedure describes authenticating as a user to the Identity Management (IdM) web UI using a certificate stored on the desktop of an Identity Management client.

**Procedure**

1. In the browser, navigate to the Identity Management web UI at, for example, https://server.idm.example.com/ipa/ui.

2. Click **Login Using Certificate**.

3. Continue to **Authenticating to the Identity Management Web UI with a Certificate as an Identity Management User**.
3. The user’s certificate should already be selected. Uncheck **Remember this decision**, then click **OK**.

You are now authenticated as the user who corresponds to the certificate.

**Additional Resources**

- For information about authenticating to the IdM web UI using a certificate stored on a smart card, see [Section 11.6, "Requesting a new user certificate and exporting it to the client"].

**11.6. CONFIGURING AN IDM CLIENT TO ENABLE AUTHENTICATING TO THE CLI USING A CERTIFICATE**

To make certificate authentication work for an IdM user in the Command Line Interface (CLI) of your IdM client, import the IdM user’s certificate and the private key to the IdM client. For details on creating and transferring the user certificate, see Section 11.2, "Requesting a new user certificate and exporting it to the client".

**Procedure**

- Log into the IdM client and have the .p12 file containing the user’s certificate and the private key ready. To obtain and cache the Kerberos ticket granting ticket (TGT), run the `kinit` command with the user’s principal, using the `-X` option with the `X509_username:/path/to/file.p12` attribute to specify where to find the user’s X509 identity information. For example, to obtain the TGT for `idm_user` using the user’s identity information stored in the `~/idm_user.p12` file:

  ```bash
  $ kinit -X X509_idm_user='PKCS12:/~idm_user.p12' idm_user
  ```

**NOTE**

The command also supports the .pem file format: `kinit -X X509_username='FILE:/path/to/cert.pem,/path/to/key' user_principal`
CHAPTER 12. CONFIGURING CERTIFICATE MAPPING RULES IN IDENTITY MANAGEMENT

12.1. CERTIFICATE MAPPING RULES FOR CONFIGURING AUTHENTICATION ON SMART CARDS

Certificate mapping rules are a convenient way of allowing users to authenticate using certificates in scenarios when the Identity Management (IdM) administrator does not have access to certain users’ certificates. This lack of access is typically caused by the fact that the certificates have been issued by an external certificate authority. A special use case is represented by certificates issued by the Certificate System of an Active Directory (AD) with which the IdM domain is in a trust relationship.

Certificate mapping rules are also convenient if the IdM environment is large with a lot of users using smart cards. In this situation, adding full certificates can be complicated. The subject and issuer are predictable in most scenarios and thus easier to add ahead of time than the full certificate. As a system administrator, you can create a certificate mapping rule and add certificate mapping data to a user entry even before a certificate is issued to a particular user. Once the certificate is issued, the user will be able to log in using the certificate even though the full certificate is not uploaded into his entry.

In addition, as certificates have to be renewed at regular intervals, certificate mapping rules reduce administrative overhead. When a user’s certificate gets renewed, the administrator does not have to update the user entry. For example, if the mapping is based on the Subject and Issuer values, and if the new certificate has the same subject and issuer as the old one, the mapping still applies. If, in contrast, the full certificate was used, then the administrator would have to upload the new certificate to the user entry to replace the old one.

To set up certificate mapping:

1. An administrator has to load the certificate mapping data (typically the issuer and subject) or the full certificate into a user account.

2. An administrator has to create a certificate mapping rule to allow successful logging into IdM for a user
   a. whose account contains a certificate mapping data entry
   b. whose certificate mapping data entry matches the information on the certificate

For details on the individual components that make up a mapping rule and how to obtain and use them, see Components of an identity mapping rule in IdM and Obtaining the issuer from a certificate for use in a matching rule.

Afterwards, when the end-user presents his certificate, stored either in the filesystem or on a smart card, he authenticates successfully.

12.1.1. Certificate mapping rules for trusts with Active Directory domains

This section outlines the different certificate mapping use cases that are possible if an IdM deployment is in a trust relationship with an Active Directory (AD) domain.

Certificate mapping rules are a convenient way to enable access to IdM resources for users who have smart card certificates that were issued by the trusted AD Certificate System. Depending on the AD configuration, the following scenarios are possible:
• If the certificate is issued by AD but the user and the certificate are stored in IdM, the mapping and the whole processing of the authentication request takes place on the IdM side. For details of configuring this scenario, see Configuring certificate mapping for users stored in IdM.

• If the user is stored in AD, the processing of the authentication request takes place in AD. There are three different subcases:
  - The AD user entry contains the whole certificate. For details how to configure IdM in this scenario, see Configuring certificate mapping for users whose AD user entry contains the whole certificate.
  - AD is configured to map user certificates to user accounts. In this case, the AD user entry does not contain the whole certificate but instead contains an attribute called altSecurityIdentities. For details how to configure IdM in this scenario, see Configuring certificate mapping if AD is configured to map user certificates to user accounts.
  - The AD user entry contains neither the whole certificate nor the mapping data. In this case, the only solution is to use the ipa idoverrideuser-add command to add the whole certificate to the AD user’s ID override in IdM. For details, see Configuring certificate mapping if AD user entry contains no certificate or mapping data.

12.1.2. Components of an identity mapping rule in IdM

This section describes the components of an identity mapping rule in IdM and how to configure them. Each component has a default value that you can override. You can define the components in either the web UI or the CLI. In the CLI, the identity mapping rule is created using the ipa certmaprule-add command.

Mapping rule

The mapping rule component associates (or maps) a certificate with one or more user accounts. The rule defines an LDAP search filter that associates a certificate with the intended user account. Certificates issued by different certificate authorities (CAs) might have different properties and might be used in different domains. Therefore, IdM does not apply mapping rules unconditionally, but only to the appropriate certificates. The appropriate certificates are defined using matching rules.

Note that if you leave the mapping rule option empty, the certificates are searched in the userCertificate attribute as a DER encoded binary file.

Define the mapping rule in the CLI using the --maprule option.

Matching rule

The matching rule component selects a certificate to which you want to apply the mapping rule. The default matching rule matches certificates with the digitalSignature key usage and clientAuth extended key usage.

Define the matching rule in the CLI using the --matchrule option.

Domain list

The domain list specifies the identity domains in which you want IdM to search the users when processing identity mapping rules. If you leave the option unspecified, IdM searches the users only in the local domain to which the IdM client belongs.

Define the domain in the CLI using the --domain option.

Priority

...
When multiple rules are applicable to a certificate, the rule with the highest priority takes precedence. All other rules are ignored.

- The lower the numerical value, the higher the priority of the identity mapping rule. For example, a rule with a priority 1 has higher priority than a rule with a priority 2.
- If a rule has no priority value defined, it has the lowest priority.

Define the mapping rule priority in the CLI using the `--priority` option.

Certificate Mapping Rule Example 1

To define, using the CLI, a certificate mapping rule called `simple_rule` that allows authentication for a certificate issued by the Smart Card CA of the EXAMPLE.ORG organisation as long as the Subject on that certificate matches a `certmapdata` entry in a user account in IdM:

```
# ipa certmaprule-add simple_rule --matchrule '<ISSUER>CN=Smart Card
CA,O=EXAMPLE.ORG' --maprule '(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>
{subject_dn!nss_x500})'
```

12.1.3. Obtaining the issuer from a certificate for use in a matching rule

This procedure describes how to obtain the issuer information from a certificate so that you can copy and paste it into the matching rule of a certificate mapping rule. To get the issuer format required by a matching rule, use the `openssl x509` utility.

Prerequisites

- You have the user certificate in a .pem or .crt format

Procedure

1. Obtain the user information from the certificate. Use the `openssl x509` certificate display and signing utility with:
   - the `-noout` option to prevent the output of an encoded version of the request
   - the `-issuer` option to output the issuer name
   - the `-in` option to specify the input filename to read the certificate from
   - the `-nameopt` option with the RFC2253 value to display the output with the most specific relative distinguished name (RDN) first

   If the input file contains an Identity Management certificate, the output of the command shows that the Issuer is defined using the Organisation information:

   ```
   # openssl x509 -noout -issuer -in idm_user.crt -nameopt RFC2253
   issuer=CN=Certificate Authority,O=REALM.EXAMPLE.COM
   ```

   If the input file contains an Active Directory certificate, the output of the command shows that the Issuer is defined using the Domain Component information:

   ```
   # openssl x509 -noout -issuer -in ad_user.crt -nameopt RFC2253
   issuer=CN=AD-WIN2012R2-CA,DC=AD,DC=EXAMPLE,DC=COM
   ```
2. Optionally, to create a new mapping rule in the CLI based on a matching rule which specifies that the certificate issuer must be the extracted **AD-WIN2012R2-CA** of the **ad.example.com** domain and the subject on the certificate must match the **certmapdata** entry in a user account in IdM:

```
# ipa certmaprule-add simple_rule --matchrule '<ISSUER>CN=AD-WIN2012R2-CA,DC=AD,DC=EXAMPLE,DC=COM' --maprule '(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})'
```

Additional information
For details about the **certmap** command, including information about the supported formats for the matching rule and the mapping rule, and an explanation of the priority and domain fields, see the **sss-certmap(5)** man page.

### 12.2. Configuring Certificate Mapping for Users Stored in IDM

This user story describes the steps a system administrator must take to enable certificate mapping in IdM if the user for whom certificate authentication is being configured is stored in IdM.

**Prerequisites**

- The user has an account in IdM.
- The administrator has either the whole certificate or the certificate mapping data to add to the user entry.

#### 12.2.1. Adding a Certificate Mapping Rule in IdM

This section describes how to set up a certificate mapping rule so that IdM users with certificates that match the conditions specified in the mapping rule and in their certificate mapping data entries can authenticate to IdM.

##### 12.2.1.1. Adding a Certificate Mapping Rule in the IdM Web UI

1. Log in to the IdM web UI as an administrator.
2. Navigate to **Authentication** → **Certificate Identity Mapping Rules** → **Certificate Identity Mapping Rules**.
3. Click **Add**.

![Figure 12.1. Adding a new certificate mapping rule in the IdM web UI](en.png)

4. Enter the rule name.

---

70
5. Enter the mapping rule. For example, to make IdM search for the **Issuer** and **Subject** entries in any certificate presented to them, and base its decision to authenticate or not on the information found in these two entries of the presented certificate:

(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})

6. Enter the matching rule. For example, to only allow certificates issued by the **Smart Card CA** of the **EXAMPLE.ORG** organization to authenticate users to IdM:

<ISSUER>CN=Smart Card CA,O=EXAMPLE.ORG

Figure 12.2. Entering the details for a certificate mapping rule in the IdM web UI

7. Click **Add** at the bottom of the dialog box to add the rule and close the box.

8. The System Security Services Daemon (SSSD) periodically re-reads the certificate mapping rules. To force the newly-created rule to be loaded immediately, restart SSSD:

```
# systemctl restart sssd
```

Now you have a certificate mapping rule set up that compares the type of data specified in the mapping rule that it finds on a smart card certificate with the certificate mapping data in your IdM user entries. Once it finds a match, it authenticates the matching user.

**12.2.1.2. Adding a certificate mapping rule in the IdM CLI**

1. Obtain the administrator’s credentials:

```
# kinit admin
```

2. Enter the mapping rule and the matching rule the mapping rule is based on. For example, to make IdM search for the **Issuer** and **Subject** entries in any certificate presented, and base its decision to authenticate or not on the information found in these two entries of the presented certificate, recognizing only certificates issued by the **Smart Card CA** of the **EXAMPLE.ORG** organization:

```
# ipa certmaprule-add rule_name --matchrule '<ISSUER>CN=Smart Card CA,O=EXAMPLE.ORG' --maprule '(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})
```

Added Certificate Identity Mapping Rule "rule_name"

Rule name: rule_name
Mapping rule: (ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})
Matching rule: <ISSUER>CN=Smart Card CA,O=EXAMPLE.ORG
Enabled: TRUE

3. The System Security Services Daemon (SSSD) periodically re-reads the certificate mapping rules. To force the newly-created rule to be loaded immediately, restart SSSD:

```
# systemctl restart sssd
```

Now you have a certificate mapping rule set up that compares the type of data specified in the mapping rule that it finds on a smart card certificate with the certificate mapping data in your IdM user entries. Once it finds a match, it authenticates the matching user.

### 12.2.2. Adding certificate mapping data to a user entry in IdM

This section describes how to enter certificate mapping data to an IdM user entry so that the user can authenticate using multiple certificates as long as they all contain the values specified in the certificate mapping data entry.

#### 12.2.2.1. Adding certificate mapping data to a user entry in the IdM web UI

1. Log into the IdM web UI as an administrator.

2. Navigate to **Users → Active users → idm_user**.

3. Find the **Certificate mapping data** option and click **Add**.

4. If you have the certificate of **idm_user** at your disposal:
   a. In the Command-Line Interface, display the certificate using the `cat` utility or a text editor:

```
[root@server ~]# cat idm_user_certificate.pem
-----BEGIN CERTIFICATE-----
MIIFFTCCA/2gAwIBAgIBEjANBgkqhkiG9w0BAQsFADA6MRgwFgYDVQQKDA9JRE0u
RVhBTVBMRS5DT00xHjAcBgNVAMMFUNlcnRzZmliYXRlEF1dGhvcml0eTAeFw0x
ODA5MDIxODE1MzlaMDA5MDIxODE1MzlaMCwxGDAWBgNVBAMMD0lETS5FWE
FN [...output truncated...]
```

   b. Copy the certificate.

   c. In the IdM web UI, click **Add** next to **Certificate** and paste the certificate into the window that opens up.
Alternatively, if you do not have the certificate of `idm_user` at your disposal but know the **Issuer** and the **Subject** of the certificate, check the radio button of **Issuer and subject** and enter the values in the two respective boxes.

Figure 12.4. Adding a user's certificate mapping data: issuer and subject

5. Click **Add**.

6. Optionally, if you have access to the whole certificate in the `.pem` format, verify that the user and certificate are linked:
   a. Use the `sss_cache` utility to invalidate the record of `idm_user` in the SSSD cache and force a reload of the `idm_user` information:

      ```bash
      # sss_cache -u idm_user
      ```
   b. Run the `ipa certmap-match` command with the name of the file containing the certificate of the IdM user:
The output confirms that now you have certificate mapping data added to `idm_user` and that a corresponding mapping rule defined in Adding a certificate mapping rule in IdM exists. This means that you can use any certificate that matches the defined certificate mapping data to authenticate as `idm_user`.

### 12.2.2.2. Adding certificate mapping data to a user entry in the IdM CLI

1. Obtain the administrator’s credentials:

   ```
   # kinit admin
   ```

2. If you have the certificate of `idm_user` at your disposal, add the certificate to the user account using the `ipa user-add-cert` command:

   ```
   # CERT=`cat idm_user_cert.pem | tail -n +2| head -n -1 | tr -d '
'
   # ipa user-add-certmapdata idm_user --certificate $CERT
   ```

   Alternatively, if you do not have the certificate of `idm_user` at your disposal but know the **Issuer** and the **Subject** of `idm_user`’s certificate:

   ```
   # ipa user-add-certmapdata idm_user --subject "O=EXAMPLE.ORG,CN=test" --issuer "CN=Smart Card CA,O=EXAMPLE.ORG"
   ```

   Added certificate mappings to user "idm_user"
   User login: idm_user
   Certificate mapping data: X509:<I>O=EXAMPLE.ORG,CN=Smart Card CA<C>N=test,O=EXAMPLE.ORG

3. Optionally, if you have access to the whole certificate in the `.pem` format, verify that the user and certificate are linked:

   a. Use the `sss_cache` utility to invalidate the record of `idm_user` in the SSSD cache and force a reload of the `idm_user` information:

      ```
      # sss_cache -u idm_user
      ```

   b. Run the `ipa certmap-match` command with the name of the file containing the certificate of the IdM user:

      ```
      # ipa certmap-match idm_user_cert.pem
      ```
The output confirms that now you have certificate mapping data added to `idm_user` and that a corresponding mapping rule defined in Adding a certificate mapping rule in IdM exists. This means that you can use any certificate that matches the defined certificate mapping data to authenticate as `idm_user`.

12.3. CONFIGURING CERTIFICATE MAPPING FOR USERS WHOSE AD USER ENTRY CONTAINS THE WHOLE CERTIFICATE

This user story describes the steps necessary for enabling certificate mapping in IdM if the IdM deployment is in trust with Active Directory (AD), the user is stored in AD and the user entry in AD contains the whole certificate.

Prerequisites

- The user does not have an account in IdM.
- The user has an account in AD which contains a certificate.
- The IdM administrator has access to data on which the IdM certificate mapping rule can be based.

12.3.1. Adding a certificate mapping rule for users whose AD entry contains whole certificates

12.3.1.1. Adding a certificate mapping rule in the IdM web UI

1. Log into the IdM web UI as an administrator.


3. Click Add.

4. Enter the rule name.

5. Enter the mapping rule. To have the whole certificate that is presented to IdM for authentication compared to what is available in AD:

   ```
   (userCertificate;binary={cert!bin})
   ```
6. Enter the matching rule. For example, to only allow certificates issued by the AD-ROOT-CA of the AD.EXAMPLE.COM domain to authenticate:

\[
{\text{"<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com"}}
\]

Figure 12.6. Certificate mapping rule for a user with a certificate stored in AD

7. Click Add.

8. The System Security Services Daemon (SSSD) periodically re-reads the certificate mapping rules. To force the newly-created rule to be loaded immediately, restart SSSD in the CLI:

```
# systemctl restart sssd
```

12.3.1.2. Adding a certificate mapping rule in the IdM CLI

1. Obtain the administrator’s credentials:

```
# kinit admin
```

2. Enter the mapping rule and the matching rule the mapping rule is based on. To have the whole certificate that is presented for authentication compared to what is available in AD, only allowing certificates issued by the AD-ROOT-CA of the AD.EXAMPLE.COM domain to authenticate:

```
# ipa certmaprule-add simpleADrule --matchrule '"<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com"' --maprule '(userCertificate;binary={cert!bin})' --domain ad.example.com
```

```
Added Certificate Identity Mapping Rule "simpleADrule"

Rule name: simpleADrule
Mapping rule: (userCertificate;binary={cert!bin})
Matching rule: "<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com"
Domain name: ad.example.com
Enabled: TRUE
```

3. The System Security Services Daemon (SSSD) periodically re-reads the certificate mapping rules. To force the newly-created rule to be loaded immediately, restart SSSD:

```
# systemctl restart sssd
```
12.4. CONFIGURING CERTIFICATE MAPPING IF AD IS CONFIGURED TO MAP USER CERTIFICATES TO USER ACCOUNTS

This user story describes the steps necessary for enabling certificate mapping in IdM if the IdM deployment is in trust with Active Directory (AD), the user is stored in AD and the user entry in AD contains certificate mapping data.

Prerequisites

- The user does not have an account in IdM.
- The user has an account in AD which contains the `altSecurityIdentities` attribute, the AD equivalent of the IdM `certmapdata` attribute.
- The IdM administrator has access to data on which the IdM certificate mapping rule can be based.

12.4.1. Adding a certificate mapping rule if the trusted AD domain is configured to map user certificates

12.4.1.1. Adding a certificate mapping rule in the IdM web UI

1. Log into the IdM web UI as an administrator.
2. Navigate to **Authentication** → **Certificate Identity Mapping Rules** → **Certificate Identity Mapping Rules**.
3. Click **Add**.

![Figure 12.7. Adding a new certificate mapping rule in the IdM web UI](image)

4. Enter the rule name.
5. Enter the mapping rule. For example, to make AD DC search for the `Issuer` and `Subject` entries in any certificate presented, and base its decision to authenticate or not on the information found in these two entries of the presented certificate:

   ```plaintext
   (altSecurityIdentities=X509:<(issuer_dn!ad_x500)<S>{subject_dn!ad_x500})
   ```

6. Enter the matching rule. For example, to only allow certificates issued by the **AD-ROOT-CA** of the **AD.EXAMPLE.COM** domain to authenticate users to IdM:

   ```plaintext
   <ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
   ```

7. Enter the domain:
Figure 12.8. Certificate mapping rule if AD is configured for mapping

8. Click Add.

9. The System Security Services Daemon (SSSD) periodically re-reads the certificate mapping rules. To force the newly-created rule to be loaded immediately, restart SSSD in the CLI:

```
# systemctl restart sssd
```

12.4.1.2. Adding a certificate mapping rule in the IdM CLI

1. Obtain the administrator’s credentials:

```
# kinit admin
```

2. Enter the mapping rule and the matching rule the mapping rule is based on. For example, to make AD search for the Issuer and Subject entries in any certificate presented, and only allow certificates issued by the AD-ROOT-CA of the AD.EXAMPLE.COM domain:

```
# ipa certmaprule-add ad_configured_for_mapping_rule --matchrule '<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com' --maprule '(altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500})' --domain=ad.example.com
```

```
Added Certificate Identity Mapping Rule "ad_configured_for_mapping_rule"
```

```
Rule name: ad_configured_for_mapping_rule
Mapping rule: (altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500})
Matching rule: <ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
Domain name: ad.example.com
Enabled: TRUE
```

3. The System Security Services Daemon (SSSD) periodically re-reads the certificate mapping rules. To force the newly-created rule to be loaded immediately, restart SSSD:

```
# systemctl restart sssd
```
12.4.2. Checking certificate mapping data on the AD side

The `altSecurityIdentities` attribute is the Active Directory (AD) equivalent of `certmapdata` user attribute in IdM. When configuring certificate mapping in IdM in the scenario when a trusted AD domain is configured to map user certificates to user accounts, the IdM system administrator needs to check that the `altSecurityIdentities` attribute is set correctly in the user entries in AD.

To check that AD contains the right information for the user stored in AD, use the `ldapsearch` command.

- For example, to check with the `adserver.ad.example.com` server that the `altSecurityIdentities` attribute is set in the user entry of `ad_user` and that the matchrule stipulates that the certificate that `ad_user` uses to authenticate to AD was issued by `AD-ROOT-CA` of the `ad.example.com` domain and that the subject is `<S>DC=com,DC=example,DC=ad,CN=Users,CN=ad_user`:

  ```
  $ ldapsearch -o ldif-wrap=no -LLL -h adserver.ad.example.com -p 389 -D cn=Administrator,cn=users,dc=ad,dc=example,dc=com -W -b cn=users,dc=ad,dc=example,dc=com "(cn=ad_user)"
  altSecurityIdentities
  Enter LDAP Password:
  dn: CN=ad_user,CN=Users,DC=ad,DC=example,DC=com
  altSecurityIdentities: X509:<I>DC=com,DC=example,DC=ad,CN=AD-ROOT-CA<S>DC=com,DC=example,DC=ad,CN=Users,CN=ad_user
  ```

12.5. CONFIGURING CERTIFICATE MAPPING IF AD USER ENTRY CONTAINS NO CERTIFICATE OR MAPPING DATA

This user story describes the steps necessary for enabling certificate mapping in IdM if the IdM deployment is in trust with Active Directory (AD), the user is stored in AD and the user entry in AD contains neither the whole certificate nor certificate mapping data.

**Prerequisites**

- The user does not have an account in IdM.
- The user has an account in AD which contains neither the whole certificate nor the `altSecurityIdentities` attribute, the AD equivalent of the IdM `certmapdata` attribute.
- The IdM administrator has the whole AD user certificate to add to the AD user’s `user ID override` in IdM.

12.5.1. Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data

12.5.1.1. Adding a certificate mapping rule in the IdM web UI

1. Log into the IdM web UI as an administrator.
3. Click **Add**.
4. Enter the rule name.

5. Enter the mapping rule. To have the whole certificate that is presented to IdM for authentication compared to the certificate stored in the user ID override entry of the AD user entry in IdM:

   (userCertificate;binary={cert!bin})

6. Enter the matching rule. For example, to only allow certificates issued by the AD-ROOT-CA of the AD.EXAMPLE.COM domain to authenticate:

   <ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com

7. Enter the domain name. For example, to search for users in the ad.example.com domain:

8. Click Add.

9. The System Security Services Daemon (SSSD) periodically re-reads the certificate mapping rules. To force the newly-created rule to be loaded immediately, restart SSSD in the CLI:

   # systemctl restart sssd

12.5.1.2. Adding a certificate mapping rule in the IdM CLI

1. Obtain the administrator’s credentials:

   # kinit admin
2. Enter the mapping rule and the matching rule the mapping rule is based on. To have the whole certificate that is presented for authentication compared to the certificate stored in the user ID override entry of the AD user entry in IdM, only allowing certificates issued by the AD-ROOT-CA of the AD.EXAMPLE.COM domain to authenticate:

```
# ipa certmaprule-add simpleADrule --matchrule '<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com' --maprule '(userCertificate;binary={cert!bin})' --domain ad.example.com
```

Added Certificate Identity Mapping Rule "simpleADrule"

<table>
<thead>
<tr>
<th>Rule name: simpleADrule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapping rule: (userCertificate;binary={cert!bin})</td>
</tr>
<tr>
<td>Matching rule: &lt;ISSUER&gt;CN=AD-ROOT-CA,DC=ad,DC=example,DC=com</td>
</tr>
<tr>
<td>Domain name: ad.example.com</td>
</tr>
<tr>
<td>Enabled: TRUE</td>
</tr>
</tbody>
</table>

3. The System Security Services Daemon (SSSD) periodically re-reads the certificate mapping rules. To force the newly-created rule to be loaded immediately, restart SSSD:

```
# systemctl restart sssd
```

12.5.2. Adding a certificate to an AD user’s ID override if the user entry in AD contains no certificate or mapping data

12.5.2.1. Adding a certificate to an AD user’s ID override in the IdM web UI

1. Navigate to Identity → ID Views → Default Trust View.

2. Click Add.

Figure 12.11. Adding a new user ID override in the IdM web UI

3. In the User to override field, enter ad_user@ad.example.com.

4. Copy and paste the certificate of ad_user into the Certificate field.
5. Click **Add**.

6. Optionally, verify that the user and certificate are linked:
   
   a. Use the `sss_cache` utility to invalidate the record of `ad_user@ad.example.com` in the SSSD cache and force a reload of the `ad_user@ad.example.com` information:

   ```
   # sss_cache -u ad_user@ad.example.com
   ```

   b. Run the `ipa certmap-match` command with the name of the file containing the certificate of the AD user:

   ```
   # ipa certmap-match ad_user_cert.pem
   1 user matched
   Domain: AD.EXAMPLE.COM
   User logins: ad_user@ad.example.com
   Number of entries returned 1
   ```

   The output confirms that you have certificate mapping data added to `ad_user@ad.example.com` and that a corresponding mapping rule defined in Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data exists. This means that you can use any certificate that matches the defined certificate mapping data to authenticate as `ad_user@ad.example.com`.
12.5.2.2. Adding a certificate to an AD user’s ID override in the IdM CLI

1. Obtain the administrator’s credentials:
   
   ```
   # kinit admin
   ```

2. Add the certificate of `ad_user@ad.example.com` to the user account using the `ipa idoverrideuser-add-cert` command:
   
   ```
   # CERT=`cat ad_user_cert.pem | tail -n +2| head -n -1 | tr -d '
'
   # ipa idoverrideuser-add-cert ad_user@ad.example.com --certificate $CERT
   ```

3. Optionally, verify that the user and certificate are linked:
   
   a. Use the `sss_cache` utility to invalidate the record of `ad_user@ad.example.com` in the SSSD cache and force a reload of the `ad_user@ad.example.com` information:
      
      ```
      # sss_cache -u ad_user@ad.example.com
      ```
   
   b. Run the `ipa certmap-match` command with the name of the file containing the certificate of the AD user:
      
      ```
      # ipa certmap-match ad_user_cert.pem
      1 user matched
      Domain: AD.EXAMPLE.COM
      User logins: ad_user@ad.example.com
      Number of entries returned 1
      ```

The output confirms that you have certificate mapping data added to `ad_user@ad.example.com` and that a corresponding mapping rule defined in Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data exists. This means that you can use any certificate that matches the defined certificate mapping data to authenticate as `ad_user@ad.example.com`.

12.6. COMBINING SEVERAL IDENTITY MAPPING RULES INTO ONE

To combine several identity mapping rules into one combined rule, use the `|` (or) character to precede the individual mapping rules, and separate them using `()` brackets, for example:

**Certificate Mapping Filter Example 1**

```bash
$ ipa certmaprule-add ad_cert_for_ipa_and_ad_users \ --maprule='|((ipacertmapdata=X509:<I>(issuer_dn!nss_x500)<S>{subject_dn!nss_x500})(altSecurityIdentities=X509:<I>(issuer_dn!ad_x500)<S>{subject_dn!ad_x500)))'| --matchrule='<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com' \ --domain=ad.example.com
```

In the above example, the filter definition in the `--maprule` option includes these criteria:
• `ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500}` is a filter that links the subject and issuer from a smart card certificate to the value of the `ipacertmapdata` attribute in an IdM user account, as described in Adding a certificate mapping rule in IdM.

• `altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500}` is a filter that links the subject and issuer from a smart card certificate to the value of the `altSecurityIdentities` attribute in an AD user account, as described in Adding a certificate mapping rule if the trusted AD domain is configured to map user certificates.

• The addition of the `--domain=ad.example.com` option means that users mapped to a given certificate are not only searched in the local `idm.example.com` domain but also in the `ad.example.com` domain.

The filter definition in the `--maprule` option accepts the logical operator `|` (or), so that you can specify multiple criteria. In this case, the rule maps all user accounts that meet at least one of the criteria.

**Certificate Mapping Filter Example 2**

```
$ ipa certmaprule-add ipa_cert_for_ad_users
    --maprule='(|((userCertificate;binary={cert!bin})(ipacertmapdata=X509:<I>
    {issuer_dn!nss_x500}<S>{subject_dn!nss_x500})(altSecurityIdentities=X509:<I>
    {issuer_dn!ad_x500}<S>{subject_dn!ad_x500}))')
    --matchrule='<ISSUER>CN=Certificate Authority,O=REALM.EXAMPLE.COM'
    --domain=idm.example.com --domain=ad.example.com
```

In the above example, the filter definition in the `--maprule` option includes these criteria:

• `userCertificate;binary={cert!bin}` is a filter that returns user entries that include the whole certificate. For AD users, creating this type of filter is described in detail in Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data.

• `ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500}` is a filter that links the subject and issuer from a smart card certificate to the value of the `ipacertmapdata` attribute in an IdM user account, as described in Adding a certificate mapping rule in IdM.

• `altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500}` is a filter that links the subject and issuer from a smart card certificate to the value of the `altSecurityIdentities` attribute in an AD user account, as described in Adding a certificate mapping rule if the trusted AD domain is configured to map user certificates.

The filter definition in the `--maprule` option accepts the logical operator `|` (or), so that you can specify multiple criteria. In this case, the rule maps all user accounts that meet at least one of the criteria.
13.1. IDM CA RENEWAL MASTER OVERVIEW

The Identity Management (IdM) certificate authority (CA) renewal master server maintains and renews IdM system certificates, ensuring nondisruptive IdM deployments.

IdM system certificates include:

- IdM CA certificate
- OCSP signing certificate
- IdM CA subsystem certificates
- IdM CA audit signing certificate
- IdM renewal agent (RA) certificate
- KRA transport and storage certificates

What characterizes system certificates is that their keys are shared by all CA replicas. The IdM service certificates (for example the LDAP, HTTP and PKINIT certificates), in contrast, have different keypairs and subject names on different IdM CA servers.

In IdM topology, by default, the first master IdM CA server is the CA renewal master.

Note that in upstream documentation, the IdM CA is called Dogtag.

The role of the CA renewal master server

The IdM CA, IdM CA subsystem, and IdM RA certificates are crucial for IdM deployment. Each certificate is stored in an NSS database in the `/etc/pki/pki-tomcat/` directory and also as an LDAP database entry. The certificate stored in LDAP must match the certificate stored in the NSS database. If they do not match, authentication failures occur between the IdM framework and IdM CA, and between IdM CA and LDAP.

All IdM CA replicas have tracking requests for every system certificate. If an IdM deployment with integrated CA did not contain a CA renewal master, requests for renewing system certificates would be made by each IdM CA server independently. This would result in different CA replicas having various system certificates and authentication failures occurring.

Appointing one CA replica as the renewal master allows the system certificates to be renewed exactly once, when required, and thus prevents authentication failures.

The role of certmonger on CA replicas

The certmonger service running on all IdM CA replicas uses the `dogtag-ipa-ca-renew-agent` renewal helper to keep track of IdM system certificates. The renewal helper program reads the CA renewal master configuration. On each CA replica that is not the CA renewal master, the renewal helper programme retrieves the latest system certificates from the `ca_renewal` LDAP entries. Due to non-determinism in when exactly certmonger renewal attempts occur, the `dogtag-ipa-ca-renew-agent` helper sometimes attempts to update a system certificate before the CA renewal master has actually renewed the certificate. If this happens, the old, soon-to-expire certificate is returned to the certmonger on the CA replica. The certmonger, realizing it is the same certificate that is already stored in its database, keeps attempting to renew the certificate with some delay between individual attempts until it can retrieve the updated certificate from the CA renewal master.
The correct functioning of IdM CA renewal master
An IdM deployment with an embedded CA is an IdM deployment that was installed with an IdM CA - or
whose IdM CA master server was installed later. An IdM deployment with an embedded CA must at all
times have exactly one CA replica configured as the renewal master. The renewal master server must be
online and fully functional, and must replicate properly with the other servers.

If the CA renewal master has been decommissioned, or is offline, then system certificates are not
renewed when they are due to expire. All nonrenewal master servers keep reinstalling the current system
certificates until the certificates expire. When this occurs, the IdM deployment is disrupted because even
one expired certificate can cause renewal failures for other certificates.

If the current CA renewal master server is being deleted using the `ipa server-del`, `ipa-replica-manage
del`, `ipa-csreplica-manage del` or `ipa-server-install --uninstall` commands, a CA replica is
automatically assigned as the CA renewal master server. This policy ensures that the renewal master
configuration remains valid.

This policy does not cover the following situations:

- **Offline renewal master**
  - If the renewal master is offline for an extended duration, it may miss a renewal window,
    resulting in expired certificates. If your current renewal master is offline and unavailable for
    an extended period of time, consider assigning a new CA renewal master manually.

- **Replication problems**
  - If replication problems exist between the renewal master and other CA replicas, renewal
    might succeed, but the other CA replicas might not be able to retrieve the updated
    certificates before they expire. To prevent this situation, make sure that your replication
    agreements are working correctly. For details, see general or specific replication
    troubleshooting guidelines in the RHEL 7 Linux Domain Identity, Authentication, and Policy
    Guide.

13.2. CHANGING AND Resetting IDM CA Renewal Master

This section describes how to change the IdM CA renewal master.

When a CA renewal master is being decommissioned, a new CA renewal master is automatically selected
from the list of IdM CA servers without the system administrator being able to influence the selection.

To be able to select the new IdM CA renewal master server, the system administrator needs to perform
the replacement manually, before starting the process of decommissioning the current renewal master.

Resetting the IdM CA renewal master can be necessary if the current CA renewal master configuration
is invalid.

Complete this procedure to change or reset the CA renewal master.

**Prerequisites**

- You have the IdM administrator credentials.

**Procedure**

1. Obtain the IdM administrator credentials:
2. Optionally, to find out which IdM servers in the deployment have the CA role necessary to be eligible to become the new CA renewal master:

```bash
~$ ipa server-role-find --role 'CA server'
2 server roles matched
```

- Server name: server.idm.example.com
  Role name: CA server
  Role status: enabled

- Server name: replica.idm.example.com
  Role name: CA server
  Role status: enabled

```
Number of entries returned 2
```

The output shows that there are two CA servers in the deployment.

3. Optionally, to find out which CA server is the CA current renewal master:

```bash
~$ ipa config-show | grep 'CA renewal master'
IPA CA renewal master: server.idm.example.com
```

The output shows that the current renewal master is `server.idm.example.com`.

4. To change the renewal master configuration, use the `ipa config-mod` utility with the `--ca-renewal-master-server` option:

```bash
~$ ipa config-mod --ca-renewal-master-server replica.idm.example.com | grep 'CA renewal master'
IPA CA renewal master: replica.idm.example.com
```

**IMPORTANT**

You can also switch to a new CA renewal master using:

- the `ipa-cacert-manage --renew` command. This command renews the CA certificate and makes the CA server on which it is run the new CA renewal master.

- the `ipa-cert-fix` command. This command renews the expired CA system certificates, thus fixing the broken deployment, and makes the CA server on which it is run the new CA renewal master.

## 13.3. SWITCHING FROM AN EXTERNALLY TO SELF-SIGNED CA IN IDM

Complete this procedure to switch from an externally-signed to a self-signed certificate of the IdM certificate authority (CA). With a self-signed CA, the renewal of the CA certificate is managed automatically: a system administrator does not need to submit a CSR to an external authority.
Switching from an externally-signed to a self-signed CA replaces only the CA certificate. The certificates signed by the previous CA are still valid and still in use. For example, the certificate chain for the LDAP certificate remains unchanged even after you have moved to a self-signed CA:

```
external_CA certificate > IdM CA certificate > LDAP certificate
```

**Prerequisites**

- You have root access to the IdM CA renewal master.
- You have the IdM administrator credentials.

**Procedure**

1. On the IdM CA renewal master, renew the CA certificate as self-signed:

```
~$ ipa-cacert-manage renew --self-signed
Renewing CA certificate, please wait
CA certificate successfully renewed
The ipa-cacert-manage command was successful
```

2. On all the IdM servers and clients, update the local IdM certificate databases with the certificates from the server:

```
[client ~]$ kinit admin
[client ~]$ ipa-certupdate
Systemwide CA database updated.
Systemwide CA database updated.
The ipa-certupdate command was successful
```

3. Optionally, to check if your update has been successful and the new CA certificate has been added to the `/etc/ipa/ca.crt` file:

```
[client ~]$ openssl crl2pkcs7 -nocrl -certfile /etc/ipa/ca.crt | openssl pkcs7 -print_certs -text -noout
[...]
Certificate:
Data:
  Version: 3 (0x2)
  Serial Number: 39 (0x27)
  Signature Algorithm: sha256WithRSAEncryption
  Issuer: O=IDM.EXAMPLE.COM, CN=Certificate Authority
  Validity
    Not Before: Jul  1 16:32:45 2019 GMT
    Not After : Jul  1 16:32:45 2039 GMT
  Subject: O=IDM.EXAMPLE.COM, CN=Certificate Authority
[...]
```

The output shows that the update has been successful as the new CA certificate is listed with the older CA certificates.

**13.4. RENEWING THE IDM CA RENEWAL MASTER WITH AN EXTERNALLY-SIGNED CERTIFICATE**
This section describes how to renew the IdM CA certificate using an external certificate authority to sign the certificate signing request (CSR). In this configuration, your IdM CA server is a subCA of the external CA. The external CA can, but does not have to, be an Active Directory Certificate Server (AD CS).

If the external certificate authority is AD CS, you can specify the template you want for the IdM CA certificate in the CSR. A certificate template defines the policies and rules that a CA uses when a certificate request is received. Certificate templates in AD correspond to certificate profiles in IdM.

You can define a specific AD CS template by its Object Identifier (OID). OIDs are unique numeric values issued by various issuing authorities to uniquely identify data elements, syntaxes, and other parts of distributed applications.

Alternatively, you can define a specific AD CS template by its name. For example, the name of the default profile used in a CSR submitted by an IdM CA to an AD CS is subCA.

To define a profile by specifying its OID or name in the CSR, use the external-ca-profile option. For details, see the ipa-cacert-manage man page.

Apart from using a ready-made certificate template, you can also create a custom certificate template in the AD CS, and use it in the CSR.

**Prerequisites**

- You have root access to the IdM CA renewal master.
- You have the IdM administrator credentials.

**Procedure**

Complete this procedure to renew the certificate of the IdM CA with external signing, regardless of whether current CA certificate is self-signed or externally-signed.

1. Create a CSR to be submitted to the external CA:

   - If the external CA is an AD CS, use the --external-ca-type=ms-cs option. If you want a different template than the default subCA template, specify it using the --external-ca-profile option:

     ```
     ~]# ipa-cacert-manage renew --external-ca --external-ca-type=ms-cs [--external-ca-profile=PROFILE]
     Exporting CA certificate signing request, please wait
     The next step is to get /var/lib/ipa/ca.csr signed by your CA and re-run ipa-cacert-manage as:
     ipa-cacert-manage renew --external-cert-file=/path/to/signed_certificate --external-cert-file=/path/to/external_ca_certificate
     The ipa-cacert-manage command was successful
     ```

   - If the external CA is not an AD CS:

     ```
     ~]# ipa-cacert-manage renew --external-ca
     Exporting CA certificate signing request, please wait
     The next step is to get /var/lib/ipa/ca.csr signed by your CA and re-run ipa-cacert-manage as:
     ipa-cacert-manage renew --external-cert-file=/path/to/signed_certificate --external-cert-file=/path/to/external_ca_certificate
     The ipa-cacert-manage command was successful
     ```
The output shows that a CSR has been created and is stored in the `/var/lib/ipa/ca.csr` file.

2. Submit the CSR located in `/var/lib/ipa/ca.csr` to the external CA. The process differs depending on the service to be used as the external CA.

3. Retrieve the issued certificate and the CA certificate chain for the issuing CA in a base 64-encoded blob, which is:
   - a PEM file if the external CA is not an AD CS.
   - a Base_64 certificate if the external CA is an AD CS.

   The process differs for every certificate service. Usually, a download link on a web page or in the notification email allows the administrator to download all the required certificates.

   If the external CA is an AD CS and you have submitted the CSR with a known template through the Microsoft Windows Certification Authority management window, the AD CS issues the certificate immediately and the Save Certificate dialog appears in the AD CS web interface, asking where to save the issued certificate.

4. Run the `ipa-cacert-manage renew` command again, adding all the CA certificate files required to supply a full certificate chain. Specify as many files as you need, using the `--external-cert-file` option multiple times:

   ```bash
   ~ ] # ipa-cacert-manage renew --external-cert-file=/path/to/signed_certificate --external-cert-file=/path/to/external_ca_certificate_1 --external-cert-file=/path/to/external_ca_certificate_2
   ```

5. On all the IdM servers and clients, update the local IdM certificate databases with the certificates from the server:

   ```bash
   [client ~ ] $ kinit admin
   [client ~ ] $ ipa-certupdate
   Systemwide CA database updated.
   Systemwide CA database updated.
   The ipa-certupdate command was successful
   ```

6. Optionally, to check if your update has been successful and the new CA certificate has been added to the `/etc/ipa/ca.crt` file:

   ```bash
   [client ~ ] $ openssl crl2pkcs7 -nocrl -certfile /etc/ipa/ca.crt | openssl pkcs7 -print_certs -text -noout
   [...] Certificate: 
   Data:  
   Version: 3 (0x2)  
   Serial Number: 39 (0x27)  
   Signature Algorithm: sha256WithRSAEncryption  
   Issuer: O=IDM.EXAMPLE.COM, CN=Certificate Authority  
   Validity  
   Not Before: Jul 1 16:32:45 2019 GMT  
   Not After : Jul 1 16:32:45 2039 GMT  
   Subject: O=IDM.EXAMPLE.COM, CN=Certificate Authority  
   [...]```
The output shows that the update has been successful as the new CA certificate is listed with the older CA certificates.
CHAPTER 14. OBTAINING AN IDM CERTIFICATE FOR A SERVICE USING CERTMONGER

14.1. CERTMONGER OVERVIEW

What certmonger does
When Identity Management (IdM) is installed with an integrated IdM Certificate Authority (CA), it uses the certmonger service to track and renew system and service certificates. When the certificate is reaching its expiration date, certmonger manages the renewal process by:

- regenerating a certificate-signing request (CSR) using the options provided in the original request.
- submitting the CSR to the IdM CA using the IdM API cert-request command.
- receiving the certificate from the IdM CA.
- executing a pre-save command if specified by the original request.
- installing the new certificate in the location specified in the renewal request: either in an NSS database or in a file.
- executing a post-save command if specified by the original request. For example, the post-save command can instruct certmonger to restart a relevant service, so that the service picks up the new certificate.

Types of certificates certmonger tracks
Certificates can be divided into system and service certificates.

Unlike service certificates (for example, for HTTP, LDAP and PKINIT), which have different keypairs and subject names on different servers, IdM system certificates and their keys are shared by all CA replicas. The IdM system certificates include:

- IdM CA certificate
- OCSP signing certificate
- IdM CA subsystem certificates
- IdM CA audit signing certificate
- IdM renewal agent (RA) certificate
- KRA transport and storage certificates

The certmonger service tracks the IdM system and service certificates that were requested during the installation of IdM environment with an integrated CA. Certmonger also tracks certificates that have been requested manually by the system administrator for other services running on the IdM host. Certmonger does not track external certificate authority certificates or user certificates.

Certmonger components
The certmonger service consists of two main components:

- The certmonger daemon, which is the engine tracking the list of certificates and launching renewal commands
• The getcert utility for the command-line interface (CLI), which allows the system administrator to actively send commands to the certmonger daemon.

More specifically, the system administrator can use the getcert utility to:

• Request a new certificate
• View the list of certificates that certmonger tracks
• Start or stop tracking a certificate
• Renew a certificate

14.2. OBTAINING AN IDM CERTIFICATE FOR A SERVICE USING CERTMONGER

To ensure that communication between browsers and the web service running on your IdM client is secure and encrypted, use a TLS certificate. Obtain the TLS certificate for your web service from the IdM CA.

This section describes how to use certmonger to obtain an IdM certificate for a service (HTTP/webserver.idm.example.com@IDM.EXAMPLE.COM) running on an IdM client.

Using certmonger to request the certificate automatically means that certmonger manages and renews the certificate when it is due for a renewal.

For a visual representation of what happens when certmonger requests a service certificate, see Section 14.3, “Communication flow for certmonger requesting a service certificate”.

Prerequisites

• The web server is enrolled as an IdM client.
• You have root access to the IdM client on which you are running the procedure.
• The service for which you are requesting a certificate does not have to pre-exist in IdM.

Procedure

1. On the webserver.idm.example.com IdM client on which the HTTP service is running, request a certificate for the service corresponding to the HTTP/webserver.idm.example.com@IDM.EXAMPLE.COM principal; specifying that

   • The certificate is to be stored in the local /etc/pki/tls/certs/httpd.pem file
   • The private key is to be stored in the local /etc/pki/tls/private/httpd.key file
   • That an extensionRequest for a SubjectAltName be added to the signing request with the DNS name of webserver.idm.example.com:

   ```bash
   # ipa-getcert request -K HTTP/webserver.idm.example.com -k /etc/pki/tls/private/httpd.key -f /etc/pki/tls/certs/httpd.pem -D webserver.idm.example.com -C "systemctl restart httpd"
   New signing request “20190604065735” added.
   ```

   In the command above:
The `ipa-getcert` command specifies that the certificate is to be obtained from the IdM CA. The `ipa-getcert` command is a shortcut for `getcert request -c IPA`.

- The `-C` option instructs certmonger to restart the `httpd` service after obtaining the certificate.
- The `-D` option specifies the `SubjectAltName` DNS value to be added to the request.

To specify that the certificate be issued:
- From a particular server, use the `-X` option.
- With a particular profile, use the `-T` option.

**NOTE**

RHEL 8 uses a different SSL module in Apache than the one used in RHEL 7. The SSL module relies on OpenSSL rather than NSS. For this reason, in RHEL 8 you cannot use an NSS database to store the HTTPS certificate and the private key.

2. Optionally, to check the status of your request:

```
# ipa-getcert list -f /etc/pki/tls/certs/httpd.pem
Number of certificates and requests being tracked: 3.
Request ID '20190604065735':
  status: MONITORING
  stuck: no
  key pair storage: type=FILE,location='/etc/pki/tls/private/httpd.key'
  certificate: type=FILE,location='/etc/pki/tls/certs/httpd.crt'
  CA: IPA
  [...]
```

The output shows that the request is in the MONITORING status, which means that a certificate has been obtained. The locations of the key pair and the certificate are those requested.

### 14.3. COMMUNICATION FLOW FOR CERTMONGER REQUESTING A SERVICE CERTIFICATE

The diagrams in this section show the stages of what happens when certmonger requests a service certificate from IdM CA server. The sequence consists of these diagrams:

- **Figure 14.1**, “Unencrypted communication”
- **Figure 14.2**, “Certmonger requesting a service certificate”
- **Figure 14.3**, “IdM CA issuing the service certificate”
- **Figure 14.4**, “Certmonger applying the service certificate”
- **Figure 14.5**, “Certmonger requesting a new certificate when the old one is nearing expiration”

*Figure 14.1*, “Unencrypted communication” shows the initial situation: without an HTTPS certificate, the communication between the web server and the browser is unencrypted.
Figure 14.1. Unencrypted communication

![Diagram showing unencrypted communication](image)

**IdM client**

- browser
  - HTTP connection (unencrypted)
- Apache web server
  - `my_company.idm.example.com`
- Certmonger

Figure 14.2, “Certmonger requesting a service certificate” shows the system administrator using **certmonger** to manually request an HTTPS certificate for the Apache web server. Note that when requesting a web server certificate, certmonger does not speak directly to the CA. It proxies through IdM.

Figure 14.2. Certmonger requesting a service certificate

![Diagram showing certmonger request](image)

**IdM client**

- browser
  - `ipa-getcert request`
- Apache web server
  - `my_company.idm.example.com`
- Certmonger
- IdM CA server
Figure 14.3, “IdM CA issuing the service certificate” shows the IdM CA issuing an HTTPS certificate for the web server.

Figure 14.3. IdM CA issuing the service certificate

Figure 14.4, “Certmonger applying the service certificate” shows certmonger placing the HTTPS certificate in appropriate locations on the IdM client and, if instructed to do so, restarting the httpd service. The Apache server subsequently uses the HTTPS certificate to encrypt the traffic between itself and the browser.
Figure 14.4. Certmonger applying the service certificate

![Diagram showing the process of Certmonger applying the service certificate.]

IdM client

- Apache web server
- Certmonger

IdM CA server

Figure 14.5, “Certmonger requesting a new certificate when the old one is nearing expiration” shows **certmonger** automatically requesting a renewal of the service certificate from the IdM CA before the expiration of the certificate. The IdM CA issues a new certificate.

Figure 14.5. Certmonger requesting a new certificate when the old one is nearing expiration

![Diagram showing the process of Certmonger requesting a new certificate.]

IdM client

- Apache web server
- Certmonger

IdM CA server

**Text**: CERTMONGER AUTOMATICALLY REQUESTING A RENEWAL OF THE SERVICE CERTIFICATE FROM THE IDM CA BEFORE THE EXPIRATION OF THE CERTIFICATE. THE IDM CA ISSUES A NEW CERTIFICATE.
14.4. VIEWING THE DETAILS OF A CERTIFICATE REQUEST TRACKED BY CERTMONGER

The certmonger service monitors certificate requests. When a request for a certificate is successfully signed, it results in a certificate. Certmonger manages certificate requests including the resulting certificates. This section describes how to view the details of a particular certificate request managed by certmonger.

Procedure

- If you know how to specify the certificate request, list the details of only that particular certificate request. You can, for example, specify:
  - The request ID
  - The location of the certificate
  - The certificate nickname
  For example, to view the details of the certificate whose request ID is 20190408143846, using the `-v` option to view all the details of errors in case your request for a certificate was unsuccessful:

```
# getcert list -i 20190408143846 -v
```

Number of certificates and requests being tracked: 16.
Request ID '20190408143846':
status: MONITORING
stuck: no
key pair storage: type=NSSDB,location='/etc/dirsrv/slapd-IDM-EXAMPLE-COM',nickname='Server-Cert',token='NSS Certificate DB',pinfile='/etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt'
certificate: type=NSSDB,location='/etc/dirsrv/slapd-IDM-EXAMPLE-COM',nickname='Server-Cert',token='NSS Certificate DB'
CA: IPA
issuer: CN=Certificate Authority,O=IDM.EXAMPLE.COM
subject: CN=r8server.idm.example.com,O=IDM.EXAMPLE.COM
expires: 2021-04-08 16:38:47 CEST
dns: r8server.idm.example.com
principal name: ldap/server.idm.example.com@IDM.EXAMPLE.COM
key usage: digitalSignature,nonRepudiation,keyEncipherment,dataEncipherment
eku: id-kp-serverAuth,id-kp-clientAuth
pre-save command:
post-save command: /usr/libexec/ipa/certmonger/restart_dirsrv IDM-EXAMPLE-COM
track: yes
auto-renew: yes
```

The output displays several pieces of information about the certificate, for example:

- the certificate location; in the example above, it is the NSS database in the `/etc/dirsrv/slapd-IDM-EXAMPLE-COM` directory
- the certificate nickname; in the example above, it is `Server-Cert`
- the file storing the pin; in the example above, it is /etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt

- the Certificate Authority that will be used to renew the certificate; in the example above, it is the IPA CA

- the expiration date; in the example above, it is 2021-04-08 16:38:47 CEST

- the status of the certificate; in the example above, the MONITORING status means that the certificate is valid and it is being tracked

- the post-save command; in the example above, it is the restart of the LDAP service

- If you do not know how to specify the certificate request, list the details of all the certificates that certmonger is monitoring or attempting to obtain:

  # getcert list

Additional information
To view the different options how to specify the certificate request displayed, see the getcert list man page.

14.5. STARTING AND STOPPING CERTIFICATE TRACKING

This section describes how you can use the getcert stop-tracking and getcert start-tracking commands to monitor certificates. The two commands are provided by the certmonger service. Enabling certificate tracking is especially useful if you have imported a certificate issued by the IdM CA onto the machine from a different IdM client. Enabling certificate tracking can also be the final step of the following provisioning scenario:

1. On the IdM server, you create a certificate for a system that does not exist yet.

2. You create the new system.

3. You enroll the new system as an IdM client.

4. You import the certificate and the key from the IdM server on to the IdM client.

5. You start tracking the certificate using certmonger to ensure that it gets renewed when it is due to expire.

Procedure

- To disable the monitoring of a certificate with the Request ID of 20190408143846:

  # getcert stop-tracking -i 20190408143846

  For more options, see the getcert stop-tracking man page.

- To enable the monitoring of a certificate stored in the /tmp/some_cert.crt file, whose private key is stored in the /tmp/some_key.key file:

  # getcert start-tracking -c IPA -f /tmp/some_cert.crt -k /tmp/some_key.key

  Certmonger cannot automatically identify the CA type that issued the certificate. For this reason, add the -c option with the IPA value to the getcert start-tracking command if the
certificate was issued by the IdM CA. Omitting to add the \texttt{-c} option results in \texttt{certmonger} entering the NEED\_CA state.

For more options, see the \texttt{getcert start-tracking} man page.

\textbf{NOTE}

The two commands do not manipulate the certificate. For example, \texttt{getcert stop-tracking} does not delete the certificate or remove it from the NSS database or from the filesystem but simply removes the certificate from the list of monitored certificates. Similarly, \texttt{getcert start-tracking} only adds a certificate to the list of monitored certificates.

\section*{14.6. RENEWING A CERTIFICATE MANUALLY}

When a certificate is near its expiration date, the \texttt{certmonger} daemon automatically issues a renewal command using the CA helper, obtains a renewed certificate and replaces the previous certificate with the new one.

It is also possible to manually renew a certificate in advance by using the \texttt{getcert resubmit} command. This way, you can update the information the certificate contains, e.g. by adding a Subject Alternative Name (SAN).

This section describes how to renew a certificate manually.

\textbf{Procedure}

- To renew a certificate with the Request ID of 20190408143846:

  \begin{verbatim}
  # getcert resubmit -i 20190408143846
  \end{verbatim}

  To obtain the Request ID for a specific certificate, use the \texttt{getcert list} command. For details, see the \texttt{getcert list} man page.

\section*{14.7. MAKING CERTMONGER RESUME TRACKING OF IDM CERTIFICATES ON A CA REPLICA}

This procedure shows how to make \texttt{certmonger} resume the tracking of IdM system certificates that are crucial for an IdM deployment with an integrated certificate authority after the tracking of certificates was interrupted. The interruption may have been caused by the IdM host being unenrolled from IdM during the renewal of the system certificates or by replication topology not working properly. The procedure also shows how to make \texttt{certmonger} resume the tracking of the IdM service certificates, namely the \texttt{HTTP}, \texttt{LDAP} and \texttt{PKINIT} certificates.

\textbf{Prerequisites}

- The host on which you want to resume tracking system certificates is an IdM server that is also an IdM CA but not the IdM CA renewal master.

\textbf{Procedure}

1. Get the PIN for the subsystem CA certificates:

  \begin{verbatim}
  # grep 'internal=' /var/lib/pki/pki-tomcat/conf/password.conf
  \end{verbatim}
2. Add tracking to the subsystem CA certificates, replacing [internal PIN] in the commands below with the PIN obtained in the previous step:

```bash
# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "caSigningCert cert-pki-ca" -c 'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B /usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert "caSigningCert cert-pki-ca"'

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "auditSigningCert cert-pki-ca" -c 'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B /usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert "auditSigningCert cert-pki-ca"'

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "ocspSigningCert cert-pki-ca" -c 'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B /usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert "ocspSigningCert cert-pki-ca"'

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "subsystemCert cert-pki-ca" -c 'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B /usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert "subsystemCert cert-pki-ca"'

```

3. Add tracking for the remaining IdM certificates, the HTTP, LDAP, IPA renewal agent and PKINIT certificates:

```bash
# getcert start-tracking -f /var/lib/ipa/certs/httpd.crt -k /var/lib/ipa/private/httpd.key -p /var/lib/ipa/passwds/idm.example.com-443-RSA -c IPA -C /usr/libexec/ipa/certmonger/restart_httpd

# getcert start-tracking -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM -n "Server-Cert" -c IPA -p /etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt -C '/usr/libexec/ipa/certmonger/restart_dirsrv "IDM-EXAMPLE-COM"'

# getcert start-tracking -f /var/kerberos/krb5kdc/kdc.crt -k /var/kerberos/krb5kdc/kdc.key -c dogtag-ipa-ca-renew-agent -B /usr/libexec/ipa/certmonger/renew_ra_cert_pre -C /usr/libexec/ipa/certmonger/renew_ra_cert

# getcert start-tracking -f /var/kerberos/krb5kdc/kdc.crt -k /var/kerberos/krb5kdc/kdc.key -c dogtag-ipa-ca-renew-agent -B /usr/libexec/ipa/certmonger/renew_kdc_cert
```

4. Restart `certmonger`:

```bash
# systemctl restart certmonger
```

5. Wait for one minute after `certmonger` has started and then check the statuses of the new certificates:
Additional resources

- If your IdM system certificates have all expired, follow the procedure described in this Knowledge Centered Support (KCS) solution to manually renew IdM system certificates on the IdM CA master that is also the CA renewal master and the CRL generation master. Then follow the procedure described in this KCS solution to manually renew IdM system certificates on all the other CA servers in the topology.
CHAPTER 15. ENABLING AD USERS TO ADMINISTER IDM

15.1. ID OVERRIDES FOR AD USERS

In Red Hat Enterprise Linux (RHEL) 7, external group membership allows AD users and groups to access IdM resources in a POSIX environment with the help of the System Security Services Daemon (SSSD).

The IdM LDAP server has its own mechanisms to grant access control. RHEL 8 introduces an update that allows adding an ID user override for an AD user as a member of an IdM group. An ID override is a record describing what a specific Active Directory user or group properties should look like within a specific ID view, in this case the Default Trust View. As a consequence of the update, the IdM LDAP server is able to apply access control rules for the IdM group to the AD user.

AD users are now able to use the self service features of IdM UI, for example to upload their SSH keys, or change their personal data. An AD administrator is able to fully administer IdM without having two different accounts and passwords.

NOTE
Currently, selected features in IdM may still be unavailable to AD users. For example, setting passwords for IdM users as an AD user from the IdM admins group might fail.

15.2. USING ID OVERRIDES TO ENABLE AD USERS TO ADMINISTER IDM

Prerequisites

- The idm:DL1 stream is enabled on your IdM server and you have switched to the RPMs delivered through this stream:

  ```
  # yum module enable idm:DL1
  # yum distro-sync
  ```

- The idm:DL1/adtrust profile is installed on your IdM server.

  ```
  # yum module install idm:DL1/adtrust
  ```

The profile contains all the packages necessary for installing an IdM server that will have a trust agreement with Active Directory, including the ipa-idoverride-memberof package.

- A working Identity Management environment is set up. For details, see Installing Identity Management.

- A working trust between your Identity Management environment and Active Directory is set up.

Procedure

This procedure describes creating and using an ID override for an Active Directory (AD) user to give that user rights identical to those of an Identity Management (IdM) user. During this procedure, work on an IdM server that is configured as a trust controller or a trust agent. For details on trust controllers and trust agents, see Trust controllers and trust agents in Planning Identity Management.

1. As an IdM administrator, create an ID override for an AD user in the Default Trust View. For example, to create an ID override for the ad_user@ad.example.com user:
   ```
   ```
# kinit admin
# ipa idoverrideuser-add 'default trust view' ad_user@ad.example.com

2. Add the ID override from the Default Trust View as a member to an IdM group. If the group in question is a member of an IdM role, the AD user represented by the ID override will gain all permissions granted by the role when using the IdM API, including both the command line interface and the IdM web UI. For example, to add the ID override for the ad_user@ad.example.com user to the admins group:

   # ipa group-add-member admins --idoverrideusers=ad_user@ad.example.com

15.3. MANAGING IDM COMMAND-LINE INTERFACE (CLI) AS AN AD USER

This procedure checks that an Active Directory user can log into Identity Management CLI and run commands appropriate for his role.

1. Destroy the current Kerberos ticket of the IdM administrator:

   # kdestroy -A

   **NOTE**
   The destruction of the Kerberos ticket is required because the GSSAPI implementation in MIT Kerberos chooses credentials from the realm of the target service by preference, which in this case is the IdM realm. This means that if a credentials cache collection, namely the KCM:, KEYRING:, or DIR: type of credentials cache is in use, a previously obtained admin or any other IdM principal’s credentials will be used to access the IdM API instead of the AD user’s credentials.

2. Obtain the Kerberos credentials of the AD user for whom an ID override has been created:

   # kinit ad_user@AD.EXAMPLE.COM
   Password for ad_user@AD.EXAMPLE.COM:

3. Test that the ID override of the AD user enjoys the same privileges stemming from membership in the IdM group as any IdM user in that group. If the ID override of the AD user has been added to the admins group, the AD user can, for example, create groups in IdM:

   # ipa group-add some-new-group

   -----------------------------
   Added group "some-new-group"
   -----------------------------
   Group name: some-new-group
   GID: 1997000011
CHAPTER 16. USING CANONICALIZED DNS HOST NAMES IN IDM

DNS canonicalization is disabled by default on IdM clients to avoid potential security risks. For example, if an attacker controls the DNS server and a host in the domain, the attacker can cause that the short host name demo is resolved to malicious.example.com. In this case, the user connects to a different server than expected.

This section describes how to use canonicalized host names on IdM clients.

16.1. ADDING AN ALIAS TO A HOST PRINCIPAL

By default, IdM clients enrolled by using the ipa-client-install command do not allow to use short host names in service principals. For example, users can use only host/demo.example.com@EXAMPLE.COM instead of host/demo@EXAMPLE.COM when accessing a service.

This section explains how to add an alias to a Kerberos principal. Note that you can alternatively enable canonicalization of host names in the /etc/krb5.conf file. For details, see Section 16.2, "Enabling canonicalization of host names in service principals on clients".

Prerequisites

- The IdM client is installed.
- The host name is unique in the network.

Procedure

1. Authenticate to IdM as the admin user:

   $ kinit admin

2. Add the alias to the host principal. For example, to add the demo alias to the demo.examle.com host principal:

   $ ipa host-add-principal demo.example.com --principal=demo

16.2. ENABLING CANONICALIZATION OF HOST NAMES IN SERVICE PRINCIPALS ON CLIENTS

This section describes how you enable canonicalization of host names in services principals on clients.

Note that if you use host principal aliases, as described in Section 16.1, "Adding an alias to a host principal", you do not need to enable canonicalization.

Prerequisites

- The IdM client is installed.
- You are logged in to the IdM client as the root user.
- The host name is unique in the network.
Procedure

1. Set the `dns_canonicalize_hostname` parameter in the `[libdefaults]` section in the `/etc/krb5.conf` file to `false`:

   ```
   [libdefaults]
   ...
   dns_canonicalize_hostname = true
   ```

16.3. OPTIONS FOR USING HOST NAMES WITH DNS HOST NAME CANONICALIZATION ENABLED

If you set `dns_canonicalize_hostname = true` in the `/etc/krb5.conf` file as explained in Section 16.2, “Enabling canonicalization of host names in service principals on clients”, you have the following options when you use a host name in a service principal:

- In IdM environments, you can use the full host name in a service principal, such as `host/demo.example.com@EXAMPLE.COM`.

- In environments without IdM, but if the RHEL host as a member of an Active Directory (AD) domain, no further considerations are required, because AD domain controllers (DC) automatically create service principals for NetBIOS names of the machines enrolled into Active Directory.