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>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4. HOST ENTRY IN IDM LDAP</td>
<td>73</td>
</tr>
<tr>
<td>12.4.1. Host entry configuration properties</td>
<td>74</td>
</tr>
<tr>
<td>12.5. ADDING IDM HOST ENTRIES FROM IDM CLI</td>
<td>75</td>
</tr>
<tr>
<td>12.6. DELETING HOST ENTRIES FROM IDM CLI</td>
<td>76</td>
</tr>
<tr>
<td>12.7. RE-ENROLLING AN IDENTITY MANAGEMENT CLIENT</td>
<td>76</td>
</tr>
<tr>
<td>12.7.1. Client re-enrollment in IdM</td>
<td>76</td>
</tr>
<tr>
<td>12.7.1.1. What happens during client re-enrollment</td>
<td>76</td>
</tr>
<tr>
<td>12.7.2. Re-enrolling a client by using user credentials: Interactive re-enrollment</td>
<td>77</td>
</tr>
<tr>
<td>12.7.3. Re-enrolling a client by using the client keytab: Non-interactive re-enrollment</td>
<td>77</td>
</tr>
<tr>
<td>12.7.4. Testing an Identity Management client after installation</td>
<td>78</td>
</tr>
<tr>
<td>12.8. RENAMING IDENTITY MANAGEMENT CLIENT SYSTEMS</td>
<td>78</td>
</tr>
<tr>
<td>12.8.1. Prerequisites</td>
<td>78</td>
</tr>
<tr>
<td>12.8.2. Uninstalling an Identity Management client</td>
<td>79</td>
</tr>
<tr>
<td>12.8.3. Renaming the host system</td>
<td>80</td>
</tr>
<tr>
<td>12.8.4. Re-installing an Identity Management client</td>
<td>80</td>
</tr>
<tr>
<td>12.8.5. Re-adding services, re-generating certificates, and re-adding host groups</td>
<td>80</td>
</tr>
<tr>
<td>12.9. DISABLING AND RE-ENABLING HOST ENTRIES</td>
<td>80</td>
</tr>
<tr>
<td>12.9.1. Disabling Hosts</td>
<td>80</td>
</tr>
<tr>
<td>12.9.2. Re-enabling Hosts</td>
<td>81</td>
</tr>
<tr>
<td>CHAPTER 13. ADDING HOST ENTRIES FROM IDM WEB UI</td>
<td>82</td>
</tr>
<tr>
<td>13.1. HOSTS IN IDM</td>
<td>82</td>
</tr>
<tr>
<td>13.2. HOST ENROLLMENT</td>
<td>82</td>
</tr>
<tr>
<td>13.2.1. User privileges required for host enrollment</td>
<td>83</td>
</tr>
<tr>
<td>User privileges for optionally manually creating a host entry in IdM LDAP</td>
<td>83</td>
</tr>
<tr>
<td>User privileges for joining the client to the IdM domain</td>
<td>83</td>
</tr>
<tr>
<td>13.2.2. Enrollment and authentication of IdM hosts and users: comparison</td>
<td>83</td>
</tr>
<tr>
<td>13.2.3. Alternative authentication options for IdM hosts</td>
<td>84</td>
</tr>
<tr>
<td>13.3. HOST ENTRY IN IDM LDAP</td>
<td>85</td>
</tr>
<tr>
<td>13.3.1. Host entry configuration properties</td>
<td>85</td>
</tr>
<tr>
<td>13.4. ADDING HOST ENTRIES FROM THE WEB UI</td>
<td>86</td>
</tr>
<tr>
<td>CHAPTER 14. ADJUSTING ID RANGES MANUALLY</td>
<td>89</td>
</tr>
<tr>
<td>14.1. ID RANGES</td>
<td>89</td>
</tr>
<tr>
<td>14.2. AUTOMATIC ID RANGES ASSIGNMENT</td>
<td>89</td>
</tr>
<tr>
<td>14.3. ASSIGNING THE IDM ID RANGE MANUALLY DURING SERVER INSTALLATION</td>
<td>90</td>
</tr>
<tr>
<td>14.4. ADDING A NEW IDM ID RANGE</td>
<td>91</td>
</tr>
<tr>
<td>14.5. DISPLAYING CURRENTLY ASSIGNED DNA ID RANGES</td>
<td>91</td>
</tr>
<tr>
<td>14.6. AUTOMATIC DNA ID RANGE EXTENSION</td>
<td>92</td>
</tr>
<tr>
<td>14.7. MANUAL DNA ID RANGE ADJUSTMENT</td>
<td>92</td>
</tr>
<tr>
<td>14.8. ADJUSTING DNA ID RANGES MANUALLY</td>
<td>93</td>
</tr>
<tr>
<td>CHAPTER 15. GRANTING SUDO ACCESS TO AN IDM USER ON AN IDM CLIENT</td>
<td>95</td>
</tr>
<tr>
<td>15.1. SUDO ACCESS ON AN IDM CLIENT</td>
<td>95</td>
</tr>
<tr>
<td>15.2. GRANTING SUDO ACCESS TO AN IDM USER ON AN IDM CLIENT USING IDM WEB UI</td>
<td>95</td>
</tr>
<tr>
<td>CHAPTER 16. PUBLIC KEY CERTIFICATES IN IDENTITY MANAGEMENT</td>
<td>98</td>
</tr>
<tr>
<td>16.1. CERTIFICATE AUTHORITIES IN IDM</td>
<td>98</td>
</tr>
<tr>
<td>16.2. COMPARISON OF CERTIFICATES AND KERBEROS</td>
<td>98</td>
</tr>
<tr>
<td>16.3. THE PROS AND CONS OF USING CERTIFICATES TO AUTHENTICATE USERS IN IDM</td>
<td>99</td>
</tr>
<tr>
<td>CHAPTER 17. CONVERTING CERTIFICATE FORMATS TO WORK WITH IDM</td>
<td>101</td>
</tr>
<tr>
<td>17.1. CERTIFICATE FORMATS AND ENCODINGS IN IDM</td>
<td>101</td>
</tr>
<tr>
<td>System configuration</td>
<td>101</td>
</tr>
</tbody>
</table>
Certificate encodings
User authentication
Useful certificate commands

17.2. CONVERTING AN EXTERNAL CERTIFICATE TO LOAD INTO AN IDM USER ACCOUNT
17.2.1. Converting an external certificate in the IdM CLI and loading it into an IdM user account
17.2.2. Converting an external certificate in the IdM web UI for loading into an IdM user account

17.3. PREPARING TO LOAD A CERTIFICATE INTO THE BROWSER
17.3.1. Exporting a certificate and private key from an NSS database into a PKCS #12 file
17.3.2. Combining certificate and private key PEM files into a PKCS #12 file

17.4. CERTIFICATE-RELATED COMMANDS AND FORMATS IN IDM

CHAPTER 18. MANAGING THE VALIDITY OF CERTIFICATES IN IDM
Managing the validity of an existing certificate that was issued by IdM CA
Managing the validity of future certificates issued by IdM CA

18.1. VIEWING THE EXPIRY DATE OF A CERTIFICATE
18.1.1. Viewing the expiry date of a certificate in IdM WebUI
18.1.2. Viewing the expiry date of a certificate in the CLI

18.2. REVOKING CERTIFICATES WITH THE INTEGRATED IDM CAS
18.2.1. Certificate revocation reasons
18.2.2. Revoking certificates with the integrated IdM CAs using IdM WebUI
18.2.3. Revoking certificates with the integrated IdM CAs using IdM CLI

18.3. RESTORING CERTIFICATES WITH THE INTEGRATED IDM CAS
18.3.1. Restoring certificates with the integrated IdM CAs using IdM WebUI
18.3.2. Restoring certificates with the integrated IdM CAs using IdM CLI

CHAPTER 19. CONFIGURING IDENTITY MANAGEMENT FOR SMART CARD AUTHENTICATION
19.1. CONFIGURING THE IDM SERVER FOR SMART CARD AUTHENTICATION
19.2. CONFIGURING THE IDM CLIENT FOR SMART CARD AUTHENTICATION
19.3. ADDING A CERTIFICATE TO A USER ENTRY IN IDM
19.3.1. Adding a certificate to a user entry in the IdM Web UI
19.3.2. Adding a certificate to a user entry in the IdM CLI

19.4. INSTALLING GNUTLS-UTILS
19.5. STORING A CERTIFICATE ON THE SMART CARD

19.6. CONFIGURING THE BROWSER FOR SMART CARD AUTHENTICATION
19.7. LOGGING IN TO IDM WITH SMART CARDS

CHAPTER 20. CONFIGURING CERTIFICATES ISSUED BY ADCS FOR SMART CARD AUTHENTICATION IN IDM
20.1. SMART CARD AUTHENTICATION
20.2. WINDOWS SERVER SETTINGS REQUIRED FOR TRUST CONFIGURATION AND CERTIFICATE USAGE
20.3. COPYING CERTIFICATES FROM ACTIVE DIRECTORY USING SFTP
20.4. CONFIGURING THE IDM SERVER AND CLIENTS FOR SMART CARD AUTHENTICATION USING ADCS CERTIFICATES
20.5. CONVERTING THE PFX FILE
20.6. INSTALLING GNUTLS-UTILS
20.7. STORING A CERTIFICATE ON THE SMART CARD
20.8. CONFIGURING TIMEOUTS IN SSSD.CONF
20.9. CREATING CERTIFICATE MAPPING RULES FOR SMART CARD AUTHENTICATION

CHAPTER 21. CONFIGURING CERTIFICATE MAPPING RULES IN IDENTITY MANAGEMENT
21.1. CERTIFICATE MAPPING RULES FOR CONFIGURING AUTHENTICATION ON SMART CARDS
21.1.2. Components of an identity mapping rule in IdM
21.1.3. Obtaining the issuer from a certificate for use in a matching rule
Additional information

21.2. CONFIGURING CERTIFICATE MAPPING FOR USERS STORED IN IDM
21.2.1. Adding a certificate mapping rule in IdM
   21.2.1.1. Adding a certificate mapping rule in the IdM web UI
   21.2.1.2. Adding a certificate mapping rule in the IdM CLI
21.2.2. Adding certificate mapping data to a user entry in IdM
   21.2.2.1. Adding certificate mapping data to a user entry in the IdM web UI
   21.2.2.2. Adding certificate mapping data to a user entry in the IdM CLI

21.3. CONFIGURING CERTIFICATE MAPPING FOR USERS WHOSE AD USER ENTRY CONTAINS THE WHOLE CERTIFICATE
   21.3.1. Adding a certificate mapping rule for users whose AD entry contains whole certificates
   21.3.1.1. Adding a certificate mapping rule in the IdM web UI
   21.3.1.2. Adding a certificate mapping rule in the IdM CLI

21.4. CONFIGURING CERTIFICATE MAPPING IF AD IS CONFIGURED TO MAP USER CERTIFICATES TO USER ACCOUNTS
   21.4.1. Adding a certificate mapping rule if the trusted AD domain is configured to map user certificates
   21.4.1.1. Adding a certificate mapping rule in the IdM web UI
   21.4.1.2. Adding a certificate mapping rule in the IdM CLI
21.4.2. Checking certificate mapping data on the AD side

21.5. CONFIGURING CERTIFICATE MAPPING IF AD USER ENTRY CONTAINS NO CERTIFICATE OR MAPPING DATA
   21.5.1. Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data
   21.5.1.1. Adding a certificate mapping rule in the IdM web UI
   21.5.1.2. Adding a certificate mapping rule in the IdM CLI
21.5.2. Adding a certificate to an AD user’s ID override if the user entry in AD contains no certificate or mapping data
   21.5.2.1. Adding a certificate to an AD user’s ID override in the IdM web UI
   21.5.2.2. Adding a certificate to an AD user’s ID override in the IdM CLI
21.6. COMBINING SEVERAL IDENTITY MAPPING RULES INTO ONE

CHAPTER 22. CONFIGURING AUTHENTICATION WITH A CERTIFICATE STORED ON THE DESKTOP OF AN IDM CLIENT
   22.1. CONFIGURING THE IDENTITY MANAGEMENT SERVER FOR CERTIFICATE AUTHENTICATION IN THE WEB UI
   22.2. REQUESTING A NEW USER CERTIFICATE AND EXPORTING IT TO THE CLIENT
   22.3. MAKING SURE THE CERTIFICATE AND USER ARE LINKED TOGETHER
   22.4. CONFIGURING A BROWSER TO ENABLE CERTIFICATE AUTHENTICATION
   22.5. AUTHENTICATING TO THE IDENTITY MANAGEMENT WEB UI WITH A CERTIFICATE AS AN IDENTITY MANAGEMENT USER
   22.6. CONFIGURING AN IDM CLIENT TO ENABLE AUTHENTICATING TO THE CLI USING A CERTIFICATE

CHAPTER 23. USING IDM CA RENEWAL MASTER
   23.1. EXPLANATION OF IDM CA RENEWAL MASTER
      The role of the CA renewal master server
      The role of certmonger on CA replicas
      The correct functioning of IdM CA renewal master
   23.2. CHANGING AND RESETTING IDM CA RENEWAL MASTER
   23.3. SWITCHING FROM AN EXTERNALLY TO SELF-SIGNED CA IN IDM
   23.4. RENEWING THE IDM CA RENEWAL MASTER WITH AN EXTERNALLY-SIGNED CERTIFICATE

CHAPTER 24. RENEWING EXPIRED SYSTEM CERTIFICATES WHEN IDM IS OFFLINE
   24.1. RENEWING EXPIRED SYSTEM CERTIFICATES ON A CA RENEWAL MASTER
   24.2. VERIFYING OTHER IDM SERVERS IN THE IDM DOMAIN AFTER RENEWAL
31.1.4. Running IdM Healthcheck
31.2. LOG ROTATION
31.3. CONFIGURING LOG ROTATION USING THE IDM HEALTHCHECK

CHAPTER 32. CHECKING SERVICES USING IDM HEALTHCHECK .............................................. 214
32.1. SERVICES HEALTHCHECK TEST
32.2. SCREENING SERVICES USING HEALTHCHECK

CHAPTER 33. VERIFYING YOUR IDM AND AD TRUST CONFIGURATION USING IDM HEALTHCHECK ... 216
33.1. IDM AND AD TRUST HEALTHCHECK TESTS
33.2. SCREENING THE TRUST WITH THE HEALTHCHECK TOOL

CHAPTER 34. VERIFYING CERTIFICATES USING IDM HEALTHCHECK ................................. 218
34.1. IDM CERTIFICATES HEALTHCHECK TESTS
34.2. SCREENING CERTIFICATES USING THE HEALTHCHECK TOOL

CHAPTER 35. VERIFYING SYSTEM CERTIFICATES USING IDM HEALTHCHECK ...................... 221
35.1. SYSTEM CERTIFICATES HEALTHCHECK TESTS
35.2. SCREENING SYSTEM CERTIFICATES USING HEALTHCHECK

CHAPTER 36. CHECKING DISK SPACE USING IDM HEALTHCHECK ........................................ 224
36.1. DISK SPACE HEALTHCHECK TEST
36.2. SCREENING DISK SPACE USING THE HEALTHCHECK TOOL

CHAPTER 37. VERIFYING PERMISSIONS OF IDM CONFIGURATION FILES USING HEALTHCHECK .... 226
37.1. FILE PERMISSIONS HEALTHCHECK TESTS
37.2. SCREENING CONFIGURATION FILES USING HEALTHCHECK

CHAPTER 38. CHECKING IDM REPPLICATION USING HEALTHCHECK ..................................... 229
38.1. REPPLICATION HEALTHCHECK TESTS
38.2. SCREENING REPPLICATION USING HEALTHCHECK

CHAPTER 39. DEMOTING OR PROMOTING HIDDEN REPLICA S .............................................. 232

CHAPTER 40. SETTING UP SAMBA ON AN IDM DOMAIN MEMBER .................................... 233
40.1. PREPARING THE IDM DOMAIN FOR INSTALLING SAMBA ON DOMAIN MEMBERS
40.2. ENABLING THE AES ENCRYPTION TYPE IN ACTIVE DIRECTORY USING A GPO
40.3. INSTALLING AND CONFIGURING A SAMBA SERVER ON AN IDM CLIENT
40.4. MANUALLY ADDING AN ID MAPPING CONFIGURATION IF IDM TRUSTS A NEW DOMAIN
40.5. ADDITIONAL RESOURCES

CHAPTER 41. IDENTITY MANAGEMENT SECURITY SETTINGS .............................................. 239
41.1. HOW IDENTITY MANAGEMENT APPLIES DEFAULT SECURITY SETTINGS
41.2. ANONYMOUS LDAP BINDS IN IDENTITY MANAGEMENT

CHAPTER 42. USING AUTOMOUNT IN IDM ................................................................. 240
42.1. SETTING UP A KERBEROS-AWARE NFS SERVER
42.2. SETTING UP A KERBEROS-AWARE NFS CLIENT
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:
  1. Make sure you are viewing the documentation in the *Multi-page HTML* format. In addition, ensure you see the **Feedback** button in the upper right corner of the document.
  2. Use your mouse cursor to highlight the part of text that you want to comment on.
  3. Click the **Add Feedback** pop-up that appears below the highlighted text.
  4. 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:

     ```
     [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:

     ```
     [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:

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

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

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

```
[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:

```
# 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.
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, `krb5kdc` and `kadmin`. The `krb5kdc` service is the Kerberos version 5 Authentication service and Key Distribution Center (KDC) deamon. The `kadmin` service is the Kerberos V5 database administration program.

- The `named` service refers to the Internet domain name service (DNS).

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

The output of the `ipactl status` command on your server depends on your IdM configuration. For example, if an IdM deployment does not include a DNS server, the `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 `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 `ipactl` utility”

To start, stop, or restart an individual IdM service, see:
Section 2.3, "Starting and stopping an individual Identity Management service: the systemctl utility"

To display the version of IdM software, see:

- Section 2.4, “Methods for Displaying Identity Management Software Version”

### 2.2. STARTING AND STOPPING THE ENTIRE IDENTITY MANAGEMENT SERVER: THE ipactl UTILITY

Use the `ipactl` utility to stop, start, or restart the entire IdM server along with all the installed services. Using the `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 `ipactl` commands.

**ipactl commands**

To start the entire IdM server:

```
# ipactl start
```

To stop the entire IdM server:

```
# ipactl stop
```

To restart the entire IdM server:

```
# ipactl restart
```

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

```
# ipactl status
```

**IMPORTANT**

You cannot use the IdM web UI to perform the `ipactl` commands.

### 2.3. STARTING AND STOPPING AN INDIVIDUAL IDENTITY MANAGEMENT SERVICE: THE 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 `systemctl` utility to stop, start, or restart an individual IdM service.

For example, use `systemctl` after customizing the Directory Server behavior, without modifying the other IdM services:

```
# systemctl restart dirsrv@REALM-NAME.service
```

Also, when initially deploying an IdM trust with Active Directory, modify the `/etc/sssd/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:

```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:

```systemctl start name.service```

To stop a particular IdM service:

```systemctl stop name.service```

To restart a particular IdM service:

```systemctl restart name.service```

To view the status of a particular IdM service:

```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 IdM → Services and selecting the service.

### 2.4. METHODS FOR DISPLAYING IDENTITY MANAGEMENT SOFTWARE VERSION

You can display the IdM version number with:

- the IdM WebUI
- `ipa` commands

---

**CHAPTER 2. VIEWING, STARTING AND STOPPING THE IDENTITY MANAGEMENT SERVICES**
rpm commands

Displaying Version through the WebUI
In the IdM WebUI, the software version can be displayed by choosing About from the username menu at the top-right.

Displaying Version with ipa commands
From the command line, use the ipa --version command.

```
[root@server ~]# ipa --version
VERSION: 4.8.0, API_VERSION: 2.233
```

Displaying Version with rpm commands
If IdM services are not operating properly, you can use the rpm utility to determine the version number of the ipa-server package that is currently installed.

```
[root@server ~]# rpm -q ipa-server
ipa-server-4.8.0-11.module+el8.1.0+4247+9f3fd721.x86_64
```
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]
```
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.

   $$
   \text{ipa help topics}
   $$

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`

   $$
   \text{ipa help user}
   $$

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

   $$
   \text{ipa help user | less}
   $$

   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.

   $ 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.

   $ 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
ipa dnsrecord-add, ipa dnsrecord-mod, ipa dnsrecord-del, ipa dnsrecord-find, ipa dnrecord-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 (IdM) 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 command line interface (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
```
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:

```bash
$ 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:

```bash
$ 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:
   ```bash
   $ 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
```
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:

```
$ 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:
  
  ```
  $ 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:

  ```
  $ 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.

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

or

```
$ 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:

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

or

```
$ ipa permission-mod --right={read,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 (`*`):

```
$ 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
   ```
To adjust the limits globally for all queries, use the `ipa config-mod` command and add the `--searchrecordslimit` and `--searchtimelimit` options. For example:

```bash
$ ipa config-mod --searchrecordslimit=500 --searchtimelimit=5
```

To adjust the limits only for a specific query, add the `--sizelimit` or `--timelimit` options to the command. For example:

```bash
$ 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.
CHAPTER 5. ACCESSING THE IDM WEB UI IN A WEB BROWSER

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 6.1, “Kerberos authentication in Identity Management”.

- **Smart card**
  For details, see Section 19.1, “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.
### Active users

<table>
<thead>
<tr>
<th>User login</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.

Prerequisites

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

6.1. 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

6.2. 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:

```
$ 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:

```
$ kinit admin
Password for 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
```

### 6.3. 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.

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

6.4. 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 6.3, “Configuring the browser for Kerberos authentication”.

### 6.5. 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:

```
# 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
```
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 6.3, “Configuring the browser for Kerberos authentication”.

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

### 6.6. 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 usertab**.

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 user name 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. **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 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. MANAGING USER ACCOUNTS USING THE IDM WEB UI

Identity Management (IdM) provides several stages that can help you to manage various user work life situations:

Creating a user account

Creating a stage user account before an employee starts their career in your company and be prepared in advance for the day when the employee appears in the office and want to activate the account.

You can omit this step and create the active user account directly. The procedure is similar to creating a stage user account.

Activating a user account

Activating the account the first working day of the employee.

Disabling a user account

If the user go to a parental leave for couple of months, you will need to disable the account temporarily.

Enabling a user account

When the user returns, you will need to re-enable the account.

Preserving a user account

If the user wants to leave the company, you will need to delete the account with a possibility to restore it because people can return to the company after some time.

Restoring a user account

Two years later, the user is back and you need to restore the preserved account.

Deleting a user account

If the employee the employee is dismissed you will delete the account without a backup.

8.1. USER LIFE CYCLE

IdM (Identity Management) supports three user account states:

- **Stage** users are not allowed to authenticate. This is an initial state. Some of the user account properties required for active users cannot be set, for example, group membership.

- **Active** users are allowed to authenticate. All required user account properties must be set in this state.

- **Preserved** users are former active users that are considered inactive and cannot authenticate to IdM. Preserved users retain most of the account properties they had as active users, but they are not part of any user groups.
You can delete user entries permanently from the IdM database.

**IMPORTANT**

Deleted user accounts cannot be restored. When you delete a user account, all the information associated with the account is permanently lost.

A new administrator can only be created by a user with administrator rights, such as the default admin user. If you accidentally delete all administrator accounts, the Directory Manager must create a new administrator manually in the Directory Server.

### 8.2. ADDING USERS IN THE WEB UI

Usually, you need to create a new user account before a new employee starts to work. Such a stage account is not accessible and you need to activate it later.

**NOTE**

Alternatively, you can create an active user account directly. For adding active user, follow the procedure below and add the user account in the **Active users** tab.

**Prerequisites**

- Administrator privileges for managing IdM or User Administrator role.

**Procedure**

1. Log in to the IdM Web UI.
   For details, see [Accessing the IdM Web UI in a web browser](#).

2. Go to **Users** → **Stage Users** tab.
   Alternatively, you can add the user account in the **Users** → **Active users** however, you cannot add user groups to the account.

3. Click the **Add** icon.
4. In the **Add stage user** dialog box, enter **First name** and **Last name** of the new user.

5. [Optional] In the **User login** field, add a login name. If you leave it empty, the IdM server creates the login name in the following pattern: The first letter of the first name and the surname. The whole login name can have up to 32 characters.

6. [Optional] In the **GID** drop down menu, select groups in which the user should be included.

7. [Optional] In the **Password** and **Verify password** fields,

8. Click on the **Add** button.

At this point, you can see the user account in the **Stage Users** table.

**NOTE**

If you click on the user name, you can edit advanced settings, such as adding a phone number, address, or occupation.
8.3. ACTIVATING STAGE USERS IN THE IDM WEB UI

A stage user account must be activated before the user can log in to IdM and before the user can be added to an IdM group. This section describes how to activate stage user accounts.

Prerequisites

- Administrator privileges for managing the IdM Web UI or User Administrator role.
- At least one staged user account in IdM.

Procedure

1. Log in to the IdM Web UI.
   For details, see Accessing the IdM Web UI in a web browser.

2. Go to Users → Stage users tab.

3. Click the check-box of the user account you want to activate.

4. Click on the Activate button.

5. In the Confirmation dialog box, click on the OK button.

If the activation is successfull, the IdM Web UI displays a green confirmation that the user has been activated and the user account has been moved to Active users. The account is active and the user can authenticate to the IdM domain and IdM Web UI. The user is prompted to change their password on the first login.

![Active users table](image)
8.4. DISABLING USER ACCOUNTS IN THE WEB UI

You can disable active user accounts. Disabling a user account deactivates the account, therefore, user accounts cannot be used to authenticate and using IdM services, such as Kerberos, or perform any tasks.

Disabled user accounts still exist within IdM and all of the associated information remains unchanged. Unlike preserved user accounts, disabled user accounts remain in the active state and can be a member of user groups.

NOTE

After disabling a user account, any existing connections remain valid until the user’s Kerberos TGT and other tickets expire. After the ticket expires, the user will not be able to renew it.

Prerequisites

- Administrator privileges for managing the IdM Web UI or User Administrator role.

Procedure

1. Log in to the IdM Web UI. For details, see Accessing the IdM Web UI in a web browser.

2. Go to Users → Active users tab.

3. Click the check-box of the user accounts you want to disable.

4. Click on the Disable button.

5. In the Confirmation dialog box, click on the OK button.

If the disabling procedure has been successful, you can verify in the Status column in the Active users table.
8.5. ENABLING USER ACCOUNTS IN THE WEB UI

With IdM you can enable disabled active user accounts. Enabling a user account activates the disabled account.

Prerequisites

- Administrator privileges for managing the IdM Web UI or User Administrator role.

Procedure

1. Log in to the IdM Web UI.
2. Go to Users → Active users tab.
3. Click the check-box of the user accounts you want to enable.
4. Click on the Enable button.
5. In the Confirmation dialog box, click on the OK button.

If the change has been successful, you can verify in the Status column in the Active users table.

8.6. PRESERVING ACTIVE USERS IN THE IDM WEB UI

Preserving user accounts enables you to remove accounts from the Active users tab, yet keeping these accounts in IdM.

Preserve the user account if the employee leaves the company. If you want to disable user accounts for a couple of weeks or months (parental leave, for example), disable the account. For details, see Section 8.4, “Disabling user accounts in the Web UI”. The preserved accounts are not active and users
cannot use them to access your internal network, however, the account stays in the database with all the data.

You can move the restored accounts back to the active mode.

NOTE

The list of users in the preserved state can provide a history of past user accounts.

Prerequisites

- Administrator privileges for managing the IdM (Identity Management) Web UI or User Administrator role.

Procedure

1. Log in to the IdM Web UI. For details, see Accessing the IdM Web UI in a web browser.

2. Go to Users → Active users tab.

3. Click the check-box of the user accounts you want to preserve.

4. Click on the Delete button.

5. In the Remove users dialog box, switch the Delete mode radio button to preserve.

6. Click on the Delete button.

As a result, the user account is moved to Preserved users.
If you need to restore preserved users, see the Restoring users in the IdM Web UI.

8.7. RESTORING USERS IN THE IDM WEB UI

IdM (Identity Management) enables you to restore preserved user accounts back in the active state.

Prerequisites

- Administrator privileges for managing the IdM Web UI or User Administrator role.

Procedure

1. Log in to the IdM Web UI.
   For details, see Accessing the IdM Web UI in a web browser.

2. Go to Users → Preserved users tab.

3. Click the check-box at the user accounts you want to restore.

4. Click on the Restore button.

5. In the Confirmation dialog box, click on the OK button.

The IdM Web UI displays a green confirmation and moves the user accounts to the Active users tab.

8.8. DELETING USERS IN THE IDM WEB UI

Deleting users is an irreversible operation, causing the user accounts to be permanently deleted from the IdM database, including group memberships and passwords. Any external configuration for the user, such as the system account and home directory, is not deleted, but is no longer accessible through IdM.

You can delete:

- Active users – the IdM Web UI offers you with the options:
  - Preserving users temporarily
    For details, see the Preserving active users in the IdM Web UI.
  - Deleting them permanently

- Stage users – you can just delete stage users permanently.

- Preserved users – you can delete preserved users permanently.

The following procedure describes deleting active users. Similarly, you can delete user accounts on:
- The **Stage users** tab
- The **Preserved users** tab

**Prerequisites**

- Administrator privileges for managing the IdM Web UI or User Administrator role.

**Procedure**

1. Log in to the IdM Web UI.
   For details, see [Accessing the IdM Web UI in a web browser](#).

2. Go to **Users → Active users** tab.
   Alternatively, you can delete the user account in the **Users → Stage users** or **Users → Preserved users**.

3. Click the **Delete** icon.

4. In the **Remove users** dialog box, switch the **Delete mode** radio button to **delete**.

5. Click on the **Delete** button.

The users accounts have been permanently deleted from IdM.
CHAPTER 9. MANAGING USER ACCOUNTS USING THE COMMAND LINE

This chapter includes basic description of user life cycle in IdM (Identity Management). The following sections show you how to:

- Create user accounts
- Activate stage user accounts
- Preserve user accounts
- Delete active, stage, or preserved user accounts
- Restore preserved user accounts

9.1. USER LIFE CYCLE

IdM (Identity Management) supports three user account states:

- **Stage** users are not allowed to authenticate. This is an initial state. Some of the user account properties required for active users cannot be set, for example, group membership.

- **Active** users are allowed to authenticate. All required user account properties must be set in this state.

- **Preserved** users are former active users that are considered inactive and cannot authenticate to IdM. Preserved users retain most of the account properties they had as active users, but they are not part of any user groups.

You can delete user entries permanently from the IdM database.

**IMPORTANT**

Deleted user accounts cannot be restored. When you delete a user account, all the information associated with the account is permanently lost.
A new administrator can only be created by a user with administrator rights, such as the default admin user. If you accidentally delete all administrator accounts, the Directory Manager must create a new administrator manually in the Directory Server.

### 9.2. ADDING USERS USING THE COMMAND LINE

You can add user as:

- **Active** — user accounts which can be actively used by their users.
- **Stage** — users cannot use these accounts. Use it if you want to prepare new user accounts. When users are ready to use their accounts, then you can activate them.

The following procedure describes adding active users to the IdM server with the `ipa user-add` command.

Similarly, you can create stage user accounts with the `ipa stageuser-add` command.

**NOTE**

IdM automatically assigns a unique user ID (UID) to the new user accounts. You can also do this manually, however, the server does not validate whether the UID number is unique. Due to this, multiple user entries might have the same ID number assigned. Red Hat recommends to prevent having multiple entries with the same UID.

**Prerequisites**

- Administrator privileges for managing IdM or User Administrator role.
- Obtained a Kerberos ticket. For details, see [Using kinit to log in to IdM manually](#).

**Procedure**

1. Open terminal and connect to the IdM server.
2. Add user login, user’s first name, last name and optionally, you can also add their email address.

```
$ ipa user-add user_login --first=first_name --last=last_name --email=email_address
```

IdM supports user names that can be described by the following regular expression:

```
[a-zA-Z0-9_.][a-zA-Z0-9_.-]{0,252}[a-zA-Z0-9_.$-]?
```

**NOTE**

User names ending with the trailing dollar sign ($) are supported to enable Samba 3.x machine support.

If you add a user name containing uppercase characters, IdM automatically converts the name to lowercase when saving it. Therefore, IdM always requires to enter user names in lowercase when logging in. Additionally, it is not possible to add user names which differ only in letter casing, such as `user` and `User`.
The default maximum length for user names is 32 characters. To change it, use the `ipa config-mod --maxusername` command. For example, to increase the maximum user name length to 64 characters:

```
$ ipa config-mod --maxusername=64
  Maximum username length: 64
...
```

The `ipa user-add` command includes a lot of parameters. To list them all, use the `ipa help` command:

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

For details about `ipa help` command, see [What is the IPA help](#).

You can verify if the new user account is successfully created by listing all IdM user accounts:

```
$ ipa $ ipa user-find
```

This command lists all user accounts with details.

### 9.3. ACTIVATING USERS USING THE COMMAND LINE

To activate a user account by moving it from stage to active, use the `ipa stageuser-activate` command.

**Prerequisites**

- Administrator privileges for managing IdM or User Administrator role.
- Obtained a Kerberos ticket. For details, see [Using kinit to log in to IdM manually](#).

**Procedure**

1. Open terminal and connect to the IdM server.
2. Activate the user account with the following command:

```
$ ipa stageuser-activate user_login
-------------------------
Stage user user_login activated
-------------------------
```

You can verify if the new user account is successfully created by listing all IdM user accounts:

```
$ ipa $ ipa user-find
```

This command lists all user accounts with details.

### 9.4. PRESERVING USERS USING THE COMMAND LINE

To preserve a user account, use the `ipa user-del` or `ipa stageuser-del` commands.
**Prerequisites**

- Administrator privileges for managing IdM or User Administrator role.
- Obtained a Kerberos ticket. For details, see *Using kinit to log in to IdM manually*.

**Procedure**

1. Open terminal and connect to the IdM server.

2. Preserve the user account with the following command:

   ```bash
   $ ipa user-del --preserve user_login
   --------------------
   Deleted user "user_login"
   --------------------
   ```

**9.5. DELETING USERS USING THE COMMAND LINE**

IdM (Identity Management) enables you to delete users permanently. You can delete:

- Active users with the following command: `ipa user-del`
- Stage users with the following command: `ipa stageuser-del`
- Preserved users with the following command: `ipa user-del`

When deleting multiple users, use the `--continue` option to force the command to continue regardless of errors. A summary of the successful and failed operations is printed to the stdout standard output stream when the command completes.

```bash
$ ipa user-del --continue user1 user2 user3
```

If you do not use `--continue`, the command proceeds with deleting users until it encounters an error, after which it stops and exits.

**Prerequisites**

- Administrator privileges for managing IdM or User Administrator role.
- Obtained a Kerberos ticket. For details, see *Using kinit to log in to IdM manually*.

**Procedure**

1. Open terminal and connect to the IdM server.

2. Delete the user account with the following command:

   ```bash
   $ ipa user-del user_login
   --------------------
   Deleted user "user_login"
   --------------------
   ```

The user account has been permanently deleted from IdM.
9.6. RESTORING USERS USING THE COMMAND LINE

You can restore a preserved users to:

- Active users: ipa user-undel
- Stage users: ipa user-stage

Restoring a user account does not restore all of the account’s previous attributes. For example, the user’s password is not restored and must be set again.

Prerequisites

- Administrator privileges for managing IdM or User Administrator role.
- Obtained a Kerberos ticket. For details, see Using kinit to log in to IdM manually.

Procedure

1. Open terminal and connect to the IdM server.

2. Activate the user account with the following command:

   ```
   $ ipa user-undel user_login
   --------------------------
   Undeleted user account "user_login"
   --------------------------
   
   Alternatively, you can restore user accounts as staged:
   
   $ ipa user-stage user_login
   --------------------------
   Staged user account "user_login"
   --------------------------
   
   You can verify if the new user account is successfully created by listing all IdM user accounts:
   
   $ ipa $ ipa user-find
   
   This command lists all user accounts with details.
CHAPTER 10. MANAGING USER GROUPS IN IDM CLI

This chapter introduces user groups management using the IdM CLI.

A user group is a set of users with common privileges, password policies, and other characteristics.

A user group in Identity Management (IdM) can include:

- IdM users
- other IdM user groups
- external users, which are users that exist outside of IdM

10.1. THE DIFFERENT GROUP TYPES IN IDM

IdM supports the following types of groups:

**POSIX groups (the default)**

POSIX groups support Linux POSIX attributes for their members. Note that groups that interact with Active Directory cannot use POSIX attributes.

POSIX attributes identify users as separate entities. Examples of POSIX attributes relevant to users include `uidNumber`, a user number (UID), and `gidNumber`, a group number (GID).

**Non-POSIX groups**

Non-POSIX groups do not support POSIX attributes. For example, these groups do not have a GID defined.

All members of this type of group must belong to the IdM domain.

**External groups**

Use external groups to add group members that exist in an identity store outside of the IdM domain, such as:

- A local system
- An Active Directory domain
- A directory service

External groups do not support POSIX attributes. For example, these groups do not have a GID defined.

Table 10.1. User groups created by default

<table>
<thead>
<tr>
<th>Group name</th>
<th>Default group members</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipausers</td>
<td>All IdM users</td>
</tr>
<tr>
<td>admins</td>
<td>Users with administrative privileges, including the default admin user</td>
</tr>
<tr>
<td>editors</td>
<td>This is a legacy group that no longer has any special privileges</td>
</tr>
</tbody>
</table>
When you add a user to a user group, the user gains the privileges and policies associated with the group. For example, to grant administrative privileges to a user, add the user to the `admins` group.

In addition, IdM creates *user private groups* by default whenever a new user is created in IdM. For more information about private groups, see Adding users without a private group.

### 10.2. DIRECT AND INDIRECT GROUP MEMBERS

User group attributes in IdM apply to both direct and indirect members: when group B is a member of group A, all users in group B are considered indirect members of group A.

For example, in the following diagram:

- User 1 and User 2 are *direct members* of group A.
- User 3, User 4, and User 5 are *indirect members* of group A.

**Figure 10.1. Direct and Indirect Group Membership**

If you set a password policy for user group A, the policy also applies to all users in user group B.

### 10.3. ADDING A USER GROUP USING IDM CLI

This section describes how to add a user group using IdM CLI.

**Prerequisites**

- You must be logged in as the administrator. For details, see Using kinit to log in to IdM manually.

**Procedure**

- Add a user group by using the `ipa group-add group_name` command. For example, to create `group_a`:

  ```
  $ ipa group-add group_a
  ```
Added group "group_a"

Group name: group_a
GID: 1133400009

By default, `ipa group-add` adds a POSIX user group. To specify a different group type, add options to `ipa group-add`:

- `--nonposix` to create a non-POSIX group
- `--external` to create an external group
  
  For details on group types, see [The different group types in IdM](#).

You can specify a custom GID when adding a user group by using the `--gid=custom_GID` option. If you do this, be careful to avoid ID conflicts. If you do not specify a custom GID, IdM automatically assigns a GID from the available ID range.

### 10.4. SEARCHING FOR USER GROUPS USING IDM CLI

This section describes how to search for existing user groups using IdM CLI.

#### Procedure

- Display all user groups by using the `ipa group-find` command. To specify a group type, add options to `ipa group-find`:
  
  - Display all POSIX groups using the `ipa group-find --posix` command.
  - Display all non-POSIX groups using the `ipa group-find --nonposix` command.
  - Display all external groups using the `ipa group-find --external` command.
    
    For more information on different group types, see [The different group types in IdM](#).

### 10.5. DELETING A USER GROUP USING IDM CLI

This section describes how to delete a user group using IdM CLI. Note that deleting a group does not delete the group members from IdM.

#### Prerequisites

- You must be logged in as the administrator. For details, see [Using kinit to log in to IdM manually](#).

#### Procedure

- Delete a user group by using the `ipa group-del group_name` command. For example, to delete `group_a`:
  
  ```bash
  $ ipa group-del group_a
  --------------------------
  Deleted group "group_a"
  --------------------------
  ```
10.6. ADDING A MEMBER TO A USER GROUP USING IDM CLI

This section describes how to add a member to a user group using IdM CLI. You can add both users and user groups as members of a user group. For more information, see The different group types in IdM and Direct and indirect group members.

Prerequisites

- You must be logged in as the administrator. For details, see Using kinit to log in to IdM manually.

Procedure

- Add a member to a user group by using the `ipa group-add-member` command. Specify the type of member using these options:
  - `--users` adds an IdM user
  - `--external` adds a user that exists outside the IdM domain, in the format of `DOMAIN\user_name` or `user_name@domain`
  - `--groups` adds an IdM user group

For example, to add group_b as a member of group_a:

```
$ ipa group-add-member group_a --groups=group_b
Group name: group_a  
GID: 1133400009
Member users: user_a
Member groups: group_b
Indirect Member users: user_b
-------------------------
Number of members added 1
-------------------------
```

Members of group_b are now indirect members of group_a.

**IMPORTANT**

When adding a group as a member of another group, do not create recursive groups. For example, if Group A is a member of Group B, do not add Group B as a member of Group A. Recursive groups can cause unpredictable behavior.

**NOTE**

After you add a member to a user group, the update may take some time to spread to all clients in your Identity Management environment. This is because when any given host resolves users, groups and netgroups, the System Security Services Daemon (SSSD) first looks into its cache and performs server lookups only for missing or expired records.

10.7. ADDING USERS WITHOUT A USER PRIVATE GROUP

By default, IdM creates user private groups (UPGs) whenever a new user is created in IdM. UPGs are a specific group type:
The UPG has the same name as the newly created user.

- The user is the only member of the UPG. The UPG cannot contain any other members.
- The GID of the private group matches the UID of the user.

However, it is possible to add users without creating a UPG.

### 10.7.1. Users without a user private group

If a NIS group or another system group already uses the GID that would be assigned to a user private group, it is necessary to avoid creating a UPG.

You can do this in two ways:

- Add a new user without a UPG, without disabling private groups globally. See Adding a user without a user private group when private groups are globally enabled.

- Disable UPGs globally for all users, then add a new user. See Disabling user private groups globally for all users and Adding a user when user private groups are globally disabled.

In both cases, IdM will require specifying a GID when adding new users, otherwise the operation will fail. This is because IdM requires a GID for the new user, but the default user group `ipausers` is a non-POSIX group and therefore does not have an associated GID. The GID you specify does not have to correspond to an already existing group.

**NOTE**

Specifying the GID does not create a new group. It only sets the GID attribute for the new user, because the attribute is required by IdM.

### 10.7.2. Adding a user without a user private group when private groups are globally enabled

You can add a user without creating a user private group (UPG) even when UPGs are enabled on the system. This requires manually setting a GID for the new user. For details on why this is needed, see Section 10.7.1, “Users without a user private group”.

**Procedure**

- To prevent IdM from creating a UPG, add the `--noprivate` option to the `ipa user-add` command.
  
  Note that for the command to succeed, you must specify a custom GID. For example, to add a new user with GID 10000:

  ```
  $ ipa user-add jsmith --first=John --last=Smith --noprivate --gid 10000
  ```

### 10.7.3. Disabling user private groups globally for all users

You can disable user private groups (UPGs) globally. This prevents the creation of UPGs for all new users. Existing users are unaffected by this change.

**Procedure**
1. Obtain administrator privileges:

   \$ kinit admin

2. IdM uses the Directory Server Managed Entries Plug-in to manage UPGs. List the instances of the plug-in:

   \$ ipa-managed-entries --list

3. To ensure IdM does not create UPGs, disable the plug-in instance responsible for managing user private groups:

   \$ ipa-managed-entries -e "UPG Definition" disable

   Disabling Plugin

   **NOTE**

   To re-enable the **UPG Definition** instance later, use the `ipa-managed-entries -e "UPG Definition" enable` command.

4. Restart Directory Server to load the new configuration.

   \$ sudo systemctl restart dirsrv.target

   To add a user after UPGs have been disabled, you need to specify a GID. For more information, see [Adding a user when user private groups are globally disabled](#).

**Verification steps**

- To check if UPGs are globally disabled, use the disable command again:

  \$ ipa-managed-entries -e "UPG Definition" disable

  Plugin already disabled

**10.7.4. Adding a user when user private groups are globally disabled**

When user private groups (UPGs) are disabled globally, IdM does not assign a GID to a new user automatically. To successfully add a user, you must assign a GID manually or by using an automember rule. For details on why this is required, see [Section 10.7.1, “Users without a user private group”](#).

**Prerequisites**

- UPGs must be disabled globally for all users. For more information, see [Disabling user private groups globally for all users](#).

**Procedure**

- To make sure adding a new user succeeds when creating UPGs is disabled, choose one of the following:

  - Specify a custom GID when adding a new user. The GID does not have to correspond to an already existing user group.
For example, when adding a user from the command line, add the \texttt{--gid} option to the \texttt{ipa user-add} command.

- Use an automember rule to add the user to an existing group with a GID.

### 10.8. VIEWING GROUP MEMBERS USING IDM CLI

This section describes how to view members of a group using IdM CLI. You can view both direct and indirect group members. For more information, see \textit{Direct and indirect group members}.

**Procedure:**

- To list members of a group, use the \texttt{ipa group-show group\_name} command. For example:

  
  
  $\text{ipa group-show group\_a}$
  
  ...
  
  Member users: user\_a
  
  Member groups: group\_b
  
  Indirect Member users: user\_b

**NOTE**

The list of indirect members does not include external users from trusted Active Directory domains. The Active Directory trust user objects are not visible in the Identity Management interface because they do not exist as LDAP objects within Identity Management.

### 10.9. REMOVING A MEMBER FROM A USER GROUP USING IDM CLI

This section describes how to remove a member from a user group using IdM CLI.

**Prerequisites**

- You must be logged in as the administrator. For details, see \textit{Using kinit to log in to IdM manually}.

**Procedure**

1. \textit{Optional}. Use the \texttt{ipa group-show} command to confirm that the group includes the member you want to remove.

2. Remove a member from a user group by using the \texttt{ipa group-remove-member} command. Specify members to remove using these options:

   - \texttt{--users} removes an IdM user
   
   - \texttt{--external} removes a user that exists outside the IdM domain, in the format of \texttt{DOMAIN\_user\_name} or \texttt{user\_name@domain}
   
   - \texttt{--groups} removes an IdM user group

For example, to remove \texttt{user1}, \texttt{user2}, and \texttt{group1} from a group called \texttt{group\_name}:

  
  $\text{ipa group-remove-member group\_name --users=user1 --users=user2 --groups=group1}$
CHAPTER 11. MANAGING USER GROUPS IN IDM WEB UI

This chapter introduces user groups management using the IdM web UI.

A user group is a set of users with common privileges, password policies, and other characteristics.

A user group in Identity Management (IdM) can include:

- IdM users
- other IdM user groups
- external users, which are users that exist outside of IdM

11.1. THE DIFFERENT GROUP TYPES IN IDM

IdM supports the following types of groups:

POSIX groups (the default)

POSIX groups support Linux POSIX attributes for their members. Note that groups that interact with Active Directory cannot use POSIX attributes.

POSIX attributes identify users as separate entities. Examples of POSIX attributes relevant to users include **uidNumber**, a user number (UID), and **gidNumber**, a group number (GID).

Non-POSIX groups

Non-POSIX groups do not support POSIX attributes. For example, these groups do not have a GID defined.

All members of this type of group must belong to the IdM domain.

External groups

Use external groups to add group members that exist in an identity store outside of the IdM domain, such as:

- A local system
- An Active Directory domain
- A directory service

External groups do not support POSIX attributes. For example, these groups do not have a GID defined.

Table 11.1. User groups created by default

<table>
<thead>
<tr>
<th>Group name</th>
<th>Default group members</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipausers</td>
<td>All IdM users</td>
</tr>
<tr>
<td>admins</td>
<td>Users with administrative privileges, including the default admin user</td>
</tr>
<tr>
<td>editors</td>
<td>This is a legacy group that no longer has any special privileges</td>
</tr>
<tr>
<td>Group name</td>
<td>Default group members</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>trust admins</td>
<td>Users with privileges to manage the Active Directory trusts</td>
</tr>
</tbody>
</table>

When you add a user to a user group, the user gains the privileges and policies associated with the group. For example, to grant administrative privileges to a user, add the user to the `admins` group.

In addition, IdM creates *user private groups* by default whenever a new user is created in IdM. For more information about private groups, see [Adding users without a private group](#).

### 11.2. DIRECT AND INDIRECT GROUP MEMBERS

User group attributes in IdM apply to both direct and indirect members: when group B is a member of group A, all users in group B are considered indirect members of group A.

For example, in the following diagram:

- User 1 and User 2 are *direct members* of group A.
- User 3, User 4, and User 5 are *indirect members* of group A.

**Figure 11.1. Direct and Indirect Group Membership**

If you set a password policy for user group A, the policy also applies to all users in user group B.

### 11.3. ADDING A USER GROUP USING IDM WEB UI

This section describes how to add a user group using the IdM Web UI.

**Prerequisites**

- You are logged in to the IdM Web UI.

**Procedure**

1. Click **Identity → Groups**, and select **User Groups** in the left sidebar.
2. Click **Add** to start adding the group.
3. Fill out the information about the group. For more information about user group types, see The different group types in IdM.
   You can specify a custom GID for the group. If you do this, be careful to avoid ID conflicts. If you do not specify a custom GID, IdM automatically assigns a GID from the available ID range.

4. Click Add to confirm.

11.4. DELETING A USER GROUP USING IDM WEB UI

This section describes how to delete a user group using the IdM Web UI. Note that deleting a group does not delete the group members from IdM.

Prerequisites

- You are logged in to the IdM Web UI.

Procedure

1. Click Identity → Groups and select User Groups.
2. Select the group to delete.
3. Click Delete.
4. Click Delete to confirm.

11.5. ADDING A MEMBER TO A USER GROUP USING IDM WEB UI
You can add both users and user groups as members of a user group. For more information, see The different group types in IdM and Direct and indirect group members.

Prerequisites

- You are logged in to the IdM Web UI.

Procedure

1. Click Identity → Groups and select User Groups in the left sidebar.

2. Click the name of the group.

3. Select the type of group member you want to add: Users, User Groups, or External.

4. Click Add.

5. Select the check box next to one or more members you want to add.

6. Click the rightward arrow to move the selected members to the group.

7. Click Add to confirm.

11.6. VIEWING GROUP MEMBERS USING IDM WEB UI
This section describes how to view members of a group using the IdM Web UI. You can view both direct and indirect group members. For more information, see Direct and indirect group members.

Prerequisites

- You are logged in to the IdM Web UI.

Procedure

1. Select Identity → Groups.

2. Select User Groups in the left sidebar.

3. Click the name of the group you want to view.

4. Switch between Direct Membership and Indirect Membership.

11.7. REMOVING A MEMBER FROM A USER GROUP USING IDM WEB UI

This section describes how to remove a member from a user group using the IdM Web UI.

Prerequisites

- You are logged in to the IdM Web UI.

Procedure

1. Click Identity → Groups and select User Groups in the left sidebar.

2. Click the name of the group.

3. Select the type of group member you want to remove: Users, User Groups, or External.

4. Select the check box next to the member you want to remove.

5. Click Delete.

6. Click Delete to confirm.
CHAPTER 12. MANAGING HOSTS IN IDM CLI

This chapter introduces hosts and host entries in Identity Management (IdM), and the following operations performed when managing hosts and host entries in IdM CLI:

- Host Enrollment
- Adding IdM host entries
- Deleting IdM host entries
- Re-enrolling hosts
- Renaming hosts
- Disabling hosts
- Re-enabling hosts

The chapter also contains an overview table of the prerequisites, the context, and the consequences of these operations.

12.1. HOSTS IN IDM

Identity Management (IdM) manages these identities:

- Users
- Services
- Hosts

A host represents a machine. As an IdM identity, a host has an entry in the IdM LDAP, that is the 389 Directory Server instance of the IdM server.

The host entry in IdM LDAP is used to establish relationships between other hosts and even services within the domain. These relationships are part of delegating authorization and control to hosts within the domain. Any host can be used in host-based access control (HBAC) rules.

IdM domain establishes a commonality between machines, with common identity information, common policies, and shared services. Any machine that belongs to a domain functions as a client of the domain, which means it uses the services that the domain provides. IdM domain provides three main services specifically for machines:

- DNS
- Kerberos
- Certificate management

Hosts in IdM are closely connected with the services running on them:

- Service entries are associated with a host.
- A host stores both the host and the service Kerberos principals.
12.2. HOST ENROLLMENT

This section describes enrolling hosts as IdM clients and what happens during and after the enrollment. The section compares the enrollment of IdM hosts and IdM users. The section also outlines alternative types of authentication available to hosts.

Enrolling a host consists of:

- Creating a host entry in IdM LDAP: possibly using the `ipa host-add` command in IdM CLI, or the equivalent IdM Web UI operation.
- Configuring IdM services on the host, for example the System Security Services Daemon (SSSD), Kerberos, and certmonger, and joining the host to the IdM domain.

The two actions can be performed separately or together.

If performed separately, they allow for dividing the two tasks between two users with different levels of privilege. This is useful for bulk deployments.

The `ipa-client-install` command can perform the two actions together. The command creates a host entry in IdM LDAP if that entry does not exist yet, and configures both the Kerberos and SSSD services for the host. The command brings the host within the IdM domain and allows it to identify the IdM server it will connect with. If the host belongs to a DNS zone managed by IdM, `ipa-client-install` adds DNS records for the host too. The command must be run on the client.

12.2.1. User privileges required for host enrollment

The host enrollment operation requires authentication to prevent an unprivileged user from adding unwanted machines to the IdM domain. The privileges required depend on several factors, for example:

- If a host entry is created separately from running `ipa-client-install`
- If a one-time password (OTP) is used for enrollment

User privileges for optionally manually creating a host entry in IdM LDAP

The user privilege required for creating a host entry in IdM LDAP using the `ipa host-add` CLI command or the IdM Web UI is Host Administrators. The Host Administrators privilege can be obtained through the IT Specialist role.

User privileges for joining the client to the IdM domain

Hosts are configured as IdM clients during the execution of the `ipa-client-install` command. The level of credentials required for executing the `ipa-client-install` command depends on which of the following enrolling scenarios you find yourself in:

- The host entry in IdM LDAP does not exist. For this scenario, you need a full administrator’s credentials or the Host Administrators role. A full administrator is a member of the admins group. The Host Administrators role provides privileges to add hosts and enroll hosts. For details about this scenario, see Installing a client using user credentials: interactive installation.

- The host entry in IdM LDAP exists. For this scenario, you need a limited administrator’s credentials to execute `ipa-client-install` successfully. The limited administrator in this case has the Enrollment Administrator role, which provides the Host Enrollment privilege. For details, see Installing a client using user credentials: interactive installation.

- The host entry in IdM LDAP exists, and an OTP has been generated for the host by a full or limited administrator. For this scenario, you can install an IdM client as an ordinary user if you run the `ipa-client-install` command with the `--password` option, supplying the correct OTP. For
After enrollment, IdM hosts authenticate every new session to be able to access IdM resources. Machine authentication is required for the IdM server to trust the machine and to accept IdM connections from the client software installed on that machine. After authenticating the client, the IdM server can respond to its requests.

### 12.2.2. Enrollment and authentication of IdM hosts and users: comparison

There are many similarities between users and hosts in IdM. This section describes some of the similarities that can be observed during the enrollment stage as well as those that concern authentication during the deployment stage.

- **The enrollment stage** (Table 12.1, “User and host enrollment”):
  - An administrator can create an LDAP entry for both a user and a host before the user or host actually join IdM: for the stage user, the command is `ipa stageuser-add`; for the host, the command is `ipa host-add`.
  - A file containing a **key table** or, abbreviated, keytab, a symmetric key resembling to some extent a user password, is created during the execution of the `ipa-client-install` command on the host, resulting in the host joining the IdM realm. Analogically, a user is asked to create a password when they activate their account, thus joining the IdM realm.
  - While the user password is the default authentication method for a user, the keytab is the default authentication method for a host. The keytab is stored in a file on the host.

<table>
<thead>
<tr>
<th>Action</th>
<th>User</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-enrollment</td>
<td><code>ipa stageuser-add user_name [-password]</code></td>
<td><code>ipa host-add host_name [--random]</code></td>
</tr>
<tr>
<td>Activating the account</td>
<td><code>ipa stageuser-activate user_name</code></td>
<td><code>ipa-client install [--password]</code></td>
</tr>
<tr>
<td></td>
<td>(must be run on the host itself)</td>
<td></td>
</tr>
</tbody>
</table>

- **The deployment stage** (Table 12.2, “User and host session authentication”):
  - When a user starts a new session, the user authenticates using a password; similarly, every time it is switched on, the host authenticates by presenting its keytab file. The System Security Services Daemon (SSSD) manages this process in the background.
  - If the authentication is successful, the user or host obtains a Kerberos ticket granting ticket (TGT).
  - The TGT is then used to obtain specific tickets for specific services.

<table>
<thead>
<tr>
<th>Default means of authentication</th>
<th>User</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Password</td>
<td>Keytabs</td>
</tr>
</tbody>
</table>
Starting a session
(ordinary user)

$ kinit user_name

[switch on the host]

The result of successful authentication

TGT to be used to obtain access to specific services

TGT to be used to obtain access to specific services

TGTs and other Kerberos tickets are generated as part of the Kerberos services and policies defined by the server. The initial granting of a Kerberos ticket, the renewing of the Kerberos credentials, and even the destroying of the Kerberos session are all handled automatically by the IdM services.

12.2.3. Alternative authentication options for IdM hosts

Apart from keytabs, IdM supports two other types of machine authentication:

- SSH keys. The SSH public key for the host is created and uploaded to the host entry. From there, the System Security Services Daemon (SSSD) uses IdM as an identity provider and can work in conjunction with OpenSSH and other services to reference the public keys located centrally in IdM.

- Machine certificates. In this case, the machine uses an SSL certificate that is issued by the IdM server’s certificate authority and then stored in IdM’s Directory Server. The certificate is then sent to the machine to present when it authenticates to the server. On the client, certificates are managed by a service called certmonger.

12.3. HOST OPERATIONS

This section lists the most common operations related to host enrollment and enablement, and explains the prerequisites, the context, and the consequences of performing them.

Table 12.3. Host operations part 1

<table>
<thead>
<tr>
<th>Action</th>
<th>What are the prerequisites of the action?</th>
<th>When does it make sense to run the command?</th>
<th>How is the action performed by a system administrator? What command(s) does he run?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enrolling a client</td>
<td>see Preparing the system for Identity Management client installation in Installing_Identity_Management</td>
<td>When you want the host to join the IdM realm.</td>
<td>Enrolling machines as clients in the IdM domain is a two-part process. A host entry is created for the client (and stored in the 389 Directory Server instance) when the <code>ipa host-add</code> command is run, and then a keytab is created to provision the client. Both parts are performed automatically by the <code>ipa-client-install</code> command. It is also possible to perform those steps separately; this allows for administrators to prepare machines and IdM in advance of actually configuring the clients. This allows more flexible setup scenarios, including bulk deployments.</td>
</tr>
<tr>
<td>Action</td>
<td>What are the prerequisites of the action?</td>
<td>When does it make sense to run the command?</td>
<td>How is the action performed by a system administrator? What command(s) does he run?</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Disabling a client</td>
<td>The host must have an entry in IdM. The host needs to have an active keytab.</td>
<td>When you want to remove the host from the IdM realm temporarily, perhaps for maintenance purposes.</td>
<td>ipa host-disable host_name</td>
</tr>
<tr>
<td>Enabling a client</td>
<td>The host must have an entry in IdM.</td>
<td>When you want the temporarily disabled host to become active again.</td>
<td>ipa-getkeytab</td>
</tr>
<tr>
<td>Re-enrolling a client</td>
<td>The host must have an entry in IdM.</td>
<td>When the original host has been lost but you have installed a host with the same host name.</td>
<td>ipa-client-install --keytab or ipa-client-install --force-join</td>
</tr>
<tr>
<td>Un-enrolling a client</td>
<td>The host must have an entry in IdM.</td>
<td>When you want to remove the host from the IdM realm permanently.</td>
<td>ipa-client-install --uninstall</td>
</tr>
</tbody>
</table>

Table 12.4. Host operations part 2

<table>
<thead>
<tr>
<th>Action</th>
<th>On which machine can the administrator run the command(s)?</th>
<th>What happens when the action is performed? What are the consequences for the host’s functioning in IdM? What limitations are introduced/removed?</th>
</tr>
</thead>
</table>

Red Hat Enterprise Linux 8 Configuring and managing Identity Management

72
<table>
<thead>
<tr>
<th>Action</th>
<th>On which machine can the administrator run the command(s)?</th>
<th>What happens when the action is performed? What are the consequences for the host’s functioning in IdM? What limitations are introduced/removed?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enrolling a client</td>
<td>In the case of a two-step enrollment: ipa host-add can be run on any IdM client; the second step of ipa-client-install must be run on the client itself</td>
<td>By default this configures SSSD to connect to an IdM server for authentication and authorization. Optionally one can instead configure the Pluggable Authentication Module (PAM) and the Name Switching Service (NSS) to work with an IdM server over Kerberos and LDAP.</td>
</tr>
<tr>
<td>Disabling a client</td>
<td>Any machine in IdM, even the host itself</td>
<td>The host’s Kerberos key and SSL certificate are invalidated, and all services running on the host are disabled.</td>
</tr>
<tr>
<td>Enabling a client</td>
<td>Any machine in IdM. If run on the disabled host, LDAP credentials need to be supplied.</td>
<td>The host’s Kerberos key and the SSL certificate are made valid again, and all IdM services running on the host are re-enabled.</td>
</tr>
<tr>
<td>Re-enrolling a client</td>
<td>The host to be re-enrolled. LDAP credentials need to be supplied.</td>
<td>A new Kerberos key is generated for the host, replacing the previous one.</td>
</tr>
<tr>
<td>Un-enrolling a client</td>
<td>The host to be un-enrolled.</td>
<td>The command unconfigures IdM and attempts to return the machine to its previous state. Part of this process is to unenroll the host from the IdM server. Unenrollment consists of disabling the principal key on the IdM server. The machine principal in /etc/krb5.keytab (host/&lt;fqdn&gt;@REALM) is used to authenticate to the IdM server to unenroll itself. If this principal does not exist then unenrollment will fail and an administrator will need to disable the host principal (ipa host-disable &lt;fqdn&gt;).</td>
</tr>
</tbody>
</table>

### 12.4. HOST ENTRY IN IDM LDAP

This section describes what a host entry in Identity Management (IdM) looks like and what attributes it can contain.

An LDAP host entry contains all relevant information about the client within IdM:

- Service entries associated with the host
- The host and service principal
Access control rules

Machine information, such as its physical location and operating system

**NOTE**

Note that the IdM Web UI **Identity → Hosts** tab does not show all the information about a particular host stored in the IdM LDAP.

### 12.4.1. Host entry configuration properties

A host entry can contain information about the host that is outside its system configuration, such as its physical location, MAC address, keys, and certificates.

This information can be set when the host entry is created if it is created manually. Alternatively, most of this information can be added to the host entry after the host is enrolled in the domain.

**Table 12.5. Host Configuration Properties**

<table>
<thead>
<tr>
<th>UI Field</th>
<th>Command-Line Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>--desc=description</td>
<td>A description of the host.</td>
</tr>
<tr>
<td>Locality</td>
<td>--locality=locality</td>
<td>The geographic location of the host.</td>
</tr>
<tr>
<td>Location</td>
<td>--location=location</td>
<td>The physical location of the host, such as its data center rack.</td>
</tr>
<tr>
<td>Platform</td>
<td>--platform=string</td>
<td>The host hardware or architecture.</td>
</tr>
<tr>
<td>Operating system</td>
<td>--os=string</td>
<td>The operating system and version for the host.</td>
</tr>
<tr>
<td>MAC address</td>
<td>--macaddress=address</td>
<td>The MAC address for the host. This is a multi-valued attribute. The MAC address is used by the NIS plug-in to create a NIS <em>ethers</em> map for the host.</td>
</tr>
<tr>
<td>SSH public keys</td>
<td>--sshpubkey=string</td>
<td>The full SSH public key for the host. This is a multi-valued attribute, so multiple keys can be set.</td>
</tr>
</tbody>
</table>
### 12.5. Adding IDM Host Entries from IDM CLI

This section describes how to add host entries in Identity Management (IdM) using the command-line interface (CLI).

Host entries are created using the `host-add` command. This command adds the host entry to the IdM Directory Server. Consult the `ipa host` manpage by typing `ipa help host` in your CLI to get the full list of options available with `host-add`.

There are a few different scenarios when adding a host to IdM:

- At its most basic, specify only the client host name to add the client to the Kerberos realm and to create an entry in the IdM LDAP server:

  ```
  $ ipa host-add client1.example.com
  
  $ ipa host-add --ip-address=192.168.166.31 client1.example.com
  ```

- If the IdM server is configured to manage DNS, add the host to the DNS resource records using the `--ip-address` option.

  Example 12.1. Creating Host Entries with Static IP Addresses

  ```
  $ ipa host-add --ip-address=192.168.166.31 client1.example.com
  ```

---

<table>
<thead>
<tr>
<th>UI Field</th>
<th>Command-Line Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principal name (not editable)</td>
<td>--principalname=principal</td>
<td>The Kerberos principal name for the host. This defaults to the host name during the client installation, unless a different principal is explicitly set in the <code>-p</code>. This can be changed using the command-line tools, but cannot be changed in the UI.</td>
</tr>
<tr>
<td>Set One-Time Password</td>
<td>--password=string</td>
<td>This option sets a password for the host which can be used in bulk enrollment.</td>
</tr>
<tr>
<td>-</td>
<td>--random</td>
<td>This option generates a random password to be used in bulk enrollment.</td>
</tr>
<tr>
<td>-</td>
<td>--certificate=string</td>
<td>A certificate blob for the host.</td>
</tr>
<tr>
<td>-</td>
<td>--updatedns</td>
<td>This sets whether the host can dynamically update its DNS entries if its IP address changes.</td>
</tr>
</tbody>
</table>
If the host to be added does not have a static IP address or if the IP address is not known at the time the client is configured, use the --force option with the ipa host-add command.

**Example 12.2. Creating Host Entries with DHCP**

```bash
ipa host-add --force client1.example.com
```

For example, laptops may be preconfigured as IdM clients, but they do not have IP addresses at the time they are configured. Using --force essentially creates a placeholder entry in the IdM DNS service. When the DNS service dynamically updates its records, the host’s current IP address is detected and its DNS record is updated.

### 12.6. DELETING HOST ENTRIES FROM IDM CLI

- Use the host-del command to delete host records. If your IdM domain has integrated DNS, use the --updatedns option to remove the associated records of any kind for the host from the DNS:

  ```bash
  ipa host-del --updatedns client1.example.com
  ```

### 12.7. RE-ENROLLING AN IDENTITY MANAGEMENT CLIENT

#### 12.7.1. Client re-enrollment in IdM

This section describes how to re-enroll an Identity Management (IdM) client.

If a client machine has been destroyed and lost connection with the IdM servers, for example due to the client’s hardware failure, and you still have its keytab, you can re-enroll the client. In this scenario, you want to get the client back in the IdM environment with the same hostname.

During the re-enrollment, the client generates a new Kerberos key and SSH keys, but the identity of the client in the LDAP database remains unchanged. After the re-enrollment, the host has its keys and other information in the same LDAP object with the same FQDN as previously, before the machine’s loss of connection with the IdM servers.

**IMPORTANT**

You can only re-enroll clients whose domain entry is still active. If you uninstalled a client (using ipa-client-install --uninstall) or disabled its host entry (using ipa host-disable), you cannot re-enroll it.

You cannot re-enroll a client after you have renamed it. This is because in Identity Management, the key attribute of the client’s entry in LDAP is the client’s hostname, its FQDN. As opposed to re-enrolling a client, during which the client’s LDAP object remains unchanged, the outcome of renaming a client is that the client has its keys and other information in a different LDAP object with a new FQDN. Thus the only way to rename a client is to uninstall the host from IdM, change the host’s hostname, and install it as an IdM client with a new name. For details on how to rename a client, see Section 12.8, “Renaming Identity Management client systems”.

#### 12.7.1.1. What happens during client re-enrollment

During re-enrollment, Identity Management:
12.7.2. Re-enrolling a client by using user credentials: Interactive re-enrollment

This procedure describes re-enrolling an Identity Management client interactively by using the credentials of an authorized user.

1. Re-create the client machine with the same host name.
2. Run the `ipa-client-install --force-join` command on the client machine:
   
   ```
   # ipa-client-install --force-join
   ```
3. The script prompts for a user whose identity will be used to re-enroll the client. This could be, for example, a `hostadmin` user with the Enrollment Administrator role:
   
   ```
   User authorized to enroll computers: hostadmin
   Password for hostadmin@EXAMPLE.COM:
   ```

Additional resources

- For a more detailed procedure on enrolling clients by using an authorized user’s credentials, see Installing a client by using user credentials: Interactive installation in Installing Identity Management.

12.7.3. Re-enrolling a client by using the client keytab: Non-interactive re-enrollment

Prerequisites

- Back up the original client keytab file, for example in the `/tmp` or `/root` directory.

Procedure

This procedure describes re-enrolling an Identity Management (IdM) client non-interactively by using the keytab of the client system. For example, re-enrollment using the client keytab is appropriate for an automated installation.

1. Re-create the client machine with the same host name.
2. Copy the keytab file from the backup location to the `/etc/` directory on the re-created client machine.
3. Use the `ipa-client-install` utility to re-enroll the client, and specify the keytab location with the `-keytab` option:

   ```
   # ipa-client-install --keytab /etc/krb5.keytab
   ```
NOTE

The keytab specified in the --keytab option is only used when authenticating to initiate the enrollment. During the re-enrollment, IdM generates a new keytab for the client.

12.7.4. Testing an Identity Management client after installation

The Command-Line Interface informs you that the ipa-client-install was successful, but you can also do your own test.

To test that the Identity Management client can obtain information about users defined on the server, check that you are able to resolve a user defined on the server. For example, to check the default admin user:

```
[user@client1 ~]$ id admin
uid=1254400000(admin) gid=1254400000(admins) groups=1254400000(admins)
```

To test that authentication works correctly, su - as another IdM user:

```
[user@client1 ~]$ su - idm_user
Last login: Thu Oct 18 18:39:11 CEST 2018 from 192.168.122.1 on pts/0
[idm_user@client1 ~]$
```

12.8. RENAMING IDENTITY MANAGEMENT CLIENT SYSTEMS

The following sections describe how to change the host name of an Identity Management client system.

WARNING

Renaming a client is a manual procedure. Do not perform it unless changing the host name is absolutely required.

Renaming an Identity Management client involves:

1. Preparing the host. For details, see Section 12.8.1, “Prerequisites”
2. Uninstalling the IdM client from the host. For details, see Section 12.8.2, “Uninstalling an Identity Management client”
3. Renaming the host. For details, see Section 12.8.3, “Renaming the host system”
4. Installing the IdM client on the host with the new name. For details, see Section 12.8.4, “Re-installing an Identity Management client”
5. Configuring the host after the IdM client installation. For details, see Section 12.8.5, “Re-adding services, re-generating certificates, and re-adding host groups”

12.8.1. Prerequisites
Before uninstalling the current client, make note of certain settings for the client. You will apply this configuration after re-enrolling the machine with a new host name.

- Identify which services are running on the machine:
  - Use the `ipa service-find` command, and identify services with certificates in the output:
    ```
    $ ipa service-find old-client-name.example.com
    ```
  - In addition, each host has a default host service which does not appear in the `ipa service-find` output. The service principal for the host service, also called a host principal, is `host/old-client-name.example.com`.

- For all service principals displayed by `ipa service-find old-client-name.example.com`, determine the location of the corresponding keytabs on the `old-client-name.example.com` system:
  ```
  # find / -name "*.keytab"
  ```
  Each service on the client system has a Kerberos principal in the form `service_name/host_name@REALM`, such as `ldap/old-client-name.example.com@EXAMPLE.COM`.

- Identify all host groups to which the machine belongs.
  ```
  # ipa hostgroup-find old-client-name.example.com
  ```

12.8.2. Uninstalling an Identity Management client

Uninstalling a client removes the client from the Identity Management domain, along with all of the specific Identity Management configuration of system services, such as System Security Services Daemon (SSSD). This restores the previous configuration of the client system.

Procedure

1. Run the `ipa-client-install --uninstall` command:

   ```
   [root@client]# ipa-client-install --uninstall
   ```

2. Remove the DNS entries for the client host manually from the server:

   ```
   [root@server]# ipa dnsrecord-del
   Record name: old-client-client
   Zone name: idm.example.com
   No option to delete specific record provided.
   Delete all? Yes/No (default No): yes
   ------------------------
   Deleted record "old-client-name"
   ```

3. For each identified keytab other than `/etc/krb5.keytab`, remove the old principals:

   ```
   [root@client ~]# ipa-rmkeytab -k /path/to/keytab -r EXAMPLE.COM
   ```
4. On an IdM server, remove the host entry. This removes all services and revokes all certificates issued for that host:

```
[root@server ~]# ipa host-del client.example.com
```

### 12.8.3. Renaming the host system

Rename the machine as required. For example:

```
[root@client #] hostnamectl set-hostname new-client-name.example.com
```

You can now re-install the Identity Management client to the Identity Management domain with the new host name.

### 12.8.4. Re-installing an Identity Management client

Install an client on your renamed host following the procedure described in Installing an Identity Management client: Basic scenario in Installing Identity Management.

### 12.8.5. Re-adding services, re-generating certificates, and re-adding host groups

1. On the Identity Management (IdM) server, add a new keytab for every service identified in Section 12.8.1, “Prerequisites”.

```
[root@server ~]# ipa service-add service_name/new-client-name
```

2. Generate certificates for services that had a certificate assigned in Section 12.8.1, “Prerequisites”. You can do this:

   - Using the IdM administration tools
   - Using the certmonger utility

3. Re-add the client to the host groups identified in Section 12.8.1, “Prerequisites”.

### 12.9. DISABLING AND RE-ENABLING HOST ENTRIES

This section describes how to disable and re-enable hosts in Identity Management (IdM).

#### 12.9.1. Disabling Hosts

Complete this procedure to disable a host entry in IdM.

Domain services, hosts, and users can access an active host. There can be situations when it is necessary to remove an active host temporarily, for maintenance reasons, for example. Deleting the host in such situations is not desired as it removes the host entry and all the associated configuration permanently. Instead, choose the option of disabling the host.

Disabling a host prevents domain users from accessing it without permanently removing it from the domain. This can be done by using the `host-disable` command. Disabling a host kills the host’s current, active keytabs.

For example:
$ kinit admin
$ ipa host-disable client.example.com

As a result of disabling a host, the host becomes unavailable to all IdM users, hosts and services.

**IMPORTANT**

Disabling a host entry not only disables that host. It disables every configured service on that host as well.

### 12.9.2. Re-enabling Hosts

This section describes how to re-enable a disabled IdM host.

Disabling a host killed its active keytabs, which removed the host from the IdM domain without otherwise touching its configuration entry.

To re-enable a host, use the `ipa-getkeytab` command, adding:

- the `-s` option to specify which IdM server to request the keytab from
- the `-p` option to specify the principal name
- the `-k` option to specify the file to which to save the keytab.

For example, to request a new host keytab from `server.example.com` for `client.example.com`, and store the keytab in the `/etc/krb5.keytab` file:

```
$ ipa-getkeytab -s server.example.com -p host/client.example.com -k /etc/krb5.keytab -D "cn=directory manager" -w password
```

**NOTE**

You can also use the administrator’s credentials, specifying `-D "uid=admin,cn=users,cn=accounts,dc=example,dc=com"`. It is important that the credentials correspond to a user allowed to create the keytab for the host.

If the `ipa-getkeytab` command is run on an active IdM client or server, then it can be run without any LDAP credentials (`-D` and `-w`) if the user has a TGT obtained using, for example, `kinit admin`. To run the command directly on the disabled host, supply LDAP credentials to authenticate to the IdM server.
CHAPTER 13. ADDING HOST ENTRIES FROM IDM WEB UI

This chapter introduces hosts in Identity Management (IdM) and the operation of adding a host entry in the IdM Web UI.

13.1. HOSTS IN IDM

Identity Management (IdM) manages these identities:

- Users
- Services
- Hosts

A host represents a machine. As an IdM identity, a host has an entry in the IdM LDAP, that is the 389 Directory Server instance of the IdM server.

The host entry in IdM LDAP is used to establish relationships between other hosts and even services within the domain. These relationships are part of delegating authorization and control to hosts within the domain. Any host can be used in host-based access control (HBAC) rules.

IdM domain establishes a commonality between machines, with common identity information, common policies, and shared services. Any machine that belongs to a domain functions as a client of the domain, which means it uses the services that the domain provides. IdM domain provides three main services specifically for machines:

- DNS
- Kerberos
- Certificate management

Hosts in IdM are closely connected with the services running on them:

- Service entries are associated with a host.
- A host stores both the host and the service Kerberos principals.

13.2. HOST ENROLLMENT

This section describes enrolling hosts as IdM clients and what happens during and after the enrollment. The section compares the enrollment of IdM hosts and IdM users. The section also outlines alternative types of authentication available to hosts.

Enrolling a host consists of:

- Creating a host entry in IdM LDAP: possibly using the `ipa host-add` command in IdM CLI, or the equivalent IdM Web UI operation.

- Configuring IdM services on the host, for example the System Security Services Daemon (SSSD), Kerberos, and certmonger, and joining the host to the IdM domain.

The two actions can be performed separately or together.
If performed separately, they allow for dividing the two tasks between two users with different levels of privilege. This is useful for bulk deployments.

The `ipa-client-install` command can perform the two actions together. The command creates a host entry in IdM LDAP if that entry does not exist yet, and configures both the Kerberos and SSSD services for the host. The command brings the host within the IdM domain and allows it to identify the IdM server it will connect with. If the host belongs to a DNS zone managed by IdM, `ipa-client-install` adds DNS records for the host too. The command must be run on the client.

### 13.2.1. User privileges required for host enrollment

The host enrollment operation requires authentication to prevent an unprivileged user from adding unwanted machines to the IdM domain. The privileges required depend on several factors, for example:

- If a host entry is created separately from running `ipa-client-install`
- If a one-time password (OTP) is used for enrollment

**User privileges for optionally manually creating a host entry in IdM LDAP**

The user privilege required for creating a host entry in IdM LDAP using the `ipa host-add` CLI command or the IdM Web UI is Host Administrators. The Host Administrators privilege can be obtained through the IT Specialist role.

**User privileges for joining the client to the IdM domain**

Hosts are configured as IdM clients during the execution of the `ipa-client-install` command. The level of credentials required for executing the `ipa-client-install` command depends on which of the following enrolling scenarios you find yourself in:

- The host entry in IdM LDAP does not exist. For this scenario, you need a full administrator’s credentials or the Host Administrators role. A full administrator is a member of the admins group. The Host Administrators role provides privileges to add hosts and enroll hosts. For details about this scenario, see [Installing a client using user credentials: interactive installation](#).

- The host entry in IdM LDAP exists. For this scenario, you need a limited administrator’s credentials to execute `ipa-client-install` successfully. The limited administrator in this case has the Enrollment Administrator role, which provides the Host Enrollment privilege. For details, see [Installing a client using user credentials: interactive installation](#).

- The host entry in IdM LDAP exists, and an OTP has been generated for the host by a full or limited administrator. For this scenario, you can install an IdM client as an ordinary user if you run the `ipa-client-install` command with the `--password` option, supplying the correct OTP. For details, see [Installing a client by using a one-time password: Interactive installation](#).

After enrollment, IdM hosts authenticate every new session to be able to access IdM resources. Machine authentication is required for the IdM server to trust the machine and to accept IdM connections from the client software installed on that machine. After authenticating the client, the IdM server can respond to its requests.

### 13.2.2. Enrollment and authentication of IdM hosts and users: comparison

There are many similarities between users and hosts in IdM. This section describes some of the similarities that can be observed during the enrollment stage as well as those that concern authentication during the deployment stage.

- The enrollment stage (Table 13.1, “User and host enrollment”):
An administrator can create an LDAP entry for both a user and a host before the user or host actually join IdM: for the stage user, the command is `ipa stageuser-add`; for the host, the command is `ipa host-add`.

A file containing a key table or, abbreviated, keytab, a symmetric key resembling to some extent a user password, is created during the execution of the `ipa-client-install` command on the host, resulting in the host joining the IdM realm. Analogically, a user is asked to create a password when they activate their account, thus joining the IdM realm.

While the user password is the default authentication method for a user, the keytab is the default authentication method for a host. The keytab is stored in a file on the host.

Table 13.1. User and host enrollment

<table>
<thead>
<tr>
<th>Action</th>
<th>User</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-enrollment</td>
<td><code>$ ipa stageuser-add user_name</code> [-password]</td>
<td><code>$ ipa host-add host_name</code> [-random]</td>
</tr>
<tr>
<td>Activating the account</td>
<td><code>$ ipa stageuser-activate user_name</code></td>
<td><code>$ ipa-client install</code> [-password] (must be run on the host itself)</td>
</tr>
</tbody>
</table>

The deployment stage (Table 13.2, "User and host session authentication"):  

- When a user starts a new session, the user authenticates using a password; similarly, every time it is switched on, the host authenticates by presenting its keytab file. The System Security Services Daemon (SSSD) manages this process in the background.

- If the authentication is successful, the user or host obtains a Kerberos ticket granting ticket (TGT).

- The TGT is then used to obtain specific tickets for specific services.

Table 13.2. User and host session authentication

<table>
<thead>
<tr>
<th>Action</th>
<th>User</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default means of authentication</td>
<td>Password</td>
<td>Keytabs</td>
</tr>
<tr>
<td>Starting a session (ordinary user)</td>
<td><code>$ kinit user_name</code></td>
<td><code>[switch on the host]</code></td>
</tr>
<tr>
<td>The result of successful authentication</td>
<td>TGT to be used to obtain access to specific services</td>
<td>TGT to be used to obtain access to specific services</td>
</tr>
</tbody>
</table>

TGTs and other Kerberos tickets are generated as part of the Kerberos services and policies defined by the server. The initial granting of a Kerberos ticket, the renewing of the Kerberos credentials, and even the destroying of the Kerberos session are all handled automatically by the IdM services.

13.2.3. Alternative authentication options for IdM hosts
Apart from keytabs, IdM supports two other types of machine authentication:

- SSH keys. The SSH public key for the host is created and uploaded to the host entry. From there, the System Security Services Daemon (SSSD) uses IdM as an identity provider and can work in conjunction with OpenSSH and other services to reference the public keys located centrally in IdM.

- Machine certificates. In this case, the machine uses an SSL certificate that is issued by the IdM server’s certificate authority and then stored in IdM’s Directory Server. The certificate is then sent to the machine to present when it authenticates to the server. On the client, certificates are managed by a service called certmonger.

### 13.3. HOST ENTRY IN IDM LDAP

This section describes what a host entry in Identity Management (IdM) looks like and what attributes it can contain.

An LDAP host entry contains all relevant information about the client within IdM:

- Service entries associated with the host
- The host and service principal
- Access control rules
- Machine information, such as its physical location and operating system

**NOTE**

Note that the IdM Web UI Identity → Hosts tab does not show all the information about a particular host stored in the IdM LDAP.

### 13.3.1. Host entry configuration properties

A host entry can contain information about the host that is outside its system configuration, such as its physical location, MAC address, keys, and certificates.

This information can be set when the host entry is created if it is created manually. Alternatively, most of this information can be added to the host entry after the host is enrolled in the domain.

**Table 13.3. Host Configuration Properties**

<table>
<thead>
<tr>
<th>UI Field</th>
<th>Command-Line Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>--desc=description</td>
<td>A description of the host.</td>
</tr>
<tr>
<td>Locality</td>
<td>--locality=locality</td>
<td>The geographic location of the host.</td>
</tr>
<tr>
<td>Location</td>
<td>--location=location</td>
<td>The physical location of the host, such as its data center rack.</td>
</tr>
<tr>
<td>UI Field</td>
<td>Command-Line Option</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Platform</td>
<td>--platform=string</td>
<td>The host hardware or architecture.</td>
</tr>
<tr>
<td>Operating system</td>
<td>--os=string</td>
<td>The operating system and version for the host.</td>
</tr>
<tr>
<td>MAC address</td>
<td>--macaddress=address</td>
<td>The MAC address for the host. This is a multi-valued attribute. The MAC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>address is used by the NIS plug-in to create a NIS ethers map for the host.</td>
</tr>
<tr>
<td>SSH public keys</td>
<td>--sshpubkey=string</td>
<td>The full SSH public key for the host. This is a multi-valued attribute, so</td>
</tr>
<tr>
<td></td>
<td></td>
<td>multiple keys can be set.</td>
</tr>
<tr>
<td>Principal name (not editable)</td>
<td>--principalname=principal</td>
<td>The Kerberos principal name for the host. This defaults to the host name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>during the client installation, unless a different principal is explicitly</td>
</tr>
<tr>
<td></td>
<td></td>
<td>set in the -p. This can be changed using the command-line tools, but cannot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>be changed in the UI.</td>
</tr>
<tr>
<td>Set One-Time Password</td>
<td>--password=string</td>
<td>This option sets a password for the host which can be used in bulk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>enrollment.</td>
</tr>
<tr>
<td></td>
<td>--random</td>
<td>This option generates a random password to be used in bulk enrollment.</td>
</tr>
<tr>
<td></td>
<td>--certificate=string</td>
<td>A certificate blob for the host.</td>
</tr>
<tr>
<td></td>
<td>--updatedns</td>
<td>This sets whether the host can dynamically update its DNS entries if its IP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>address changes.</td>
</tr>
</tbody>
</table>

13.4. ADDING HOST ENTRIES FROM THE WEB UI

1. Open the **Identity** tab, and select the **Hosts** subtab.
2. Click **Add** at the top of the hosts list.
3. Enter the machine name and select the domain from the configured zones in the drop-down list. If the host has already been assigned a static IP address, then include that with the host entry so that the DNS entry is fully created. The **Class** field has no specific purpose at the moment.

![Figure 13.1. Adding Host Entries](image)

DNS zones can be created in IdM. If the IdM server does not manage the DNS server, the zone can be entered manually in the menu area, like a regular text field.

**NOTE**

Select the **Force** check box if you want to skip checking whether the host is resolvable via DNS.

4. Click the **Add and Edit** button to go directly to the expanded entry page and enter more attribute information. Information about the host hardware and physical location can be included with the host entry.
Host: server.zone.example.com

Host name: server.zone.example.com
Principal name: host/server.zone.example.com@EXAMPLE.COM
Description: 
Class: 
Locality: 

server.zone.example.com is a member of:
CHAPTER 14. ADJUSTING ID RANGES MANUALLY

An IdM master generates unique user ID (UID) and group ID (GID) numbers. By creating and assigning different ID ranges to replicas, it also ensures that they never generate the same ID numbers. By default, this process is automatic. However, you can manually adjust the IdM ID range during the IdM master installation, or manually define a replica’s DNA ID range.

14.1. ID RANGES

ID numbers are divided into ID ranges. Keeping separate numeric ranges for individual servers and replicas eliminates the chance that an ID number issued for an entry is already used by another entry on another server or replica.

Note that there are two distinct types of ID ranges:

- The IdM ID range, which is assigned during the IdM master installation. This range cannot be modified after it is created. However, if you need to, you can create a new IdM ID range in addition to the original one. For more information, see Automatic ID ranges assignment and Adding a new IdM ID range.

- The Distributed Numeric Assignment (DNA) ID ranges, which can be modified by the user. These have to fit within an existing IdM ID range. For more information, see Adjusting DNA ID ranges manually.

Replicas can also have a next DNA ID range assigned. A replica uses its next range when it runs out of IDs in its current range. Next ranges are assigned automatically when a replica is deleted or you can set them manually.

The ranges are updated and shared between the master and replicas by the DNA plug-in, as part of the back end 389 Directory Server instance for the domain.

The DNA range definition is set by two attributes: the server’s next available number (the low end of the DNA range) and its maximum value (the top end of the DNA range). The initial bottom range is set during the plug-in instance configuration. After that, the plug-in updates the bottom value. Breaking the available numbers into ranges allows the servers to continually assign numbers without overlapping with each other.

14.2. AUTOMATIC ID RANGES ASSIGNMENT

By default, an IdM ID range is automatically assigned during the IdM master installation. The ipa-server-install command randomly selects and assigns a range of 200,000 IDs from a total of 10,000 possible ranges. Selecting a random range in this way significantly reduces the probability of conflicting IDs in case you decide to merge two separate IdM domains in the future.

NOTE

This IdM ID range cannot be modified after it is created. You can only manually adjust the Distributed Numeric Assignment (DNA) ID ranges, using the commands described in Adjusting DNA ID ranges manually. A DNA range matching the IdM ID range is automatically created during installation.

If you have a single IdM server installed, it controls the whole DNA ID range. When you install a new replica and the replica requests its own DNA ID range, the initial ID range for the master splits and is distributed between the master and replica: the replica receives half of the remaining DNA ID range that is available on the initial master. The master and replica then use their respective portions of the original
ID range for new user or group entries. Also, if the replica is close to depleting its allocated ID range and fewer than 100 IDs remain, the replica contacts the other available servers to request a new DNA ID range.

**IMPORTANT**

When you install a replica, it *does not* immediately receive an ID range. A replica receives an ID range the first time the DNA plug-in is used, for example when you first add a user. Until then, the replica has no ID range defined.

If the initial master stops functioning before the replica requests a DNA ID range from it, the replica is unable to contact the master to request the ID range. Attempting to add a new user on the replica then fails. In such situations, you can find out what ID range is assigned to the disabled master, and assign an ID range to the replica manually.

### 14.3. ASSIGNING THE IDM ID RANGE MANUALLY DURING SERVER INSTALLATION

You can override the default behavior and set an IdM ID range manually instead of having it assigned randomly.

**IMPORTANT**

Do not set ID ranges that include UID values of 1000 and lower; these values are reserved for system use. Also, do not set an ID range that would include the 0 value; the SSSD service does not handle the 0 ID value.

**Procedure**

- You can define the IdM ID range manually during server installation by using the following two options with `ipa-server-install`:
  - `--idstart` gives the starting value for UID and GID numbers.
  - `--idmax` gives the maximum UID and GID number; by default, the value is the `--idstart` starting value plus 199,999.

**Verification steps**

- To check if the ID range was assigned correctly, you can display the assigned IdM ID range by using the `ipa idrange-find` command:

  ```
  # ipa idrange-find
  --------------
  1 range matched
  --------------
  Range name: IDM.EXAMPLE.COM_id_range
  First Posix ID of the range: 882200000
  Number of IDs in the range: 200000
  Range type: local domain range
  --------------
  Number of entries returned 1
  --------------
  ```
14.4. ADDING A NEW IDM ID RANGE

In some cases, you may want to create a new IdM ID range in addition to the original one; for example, when a replica has run out of IDs and the original IdM ID range is depleted.

**IMPORTANT**

Adding a new IdM ID range does not create new DNA ID ranges automatically. You need to assign new DNA ID ranges manually as needed. For more information on how to do this, see Adjusting DNA ID ranges manually.

**Procedure**

- To create a new IdM ID range, use the `ipa idrange-add` command. You need to specify the new range name, the first ID number of the range and the range size:

  ```
  # ipa idrange-add IDM.EXAMPLE.COM_new_range --base-id=1000000 --range-size=200000
  ------------------------------------------
  Added ID range "IDM.EXAMPLE.COM_new_range"
  ------------------------------------------
  Range name: IDM.EXAMPLE.COM_new_range
  First Posix ID of the range: 1000000
  Number of IDs in the range: 200000
  Range type: local domain range
  ```

**Verification steps**

- You can check if the new range is set correctly by using the `ipa idrange-find` command:

  ```
  # ipa idrange-find
  ----------------
  2 ranges matched
  ----------------
  Range name: IDM.EXAMPLE.COM_id_range
  First Posix ID of the range: 882200000
  Number of IDs in the range: 200000
  Range type: local domain range
  
  Range name: IDM.EXAMPLE.COM_new_range
  First Posix ID of the range: 1000000
  Number of IDs in the range: 200000
  Range type: local domain range
  
  ----------------
  Number of entries returned 2
  ----------------
  ```

14.5. DISPLAYING CURRENTLY ASSIGNED DNA ID RANGES

You can display both the currently active Distributed Numeric Assignment (DNA) ID range on a server, as well as its next DNA range if it has one assigned.

**Procedure**
To display which DNA ID ranges are configured for the servers in the topology, use the following commands:

- `ipa-replica-manage dnarange-show` displays the current DNA ID range that is set on all servers or, if you specify a server, only on the specified server, for example:

```bash
# ipa-replica-manage dnarange-show
masterA.example.com: 1001-1500
masterB.example.com: 1501-2000
masterC.example.com: No range set

# ipa-replica-manage dnarange-show masterA.example.com
masterA.example.com: 1001-1500
```

- `ipa-replica-manage dnanextrange-show` displays the next DNA ID range currently set on all servers or, if you specify a server, only on the specified server, for example:

```bash
# ipa-replica-manage dnanextrange-show
masterA.example.com: 2001-2500
masterB.example.com: No on-deck range set
masterC.example.com: No on-deck range set

# ipa-replica-manage dnanextrange-show masterA.example.com
masterA.example.com: 2001-2500
```

### 14.6. AUTOMATIC DNA ID RANGE EXTENSION

When you delete a functioning replica, the `ipa-replica-manage del` command retrieves the DNA ID ranges that were assigned to the replica and adds them as a next range to another available IdM replica. This ensures that DNA ID ranges are used efficiently.

After you delete a replica, you can verify which DNA ID ranges are configured for other servers by using the commands described in Displaying currently assigned DNA ID ranges.

### 14.7. MANUAL DNA ID RANGE ADJUSTMENT

In certain situations, it is necessary to manually adjust a Distributed Numeric Assignment (DNA) ID range, for example when:

- A replica has run out of IDs and the IdM ID range is depleted
  A replica has exhausted the DNA ID range that was assigned to it, and requesting additional IDs failed because no more free IDs are available in the IdM range.

To solve this situation, extend the DNA ID range assigned to the replica. You can do this in two ways:

- Shorten the DNA ID range assigned to a different replica, then assign the newly available values to the depleted replica.
- Create a new IdM ID range, then set a new DNA ID range for the replica within this created IdM range.
  For information on how to create a new IdM ID range, see Adding a new IdM ID range.

- A replica stopped functioning
A replica’s DNA ID range is not automatically retrieved when the replica dies and needs to be deleted, which means the DNA ID range previously assigned to the replica becomes unavailable. You want to recover the DNA ID range and make it available for other replicas.

If you want to recover a DNA ID range belonging to a replica that stopped functioning and assign it to another server, you first need to find out what the ID range values are, before manually assigning that range to a different server. Also, to avoid duplicate UIDs or GIDs, make sure that no ID value from the recovered range was previously assigned to a user or group; you can do this by examining the UIDs and GIDs of existing users and groups.

You can manually adjust a DNA ID range for a replica using the commands in Adjusting DNA ID ranges manually.

**NOTE**

If you assign a new DNA ID range, the UIDs of the already existing entries on the server or replica stay the same. This does not pose a problem because even if you change the current DNA ID range, IdM keeps a record of what ranges were assigned in the past.

### 14.8. ADJUSTING DNA ID RANGES MANUALLY

In some cases, you may need to manually adjust Distributed Numeric Assignment (DNA) ID ranges for existing replicas, for example to reassign a DNA ID range assigned to a non-functioning replica. For more information, see Manual DNA ID range adjustment.

When adjusting a DNA ID range manually, make sure that the newly adjusted range is included in the IdM ID range; you can check this using the `ipa idrange-find` command. Otherwise, the command will fail.

**IMPORTANT**

Be careful not to create overlapping ID ranges. If any of the ID ranges you assign to servers or replicas overlap, it could result in two different servers assigning the same ID value to different entries.

**Prerequisites**

- **Optional.** If you are recovering a DNA ID range from a non-functioning replica, first find the ID range using the commands described in Displaying currently assigned DNA ID ranges.

**Procedure**

- To define the current DNA ID range for a specified server, use the `ipa-replica-manage dnarange-set`:

  ```
  # ipa-replica-manage dnarange-set masterA.example.com 1250-1499
  ```

- To define the next DNA ID range for a specified server, use the `ipa-replica-manage dnanextrange-set`:

  ```
  # ipa-replica-manage dnanextrange-set masterB.example.com 1500-5000
  ```

**Verification steps**

You can check that the new DNA ranges are set correctly by using the commands described in ID Ranges.
You can check that the new DNA ranges are set correctly by using the commands described in "Displaying the currently assigned DNA ID ranges."
CHAPTER 15. GRANTING SUDO ACCESS TO AN IDM USER ON AN IDM CLIENT

15.1. SUDO ACCESS ON AN IDM CLIENT

System administrators can grant `sudo` access to allow non-root users to execute administrative commands that are normally reserved for the `root` user. Consequently, when users need to perform an administrative command normally reserved for the `root` user, they precede that command with `sudo`. After entering their password, the command is executed as if they were the `root` user.

If a Red Hat Enterprise Linux (RHEL) 8 host is enrolled as an Identity Management (IdM) client, you can specify `sudo` rules defining which IdM users can perform which commands on the host in the following ways:

- Locally in the `/etc/sudoers` file
- Centrally in IdM

This chapter describes creating a central `sudo` rule for an IdM client using IdM Web UI. For details on creating local sudo rules on a RHEL 8 host, see Managing sudo access.

Note that you can also define central IdM `sudo` rules using the IdM command-line interface.

15.2. GRANTING SUDO ACCESS TO AN IDM USER ON AN IDM CLIENT USING IDM WEB UI

In Identity Management (IdM), you can grant `sudo` access for a specific command to an IdM user account on a specific IdM host. First, add a `sudo` command and then create a `sudo` rule for one or more commands.

Complete this procedure to create the `idm_user_reboot` sudo rule to grant `idm_user` the permission to run the `/usr/sbin/reboot` command on the `idmclient` machine.

Prerequisites

- You are logged in as IdM administrator.
- You have created a user account for `idm_user` in IdM and unlocked the account by creating a password for the user. For details on adding a new IdM user using the command-line interface, see Adding users using the command line.
- No local `idm_user` account has been created on `idmclient`. The `idm_user` user is not listed in the local `/etc/passwd` file.

Procedure

1. Add the `/usr/sbin/reboot` command to the IdM database of `sudo` commands:
   a. Navigate to Policy → Sudo → Sudo Commands.
   b. Click Add in the upper right corner to open the Add sudo command dialog box.
   c. Enter the command you want the user to be able to perform using `sudo: /usr/sbin/reboot`.  

   Figure 15.1. Adding IdM sudo command
Figure 15.1. Adding IdM sudo command

![Add sudo command dialog box]

- Click Add.

2. Use the new sudo command entry to create a sudo rule to allow idm_user to reboot the idmclient machine:
   b. Click Add in the upper right corner to open the Add sudo rule dialog box.
   c. Enter the name of the sudo rule: idm_user_reboot.
   d. Click Add and Edit.
   e. Specify the user:
      i. In the Who section, check the Specified Users and Groups radio button.
      ii. In the User category the rule applies to subsection, click Add to open the Add users into sudo rule "idm_user_reboot" dialog box.
      iii. In the Add users into sudo rule "idm_user_reboot" dialog box in the Available column, check the idm_user checkbox, and move it to the Prospective column.
      iv. Click Add.
   f. Specify the host:
      i. In the Access this host section, check the Specified Hosts and Groups radio button.
      ii. In the Host category this rule applies to subsection, click Add to open the Add hosts into sudo rule "idm_user_reboot" dialog box.
      iii. In the Add hosts into sudo rule "idm_user_reboot" dialog box in the Available column, check the idmclient.idm.example.com checkbox, and move it to the Prospective column.
iv. Click **Add**.

a. Specify the commands:

i. In the **Command category the rule applies to** subsection of the **Run Commands** section, check the **Specified Commands and Groups** radio button.

ii. In the **Sudo Allow Commands** subsection, click **Add** to open the **Add allow sudo commands into sudo rule “idm_user_reboot”** dialog box.

iii. In the **Add allow sudo commands into sudo rule “idm_user_reboot”** dialog box in the **Available** column, check the `/usr/sbin/reboot` checkbox, and move it to the **Prospective** column.

iv. Click **Add** to return to the **idm_sudo_reboot** page.

Figure 15.2. Adding IdM sudo rule

<table>
<thead>
<tr>
<th>Who</th>
</tr>
</thead>
<tbody>
<tr>
<td>User category the rule applies to</td>
</tr>
<tr>
<td>Users</td>
</tr>
<tr>
<td>User Groups</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Access the host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
</tr>
<tr>
<td>idmclient.example.com</td>
</tr>
<tr>
<td>Other Groups</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Run Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command category the rule applies to</td>
</tr>
<tr>
<td>Allow</td>
</tr>
<tr>
<td>Sudo Allow Commands</td>
</tr>
<tr>
<td>/sbin/reboot</td>
</tr>
<tr>
<td>sudo Allow Command Groups</td>
</tr>
</tbody>
</table>


g. Click **Save** in the top left corner.

The new rule is enabled by default.

**Verification steps**

Test that the sudo rule that you have set up on the IdM server works on **idmclient** by verifying that **idm_user** can now reboot **idmclient** using **sudo**. Note that propagating the changes from the server to the client can take a few minutes.

1. Log in to **idmclient** as **idm_user**.

2. Reboot the machine using **sudo**. Enter the password for **idm_user** when prompted:

   ```bash
   $ sudo /usr/sbin/reboot
   [sudo] password for idm_user:
   ```

   If sudo is configured correctly, the machine reboots.
CHAPTER 16. 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.

16.1. 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

16.2. 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>
</table>

Table 16.1. Comparison of certificates and Kerberos
### Authentication

<table>
<thead>
<tr>
<th>Type</th>
<th>Yes</th>
<th>Yes</th>
</tr>
</thead>
</table>

### Privacy

<table>
<thead>
<tr>
<th>Type</th>
<th>Optional</th>
<th>Yes</th>
</tr>
</thead>
</table>

### Integrity

<table>
<thead>
<tr>
<th>Type</th>
<th>Optional</th>
<th>Yes</th>
</tr>
</thead>
</table>

### Type of cryptography involved

<table>
<thead>
<tr>
<th>Type</th>
<th>Symmetrical</th>
<th>Asymmetrical</th>
</tr>
</thead>
</table>

### Default validity

<table>
<thead>
<tr>
<th>Type</th>
<th>Short (1 day)</th>
<th>Long (2 years)</th>
</tr>
</thead>
</table>

By default, Kerberos in Identity Management only ensures the identity of the communicating parties.

### 16.3. 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 17. 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 17.2, “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 17.3.1, “Exporting a certificate and private key from an NSS database into a PKCS #12 file”.

17.1. 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>
<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>

**Table 17.1. Certificate encodings**

Section 17.4, “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
```

### 17.2. 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.

17.2.1. 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:

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

17.2.2. 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
     
     $ openssl pkcs12 -in cert_and_key.p12 -clcerts -nokeys -out cert.pem
     Enter Import Password:
     ```
   - 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
```

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.

**17.3. 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 17.3.1, "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 17.3.2, "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 22.4, “Configuring a browser to enable certificate authentication” and Section 22.5, “Authenticating to the Identity Management Web UI with a Certificate as an Identity Management User”.

### 17.3.1. 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:

   ```bash
   $ 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:

   ```bash
   # 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.
   ```

### 17.3.2. 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:

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

### 17.4. CERTIFICATE-RELATED COMMANDS AND FORMATS IN IDM
Table **IdM certificate commands and formats** displays certificate-related commands in IdM with acceptable 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 18. MANAGING THE VALIDITY OF CERTIFICATES IN IDM

In Identity Management (IdM), you can manage the validity of both already existing certificates and certificates you want to issue in the future, but the methods are different.

Managing the validity of an existing certificate that was issued by IdM CA
In IdM, the following methods of viewing the expiry date of a certificate are available:

- Viewing the expiry date in IdM WebUI;
- Viewing the expiry date in the CLI.

You can manage the validity of an already existing certificate that was issued by IdM CA in the following ways:

- Renew a certificate by requesting a new certificate using either the original certificate signing request (CSR) or a new CSR generated from the private key. You can request a new certificate using the following utilities:
  - `certmonger` You can use `certmonger` to request a service certificate. Before the certificate is due to expire, `certmonger` will automatically renew the certificate, thereby ensuring a continuing validity of the service certificate. For details, see Obtaining an IdM certificate for a service using certmonger;
  - `certutil` You can use `certutil` to renew user, host, and service certificates. For details on requesting a user certificate, see Requesting a new user certificate and exporting it to the client;
  - `openssl` You can use `openssl` to renew user, host, and service certificates.

- Revoke a certificate. For details, see:
  - Revoking certificates with the integrated IdM CAs using IdM WebUI;
  - Revoking certificates with the integrated IdM CAs using IdM CLI;

- Restore a certificate if it has been temporarily revoked. For details, see:
  - Restoring certificates with the integrated IdM CAs using IdM WebUI;
  - Restoring certificates with the integrated IdM CAs using IdM CLI.

Managing the validity of future certificates issued by IdM CA
To manage the validity of future certificates issued by IdM CA, modify, import, or create a certificate profile. For details, see Certificate profiles in RHEL 7 documentation.

18.1. VIEWING THE EXPIRY DATE OF A CERTIFICATE

18.1.1. Viewing the expiry date of a certificate in IdM WebUI
You can use IdM WebUI to view the expiry date of all the certificates that have been issued by IdM CA.
Prerequisites

- Ensure that you have obtained the administrator’s credentials.

Procedure

1. In the Authentication menu, click Certificates > Certificates.

2. Click the serial number of the certificate to open the certificate information page.

Figure 18.1. List of Certificates

```
Certificates

<table>
<thead>
<tr>
<th>Serial Number</th>
<th>Subject</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CN=Certificate Authority,O=EXAMPLE.COM</td>
</tr>
<tr>
<td>2</td>
<td>CN=OCSP Subsystem,O=EXAMPLE.COM</td>
</tr>
<tr>
<td>3</td>
<td>CN=server.example.com,O=EXAMPLE.COM</td>
</tr>
<tr>
<td>4</td>
<td>CN=CA Subsystem,O=EXAMPLE.COM</td>
</tr>
</tbody>
</table>
```

3. In the certificate information page, locate the Expires On information.

18.1.2. Viewing the expiry date of a certificate in the CLI

You can use the command-line interface (CLI) to view the expiry date of a certificate.

Procedure

- Use the `openssl` utility to open the file in a human-readable format:

```
$ openssl x509 -noout -text -in ca.pem
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number: 1 (0x1)
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: O = IDM.EXAMPLE.COM, CN = Certificate Authority
    Validity
      Not After : Oct 30 19:39:14 2037 GMT
```

18.2. REVOKING CERTIFICATES WITH THE INTEGRATED IDM CAS

18.2.1. Certificate revocation reasons

A revoked certificate is invalid and cannot be used for authentication. All revocations are permanent, except for reason 6: Certificate Hold.

The default revocation reason is 0: unspecified.
Table 18.1. Revocation Reasons

<table>
<thead>
<tr>
<th>ID</th>
<th>Reason</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Key Compromised</td>
<td>The key that issued the certificate is no longer trusted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Possible causes: lost token, improperly accessed file.</td>
</tr>
<tr>
<td>2</td>
<td>CA Compromised</td>
<td>The CA that issued the certificate is no longer trusted.</td>
</tr>
<tr>
<td>3</td>
<td>Affiliation Changed</td>
<td>Possible causes:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* A person has left the company or moved to another department.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* A host or service is being retired.</td>
</tr>
<tr>
<td>4</td>
<td>Superseded</td>
<td>A newer certificate has replaced the current certificate.</td>
</tr>
<tr>
<td>5</td>
<td>Cessation of Operation</td>
<td>The host or service is being decommissioned.</td>
</tr>
<tr>
<td>6</td>
<td>Certificate Hold</td>
<td>The certificate is temporarily revoked. You can restore the certificate later.</td>
</tr>
<tr>
<td>8</td>
<td>Remove from CRL</td>
<td>The certificate is not included in the certificate revocation list (CRL).</td>
</tr>
<tr>
<td>9</td>
<td>Privilege Withdrawn</td>
<td>The user, host, or service is no longer permitted to use the certificate.</td>
</tr>
<tr>
<td>10</td>
<td>Attribute Authority (AA)</td>
<td>The AA certificate is no longer trusted.</td>
</tr>
<tr>
<td></td>
<td>Compromise</td>
<td></td>
</tr>
</tbody>
</table>

18.2.2. Revoking certificates with the integrated IdM CAs using IdM WebUI

If you know you have lost the private key for your certificate, you must revoke the certificate to prevent its abuse. Complete this procedure to use the IdM WebUI to revoke a certificate issued by the IdM CA.

Procedure

1. Click **Authentication > Certificates > Certificates**.

2. Click the serial number of the certificate to open the certificate information page.
3. In the certificate information page, click Actions → Revoke Certificate.

4. Select the reason for revoking and click Revoke. See Section 18.2.1, “Certificate revocation reasons” for details.

18.2.3. Revoking certificates with the integrated IdM CAs using IdM CLI

If you know you have lost the private key for your certificate, you must revoke the certificate to prevent its abuse. Complete this procedure to use the IdM CLI to revoke a certificate issued by the IdM CA.

Procedure

- Use the `ipa cert-revoke` command, and specify:
  - the certificate serial number
  - the ID number for the revocation reason; see Section 18.2.1, “Certificate revocation reasons” for details

For example, to revoke the certificate with serial number 1032 because of reason 1: Key Compromised, enter:

```
$ ipa cert-revoke 1032 --revocation-reason=1
```

For details on requesting a new certificate, see the following documentation:

- Requesting a new user certificate and exporting it to the client
- Obtaining an IdM certificate for a service using certmonger

18.3. RESTORING CERTIFICATES WITH THE INTEGRATED IDM CAS

If you have revoked a certificate because of reason 6: Certificate Hold, you can restore it again if the private key for the certificate has not been compromised. To restore a certificate, use one of the following procedures:

- Restore certificates with the integrated IdM CAs using IdM WebUI
- Restore certificates with the integrated IdM CAs using IdM CLI
18.3.1. Restoring certificates with the integrated IdM CAs using IdM WebUI

Complete this procedure to use the IdM WebUI to restore an IdM certificate that has been revoked because of Reason 6: Certificate Hold.

Procedure

1. In the Authentication menu, click Certificates > Certificates.

2. Click the serial number of the certificate to open the certificate information page.

3. In the certificate information page, click Actions → Restore Certificate.

18.3.2. Restoring certificates with the integrated IdM CAs using IdM CLI

Complete this procedure to use the IdM CLI to restore an IdM certificate that has been revoked because of Reason 6: Certificate Hold.

Procedure

- Use the `ipa cert-remove-hold` command and specify the certificate serial number. For example:

  ```bash
  $ ipa cert-remove-hold 1032
  ```
CHAPTER 19. 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 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 19.1, “Configuring the IdM server for smart card authentication”
Section 19.2, “Configuring the IdM client for smart card authentication”
Section 19.3, “Adding a certificate to a user entry in IdM”
Section 19.4, “Installing gnutls-utils”
Section 19.5, “Storing a certificate on the smart card”
Section 19.6, “Configuring the browser for smart card authentication”
Section 19.7, “Logging in to IdM with smart cards”

19.1. 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 22.4, “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-advisе 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:

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

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
   ```

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](https://httpd.apache.org/docs/current/mod/mod_ssl.html).

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.

### 19.2. 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
  For details see [Configuring SSH access using smart card authentication](https).
- 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 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 19.1, “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
The client is now configured for smart card authentication.

19.3. 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 21, 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.

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

**Prerequisites**

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

**Procedure**

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 in the **PEM** format using the **cat** utility or a text editor:

   ```
   [user@client SmartCard]$ cat testuser.crt
   ```

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

6. Click **Add**.
The `sc_user` entry now contains an external certificate.

### 19.3.2. Adding a certificate to a user entry in the IdM CLI

**Prerequisites**

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

**Procedure**

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

   ```bash
   [user@client SmartCard]$ kinit admin
   ```

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

   ```bash
   [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:

   ```bash
   [user@client SmartCard]$ export CERT=`openssl x509 -outform der -in testuser.crt | base64 -w0`
   ```
Note that certificate in the \texttt{testuser.crt} file must be in the \texttt{PEM} format.

3. Add the certificate to the profile of \texttt{sc_user} using the \texttt{ipa user-add-cert} command:

\begin{verbatim}
[user@client SmartCard]$ ipa user-add-cert sc_user --certificate=$CERT
\end{verbatim}

The \texttt{sc_user} entry now contains an external certificate.

\section*{19.4. INSTALLING GNUTLS-UTILS}

To configure your smart card, you need a tool that can generate certificates and store them in the smart card.

You must:

\begin{itemize}
  \item Install the \texttt{gnutls-utils} program which helps you to manage certificates.
  \item Start the \texttt{pcscd} service which communicates with the smart card reader.
\end{itemize}

\subsection*{Procedure}

1. Install the \texttt{gnutls-utils} package which allows you to manage smart card settings for reading and writing from the smart card:

\begin{verbatim}
# dnf -y install opensc gnutls-utils
\end{verbatim}

2. Start the \texttt{pcscd} service.

\begin{verbatim}
# systemctl start pcscd
\end{verbatim}

Verify that the \texttt{pcscd} service is up and running.

\section*{19.5. STORING A CERTIFICATE ON THE SMART CARD}

This section describes smart card configuration with \texttt{gnutls-utils} which helps you to configure:

\begin{itemize}
  \item Erasing your smart card
  \item Setting a new PIN and PUK
  \item Creating a new slot in the smart card
  \item Storing the certificate and private key in the slot
  \item Locking the smart card settings (some smart cards need this type of finalization)
\end{itemize}

\subsection*{Prerequisites}

\begin{itemize}
  \item The \texttt{gnutls-utils} package is installed. For details, see \texttt{Installing gnutls-utils}.
  \item The card is inserted in the reader and connected to the computer.
\end{itemize}
Procedure

1. Erase your smart card and authenticate yourself with PIN:

   ```bash
   # spawn pkcs15-init --erase-card --use-default-transport-keys
   Using reader with a card: Smart Card name
   PIN [Security Officer PIN] required.
   Please enter PIN [Security Officer PIN]:
   
   The card has been erased.
   
2. Set your PIN and PUK – both twice for verification:

   ```bash
   # pkcs15-init --create-pkcs15 --use-default-transport-keys \
   --pin redhat --puk redhat --so-pin redhat --so-puk redhat
   Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface]
   (53311657131456) 00 00
   
   The `pkcs15-init` creates a new slot in the smart card and marks it with a label and an authentication ID.

3. Add a label that is the name of the slot and the authentication ID:

   ```bash
   # pkcs15-init --store-pin --label testuser \ 
   --auth-id 01 --so-pin redhat --pin redhat --puk redhat
   Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface]
   (53311657131456) 00 00
   
   The label auth-id is set to 01.

4. Store the private key in the new slot of the smart card:

   ```bash
   # pkcs15-init --store-private-key testuser.key --label testuser_key \ 
   --auth-id 01 --id 01 --so-pin redhat --pin redhat
   Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface]
   (53311657131456) 00 00
   
5. Store the certificate in the new slot of the smart card:

   ```bash
   # pkcs15-init --store-certificate testuser.crt --label testuser_crt \ 
   --auth-id 01 --id 01 --format pem --so-pin redhat --pin redhat
   Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface]
   (53311657131456) 00 00
   
6. Optionally finalize the card settings with locking the settings:

   ```bash
   # pkcs15-init -F
   
   IMPORTANT
   Some smart cards require this step.

At this stage, your smart card includes the certificate and private key in the newly created slot. PIN and PUK has been successfully created.
19.6. 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 19.1, “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, see Section 19.5, “Storing a certificate on the smart card”.
- 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 19.3, “Adding a certificate to a user entry in IdM”.

Procedure

1. Open Firefox, click on Preferences.
2. Navigate to **Privacy & Security**.

3. Click on **Security Devices**.

**Figure 19.2. Firefox preferences**

**Figure 19.3. Security devices**
4. In the new **Device Manager** dialogue window, click on **Load**.

**Figure 19.4. 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.

**Figure 19.5. Entering the security device information**

6. Optionally, check that module can log in to Firefox.
   a. Click on **PIV Card Holder pin (PIV_I)** in the left pane and click **Log In** in the right pane.

   **Figure 19.6. Logging in with the security device**

   b. Enter the PIN of the smart card and click **OK**.
19.7. 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.

7. Click **OK**.

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

1. Open the IdM Web UI in the browser.

2. Click on Log In Using Certificate

   ![Image of the RED HAT® IDENTITY MANAGEMENT interface with Log In Using Certificate highlighted]

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 20. CONFIGURING CERTIFICATES ISSUED BY ADCS FOR SMART CARD AUTHENTICATION IN IDM

This scenario describes the following situation:

- Your deployment is based on cross-forest trust between Identity Management (IdM) and Active Directory (AD).
- You want to allow smart card authentication for users whose accounts are stored in AD.
- Certificates are created and stored in Active Directory Certificate Services (ADCS).

Configuration will be accomplished in the following steps:

- Copying CA and user certificates from Active Directory to the IdM server and client
- Configuring the IdM server and clients for smart card authentication using ADCS certificates
- Converting a PFX (PKCS#12) file to be able to store the certificate and private key into the smart card
- Configuring timeouts in the sssd.conf file
- Creating certificate mapping rules for smart card authentication

Prerequisites

- Identity Management (IdM) and Active Directory (AD) trust is installed
  For details, see Installing trust between IdM and AD.
- Active Directory Certificate Services (ADCS) is installed and certificates for users are generated

20.1. SMART CARD AUTHENTICATION

A smart card is a physical device which can provide personal authentication using certificates stored on the card. Personal authentication means that you can use smart cards in the same way as user 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 password.

You can configure how you want smart card authentication to work in a particular IdM client:

- Users can authenticate with the user name and password or with their smart cards
- Users can authenticate with their smart cards, and passwords are not allowed
- Users can use the smart card for logout with a function lock on removal, and passwords are not allowed

If the default smart card setup done by the `ipa-advise` script does not work, you can find further details about smart card configuration with `authselect`, see Configuring smart cards using authselect.

Identity Management (IdM) supports smart card authentication with:
• User certificates issued by the IdM certificate authority. For details, see Configuring Identity Management for smart card authentication.

• User certificates issued by the ADCS certificate authority.

• User certificates issued by local certification authority generated on a RHEL system. For details, see Configuring and importing local certificates to a smart card.

• User certificates issued by an external certificate authority.

**NOTE**

If you want to start to use smart card authentication, see the hardware requirements: Smart Card support in RHEL8.

### 20.2. WINDOWS SERVER SETTINGS REQUIRED FOR TRUST CONFIGURATION AND CERTIFICATE USAGE

This section summarizes what must be configured on Windows Server:

- Active Directory Certificate Services (ADCS) is installed
- Certificate Authority is created
- [Optional] If you are using Certificate Authority Web Enrollment, the Internet Information Services (IIS) must be configured

Export the certificate:

- Key must have 2048 bits or more
- Include a private key
- You will need a certificate in the following format: Personal Information Exchange – PKCS #12(.PFX)
  - Enable certificate privacy

### 20.3. COPYING CERTIFICATES FROM ACTIVE DIRECTORY USING SFTP

To be able to use smart card authentication, you need to copy the following certificate files:

- A root CA certificate in the CER format: adcs-winserver-ca.cer on your IdM server.
- A user certificate with a private key in the PFX format: aduser1.pfx on an IdM client.

**NOTE**

This procedure expects SSH access is allowed. If SSH is unavailable the user must copy the file from the AD Server to the IdM server and client.

Procedure
1. Connect from the **IdM server** and copy the **adcs-winserver-ca.cer** root certificate to the IdM server:

```
root@idmserver ~]# sftp Administrator@winserver.ad.example.com
Administrator@winserver.ad.example.com's password:
Connected to Administrator@winserver.ad.example.com.
sftp> cd <Path to certificates>
sftp> ls
adcs-winserver-ca.cer  aduser1.pfx
sftp>
```

```
sftp> get adcs-winserver-ca.cer
Fetching <Path to certificates>/adcs-winserver-ca.cer to adcs-winserver-ca.cer
<Path to certificates>/adcs-winserver-ca.cer                 100%  1254    15KB/s 00:00
sftp quit
```

2. Connect from the **IdM client** and copy the **aduser1.pfx** user certificate to the client:

```
[root@client1 ~]# sftp Administrator@winserver.ad.example.com
Administrator@winserver.ad.example.com's password:
Connected to Administrator@winserver.ad.example.com.
sftp> cd /<Path to certificates>
sftp> get aduser1.pfx
Fetching <Path to certificates>/aduser1.pfx to aduser1.pfx
<Path to certificates>/aduser1.pfx                 100%  1254    15KB/s 00:00
sftp quit
```

Now the CA certificate is stored in the IdM server and the user certificates is stored on the client machine.

### 20.4. Configuring the IdM Server and Clients for Smart Card Authentication Using ADCS Certificates

You must configure the IdM (Identity Management) server and clients to be able to use smart card authentication in the IdM environment. IdM includes the **ipa-advise** scripts which makes all necessary changes:

- install necessary packages
- it configures IdM server and clients
- copy the CA certificates into expected locations

You can run **ipa-advise** on your IdM server.

This procedure describes:

- On an IdM server: Preparing the **ipa-advise** script to configure your IdM server for smart card authentication.
- On an IdM server: Preparing the **ipa-advise** script to configure your IdM client for smart card authentication.
- On an IdM server: Applying the the **ipa-advise** server script on the IdM server using the AD certificate.
• Moving the client script to the IdM client machine.
• On an IdM client: Applying the the `ipa-advise` client script on the IdM client using the AD certificate.

Prerequisites
• The certificate has been copied to the IdM server.
• Obtain the Kerberos ticket.
• Log in as a user with administration rights.

Procedure
1. On the IdM server, use the `ipa-advise` script for configuring a client:

   ```bash
   [root@idmserver ~]# ipa-advise config-client-for-smart-card-auth > sc_client.sh
   ```

2. On the IdM server, use the `ipa-advise` script for configuring a server:

   ```bash
   [root@idmserver ~]# ipa-advise config-server-for-smart-card-auth > sc_server.sh
   ```

3. On the IdM server, execute the script:

   ```bash
   [root@idmserver ~]# sh -x sc_server.sh adcs-winserver-ca.cer
   ```

   • 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.

4. Copy the `sc_client.sh` script to the client system:

   ```bash
   [root@idmserver ~]# scp sc_client.sh root@client1.idm.example.com:/root
   Password: 
   sc_client.sh                  100%  2857   1.6MB/s   00:00
   ```

5. Copy the Windows certificate to the client system:

   ```bash
   [root@idmserver ~]# scp adcs-winserver-ca.cer root@client1.idm.example.com:/root
   Password: 
   adcs-winserver-ca.cer                 100%  1254   952.0KB/s   00:00
   ```

6. On the client system, run the client script:

   ```bash
   [root@idmclient1 ~]# sh -x sc_client.sh adcs-winserver-ca.cer
   ```

The CA certificate is installed in the correct format on the IdM server and client systems and next step is to copy the user certificates onto the smart card itself.
20.5. CONVERTING THE PFX FILE

Before you store the PFX (PKCS#12) file into the smart card, you must:

- convert the file to the PEM format
- extract the private key and the certificate to two different files

Prerequisites

- The PFX file is copied into the IdM client machine.

Procedure

1. On the IdM client, into the PEM format:

   [root@idmclient1 ~]# openssl pkcs12 -in aduser1.pfx -out aduser1_cert_only.pem -clcerts -nodes
   Enter Import Password:

2. Extract the key into the separate file:

   [root@idmclient1 ~]# openssl pkcs12 -in adduser1.pfx -nocerts -out adduser1.pem > aduser1.key

3. Extract the public certificate into the separate file:

   [root@idmclient1 ~]# openssl pkcs12 -in adduser1.pfx -clcerts -nokeys -out aduser1_cert_only.pem > aduser1.crt

At this point, you can store the **aduser1.key** and **aduser1.crt** into the smart card.

20.6. INSTALLING GNUTLS-UTILS

To configure your smart card, you need a tool that can generate certificates and store them in the smart card.

You must:

- Install the **gnutls-utils** program which helps you to manage certificates.
- Start the **pcscd** service which communicates with the smart card reader.

Procedure

1. Install the the **gnutls-utils** package which allows you to manage smart card settings for reading and writing from the smart card:

   # dnf -y install opensc gnutls-utils

2. Start the **pcscd** service.

   # systemctl start pcscd
Verify that the `pcscd` service is up and running.

## 20.7. STORING A CERTIFICATE ON THE SMART CARD

This section describes smart card configuration with `gnutls-utils` which helps you to configure:

- Erasing your smart card
- Setting a new PIN and PUK
- Creating a new slot in the smart card
- Storing the certificate and private key in the slot
- Locking the smart card settings (some smart cards need this type of finalization)

### Prerequisites

- The `gnutls-utils` package is installed. For details, see [Installing gnutls-utils](#).
- The card is inserted in the reader and connected to the computer.

### Procedure

1. Erase your smart card and authenticate yourself with PIN:

   ```bash
   # spawn pkcs15-init --erase-card --use-default-transport-keys
   Using reader with a card: Smart Card name
   PIN [Security Officer PIN] required.
   Please enter PIN [Security Officer PIN]:
   
   The card has been erased.
   ```

2. Set your PIN and PUK – both twice for verification:

   ```bash
   # pkcs15-init --create-pkcs15 --use-default-transport-keys \
   --pin redhat --puk redhat --so-pin redhat --so-puk redhat
   Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface]
   (53311657131456) 00 00
   
   The pkcs15-init creates a new slot in the smart card and marks it with a label and an authentication ID.
   ```

3. Add a label that is the name of the slot and the authentication ID:

   ```bash
   # pkcs15-init --store-pin --label testuser \ 
   --auth-id 01 --so-pin redhat --pin redhat --puk redhat
   Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface]
   (53311657131456) 00 00
   
   The label auth-id is set to 01.
   ```

4. Store the private key in the new slot of the smart card:
5. Store the certificate in the new slot of the smart card:

   ```bash
   # pkcs15-init --store-certificate testuser.crt --label testuser.crt
     --auth-id 01 --id 01 --format pem --so-pin redhat --pin redhat
   Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface]
   (53311657131456) 00 00
   ```

6. Optionally finalize the card settings with locking the settings:

   ```bash
   # pkcs15-init -F
   ```

   **IMPORTANT**
   Some smart cards require this step.

At this stage, your smart card includes the certificate and private key in the newly created slot. PIN and PUK has been successfully created.

### 20.8. CONFIGURING TIMEOUTS IN SSSD.CONF

Authentication with a smart card certificate might take longer than the default timeouts used by SSSD. Time out expiration can be caused by:

- slow reader
- a forwarding form a physical device into a virtual environment
- too many certificates stored on the smart card
- slow response from the OCSP (Online Certificate Status Protocol) responder if OCSP is used to verify the certificates

In this case you can prolong the following timeouts in the `sssd.conf` file, for example, to 60 seconds:

- `p11_child_timeout`
- `krb5_auth_timeout`

**Prerequisites**

- You must be logged in as root.

**Procedure**

1. Open the `sssd.conf` file:

   ```bash
   [root@idmclient1 ~]# vim /etc/sssd/sssd.conf
   ```
2. Change the value of `p11_child_timeout`:

   ```
   [pam]
   p11_child_timeout = 60
   ```

3. Change the value of `krb5_auth_timeout`:

   ```
   [domain/IDM.EXAMPLE.COM]
   krb5_auth_timeout = 60
   ```

4. Save the settings.

Now, the interaction with the smart card is allowed to run for 1 minute (60 seconds) before authentication will fail with a timeout.

### 20.9. CREATING CERTIFICATE MAPPING RULES FOR SMART CARD AUTHENTICATION

If you want to use one certificate for the same user who has accounts in AD (Active Directory) and in IdM (Identity Management) too, you can create a certificate mapping rule on the IdM server. After creating such a rule, the user is able to authenticate with their smart card in both domains.

For details about certificate mapping rules, see [Certificate mapping rules for configuring authentication on smart cards](#).
21.1. 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.


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.

21.1.2. 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.
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**

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}))'
```

21.1.3. 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 file name 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 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.

21.2. 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.

21.2.1. 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.

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

1. Log in to the IdM web UI as an administrator.


3. Click Add.

4. Enter the rule name.
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 21.2. 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.

### 21.2.1.2. 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.

### 21.2.2. 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.

#### 21.2.2.1. 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-----
MIIFCTCCA/2gAwIBAgIBEjANBgkqhkiG9w0BAQsFADA6MRgwFgYDVQQKDA9JRE0u
RVhBTVBMRS5DT00xHjAcBgNVBAMMFUNlcnRpZmljYXRlIEF1dGhvcml0eTAeFw0x
ODA5MDIxODE1MzlaTENBMF4XLRQgXMD0iMTIwMjAxMjAwMTE4OTI5OQYDVQQK
-----END CERTIFICATE-----
```

   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.

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:

```
# ipa certmap-match idm_user_cert.pem
---------
1 user matched
---------
Domain: IDM.EXAMPLE.COM
User logins: idm_user

Number of entries returned 1
```

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`.

### 21.2.2.2. 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-certmapdata` command:

```
# CERT=`cat idm_user_cert.pem | tail -n +2 | head -n -1 | tr -d \"\n\"
# 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</I><S>CN=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**.

### 21.3. 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.

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

#### 21.3.1.1. Adding a certificate mapping rule in the IdM web UI

1. Log into the IdM web UI as an administrator.
3. Click **Add**.

*Figure 21.5. Adding a new certificate mapping rule in the IdM web UI*

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:

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

Figure 21.6. 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
```

21.3.1.2. 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
```

### 21.4. 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.

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

**21.4.1.1. Adding a certificate mapping rule in the IdM web UI**

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


3. Click **Add**.

   ![Figure 21.7. 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:

   ```
   (altSecurityIdentities=X509:<I>{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:

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

7. Enter the domain:

   `ad.example.com`

Figure 21.8. 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`

21.4.1.2. 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
```

### 21.4.2. 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`:

  ```bash
  $ 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<SI>DC=com,DC=example,DC=ad,CN=Users,CN=ad_user
  ```

### 21.5. 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.

#### 21.5.1. Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data
21.5.1.1. 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:

   \[
   \text{(userCertificate;binary={cert!bin})}
   \]

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:

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

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

   \[
   \text{domain} = \text{ad.example.com}
   \]

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:
21.5.1.2. Adding a certificate mapping rule in the IdM CLI

1. Obtain the administrator’s credentials:

   ```sh
   # 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:

   ```sh
   # 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:

   ```sh
   # systemctl restart sssd
   ```

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

21.5.2.1. 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 21.11. 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:

   ```bash
   # 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:

   ```bash
   # 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**.
21.5.2.2. 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`.

21.6. 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:<|{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:<|{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:<|{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})(altSecurityIdentities=X509:<|{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:<|{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:<|{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.
CHAPTER 22. 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 22.2, “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 22.3, “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.

22.1. 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:

```bash
# ./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.

### 22.2. 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:

```bash
# 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`:

```bash
# 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 randomlly 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: 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:

```bash
# chmod o-rwx /home/idm_user/
```

10. For security reasons, remove the temporary NSS database and the .pkcs12 file from the server:

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

22.3. 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 22.2, “Requesting a new user certificate and exporting it to the client”.

22.4. CONFIGURING A BROWSER TO ENABLE CERTIFICATE AUTHENTICATION

To be able to authenticate with a certificate when using the WebUI to log into Identity Management (IdM), you need to import the user and the relevant certificate authority (CA) certificates into the Mozilla Firefox or Google Chrome browser. The host itself on which the browser is running does not have to be part of the IdM domain.

IdM supports the following browsers for connecting to the WebUI:

- 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**

- You have the user certificate that you want to import to the browser at your disposal in the PKCS#12 format.

**Procedure**
1. Open Firefox, then navigate to **Preferences → Privacy & Security**.

**Figure 22.1. Privacy and Security section in Preferences**

2. Click **View Certificates**.

**Figure 22.2. 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.

**Figure 22.3. Insecure Connection**

- **Add Exception**. Click **View**.
Figure 22.4. View the Details of a Certificate

- In the **Details** tab, highlight the **Certificate Authority** fields.

Figure 22.5. 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**.

**Figure 22.6. Privacy and Security section in Preferences**

- Click **View Certificates**.

**Figure 22.7. View Certificates in Privacy and Security**

- 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**.

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

### 22.5. 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**. **Login Using Certificate** in the Identity Management web UI
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

**22.6. 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 22.2, “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:

  ```
  $ 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 23. USING IDM CA RENEWAL MASTER

23.1. EXPLANATION OF IDM CA RENEWAL MASTER

In an Identity Management (IdM) deployment that uses an embedded certificate authority (CA), the CA renewal master server maintains and renews IdM system certificates. It ensures 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. In contrast, the IdM service certificates (for example, **LDAP**, **HTTP**, and **PKINIT** certificates), 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**

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 does not contain a CA renewal master, each IdM CA server requests the renewal of system certificates independently. This results 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 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. In this situation, 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. To prevent this situation: 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.

### 23.2. CHANGING AND RESETTING IDM CA RENEWAL MASTER

When a certificate authority (CA) renewal master is being decommissioned, Identity Management (IdM) automatically selects a new CA renewal master from the list of IdM CA servers. The system administrator cannot influence the selection.

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

If the current CA renewal master configuration is invalid, reset the IdM CA renewal master.

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
~]$ kinit admin
Password for admin@IDM.EXAMPLE.COM:

~]$ 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
----------------------
```

There are two CA servers in the deployment.

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

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

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 both renews the CA certificate and makes the CA server on which you execute the command the new CA renewal master.

- the `ipa-cert-fix` command. This command recovers the deployment when expired certificates are causing failures. It also makes the CA server on which you execute the command the new CA renewal master.

For details, see [Renewing expired system certificates when IdM is offline](#).

### 23.3. 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 Identity Management (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 certificate signing request (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.
23.4. RENEWING THE IDM CA RENEWAL MASTER WITH AN EXTERNALLY-SIGNED CERTIFICATE

This section describes how to renew the Identity Management (IdM) certificate authority (CA) certificate using an external CA 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:

   ~]# 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:

   [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:

   [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
The output shows that the update has been successful as the new CA certificate is listed with the older CA certificates.
CHAPTER 24. RENEWING EXPIRED SYSTEM CERTIFICATES WHEN IDM IS OFFLINE

When a system certificate has expired, Identity Management (IdM) fails to start. IdM supports renewing system certificates when IdM is offline using the `ipa-cert-fix` tool.

Prerequisites

- IdM is installed only on Red Hat Enterprise Linux 8.1 or later

24.1. RENEWING EXPIRED SYSTEM CERTIFICATES ON A CA RENEWAL MASTER

This section describes how to apply the `ipa-cert-fix` tool on expired IdM certificates.

**IMPORTANT**

If you run the `ipa-cert-fix` tool on a CA (Certificate Authority) host that is not the CA Renewal Master, and the utility renews shared certificates, that host automatically becomes the new CA Renewal Master in the domain. There must be always only one CA Renewal Master in the domain to avoid inconsistencies.

Prerequisites

- Log in to the server with administration rights

Procedure

1. Start the `ipa-cert-fix` tool to analyze the system and list expired certificates that require renewal:

```
# ipa-cert-fix
... The following certificates will be renewed:
Dogtag sslserver certificate:
Subject: CN=ca1.example.com,O=EXAMPLE.COM 201905222205
Serial:  13
...
Enter "yes" to proceed:
```

2. Enter `yes` to start the renewal process:

```
Enter "yes" to proceed: yes
Proceeding.
Renewed Dogtag sslserver certificate:
Subject: CN=ca1.example.com,O=EXAMPLE.COM 201905222205
Serial:  268369925
Expires: 2021-08-14 02:19:33
...
```
Becoming renewal master.
The ipa-cert-fix command was successful

It can take up to one minute before ipa-cert-fix renews all expired certificates.

3. Optionally, verify that all services are now running:

   # ipactl status
   Directory Service: RUNNING
   krb5kdc Service: RUNNING
   kadmin Service: RUNNING
   httpd Service: RUNNING
   ipa-custodia Service: RUNNING
   pki-tomcat Service: RUNNING
   ipa-otpd Service: RUNNING
   ipa: INFO: The ipactl command was successful

At this point, certificates have been renewed and services are running. The next step is to check other servers in the IdM domain.

24.2. VERIFYING OTHER IDM SERVERS IN THE IDM DOMAIN AFTER RENEWAL

After the renewing the CA Renewal Master’s certificates with the ipa-cert-fix tool, you must:

- Restart all other Identity Management (IdM) servers in the domain.
- Check if certmonger renewed certificates.
- If there are other Certificate Authority (CA) replicas with expired system certificates, renew those certificates with the ipa-cert-fix tool as well.

Prerequisites

- Log in to the server with administration rights.

Procedure

1. Restart IdM with the --force parameter:

   # ipactl restart --force

   With the --force parameter, the ipactl utility ignores individual service startup failures. For example, if the server is also a CA with expired certificates, the pki-tomcat service fails to start. This is expected and ignored because of using the --force parameter.

2. After the restart, verify that the certmonger service renewed the certificates (certificate status says MONITORING):

   # getcert list | egrep '^Request|status:|subject:'
   Request ID '20190522120745':
       status: MONITORING
       subject: CN=IPA RA,O=EXAMPLE.COM 201905222205
Request ID '20190522120834':
  status: MONITORING
  subject: CN=Certificate Authority,O=EXAMPLE.COM 2019052222205
...

It can take some time before certmonger renews the shared certificates on the replica.

3. If the server is also a CA, the previous command reports **CA_UNREACHABLE** for the certificate the **pki-tomcat** service uses:

Request ID '20190522120835':
  status: CA_UNREACHABLE
  subject: CN=ca2.example.com,O=EXAMPLE.COM 2019052222205
...

4. To renew this certificate, use the **ipa-cert-fix** utility:

```
# ipa-cert-fix
Dogtag sslserver certificate:
  Subject: CN=ca2.example.com,O=EXAMPLE.COM
  Serial: 3

Enter "yes" to proceed: yes
Proceeding.
Renewed Dogtag sslserver certificate:
  Subject: CN=ca2.example.com,O=EXAMPLE.COM 2019052222205
  Serial: 15
  Expires: 2019-08-14 04:25:05

The ipa-cert-fix command was successful
```

Now, all IdM certificates have been renewed and work correctly.
CHAPTER 25. GENERATING CRL ON THE IDM CA SERVER

If your IdM deployment uses an embedded certificate authority (CA), you may need to move generating the Certificate Revocation List (CRL) from one Identity Management (IdM) server to another. It can be necessary, for example, when you want to migrate the server to another system.

Only one server must generate CRL. The CRL generation role is usually co-located with the IdM CA Renewal Master, but this is not mandatory. Before the CRL Generation Master is decommissioned, a new CRL Generation Master must be selected by the administrator and configured.

This chapter describes:

- Stopping CRL generation on the IdM master.
- Starting to generate CRL on the IdM replica.

25.1. STOPPING CRL GENERATION ON IDM MASTER SERVER

To stop the Certificate Revocation List (CRL) generation on the IdM master server, use the `ipa-crlgen-manage` command. Before you disable the generation, verify that the server really generates CRL. You can then disable it.

Prerequisites

- Identity Management (IdM) server is installed on the RHEL 8.1 system or newer.
- You must be logged in as root.

Procedure

1. Check if your master server is generating the CRL:

   ```
   [root@master ~]# ipa-crlgen-manage status
   CRL generation: enabled
   Last CRL update: 2019-10-31 12:00:00
   Last CRL Number: 6
   The ipa-crlgen-manage command was successful
   ```

2. Stop generating CRL on the master server:

   ```
   [root@master ~]# ipa-crlgen-manage disable
   Stopping pki-tomcatd
   Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
   Starting pki-tomcatd
   Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
   Restarting httpd
   CRL generation disabled on the local host. Please make sure to configure CRL generation on another master with ipa-crlgen-manage enable.
   The ipa-crlgen-manage command was successful
   ```

3. Check if the master server stopped generating CRL:

   ```
   [root@master ~]# ipa-crlgen-manage status
   ```
The master server stopped generating CRL. The next step is to enable CRL generation on the new master server.

25.2. STARTING CRL GENERATION ON IDM REPLICA SERVER

You can start the Certificate Revocation List (CRL) generation with the following command: `ipa-crlgen-manage`

Prerequisites

- Identity Management (IdM) server is installed on the RHEL 8.1 system or newer.
- The RHEL system must be an IdM Certificate Authority server.
- You must be logged in as root.

Procedure

1. To start with generating CRL:

   ```bash
   [root@replica1 ~]# ipa-crlgen-manage enable
   Stopping pki-tomcatd
   Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
   Starting pki-tomcatd
   Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
   Restarting httpd
   Forcing CRL update
   CRL generation enabled on the local host. Please make sure to have only a single CRL generation master.
   The ipa-crlgen-manage command was successful
   ```

2. To check if CRL is generated:

   ```bash
   [root@replica1 ~]# ipa-crlgen-manage status
   CRL generation: enabled
   Last CRL update: 2019-10-31 12:10:00
   Last CRL Number: 7
   The ipa-crlgen-manage command was successful
   ```
CHAPTER 26. OBTAINING AN IDM CERTIFICATE FOR A SERVICE USING CERTMONGER

26.1. 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 CA 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

### 26.2. Obtaining an IDM Certificate for a Service Using Certmonger

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

This section describes how to use `certmonger` to obtain an IdM certificate for a service (HTTP/my_company.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 26.3, “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 my_company.idm.example.com IdM client on which the HTTP service is running, request a certificate for the service corresponding to the HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM principal, and specify 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 my_company.idm.example.com:

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

   In the command above:
The `ipa-getcert request` command specifies that the certificate is to be obtained from the IdM CA. The `ipa-getcert request` 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 with a particular profile, use the `-T` option.

To request a certificate using the named issuer from the specified CA, use the `-X ISSUER` 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:

```bash
# 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.

### 26.3. 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 Identity Management (IdM) certificate authority (CA) server. The sequence consists of these diagrams:

- **Figure 26.1**, "Unencrypted communication"
- **Figure 26.2**, "Certmonger requesting a service certificate"
- **Figure 26.3**, "IdM CA issuing the service certificate"
- **Figure 26.4**, "Certmonger applying the service certificate"
- **Figure 26.5**, "Certmonger requesting a new certificate when the old one is nearing expiration"

*Figure 26.1, “Unencrypted communication”* shows the initial situation: without an HTTPS certificate, the communication between the web server and the browser is unencrypted.
Figure 26.1. Unencrypted communication

Figure 26.2, “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 communicate directly with the CA. It proxies through IdM.

Figure 26.2. Certmonger requesting a service certificate
Figure 26.3, “IdM CA issuing the service certificate” shows an IdM CA issuing an HTTPS certificate for the web server.

Figure 26.3. IdM CA issuing the service certificate

Figure 26.4, “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 26.4. Certmonger applying the service certificate

Figure 26.5. "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 26.5. Certmonger requesting a new certificate when the old one is nearing expiration
26.4. 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 (CA) 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.

### 26.5. 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 Identity Management (IdM) certificate authority (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 `-c` option results in certmonger entering the NEED_CA state.

For more options, see the `getcert start-tracking` man page.

**NOTE**

The two commands do not manipulate the certificate. For example, `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, `getcert start-tracking` only adds a certificate to the list of monitored certificates.

### 26.6. RENEWING A CERTIFICATE MANUALLY

When a certificate is near its expiration date, the certmonger daemon automatically issues a renewal command using the certificate authority (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 `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.

**Procedure**

- To renew a certificate with the Request ID of 20190408143846:

  ```
  # getcert resubmit -i 20190408143846
  ```

To obtain the Request ID for a specific certificate, use the `getcert list` command. For details, see the `getcert list` man page.

### 26.7. MAKING CERTMONGER RESUME TRACKING OF IDM CERTIFICATES ON A CA REPLICA

This procedure shows how to make certmonger resume the tracking of Identity Management (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 certmonger resume the tracking of the IdM service certificates, namely the HTTP, LDAP and PKINIT certificates.

**Prerequisites**

- The host on which you want to resume tracking system certificates is an IdM server that is also an IdM certificate authority (CA) but not the IdM CA renewal master.

**Procedure**

1. Get the PIN for the subsystem CA certificates:
2. Add tracking to the subsystem CA certificates, replacing [internal PIN] in the commands below with the PIN obtained in the previous step:

```
# grep 'internal=' /var/lib/pki/pki-tomcat/conf/password.conf

# 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"'

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "Server-Cert 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 "Server-Cert cert-pki-ca"'
```

3. Add tracking for the remaining IdM certificates, the HTTP, LDAP, IPA renewal agent and PKINIT certificates:

```
# 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

```

4. Restart certmonger:

```
# systemctl restart certmonger
```
5. Wait for one minute after `certmonger` has started and then check the statuses of the new certificates:

```
# getcert list
```

### 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 27. RESTRICTING AN APPLICATION TO TRUST ONLY A SUBSET OF CERTIFICATES

If your Identity Management (IdM) installation is configured with the integrated Certificate System (CS) certificate authority (CA), you are able to create lightweight sub-CAs. All sub-CAs you create are subordinated to the primary CA of the certificate system, the ipa CA.

A lightweight sub-CA in this context means a sub-CA issuing certificates for a specific purpose. For example, a lightweight sub-CA enables you to configure a service, such as a virtual private network (VPN) gateway and a web browser, to accept only certificates issued by sub-CA A. By configuring other services to accept certificates only issued by sub-CA B, you prevent them from accepting certificates issued by sub-CA A, the primary CA, that is the ipa CA, and any intermediate sub-CA between the two.

If you revoke the intermediate certificate of a sub-CA, all certificates issued by this sub-CA are automatically considered invalid by correctly configured clients. All the other certificates issued directly by the root CA, ipa, or another sub-CA, remain valid.

This section uses the example of the Apache web server to illustrate how to restrict an application to trust only a subset of certificates. Complete this section to restrict the web server running on your IdM client to use a certificate issued by the webserver-ca IdM sub-CA, and to require the users to authenticate to the web server using user certificates issued by the webclient-ca IdM sub-CA.

The steps you need to take are:

1. Create an IdM sub-CA
2. Download the sub-CA certificate from IdM WebUI
3. Create a CA ACL specifying the correct combination of users, services and CAs, and the certificate profile used
4. Request a certificate for the web service running on an IdM client from the IdM sub-CA
5. Set up a single-instance Apache HTTP Server
6. Add TLS encryption to the Apache HTTP Server
7. Set the supported TLS protocol versions on an Apache HTTP Server
8. Set the supported ciphers on the Apache HTTP Server
9. Configure TLS client certificate authentication on the web server
10. Request a certificate for the user from the IdM sub-CA and export it to the client
11. Import the user certificate into the browser and configure the browser to trust the sub-CA certificate

27.1. CREATING A LIGHTWEIGHT SUB-CA

For details on creating a sub-CA, see:

- Section 27.1.1, “Creating a sub-CA from IdM WebUI”
- Section 27.1.2, “Creating a sub-CA from IdM CLI”
27.1.1. Creating a sub-CA from IdM WebUI

This procedure describes how to use IdM WebUI to create new sub-CAs named `webserver-ca` and `webclient-ca`.

Prerequisites

- Make sure you have obtained the administrator’s credentials.

Procedure

1. In the Authentication menu, click Certificates.

2. Select Certificate Authorities and click Add.

3. Enter the name of the `webserver-ca` sub-CA. Enter the Subject DN, for example `CN=WEBSERVER,O=IDM.EXAMPLE.COM`, in the Subject DN field. Note that the Subject DN must be unique in the IdM CA infrastructure.

4. Enter the name of the `webclient-ca` sub-CA. Enter the Subject DN `CN=WEBCLIENT,O=IDM.EXAMPLE.COM` in the Subject DN field.

5. In the command-line interface, run the `ipa-certupdate` command to create a `certmonger` tracking request for the `webserver-ca` and `webclient-ca` sub-CAs certificates:

   ```
   [root@ipaserver ~]# ipa-certupdate
   ```

   IMPORTANT
   
   Forgetting to run the `ipa-certupdate` command after creating a sub-CA means that if the sub-CA certificate expires, end-entity certificates issued by the sub-CA are considered invalid even if the end-entity certificate has not expired.

6. Optionally, to verify that the signing certificate of the new sub-CA has been added to the IdM database, enter:

   ```
   [root@ipaserver ~]# certutil -d /etc/pki/pki-tomcat/alias/ -L
   Certificate Nickname                      Trust Attributes
   SSL,S/MIME,JAR/XPI                        
   caSigningCert cert-pki-ca                 CTu,Cu,Cu
   Server-Cert cert-pki-ca                   u,u,u
   auditSigningCert cert-pki-ca              u,u,Pu
   caSigningCert cert-pki-ca ba83f324-5e50-4114-b109-acca05d6f1dc u,u,u
   ocspSigningCert cert-pki-ca               u,u,u
   subsystemCert cert-pki-ca                 u,u,u
   ```

   NOTE
   
   The new sub-CA certificate is automatically transferred to all the replicas that have a certificate system instance installed.
27.1.2. Creating a sub-CA from IdM CLI

This procedure describes how to use IdM CLI to create new sub-CAs named webserver-ca and webclient-ca.

Prerequisites

- Make sure that you have obtained the administrator’s credentials.
- Make sure you are logged in to an IdM server that is a CA server.

Procedure

1. Enter the `ipa ca-add` command, and specify the name of the webserver-ca sub-CA and its Subject Distinguished Name (DN):

```
[root@ipaserver ~]# ipa ca-add webserver-ca --
subject="CN=WEBSERVER,O=IDM.EXAMPLE.COM"
-------------------
Created CA "webserver-ca"
-------------------
Name: webserver-ca
Authority ID: ba83f324-5e50-4114-b109-acca05d6f1dc
Subject DN: CN=WEBSERVER,O=IDM.EXAMPLE.COM
Issuer DN: CN=Certificate Authority,O=IDM.EXAMPLE.COM
```

**Name**

Name of the CA.

**Authority ID**

Automatically created, individual ID for the CA.

**Subject DN**

Subject Distinguished Name (DN). The Subject DN must be unique in the IdM CA infrastructure.

**Issuer DN**

Parent CA that issued the sub-CA certificate. All sub-CAs are created as a child of the IdM root CA.

2. Create the webclient-ca sub-CA for issuing certificates to web clients:

```
[root@ipaserver ~]# ipa ca-add webclient-ca --
subject="CN=WEBCLIENT,O=IDM.EXAMPLE.COM"
-------------------
Created CA "webclient-ca"
-------------------
Name: webclient-ca
Authority ID: 8a479f3a-0454-4a4d-8ade-fd3b5a54ab2e
Subject DN: CN=WEBCLIENT,O=IDM.EXAMPLE.COM
Issuer DN: CN=Certificate Authority,O=IDM.EXAMPLE.COM
```

3. In the command-line interface, run the `ipa-certupdate` command to create a certmonger tracking request for the webserver-ca and webclient-ca sub-CAs certificates:

```
[root@ipaserver ~]# ipa-certupdate
```
IMPORTANT

Forgetting to run the `ipa-certupdate` command after creating a sub-CA means that if the sub-CA certificate expires, end-entity certificates issued by the sub-CA are considered invalid even if the end-entity certificate has not expired.

4. Optionally, to verify that the signing certificate of the new sub-CA has been added to the IdM database, enter:

```
[root@ipaserver ~]# certutil -d /etc/pki/pki-tomcat/alias/ -L
```

<table>
<thead>
<tr>
<th>Certificate Nickname</th>
<th>Trust Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>caSigningCert cert-pki-ca</td>
<td>CTu,Cu,Cu</td>
</tr>
<tr>
<td>Server-Cert cert-pki-ca</td>
<td>u,u,u</td>
</tr>
<tr>
<td>auditSigningCert cert-pki-ca</td>
<td>u,u,Pu</td>
</tr>
<tr>
<td>caSigningCert cert-pki-ca ba83f324-5e50-4114-b109-acca05d6f1dc</td>
<td>u,u,u</td>
</tr>
<tr>
<td>ocspSigningCert cert-pki-ca</td>
<td>u,u,u</td>
</tr>
<tr>
<td>subsystemCert cert-pki-ca</td>
<td>u,u,u</td>
</tr>
</tbody>
</table>

27.2. DOWNLOADING THE SUB-CA CERTIFICATE FROM IDM WEBUI

Prerequisites

- Make sure that you have obtained the IdM administrator’s credentials.

Procedure

1. In the Authentication menu, click Certificates > Certificates.

   ![Figure 27.1. sub-CA certificate in the list of certificates](image)

2. Click the serial number of the sub-CA certificate to open the certificate information page.

3. In the certificate information page, click Actions > Download.

27.3. CREATING CA ACLS FOR WEB SERVER AND CLIENT AUTHENTICATION

Certificate authority access control list (CA ACL) rules define which profiles can be used to issue certificates to which users, services, or hosts. By associating profiles, principals, and groups, CA ACLs permit principals or groups to request certificates using particular profiles.
For example, using CA ACLs, the administrator can restrict the use of a profile intended for employees working from an office located in London only to users that are members of the London office-related group.

27.3.1. Viewing CA ACLs in IdM CLI

Complete this section to view the list of certificate authority access control lists (CA ACLs) available in your IdM deployment and the details of a specific CA ACL.

Procedure

1. To view all the CA ACLs in your IdM environment, enter the `ipa caacl-find` command:

```
$ ipa caacl-find
-------------
1 CA ACL matched
-------------
ACL name: hosts_services_calIPAserviceCert
Enabled: TRUE
```

2. To view the details of a CA ACL, enter the `ipa caacl-show` command, and specify the CA ACL name. For example, to view the details of the `hosts_services_calIPAserviceCert` CA ACL, enter:

```
$ ipa caacl-show hosts_services_calIPAserviceCert
ACL name: hosts_services_calIPAserviceCert
Enabled: TRUE
Host category: all
Service category: all
CAs: ipa
Profiles: calIPAserviceCert
Users: admin
```

27.3.2. Creating a CA ACL for web servers authenticating to web clients using certificates issued by webserver-ca

This section describes how to create a CA ACL that requires the system administrator to use the `webserver-ca` sub-CA and the `calIPAserviceCert` profile when requesting a certificate for the `HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM` service. If the user requests a certificate from a different sub-CA or of a different profile, the request fails. The only exception is when there is another matching CA ACL that is enabled. To view the available CA ACLs, see Viewing CA ACLs in IdM CLI.

Prerequisites

- Make sure that the `HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM` service is part of IdM.
- Make sure you have obtained IdM administrator’s credentials.

Procedure

1. Create a CA ACL using the `ipa caacl` command, and specify its name:

```
$ ipa caacl-add TLS_web_server_authentication
```
2. Modify the CA ACL using the `ipa caacl-mod` command to specify the description of the CA ACL:

```
$ ipa caacl-mod TLS_web_server_authentication --desc="CAACL for web servers authenticating to web clients using certificates issued by webserver-ca"
```

Modified CA ACL "TLS_web_server_authentication"

ACL name: TLS_web_server_authentication
Description: CAACL for web servers authenticating to web clients using certificates issued by webserver-ca
Enabled: TRUE

3. Add the `webserver-ca` sub-CA to the CA ACL:

```
$ ipa caacl-add-ca TLS_web_server_authentication --ca=webserver-ca
```

ACL name: TLS_web_server_authentication
Description: CAACL for web servers authenticating to web clients using certificates issued by webserver-ca
Enabled: TRUE
CAs: webserver-ca

Number of members added 1

4. Use the `ipa caacl-add-service` command to specify the service whose principal will be able to request a certificate:

```
$ ipa caacl-add-service TLS_web_server_authentication --service=HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM
```

ACL name: TLS_web_server_authentication
Description: CAACL for web servers authenticating to web clients using certificates issued by webserver-ca
Enabled: TRUE
CAs: webserver-ca
Services: HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM

Number of members added 1

5. Use the `ipa caacl-add-profile` command to specify the certificate profile for the requested certificate:

```
$ ipa caacl-add-profile TLS_web_server_authentication --certprofiles=caIPAserviceCert
```

ACL name: TLS_web_server_authentication
Description: CAACL for web servers authenticating to web clients using certificates issued by webserver-ca
You can use the newly-created CA ACL straight away. It is enabled after its creation by default.

**NOTE**

The point of CA ACLs is to specify which CA and profile combinations are allowed for requests coming from particular principals or groups. CA ACLs do not affect certificate validation or trust. They do not affect how the issued certificates will be used.

### 27.3.3. Creating a CA ACL for user web browsers authenticating to web servers using certificates issued by webclient-ca

This section describes how to create a CA ACL that requires the system administrator to use the **webclient-ca** sub-CA and the **IECUserRoles** profile when requesting a certificate. If the user requests a certificate from a different sub-CA or of a different profile, the request fails. The only exception is when there is another matching CA ACL that is enabled. To view the available CA ACLs, see [Viewing CA ACLs in IdM CLI](#).

#### Prerequisites

- Make sure that you have obtained IdM administrator’s credentials.

#### Procedure

1. Create a CA ACL using the `ipa caacl` command and specify its name:

   ```bash
   $ ipa caacl-add TLS_web_client_authentication
   ---------------------------------------------
   Added CA ACL "TLS_web_client_authentication"
   ---------------------------------------------
   ACL name: TLS_web_client_authentication
   Enabled: TRUE
   ```

2. Modify the CA ACL using the `ipa caacl-mod` command to specify the description of the CA ACL:

   ```bash
   $ ipa caacl-mod TLS_web_client_authentication --desc="CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca"
   -----------------------------------------------
   Modified CA ACL "TLS_web_client_authentication"
   -----------------------------------------------
   ACL name: TLS_web_client_authentication
   Description: CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca
   Enabled: TRUE
   ```

3. Add the **webclient-ca** sub-CA to the CA ACL:
$ ipa caacl-add-ca TLS_web_client_authentication --ca=webclient-ca
ACL name: TLS_web_client_authentication
Description: CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca
Enabled: TRUE
CAs: webclient-ca

Number of members added 1

4. Use the ipa caacl-add-profile command to specify the certificate profile for the requested certificate:

$ ipa caacl-add-profile TLS_web_client_authentication --certprofiles=IECUserRoles
ACL name: TLS_web_client_authentication
Description: CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca
Enabled: TRUE
CAs: webclient-ca
Profiles: IECUserRoles

Number of members added 1

5. Modify the CA ACL using the ipa caacl-mod command to specify that the CA ACL applies to all IdM users:

$ ipa caacl-mod TLS_web_client_authentication --usercat=all

Modified CA ACL "TLS_web_client_authentication"

ACL name: TLS_web_client_authentication
Description: CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca
Enabled: TRUE
User category: all
CAs: webclient-ca
Profiles: IECUserRoles

You can use the newly-created CA ACL straight away. It is enabled after its creation by default.

NOTE

The point of CA ACLs is to specify which CA and profile combinations are allowed for requests coming from particular principals or groups. CA ACLs do not affect certificate validation or trust. They do not affect how the issued certificates will be used.

27.4. 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. If you want to restrict web browsers to trust certificates issued by the webserver-ca sub-CA but no other IdM sub-CA, obtain the TLS certificate for your web service from the webserver-ca sub-CA.
This section describes how to use `certmonger` to obtain an IdM certificate for a service (HTTP/my_company.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 27.5, “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 my_company.idm.example.com IdM client on which the HTTP service is running, request a certificate for the service corresponding to the HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM principal, and specify 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
   - The webserver-ca sub-CA is to be the issuing certificate authority
   - That an extensionRequest for a SubjectAltName be added to the signing request with the DNS name of my_company.idm.example.com:

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

In the command above:

- The `ipa-getcert request` command specifies that the certificate is to be obtained from the IdM CA. The `ipa-getcert request` 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.
- The `-X` option specifies that the issuer of the certificate must be `webserver-ca`, not `ipa`.
- To specify that the certificate be issued 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
  issuer: CN=WEBSERVER,O=IDM.EXAMPLE.COM

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.

27.5. 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 Identity Management (IdM) certificate authority (CA) server. The sequence consists of these diagrams:

- Figure 27.2, "Unencrypted communication"
- Figure 27.3, "Certmonger requesting a service certificate"
- Figure 27.4, "IdM CA issuing the service certificate"
- Figure 27.5, "Certmonger applying the service certificate"
- Figure 27.6, "Certmonger requesting a new certificate when the old one is nearing expiration"

In the diagrams, the webserver-ca sub-CA is represented by the generic IdM CA server.

Figure 27.2, "Unencrypted communication" shows the initial situation: without an HTTPS certificate, the communication between the web server and the browser is unencrypted.
Figure 27.2. Unencrypted communication

Figure 27.3, “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 communicate directly with the CA. It proxies through IdM.

Figure 27.3. Certmonger requesting a service certificate
Figure 27.4, “IdM CA issuing the service certificate” shows an IdM CA issuing an HTTPS certificate for the web server.

Figure 27.4. IdM CA issuing the service certificate

Figure 27.5, “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 27.5. Certmonger applying the service certificate

Figure 27.6, “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 27.6. Certmonger requesting a new certificate when the old one is nearing expiration
27.6. SETTING UP A SINGLE-INSTANCE APACHE HTTP SERVER

This section describes how to set up a single-instance Apache HTTP Server to serve static HTML content.

Follow the procedure in this section if the web server should provide the same content for all domains associated with the server. If you want to provide different content for different domains, set up name-based virtual hosts. For details, see Configuring Apache name-based virtual hosts.

**Procedure**

1. Install the `httpd` package:
   
   ```
   # yum install httpd
   ```

2. Open the TCP port **80** in the local firewall:
   
   ```
   # firewall-cmd --permanent --add-port=80/tcp
   # firewall-cmd --reload
   ```

3. Enable and start the `httpd` service:
   
   ```
   # systemctl enable --now httpd
   ```


**Verification steps**

- Connect with a web browser to `http://my_company.idm.example.com/` or `http://server_IP/`. If the `/var/www/html/` directory is empty or does not contain an `index.html` or `index.htm` file, Apache displays the Red Hat Enterprise Linux Test Page. If `/var/www/html/` contains HTML files with a different name, you can load them by entering the URL to that file, such as `http://server_IP/example.html` or `http://my_company.idm.example.com/example.html`.

**Additional resources**

- For further details about configuring Apache and adapting the service to your environment, refer to the Apache manual. For details about installing the manual, see Installing the Apache HTTP Server manual.

- For details about using or adjusting the `httpd systemd` service, see the `httpd.service(8)` man page.

27.7. ADDING TLS ENCRYPTION TO AN APACHE HTTP SERVER

This section describes how to enable TLS encryption on the `my_company.idm.example.com` Apache HTTP Server for the `idm.example.com` domain.

**Prerequisites**

- The `my_company.idm.example.com` Apache HTTP Server is installed and running.
You have obtained the TLS certificate from the `webserver-ca` sub-CA, and stored it in the `/etc/pki/tls/certs/httpd.pem` file as described in Section 27.4, “Obtaining an IdM certificate for a service using certmonger”. If you use a different path, adapt the corresponding steps of the procedure.

The corresponding private key is stored in the `/etc/pki/tls/private/httpd.key` file. If you use a different path, adapt the corresponding steps of the procedure.

The `webserver-ca` CA certificate is stored in the `/etc/pki/tls/private/ca.crt` file. If you use a different path, adapt the corresponding steps of the procedure.

Clients and the `my_company.idm.example.com` web server resolve the host name of the server to the IP address of the web server.

**Procedure**

1. Install the `mod_ssl` package:

   ```
   # dnf install mod_ssl
   ```

2. Edit the `/etc/httpd/conf.d/ssl.conf` file and add the following settings to the `<VirtualHost _default_:443>` directive:

   a. Set the server name:

   ```
   ServerName my_company.idm.example.com
   ```

   IMPORTANT

   The server name must match the entry set in the `Common Name` field of the certificate.

   b. Optional: If the certificate contains additional host names in the `Subject Alt Names` (SAN) field, you can configure `mod_ssl` to provide TLS encryption also for these host names. To configure this, add the `ServerAliases` parameter with corresponding names:

   ```
   ServerAlias www.my_company.idm.example.com server.my_company.idm.example.com
   ```

   c. Set the paths to the private key, the server certificate, and the CA certificate:

   ```
   SSLCertificateKeyFile "/etc/pki/tls/private/httpd.key"
   SSLCertificateFile "/etc/pki/tls/certs/httpd.pem"
   SSLCACertificateFile "/etc/pki/tls/certs/ca.crt"
   ```

3. For security reasons, configure that only the `root` user can access the private key file:

   ```
   # chown root:root /etc/pki/tls/private/httpd.key
   # chmod 600 /etc/pki/tls/private/httpd.key
   ```
4. Open port **443** in the local firewall:

   ```
   # firewall-cmd --permanent --add-port=443
   # firewall-cmd --reload
   ```

5. Restart the **httpd** service:

   ```
   # systemctl restart httpd
   ```

**NOTE**

If you protected the private key file with a password, you must enter this password each time when the **httpd** service starts.

- Use a browser and connect to `https://my_company.idm.example.com`.

**Additional resources**

- For further details about configuring TLS, refer to the **SSL/TLS Encryption** documentation in the Apache manual. For details about installing the manual, see Installing the Apache HTTP Server manual.

### 27.8. SETTING THE SUPPORTED TLS PROTOCOL VERSIONS ON AN APACHE HTTP SERVER

By default, the Apache HTTP Server on RHEL 8 uses the system-wide crypto policy that defines safe default values, which are also compatible with recent browsers. For example, the **DEFAULT** policy defines that only the **TLSv1.2** and **TLSv1.3** protocol versions are enabled in apache.

This section describes how to manually configure which TLS protocol versions your `my_company.idm.example.com` Apache HTTP Server supports. Follow the procedure if your environment requires to enable only specific TLS protocol versions, for example:

- If your environment requires that clients can also use the weak **TLS1** (TLSv1.0) or **TLS1.1** protocol.
- If you want to configure that Apache only supports the **TLSv1.2** or **TLSv1.3** protocol.

**Prerequisites**

- TLS encryption is enabled on the `my_company.idm.example.com` server as described in Section 27.7, “Adding TLS encryption to an Apache HTTP Server”.
**Procedure**

1. Edit the `/etc/httpd/conf/httpd.conf` file, and add the following setting to the `<VirtualHost>` directive for which you want to set the TLS protocol version. For example, to enable only the **TLSv1.3** protocol:

   ```
   SSLProtocol -All TLSv1.3
   ```

2. Restart the `httpd` service:

   ```
   # systemctl restart httpd
   ```

**Verification steps**

1. Use the following command to verify that the server supports **TLSv1.3**:

   ```
   # openssl s_client -connect example.com:443 -tls1_3
   ```

2. Use the following command to verify that the server does not support **TLSv1.2**:

   ```
   # openssl s_client -connect example.com:443 -tls1_2
   ```

   If the server does not support the protocol, the command returns an error:

   ```
   140111600609088:error:1409442E:SSL routines:ssl3_read_bytes:tlsv1 alert protocol
   version:ssl/record/rec_layer_s3.c:1543:SSL alert number 70
   ```

3. Optional: Repeat the command for other TLS protocol versions.

**Additional resources**

- For further details about the system-wide crypto policy, see the `update-crypto-policies(8)` man page and Using system-wide cryptographic policies.

- For further details about the `SSLProtocol` parameter, refer to the `mod_ssl` documentation in the Apache manual. For details about installing the manual, see Installing the Apache HTTP Server manual.

**27.9. Setting the Supported Ciphers on an Apache HTTP Server**

By default, the Apache HTTP Server on RHEL 8 uses the system-wide crypto policy that defines safe default values, which are also compatible with recent browsers. For the list of ciphers the system-wide crypto allows, see the `/etc/crypto-policies/back-ends/openssl.config` file.

This section describes how to manually configure which ciphers the `my_company.idm.example.com` Apache HTTP server supports. Follow the procedure if your environment requires specific ciphers.

**Prerequisites**

- TLS encryption is enabled on the `my_company.idm.example.com` server as described in Section 27.7, “Adding TLS encryption to an Apache HTTP Server”.
Procedure

1. Edit the /etc/httpd/conf/httpd.conf file, and add the **SSLCipherSuite** parameter to the `<VirtualHost>` directive for which you want to set the TLS ciphers:

   ```
   SSLCipherSuite
   "EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH:!SHA1:!SHA256"
   ```

   This example enables only the **EECDH+AESGCM**, **EDH+AESGCM**, **AES256+EECDH**, and **AES256+EDH** ciphers and disables all ciphers which use the **SHA1** and **SHA256** message authentication code (MAC).

2. Restart the **httpd** service:

   ```
   # systemctl restart httpd
   ```

Verification steps

1. To display the list of ciphers the Apache HTTP Server supports:
   a. Install the **nmap** package:

      ```
      # yum install nmap
      ```
   b. Use the **nmap** utility to display the supported ciphers:

      ```
      # nmap --script ssl-enum-ciphers -p 443 example.com
      ```

      ... PORT STATE SERVICE
      443/tcp open  https
      | ssl-enum-ciphers:
      |   TLSv1.2:
      |     ciphers:
      |     |   TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (ecdh_x25519) - A
      |     |   TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (dh 2048) - A
      |     |   TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (ecdh_x25519) - A
      |   TLSv1.3:
      |     ciphers:
      |     |   TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (ecdh_x25519) - A
      |     |   TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (ecdh_x25519) - A
      ```

Additional resources

- For further details about the system-wide crypto policy, see the `update-crypto-policies(8)` man page and [Using system-wide cryptographic policies](#).

- For further details about the **SSLCipherSuite** parameter, refer to the **mod_ssl** documentation in the Apache manual. For details about installing the manual, see [Installing the Apache HTTP Server manual](#).

### 27.10. CONFIGURING TLS CLIENT CERTIFICATE AUTHENTICATION

Client certificate authentication enables administrators to allow only users who authenticate using a certificate to access resources on the **my_company.idm.example.com** web server. This section describes how to configure client certificate authentication for the /var/www/html/Example/ directory.
IMPORTANT

If the my_company.idm.example.com Apache server uses the TLS 1.3 protocol, certain clients require additional configuration. For example, in Firefox, set the security.tls.enable_post_handshake_auth parameter in the about:config menu to true. For further details, see Transport Layer Security version 1.3 in Red Hat Enterprise Linux 8.

Prerequisites

- TLS encryption is enabled on the my_company.idm.example.com server as described in Section 27.7, “Adding TLS encryption to an Apache HTTP Server”.

Procedure

1. Edit the /etc/httpd/conf/httpd.conf file and add the following settings to the <VirtualHost> directive for which you want to configure client authentication:

   <Directory "/var/www/html/Example/">
   SSLVerifyClient require
   </Directory>

   The SSLVerifyClient require setting defines that the server must successfully validate the client certificate before the client can access the content in the /var/www/html/Example/ directory.

2. Restart the httpd service:

   # systemctl restart httpd

Verification steps

1. Use the curl utility to access the https://my_company.idm.example.com/Example/ URL without client authentication:

   $ curl https://my_company.idm.example.com/Example/
   curl: (56) OpenSSL SSL_read: error:1409445C:SSL routines:ssl3_read_bytes:tlsv13 alert certificate required, errno 0

   The error indicates that the my_company.idm.example.com web server requires a client certificate authentication.

2. Pass the client private key and certificate, as well as the CA certificate to curl to access the same URL with client authentication:

   $ curl --cacert ca.crt --key client.key --cert client.crt
   https://my_company.idm.example.com/Example/

   If the request succeeds, curl displays the index.html file stored in the /var/www/html/Example/ directory.

Additional resources

For further details about client authentication, see the mod_ssl Configuration How-To Red Hat Enterprise Linux 8 Configuring and managing Identity Management 200
For further details about client authentication, see the mod_ssl Configuration How-To documentation in the Apache manual. For details about installing the manual, see Installing the Apache HTTP Server manual.

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

As an Identity Management (IdM) administrator, you can configure a web server running on an IdM client to request users that use web browsers to access the server to authenticate with certificates issued by a specific IdM sub-CA. Complete this section to request a user certificate from a specific IdM sub-CA and to export the certificate and the corresponding private key on to the host from which the user wants to access the web server using a web browser. Afterwards, import the certificate and the private key into the browser.

### 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:

   ```bash
   # 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`:

   ```bash
   # 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:

   ```bash
   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. Specify the IdM sub-CA that you want to issue the certificate. 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 from webclient-ca, and save the certificate in the ~/.idm_user.pem file:

```bash
# ipa cert-request certificate_request.csr --principal=idm_user@IDM.EXAMPLE.COM --profile-id=IECUserRoles --ca=webclient-ca --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:

```bash
# 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:

```bash
# certutil -K -d ~/certdb/
< 0> rsa 5ad14d41463b87a095b1896cf0068ccc467df395  NSS Certificate DB: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:

```bash
# 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:

```bash
# 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:

```bash
# chmod o-rwx /home/idm_user/
```

10. For security reasons, remove the temporary NSS database and the .pkcs12 file from the server:

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

### 27.12. Configuring a Browser to Enable Certificate Authentication

---

202
To be able to authenticate with a certificate when using the WebUI to log into Identity Management (IdM), you need to import the user and the relevant certificate authority (CA) certificates into the Mozilla Firefox or Google Chrome browser. The host itself on which the browser is running does not have to be part of the IdM domain.

IdM supports the following browsers for connecting to the WebUI:

- 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**

- You have the user certificate that you want to import to the browser at your disposal in the PKCS#12 format.
- You have downloaded the sub-CA certificate and have it at your disposal in the PEM format.

**Procedure**

1. Open Firefox, then navigate to Preferences → Privacy & Security.

   **Figure 27.7. Privacy and Security section in Preferences**

   - General
   - Search
   - Privacy & Security

   - Startup
     - Always check if Firefox is your default browser
     - Firefox is not your default browser [Make Default...]

   - When Firefox starts
     - Show your home page
     - Show a blank page
     - Show your windows and tabs from last time

2. Click View Certificates.

   **Figure 27.8. 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. To make sure that your IdM sub-CA is recognized by Firefox as a trusted authority, import the
IdM sub-CA certificate that you saved in Section 27.2, “Downloading the sub-CA certificate
from IdM WebUI” as a trusted certificate authority certificate:

   a. Open Firefox, navigate to Preferences and click **Privacy & Security**.

   Figure 27.9. Privacy and Security section in Preferences

   ![Privacy and Security section in Preferences](image)

   b. Click **View Certificates**.

   Figure 27.10. View Certificates in Privacy and Security

   ![View Certificates in Privacy and Security](image)

   c. In the **Authorities** tab, click **Import**. Locate and open the sub-CA certificate. Trust the
certificate to identify websites, then click **OK** and **OK**.
As a system administrator, if you want to be able to invalidate a specific group of related certificates quickly:

- Design your applications so that they only trust certificates that were issued by a specific lightweight Identity Management (IdM) sub-CA. Afterwards, you will be able to invalidate all these certificates by only revoking the certificate of the Identity Management (IdM) sub-CA that issued these certificates. For details on how to create and use a lightweight sub-CA in IdM, see Chapter 27, Restricting an application to trust only a subset of certificates.

- Ensure that all the certificates that have been issued by the to-be-revoked IdM sub-CA are immediately invalid, configure applications that rely on such certificates to use the IdM OCSP responders. For example, to configure the Firefox browser to use OCSP responders, make sure that the Query OCSP responder servers to confirm the current validity of certificates checkbox is checked in Firefox Preferences.

In IdM, the certificate revocation list (CRL) is updated every four hours.

To invalidate all the certificates issued by an IdM sub-CA, revoke the IdM sub-CA certificate. For details, see Section 18.2, “Revoking certificates with the integrated IdM CAs”. In addition, disable the relevant CA ACLs.

### 28.1. DISABLING CA ACLS IN IDM CLI

When you want to retire an IdM service or a group of IdM services, consider disabling any existing corresponding CA ACLs.

Complete this section to disable the `TLS_web_server_authentication` CA ACL that restricts the web server running on your IdM client to request a certificate to be issued by the `webserver-ca` IdM sub-CA, and to disable the `TLS_web_client_authentication` CA ACL that restricts IdM users to request a user certificate to be issued by the `webclient-ca` IdM sub-CA.

**Procedure**

1. Optionally, to view all the CA ACLs in your IdM environment, enter the `ipa caacl-find` command:

   ```
   $ ipa caacl-find
   ---------------
   3 CA ACLs matched
   ---------------
   ACL name: hosts_services_caliPAserviceCert
   Enabled: TRUE

   ACL name: TLS_web_server_authentication
   Enabled: TRUE

   ACL name: TLS_web_client_authentication
   Enabled: TRUE
   ```

2. Optionally, to view the details of a CA ACL, enter the `ipa caacl-show` command, and specify the CA ACL name:

   ```
   $ ipa caacl-show TLS_web_server_authentication
   ```
ACL name: TLS_web_server_authentication
Description: CAACL for web servers authenticating to web clients using certificates issued by webserver-ca
Enabled: TRUE
CAs: webserver-ca
Profiles: calIPAserviceCert
Services: HTTP/rhel8server.idm.example.com@IDM.EXAMPLE.COM

3. To disable a CA ACL, enter the `ipa caacl-disable` command, and specify the CA ACL name.

- To disable the `TLS_web_server_authentication` CA ACL, enter:

  ```
  $ ipa caacl-disable TLS_web_server_authentication
  Disabled CA ACL "TLS_web_server_authentication"
  ```

- To disable the `TLS_web_client_authentication` CA ACL, enter:

  ```
  $ ipa caacl-disable TLS_web_client_authentication
  Disabled CA ACL "TLS_web_client_authentication"
  ```

The only enabled CA ACL now is the `hosts_services_calIPAserviceCert` CA ACL.

**IMPORTANT**

Be extremely careful about disabling the `hosts_services_calIPAserviceCert` CA ACL. Disabling `hosts_services_calIPAserviceCert`, without another CA ACL granting IdM servers use of the `ipa` CA with the `calIPAserviceCert` profile means that certificate renewal of the IdM HTTP and LDAP certificates will fail. The expired IdM HTTP and LDAP certificates will eventually cause IdM system failure.
29.1. ID OVERRIDES FOR AD USERS

In Red Hat Enterprise Linux (RHEL) 7, external group membership allows Active Directory (AD) users and groups to access Identity Management (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.

29.2. USING ID OVERRIDES TO ENABLE AD USERS TO ADMINISTER IDM

Prerequisites

- The idm:DL1 stream is enabled on your Identity Management (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 (AD), including the ipa-idoverride-memberof package.

- A working IdM environment is set up. For details, see Installing Identity Management.

- A working trust between your IdM environment and AD is set up.

Procedure

This procedure describes creating and using an ID override for an AD user to give that user rights identical to those of an 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
```

### 29.3. MANAGING IDM CLI AS AN AD USER

This procedure checks that an Active Directory (AD) user can log into Identity Management (IdM) command-line interface (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
```
DNS canonicalization is disabled by default on Identity Management (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.

### 30.1. ADDING AN ALIAS TO A HOST PRINCIPAL

By default, Identity Management (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 30.2, “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.example.com host principal:

   ```
   $ ipa host-add-principal demo.example.com --principal=demo
   ```

### 30.2. 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 30.1, “Adding an alias to a host principal”, you do not need to enable canonicalization.

**Prerequisites**

- The Identity Management (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
   ```

30.3. 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 30.2, “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 Identity Management (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 AD.
CHAPTER 31. COLLECTING IDM HEALTHCHECK INFORMATION

Healthcheck has been designed as a manual command line tool which should help you to identify possible problems in Identity Management (IdM).

This chapter describes how you can create a collection of logs based on the Healthcheck output with 30-day rotation.

Prerequisites

- The Healthcheck tool is only available on RHEL 8.1 or newer

31.1. HEALTHCHECK IN IDM

The Healthcheck tool in Identity Management (IdM) helps find issues that may impact the health of your IdM environment.

NOTE

The Healthcheck tool is a command line tool that can be used without Kerberos authentication.

31.1.1. Modules are Independent

Healthcheck consists of independent modules which test for:

- Replication issues
- Certificate validity
- Certificate Authority infrastructure issues
- IdM and Active Directory trust issues
- Correct file permissions and ownership settings

31.1.2. Two output formats

Healthcheck generates the following outputs:

- Human-readable output
- Machine-readable output in JSON format

The output destination for both human and JSON is standard output by default. You can specify a different destination with the --output-file option.

31.1.3. Results

Each Healthcheck module returns one of the following results:

SUCCESS
configured as expected

**WARNING**
not an error, but worth keeping an eye on or evaluating

**ERROR**
not configured as expected

**CRITICAL**
not configured as expected, with a high possibility for impact

### 31.1.4. Running IdM Healthcheck

Healthcheck can be run:

- Manually
  ```
  [root@master ~]# ipa-healthcheck
  ```
  For all options, see the man page: `man ipa-healthcheck`.

- Automatically using log rotation.

### 31.2. LOG ROTATION

Log rotation creates a new log file every day, and the files are organized by date. Since log files are saved in the same directory, you can select a particular log file according to the date.

Rotation means that there is configured a number for max number of log files and if the number is exceeded, the newest file rewrites and renames the oldest one. For example, if the rotation number is 30, the thirty-first log file replaces the first (oldest) one.

Log rotation reduces voluminous log files and organizes them, which can help with analysis of the logs.

### 31.3. CONFIGURING LOG ROTATION USING THE IDM HEALTHCHECK

This section describes how to configure a log rotation with:

- the **systemd** timer
- the **crond** service

The **systemd** timer runs the Healthcheck tool periodically and generates the logs. The default value is set to 4 am every day.

The **crond** service is used for log rotation.

The default log name is **healthcheck.log** and the rotated logs use the **healthcheck.log-YYYYMMDD** format.

**Prerequisites**

- You must execute commands as root.

**Procedure**
1. Enable a `systemd` timer:

   ```
   # systemctl enable ipa-healthcheck.timer
   Created symlink /etc/systemd/system/multi-user.target.wants/ipa-healthcheck.timer ->
   /usr/lib/systemd/system/ipa-healthcheck.timer.
   ```

2. Start the `systemd` timer:

   ```
   # systemctl start ipa-healthcheck.timer
   ```

3. Open the `/etc/logrotate.d/ipahealthcheck` file to configure the number of logs which should be saved.
   By default, log rotation is set up for 30 days.

4. In the `/etc/logrotate.d/ipahealthcheck` file, configure the path to the logs.
   By default, logs are saved in the `/var/log/ipa/healthcheck/` directory.

5. In the `/etc/logrotate.d/ipahealthcheck` file, configure the time for log generation.
   By default, a log is created daily at 4 AM.

6. To use log rotation, ensure that the `crond` service is enabled and running:

   ```
   # systemctl enable crond
   # systemctl start crond
   ```

   To start with generating logs, start the IPA healthcheck service:

   ```
   # systemctl start ipa-healthcheck
   ```

   To verify the result, go to `/var/log/ipa/healthcheck/` and check if logs are created correctly.
CHAPTER 32. CHECKING SERVICES USING IDM

HEALTHCHECK

This section describes monitoring services used by the Identity Management (IdM) server using the Healthcheck tool.

For details, see Healthcheck in IdM.

Prerequisites

- The Healthcheck tool is only available on RHEL 8.1 and newer
- Enable Healtheck on the IdM server:

```
# systemctl start ipa-healthcheck
```

32.1. SERVICES HEALTHCHECK TEST

The Healthcheck tool includes a test to check if any IdM services is not running. This test is important because services which are not running can cause failures in other tests. Therefore, check that all services are running first. You can then check all other test results.

To see all services tests, run ipa-healthcheck with the --list-sources option:

```
# ipa-healthcheck --list-sources
```

You can find all services tested with Healthcheck under the ipahealthcheck.meta.services source:

- certmonger
- dirsrv
- gssproxy
- httpd
- ipa_custodia
- ipa_dnskeysyncd
- ipa_otp
- kadmin
- krb5kdc
- named
- pki_tomcatd
- sssd
NOTE
Run these tests on all IdM master servers when trying to discover issues.

32.2. SCREENING SERVICES USING HEALTHCHECK

This section describes a standalone manual test of services running on the Identity Management (IdM) server using the Healthcheck tool.

The Healthcheck tool includes many tests, whose results can be shortened with:

- excluding all successful test: `--failures-only`
- including only services tests: `--source=ipahealthcheck.meta.services`

Procedure

- To run Healthcheck with warnings, errors and critical issues regarding services, enter:

  ```shell
  # ipa-healthcheck --source=ipahealthcheck.meta.services --failures-only
  ```

A successful test displays empty brackets:

```
[
]
```

If one of the services fails, the result can looks similarly to this example:

```
{
    "source": "ipahealthcheck.meta.services",
    "check": "httpd",
    "result": "ERROR",
    "kw": {
        "status": false,
        "msg": "httpd: not running"
    }
}
```

Additional resources

- For reviewing detailed reference, enter `man ipa-healthcheck` in the command line.
CHAPTER 33. VERIFYING YOUR IDM AND AD TRUST CONFIGURATION USING IDM HEALTHCHECK

This section helps you understand and use the Healthcheck tool in Identity management (IdM) to identify issues with IdM and an Active Directory trust.

For details, see Section 31.1, “Healthcheck in IdM”.

Prerequisites

- The Healthcheck tool is only available on RHEL 8.1 or newer
- Enable Healtheck on the IdM server:

  # systemctl start ipa-healthcheck

33.1. IDM AND AD TRUST HEALTHCHECK TESTS

The Healthcheck tool includes several tests for testing the status of your Identity Management (IdM) and Active Directory (AD) trust.

To see all trust tests, run ipa-healthcheck with the --list-sources option:

  # ipa-healthcheck --list-sources

You can find all tests under the ipahealthcheck.ipa.trust source:

IPATrustAgentCheck

This test checks the SSSD configuration when the machine is configured as a trust agent. For each domain in /etc/sssdist/sssdist.conf where id_provider=ipa ensure that ipa_server_mode is True.

IPATrustDomainsCheck

This test checks if the trust domains match SSSD domains by comparing the list of domains in sssctl domain-list with the list of domains from ipa trust-find excluding the IPA domain.

IPATrustCatalogCheck

This test resolves resolves an AD user, Administrator@REALM. This populates the AD Global catalog and AD Domain Controller values in sssctl domain-status output.

For each trust domain look up the user with the id of the SID + 500 (the administrator) and then check the output of sssctl domain-status <domain> --active-server to ensure that the domain is active.

IPAsidgenpluginCheck

This test verifies that the sidgen plugin is enabled in the IPA 389-ds instance. The test also verifies that the IPA SIDGEN and ipa-sidgen-task plugins in cn=plugins,cn=config include the nsslapd-pluginEnabled option.

IPATrustAgentMemberCheck

This test verifies that the current host is a member of cn=adtrust agents,cn=sysaccounts,cn=etc,SUFFIX.

IPATrustControllerPrincipalCheck

This test verifies that the current host is a member of cn=adtrust agents,cn=sysaccounts,cn=etc,SUFFIX.
IPATrustControllerServiceCheck
This test verifies that the current host starts the ADTRUST service in ipactl.

IPATrustControllerConfCheck
This test verifies that ldapi is enabled for the passdb backend in the output of net conf list.

IPATrustControllerGroupSIDCheck
This test verifies that the admins group’s SID ends with 512 (Domain Admins RID).

IPATrustPackageCheck
This test verifies that the trust-ad package is installed if the trust controller and AD trust are not enabled.

NOTE
Run these tests on all IdM master servers when trying to find an issue.

33.2. SCREENING THE TRUST WITH THE HEALTHCHECK TOOL

This section describes a standalone manual test of an Identity Management (IdM) and Active Directory (AD) trust health check using the Healthcheck tool.

The Healthcheck tool includes many tests, therefore, you can shorten the results by:

- excluding all successful test: --failures-only
- including only trust tests: --source=ipahealthcheck.ipa.trust

Procedure

- To run Healthcheck with warnings, errors and critical issues in the trust, enter:
  
  ```bash
  # ipa-healthcheck --source=ipahealthcheck.ipa.trust --failures-only
  ```

Successful test displays empty brackets:

  ```bash
  # ipa-healthcheck --source=ipahealthcheck.ipa.trust --failures-only
  []
  ```

Additional resources

- For reviewing detailed reference, enter man ipa-healthcheck in the command line.
CHAPTER 34. VERIFYING CERTIFICATES USING IDM HEALTHCHECK

This section helps in understanding and using the Healthcheck tool in Identity management (IdM) to identify issues with IPA certificates maintained by certmonger.

For details, see Healthcheck in IdM.

Prerequisites

- The Healthcheck tool is only available in RHEL 8.1 and newer.
- Enable Healtheck on the IdM server:

  # systemctl start ipa-healthcheck

34.1. IDM CERTIFICATES HEALTHCHECK TESTS

The Healthcheck tool includes several tests for verifying the status of certificates maintained by certmonger in Identity Management (IdM). For details about certmonger, see Obtaining an IdM certificate for a service using certmonger.

This suite of tests checks expiration, validation, trust and other issues. Multiple errors may be thrown for the same underlying issue.

To see all certificate tests, run the `ipa-healthcheck` with the `--list-sources` option:

# ipa-healthcheck --list-sources

You can find all tests under the `ipahealthcheck.ipa.certs` source:

IPACertmongerExpirationCheck

This test checks expirations in certmonger. If an error is reported, the certificate has expired.

If a warning appears, the certificate will expire soon. By default, this test applies within 28 days or fewer days before certificate expiration.

You can configure the number of days in the `/etc/ipahealthcheck/ipahealthcheck.conf` file. After opening the file, change the `cert_expiration_days` option located in the default section.

NOTE

Certmonger loads and maintains its own view of the certificate expiration. This check does not validate the on-disk certificate.

IPACertfileExpirationCheck

This test checks if the certificate file or NSS database cannot be opened. This test also checks expiration. Therefore, carefully read the `msg` attribute in the error or warning output. The message specifies the problem.
NOTE

This test checks the on-disk certificate. If a certificate is missing, unreadable, etc a separate error can also be raised.

IPACertNSSTrust

This test compares the trust for certificates stored in NSS databases. For the expected tracked certificates in NSS databases the trust is compared to an expected value and an error raised on a non-match.

IPANSSChainValidation

This test validates the certificate chain of the NSS certificates. The test executes: `certutil -V -u V -e -d [dbdir] -n [nickname]`

IPAOpenSSLChainValidation

This test validates the certificate chain of the OpenSSL certificates. To be comparable to the NSSChain validation here is the OpenSSL command we execute:

```bash
openssl verify -verbose -show_chain -CAfile /etc/ipa/ca.crt [cert file]
```

IPARAAgent

This test compares the certificate on disk with the equivalent record in LDAP in uid=ipara,ou=People,o=ipaca.

IPACertRevocation

This test uses certmonger to verify that certificates have not been revoked. Therefore, the test can find issues connected with certificates maintained by certmonger only.

IPACertmongerCA

This test verifies the certmonger Certificate Authority (CA) configuration. IdM cannot issue certificates without CA.

Certmonger maintains a set of CA helpers. In IdM, there is a CA named IPA which issues certificates through IdM, authenticating as a host or user principal, for host or service certs.

There are also `dogtag-ipa-ca-renew-agent` and `dogtag-ipa-ca-renew-agent-reuse` which renew the CA subsystem certificates.

NOTE

Run these tests on all IdM master servers when trying to check for issues.

34.2. SCREENING CERTIFICATES USING THE HEALTHCHECK TOOL

This section describes a standalone manual test of an Identity Management (IdM) certificate health check using the Healthcheck tool.

The Healthcheck tool includes many tests, therefore, you can shorten the results with:

- excluding all successful test: `--failures-only`
- including only certificate tests: `--source=ipahealthcheck.ipa.certs`

Prerequisites
Healthcheck tests must be performed as the root user.

Procedure

- To run Healthcheck with warnings, errors and critical issues regarding certificates, enter:
  
  ```
  # ipa-healthcheck --source=ipahealthcheck.ipa.certs --failures-only
  ```

Successful test displays empty brackets:

```
[]
```

Failed test shows you the following output:

```
{
  "source": "ipahealthcheck.ipa.certs",
  "check": "IPACertfileExpirationCheck",
  "result": "ERROR",
  "kw": {
    "key": 1234,
    "dbdir": "/path/to/nssdb",
    "error": [error],
    "msg": "Unable to open NSS database '/path/to/nssdb': [error]"
  }
}
```

This **IPACertfileExpirationCheck** test failed on opening the NSS database.

Additional resources

- For reviewing detailed reference, enter `man ipa-healthcheck` in the command line.
CHAPTER 35. VERIFYING SYSTEM CERTIFICATES USING IDM HEALTHCHECK

This section describes a Healthcheck tool in Identity Management (IdM) to identify issues with system certificates.

For details, see Healthcheck in IdM.

Prerequisites

- The Healthcheck tool is only available on RHEL 8.1 or newer.
- Enable Healtheck on the IdM server:

  # systemctl start ipa-healthcheck

35.1. SYSTEM CERTIFICATES HEALTHCHECK TESTS

The Healthcheck tool includes several tests for verifying system (DogTag) certificates.

To see all tests, run the ipa-healthcheck with the --list-sources option:

  # ipa-healthcheck --list-sources

You can find all tests under the ipahealthcheck.dogtag.ca source:

**DogtagCertsConfigCheck**

This test compares the CA (Certificate Authority) certificates in its NSS database to the same values stored in CS.cfg. If they don’t match, CA fails to start.

Specifically, it checks:

- auditSigningCert cert-pki-ca against ca.audit_signing.cert
- ocspSigningCert cert-pki-ca against ca.ocsp_signing.cert
- caSigningCert cert-pki-ca against ca.signing.cert
- subsystemCert cert-pki-ca against ca.subsystem.cert
- Server-Cert cert-pki-ca against ca.sslserver.cert

If Key Recovery Authority (KRA) is installed:

- transportCert cert-pki-kra against ca.connector.KRA.transportCert

**DogtagCertsConnectivityCheck**

This test verifies connectivity. This test is equivalent to the ipa cert-show 1 command which checks:

- The PKI proxy configuration in Apache
- IdM being able to find a CA
- The RA agent client certificate
Correctness of CA replies to requests

Note that the test checks a certificate with serial #1 because you want to verify that a `cert-show` can be executed and get back an expected result from CA (either the certificate or a not found).

NOTE

Run these tests on all IdM master servers when trying to find an issue.

35.2. SCREENING SYSTEM CERTIFICATES USING HEALTHCHECK

This section describes a standalone manual test of Identity Management (IdM) certificates using the Healthcheck tool.

Since, the Healthcheck tool includes many tests, you can narrow the results by including only DogTag tests: `--source=ipahealthcheck.dogtag.ca`

Procedure

- To run Healthcheck restricted to DogTag certificates, enter:

  ```
  # ipa-healthcheck --source=ipahealthcheck.dogtag.ca
  ```

An example of a successful test:

```json
{
    "source": "ipahealthcheck.dogtag.ca",
    "check": "DogtagCertsConfigCheck",
    "result": "SUCCESS",
    "uuid": "9b366200-9ec8-4bd9-bb5e-9a280c803a9c",
    "when": "20191008135826Z",
    "duration": 0.252280,
    "kw": {
        "key": "Server-Cert cert-pki-ca",
        "configfile": "/var/lib/pki/pki-tomcat/conf/ca/CS.cfg"
    }
}
```

An example of a failed test:

```json
{
    "source": "ipahealthcheck.dogtag.ca",
    "check": "DogtagCertsConfigCheck",
    "result": "CRITICAL",
    "uuid": "59d66200-1447-4b3b-be01-89810c803a98",
    "when": "20191008135912Z",
    "duration": 0.002022,
    "kw": {
        "exception": "NSDB /etc/pki/pki-tomcat/alias not initialized",
    }
}
```

Additional resources
• For reviewing detailed reference, enter `man ipa-healthcheck` in the command line.
CHAPTER 36. CHECKING DISK SPACE USING IDM HEALTHCHECK

This section describes how to monitor the Identity Management server’s free disk space using the Healthcheck tool.

For details, see Healthcheck in IdM.

Prerequisites

- The Healthcheck tool is only available on RHEL 8.1 and newer.
- Enable Healtheck on the IdM server:

  # systemctl start ipa-healthcheck

36.1. DISK SPACE HEALTHCHECK TEST

The Healthcheck tool includes a test for checking available disk space. Insufficient free disk space can cause issues with:

- Logging
- Execution
- Backups

The test checks the following paths:

Table 36.1. Tested paths

<table>
<thead>
<tr>
<th>Paths checked by the test</th>
<th>Minimal disk space in MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/lib/dirsrv/</td>
<td>1024</td>
</tr>
<tr>
<td>/var/lib/ipa/backup/</td>
<td>512</td>
</tr>
<tr>
<td>/var/log/</td>
<td>1024</td>
</tr>
<tr>
<td>var/log/audit/</td>
<td>512</td>
</tr>
<tr>
<td>/var/tmp/</td>
<td>512</td>
</tr>
<tr>
<td>/tmp/</td>
<td>512</td>
</tr>
</tbody>
</table>

To list all tests, run the `ipa-healthcheck` with the `--list-sources` option:

# ipa-healthcheck --list-sources

The file system space check test is placed under the `ipahealthcheck.system.filesystemspace` source:
FileSystemSpaceCheck

This test checks available disk space in the following ways:

- The minimum raw free bytes needed.
- The percentage – the minimum free disk space is hardcoded to 20%.

### 36.2. SCREENING DISK SPACE USING THE HEALTHCHECK TOOL

This section describes a standalone manual test of available disk space on an Identity Management (IdM) server using the Healthcheck tool.

Since Healthcheck includes many tests, you can narrow the results by:

- excluding all successful test: `--failures-only`
- including only space check tests: `--source=ipahealthcheck.system.filesystemspace`

#### Procedure

To run Healthcheck with warnings, errors and critical issues regarding available disk space, enter:

```bash
# ipa-healthcheck --source=ipahealthcheck.system.filesystemspace --failures-only
```

A successful test displays empty brackets:

```
[]
```

As an example, a failed test can display:

```json
{
  "source": "ipahealthcheck.system.filesystemspace",
  "check": "FileSystemSpaceCheck",
  "result": "ERROR",
  "kw": {
    "msg": "/var/lib/dirsrv: free space under threshold: 0 MiB < 1024 MiB",
    "store": "/var/lib/dirsrv",
    "free_space": 0,
    "threshold": 1024
  }
}
```

The failed test informs you that the `/var/lib/dirsrv` directory has run out of space.

#### Additional resources

- For reviewing detailed reference, enter `man ipa-healthcheck` in the command line.
CHAPTER 37. VERIFYING PERMISSIONS OF IDM CONFIGURATION FILES USING HEALTHCHECK

This section describes how to test Identity Management (IdM) configuration files using the Healthcheck tool.

For details, see Healthcheck in IdM.

Prerequisites

- The Healthcheck tool is only available on RHEL 8.1 or newer systems.
- Enable Healtech on the IdM server:
  
  # systemctl start ipa-healthcheck

37.1. FILE PERMISSIONS HEALTHCHECK TESTS

The Healthcheck tool tests ownership and permissions of some important files installed or configured by Identity Management (IdM).

If you change the ownership or permissions of any tested file, the test returns a warning in the result section. While it does not necessarily mean that the configuration will not work, it means that the file differs from the default configuration.

To see all tests, run the ipa-healthcheck with the --list-sources option:

# ipa-healthcheck --list-sources

The file permissions test is placed under the ipahealthcheck.ipa.files source:

IPAFileNSSDBCheck

This test checks the 389-ds NSS database and the Certificate Authority (CA) database. The 389-ds database is located in /etc/dirsrv/slapd-<dashed-REALM> and the CA database is located in /etc/pki/pki-tomcat/alias/.

IPAFileCheck

This test checks the following files:

- /var/lib/ipa/ra-agent.{key|pem}
- /var/lib/ipa/certs/httpd.pem
- /var/lib/ipa/private/httpd.key
- /etc/httpd/alias/ipasession.key
- /etc/dirsrv/ds.keytab
- /etc/ipa/ca.crt
- /etc/ipa/custodia/server.keys

If PKINIT is enabled:
• /var/lib/ipa/certs/kdc.pem
• /var/lib/ipa/private/kdc.key
  If DNS is configured:
• /etc/named.keytab
• /etc/ipa/dnssec/ipa-dnskeysyncd.keytab

TomcatFileCheck
This test checks some tomcat-specific files if a CA is configured:
• /etc/pki/pki-tomcat/password.conf
• /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
• /etc/pki/pki-tomcat/server.xml

NOTE
Run these tests on all IdM master servers when trying to find issues.

37.2. SCREENING CONFIGURATION FILES USING HEALTHCHECK

This section describes a standalone manual test of an Identity Management (IdM) server’s configuration files using the Healthcheck tool.

The Healthcheck tool includes many tests. Results can be narrowed down by:
- excluding all successful test: --failures-only
- including only ownership and permissions tests: --source=ipahealthcheck.ipa.files

Procedure

1. To run Healthcheck tests on IdM configuration file ownership and permissions, while displaying only warnings, errors and critical issues, enter:

   # ipa-healthcheck --source=ipahealthcheck.ipa.files --failures-only

A successful test displays empty brackets:

   # ipa-healthcheck --source=ipahealthcheck.ipa.files --failures-only
   {}

Failed tests display results similar to the following WARNING:

   {
   "source": "ipahealthcheck.ipa.files",
   "check": "IPAFileNSSDBCheck",
   "result": "WARNING",
   "kw": { 
   "key": ".etc_dirsrv_slapd-EXAMPLE-TEST_pkcs11.txt_mode",
   }
"path": "/etc/dirsrv/slapd-EXAMPLE-TEST/pkcs11.txt",
"type": "mode",
"expected": "0640",
"got": "0666",
"msg": "Permissions of /etc/dirsrv/slapd-EXAMPLE-TEST/pkcs11.txt are 0666 and should be 0640"
}
}

Additional resources

- For reviewing detailed reference material, open `man ipa-healthcheck` in the command line.
CHAPTER 38. CHECKING IDM REPLICATION USING HEALTHCHECK

This section describes how to test Identity Management (IdM) replication using the Healthcheck tool.

For details, see Healthcheck in IdM.

Prerequisites

- The Healthcheck tool is only available on RHEL 8.1 or newer.
- Enable Healtheck on the IdM server:

  ```
  # systemctl start ipa-healthcheck
  ```

38.1. REPLICATION HEALTHCHECK TESTS

The Healthcheck tool tests the Identity Management (IdM) topology configuration and searches for replication conflict issues.

To list all tests, run the `ipa-healthcheck` with the `--list-sources` option:

  ```
  # ipa-healthcheck --list-sources
  ```

The topology tests are placed under the `ipahealthcheck.ipa.topology` and `ipahealthcheck.ds.replication` sources:

- **IPATopologyDomainCheck**

  This test verifies:

  - whether topology is not disconnected and there are replication paths between all servers.
  - if servers don’t have more than the recommended number of replication agreements.

  If the test fails, the test returns errors, such as connection errors or too many replication agreements.

  If the test succeeds, the test returns the configured domains.

  **NOTE**

  The test runs the `ipa topologysuffix-verify` command for both the domain and ca suffixes (assuming the Certificate Authority is configured on this master).

- **ReplicationConflictCheck**

  The test searches for entries in LDAP matching `(&(!(objectclass=nstombstone)) (nsds5ReplConflict=*))`.

  **NOTE**

  Run these tests on all IdM master servers when trying to check for issues.
38.2. SCREENING REPLICATION USING HEALTHCHECK

This section describes a standalone manual test of an Identity Management (IdM) replication topology and configuration using the Healthcheck tool.

The Healthcheck tool includes many tests, therefore, you can shorten the results with:

- Replication conflict test: `--source=ipahealthcheck.ds.replication`
- Correct topology test: `--source=ipahealthcheck.ipa.topology`

Prerequisites

- Healthcheck tests must be performed as the root user.

Procedure

- To run Healthcheck replication conflict and topology checks, enter:

  ```
  # ipa-healthcheck --source=ipahealthcheck.ds.replication --
  source=ipahealthcheck.ipa.topology
  ```

  Four different results are possible:

- **SUCCESS** – the test passed successfully.

  ```json
  {   "source": "ipahealthcheck.ipa.topology",   "check": "IPATopologyDomainCheck",   "result": "SUCCESS",   "kw": {     "suffix": "domain"   } }
  ```

- **WARNING** – the test passed but there might be a problem.

  ```json
  {   "source": "ipahealthcheck.ipa.topology",   "check": "IPATopologyDomainCheck",   "result": "ERROR",   "uuid": d6ce3332-92da-423d-9818-e79f49ed321f   "when": 20191007115449Z   "duration": 0.005943   "kw": {     "msg": "topologysuffix-verify domain failed, server2 is not connected (server2_139664377356472 in MainThread)"   } }
  ```

- **ERROR** – the test failed.

- **CRITICAL** – the test failed and it affects the IdM server functionality.

Additional resources
• For reviewing detailed reference, open `man ipa-healthcheck` in the command line.
CHAPTER 39. DEMOTING OR PROMOTING HIDDEN REPLICA

After a replica has been installed, you can change whether the replica is hidden or visible.

For details about hidden replicas, see The hidden replica mode.

If the replica is a CA renewal master, move the service to another replica. For details, see Changing and resetting IdM CA Renewal Master.

NOTE

The hidden replica feature is available in Red Hat Enterprise Linux 8.1 and later as a Technology Preview and, therefore, not supported.

Procedure

- To hide the replica, enter:

```
# ipa server-state replica.idm.example.com --state=hidden
```

Alternatively, you can make the replica visible with the following command

```
# ipa server-state replica.idm.example.com --state=enabled
```
CHAPTER 40. SETTING UP SAMBA ON AN IDM DOMAIN MEMBER

This section describes how to set up Samba on a host that is joined to a Red Hat Identity Management (IdM) domain. Users from IdM and also, if available, from trusted Active Directory (AD) domains, can access shares and printer services provided by Samba.

**Important**

Using Samba on an IdM domain member is an unsupported Technology Preview feature.

**Prerequisites**

- The host is joined as a client to the IdM domain.
- Both the IdM servers and the client must run on RHEL 8.1 or later.

**40.1. PREPARING THE IDM DOMAIN FOR INSTALLING SAMBA ON DOMAIN MEMBERS**

Before you can establish a trust with AD and if you want to set up Samba on an IdM client, you must prepare the IdM domain using the `ipa-adtrust-install` utility on an IdM server. However, even if both situations apply, you must run `ipa-adtrust-install` only once on an IdM master.

**Prerequisites**

- IdM is installed.

**Procedure**

1. Install the required packages:

```
[root@ipaserver ~]# yum install ipa-server ipa-server-trust-ad samba-client
```

2. Authenticate as the IdM administrative user:

```
[root@ipaserver ~]# kinit admin
```

3. Run the `ipa-adtrust-install` utility:

```
[root@ipaserver ~]# ipa-adtrust-install
```

The DNS service records are created automatically if IdM was installed with an integrated DNS server.

If IdM was installed without an integrated DNS server, `ipa-adtrust-install` prints a list of service records that must be manually added to DNS before you can continue.

4. The script prompts you that the `/etc/samba/smb.conf` already exists and will be rewritten:

```
WARNING: The smb.conf already exists. Running ipa-adtrust-install will break your existing Samba configuration.
```
Do you wish to continue? [no]: yes

5. The script prompts you to configure the slapi-nis plug-in, a compatibility plug-in that allows older Linux clients to work with trusted users:

Do you want to enable support for trusted domains in Schema Compatibility plugin? This will allow clients older than SSSD 1.9 and non-Linux clients to work with trusted users.

Enable trusted domains support in slapi-nis? [no]: yes

6. When prompted, enter the NetBIOS name for the IdM domain or press Enter to accept the name suggested:

Trust is configured but no NetBIOS domain name found, setting it now.
Enter the NetBIOS name for the IPA domain.
Only up to 15 uppercase ASCII letters, digits and dashes are allowed.
Example: EXAMPLE.

NetBIOS domain name [IDM]:

7. You are prompted to run the SID generation task to create a SID for any existing users:

Do you want to run the ipa-sidgen task? [no]: yes

When the directory is first installed, at least one user (the IdM administrator) exists and as this is a resource-intensive task, if you have a high number of users, you can run this at another time.

8. Restart the ipa service:

[root@ipaserver ~]# systemctl restart ipa

9. Use the smbclient utility to verify that Samba responds to Kerberos authentication from the IdM side:

[root@ipaserver ~]# smbclient -L server.idm.example.com -k
lp_load_ex: changing to config backend registry
Sharename       Type      Comment
---------       ----      -------
IPC$            IPC       IPC Service (Samba 4.10.4)
...

40.2. ENABLING THE AES ENCRYPTION TYPE IN ACTIVE DIRECTORY USING A GPO

This section describes how to enable the AES encryption type in Active Directory (AD) using a group policy object (GPO). Certain Identity Management (IdM) features, such as running a Samba server on an IdM client, require this encryption type.

Note that RHEL 8 does not support the weak DES and RC4 encryption types.

Prerequisites
You are logged into AD as a user who can edit group policies.

The Group Policy Management Console is installed on the computer.

Procedure

1. Open the Group Policy Management Console.
2. Right-click Default Domain Policy, and select Edit. The Group Policy Management Editor opens.
4. Double-click the Network security: Configure encryption types allowed for Kerberos policy.
5. Select AES256_HMAC_SHA1 and, optionally, Future encryption types.
6. Click OK.
7. Close the Group Policy Management Editor.
8. Repeat the steps for the Default Domain Controller Policy.
9. Wait until the Windows domain controllers (DC) applied the group policy automatically. Alternatively, to apply the GPO manually on a DC, enter the following command using an account that has administrator permissions:

   C:\> gpupdate /force /target:computer

40.3. INSTALLING AND CONFIGURING A SAMBA SERVER ON AN IDM CLIENT

This section describes how to install and configure Samba on a client enrolled in an IdM domain.

Prerequisites

- Both the IdM servers and the client must run on RHEL 8.1 or later.
- The IdM domain is prepared as described in Section 40.1, "Preparing the IdM domain for installing Samba on domain members".
- If IdM has a trust configured with AD, enable the AES encryption type for Kerberos. For example, use a group policy object (GPO) to enable the AES encryption type. For details, see Section 40.2, "Enabling the AES encryption type in Active Directory using a GPO".

Procedure

1. Install the ipa-client-samba package:

   [root@idm_client]# yum install ipa-client-samba

2. Use the ipa-client-samba utility to prepare the client and create an initial Samba configuration:
By default, ipa-client-samba automatically adds the [homes] section to the /etc/samba/smb.conf file that dynamically shares a user’s home directory when the user connects. If users do not have home directories on this server, or if you do not want to share them, remove the following lines from /etc/samba/smb.conf:

```
[homes]
   read only = no
```

Share directories and printers. For details, see the following sections in the Deploying different types of servers documentation for RHEL 8:

- Configuring file shares on a Samba server
- Setting up Samba as a print server

Open the ports required for a Samba client in the local firewall:

```
[root@idm_client]# firewall-cmd --permanent --add-service=samba-client
[root@idm_client]# firewall-cmd --reload
```

Enable and start the smb and winbind services:

```
[root@idm_client]# systemctl enable --now smb winbind
```

Verification steps

Run the following verification steps on a different IdM domain member that has the samba-client package installed:

1.Authenticate and obtain a Kerberos ticket-granting ticket:

```
$ kinit example_user
```
2. List the shares on the Samba server using Kerberos authentication:

   $ smbclient -L idm_client.idm.example.com -k

   lp_load_ex: changing to config backend registry

   Sharename       Type      Comment
   ---------       ----      -------
   example        Disk
   IPC$           IPC       IPC Service (Samba 4.10.4)
   ...

Additional resources

- For details about which steps `ipa-client-samba` performs during the configuration, see the `ipa-client-samba(1)` man page.

40.4. MANUALLY ADDING AN ID MAPPING CONFIGURATION IF IDM TRUSTS A NEW DOMAIN

Samba requires an ID mapping configuration for each domain from which users access resources. On an existing Samba server running on an IdM client, you must manually add an ID mapping configuration after the administrator added a new trust to an Active Directory (AD) domain.

Prerequisites

- You configured Samba on an IdM client as described in Section 40.3, “Installing and configuring a Samba server on an IdM client”. Afterward, new trust was added to IdM.

- The DES and RC4 encryption types for Kerberos must be disabled in the trusted AD domain. For security reasons, RHEL 8 does not support these weak encryption types.

Procedure

1. Authenticate using the host’s keytab:

   [root@idm_client]# kinit -k

2. Use the `ipa idrange-find` command to display both the base ID and the ID range size of the new domain. For example, the following command displays the values for the `ad.example.com` domain:

   [root@idm_client]# ipa idrange-find --name="AD.EXAMPLE.COM_id_range" --raw

   1 range matched
   ---------------
   on: AD.EXAMPLE.COM_id_range
   ipabaseid: 1918400000
   ipaidrangesize: 200000
   ipabaserid: 0
   ipantrusteddomainsid: S-1-5-21-968346183-862388825-1738313271
   iparangetype: ipa-ad-trust
   ---------------
   Number of entries returned 1
   ---------------
You need the values from the \texttt{ipabaseid} and \texttt{ipaidrangesize} attributes in the next steps.

3. To calculate the highest usable ID, use the following formula:

   \[ \text{maximum\_range} = \text{ipabaseid} + \text{ipaidrangesize} - 1 \]

   With the values from the previous step, the highest usable ID for the \texttt{ad.example.com} domain is \texttt{1918599999} (1918400000 + 200000 - 1).

4. Edit the \texttt{/etc/samba/smb.conf} file, and add the ID mapping configuration for the domain to the \texttt{[global]} section:

   \begin{verbatim}
   idmap config AD: range = 1918400000 - 1918599999
   idmap config AD: backend = sss
   \end{verbatim}

   Specify the value from \texttt{ipabaseid} attribute as the lowest and the computed value from the previous step as the highest value of the range.

5. Restart the \texttt{ smb} and \texttt{winbind} services:

   \begin{verbatim}
   [root@idm_client]# systemctl restart smb winbind
   \end{verbatim}

**Verification steps**

1. Authenticate as a user from the new domain and obtain a Kerberos ticket-granting ticket:

   \begin{verbatim}
   $ kinit example_user
   \end{verbatim}

2. List the shares on the Samba server using Kerberos authentication:

   \begin{verbatim}
   $ smbclient -L idm_client.idm.example.com -k
   lp_load_ex: changing to config backend registry
   
   Sharename  Type    Comment
   -------    ----    -------
   example    Disk
   IPC$       IPC     IPC Service (Samba 4.10.4)
   ...        
   \end{verbatim}

**40.5. ADDITIONAL RESOURCES**

- For details about joining RHEL 8 to an IdM domain, see the \texttt{Installing an Identity Management client} section in the \texttt{Installing Identity Management} guide.
CHAPTER 41. IDENTITY MANAGEMENT SECURITY SETTINGS

This section describes security-related features of Identity Management.

41.1. HOW IDENTITY MANAGEMENT APPLIES DEFAULT SECURITY SETTINGS

By default, Identity Management (IdM) on RHEL 8 uses the system-wide crypto policy. The benefit of this policy is that you do not need to harden individual IdM components manually.

IMPORTANT

Red Hat recommends that you use the system-wide crypto policy. Changing individual security settings can break components of IdM. For example, Java in RHEL 8 does not fully support the TLS 1.3 protocol. Therefore, using this protocol can cause failures in IdM.

Additional resources

- For further details about the system-wide crypto policies, see the `crypto-policies(7)` man page.

41.2. ANONYMOUS LDAP BINDS IN IDENTITY MANAGEMENT

By default, anonymous binds to the Identity Management (IdM) LDAP server are enabled. Anonymous binds can expose certain configuration settings or directory values. However, some utilities, such as `reald`, or older RHEL clients require anonymous binds enabled to discover domain settings when enrolling a client.

Additional resources

- For details about disabling anonymous binds in the IdM LDAP server, see the `Disabling Anonymous Binds` section in the `Red Hat Directory Server 11 Administration Guide`. 

CHAPTER 42. USING AUTOMOUNT IN IDM

Automount is a way to manage, organize, and access directories across multiple systems. Automount program automatically mounts a directory whenever access to it is requested. This works well within an IdM domain since it allows directories on clients within the domain to be shared easily. This is especially important with user home directories.

In IdM, automount works with the internal LDAP directory and also with DNS services if configured.

42.1. SETTING UP A KERBEROS-AWARE NFS SERVER

This procedure describes how to set up a Kerberos-aware NFS server.

Prerequisites

- IdM domain set up. For more information, see Installing Identity Management.
- IPA client installed. For more information, see Installing ipa-client packages.

Procedure

1. If any of your NFS clients support only weak cryptography, such as Red Hat Enterprise Linux 5 clients:
   a. Update the IdM server Kerberos configuration to enable the weak des-cbc-crc encryption type:

      ```bash
      $ ldapmodify -x -D "cn=directory manager" -w password -h ipaserver.example.com -p 389
dn: cn=REALM_NAME,cn=kerberos,dc=example,dc=com
      changetype: modify
      add: krbSupportedEncSaltTypes
      krbSupportedEncSaltTypes: des-cbc-crc:normal
      -
      add: krbSupportedEncSaltTypes
      krbSupportedEncSaltTypes: des-cbc-crc:special
      -
      add: krbDefaultEncSaltTypes
      krbDefaultEncSaltTypes: des-cbc-crc:special
      ```

   b. On the NFS server, add the following entry to the /etc/krb5.conf file of the NFS server enable weak cryptography support:

      ```bash
      allow_weak_crypto = true
      ```

2. Obtain a Kerberos ticket:

   ```bash
   [root@nfs-server ~]# kinit admin
   ```

3. If the NFS host machine has not been added as a client to the IdM domain, create the host entry. See Adding IdM host entries from IdM CLI.

4. Create the NFS service entry:
5. Retrieve an NFS service keytab for the NFS server using the following `ipa-getkeytab` command that saves the keys in the `/etc/krb5.keytab` file:

   ```
   # ipa-getkeytab -s ipaserver.example.com -p nfs/nfs-server.example.com -k /etc/krb5.keytab
   ```

   If any of your NFS clients support only weak cryptography, additionally pass the `-e des-cbc-crc` option to the command to request a DES-encrypted keytab.

6. Verify that the NFS service has been properly configured in IdM, with its keytab, by checking the service entry:

   ```
   # ipa-service-show nfs/nfs-server.example.com
   ```

   ```
   Principal name: nfs/nfs-server.example.com@IDM.EXAMPLE.COM
   Principal alias: nfs/nfs-server.example.com@IDM.EXAMPLE.COM
   Keytab: True
   Managed by: nfs-server.example.com
   ```

7. Install the `nfs-utils` package:

   ```
   # yum install nfs-utils
   ```

8. Run the `ipa-client-automount` utility to configure the NFS settings:

   ```
   # ipa-client-automount
   ```

   ```
   Searching for IPA server...
   IPA server: DNS discovery
   Location: default
   Continue to configure the system with these values? [no]: yes
   Configured /etc/idmapd.conf
   Restarting sssd, waiting for it to become available.
   Started autofs
   ```

   By default, this command enables secure NFS and sets the `Domain` parameter in the `/etc/idmapd.conf` file to the IdM DNS domain. If you use a different domain, specify it using the `--idmap-domain domain_name` parameter.

9. Edit the `/etc/exports` file and add shares with the `krb5p` Kerberos security setting:

   ```
   /export *(rw,sec=krb5:krb5i:krb5p)
   /home *(rw,sec=krb5:krb5i:krb5p)
   ```

   This example shares the `/export` and `/home` directories in read-write mode with Kerberos authentication enabled.

10. Restart and enable nfs-server:

    ```
    # systemctl restart nfs-server
    # systemctl enable nfs-server
    ```

11. Re-export the shared directories:
12. Optionally, configure the NFS server as an NFS client. See Section 42.2, “Setting up a Kerberos-aware NFS client”.

42.2. SETTING UP A KERBEROS-AWARE NFS CLIENT

This procedure describes how to set up a kerberos-aware NFS client.

Prerequisites

- IdM domain set up. For more information, see Installing Identity Management.
- IPA client installed. For more information, see Installing ipa-client packages.

Procedure

1. If the NFS clients supports only weak cryptography, such as a Red Hat Enterprise Linux 5 client, set the following entry in the /etc/krb5.conf file of the server to allow weak cryptography:

```
allow_weak_crypto = true
```

2. If the NFS client is not enrolled as a client in the IdM domain, set up the required host entries, as described in Adding IdM host entries from IdM CLI.

3. Install the nfs-utils package:

```
[root@nfs-client ~]# yum install nfs-utils
```

4. Obtain a Kerberos ticket before running IdM tools.

```
[root@nfs-client ~]# kinit admin
```

5. Run the ipa-client-automount utility to configure the NFS settings:

```
[root@nfs-client ~] ipa-client-automount
Searching for IPA server...
IPA server: DNS discovery
Location: default
Continue to configure the system with these values? [no]: yes
Configured /etc/idmapd.conf
Restarting sssd, waiting for it to become available.
Started autofs
```

By default, this enables secure NFS in the /etc/sysconfig/nfs file and sets the IdM DNS domain in the Domain parameter in the /etc/idmapd.conf file.

6. Add the following entries to the /etc/fstab file to mount the NFS shares from the nfs-server.example.com host when the system boots:

```
nfs-server.example.com:/export /mnt  nfs4  sec=krb5p,rw
nfs-server.example.com:/home  /home  nfs4  sec=krb5p,rw
```
These settings configure Red Hat Enterprise Linux to mount the `/export` share to the `/mnt` and the `/home` share to the `/home` directory.

7. Create the mount points if they do not exist. In our case both should exist.

8. Mount the NFS shares:

   ```bash
   [root@nfs-client ~]# mount /mnt/
   [root@nfs-client ~]# mount /home
   ```

   The command uses the information from the `/etc/fstab` entry.

9. Configure SSSD to renew Kerberos tickets:
   a. Set the following parameters in the IdM domain section of the `/etc/sssd/sssd.conf` file to configure SSSD to automatically renew tickets:

   ```
   [domain/EXAMPLE.COM]
   ...
   krb5_renewable_lifetime = 50d
   krb5_renew_interval = 3600
   ```

   b. Restart SSSD:

   ```bash
   [root@nfs-client ~]# systemctl restart sssd
   ```

   **IMPORTANT**

   The `pam_oddjob_mkhomedir` module does not support automatic creation of home directories on an NFS share. Therefore, you must manually create the home directories on the server in the root of the share that contains the home directories.