



Red Hat Enterprise Linux 8

Integrating RHEL systems directly with Windows Active Directory

Understanding and configuring RHEL systems to connect directly with Active Directory

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

Understanding and configuring RHEL systems to connect directly with Active Directory

Legal Notice

Copyright © 2020 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This documentation collection provides instructions on how to integrate RHEL systems directly with Windows Active Directory using SSSD.

Table of Contents

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION	3
CHAPTER 1. CONNECTING RHEL SYSTEMS DIRECTLY TO AD USING SSSD	4
1.1. OVERVIEW OF DIRECT INTEGRATION USING SSSD	4
1.2. SUPPORTED WINDOWS PLATFORMS FOR DIRECT INTEGRATION	5
1.3. ENSURING SUPPORT FOR COMMON ENCRYPTION TYPES IN AD AND RHEL	5
1.4. CONNECTING DIRECTLY TO AD	6
1.4.1. Connecting a RHEL system to an AD Domain	6
1.4.2. Options for integrating with AD: using ID mapping or POSIX attributes	7
1.4.2.1. Automatically generate new UIDs and GIDs for AD users	7
1.4.2.2. Use POSIX attributes defined in AD	8
1.4.3. Connecting to AD using POSIX attributes defined in Active Directory	8
1.4.4. Connecting to multiple domains in different AD forests with SSSD	10
1.5. HOW THE AD PROVIDER HANDLES TRUSTED DOMAINS	13
1.6. REALM COMMANDS	13
CHAPTER 2. CONNECTING RHEL SYSTEMS DIRECTLY TO AD USING SAMBA WINBIND	15
2.1. OVERVIEW OF DIRECT INTEGRATION USING SAMBA WINBIND	15
2.2. SUPPORTED WINDOWS PLATFORMS FOR DIRECT INTEGRATION	15
2.3. ENSURING SUPPORT FOR COMMON ENCRYPTION TYPES IN AD AND RHEL	16
2.4. JOINING A RHEL SYSTEM TO AN AD DOMAIN	16
2.5. REALM COMMANDS	18
CHAPTER 3. MANAGING DIRECT CONNECTIONS TO AD	20
3.1. MODIFYING THE DEFAULT KERBEROS HOST KEYTAB RENEWAL INTERVAL	20
3.2. REMOVING A RHEL SYSTEM FROM AN AD DOMAIN	20
3.3. MANAGING LOGIN PERMISSIONS FOR DOMAIN USERS	21
3.3.1. Enabling access to users within a domain	22
3.3.2. Denying access to users within a domain	23

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

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

- For simple comments on specific passages:
 1. Make sure you are viewing the documentation in the *Multi-page HTML* format. In addition, ensure you see the **Feedback** button in the upper right corner of the document.
 2. Use your mouse cursor to highlight the part of text that you want to comment on.
 3. Click the **Add Feedback** pop-up that appears below the highlighted text.
 4. Follow the displayed instructions.
- For submitting more complex feedback, create a Bugzilla ticket:
 1. Go to the [Bugzilla](#) website.
 2. As the Component, use **Documentation**.
 3. Fill in the **Description** field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.
 4. Click **Submit Bug**.

CHAPTER 1. CONNECTING RHEL SYSTEMS DIRECTLY TO AD USING SSSD

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

- [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 trusted domains](#)
- [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 on 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 on 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 2019
- Windows Server 2016
- Windows Server 2012 R2



NOTE

Windows Server 2019 does not introduce a new functional level. The highest functional level Windows Server 2019 uses is Windows Server 2016.

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

By default, SSSD supports RC4, AES-128, and AES-256 Kerberos encryption types.

RC4 encryption has been deprecated and disabled by default in RHEL 8, 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 remedy this situation, modify one of the following configurations:

- **Enable AES encryption support in Active Directory (recommended option)** To ensure trusts between 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](#)
- **Enable RC4 support in RHEL:** To enable support for the deprecated RC4 encryption type, use the **update-crypto-policies** command to enable the **AD-SUPPORT** cryptographic subpolicy in addition to the **DEFAULT** cryptographic policy in RHEL. Enter the following command on every

RHEL host where authentication against AD Domain Controllers takes place:

```
[root@host ~]# update-crypto-policies --set DEFAULT:AD-SUPPORT
```

Additional resources

- For more information on working with RHEL cryptographic policies, see [Using system-wide cryptographic policies](#) in the Security Hardening guide.

1.4. CONNECTING DIRECTLY TO AD

This section describes how to integrate directly with AD using either ID mapping or POSIX attributes.

- [Connecting a RHEL system to an AD domain](#)
- [Options for integrating with AD: using ID mapping or POSIX attributes](#)
- [Connecting to AD using POSIX attributes defined in Active Directory](#)
- [Connecting to multiple domains in different AD forests with SSSD](#)

1.4.1. Connecting a RHEL system to an AD Domain

This procedure describes how to connect a RHEL system to an AD domain using SSSD.

Prerequisites

- Ensure that none of the following ports are blocked to the AD domain controllers.

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	
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 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
```

If you do not want to use **realmd**, you can configure the system manually. See [Manually Connecting an SSSD Client to an Active Directory Domain](#) in the Red Hat Knowledgebase.

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
```

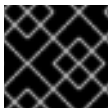
Additional resources

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

1.4.2. 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 [Managing Users and Groups](#) in *Configuring Basic System Settings*. Linux UIDs and GIDs are compliant with the POSIX standard.
- Windows use *security IDs* (SID).



IMPORTANT

Do not use the same user name in Windows and Linux.

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.

1.4.2.1. 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 [Connecting a RHEL system to 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.

1.4.2.2. 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](#) .

1.4.3. 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 none of the following ports are blocked 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

Service	Port	Protocol	Notes
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/sss/sss.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

This procedure describes joining and authenticating to multiple Active Directory (AD) domains in different forests where there is no trust between them.

This example describes joining two domains, **addomain1.com** and **addomain2.com**. Use **realmd** to join the first domain and automatically configure SSSD, Kerberos, and other utilities for that domain. Use **adcli** to join additional domains, and manually edit configuration files to include those domains.

Prerequisites

- Ensure that none of the following ports are blocked to the AD domain controllers.

Table 1.3. 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.
- Ensure you have credentials for an AD administrator account in each AD domain which has rights to join machines to that domain

Procedure

1. Install required packages.

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

2. Use **realmd** to join the first AD domain, **addomain1.com**.

```
# realm join ADDOMAIN1.COM
```

3. Rename the system keytab to a unique name.

```
# mv /etc/krb5.keytab /etc/addomain1.com.krb5.keytab
```

4. Use **adcli** to join the second AD domain, and any additional domains. Use the **-K** option to specify a unique path for the Kerberos keytab where host credentials will be written.

```
# adcli join -D dc2.addomain2.com -K /etc/addomain2.com.krb5.keytab
```

5. Modify **/etc/krb5.conf**.

- Add the **includedir** option to include SSSD configuration files.
- Enable DNS lookups for AD Domain Controllers with the **dns_lookup_kdc** option.

```
includedir /var/lib/sss/pubconf/krb5.include.d/

[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log

[libdefaults]
default_realm = ADDOMAIN1.COM
dns_lookup_realm = false
dns_lookup_kdc = true
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true

...
```

6. Modify **/etc/sss/sss.conf** to include information about all AD domains in use.

```
[sss]
services = nss, pam
config_file_version = 2
domains = addomain1.com, addomain2.com

[domain/addomain1.com]
id_provider = ad
access_provider = ad
krb5_keytab = /etc/addomain1.com.krb5.keytab
ldap_krb5_keytab = /etc/addomain1.com.krb5.keytab
ad_server = dc1.addomain1.com
ad_maximum_machine_account_password_age = 0
use_fully_qualified_names = true
default_shell=/bin/bash
```

```
override_homedir=/home/%d/%u
```

```
[domain/addomain2.com]
```

```
id_provider = ad
```

```
access_provider = ad
```

```
krb5_keytab = /etc/addomain2.com.krb5.keytab
```

```
ldap_krb5_keytab = /etc/addomain2.com.krb5.keytab
```

```
ad_server = dc2.addomain2.com
```

```
ad_maximum_machine_account_password_age = 0
```

```
use_fully_qualified_names = true
```

```
default_shell=/bin/bash
```

```
override_homedir=/home/%d/%u
```

```
[nss]
```

```
[pam]
```

- For each domain section, specify the path to the Kerberos keytab that corresponds to each domain with the **krb5_keytab** and **ldap_krb5_keytab** options.
 - Set **ad_maximum_machine_account_password_age = 0** to disable renewing host Kerberos keys.
 - Set **use_fully_qualified_names = true** to differentiate users from different domains.
 - Set **override_homedir = /home/%d/%u** so users (%u) from different domains (%d) each receive unique home directories. For example, the home directory for user **linuxuser@addomain1.com** is **/home/addomain1.com/linuxuser**.
7. SSH retrieves host keys from the system keytab and provides single sign-on functionality through GSSAPI/Kerberos. If you would like to use single sign-on, copy all current Kerberos host keys to the **/etc/krb5.keytab** system keytab.

```
# ktutil
```

```
ktutil: rkt /etc/addomain1.com.krb5.keytab
```

```
ktutil: rkt /etc/addomain2.com.krb5.keytab
```

```
ktutil: wkt /etc/krb5.keytab
```

8. Restart and enable the SSSD service.

```
# systemctl restart sssd
```

```
# systemctl enable sssd
```

Verification steps

1. Display user details for users from each AD domain:

```
# id administrator@addomain1.com
```

```
uid=1240800500(administrator@addomain1.com) gid=1240800513(domain
```

```
users@addomain1.com) groups=1240800513(domain
```

```
users@addomain1.com),1240800512(domain
```

```
admins@addomain1.com),1240800518(schema
```

```
admins@addomain1.com),1240800520(group policy creator
```

```
owners@addomain1.com),1240800572(denied rodc password replication
```

```
group@addomain1.com),1240800519(enterprise admins@addomain1.com)
```



```
# id administrator@addomain2.com
uid=1013800500(administrator@addomain2.com)
gid=1013800500(administrator@addomain2.com)
groups=1013800500(administrator@addomain2.com),1013800513(domain
users@addomain2.com)
```

2. Log in as a user from each domain and verify the correct home directory is created for the user.

```
# ssh administrator@addomain1.com@localhost
administrator@addomain1.com@localhost's password:
Creating directory '/home/addomain1.com/administrator'.
```

```
$ pwd
/home/addomain1.com/administrator
```

```
# ssh administrator@addomain2.com@localhost
administrator@addomain2.com@localhost's password:
Creating directory '/home/addomain2.com/administrator'.
```

```
$ pwd
/home/addomain2.com/administrator
```

1.5. HOW THE AD PROVIDER HANDLES TRUSTED DOMAINS

This section describes how SSSD handles trusted domains if you set **id_provider = ad** in the **/etc/sss/sss.conf** file.

- 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/sss/sss.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 **sss.conf(5)** man page.

1.6. REALM COMMANDS

The **realm** 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 1.4. realmd Commands

Command	Description
<i>Realm Commands</i>	
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.
<i>Login Commands</i>	
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.

For more information about the **realm** commands, see the **realm(8)** man page.

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

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

- [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 2019
- Windows Server 2016
- Windows Server 2012 R2



NOTE

Windows Server 2019 does not introduce a new functional level. The highest functional level Windows Server 2019 uses is Windows Server 2016.

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

By default, Samba Winbind supports RC4, AES-128, and AES-256 Kerberos encryption types.

RC4 encryption has been deprecated and disabled by default in RHEL 8, 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 remedy this situation, modify one of the following configurations:

- **Enable AES encryption support in Active Directory (recommended option)** To ensure trusts between 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](#)
- **Enable RC4 support in RHEL:** To enable support for the deprecated RC4 encryption type, use the **update-crypto-policies** command to enable the **AD-SUPPORT** cryptographic subpolicy in addition to the **DEFAULT** cryptographic policy in RHEL. Enter the following command on every RHEL host where authentication against AD Domain Controllers takes place:

```
[root@host ~]# update-crypto-policies --set DEFAULT:AD-SUPPORT
```

Additional resources

- For more information on working with RHEL cryptographic policies, see [Using system-wide cryptographic policies](#) in the Security Hardening guide.

2.4. JOINING A RHEL SYSTEM TO AN AD DOMAIN

This section describes how to join a Red Hat Enterprise Linux system to an AD domain by using **realmd** to configure Samba Winbind.

Procedure

1. Install the following packages:

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

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

```
# yum install samba
```

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

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

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

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

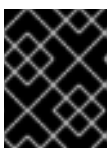
Using the previous command, the **realm** utility automatically:

- 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
5. Optionally, set an alternative ID mapping back end or customized ID mapping settings in the **/etc/samba/smb.conf** file. For details, see the [Samba ID mapping](#) section in the **Deploying different types of servers** documentation.
 6. 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
    }
```

7. 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**.

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

```
# systemctl start smb
```

Verification steps

- 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
```

- Query the members of the domain users group in the AD domain:

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

Additional resources

- For further details about the **realm** utility, see the **realm(8)** man page.

2.5. 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
<i>Realm Commands</i>	
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.
<i>Login Commands</i>	
permit	Enable access for specific users or for all users within a configured domain to access the local system.

Command	Description
deny	Restrict access for specific users or for all users within a configured domain to access the local system.

For more information about the **realm** commands, see the **realm(8)** man page.

CHAPTER 3. MANAGING DIRECT CONNECTIONS TO AD

This section describes how to modify and manage your connection to Active Directory.

Prerequisites

- You have connected your RHEL system to the Active Directory domain.

3.1. 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/sss/sss.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

- The **adcli(8)** man page.
- The **sss.conf(5)** man page.

3.2. REMOVING A RHEL SYSTEM FROM AN AD DOMAIN

This procedure describes how to remove a RHEL system from an Active Directory (AD) domain.

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.

3.3. 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.

3.3.1. Enabling access to users within a domain

This section describes how to enable access to users within a 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. 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** 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.

3.3.2. Denying access to users within a domain

This section describes how to deny access to all users within a 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.