Red Hat Enterprise Linux 8

Integrating RHEL systems directly with Windows Active Directory

Joining RHEL hosts to AD and accessing resources in AD

Last Updated: 2024-04-08

Red Hat Enterprise Linux 8 Integrating RHEL systems directly with Windows Active Directory

Joining RHEL hosts to AD and accessing resources in AD

Legal Notice

Abstract

Administrators can join Red Hat Enterprise Linux (RHEL) hosts to an Active Directory (AD) domain by using the System Security Services Daemon (SSSD) or the Samba Winbind service to access AD resources. Alternatively, it is also possible to access AD resources without domain integration by using a Managed Service Account (MSA).

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.

CHAPTER 1. CONNECTING RHEL SYSTEMS DIRECTLY TO AD USING SSSD

You need two components to connect a RHEL system to Active Directory (AD). One component, SSSD, interacts with the central identity and authentication source, and the other component, **realmd**, detects available domains and configures the underlying RHEL system services, in this case SSSD, to connect to the domain.

This section describes using the System Security Services Daemon (SSSD) to connect a RHEL system to Active Directory (AD).

- Overview of direct integration using SSSD
- Supported Windows platforms for direct integration
- Ensuring support for common encryption types in AD and RHEL
- Connecting directly to AD
- How the AD provider handles dynamic DNS updates
- Modifying dynamic DNS settings for the AD provider
- How the AD provider handles trusted domains
- Overriding Active Directory site autodiscovery with SSSD
- realm commands

1.1. OVERVIEW OF DIRECT INTEGRATION USING SSSD

You use SSSD to access a user directory for authentication and authorization through a common framework with user caching to permit offline logins. SSSD is highly configurable; it provides Pluggable Authentication Modules (PAM) and Name Switch Service (NSS) integration and a database to store local users as well as extended user data retrieved from a central server. SSSD is the recommended component to connect a RHEL system with one of the following types of identity server:

- Active Directory
- Identity Management (IdM) in RHEL
- Any generic LDAP or Kerberos server

NOTE

Direct integration with SSSD works only within a single AD forest by default.

The most convenient way to configure SSSD to directly integrate a Linux system with AD is to use the **realmd** service. It allows callers to configure network authentication and domain membership in a standard way. The **realmd** service automatically discovers information about accessible domains and realms and does not require advanced configuration to join a domain or realm.

You can use SSSD for both direct and indirect integration with AD and it allows you to switch from one integration approach to another. Direct integration is a simple way to introduce RHEL systems to an AD environment. However, as the share of RHEL systems grows, your deployments usually need a better

centralized management of the identity-related policies such as host-based access control, sudo, or SELinux user mappings. Initially, you can maintain the configuration of these aspects of the RHEL systems in local configuration files. However, with a growing number of systems, distribution and management of the configuration files is easier with a provisioning system such as Red Hat Satellite. When direct integration does not scale anymore, you should consider indirect integration. For more information about moving from direct integration (RHEL clients are in the AD domain) to indirect integration (IdM with trust to AD), see Moving RHEL clients from AD domain to IdM Server.

For more information about which type of integration fits your use case, see Deciding between indirect and direct integration.

Additional resources

- The **realm(8)** man page.
- The **sssd-ad(5)** man page.
- The **sssd(8)** man page.

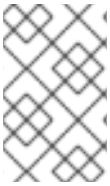
1.2. SUPPORTED WINDOWS PLATFORMS FOR DIRECT INTEGRATION

You can directly integrate your RHEL system with Active Directory forests that use the following forest and domain functional levels:

- Forest functional level range: Windows Server 2008 - Windows Server 2016
- Domain functional level range: Windows Server 2008 - Windows Server 2016

Direct integration has been tested on the following supported operating systems:

- Windows Server 2022 (RHEL 8.7 and above)
- Windows Server 2019
- Windows Server 2016
- Windows Server 2012 R2



NOTE

Windows Server 2019 and Windows Server 2022 do not introduce a new functional level. The highest functional level Windows Server 2019 and Windows Server 2022 use is Windows Server 2016.

1.3. ENSURING SUPPORT FOR COMMON ENCRYPTION TYPES IN AD AND RHEL

By default, Identity Management establishes a cross-realm trust with support for RC4, AES-128, and AES-256 Kerberos encryption types. Additionally, by default SSSD and Samba Winbind support RC4, AES-128, and AES-256 Kerberos encryption types.

RC4 encryption has been deprecated and disabled by default, as it is considered less secure than the newer AES-128 and AES-256 encryption types. In contrast, Active Directory (AD) user credentials and trusts between AD domains support RC4 encryption and they might not support AES encryption types.

Without any common encryption types, communication between RHEL hosts and AD domains might not work, or some AD accounts might not be able to authenticate. To address this situation, modify one of the configurations outlined below.

1.3.1. Enabling AES encryption in AD (recommended)

To ensure trusts between Active Directory (AD) domains in an AD forest support strong AES encryption types, see the following Microsoft article: AD DS: Security: Kerberos "Unsupported etype" error when accessing a resource in a trusted domain

1.3.2. 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 features on RHEL, such as running a Samba server on an IdM client, require this encryption type.

Note that RHEL no longer supports 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.
3. Navigate to **Computer Configuration → Policies → Windows Settings → Security Settings → Local Policies → Security Options**.
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
```

1.3.3. Enabling RC4 support in RHEL

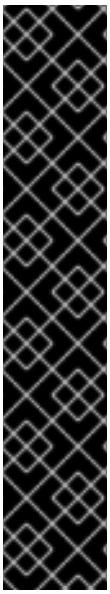
On every RHEL host where authentication against AD Domain Controllers takes place, complete the steps outlined below.

Procedure

1. Use the **update-crypto-policies** command to enable the **AD-SUPPORT** cryptographic subpolicy in addition to the **DEFAULT** cryptographic policy.

```
[root@host ~]# update-crypto-policies --set DEFAULT:AD-SUPPORT
Setting system policy to DEFAULT:AD-SUPPORT
Note: System-wide crypto policies are applied on application start-up.
It is recommended to restart the system for the change of policies
to fully take place.
```

2. Restart the host.



IMPORTANT

The **AD-SUPPORT** cryptographic subpolicy is only available on RHEL 8.3 and newer.

- To enable support for RC4 in RHEL 8.2, create and enable a custom cryptographic module policy with **cipher = RC4-128+**. For more details, see Customizing system-wide cryptographic policies with subpolicies.
- To enable support for RC4 in RHEL 8.0 and RHEL 8.1, add **+rc4** to the **permitted_enctypes** option in the **/etc/crypto-policies/back-ends/krb5.config** file:

```
[libdefaults]
permitted_enctypes = aes256-cts-hmac-sha1-96 aes256-cts-hmac-sha384-
192 camellia256-cts-cmac aes128-cts-hmac-sha1-96 aes128-cts-hmac-
sha256-128 camellia128-cts-cmac +rc4
```

1.3.4. Additional resources

- See Using system-wide cryptographic policies.
- See Trust controllers and trust agents.

1.4. CONNECTING DIRECTLY TO AD

The System Security Services Daemon (SSSD) is the recommended component to connect a Red Hat Enterprise Linux (RHEL) system with Active Directory (AD). This section describes how to integrate directly with AD by using either ID mapping, which is the default for SSSD, or by using POSIX attributes.

- Options for integrating with AD: using ID mapping or POSIX attributes
- Discovering and joining an AD domain using SSSD
- Connecting to AD using POSIX attributes defined in Active Directory
- Connecting to multiple domains in different AD forests with SSSD

IMPORTANT

Before joining your system to AD, ensure you configured your system correctly by following the procedure in Basic Prechecks Steps: RHEL Join With Active Directory using 'adcli', 'realm' and 'net' commands.

1.4.1. Options for integrating with AD: using ID mapping or POSIX attributes

Linux and Windows systems use different identifiers for users and groups:

- Linux uses *user IDs* (UID) and *group IDs* (GID). See Introduction to managing user and group accounts in *Configuring Basic System Settings*. Linux UIDs and GIDs are compliant with the POSIX standard.
- Windows use *security IDs* (SID).

IMPORTANT

After connecting a RHEL system to AD, you can authenticate with your AD username and password. Do not create a Linux user with the same name as a Windows user, as duplicate names might cause a conflict and interrupt the authentication process.

To authenticate to a RHEL system as an AD user, you must have a UID and GID assigned. SSSD provides the option to integrate with AD either using ID mapping or POSIX attributes. The default is to use ID mapping.

Automatically generate new UIDs and GIDs for AD users

SSSD can use the SID of an AD user to algorithmically generate POSIX IDs in a process called *ID mapping*. ID mapping creates a map between SIDs in AD and IDs on Linux.

- When SSSD detects a new AD domain, it assigns a range of available IDs to the new domain.
- When an AD user logs in to an SSSD client machine for the first time, SSSD creates an entry for the user in the SSSD cache, including a UID based on the user's SID and the ID range for that domain.
- Because the IDs for an AD user are generated in a consistent way from the same SID, the user has the same UID and GID when logging in to any Red Hat Enterprise Linux system.

See Discovering and joining an AD domain using SSSD .

NOTE

When all client systems use SSSD to map SIDs to Linux IDs, the mapping is consistent. If some clients use different software, choose one of the following:

- Ensure that the same mapping algorithm is used on all clients.
- Use explicit POSIX attributes defined in AD.

Use POSIX attributes defined in AD

AD can create and store POSIX attributes, such as **uidNumber**, **gidNumber**, **unixHomeDirectory**, or **loginShell**.

When using ID mapping described above, SSSD creates new UIDs and GIDs, which overrides the values defined in AD. To keep the AD-defined values, you must disable ID mapping in SSSD.

See Connecting to AD using POSIX attributes defined in Active Directory .

1.4.2. Discovering and joining an AD Domain using SSSD

Follow this procedure to discover an AD domain and connect a RHEL system to that domain using SSSD.

Prerequisites

- Ensure that the following ports on the AD domain controllers are open and accessible to the RHEL host.

  Table 1.1. Ports Required for Direct Integration of Linux Systems into AD Using SSSD

Service	Port	Protocol	Notes
DNS	53	UDP and TCP	
LDAP	389	UDP and TCP	
Samba	445	UDP and TCP	For AD Group Policy Objects (GPOs)
Kerberos	88	UDP and TCP	
Kerberos	464	UDP and TCP	Used by **kadmin** for setting and changing a password
LDAP Global Catalog	3268	TCP	If the **id_provider = ad** option is being used
NTP	123	UDP	Optional

- Ensure that you are using the AD domain controller server for DNS.
- Verify that the system time on both systems is synchronized. This ensures that Kerberos is able to work correctly.

Procedure

1. Install the following packages:

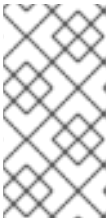
```
# yum install samba-common-tools realmd oddjob oddjob-mkhomedir sssd adcli krb5-workstation
```

2. To display information for a specific domain, run **realm discover** and add the name of the domain you want to discover:

```
# realm discover ad.example.com
ad.example.com
  type: kerberos
  realm-name: AD.EXAMPLE.COM
  domain-name: ad.example.com
  configured: no
```

```
server-software: active-directory
client-software: sssd
required-package: oddjob
required-package: oddjob-mkhomedir
required-package: sssd
required-package: adcli
required-package: samba-common
```

The **realmd** system uses DNS SRV lookups to find the domain controllers in this domain automatically.



NOTE

The **realmd** system can discover both Active Directory and Identity Management domains. If both domains exist in your environment, you can limit the discovery results to a specific type of server using the **--server-software=active-directory** option.

3. Configure the local RHEL system with the **realm join** command. The **realmd** suite edits all required configuration files automatically. For example, for a domain named **ad.example.com**:

```
# realm join ad.example.com
```

Verification steps

- Display an AD user details, such as the administrator user:

```
# getent passwd administrator@ad.example.com
administrator@ad.example.com:*:1450400500:1450400513:Administrator:/home/administrator
@ad.example.com:/bin/bash
```

Additional resources

- See the **realm(8)** man page.
- See the **nmcli(1)** man page.

1.4.3. Connecting to AD using POSIX attributes defined in Active Directory

For best performance, publish the POSIX attributes to the AD global catalog. If POSIX attributes are not present in the global catalog, SSSD connects to the individual domain controllers directly on the LDAP port.

Prerequisites

- Ensure that the following ports on the RHEL host are open and accessible to the AD domain controllers.

Table 1.2. Ports Required for Direct Integration of Linux Systems into AD Using SSSD

Service	Port	Protocol	Notes
DNS	53	UDP and TCP	
LDAP	389	UDP and TCP	
Kerberos	88	UDP and TCP	
Kerberos	464	UDP and TCP	Used by kadmin for setting and changing a password
LDAP Global Catalog	3268	TCP	If the **id_provider = ad** option is being used
NTP	123	UDP	Optional

- Ensure that you are using the AD domain controller server for DNS.
- Verify that the system time on both systems is synchronized. This ensures that Kerberos is able to work correctly.

Procedure

1. Install the following packages:

```
# yum install realmd oddjob oddjob-mkhomedir sssd adcli krb5-workstation
```

2. Configure the local RHEL system with ID mapping disabled using the **realm join** command with the **--automatic-id-mapping=no** option. The **realmd** suite edits all required configuration files automatically. For example, for a domain named **ad.example.com**:

```
# realm join --automatic-id-mapping=no ad.example.com
```

3. If you already joined a domain, you can manually disable ID Mapping in SSSD:
   a. Open the **/etc/sssd/sssd.conf** file.
   b. In the AD domain section, add the **ldap_id_mapping = false** setting.
   c. Remove the SSSD caches:

```
rm -f /var/lib/sss/db/*
```

   d. Restart SSSD:

```
systemctl restart sssd
```

SSSD now uses POSIX attributes from AD, instead of creating them locally.

NOTE

You must have the relevant POSIX attributes (**uidNumber**, **gidNumber**, **unixHomeDirectory**, and **loginShell**) configured for the users in AD.

Verification steps

- Display an AD user details, such as the administrator user:

```
# getent passwd administrator@ad.example.com
administrator@ad.example.com:*:10000:10000:Administrator:/home/Administrator:/bin/bash
```

Additional resources

- For further details about ID mapping and the **ldap_id_mapping** parameter, see the **sssd-ldap(8)** man page.

1.4.4. Connecting to multiple domains in different AD forests with SSSD

You can use an Active Directory (AD) Managed Service Account (MSA) to access AD domains from different forests where there is no trust between them.

See Accessing AD with a Managed Service Account.

1.5. HOW THE AD PROVIDER HANDLES DYNAMIC DNS UPDATES

Active Directory (AD) actively maintains its DNS records by timing out (*aging*) and removing (*scavenging*) inactive records.

By default, the SSSD service refreshes a RHEL client's DNS record at the following intervals:

- Every time the identity provider comes online.
- Every time the RHEL system reboots.
- At the interval specified by the **dyndns_refresh_interval** option in the **/etc/sssd/sssd.conf** configuration file. The default value is **86400** seconds (24 hours).

NOTE

If you set the **dyndns_refresh_interval** option to the same interval as the DHCP lease, you can update the DNS record after the IP lease is renewed.

SSSD sends dynamic DNS updates to the AD server using Kerberos/GSSAPI for DNS (GSS-TSIG). This means that you only need to enable secure connections to AD.

Additional resources

- The **sssd-ad(5)** man page.

1.6. MODIFYING DYNAMIC DNS SETTINGS FOR THE AD PROVIDER

The System Security Services Daemon (SSSD) service refreshes the DNS record of a Red Hat Enterprise Linux (RHEL) client joined to an AD environment at default intervals. The following procedure adjusts these intervals.

Prerequisites

- You have joined a RHEL host to an Active Directory environment with the SSSD service.
- You need **root** permissions to edit the **/etc/sssd/sssd.conf** configuration file.

Procedure

1. Open the **/etc/sssd/sssd.conf** configuration file in a text editor.
2. Add the following options to the **[domain]** section for your AD domain to set the DNS record refresh interval to 12 hours, disable updating PTR records, and set the DNS record Time To Live (TTL) to 1 hour.

```
[domain/ad.example.com]
id_provider = ad
...
dyndns_refresh_interval = 43200
dyndns_update_ptr = false
dyndns_ttl = 3600
```

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

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

NOTE

You can disable dynamic DNS updates by setting the **dyndns_update** option in the **sssd.conf** file to **false**:

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

dyndns_update = false
```

Additional resources

- How the AD provider handles dynamic DNS updates
- **sssd-ad(5)** man page

1.7. HOW THE AD PROVIDER HANDLES TRUSTED DOMAINS

If you set the **id_provider = ad** option in the **/etc/sssd/sssd.conf** configuration file, SSSD handles trusted domains as follows:

- SSSD only supports domains in a single AD forest. If SSSD requires access to multiple domains from multiple forests, consider using IPA with trusts (preferred) or the **winbindd** service instead of SSSD.

- By default, SSSD discovers all domains in the forest and, if a request for an object in a trusted domain arrives, SSSD tries to resolve it.
  If the trusted domains are not reachable or geographically distant, which makes them slow, you can set the **ad_enabled_domains** parameter in **/etc/sssd/sssd.conf** to limit from which trusted domains SSSD resolves objects.

- By default, you must use fully-qualified user names to resolve users from trusted domains.

Additional resources

- The **sssd.conf(5)** man page.

1.8. OVERRIDING ACTIVE DIRECTORY SITE AUTODISCOVERY WITH SSSD

Active Directory (AD) forests can be very large, with numerous different domain controllers, domains, child domains and physical sites. AD uses the concept of **sites** to identify the physical location for its domain controllers. This enables clients to connect to the domain controller that is geographically closest, which increases client performance.

This section describes how SSSD uses autodiscovery to find an AD site to connect to, and how you can override autodiscovery and specify a site manually.

1.8.1. How SSSD handles AD site autodiscovery

By default, SSSD clients use autodiscovery to find its AD site and connect to the closest domain controller. The process consists of these steps:

1. SSSD performs an SRV query to find Domain Controllers (DCs) in the domain. SSSD reads the discovery domain from the **dns_discovery_domain** or the **ad_domain** options in the SSSD configuration file.

2. SSSD performs Connection-Less LDAP (CLDAP) pings to these DCs in 3 batches to avoid pinging too many DCs and avoid timeouts from unreachable DCs. If SSSD receives site and forest information during any of these batches, it skips the rest of the batches.

3. SSSD creates and saves a list of site-specific and backup servers.

1.8.2. Overriding AD site autodiscovery

To override the autodiscovery process, specify the AD site to which you want the client to connect by adding the **ad_site** option to the **[domain]** section of the **/etc/sssd/sssd.conf** file. This example configures the client to connect to the **ExampleSite** AD site.

Prerequisites

- You have joined a RHEL host to an Active Directory environment using the SSSD service.

- You can authenticate as the **root** user so you can edit the **/etc/sssd/sssd.conf** configuration file.

Procedure

1. Open the **/etc/sssd/sssd.conf** file in a text editor.
2. Add the **ad_site** option to the **[domain]** section for your AD domain:

```
[domain/ad.example.com]
id_provider = ad
...
ad_site = ExampleSite
```

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

```
# systemctl restart sssd
```

1.9. REALM COMMANDS

The **realmd** system has two major task areas:

- Managing system enrollment in a domain.
- Controlling which domain users are allowed to access local system resources.

In **realmd** use the command line tool **realm** to run commands. Most **realm** commands require the user to specify the action that the utility should perform, and the entity, such as a domain or user account, for which to perform the action.

Table 1.3. realmd Commands

Command	Description
Realm Commands	
discover	Run a discovery scan for domains on the network.
join	Add the system to the specified domain.
leave	Remove the system from the specified domain.
list	List all configured domains for the system or all discovered and configured domains.
Login Commands	
permit	Enable access for specific users or for all users within a configured domain to access the local system.
deny	Restrict access for specific users or for all users within a configured domain to access the local system.

Additional resources

- The **realm(8)** man page.

CHAPTER 2. CONNECTING RHEL SYSTEMS DIRECTLY TO AD USING SAMBA WINBIND

You need two components to connect a RHEL system to AD. One component, Samba Winbind, interacts with the AD identity and authentication source, and the other component, **realmd**, detects available domains and configures the underlying RHEL system services, in this case Samba Winbind, to connect to the AD domain.

This section describes using Samba Winbind to connect a RHEL system to Active Directory (AD).

- Overview of direct integration using Samba Winbind
- Supported Windows platforms for direct integration
- Ensuring support for common encryption types in AD and RHEL
- Joining a RHEL system to an AD domain
- realm commands

2.1. OVERVIEW OF DIRECT INTEGRATION USING SAMBA WINBIND

Samba Winbind emulates a Windows client on a Linux system and communicates with AD servers.

You can use the **realmd** service to configure Samba Winbind by:

- Configuring network authentication and domain membership in a standard way.
- Automatically discovering information about accessible domains and realms.
- Not requiring advanced configuration to join a domain or realm.

Note that:

- Direct integration with Winbind in a multi-forest AD setup requires bidirectional trusts.
- Remote forests must trust the local forest to ensure that the **idmap_ad** plug-in handles remote forest users correctly.

Samba's **winbindd** service provides an interface for the Name Service Switch (NSS) and enables domain users to authenticate to AD when logging into the local system.

Using **winbindd** provides the benefit that you can enhance the configuration to share directories and printers without installing additional software. For further detail, see the section about Using Samba as a server in the Deploying Different Types of Servers Guide .

Additional resources

- See the **realmd** man page.
- See the **winbindd** man page.

2.2. SUPPORTED WINDOWS PLATFORMS FOR DIRECT INTEGRATION

You can directly integrate your RHEL system with Active Directory forests that use the following forest and domain functional levels:

- Forest functional level range: Windows Server 2008 - Windows Server 2016
- Domain functional level range: Windows Server 2008 - Windows Server 2016

Direct integration has been tested on the following supported operating systems:

- Windows Server 2022 (RHEL 8.7 and above)
- Windows Server 2019
- Windows Server 2016
- Windows Server 2012 R2

NOTE

Windows Server 2019 and Windows Server 2022 do not introduce a new functional level. The highest functional level Windows Server 2019 and Windows Server 2022 use is Windows Server 2016.

2.3. JOINING A RHEL SYSTEM TO AN AD DOMAIN

Samba Winbind is an alternative to the System Security Services Daemon (SSSD) for connecting a Red Hat Enterprise Linux (RHEL) system with Active Directory (AD). You can join a RHEL system to an AD domain by using **realmd** to configure Samba Winbind.

Procedure

1. If your AD requires the deprecated RC4 encryption type for Kerberos authentication, enable support for these ciphers in RHEL:

```
# update-crypto-policies --set DEFAULT:AD-SUPPORT
```

2. Install the following packages:

```
# yum install realmd oddjob-mkhomedir oddjob samba-winbind-clients \
    samba-winbind samba-common-tools samba-winbind-krb5-locator
```

3. To share directories or printers on the domain member, install the **samba** package:

```
# yum install samba
```

4. Backup the existing **/etc/samba/smb.conf** Samba configuration file:

```
# mv /etc/samba/smb.conf /etc/samba/smb.conf.bak
```

5. Join the domain. For example, to join a domain named **ad.example.com**:

```
# realm join --membership-software=samba --client-software=winbind ad.example.com
```

   Using the previous command, the **realm** utility automatically:

- Creates a **/etc/samba/smb.conf** file for a membership in the **ad.example.com** domain
- Adds the **winbind** module for user and group lookups to the **/etc/nsswitch.conf** file
- Updates the Pluggable Authentication Module (PAM) configuration files in the **/etc/pam.d/** directory
- Starts the **winbind** service and enables the service to start when the system boots

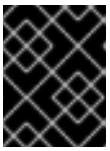
6. Optionally, set an alternative ID mapping back end or customized ID mapping settings in the **/etc/samba/smb.conf** file. For details, see the Understanding and configuring Samba ID mapping

7. Edit the **/etc/krb5.conf** file and add the following section:

```
[plugins]
  localauth = {
    module = winbind:/usr/lib64/samba/krb5/winbind_krb5_localauth.so
    enable_only = winbind
  }
```

8. Verify that the **winbind** service is running:

```
# systemctl status winbind
...
  Active: active (running) since Tue 2018-11-06 19:10:40 CET; 15s ago
```



IMPORTANT

To enable Samba to query domain user and group information, the **winbind** service must be running before you start **smb**.

9. If you installed the **samba** package to share directories and printers, enable and start the **smb** service:

```
# systemctl enable --now smb
```

Verification steps

1. Display an AD user's details, such as the AD administrator account in the AD domain:

```
# getent passwd "AD\administrator"
AD\administrator:*:10000:10000::/home/administrator@AD:/bin/bash
```

2. Query the members of the domain users group in the AD domain:

```
# getent group "AD\Domain Users"
  AD\domain users:x:10000:user1,user2
```

3. Optionally, verify that you can use domain users and groups when you set permissions on files and directories. For example, to set the owner of the **/srv/samba/example.txt** file to **AD\administrator** and the group to **AD\Domain Users**:

```
# chown "AD\administrator":"AD\Domain Users" /srv/samba/example.txt
```

4. Verify that Kerberos authentication works as expected:

   a. On the AD domain member, obtain a ticket for the **administrator@AD.EXAMPLE.COM** principal:

   ```
   # kinit administrator@AD.EXAMPLE.COM
   ```

   b. Display the cached Kerberos ticket:

   ```
   # klist
   Ticket cache: KCM:0
   Default principal: administrator@AD.EXAMPLE.COM

   Valid starting       Expires              Service principal
   01.11.2018 10:00:00  01.11.2018 20:00:00
   krbtgt/AD.EXAMPLE.COM@AD.EXAMPLE.COM
           renew until 08.11.2018 05:00:00
   ```

5. Display the available domains:

```
# wbinfo --all-domains
BUILTIN
SAMBA-SERVER
AD
```

Additional resources

- If you do not want to use the deprecated RC4 ciphers, you can enable the AES encryption type in AD. See
- Enabling the AES encryption type in Active Directory using a GPO
- **realm(8)** man page

2.4. REALM COMMANDS

The **realmd** system has two major task areas:

- Managing system enrollment in a domain.
- Controlling which domain users are allowed to access local system resources.

In **realmd** use the command line tool **realm** to run commands. Most **realm** commands require the user to specify the action that the utility should perform, and the entity, such as a domain or user account, for which to perform the action.

Table 2.1. realmd Commands

Command	Description
Realm Commands	

Command	Description
discover	Run a discovery scan for domains on the network.
join	Add the system to the specified domain.
leave	Remove the system from the specified domain.
list	List all configured domains for the system or all discovered and configured domains.
Login Commands	
permit	Enable access for specific users or for all users within a configured domain to access the local system.
deny	Restrict access for specific users or for all users within a configured domain to access the local system.

Additional resources

- The **realm(8)** man page.

CHAPTER 3. INTEGRATING RHEL SYSTEMS INTO AD DIRECTLY WITH ANSIBLE BY USING RHEL SYSTEM ROLES

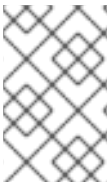
With the **ad_integration** system role, you can automate a direct integration of a RHEL system with Active Directory (AD) by using Red Hat Ansible Automation Platform.

3.1. THE AD_INTEGRATION SYSTEM ROLE

Using the **ad_integration** system role, you can directly connect a RHEL system to Active Directory (AD).

The role uses the following components:

- SSSD to interact with the central identity and authentication source
- **realmd** to detect available AD domains and configure the underlying RHEL system services, in this case SSSD, to connect to the selected AD domain



NOTE

The **ad_integration** role is for deployments using direct AD integration without an Identity Management (IdM) environment. For IdM environments, use the **ansible-freeipa** roles.

Additional resources

- **/usr/share/ansible/roles/rhel-system-roles.ad_integration/README.md** file
- **/usr/share/doc/rhel-system-roles/ad_integration/** directory
- Connecting RHEL systems directly to AD using SSSD

3.2. VARIABLES OF THE AD_INTEGRATION RHEL SYSTEM ROLE

The **ad_integration** RHEL system role uses the following parameters:

Role Variable	Description
ad_integration_realm	Active Directory realm, or domain name to join.
ad_integration_password	The password of the user used to authenticate with when joining the machine to the realm. Do not use plain text. Instead, use Ansible Vault to encrypt the value.
ad_integration_manage_crypto_policies	If **true**, the **ad_integration** role will use **fedora.linux_system_roles.crypto_policies** as needed. Default: **false**

Role Variable	Description
ad_integration_allow_rc4_crypto	If **true**, the **ad_integration** role will set the crypto policy to allow RC4 encryption. Providing this variable automatically sets **ad_integration_manage_crypto_policies** to **true**. Default: **false**
ad_integration_timesync_source	Hostname or IP address of time source to synchronize the system clock with. Providing this variable automatically sets **ad_integration_manage_timesync** to **true**.

Additional resources

- **/usr/share/ansible/roles/rhel-system-roles.ad_integration/README.md** file
- **/usr/share/doc/rhel-system-roles/ad_integration/** directory

3.3. CONNECTING A RHEL SYSTEM DIRECTLY TO AD BY USING THE AD_INTEGRATION SYSTEM ROLE

You can use the **ad_integration** system role to configure a direct integration between a RHEL system and an AD domain by running an Ansible playbook.

NOTE

Starting with RHEL8, RHEL no longer supports RC4 encryption by default. If it is not possible to enable AES in the AD domain, you must enable the **AD-SUPPORT** crypto policy and allow RC4 encryption in the playbook.

IMPORTANT

Time between the RHEL server and AD must be synchronized. You can ensure this by using the **timesync** system role in the playbook.

In this example, the RHEL system joins the **domain.example.com** AD domain, by using the AD **Administrator** user and the password for this user stored in the Ansible vault. The playbook also sets the **AD-SUPPORT** crypto policy and allows RC4 encryption. To ensure time synchronization between the RHEL system and AD, the playbook sets the **adserver.domain.example.com** server as the **timesync** source.

Prerequisites

- You have prepared the control node and the managed nodes
- You are logged in to the control node as a user who can run playbooks on the managed nodes.

- The account you use to connect to the managed nodes has **sudo** permissions on them.
- The following ports on the AD domain controllers are open and accessible from the RHEL server:

Table 3.1. Ports Required for Direct Integration of Linux Systems into AD Using the ad_integration system role

Source Port	Destination Port	Protocol	Service
1024:65535	53	UDP and TCP	DNS
1024:65535	389	UDP and TCP	LDAP
1024:65535	636	TCP	LDAPS
1024:65535	88	UDP and TCP	Kerberos
1024:65535	464	UDP and TCP	Kerberos change/set password (**kadmin**)
1024:65535	3268	TCP	LDAP Global Catalog
1024:65535	3269	TCP	LDAP Global Catalog SSL/TLS
1024:65535	123	UDP	NTP/Chrony (Optional)
1024:65535	323	UDP	NTP/Chrony (Optional)

Procedure

1. Create a playbook file, for example ~/**playbook.yml**, with the following content:

```
---
- name: Configure a direct integration between a RHEL system and an AD domain
  hosts: managed-node-01.example.com
  roles:
    - rhel-system-roles.ad_integration
  vars:
    ad_integration_realm: "domain.example.com"
    ad_integration_password: !vault | vault encrypted password
    ad_integration_manage_crypto_policies: true
    ad_integration_allow_rc4_crypto: true
    ad_integration_timesync_source: "adserver.domain.example.com"
```

2. Validate the playbook syntax:

```
$ ansible-playbook --syntax-check ~/playbook.yml
```

Note that this command only validates the syntax and does not protect against a wrong but valid configuration.

3. Run the playbook:

```
$ ansible-playbook ~/playbook.yml
```

Verification

- Display an AD user details, such as the **administrator** user:

```
$ getent passwd administrator@ad.example.com
administrator@ad.example.com:*:1450400500:1450400513:Administrator:/home/administrator
@ad.example.com:/bin/bash
```

Additional resources

- **/usr/share/ansible/roles/rhel-system-roles.ad_integration/README.md** file
- **/usr/share/doc/rhel-system-roles/ad_integration/** directory

CHAPTER 4. MANAGING DIRECT CONNECTIONS TO AD

You can use the System Security Services Daemon (SSSD) or Samba Winbind to connect your Red Hat Enterprise Linux (RHEL) system to Active Directory (AD). This section describes how to modify and manage your connection to AD when your RHEL system is already configured as an AD client.

Prerequisites

- You have connected your RHEL system to the Active Directory domain, either with SSSD or Samba Winbind.

4.1. MODIFYING THE DEFAULT KERBEROS HOST KEYTAB RENEWAL INTERVAL

SSSD automatically renews the Kerberos host keytab file in an AD environment if the **adcli** package is installed. The daemon checks daily if the machine account password is older than the configured value and renews it if necessary.

The default renewal interval is 30 days. To change the default, follow the steps in this procedure.

Procedure

1. Add the following parameter to the AD provider in your **/etc/sssd/sssd.conf** file:

```
ad_maximum_machine_account_password_age = value_in_days
```

2. Restart SSSD:

```
# systemctl restart sssd
```

3. To disable the automatic Kerberos host keytab renewal, set **ad_maximum_machine_account_password_age = 0**.

Additional resources

- **adcli(8)**
- **sssd.conf(5)**
- SSSD service is failing with an error 'Failed to initialize credentials using keytab [MEMORY:/etc/krb5.keytab]: Preauthentication failed.'

4.2. REMOVING A RHEL SYSTEM FROM AN AD DOMAIN

Follow this procedure to remove a Red Hat Enterprise Linux (RHEL) system that is integrated into Active Directory (AD) directly from the AD domain.

Prerequisites

- You have used the System Security Services Daemon (SSSD) or Samba Winbind to connect your RHEL system to AD.

Procedure

1. Remove a system from an identity domain using the **realm leave** command. The command removes the domain configuration from SSSD and the local system.

```
# realm leave ad.example.com
```

NOTE

When a client leaves a domain, the account is not deleted from AD; the local client configuration is only removed. If you want to delete the AD account, run the command with the **--remove** option. You are prompted for your user password and you must have the rights to remove an account from Active Directory.

2. Use the **-U** option with the **realm leave** command to specify a different user to remove a system from an identity domain.
By default, the **realm leave** command is executed as the default administrator. For AD, the administrator account is called **Administrator**. If a different user was used to join to the domain, it might be required to perform the removal as that user.

```
# realm leave [ad.example.com] -U [AD.EXAMPLE.COM\user]'
```

The command first attempts to connect without credentials, but it prompts for a password if required.

Verification steps

- Verify the domain is no longer configured:

```
# realm discover [ad.example.com]
ad.example.com
   type: kerberos
   realm-name: EXAMPLE.COM
   domain-name: example.com
   configured: no
   server-software: active-directory
   client-software: sssd
   required-package: oddjob
   required-package: oddjob-mkhomedir
   required-package: sssd
   required-package: adcli
   required-package: samba-common-tools
```

Additional resources

- See the **realm(8)** man page.

4.3. SETTING THE DOMAIN RESOLUTION ORDER IN SSSD TO RESOLVE SHORT AD USER NAMES

By default, you must specify fully qualified usernames, like **ad_username@ad.example.com** and **group@ad.example.com**, to resolve Active Directory (AD) users and groups on a RHEL host connected to AD with the SSSD service.

This procedure sets the domain resolution order in the SSSD configuration so you can resolve AD users and groups using short names, like **ad_username**. This example configuration searches for users and groups in the following order:

1. Active Directory (AD) child domain **subdomain2.ad.example.com**
2. AD child domain **subdomain1.ad.example.com**
3. AD root domain **ad.example.com**

Prerequisites

- You have used the SSSD service to connect the RHEL host directly to AD.

Procedure

1. Open the **/etc/sssd/sssd.conf** file in a text editor.
2. Set the **domain_resolution_order** option in the **[sssd]** section of the file.

```
domain_resolution_order = subdomain2.ad.example.com, subdomain1.ad.example.com,
ad.example.com
```

3. Save and close the file.
4. Restart the SSSD service to load the new configuration settings.

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

Verification Steps

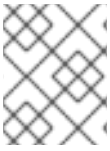
- Verify you can retrieve user information for a user from the first domain using only a short name.

```
[root@ad-client ~]# id <user_from_subdomain2>
uid=1916901142(user_from_subdomain2) gid=1916900513(domain users)
groups=1916900513(domain users)
```

4.4. MANAGING LOGIN PERMISSIONS FOR DOMAIN USERS

By default, domain-side access control is applied, which means that login policies for Active Directory (AD) users are defined in the AD domain itself. This default behavior can be overridden so that client-side access control is used. With client-side access control, login permission is defined by local policies only.

If a domain applies client-side access control, you can use the **realmd** to configure basic allow or deny access rules for users from that domain.



NOTE

Access rules either allow or deny access to all services on the system. More specific access rules must be set on a specific system resource or in the domain.

4.4.1. Enabling access to users within a domain

By default, login policies for Active Directory (AD) users are defined in the AD domain itself. Follow this procedure to override this default behavior and configure a RHEL host to enable access for users within an AD domain.

IMPORTANT

It is not recommended to allow access to all by default while only denying it to specific users with realm permit **-x**. Instead, Red Hat recommends maintaining a default no access policy for all users and only grant access to selected users using realm permit.

Prerequisites

- Your RHEL system is a member of the Active Directory domain.

Procedure

1. Grant access to all users:

```
# realm permit --all
```

2. Grant access to specific users:

```
$ realm permit aduser01@example.com
$ realm permit 'AD.EXAMPLE.COM\aduser01'
```

Currently, you can only allow access to users in primary domains and not to users in trusted domains. This is due to the fact that user login must contain the domain name and SSSD cannot currently provide **realmd** with information about available child domains.

Verification steps

1. Use SSH to log in to the server as the **aduser01@example.com** user:

```
$ ssh aduser01@example.com@server_name
[aduser01@example.com@server_name ~]$
```

2. Use the ssh command a second time to access the same server, this time as the **aduser02@example.com** user:

```
$ ssh aduser02@example.com@server_name
Authentication failed.
```

Notice how the **aduser02@example.com** user is denied access to the system. You have granted the permission to log in to the system to the **aduser01@example.com** user only. All other users from that Active Directory domain are rejected because of the specified login policy.

NOTE

If you set **use_fully_qualified_names** to true in the **sssd.conf** file, all requests must use the fully qualified domain name. However, if you set **use_fully_qualified_names** to false, it is possible to use the fully-qualified name in the requests, but only the simplified version is displayed in the output.

Additional resources

- See the **realm(8)** man page.

4.4.2. Denying access to users within a domain

By default, login policies for Active Directory (AD) users are defined in the AD domain itself. Follow this procedure to override this default behavior and configure a RHEL host to deny access to users within an AD domain.

IMPORTANT

It is safer to only allow access to specific users or groups than to deny access to some, while enabling it to everyone else. Therefore, it is not recommended to allow access to all by default while only denying it to specific users with realm permit **-x**. Instead, Red Hat recommends maintaining a default no access policy for all users and only grant access to selected users using realm permit.

Prerequisites

- Your RHEL system is a member of the Active Directory domain.

Procedure

1. Deny access to all users within the domain:

```
# realm deny --all
```

   This command prevents **realm** accounts from logging into the local machine. Use **realm permit** to restrict login to specific accounts.

2. Verify that the domain user's **login-policy** is set to **deny-any-login**:

```
[root@replica1 ~]# realm list
example.net
  type: kerberos
  realm-name: EXAMPLE.NET
  domain-name: example.net
  configured: kerberos-member
  server-software: active-directory
  client-software: sssd
  required-package: oddjob
  required-package: oddjob-mkhomedir
  required-package: sssd
  required-package: adcli
  required-package: samba-common-tools
  login-formats: %U@example.net
  login-policy: deny-any-login
```

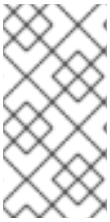
3. Deny access to specific users by using the **-x** option:

```
$ realm permit -x 'AD.EXAMPLE.COM\aduser02'
```

Verification steps

- Use SSH to log in to the server as the **aduser01@example.net** user.

```
$ ssh aduser01@example.net@server_name
Authentication failed.
```



NOTE

If you set **use_fully_qualified_names** to true in the **sssd.conf** file, all requests must use the fully qualified domain name. However, if you set **use_fully_qualified_names** to false, it is possible to use the fully-qualified name in the requests, but only the simplified version is displayed in the output.

Additional resources

- See the **realm(8)** man page.

4.5. APPLYING GROUP POLICY OBJECT ACCESS CONTROL IN RHEL

A *Group Policy Object* (GPO) is a collection of access control settings stored in Microsoft Active Directory (AD) that can apply to computers and users in an AD environment. By specifying GPOs in AD, administrators can define login policies honored by both Windows clients and Red Hat Enterprise Linux (RHEL) hosts joined to AD.

The following sections describe how you can manage GPOs in your environment:

- How SSSD interprets GPO access control rules
- List of GPO settings that SSSD supports
- List of SSSD options to control GPO enforcement
- Changing the GPO access control mode
- Creating and configuring a GPO for a RHEL host

4.5.1. How SSSD interprets GPO access control rules

By default, SSSD retrieves Group Policy Objects (GPOs) from Active Directory (AD) domain controllers and evaluates them to determine if a user is allowed to log in to a particular RHEL host joined to AD.

SSSD maps AD *Windows Logon Rights* to Pluggable Authentication Module (PAM) service names to enforce those permissions in a GNU/Linux environment.

As an AD Administrator, you can limit the scope of GPO rules to specific users, groups, or hosts by listing them in a *security filter*.

Limitations on filtering by hosts

Older versions of SSSD do not evaluate hosts in AD GPO security filters.

- **RHEL 8.3.0 and newer:** SSSD supports users, groups, and hosts in security filters.
- **RHEL versions older than 8.3.0:** SSSD ignores host entries and only supports users and groups in security filters.
  To ensure that SSSD applies GPO-based access control to a specific host, create a new

Organizational Unit (OU) in the AD domain, move the system to the new OU, and then link the GPO to this OU.

Limitations on filtering by groups

SSSD currently does not support Active Directory's built-in groups, such as **Administrators** with Security Identifier (SID) **S-1-5-32-544**. Red Hat recommends against using AD built-in groups in AD GPOs targeting RHEL hosts.

Additional resources

- For a list of Windows GPO options and their corresponding SSSD options, see List of GPO settings that SSSD supports.

4.5.2. List of GPO settings that SSSD supports

The following table shows the SSSD options that correspond to Active Directory GPO options as specified in the *Group Policy Management Editor* on Windows.

Table 4.1. GPO access control options retrieved by SSSD

GPO option	Corresponding **sssd.conf** option
Allow log on locally Deny log on locally	**ad_gpo_map_interactive**
Allow log on through Remote Desktop Services Deny log on through Remote Desktop Services	**ad_gpo_map_remote_interactive**
Access this computer from the network Deny access to this computer from the network	**ad_gpo_map_network**
Allow log on as a batch job Deny log on as a batch job	**ad_gpo_map_batch**
Allow log on as a service Deny log on as a service	**ad_gpo_map_service**

Additional resources

- For more information about these **sssd.conf** settings, such as the Pluggable Authentication Module (PAM) services that map to GPO options, see the **sssd-ad(5)** Manual page entry.

4.5.3. List of SSSD options to control GPO enforcement

You can set the following SSSD options to limit the scope of GPO rules.

The ad_gpo_access_control option

You can set the **ad_gpo_access_control** option in the **/etc/sssd/sssd.conf** file to choose between three different modes in which GPO-based access control operates.

Table 4.2. Table of ad_gpo_access_control values

Value of ad_gpo_access_control	Behavior
enforcing	GPO-based access control rules are evaluated and enforced. **This is the default setting in RHEL 8.**
permissive	GPO-based access control rules are evaluated but **not** enforced; a **syslog** message is recorded every time access would be denied. This is the default setting in RHEL 7. This mode is ideal for testing policy adjustments while allowing users to continue logging in.
disabled	GPO-based access control rules are neither evaluated nor enforced.

The ad_gpo_implicit_deny option

The **ad_gpo_implicit_deny** option is set to **False** by default. In this default state, users are allowed access if applicable GPOs are not found. If you set this option to **True**, you must explicitly allow users access with a GPO rule.

You can use this feature to harden security, but be careful not to deny access unintentionally. Red Hat recommends testing this feature while **ad_gpo_access_control** is set to **permissive**.

The following two tables illustrate when a user is allowed or rejected access based on the allow and deny login rights defined on the AD server-side and the value of **ad_gpo_implicit_deny**.

Table 4.3. Login behavior with ad_gpo_implicit_deny set to False (default)

allow-rules	deny-rules	result
missing	missing	all users are allowed
missing	present	only users not in deny-rules are allowed
present	missing	only users in allow-rules are allowed
present	present	only users in allow-rules and not in deny-rules are allowed

Table 4.4. Login behavior with ad_gpo_implicit_deny set to True

allow-rules	deny-rules	result
missing	missing	no users are allowed
missing	present	no users are allowed
present	missing	only users in allow-rules are allowed

allow-rules	deny-rules	result
present	present	only users in allow-rules and not in deny-rules are allowed

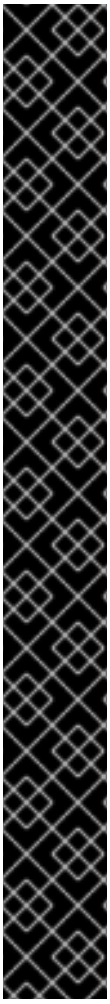
Additional resources

- For the procedure to change the GPO enforcement mode in SSSD, see Changing the GPO access control mode.
- For more details on each of the different GPO modes of operation, see the **ad_gpo_access_control** entry in the **sssd-ad(5)** Manual page.

4.5.4. Changing the GPO access control mode

This procedure changes how GPO-based access control rules are evaluated and enforced on a RHEL host joined to an Active Directory (AD) environment.

In this example, you will change the GPO operation mode from **enforcing** (the default) to **permissive** for testing purposes.



IMPORTANT

If you see the following errors, Active Directory users are unable to log in due to GPO-based access controls:

- In **/var/log/secure**:

> Oct 31 03:00:13 client1 sshd[124914]: pam_sss(sshd:account): **Access denied for user aduser1: 6 (Permission denied)**
> Oct 31 03:00:13 client1 sshd[124914]: Failed password for aduser1 from 127.0.0.1 port 60509 ssh2
> Oct 31 03:00:13 client1 sshd[124914]: fatal: **Access denied for user aduser1 by PAM account configuration [preauth]**

- In **/var/log/sssd/sssd__example.com_.log**:

> (Sat Oct 31 03:00:13 2020) [sssd[be[example.com]]] [ad_gpo_perform_hbac_processing] (0x0040): **GPO access check failed: [1432158236](Host Access Denied)**
> (Sat Oct 31 03:00:13 2020) [sssd[be[example.com]]] [ad_gpo_cse_done] (0x0040): HBAC processing failed: [1432158236](Host Access Denied}
> (Sat Oct 31 03:00:13 2020) [sssd[be[example.com]]] [ad_gpo_access_done] (0x0040): **GPO-based access control failed.**

If this is undesired behavior, you can temporarily set **ad_gpo_access_control** to **permissive** as described in this procedure while you troubleshoot proper GPO settings in AD.

Prerequisites

- You have joined a RHEL host to an AD environment using SSSD.

- Editing the **/etc/sssd/sssd.conf** configuration file requires **root** permissions.

Procedure

1. Stop the SSSD service.

```
[root@server ~]# systemctl stop sssd
```

2. Open the **/etc/sssd/sssd.conf** file in a text editor.

3. Set **ad_gpo_access_control** to **permissive** in the **domain** section for the AD domain.

```
[domain/example.com]
ad_gpo_access_control=permissive
...
```

4. Save the **/etc/sssd/sssd.conf** file.

5. Restart the SSSD service to load configuration changes.

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

Additional resources

- For the list of different GPO access control modes, see List of SSSD options to control GPO enforcement.

4.5.5. Creating and configuring a GPO for a RHEL host in the AD GUI

A Group Policy Object (GPO) is a collection of access control settings stored in Microsoft Active Directory (AD) that can apply to computers and users in an AD environment. The following procedure creates a GPO in the AD graphical user interface (GUI) to control logon access to a RHEL host that is integrated directly to the AD domain.

Prerequisites

- You have joined a RHEL host to an AD environment using SSSD.
- You have AD Administrator privileges to make changes in AD using the GUI.

Procedure

1. Within **Active Directory Users and Computers**, create an Organizational Unit (OU) to associate with the new GPO:
   a. Right-click on the domain.
   b. Choose **New**.
   c. Choose **Organizational Unit**.
2. Click on the name of the Computer Object that represents the RHEL host (created when it joined Active Directory) and drag it into the new OU. By having the RHEL host in its own OU, the GPO targets this host.

3. Within the **Group Policy Management Editor**, create a new GPO for the OU you created:
    a. Expand **Forest**.
    b. Expand **Domains**.
    c. Expand your domain.
    d. Right-click on the new OU.
    e. Choose **Create a GPO in this domain**.
4. Specify a name for the new GPO, such as **Allow SSH access** or **Allow Console/GUI access** and click **OK**.
5. Edit the new GPO:
    a. Select the OU within the **Group Policy Management** editor.
    b. Right-click and choose **Edit**.
    c. Select **User Rights Assignment**.
    d. Select **Computer Configuration**
    e. Select **Policies**.
    f. Select **Windows Settings**.
    g. Select **Security Settings**.
    h. Select **Local Policies**.
    i. Select **User Rights Assignment**.
6. Assign login permissions:
    a. Double-Click on **Allow log on locally** to grant local console/GUI access.
    b. Double-click on **Allow log on through Remote Desktop Services** to grant SSH access.
7. Add the user(s) you want to access either of these policies to the policies themselves:
    a. Click **Add User or Group**.
    b. Enter the username within the blank field.
    c. Click **OK**.

Additional resources

- For more details on Group Policy Objects, see Group Policy Objects in Microsoft documentation.

4.5.6. Additional resources

- For more information about joining a RHEL host to an Active Directory environment, see Connecting RHEL systems directly to AD using SSSD .

CHAPTER 5. ACCESSING AD WITH A MANAGED SERVICE ACCOUNT

Active Directory (AD) Managed Service Accounts (MSAs) allow you to create an account in AD that corresponds to a specific computer. You can use an MSA to connect to AD resources as a specific user principal, without joining the RHEL host to the AD domain.

This section discusses the following topics:

- The benefits of a Managed Service Account
- Configuring a Managed Service Account for a RHEL host
- Updating the password for a Managed Service Account
- Managed Service Account specifications
- Options for the adcli create-msa command

5.1. THE BENEFITS OF A MANAGED SERVICE ACCOUNT

If you want to allow a RHEL host to access an Active Directory (AD) domain without joining it, you can use a Managed Service Account (MSA) to access that domain. An MSA is an account in AD that corresponds to a specific computer, which you can use to connect to AD resources as a specific user principal.

For example, if the AD domain **production.example.com** has a one-way trust relationship with the **lab.example.com** AD domain, the following conditions apply:

- The **lab** domain trusts users and hosts from the **production** domain.
- The **production** domain does **not** trust users and hosts from the **lab** domain.

This means that a host joined to the **lab** domain, such as **client.lab.example.com**, cannot access resources from the **production** domain through the trust.

If you want to create an exception for the **client.lab.example.com** host, you can use the **adcli** utility to create a MSA for the **client** host in the **production.example.com** domain. By authenticating with the Kerberos principal of the MSA, you can perform secure LDAP searches in the **production** domain from the **client** host.

5.2. CONFIGURING A MANAGED SERVICE ACCOUNT FOR A RHEL HOST

This procedure creates a Managed Service Account (MSA) for a host from the **lab.example.com** Active Directory (AD) domain, and configures SSSD so you can access and authenticate to the **production.example.com** AD domain.

NOTE

If you need to access AD resources from a RHEL host, Red Hat recommends that you join the RHEL host to the AD domain with the **realm** command. See Connecting RHEL systems directly to AD using SSSD.

Only perform this procedure if one of the following conditions applies:

- You cannot join the RHEL host to the AD domain, and you want to create an account for that host in AD.
- You have joined the RHEL host to an AD domain, and you need to access another AD domain where the host credentials from the domain you have joined are not valid, such as with a one-way trust.

Prerequisites

- Ensure that the following ports on the RHEL host are open and accessible to the AD domain controllers.

Service	Port	Protocols
DNS	53	TCP, UDP
LDAP	389	TCP, UDP
LDAPS (optional)	636	TCP, UDP
Kerberos	88	TCP, UDP

- You have the password for an AD Administrator that has rights to create MSAs in the **production.example.com** domain.
- You have root permissions that are required to run the **adcli** command, and to modify the **/etc/sssd/sssd.conf** configuration file..
- *(Optional)* You have the **krb5-workstation** package installed, which includes the **klist** diagnostic utility.

Procedure

1. Create an MSA for the host in the **production.example.com** AD domain.

```
[root@client ~]# adcli create-msa --domain=production.example.com
```

2. Display information about the MSA from the Kerberos keytab that was created. Make note of the MSA name:

```
[root@client ~]# klist -k /etc/krb5.keytab.production.example.com
Keytab name: FILE:/etc/krb5.keytab.production.example.com
KVNO Principal
```

```
---- ------------------------------------------------------------
   2 CLIENT!S3A$@PRODUCTION.EXAMPLE.COM (aes256-cts-hmac-sha1-96)
   2 CLIENT!S3A$@PRODUCTION.EXAMPLE.COM (aes128-cts-hmac-sha1-96)
```

3. Open the **/etc/sssd/sssd.conf** file and choose the appropriate SSSD domain configuration to add:

   - If the MSA corresponds to an **AD domain from a different forest**, create a new domain section named **[domain/<name_of_domain>]**, and enter information about the MSA and the keytab. The most important options are **ldap_sasl_authid**, **ldap_krb5_keytab**, and **krb5_keytab**:

     ```
     [domain/production.example.com]
     ldap_sasl_authid = CLIENT!S3A$@PRODUCTION.EXAMPLE.COM
     ldap_krb5_keytab = /etc/krb5.keytab.production.example.com
     krb5_keytab = /etc/krb5.keytab.production.example.com
     ad_domain = production.example.com
     krb5_realm = PRODUCTION.EXAMPLE.COM
     access_provider = ad
     ...
     ```

   - If the MSA corresponds to an **AD domain from the local forest**, create a new sub-domain section in the format **[domain/root.example.com/sub-domain.example.com]**, and enter information about the MSA and the keytab. The most important options are **ldap_sasl_authid**, **ldap_krb5_keytab**, and **krb5_keytab**:

     ```
     [domain/ad.example.com/production.example.com]
     ldap_sasl_authid = CLIENT!S3A$@PRODUCTION.EXAMPLE.COM
     ldap_krb5_keytab = /etc/krb5.keytab.production.example.com
     krb5_keytab = /etc/krb5.keytab.production.example.com
     ad_domain = production.example.com
     krb5_realm = PRODUCTION.EXAMPLE.COM
     access_provider = ad
     ...
     ```

Verification steps

- Verify you can retrieve a Kerberos ticket-granting ticket (TGT) as the MSA:

  ```
  [root@client ~]# kinit -k -t /etc/krb5.keytab.production.example.com 'CLIENT!S3A$'
  [root@client ~]# klist
  Ticket cache: KCM:0:54655
  Default principal: CLIENT!S3A$@PRODUCTION.EXAMPLE.COM

  Valid starting       Expires              Service principal
  11/22/2021 15:48:03  11/23/2021 15:48:03
  krbtgt/PRODUCTION.EXAMPLE.COM@PRODUCTION.EXAMPLE.COM
  ```

- In AD, verify you have an MSA for the host in the Managed Service Accounts Organizational Unit (OU).

Additional resources

- Connecting RHEL systems directly to AD using SSSD

5.3. UPDATING THE PASSWORD FOR A MANAGED SERVICE ACCOUNT

Managed Service Accounts (MSAs) have a complex password that is maintained automatically by Active Directory (AD). By default, the System Services Security Daemon (SSSD) automatically updates the MSA password in the Kerberos keytab if it is older than 30 days, which keeps it up to date with the password in AD. This procedure explains how to manually update the password for your MSA.

Prerequisites

- You have previously created an MSA for a host in the production.example.com AD domain.
- *(Optional)* You have the **krb5-workstation** package installed, which includes the **klist** diagnostic utility.

Procedure

1. *(Optional)* Display the current Key Version Number (KVNO) for the MSA in the Kerberos keytab. The current KVNO is 2.

```
[root@client ~]# klist -k /etc/krb5.keytab.production.example.com
Keytab name: FILE:/etc/krb5.keytab.production.example.com
KVNO Principal
---- --------------------------------------------------------------
   2 CLIENT!S3A$@PRODUCTION.EXAMPLE.COM (aes256-cts-hmac-sha1-96)
   2 CLIENT!S3A$@PRODUCTION.EXAMPLE.COM (aes128-cts-hmac-sha1-96)
```

2. Update the password for the MSA in the **production.example.com** AD domain.

```
[root@client ~]# adcli update --domain=production.example.com --host-
keytab=/etc/krb5.keytab.production.example.com --computer-password-lifetime=0
```

Verification steps

- Verify that you have incremented the KVNO in the Kerberos keytab:

```
[root@client ~]# klist -k /etc/krb5.keytab.production.example.com
Keytab name: FILE:/etc/krb5.keytab.production.example.com
KVNO Principal
---- --------------------------------------------------------------
   3 CLIENT!S3A$@PRODUCTION.EXAMPLE.COM (aes256-cts-hmac-sha1-96)
   3 CLIENT!S3A$@PRODUCTION.EXAMPLE.COM (aes128-cts-hmac-sha1-96)
```

5.4. MANAGED SERVICE ACCOUNT SPECIFICATIONS

The Managed Service Accounts (MSAs) that the **adcli** utility creates have the following specifications:

- They cannot have additional service principal names (SPNs).
- By default, the Kerberos principal for the MSA is stored in a Kerberos keytab named ***<default_keytab_location>.<Active_Directory_domain>***, like ***/etc/krb5.keytab.production.example.com***.

- MSA names are limited to 20 characters or fewer. The last 4 characters are a suffix of 3 random characters from number and upper- and lowercase ASCII ranges appended to the short host name you provide, using a **!** character as a separator. For example, a host with the short name **myhost** receives an MSA with the following specifications:

Specification	Value
Common name (CN) attribute	**myhost!A2c**
NetBIOS name	**myhost!A2c$**
sAMAccountName	**myhost!A2c$**
Kerberos principal in the **production.example.com** AD domain	**myhost!A2c$@PRODUCTION.EXAMPLE.COM**

5.5. OPTIONS FOR THE ADCLI CREATE-MSA COMMAND

In addition to the global options you can pass to the **adcli** utility, you can specify the following options to specifically control how it handles Managed Service Accounts (MSAs).

-N, --computer-name

The short non-dotted name of the MSA that will be created in the Active Directory (AD) domain. If you do not specify a name, the first portion of the **--host-fqdn** or its default is used with a random suffix.

-O, --domain-ou=OU=*<path_to_OU>*

The full distinguished name of the Organizational Unit (OU) in which to create the MSA. If you do not specify this value, the MSA is created in the default location **OU=CN=Managed Service Accounts,DC=EXAMPLE,DC=COM**.

-H, --host-fqdn=host

Override the local machine's fully qualified DNS domain name. If you do not specify this option, the host name of the local machine is used.

-K, --host-keytab=*<path_to_keytab>*

The path to the host keytab to store MSA credentials. If you do not specify this value, the default location **/etc/krb5.keytab** is used with the lower-cased Active Directory domain name added as a suffix, such as **/etc/krb5.keytab.domain.example.com**.

--use-ldaps

Create the MSA over a Secure LDAP (LDAPS) channel.

--verbose

Print out detailed information while creating the MSA.

--show-details

Print out information about the MSA after creating it.

--show-password

Print out the MSA password after creating the MSA.