Red Hat Enterprise Linux 8

Migrating to Identity Management on RHEL 8

Upgrading a RHEL 7 IdM environment to RHEL 8 and migrating FreeIPA or external LDAP solutions to IdM

Last Updated: 2024-04-08

Red Hat Enterprise Linux 8 Migrating to Identity Management on RHEL 8

Upgrading a RHEL 7 IdM environment to RHEL 8 and migrating FreeIPA or external LDAP solutions to IdM

Legal Notice

Abstract

Red Hat only supports Identity Management (IdM) on Red Hat Enterprise Linux (RHEL). If you run IdM on RHEL 7, FreeIPA on other Linux distributions, or an LDAP directory, you can migrate these solutions to IdM on RHEL 8.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message .

In Identity Management, planned terminology replacements include:

- ***block list*** replaces *blacklist*
- ***allow list*** replaces *whitelist*
- ***secondary*** replaces *slave*
- The word *master* is being replaced with more precise language, depending on the context:
  - ***IdM server*** replaces *IdM master*
  - ***CA renewal server*** replaces *CA renewal master*
  - ***CRL publisher server*** replaces *CRL master*
  - ***multi-supplier*** replaces *multi-master*

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our documentation. Let us know how we can improve it.

Submitting feedback through Jira (account required)

1. Log in to the Jira website.
2. Click **Create** in the top navigation bar.
3. Enter a descriptive title in the **Summary** field.
4. Enter your suggestion for improvement in the **Description** field. Include links to the relevant parts of the documentation.
5. Click **Create** at the bottom of the dialogue.

PART I. MIGRATING IDM FROM RHEL 7 TO RHEL 8

CHAPTER 1. MIGRATING YOUR IDM ENVIRONMENT FROM RHEL 7 SERVERS TO RHEL 8 SERVERS

To upgrade a RHEL 7 IdM environment to RHEL 8, you must first add new RHEL 8 IdM replicas to your RHEL 7 IdM environment, and then retire the RHEL 7 servers.

WARNING

- Performing an in-place upgrade of RHEL 7 IdM servers to RHEL 8 is not supported.
- Migrating directly to RHEL 8 from RHEL 6 or earlier versions is not supported. To properly update your IdM data, you must perform incremental migrations.
  For example, to migrate a RHEL 6 IdM environment to RHEL 8:
  a. Migrate from RHEL 6 servers to RHEL 7 servers. See Migrating Identity Management from Red Hat Enterprise Linux 6 to Version 7.
  b. Migrate from RHEL 7 servers to RHEL 8 servers, as described in this section.

IMPORTANT

RHEL 8 supports SPAKE and IdP pre-authentication, but RHEL 7 does not. Having RHEL 8 servers with SPAKE or IdP enabled in a RHEL 7 IdM deployment may lead to problems such as users not being able to log in.

Red Hat strongly recommends that all servers in an IdM deployment are migrated as quickly as possible and that older systems should not be left in operation for extended periods of time.

For more information, see:

- https://access.redhat.com/solutions/7053377
- https://access.redhat.com/solutions/3529911

This procedure describes how to **migrate** all Identity Management (IdM) data and configuration from a Red Hat Enterprise Linux (RHEL) 7 server to a RHEL 8 server. You can also use this procedure to migrate from FreeIPA servers on non-RHEL Linux distributions to IdM on RHEL 8 servers.

The migration procedure includes:

1. Configuring a RHEL 8 IdM server and adding it as a replica to your current RHEL 7 IdM environment. For details, see Installing the RHEL 8 Replica .
2. Making the RHEL 8 server the certificate authority (CA) renewal server. For details, see Assigning the CA renewal server role to the RHEL 8 IdM server .

3. Stopping the generation of the certificate revocation list (CRL) on the RHEL 7 server and redirecting CRL requests to RHEL 8. For details, see Stopping CRL generation on a RHEL 7 IdM CA server.

4. Starting the generation of the CRL on the RHEL 8 server. For details, see Starting CRL generation on the new RHEL 8 IdM CA server.

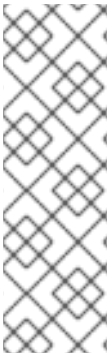
5. Stopping and decommissioning the original RHEL 7 CA renewal server. For details, see Stopping and decommissioning the RHEL 7 server .

In the following procedures:

- **rhel8.example.com** is the RHEL 8 system that will become the new CA renewal server.
- **rhel7.example.com** is the original RHEL 7 CA renewal server. To identify which Red Hat Enterprise Linux 7 server is the CA renewal server, run the following command on any IdM server:

```
[root@rhel7 ~]# ipa config-show | grep "CA renewal"
IPA CA renewal master: rhel7.example.com
```

  If your IdM deployment does not use a certificate authority (CA), any IdM server running on RHEL 7 can be **rhel7.example.com**.



NOTE

Complete the steps in the following sections **only** if your IdM deployment uses an embedded CA:

- Assigning the CA renewal server role to the RHEL 8 IdM server
- Stopping CRL generation on a RHEL 7 IdM CA server
- Starting CRL generation on the new RHEL 8 IdM CA server

1.1. PREREQUISITES FOR MIGRATING IDM FROM RHEL 7 TO 8

On **rhel7.example.com**:

1. Upgrade the system to the latest RHEL 7 version.

2. Ensure that the domain level for your domain is set to 1. For more information, see Displaying and Raising the Domain Level in the *Linux Domain Identity, Authentication, and Policy Guide* for RHEL 7.

3. Update the **ipa-*** packages to their latest version:

```
[root@rhel7 ~]# yum update ipa-*
```

WARNING

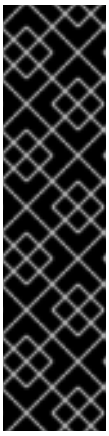
When upgrading multiple Identity Management (IdM) servers, wait at least 10 minutes between each upgrade.

When two or more servers are upgraded simultaneously or with only short intervals between the upgrades, there is not enough time to replicate the post-upgrade data changes throughout the topology, which can result in conflicting replication events.

On **rhel8.example.com**:

1. The latest version of Red Hat Enterprise Linux is installed on the system. For more information, see Performing a standard RHEL 8 installation .
2. Ensure you know the time server **rhel7.example.com** is synchronized with:

```
[root@rhel7 ~]# ntpstat
synchronised to NTP server (ntp.example.com) at stratum 3
   time correct to within 42 ms
   polling server every 1024 s
```

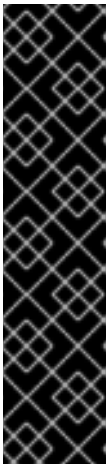


IMPORTANT

In RHEL 8, IdM does not provide its own time server: the installation of IdM on **rhel8.example.com** does not result in the installation of an NTP server on the host. Therefore, you need to use a separate NTP server, for example **ntp.example.com**. For more information, see Migrating to chrony and Time service requirements for IdM.

While **rhel7.example.com** can be used in an NTP server role, you will decommission the server as part of the migration process. Therefore, **rhel8.example.com** needs to be synchronized directly with **ntp.example.com** instead. You can specify this during the client installation process.

3. Ensure the system is an IdM client enrolled into the domain for which **rhel7.example.com** IdM server is authoritative. For more information, see Installing an IdM client .



IMPORTANT

When installing the client, specify the time server from the previous step using the **--ntp-server** option. If you are using a pool of NTP servers, use the **--ntp-pool** option.

If you do not specify an NTP server manually, it will be automatically set from DNS records. This can lead to **rhel8.example.com** synchronizing with **rhel7.example.com**. This will cause issues when the RHEL 7 server is decommissioned.

If the RHEL8 system is already properly configured as an NTP client, you can use the **--no-ntp** option when performing the IdM client installation.

4. Ensure the system meets the requirements for IdM server installation. See Preparing the system for IdM server installation.

5. Ensure the system is authorized for the installation of an IdM replica. See Authorizing the installation of a replica on an IdM client.

6. Update the **ipa-*** packages to their latest version:

```
[root@rhel7 ~]# yum update ipa-*
```

Additional resources

- Planning your CA services
- Planning your DNS services and host names
- Planning a cross-forest trust between IdM and AD
- Installing packages required for an IdM server .
- Upgrading from RHEL 7 to RHEL 8 .

1.2. INSTALLING THE RHEL 8 REPLICA

1. List which server roles are present in your RHEL 7 environment:

```
[root@rhel7 ~]# ipa server-role-find --status enabled --server rhel7.example.com
----------------------
3 server roles matched
----------------------
  Server name: rhel7.example.com
  Role name: CA server
  Role status: enabled

  Server name: rhel7.example.com
  Role name: DNS server
  Role status: enabled

  Server name: rhel7.example.com
```

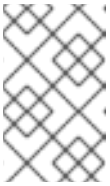
```
  Role name: NTP server
  Role status: enabled
[... output truncated ...]
```

2. [Optional] If you want to use the same per-server forwarders for **rhel8.example.com** that **rhel7.example.com** is using, view the per-server forwarders for **rhel7.example.com**:

```
[root@rhel7 ~]# ipa dnsserver-show rhel7.example.com
----------------------------
1 DNS server matched
----------------------------
  Server name: rhel7.example.com
  SOA mname: rhel7.example.com.
  Forwarders: 192.0.2.20
  Forward policy: only
------------------------------------------------
Number of entries returned 1
------------------------------------------------
```

3. Install the IdM server on **rhel8.example.com** as a replica of the IdM RHEL 7 server, including all the server roles present on your **rhel7.example.com** except the NTP server role. To install the roles from the example above, use these options with the **ipa-replica-install** command:

   - **--setup-ca** to set up the Certificate System component
   - **--setup-dns** and **--forwarder** to configure an integrated DNS server and set a per-server forwarder to take care of DNS queries that go outside the IdM domain

   NOTE

   Additionally, if your IdM deployment is in a trust relationship with Active Directory (AD), add the **--setup-adtrust** option to the **ipa-replica-install** command to configure AD trust capability on **rhel8.example.com**.

   To set up an IdM server with the IP address of 192.0.2.1 that uses a per-server forwarder with the IP address of 192.0.2.20:

   ```
   [root@rhel8 ~]# ipa-replica-install --setup-ca --ip-address 192.0.2.1 --setup-dns --forwarder 192.0.2.20
   ```

   You do not need to specify the RHEL 7 IdM server itself because if DNS is working correctly, **rhel8.example.com** will find it using DNS autodiscovery.

4. [Optional] Add an **_ntp._udp** service (SRV) record for your external **NTP** time server to the DNS of the newly-installed IdM server, **rhel8.example.com**. Doing this is recommended because IdM in RHEL 8 does not provide its own time service. The presence of the SRV record for the time server in IdM DNS ensures that future RHEL 8 replica and client installations are automatically configured to synchronize with the time server used by **rhel8.example.com**. This is because **ipa-client-install** looks for the **_ntp._udp** DNS entry unless **--ntp-server** or **--ntp-pool** options are provided on the install command-line interface (CLI).

Verification

1. Verify that the IdM services are running on **rhel8.example.com**:

```
[root@rhel8 ~]# ipactl status
Directory Service: RUNNING
[... output truncated ...]
ipa: INFO: The ipactl command was successful
```

2. Verify that server roles for **rhel8.example.com** are the same as for **rhel7.example.com** except the NTP server role:

```
[root@rhel8 ~]$ kinit admin
[root@rhel8 ~]$ ipa server-role-find --status enabled --server rhel8.example.com
----------------------
2 server roles matched
----------------------
  Server name: rhel8.example.com
  Role name: CA server
  Role status: enabled

  Server name: rhel8.example.com
  Role name: DNS server
  Role status: enabled
```

3. [Optional] Display details about the replication agreement between **rhel7.example.com** and **rhel8.example.com**:

```
[root@rhel8 ~]# ipa-csreplica-manage list --verbose rhel8.example.com
Directory Manager password:

rhel7.example.com
last init status: None
last init ended: 1970-01-01 00:00:00+00:00
last update status: Error (0) Replica acquired successfully: Incremental update succeeded
last update ended: 2019-02-13 13:55:13+00:00
```

4. [Optional] If your IdM deployment is in a trust relationship with AD, verify that it is working:

   a. link: Verify the Kerberos configuration

   b. Attempt to resolve an AD user on **rhel8.example.com**:

```
[root@rhel8 ~]# id aduser@ad.domain
```

5. Verify that **rhel8.example.com** is synchronized with the **NTP** server:

```
[root@rhel8 ~]# chronyc tracking
Reference ID    : CB00710F (ntp.example.com)
Stratum         : 3
Ref time (UTC)  : Tue Nov 16 09:49:17 2021
[... output truncated ...]
```

Additional resources

- DNS configuration priorities
- Time service requirements for IdM

- Migrating to chrony

1.3. ASSIGNING THE CA RENEWAL SERVER ROLE TO THE RHEL 8 IDM SERVER

Follow this procedure to make the RHEL 8 server the certificate authority (CA) renewal server.

NOTE

Follow these steps only if your IdM deployment uses an embedded certificate authority (CA).

On **rhel8.example.com**, configure **rhel8.example.com** as the new CA renewal server:

1. Configure **rhel8.example.com** to handle CA subsystem certificate renewal:

```
[root@rhel8 ~]# ipa config-mod --ca-renewal-master-server rhel8.example.com
...
IPA masters: rhel7.example.com, rhel8.example.com
IPA CA servers: rhel7.example.com, rhel8.example.com
IPA NTP servers: rhel7.example.com, rhel8.example.com
IPA CA renewal master: rhel8.example.com
```

   The output confirms that the update was successful.

2. On **rhel8.example.com**, enable the certificate updater task:
   a. Open the **/etc/pki/pki-tomcat/ca/CS.cfg** configuration file for editing.
   b. Remove the **ca.certStatusUpdateInterval** entry, or set it to the desired interval in seconds. The default value is **600**.
   c. Save and close the **/etc/pki/pki-tomcat/ca/CS.cfg** configuration file.
   d. Restart IdM services:

```
[user@rhel8 ~]$ ipactl restart
```

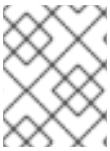
3. On **rhel7.example.com**, disable the certificate updater task:
   a. Open the **/etc/pki/pki-tomcat/ca/CS.cfg** configuration file for editing.
   b. Change **ca.certStatusUpdateInterval** to **0**, or add the following entry if it does not exist:

```
ca.certStatusUpdateInterval=0
```

   c. Save and close the **/etc/pki/pki-tomcat/ca/CS.cfg** configuration file.
   d. Restart IdM services:

```
[user@rhel7 ~]$ ipactl restart
```

1.4. STOPPING CRL GENERATION ON A RHEL 7 IDM CA SERVER



NOTE

Follow these steps only if your IdM deployment uses an embedded certificate authority (CA).

Follow this procedure to stop generating the Certificate Revocation List (CRL) on the **rhel7.example.com** CA server using the **ipa-crlgen-manage** command.

Prerequisites

- You must be logged in as root.

Procedure

1. Optionally, check if **rhel7.example.com** is generating the CRL:

```
[root@rhel7 ~]# 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 the CRL on the **rhel7.example.com** server:

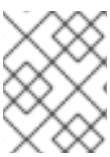
```
[root@rhel7 ~]# ipa-crlgen-manage disable
Stopping pki-tomcatd
Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
Starting pki-tomcatd
Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
Restarting httpd
CRL generation disabled on the local host. Please make sure to configure CRL generation on another master with ipa-crlgen-manage enable.
The ipa-crlgen-manage command was successful
```

3. Optionally, check if the **rhel7.example.com** server stopped generating the CRL:

```
[root@rhel7 ~]# ipa-crlgen-manage status
```

The **rhel7.example.com** server stopped generating the CRL. The next step is to enable generating the CRL on **rhel8.example.com**.

1.5. STARTING CRL GENERATION ON THE NEW RHEL 8 IDM CA SERVER



NOTE

Follow these steps only if your IdM deployment uses an embedded certificate authority (CA).

Prerequisites

- You must be logged in as root on the **rhel8.example.com** machine.

Procedure

1. To start generating CRL on **rhel8.example.com**, use the **ipa-crlgen-manage enable** command:

```
[root@rhel8 ~]# 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 generation is enabled, use the **ipa-crlgen-manage status** command:

```
[root@rhel8 ~]# 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
```

1.6. STOPPING AND DECOMMISSIONING THE RHEL 7 SERVER

1. Ensure that all data, including the latest changes, have been correctly migrated from **rhel7.example.com** to **rhel8.example.com**. For example:

   a. Add a new user on **rhel7.example.com**:

```
[root@rhel7 ~]# ipa user-add random_user
First name: random
Last name: user
```

   b. Check that the user has been replicated to **rhel8.example.com**:

```
[root@rhel8 ~]# ipa user-find random_user
--------------
1 user matched
--------------
  User login: random_user
  First name: random
  Last name: user
```

2. Ensure that a Distributed Numeric Assignment (DNA) ID range is allocated to **rhel8.example.com**. Use one of the following methods:

   - Activate the DNA plug-in on **rhel8.example.com** directly by creating another test user:

```
[root@rhel8 ~]# ipa user-add another_random_user
First name: another
Last name: random_user
```

   - Assign a specific DNA ID range to **rhel8.example.com**:

i. On **rhel7.example.com**, display the IdM ID range:

```
[root@rhel7 ~]# ipa idrange-find
----------------
3 ranges matched
----------------
  Range name: EXAMPLE.COM_id_range
  First Posix ID of the range: 196600000
  Number of IDs in the range: 200000
  First RID of the corresponding RID range: 1000
  First RID of the secondary RID range: 100000000
  Range type: local domain range
```

ii. On **rhel7.example.com**, display the allocated DNA ID ranges:

```
[root@rhel7 ~]# ipa-replica-manage dnarange-show
rhel7.example.com: 196600026-196799999
rhel8.example.com: No range set
```

iii. Reduce the DNA ID range allocated to **rhel7.example.com** so that a section becomes available to **rhel8.example.com**:

```
[root@rhel7 ~]# ipa-replica-manage dnarange-set rhel7.example.com
196600026-196699999
```

iv. Assign the remaining part of the IdM ID range to **rhel8.example.com**:

```
[root@rhel7 ~]# ipa-replica-manage dnarange-set rhel8.example.com
196700000-196799999
```

3. Stop all IdM services on **rhel7.example.com** to force domain discovery to the new **rhel8.example.com** server.

```
[root@rhel7 ~]# ipactl stop
Stopping CA Service
Stopping pki-ca:                                   [  OK  ]
Stopping HTTP Service
Stopping httpd:                                    [  OK  ]
Stopping MEMCACHE Service
Stopping ipa_memcached:                            [  OK  ]
Stopping DNS Service
Stopping named: .                                  [  OK  ]
Stopping KPASSWD Service
Stopping Kerberos 5 Admin Server:                  [  OK  ]
Stopping KDC Service
Stopping Kerberos 5 KDC:                           [  OK  ]
Stopping Directory Service
Shutting down dirsrv:
   EXAMPLE-COM...                                  [  OK  ]
   PKI-IPA...                                      [  OK  ]
```

After this, the **ipa** utility will contact the new server through a remote procedure call (RPC).

4. Remove the RHEL 7 server from the topology by executing the removal commands on the RHEL 8 server. For details, see Uninstalling an IdM server.

Additional resources

- Adjusting ID ranges manually

CHAPTER 2. UPGRADING AN IDM CLIENT FROM RHEL 7 TO RHEL 8

Unlike IdM servers, performing an in-place upgrade of an IdM client from RHEL 7 to RHEL 8 is supported.

In RHEL 8, some uncommon options and unused functionality have been removed from the System Security Services Daemon (SSSD), the service responsible for authentication in an IdM environment. See the following sections for steps to remove those options.

- Updating the SSSD configuration after upgrading to RHEL 8
- List of SSSD functionality removed in RHEL 8

2.1. UPDATING THE SSSD CONFIGURATION AFTER UPGRADING TO RHEL 8

After upgrading an Identity Management (IdM) client from Red Hat Enterprise Linux (RHEL) 7 to RHEL 8, certain SSSD configuration options might no longer be supported. The **leapp** upgrade application might provide more details about such options in the generated pre-upgrade report.

The following procedures describe how to update your SSSD configuration to address these issues.

Prerequisites

- You have upgraded an IdM client from RHEL 7 to RHEL 8.
- You have **root** permissions to edit **/etc/sssd/sssd.conf**.

2.1.1. Switching from the local ID provider to the files ID provider

If you see the following error, replace the **local** ID provider with the **files** ID provider:

```
SSSD Domain "example.com": local provider is no longer supported and the domain will be ignored.
Local provider is no longer supported.
```

Procedure

1. Ensure any users and groups you retrieved with the **local** ID provider are also in the **/etc/passwd** and **/etc/group** files. This ensures that the **files** provider can access those users and groups.

   a. If you need to create users, use the **useradd** command. If you need to specify the UID, add the **-u** option:

   ```
   [root@client ~]# useradd -u 3001 username
   ```

   b. If you need to create groups, use the **groupadd** command. If you need to specify the GID, add the **-g** option:

   ```
   [root@client ~]# groupadd -g 5001 groupname
   ```

2. Open the **/etc/sssd/sssd.conf** configuration file in a text editor.
3. Replace **id_provider=local** with **id_provider=files**.

```
[domain/example.com]
id_provider = files
...
```

4. Save the **/etc/sssd/sssd.conf** configuration file.
5. Restart SSSD to load the configuration changes.

```
[root@client ~]# systemctl restart sssd
```

2.1.2. Removing deprecated options

If you see either of the following errors regarding deprecated options, Red Hat recommends removing those options from the **/etc/sssd/sssd.conf** configuration file:

```
SSSD Domain "example.com": option ldap_groups_use_matching_rule_in_chain has no longer any effect
Option ldap_groups_use_matching_rule_in_chain was removed and it will be ignored.
```

```
SSSD Domain "example.com": option ldap_initgroups_use_matching_rule_in_chain has no longer any effect
Option ldap_initgroups_use_matching_rule_in_chain was removed and it will be ignored.
```

Procedure

1. Open the **/etc/sssd/sssd.conf** configuration file in a text editor.
2. Remove any occurrences of **ldap_groups_use_matching_rule_in_chain** or **ldap_initgroups_use_matching_rule_in_chain** options.
3. Save the **/etc/sssd/sssd.conf** configuration file.
4. Restart SSSD to load the configuration changes.

```
[root@client ~]# systemctl restart sssd
```

2.1.3. Enabling wildcard matching for sudo rules

The following warning indicates that **sudo** rules with wildcards in them will not work by default in RHEL 8, as the **ldap_sudo_include_regexp** option is now set to **false** by default.

```
SSSD Domain "example.com": sudo rules containing wildcards will stop working.
Default value of ldap_sudo_include_regexp changed from true to false for performance reason.
```

If you use **sudo** rules with wildcards and want to enable wildcard matching, manually set the **ldap_sudo_include_regexp** option to **true**.

NOTE

Red Hat recommends against using wildcards to match **sudo** rules.

If the **ldap_sudo_include_regexp** option is set to **true**, SSSD downloads every **sudo** rule that contains a wildcard in the **sudoHost** attribute, which negatively impacts LDAP search performance.

Procedure

1. Open the **/etc/sssd/sssd.conf** configuration file in a text editor.
2. In the **example.com** domain, set **ldap_sudo_include_regexp=true**.

```
[domain/example.com]
...
ldap_sudo_include_regexp = true
...
```

3. Save the **/etc/sssd/sssd.conf** configuration file.
4. Restart SSSD to load the configuration changes.

```
[root@client ~]# systemctl restart sssd
```

2.2. LIST OF SSSD FUNCTIONALITY REMOVED IN RHEL 8

The following SSSD functionality has been removed in RHEL 8.

The local ID provider has been removed

The **local** ID provider, used to serve user information from the local SSSD cache, was deprecated in RHEL 7 and is no longer supported in RHEL 8. If you have a domain with **id_provider=local** in your **/etc/sssd/sssd.conf** configuration, SSSD ignores this domain and starts normally.

Command line tools to manage users and groups in local domains have been removed

The following commands, which only affected **local** domains, have been removed:

- **sss_useradd**
- **sss_userdel**
- **sss_groupadd**
- **sss_groupdel**

Support for the ldap_groups_use_matching_rule_in_chain option has been removed

This Active Directory-specific option does not provide a significant performance benefit and is ignored in any RHEL 8 **sssd.conf** configuration.

Support for the ldap_initgroups_use_matching_rule_in_chain option has been removed

This Active Directory-specific option does not provide a significant performance benefit and is ignored in any RHEL 8 **sssd.conf** configuration.

The ldap_sudo_include_regexp option now defaults to false

In RHEL 7, this option was set to **true** by default. If this option is set to **true**, SSSD downloads every **sudo** rule that contains a wildcard in the **sudoHost** attribute, which negatively impacts LDAP search performance.

The sssd-secrets responder has been removed

As the Kerberos Cache Manager (KCM) no longer relies on the **sssd-secrets** responder, and no other IdM process uses it, it has been removed.

2.3. ADDITIONAL RESOURCES

- For more details on upgrading to RHEL 8, see Upgrading from RHEL 7 to RHEL 8 .

PART II. MIGRATING TO IDM FROM EXTERNAL SOURCES

CHAPTER 3. MIGRATING TO IDM ON RHEL 8 FROM FREEIPA ON NON-RHEL LINUX DISTRIBUTIONS

To migrate a FreeIPA deployment on a non-RHEL Linux distribution to an Identity Management (IdM) deployment on RHEL 8 servers, you must first add a new RHEL 8 IdM Certificate Authority (CA) replica to your existing FreeIPA environment, transfer certificate-related roles to it, and then retire the non-RHEL FreeIPA servers.

WARNING

Performing an in-place conversion of a non-RHEL FreeIPA server to a RHEL 8 IdM server using the Convert2RHEL tool is not supported.

To perform the migration, follow the same procedure as Migrating your IdM environment from RHEL 7 servers to RHEL 8 servers, with your non-RHEL FreeIPA CA replica acting as the RHEL 7 server:

1. Configure a RHEL 8 server and add it as an IdM replica to your current FreeIPA environment on the non-RHEL Linux distribution. For details, see Installing the RHEL 8 Replica .
2. Make the RHEL 8 replica the certificate authority (CA) renewal server. For details, see Assigning the CA renewal server role to the RHEL 8 IdM server .
3. Stop generating the certificate revocation list (CRL) on the non-RHEL server and redirect CRL requests to the RHEL 8 replica. For details, see Stopping CRL generation on a RHEL 7 IdM CA server.
4. Start generating the CRL on the RHEL 8 server. For details, see Starting CRL generation on the new RHEL 8 IdM CA server.
5. Stop and decommission the original non-RHEL FreeIPA CA renewal server. For details, see Stopping and decommissioning the RHEL 7 server .

Additional resources

- Migrating your IdM environment from RHEL 7 servers to RHEL 8 servers

CHAPTER 4. MIGRATING FROM AN LDAP DIRECTORY TO IDM

If you previously deployed an LDAP server for identity and authentication lookups, you can migrate the lookup service to Identity Management (IdM). IdM offers a migration tool to help you with the following tasks:

- Transferring user accounts, including passwords and group membership, without losing data.
- Avoiding expensive configuration updates on the clients.

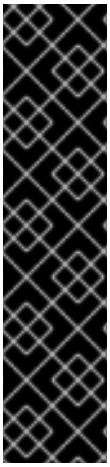
The migration process described here assumes a simple deployment scenario with one namespace in LDAP and one in IdM. For more complex environments, such as those with multiple namespaces or custom schemas, contact the Red Hat support services.

4.1. CONSIDERATIONS IN MIGRATING FROM LDAP TO IDM

The process of moving from an LDAP server to Identity Management (IdM) has the following stages:

- Migrating the *clients*. Plan this stage carefully. Determine which services each client in your current infrastructure uses. These may include for example Kerberos or Systems Security Services Daemon (SSSD). Then determine which of these services you can use in the final IdM deployment. See Planning the client configuration when migrating from LDAP to IdM for more information.
- Migrating the *data*.
- Migrating the *passwords*. Plan this stage carefully. IdM requires Kerberos hashes for every user account in addition to passwords. Some of the considerations and migration paths for passwords are covered in Planning password migration when migrating from LDAP to IdM .

You can first migrate the server part and then the clients or first the clients and then the server. For more information about the two types of migration, see LDAP to IdM migration sequence .



IMPORTANT

It is strongly recommended that you set up a test LDAP environment and test the migration process before attempting to migrate the real LDAP environment. When testing the environment, do the following:

1. Create a test user in IdM and compare the output of migrated users to that of the test user. Ensure that the migrated users contain the minimal set of attributes and object classes present on the test user.
2. Compare the output of migrated users, as seen on IdM, to the source users, as seen on the original LDAP server. Ensure that imported attributes are not copied twice and that they have the correct values.

4.2. PLANNING THE CLIENT CONFIGURATION WHEN MIGRATING FROM LDAP TO IDM

Identity Management (IdM) can support a number of different client configurations, with varying degrees of functionality, flexibility, and security. Decide which configuration is best for each individual client based on its operating system and your IT maintenance priorities. Consider also the client's functional area: a development machine typically requires a different configuration than production servers or user laptops do.

IMPORTANT

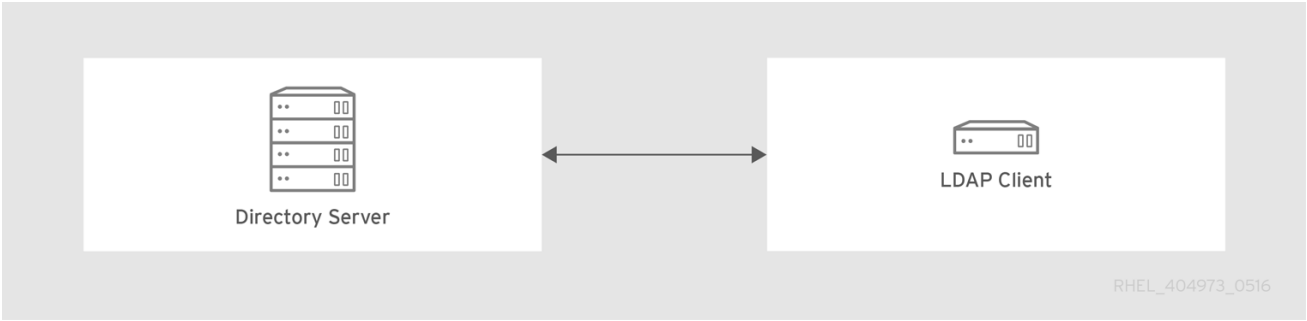
Most environments have a mixture of different ways in which clients connect to the IdM domain. Administrators must decide which scenario is best for each individual client.

4.2.1. Initial, pre-migration client configuration

Before deciding on the specifics of the client configuration in Identity Management (IdM), first establish the specifics of the current, pre-migration configuration.

The initial state for almost all LDAP deployments that are to be migrated is that there is an LDAP service providing identity and authentication services.

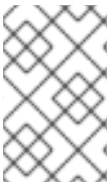
Figure 4.1. Basic LDAP directory and client configuration



Linux and Unix clients use the PAM_LDAP and NSS_LDAP libraries to connect directly to the LDAP services. These libraries allow clients to retrieve user information from the LDAP directory as if the data were stored in **/etc/passwd** or **/etc/shadow**. In real life, the infrastructure may be more complex if a client uses LDAP for identity lookups and Kerberos for authentication or other configurations.

There are structural differences between an LDAP directory and an Identity Management (IdM) server, particularly in schema support and the structure of the directory tree. For more background on those differences, see the **Contrasting IdM with a Standard LDAP Directory**section from the Planning the client configuration when migrating from LDAP to IdM. Those differences may impact data, especially with the directory tree, which affects entry names. However, the differences have little impact on the client configuration and on migrating clients to IdM.

4.2.2. Recommended configuration for RHEL clients



NOTE

The client configuration described is only supported for RHEL 6.1 and later and RHEL 5.7 later, which support the latest versions of SSSD and the **ipa-client** package. Older versions of RHEL can be configured as described in Alternative supported configuration .

The System Security Services Daemon (SSSD) in Red Hat Enterprise Linux (RHEL) uses special PAM and NSS libraries, **pam_sss** and **nss_sss**. Using these libraries, SSSD can integrate very closely with Identity Management (IdM) and benefit from its full authentication and identity features. SSSD has a number of useful features, such as caching identity information so that users can log in even if the connection to the central server is lost.

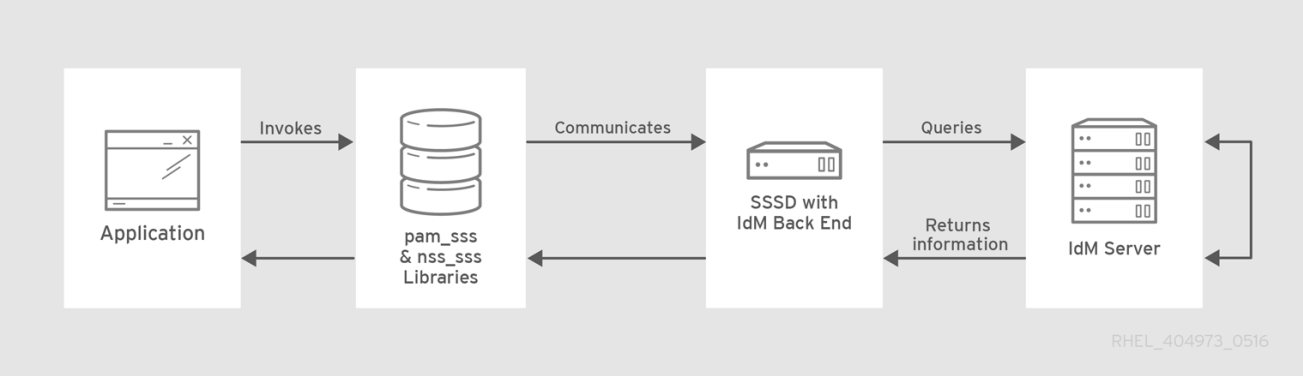
Unlike generic LDAP directory services that use the **pam_ldap** and **nss_ldap** libraries, SSSD establishes relationships between identity and authentication information by defining *domains*. A domain in SSSD defines the following back end functions:

- Authentication
- Identity lookups
- Access
- Password changes

The SSSD domain is then configured to use a *provider* to supply the information for any one, or all, of these functions. The domain configuration always requires an *identity* provider. The other three providers are optional; if an authentication, access, or password provider is not defined, then the identity provider is used for that function.

SSSD can use IdM for all of its back end functions. This is the ideal configuration because it provides the full range of IdM functionality, unlike generic LDAP identity providers or Kerberos authentication. For example, during daily operation, SSSD enforces host-based access control rules and security features in IdM.

Figure 4.2. Clients and SSSD with an IdM back end



The **ipa-client-install** script automatically configures SSSD to use IdM for all its back end services, so that RHEL clients are set up with the recommended configuration by default.

Additional information

- Understanding SSSD and its benefits

4.2.3. Alternative supported configuration

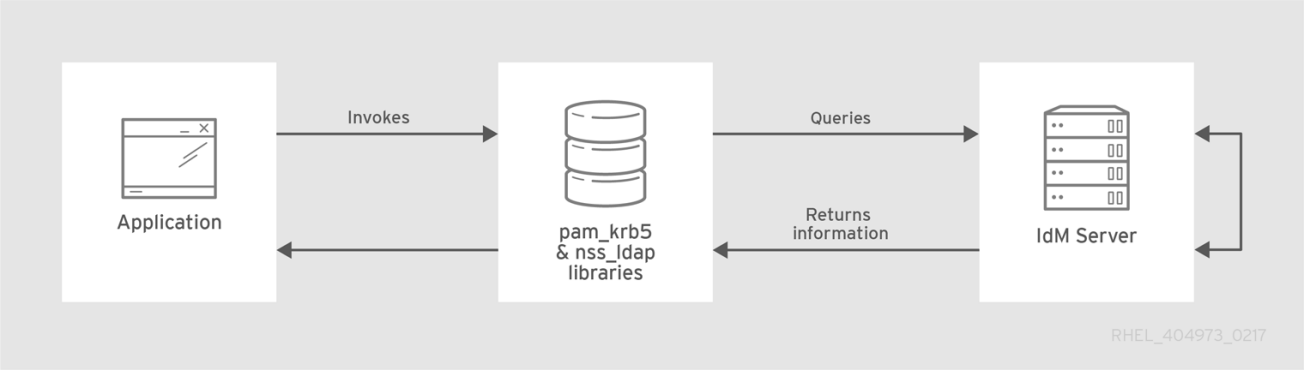
Unix and Linux systems such as Mac, Solaris, HP-UX, AIX, and Scientific Linux support all of the services that Identity Management (IdM) manages but do not use SSSD. Similarly, older Red Hat Enterprise Linux (RHEL) versions, specifically 6.1 and 5.6, support SSSD but have an older version, which does not support IdM as an identity provider.

If it is not possible to use a modern version of SSSD on a system, then clients can be configured in the following way:

- The client connects to the IdM server as if it were an LDAP directory server for identity lookups, by using **nss_ldap**.
- The client connects to the IdM server as if it were a regular Kerberos KDC, by using **pam_krb5**.

For more information about configuring a *RHEL client with an older version of SSSD* to use the IdM server as its identity provider and its Kerberos authentication domain, see the Configuring identity and authentication providers for SSSD section of the RHEL 7 *System-Level Authentication Guide*.

Figure 4.3. Clients and IdM with LDAP and Kerberos



It is generally best practice to use the most secure configuration possible for a client. This means SSSD or LDAP for identities and Kerberos for authentication. However, for some maintenance situations and IT structures, you may need to resort to the simplest possible scenario: configuring LDAP to provide both identity and authentication by using the **nss_ldap** and **pam_ldap** libraries on the clients.

4.3. PLANNING PASSWORD MIGRATION WHEN MIGRATING FROM LDAP TO IDM

A crucial question to answer before migrating users from LDAP to Identity Management (IdM) is whether to migrate user passwords or not. The following options are available:

Migrating users without passwords

Can be performed more quickly but requires more manual work by administrators and users. In certain situations, this is the only available option: for example, if the original LDAP environment stored cleartext user passwords or if the passwords do not meet the password policy requirements defined in IdM.

When migrating user accounts without passwords, you reset all user passwords. The migrated users are assigned a temporary password that they change at the first login. For more information about how to reset passwords, see Changing and resetting user passwords in HREL 7 IdM documentation.

Migrating users with their passwords

Provides a smoother transition but also requires parallel management of LDAP directory and IdM during the migration and transition process. The reason for this is that by default, IdM uses Kerberos for authentication and requires that each user has a Kerberos hash stored in the IdM Directory Server in addition to the standard user password. To generate the hash, the user password needs to be available to the IdM server in clear text. When you create a new user password, the password is available in clear text before it is hashed and stored in IdM. However, when the user is migrated from an LDAP directory, the associated user password is already hashed, so the corresponding Kerberos key cannot be generated.

IMPORTANT

By default, users cannot authenticate to the IdM domain or access IdM resources until they have Kerberos hashes - even if the user accounts already exist. One workaround is available: using LDAP authentication in IdM instead of Kerberos authentication. With this workaround, Kerberos hashes are not required for users. However, this workaround limits the capabilities of IdM and is not recommended.

The following sections explain how to migrate users and their passwords:

- Methods for migrating passwords when migrating LDAP to IdM
  - Using a web page
  - Using SSSD
- Planning the migration of cleartext LDAP passwords
- Planning the migration of LDAP passwords that do not meet the IdM requirements

4.3.1. Methods for migrating passwords when migrating LDAP to IdM

To migrate user accounts from LDAP to Identity Management (IdM) without forcing the users to change their passwords, you can use the following methods:

Method 1: Using the migration web page

Tell users to enter their LDAP credentials once into a special page in the IdM Web UI, ***https://ipaserver.example.com/ipa/migration***. A script running in the background then captures the clear text password and properly updates the user account with the password and an appropriate Kerberos hash.

Method 2 (recommended): Using SSSD

Mitigate the user impact of the migration by using the System Security Services Daemon (SSSD) to generate the required user keys. For deployments with a lot of users or where users should not be burdened with password changes, this is the best scenario.

Workflow

1. A user tries to log into a machine with SSSD.
2. SSSD attempts to perform Kerberos authentication against the IdM server.
3. Even though the user exists in the system, the authentication fails with the error *key type is not supported* because the Kerberos hashes do not exist yet.
4. SSSD performs a plain text LDAP bind over a secure connection.
5. IdM intercepts this bind request. If the user has a Kerberos principal but no Kerberos hashes, then the IdM identity provider generates the hashes and stores them in the user entry.
6. If authentication is successful, SSSD disconnects from IdM and tries Kerberos authentication again. This time, the request succeeds because the hash exists in the entry.

With method 2, the entire process is invisible to the users. They log in to a client service without noticing that their password has been moved from LDAP to IdM.

4.3.2. Planning the migration of cleartext LDAP passwords

Although in most deployments LDAP passwords are stored encrypted, there may be some users or some environments that use cleartext passwords for user entries.

When users are migrated from the LDAP server to the IdM server, their cleartext passwords are not migrated over because IdM does not allow cleartext passwords. Instead, a Kerberos principal is created for each user, the keytab is set to true, and the password is set as expired. This means that IdM requires

the user to reset the password at the next login. For more information, see Planning the migration of LDAP passwords that do not meet the IdM requirements.

4.3.3. Planning the migration of LDAP passwords that do not meet the IdM requirements

If user passwords in the original directory do not meet the password policies defined in Identity Management (IdM), the passwords become invalid after the migration.

Password reset is done automatically the first time a user attempts to obtain a Kerberos ticket-granting ticket (TGT) in the IdM domain by entering **kinit**. The user is forced to change his or her password:

```
[migrated_idm_user@idmclient ~]$ kinit
Password for migrated_idm_user@IDM.EXAMPLE.COM:
Password expired.  You must change it now.
Enter new password:
Enter it again:
```

4.4. FURTHER MIGRATION CONSIDERATIONS AND REQUIREMENTS

As you are planning a migration from an LDAP server to Identity Management (IdM), ensure that your LDAP environment is able to work with the IdM migration script.

4.4.1. LDAP servers supported for migration

The migration process from an LDAP server to IdM uses a special script, **ipa migrate-ds**, to perform the migration. This script has specific requirements regarding the structure of the LDAP directory and LDAP entries. Migration is supported only for LDAPv3-compliant directory services, which include several common directories:

- Sun ONE Directory Server
- Apache Directory Server
- OpenLDAP

Migration from an LDAP server to IdM has been tested with Red Hat Directory Server and OpenLDAP.

NOTE

Migration using the migration script is *not* supported for Microsoft Active Directory because it is not an LDAPv3-compliant directory. For assistance with migrating from Active Directory, contact Red Hat Professional Services.

4.4.2. LDAP environment requirements for migration

Many different possible configuration scenarios exist for LDAP servers and for Identity Management (IdM), which affects the smoothness of the migration process. For the example migration procedures, these are the assumptions about the environment:

- A single LDAP directory domain is being migrated to one IdM realm. No consolidation is involved.
- A user password is stored as a hash in the LDAP directory. For a list of supported hashes, see

the Password Storage Schemes section in the *Configuration, Command, and File Reference* title available in the Red Hat Directory Server 10 section of Red Hat Directory Server Documentation.

- The LDAP directory instance is both the identity store and the authentication method. Client machines are configured to use the **pam_ldap** or **nss_ldap** library to connect to the LDAP server.
- Entries use only the standard LDAP schema. Entries that contain custom object classes or attributes are not migrated to IdM.
- The **migrate-ds** command only migrates the following accounts:
  - Those containing a **gidNumber** attribute. The attribute is required by the **posixAccount** object class.
  - Those containing an **sn** attribute. The attribute is required by the **person** object class.

4.4.3. IdM system requirements for migration

With a moderately-sized directory of around 10,000 users and 10 groups, it is necessary to have a powerful enough target IdM system to allow the migration to proceed. The minimum requirements for a migration are:

- 4 cores
- 4GB of RAM
- 30GB of disk space
- A SASL buffer size of 2MB, which is the default for an IdM server
  In case of migration errors, increase the buffer size:

```
[root@ipaserver ~]# ldapmodify -x -D 'cn=directory manager' -w password -h
ipaserver.example.com -p 389

dn: cn=config
changetype: modify
replace: nsslapd-sasl-max-buffer-size
nsslapd-sasl-max-buffer-size: 4194304

modifying entry "cn=config"
```

  Set the **nsslapd-sasl-max-buffer-size** value in bytes.

Additional resources

- IdM server hardware recommendations

4.4.4. User and group ID numbers

When migrating from LDAP to an IdM deployment, ensure that no user ID (UID) and group ID (GID) conflict exists between the deployments. Before migration, verify that:

- You know your LDAP ID range.

- You know your IdM ID range.
- No overlap exists between UIDs and GIDs on the LDAP server and existing UIDs or GIDs on the RHEL system or IdM deployment.
- The migrated LDAP UIDs and GIDs fit into the IdM ID range.
  - If needed, create a new IdM ID range prior to migration.

Additional resources

- Adding a new IdM ID range

4.4.5. Considerations about sudo rules

If you are using **sudo** with LDAP, you must migrate the **sudo** rules stored in LDAP to Identity Management (IdM) manually. Red Hat recommends that you recreate netgroups in IdM as hostgroups. IdM presents hostgroups automatically as traditional netgroups for **sudo** configurations that do not use the SSSD **sudo** provider.

4.4.6. LDAP to IdM migration tools

Identity Management (IdM) uses a specific command, **ipa migrate-ds**, to execute the migration process so that LDAP directory data are properly formatted and imported cleanly into the IdM server. When using **ipa migrate-ds**, the remote system user, specified by the **--bind-dn** option, must have read access to the **userPassword** attribute, otherwise passwords will not be migrated.

The IdM server must be configured to run in migration mode, and then the migration script can be used. For details, see Migrating an LDAP server to IdM .

4.4.7. Improving LDAP to IdM migration performance

An LDAP migration is essentially a specialized import operation for the 389 Directory Server (DS) instance within the IdM server. Tuning the 389 DS instance for better import operation performance can help improve the overall migration performance.

There are two parameters that directly affect import performance:

- The **nsslapd-cachememsize** attribute, which defines the size allowed for the entry cache. This is a buffer that is automatically set to 80% of the total cache memory size. For large import operations, you can increase this parameter and possibly the memory cache itself. This increase will improve the efficiency of the directory service in handling a large number of entries or entries with large attributes.
  For details on how to modify the attribute using the **dsconf** command, see Adjusting the entry cache size.
- The system **ulimit** configuration option sets the maximum number of allowed processes for a system user. Processing a large database can exceed the limit. If this happens, increase the value:

```
[root@server ~]# ulimit -u 4096
```

Additional resources

- Adjusting IdM Directory Server performance

4.4.8. LDAP to IdM migration sequence

There are four major steps when migrating to IdM, but their order varies depending on whether you want to first migrate the *server* or the *client*s.

IMPORTANT

Both the client-first and server-first migrations provide a general migration procedure, but they may not work in every environment. Set up a test LDAP environment and test the migration process before attempting to migrate the real LDAP environment.

Client-first migration

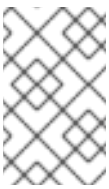
SSSD is used to change the client configuration while an Identity Management (IdM) server is configured:

1. Deploy SSSD.
2. Reconfigure clients to connect to the current LDAP server and then fail over to IdM.
3. Install the IdM server.
4. Migrate the user data using the IdM **ipa migrate-ds** script. This exports the data from the LDAP directory, formats for the IdM schema, and then imports it into IdM.
5. Take the LDAP server offline and allow clients to fail over to IdM transparently.

Server-first migration

The LDAP to IdM migration comes first:

1. Install the IdM server.
2. Migrate the user data using the IdM **ipa migrate-ds** script. This exports the data from the LDAP directory, formats it for the IdM schema, and then imports it into IdM.
3. *Optional.* Deploy SSSD.
4. Reconfigure clients to connect to IdM. It is not possible to simply replace the LDAP server. The IdM directory tree – and therefore user entry DNs – is different from the previous directory tree.
   While it is required that clients must be reconfigured, clients do not need to be reconfigured immediately. Updated clients can point to the IdM server while other clients point to the old LDAP directory, allowing a reasonable testing and transition phase after the data are migrated.



NOTE

Do not run both an LDAP directory service and the IdM server for very long in parallel. This introduces the risk of user data becoming inconsistent between the two services.

4.5. CUSTOMIZING THE MIGRATION FROM LDAP TO IDM

You can migrate your authentication and authorization services from an LDAP server to

Identity Management (IdM) using the **ipa migrate-ds** command. Without additional options, the command takes the LDAP URL of the directory to migrate and exports the data based on common default settings.

You can customize the migration process and how data is identified and exported by using different **ipa migrate-ds** command options. Customize the migration if your LDAP directory tree has a unique structure or if you know you must exclude certain entries or attributes within entries.

4.5.1. Examples of customizing the Bind DN and Base DN during the migration from LDAP to IdM

Use the **ipa migrate-ds** command to migrate from LDAP to Identity Management (IdM). Without additional options, the command takes the LDAP URL of the directory to migrate and exports the data based on common default settings. The following are examples of modifying the default settings:

```
# ipa migrate-ds ldap://ldap.example.com:389
```

Customizing the Bind DN

By default, the DN "**cn=Directory Manager**" is used to bind to the remote LDAP directory. Use the **--bind-dn** option to specify a custom bind DN:

```
# ipa migrate-ds ldap://ldap.example.com:389 --bind-dn=cn=Manager,dc=example,dc=com
```

Customizing the naming context

If the LDAP server naming context differs from the one used in IdM, the base DNs for objects are transformed. For example: **uid=*user*,ou=people,dc=ldap,dc=example,dc=com** is migrated to **uid=*user*,ou=people,dc=idm,dc=example,dc=com**. Using the **--base-dn** option, you can change the target for container subtrees and thus set the base DN used on the remote LDAP server for the migration:

```
# ipa migrate-ds --base-dn="ou=people,dc=example,dc=com" ldap://ldap.example.com:389
```

Additional resources

- **ipa migrate-ds --help**

4.5.2. The migration of specific subtrees

The default directory structure places person entries in the **ou=People** subtree and group entries in the **ou=Groups** subtree. These subtrees are container entries for those different types of directory data. If you do not use any options with the **migrate-ds** command, then the utility assumes that the given LDAP directory uses the **ou=People** and **ou=Groups** structure.

Many deployments may have an entirely different directory structure or you may only want to export certain parts of the original directory tree. As an administrator, you can use the following options to specify the RDN of a different user or group subtree on the source LDAP server:

- **--user-container**
- **--group-container**

NOTE

In both cases, the subtree must be a relative distinguished name (RDN) and must be relative to the base DN. For example, you can migrate the **>ou=Employees,dc=example,dc=com** directory tree by using **--user-container=ou=Employees**.

For example:

```
[ipaserver ~]# ipa migrate-ds --user-container=ou=employees \
--group-container="ou=employee groups" ldap://ldap.example.com:389
```

Optionally, add the **--scope** option to the **ipa migrate-ds** command to set the scope:

- **onelevel**: Default. Only entries in the specified container are migrated.
- **subtree**: Entries in the specified container and all subcontainers are migrated.
- **base**: Only the specified object itself is migrated.

4.5.3. The inclusion and exclusion of entries

By default, the **ipa migrate-ds** script imports every user entry with the **person** object class and every group entry with the **groupOfUniqueNames** or **groupOfNames** object class.

In some migration paths, only specific types of users and groups may need to be exported, or, alternatively, specific users and groups may need to be excluded. You can select which *types* of users and groups to include by setting which object classes to search for when looking for user or group entries.

This option is particularly useful when you use custom object classes for different *user* types. For example, the following command migrates only users with the custom **fullTimeEmployee** object class:

```
[root@ipaserver ~]# ipa migrate-ds --user-objectclass=fullTimeEmployee
ldap://ldap.example.com:389
```

Because of the different types of groups, this is also very useful for migrating only certain types of *groups*, such as user groups, while excluding other types of groups, like certificate groups. For example:

```
[root@ipaserver ~]# ipa migrate-ds --group-objectclass=groupOfNames --group-
objectclass=groupOfUniqueNames ldap://ldap.example.com:389
```

Specifying user and group entries to migrate based on object class implicitly excludes all other users and groups from migration.

Alternatively, it can be useful to migrate all user and group entries except for just a small handful of entries. You can exclude specific user or group accounts while migrating all others of that type. For example, this excludes only a hobbies group and two users:

```
[root@ipaserver ~]# ipa migrate-ds --exclude-groups="Golfers Group" --exclude-
users=idmuser101 --exclude-users=idmuser102 ldap://ldap.example.com:389
```

Exclude statements are applied to users matching the pattern in the **uid** and to groups matching it in the **cn** attribute.

You can migrate a general object class but exclude specific entries of that class. For example, this specifically includes users with the **fullTimeEmployee** object class, yet excludes three managers:

```
[root@ipaserver ~]# ipa migrate-ds --user-objectclass=fullTimeEmployee --exclude-users=jsmith --exclude-users=bjensen --exclude-users=mreynolds ldap://ldap.example.com:389
```

4.5.4. The exclusion of entry attributes

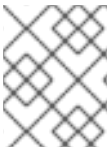
By default, every attribute and object class for a user or group entry is migrated. In certain scenarios, that may not be realistic, either because of bandwidth and network constraints or because the attribute data are no longer relevant. For example, if users are going to be assigned new user certificates as they join the Identity Management (IdM) domain, then migrating the **userCertificate** attribute would be useless.

You can ignore specific object classes and attributes by using the following options with the **migrate-ds** command:

- **--user-ignore-objectclass**
- **--user-ignore-attribute**
- **--group-ignore-objectclass**
- **--group-ignore-attribute**

For example, to exclude the **userCertificate** attribute and **strongAuthenticationUser** object class for users and the **groupOfCertificates** object class for groups:

```
[root@ipaserver ~]# ipa migrate-ds --user-ignore-attribute=userCertificate --user-ignore-objectclass=strongAuthenticationUser --group-ignore-objectclass=groupOfCertificates ldap://ldap.example.com:389
```



NOTE

Make sure not to ignore any required attributes. Also, when excluding object classes, make sure to exclude any attributes that only that object class supports.

Additional resources

- LDAP environment requirements for migration

4.5.5. The schema to use when migrating from LDAP to IdM and the schema compat feature

Identity Management (IdM) uses the RFC2307bis schema to define user, host, host group, and other network identities. However, if the LDAP server used as the source for the migration uses the RFC2307 schema instead, specify the **--schema** option with the **ipa migrate-ds** command:

```
[root@ipaserver ~]# ipa migrate-ds --schema=RFC2307 ldap://ldap.example.com:389
```

Alternatively, IdM has a built-in **schema compat feature** that allows IdM to reformat data for systems that do not support RFC2307bis. The compat plugin is enabled by default, which means that the

directory server computes an alternate view of the users and groups and provides this view in the **cn=users,cn=compat,dc=example,dc=com** container entry. It does this by precomputing the contents of its entries at startup-time and refreshing its entries as needed.

It is recommended that this feature is disabled during the migration to reduce system overhead.

4.6. MIGRATING AN LDAP SERVER TO IDM

You can migrate your authentication and authorization services from an LDAP server to Identity Management (IdM) using the **ipa migrate-ds** command.

WARNING

This is a general migration procedure that may not work in every environment.

It is strongly recommended that you set up a test LDAP environment and test the migration process before attempting to migrate the real LDAP environment. When testing the environment, do the following:

1. Create a test user in IdM and compare the output of migrated users to that of the test user.
2. Compare the output of migrated users, as seen on IdM, to the source users, as seen on the original LDAP server.

For more guidance, see the **Verification** section below.

Prerequisites

- You have administrator privileges to the LDAP directory.
- If IdM is already installed, you have administrator privileges to IdM.
- You are logged in as **root** on the RHEL system on which you are executing the procedure below.
- You have read and understood the following chapters:
  - Considerations in migrating from LDAP to IdM .
  - Planning the client configuration when migrating from LDAP to IdM .
  - Planning password migration when migrating from LDAP to IdM .
  - Further migration considerations and requirements .
  - Customizing the migration from LDAP to IdM .

Procedure

1. If IdM is not yet installed: install the IdM server, including any custom LDAP directory schema, on a different machine from the one on which the existing LDAP directory is installed. For details, see Installing Identity Management .

NOTE

Custom user or group schemas have limited support in IdM. They can cause problems during the migration because of incompatible object definitions.

2. For performance reasons, disable the compat plug-in:

```
# ipa-compat-manage disable
```

For more information about the schema compat feature and the benefits of disabling it for the migration, see The schema to use when migrating from LDAP to IdM and the schema compat feature.

3. Restart the IdM Directory Server instance:

```
# systemctl restart dirsrv.target
```

4. Configure the IdM server to allow migration:

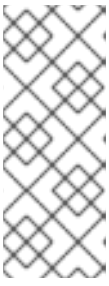
```
# ipa config-mod --enable-migration=TRUE
```

By setting **--enable-migration** to TRUE, you do the following:

- Allow pre-hashed passwords during an LDAP add operation.
- Configure SSSD to try the password migration sequence if the initial Kerberos authentication fails. For more information, see the Workflow section in Using SSSD when migrating passwords from LDAP to IdM.

5. Run the IdM migration script, **ipa migrate-ds**, with the options that are relevant for your use case. For more information, see Customizing the migration from LDAP to IdM .

```
# ipa migrate-ds --your-options ldap://ldap.example.com:389
```



NOTE

If you did not disable the compat plug-in in one of the previous steps, add the **--with-compat** option to **ipa migrate-ds**:

```
# ipa migrate-ds --your-options --with-compat
ldap://ldap.example.com:389
```

6. Re-enable the compat plug-in:

```
# ipa-compat-manage enable
```

7. Restart the IdM Directory Server:

```
# systemctl restart dirsrv.target
```

8. When all users have had their passwords migrated, disable the migration mode:

```
# ipa config-mod --enable-migration=FALSE
```

9. [Optional] When all of the users have been migrated, reconfigure non-SSSD clients to use Kerberos authentication, that is **pam_krb5**, instead of LDAP authentication, that is **pam_ldap**. For more information, see Configuring a Kerberos Client in the RHEL 7 *System-level Authentication Guide*.

10. Have users generate their hashed Kerberos passwords. Choose one of the methods described in Planning password migration when migrating from LDAP to IdM .

    - If you decide on the SSSD method:
        - Move clients that have SSSD installed from the LDAP directory to the IdM directory, and enroll them as clients with IdM. This downloads the required keys and certificates. On Red Hat Enterprise Linux clients, this can be done using the **ipa-client-install** command. For example:

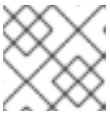
```
# ipa-client-install --enable-dns-update
```

    - If you decide on the IdM migration web page method:
        - Instruct users to log into IdM using the migration web page:

```
https://ipaserver.example.com/ipa/migration
```

11. To monitor the user migration process, query the existing LDAP directory to see which user accounts have a password but do not yet have a Kerberos principal key.

```
$ ldapsearch -LL -x -D 'cn=Directory Manager' -w secret -b
'cn=users,cn=accounts,dc=example,dc=com' '(&(!(krbprincipalkey=))(userpassword=))'
uid
```



NOTE

Include the single quotes around the filter so that it is not interpreted by the shell.

12. When the migration of all clients and users is complete, decommission the LDAP directory.

Verification

1. Create a test user in IdM by using the **ipa user-add** command. Compare the output of migrated users to that of the test user. Ensure that the migrated users contain the minimal set of attributes and object classes present on the test user. For example:

```
$ ipa user-show --all testing_user
dn: uid=testing_user,cn=users,cn=accounts,dc=idm,dc=example,dc=com
User login: testing_user
First name: testing
Last name: user
```

```
Full name: testing user
Display name: testing user
Initials: tu
Home directory: /home/testing_user
GECOS: testing user
Login shell: /bin/sh
Principal name: testing_user@IDM.EXAMPLE.COM
Principal alias: testing_user@IDM.EXAMPLE.COM
Email address: testing_user@idm.example.com
UID: 1689700012
GID: 1689700012
Account disabled: False
Preserved user: False
Password: False
Member of groups: ipausers
Kerberos keys available: False
ipauniqueid: 843b1ac8-6e38-11ec-8dfe-5254005aad3e
mepmanagedentry: cn=testing_user,cn=groups,cn=accounts,dc=idm,dc=example,dc=com
objectclass: top, person, organizationalperson, inetorgperson, inetuser, posixaccount,
krbprincipalaux, krbticketpolicyaux, ipaobject,
        ipasshuser, ipaSshGroupOfPubKeys, mepOriginEntry
```

2. Compare the output of migrated users, as seen on IdM, to the source users, as seen on the original LDAP server. Ensure that imported attributes are not copied twice and that they have the correct values.

Additional resources

- Migrating from LDAP to IdM over SSL

4.7. MIGRATING FROM LDAP TO IDM OVER SSL

You can migrate your authentication and authorization services from an LDAP server to Identity Management (IdM) using the **ipa migrate-ds** command. Follow this procedure to encrypt the data transmitted during the migration.

WARNING

This is a general migration procedure that may not work in every environment.

It is strongly recommended that you set up a test LDAP environment and test the migration process before attempting to migrate the real LDAP environment. When testing the environment, do the following:

1. Create a test user in IdM and compare the output of migrated users to that of the test user.
2. Compare the output of migrated users, as seen on IdM, to the source users, as seen on the original LDAP server.

For more guidance, see the **Verification** section below.

Prerequisites

- You have administrator privileges to the LDAP directory.
- If IdM is already installed, you have administrator privileges to IdM.
- You are logged in as **root** on the RHEL system on which you are executing the procedure below.
- You have read and understood the following chapters:
  - Considerations in migrating from LDAP to IdM .
  - Planning the client configuration when migrating from LDAP to IdM .
  - Planning password migration when migrating from LDAP to IdM .
  - Further migration considerations and requirements .
  - Customizing the migration from LDAP to IdM .

Procedure

1. Store the certificate of the CA that issued the remote LDAP server certificate in a file on the future IdM server. For example: **/tmp/remote.crt**.
2. Follow the steps described in Migrating an LDAP server to IdM . However, for an encrypted LDAP connection during the migration, use the **ldaps** protocol in the URL and pass the **--ca-cert-file** option to the **ipa migrate-ds** command. For example:

```
# ipa migrate-ds --ca-cert-file=/tmp/remote.crt --your-other-options
ldaps://ldap.example.com:636
```

Verification

1. Create a test user in IdM by using the **ipa user-add** command. Compare the output of migrated users to that of the test user. Ensure that the migrated users contain the minimal set of attributes and object classes present on the test user. For example:

```
$ ipa user-show --all testing_user
dn: uid=testing_user,cn=users,cn=accounts,dc=idm,dc=example,dc=com
User login: testing_user
First name: testing
Last name: user
Full name: testing user
Display name: testing user
Initials: tu
Home directory: /home/testing_user
GECOS: testing user
Login shell: /bin/sh
Principal name: testing_user@IDM.EXAMPLE.COM
Principal alias: testing_user@IDM.EXAMPLE.COM
Email address: testing_user@idm.example.com
UID: 1689700012
GID: 1689700012
Account disabled: False
Preserved user: False
```

```
Password: False
Member of groups: ipausers
Kerberos keys available: False
ipauniqueid: 843b1ac8-6e38-11ec-8dfe-5254005aad3e
mepmanagedentry: cn=testing_user,cn=groups,cn=accounts,dc=idm,dc=example,dc=com
objectclass: top, person, organizationalperson, inetorgperson, inetuser, posixaccount,
krbprincipalaux, krbticketpolicyaux, ipaobject,
        ipasshuser, ipaSshGroupOfPubKeys, mepOriginEntry
```

2. Compare the output of migrated users, as seen on IdM, to the source users, as seen on the original LDAP server. Ensure that imported attributes are not copied twice and that they have the correct values.

PART III. MIGRATING AN EXISTING ENVIRONMENT FROM SYNCHRONIZATION TO TRUST IN THE CONTEXT OF INTEGRATING A LINUX DOMAIN WITH AN ACTIVE DIRECTORY DOMAIN

In RHEL 7, *synchronization* and *trust* were two possible approaches to indirect integration of RHEL systems to Active Directory (AD). In RHEL 8, synchronization is deprecated. If you have configured synchronisation between RHEL Identity Management (IdM) and AD in RHEL 7, Red Hat recommends migrating to the approach based on an IdM-AD trust instead.

This chapter describes how to migrate an existing synchronization-based setup to AD trust. The following migrating options are available in IdM:

- Migrating from synchronization to trust automatically, by using ipa-winsync-migrate
- Migrating from synchronization to trust manually, by using ID views

CHAPTER 5. MIGRATING FROM SYNCHRONIZATION TO TRUST AUTOMATICALLY BY USING IPA-WINSYNC-MIGRATE

In RHEL 8, the synchronization approach to integrating RHEL systems into Active Directory (AD) indirectly is deprecated. Red Hat recommends migrating to the approach based on a trust between Identity Management (IdM) and AD instead. This chapter describes how to migrate from synchronization to trust automatically, by using the **ipa-winsync-migrate** utility.

5.1. AUTOMATIC MIGRATION FROM SYNCHRONIZATION TO TRUST BY USING IPA-WINSYNC-MIGRATE

The **ipa-winsync-migrate** utility migrates all synchronized users from an AD forest, while preserving the existing configuration in the Winsync environment and transferring it into the AD trust. For each AD user created by the Winsync agreement, **ipa-winsync-migrate** creates an ID override in the Default Trust View.

After the migration completes:

- The ID overrides for the AD users have the following attributes copied from the original entry in Winsync:
  - Login name (**uid**)
  - UID number (**uidnumber**)
  - GID number (**gidnumber**)
  - Home directory (**homedirectory**)
  - GECOS entry (**gecos**)
- The user accounts in the AD trust keep their original configuration in IdM, which includes:
  - POSIX attributes
  - User groups
  - Role-based access control rules
  - Host-based access control rules
  - SELinux membership
  - **sudo** rules
- The new AD users are added as members of an external IdM group.
- The original Winsync replication agreement, the original synchronized user accounts, and all local copies of the user accounts are removed.

Additional resources

- How the Default Trust View works

5.2. MIGRATING FROM SYNCHRONIZATION TO TRUST BY USING IPA-WINSYNC-MIGRATE

Prerequisites

- On RHEL 7, you configured synchronisation between RHEL Identity Management (IdM) and AD.

Procedure

1. Back up your IdM setup using the **ipa-backup** utility. See Backing up and restoring IdM.

   NOTE

   The migration affects a significant part of the IdM configuration and many user accounts. Creating a backup enables you to restore your original setup if necessary.

2. Create a trust with the synchronized domain. See For details, see Installing trust between IdM and AD.

3. Run **ipa-winsync-migrate** and specify the AD realm and the host name of the AD domain controller:

   ```
   # ipa-winsync-migrate --realm example.com --server ad.example.com
   ```

   If a conflict occurs in the overrides created by **ipa-winsync-migrate**, information about the conflict is displayed, but the migration continues.

4. Uninstall the Password Sync service from the AD server. This removes the synchronization agreement from the AD domain controllers.

Additional resources

- **ipa-winsync-migrate**(1)
- Backing up and restoring IdM servers using Ansible playbooks

CHAPTER 6. MIGRATING FROM SYNCHRONIZATION TO TRUST MANUALLY BY USING ID VIEWS

In RHEL 8, the synchronization approach to integrating RHEL systems into Active Directory (AD) indirectly is deprecated. Red Hat recommends migrating to the approach based on a trust between Identity Management (IdM) and AD instead. This chapter describes how to migrate from synchronization to trust manually, by using ID views.

6.1. MIGRATING FROM SYNCHRONIZATION TO TRUST MANUALLY USING ID VIEWS

You can use ID views to manually change the POSIX attributes that Active Directory (AD) previously generated for AD users.

Prerequisites

- On RHEL 7, you configured synchronisation between RHEL Identity Management (IdM) and AD.

Procedure

1. Create a backup of the original synchronized user and group entries.
2. Create a trust with the synchronized domain. See For details, see Installing trust between IdM and AD.
3. For every synchronized user or group, preserve the UID and GIDs generated by IdM by doing one of the following:
   - Individually create an ID view applied to the specific host and add user ID overrides to the view.
   - Create user ID overrides in the Default Trust View.

   NOTE

   Only IdM users can manage ID views. AD users cannot.
4. Delete the original synchronized user or group entries.

Additional resources

- Using ID views for Active Directory users
- Defining a different attribute value for a user account on different hosts