



# How to join a Linux system to an Active Directory domain

Do you need to centrally manage Linux systems and user accounts under an Active Directory domain? Here's how to do it.

Posted: October 13, 2020 || [Edem Afenyo](#)



Image by [Kim Newberg](#) from [Pixabay](#)

Microsoft's Active Directory (AD) is the go-to directory service for many organizations. If you and your team are responsible for a mixed Windows and Linux environment, then you probably would like to centralize authentication for both platforms. I'll cover how to add Linux computers to an Active Directory domain.

## Active Directory and the need for centralized access management

---

### Great Linux resources

- [Advanced Linux commands cheat sheet](#)
- [Download RHEL 9 at no charge through the Red Hat Developer program](#)
- [A guide to installing applications on Linux](#)
- [Linux system administration skills assessment](#)



- [How well do you know Linux? Take a quiz and get a badge](#)

Microsoft's Active Directory, more popularly known as AD, has held the lion's share of the market for enterprise access management for many years now. It is used by institutions and individuals the world over to centrally control access to resources belonging to the organization. It gives you the ability to manage users, passwords, resources such as computers, and dictate who has access to what. For some of you reading this write-up, especially those who work in large institutions, you have interacted with AD before. Usually, the interaction is using one set of login credentials to log in to any workstation in the organization. That is just the tip of a large iceberg.

Imagine a collection of 40 computer systems and 70 users in a firm. Some employees run shifts while others work regular hours. Some have access to printing; others don't. The traditional way of working is to create local user accounts on each computer a user needs to access. Imagine the workload on the end-user support team. When a user changes his password for any reason, that user has to change the password on all computers he previously had access to, to keep things in sync. In no time, there will be mayhem. Now, imagine two members of the staff resign. I do not need to tell you the monotonous work that has to be repeated any time there's a change to the staffing or any workstations. For IT teams, this is a nightmare. Time that could be used for innovative tasks is now spent reinventing the wheel. I have not even spoken about managing access to the printers.

This is where a directory service such as Active Directory thrives. It can literally be a lifesaver. With Active Directory, each user is uniquely created as an object in a central database, with a single set of credentials. Each computer system is also created as an object. Automatically, every user can access every workstation with that same set of credentials. Any account changes that need to be made are made once at the central database. Members of staff can access the printers using the same set of credentials. The printers' authentication mechanism can be coupled with AD to achieve that. Happy users, happy IT team.

**[ Related reading: [How to integrate Active Directory Federation Services \(ADFS\) authentication with Red Hat SSO using SAML.](#) ]**

Using groups and organizational units, access to various resources can be tailored and maintained. It gets even better. This directory can store staff phone numbers, email addresses, and can be extended to store other information. What if someone resigns? No problem. Just disable the user's account. That person's access to all resources is nullified on the spot. The bigger the organization, the greater the need for centralized management. It saves time; it saves emotions.

At its heart, a directory service is just an organized way of itemizing all the resources in an organization while facilitating easy access to those resources. Basically, AD is a kind of distributed database, which is accessed remotely via the Lightweight Directory Access Protocol (LDAP). LDAP is an open protocol for remotely accessing directory services over a connection-oriented medium such as TCP/IP.

AD is not the only directory service based on the x.500 standard, or that can be accessed using LDAP. Other directory services include OpenLDAP and FreeIPA. However, AD is a mature Windows-based service that comes incorporated with Windows Server systems. In other words, it's going to be the automatic winner when your organization has many Windows systems. This is one of the reasons for its ubiquity. Directory services such as FreeIPA are Linux-based and provide an excellent service for a Linux stable. When the rubber hits the road, the choice boils down to which of the two you can set up quickly, given your current environment and your team's skill set.

**[ Learn how to [manage your Linux environment for success](#) by downloading this free eBook. ]**

But what happens when you choose AD, and you have a few CentOS servers, and you do not want to maintain a separate set of credentials for your Linux users? That overhead is entirely avoidable. What you need to do is join the Linux servers to the AD domain, like you would a Windows server.

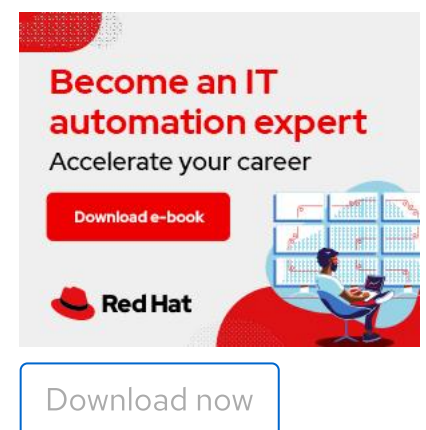
If that is what you need to do, then read on to find out just how to do it. It is possible to join a Windows system to a FreeIPA domain, but that is outside the scope of this article.

## Prerequisites

---

### Career advice

- [Take a sysadmin skills assessment](#)
- [Explore training and certification options](#)



- [Red Hat Certification remote exams FAQ](#)
- [10 resources to make you a better communicator](#)
- [How to explain modern software development in plain English](#)
- [Learning path: Getting started with Red Hat OpenShift Service on AWS \(ROSA\)](#)

This article presupposes that you have at least some introductory-level experience with Active Directory, especially around user and computer account management. Aside from that, the following obvious requirements need to be met:

- An account in AD that has the privileges necessary to join a system to the domain.
- A Linux server (a CentOS 7 server was used for this demonstration).
- A Domain Controller.
- Ensure your Linux server knows how to find the domain controller via DNS.

To make this article easier on everyone, here's a list of key details. This is how the lab I used for this write up is set up, so you should modify accordingly.

- AD Domain Name: Hope.net
- User account for joining the domain: fkorea (Fullname - Fiifi Korea)
- Linux server hostname: centy2

## Packages to install

For this configuration, the essential package to install is `realmd`. Aside from `realmd`, there are a host of packages that need to be installed to make this work.

```
# yum install sssd realmd oddjob oddjob-mkhomedir adcli samba-common samba-common-tools krb5-workstation openldap-clients polycoreutils-python
```

`Realmd` provides a simplified way to discover and interact with Active Directory domains. It employs `sssd` to do the actual lookups required for remote authentication and other heavy work of interacting with the domain. In the interest of brevity, I won't dwell on the other packages in the list.

However, for those interested in the details, a quick Google search should be of great help.

**[ Learn about setting up [multi-factor authentication on Linux systems](#). ]**

## Realmd (interacting with the domain)

Now that all packages have been installed, the first thing to do is to join the CentOS system to the Active Directory domain. We use the `realm` application for that. The `realm` client is installed at the same time as `realmd`. It is used to join, remove, control access, and accomplish many other tasks. Here is the expected syntax for a simple domain join:

```
realm join --user=[domain user account] [domain name]
```

The space between the user account and the domain account is not a typo. By inserting the corresponding details, we get the following command:

```
# realm join --user=fkorea hope.net
```

Supply the password when the prompt appears and wait for the process to end.

```
[root@centy2 ~]# realm join --user=fkorea hope.net
Password for fkorea:
[root@centy2 ~]#
```

Don't let the short absence of output deceive you. There are a number of operations that go on as part of the process. You can tack on the `-v` switch for more verbose output. However, the best way to check if the computer is now a member of the domain is by running the `realm list` command. The command attempts to display the current state of the server with regard to the domain. It is a quick and dirty way to know which groups or users can access the server.

Have a look at its output:

```
[root@centy2 ~]# realm list
Hope.net
  type: kerberos
  realm-name: HOPE.NET
  domain-name: hope.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@hope.net
  login-policy: allow-realm-logins
```

It is also quite trivial to place the newly-created AD computer object in a specific Organizational Unit (OU) from the onset. I'll leave that for further reading, but, as a tip, you can consult the man page. Using the `realm` client, you can grant or revoke access to domain users and groups. A deep dive on using `realmd` in a more fine-grained way is enough to make another article. However, I will not be out of order to pick out a few parameters for your attention, namely client-software and the server-software. By now, you should understand why we had to install so many packages.

To leave the domain altogether, you need two words: `realm leave`

## Further configuration

So now that the Linux server is part of the AD domain, domain users can access the server with their usual credentials. We are done, right? Wrong. "What's the problem?" I hear you say.

**[ Free cheat sheet: Get a list of [Linux utilities and commands for managing servers and networks](#). ]**

Well, for starters, this is the barebones configuration to get you up and running. But the experience is clunky, to say the least. We need to configure the service further to give it a true AD feel. It should be just like logging on to a domain-joined Windows 10 workstation.

Secondly, there is the big elephant in the room for sysadmins called Dynamic DNS Updates (DynDNS). If it is not set up correctly, we create extra overhead by having to maintain DNS records manually. For an environment that relies heavily on DNS, that could be a problem. For Windows systems, joining a system to the domain means two entries are automatically managed and maintained on the DNS server. When IP addresses change, the change is automatically reflected in DNS. This means you can change the IPs of systems without incurring the cost of manual maintenance. This will only make sense to people who already take advantage of DNS in their environments. Aside from the noticeable productivity gains of automation, it helps to have both Windows and Linux environments working the same way.



The third issue is DNS Scavenging. In an Active Directory domain, DNS is usually provided by the Domain Controllers. Every system joined to the domain has an automatic DNS entry with a corresponding IP address. This is super convenient. Automatically, at a specified interval, stale DNS records are deleted to prevent misdirected packets and also take care of deleted computer objects. This is known as *scavenging*, and it is not turned on by default in AD. However, if it is turned on, we need to configure it. Typically, the scavenging interval is seven days. If, after that period, there has been no update to the record, it is deleted, unless it is a static record. For Windows systems, the Dynamic Updates feature is automatically set up. However, with Linux servers, a few modifications need to be made. Without doing that, we will have services going down after a while because their records are deleted from DNS, and no one knows how to reach their component parts.

Now that we know some of the potential issues we need to address, let's take a look at some of the things we can tweak to deliver a more seamless experience to the end-user and the sysadmin.

## SSSD (easier logins and dynamic updates)

**sssd** on a Linux system is responsible for enabling the system to access authentication services from a remote source such as Active Directory. In other words, it is the primary interface between the directory service and the module requesting authentication services, **realmd**. Its main configuration file is located at **/etc/sssd/sssd.conf**. As a matter of fact, this is the main configuration file we will modify.

**[ Discover [3 ways SSSD logging improvements make sysadmins' lives easier.](#) ]**

Let's have a look at its contents before configuration. Once you join the domain, it is immediately modified to contain the minimum information required for a successful logon. My file looked like this:

```
[root@centy2 ~]# cat /etc/sssd/sssd.conf

[sssd]
domains = Hope.net
config_file_version = 2
services = nss, pam

[domain/Hope.net]
ad_domain = Hope.net
krb5_realm = HOPE.NET
realmd_tags = manages-system joined-with-samba
cache_credentials = True
id_provider = ad
krb5_store_password_if_offline = True
default_shell = /bin/bash
ldap_id_mapping = True
use_fully_qualified_names = True
fallback_homedir = /home/%u@%d
access_provider = ad
```

In order to solve all three of the problems I mentioned earlier, edit your file to look like the one below:

```
[sssd]
domains = Hope.net
config_file_version = 2
services = nss, pam
default_domain_suffix = hope.net

[domain/Hope.net]
ad_domain = Hope.net
krb5_realm = HOPE.NET
realmd_tags = manages-system joined-with-samba
cache_credentials = True
id_provider = ad
krb5_store_password_if_offline = True
default_shell = /bin/bash
ldap_id_mapping = True
use_fully_qualified_names = True
fallback_homedir = /home/%u@%d
access_provider = simple
ad_hostname = centy2.hope.net
dyndns_update = true
dyndns_refresh_interval = 43200
dyndns_update_ptr = true
dyndns_ttl = 3600
dyndns_auth = GSS-TSIG
```

Most of the options are self-explanatory, and you can modify yours accordingly while we step through what some of the key options represent. More information on all the options can be obtained by checking the man page. I think it is well written. Just type `man 5 sssd.conf` at the command line. You can also view the man page for `sssd_ad` for further information.

First and foremost, the configuration file is separated into two sections. The global section, under **[sssd]** and the domain-specific options section, **[domain/[domain name]]**.

The global section contains options that affect the general behavior of `sssd`, such as the version information and related services. One key parameter under this section is shown below:

- **default\_domain\_suffix** - Set this to the domain name if you do not want to have to type the full user account name when logging in. Instead of having to type `fkorea@hope.net` always, you can just type `fkorea` and the password. This helps a lot when you have a long domain name.

The domain-specific section contains parameters that are specific to the domain you have joined. Key parameters are:

- 

## Kubernetes and OpenShift

- [Kubernetes cheat sheet](#)
- [Kubernetes: Everything you need to know](#)
- [Interactive course: Getting started with OpenShift](#)
- [Red Hat OpenShift and Kubernetes ... what's the difference?](#)
- [Interactive course: Deploy a cluster in Red Hat OpenShift Service on AWS \(ROSA\)](#)

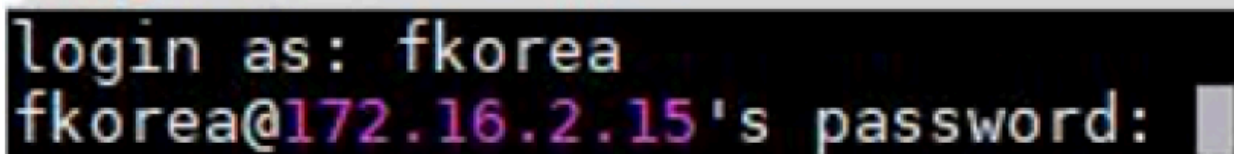
**access\_provider** - Allows you to select a provider optimized and used for interacting with AD servers for authentication purposes. It should be set to **ad**. Other values that can be used here are **ldap** and **ipa**, assuming you use those directory services.

- **id\_provider** - Allows you to select a provider optimized and used for interacting with AD servers for identification purposes. It should be set to **ad**.
- **ad\_hostname** - This should be the fully qualified hostname of the server. It should be set if the system's hostname is anything other than the fully qualified domain name. If this is not set and the **sssd** does not have access to the fully qualified hostname, dynamic updates will fail.
- **ad\_domain** - This should be the full domain name (**hope.net** in this case).
- **cache\_credentials** - This enables AD users to log in when the domain controller is offline. When this is set to **true**, credentials are cached for a period such that authentication does not fail when the back end is offline. The period of storage is also configurable.
- **fallback\_homedir** - This helps you set a home directory for AD users who do not have a home directory attribute in AD. This is different from the **override\_home** parameter that works when a home directory is set in AD for users.
- **dyndns\_update** - This enables dynamic DNS updates and accepts either **true** or **false** as a value. When dynamic updates are enabled, updates occur primarily under three conditions:
  - When the Linux server restarts.
  - When the provider comes online.
  - When the refresh interval is due.
- **dyndns\_refresh\_interval** - This value is in seconds with a practical minimum of 60 seconds. It accepts integer values and has a default of 24 hours (86400s). In this example, we set it to 12 hours. If nothing else triggers an update, an update is regularly done between.
- **dyndns\_update\_ptr** - A boolean value that specifies whether the associated PTR record is to be updated in every update cycle. PTR records are used for reverse lookups, and unless there is a good reason, this should be set to **true**.
- **dyndns\_auth** - Specifies whether the dynamic updates should be done securely or not. The setting depends on the mode accepted by AD. If AD is set to **Accept Secure Updates Only**, this value should be set to **GSS-TSIG**. If not, and you do not care for the security benefits of secure dynamic updates (despite the strong warning in AD), this value can be set to **none**.

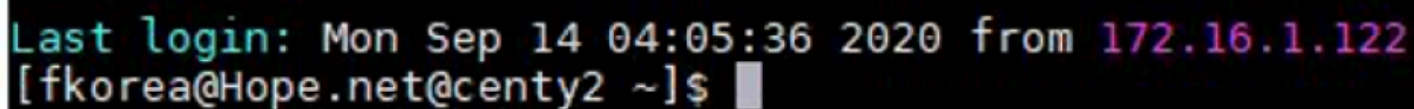
Once the configuration is complete, restart **sssd** to apply settings immediately.

```
# systemctl restart sssd
```

At this point, we are set. We can now login like we would at a Windows workstation or server.



```
login as: fkorea
fkorea@172.16.2.15's password: 
```



```
Last login: Mon Sep 14 04:05:36 2020 from 172.16.1.122
[fkorea@Hope.net@centy2 ~]$
```



# Visudo (granting admin privileges)

Users that are granted access have unprivileged access to the Linux server. For all intents and purposes, all Active Directory accounts are now accessible to the Linux system, in the same way natively-created local accounts are accessible to the system. You can now do the regular sysadmin tasks of adding them to groups, making them owners of resources, and configure other needed settings. If the user tries any activity that requires `sudo` access, the familiar error is presented. As can be seen in the inset, our user is not in the `sudoers` file.

```
[fkorea@Hope.net@centy2 ~]$ sudo ls
[sudo] password for fkorea@Hope.net:
fkorea@Hope.net is not in the sudoers file.  This incident will be reported.
[fkorea@Hope.net@centy2 ~]$ sudo ls
```

In that light, we can edit the `sudoers` file directly to grant them superuser privileges. This is not an article on granting superuser privileges, but we can use the `visudo` tool to interact safely with the `sudoers` file.

```
fkorea@hope.net  ALL=(ALL)          ALL
```

```
[fkorea@Hope.net@centy2 ~]$ sudo ls
[sudo] password for fkorea@Hope.net:
[fkorea@Hope.net@centy2 ~]$ █
```

Alternatively, we could have just added the user to the `wheel` group. The point is the user account is now available to be used by the system.

**[ Network getting out of control? [Check out Network automation for everyone](#), a free book from Red Hat. ]**

## Wrap up

Try this out in your organization or lab environment. It is obvious I just scratched the surface on this topic but this will get you pretty far into the process. Check out the respective documentation if you want to explore options not covered in this article.

Joining a Linux system to an Active Directory domain allows you to get the best of both worlds. The process is very simple and can be scripted using Bash or automated using Ansible, especially during the system's initial setup. If you are still managing a group of more than five systems without a directory service and a good reason, please do yourself a favor and get one set up. You can thank me later.





[Windows and Linux interoperability: A look at Samba](#)

Got Windows? It's time to talk about Samba, an easy to implement and free to use interoperability suite.

Posted: February 7, 2020

Author: [Ken Hess](#) (Sudoer alumni)

---



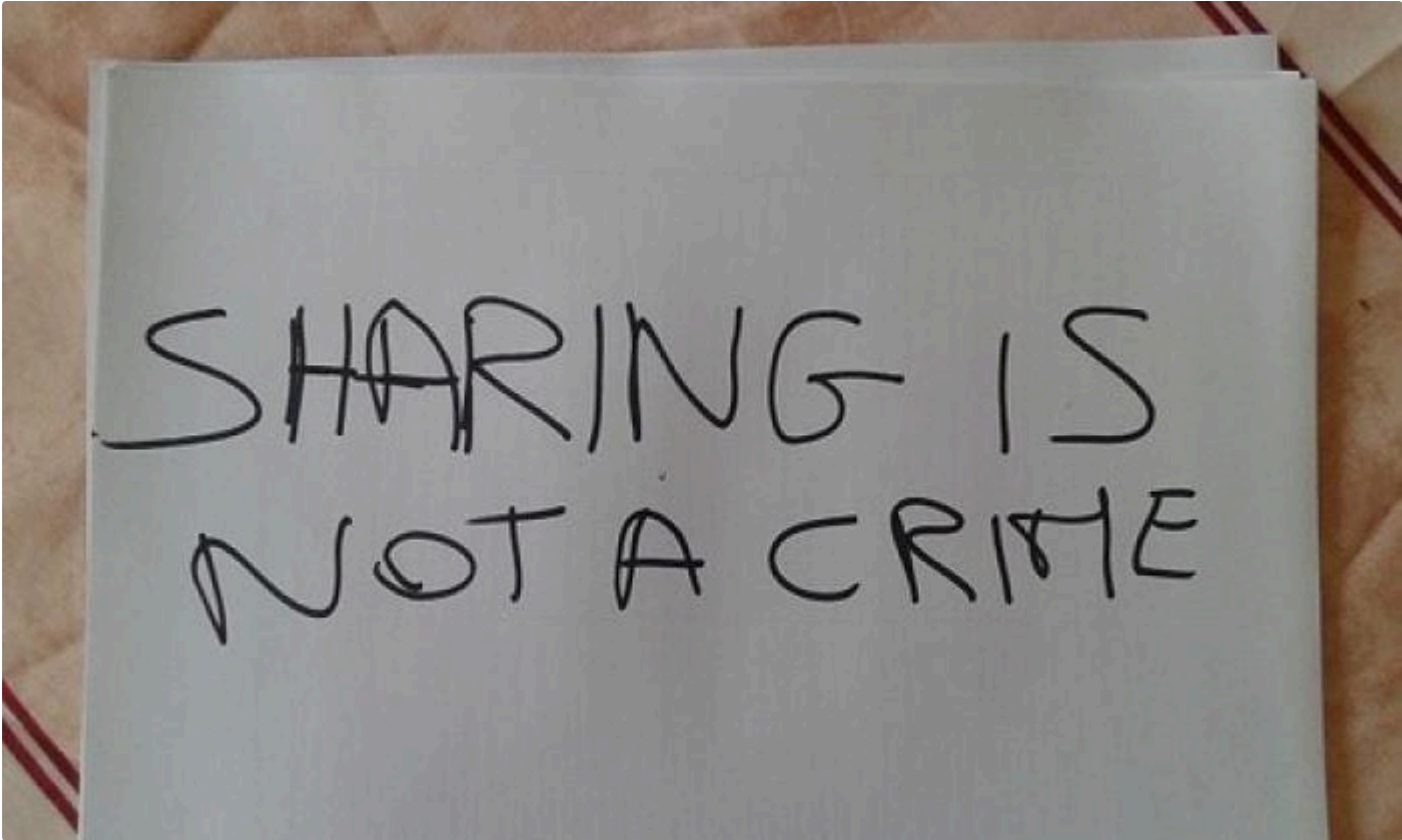
[A quick introduction to the Linux filesystem for Windows users.](#)

This quick tour offers a non-threatening introduction for DOS/Windows users to the not-so-different Linux filesystem structure.

Posted: September 5, 2019

Author: [Ken Hess](#) (Sudoer alumni)

---



[Mounting and mapping shares between Windows and Linux with Samba](#)

Create an air of interoperability in your network with Samba. Your Windows and Linux systems can work together.

Posted: March 31, 2020

Author: [Keerthi Chinthaguntla](#) (Sudoer alumni)

Topics:

Linux

Security



Edem Afenyo

Edem is currently a sysadmin with a financial services institution where he works primarily with Windows and Linux systems. He also administers VMware Virtualization environments daily. [More about me](#)

Try Red Hat Enterprise Linux

Download it at no charge from the Red Hat Developer program.



# Related Content



## [8 open source 'Easter eggs' to have fun with your Linux terminal](#)

Hunt these 8 hidden or surprising features to make your Linux experience more entertaining.

Posted: April 10, 2023  
Author: [Ricardo Gerardi](#) (Editorial Team, Sudoer alumni, Red Hat).



## [Troubleshooting Linux performance, building a golden image for your RHEL homelab, and more tips for sysadmins](#)

Check out Enable Sysadmin's top 10 articles from March 2023.

Posted: April 4, 2023  
Author: [Vicki Walker](#) (Editorial Team, Red Hat).



## [Do advanced Linux disk usage diagnostics with this sysadmin tool](#)

Use topdiskconsumer to address disk space issues when you're unable to interrupt production.

Posted: March 31, 2023  
Author: [Kimberly Lazarski](#) (Red Hat).

The opinions expressed on this website are those of each author, not of the author's employer or of Red Hat. The content published on this site are community contributions and are for informational purpose only AND ARE NOT, AND ARE NOT INTENDED TO BE, RED HAT DOCUMENTATION, SUPPORT, OR ADVICE.

Red Hat and the Red Hat logo are trademarks of Red Hat, Inc., registered in the United States and other countries.



- About Red Hat
- Jobs
- Events
- Locations
- Contact Red Hat
- Red Hat Blog
- Diversity, equity, and inclusion
- Cool Stuff Store
- Red Hat Summit

- 
- © 2023 Red Hat, Inc.
  - Privacy statement
  - Terms of use
  - All policies and guidelines
  - Digital accessibility
  - Cookie preferences