FreeIPA in 2025

- Install FreeIPA in 2025

Install FreeIPA in 2025

FreeIPA is an open source Identity, Provider, and Authorization management system aimed at Linux usage more widely than Windows. It's not thta Windows can't be used, it's just hat it is more difficult to setup than some may want (at least at first).

Today, we'll install the FreeIPA server and we'll do our best to configure it for your best case usage scenaorio. I'll talk about some of the gotchas you might run into, and try to explain why we are doing some of the things we are doing, as well as wlak you thorugh the terminology of what we are entering. I do this becuase I hope to help you understand more quickly what I had to muddle through at a snails pace to understand more clearly.

Installing the FreeIPA Server

I highly recommend you setup a server, virual machine, or LXC, LXD, Incus container running CentOS Stream 9 or later for hosting the FreeIPA Server. It will simply make getting things setup much smoother for you.

When you've setup your server instance, create a non-root user, and give the user root privileges.

```
adduser <username>
```

Redhat / CentOS / Fedora
```
usermod -aG wheel <username>
```

Debian / Ubuntu
```
usermod -aG sudo <username>
```

In the RedHat / Fedora / CentOS world, the sudo user is in the 'wheel' group, and not the 'sudo' group like on the Debian based OSes.

If you haven't already, install openssh-server, so you can SSH to your system, just to make it easier to work on if you are remote.

```
dnf install openssh-server -y
```

We need to start and enable the 'sshd' service in order to use SSH.

```
systemctl start sshd
```

```
systemctl enable sshd
```

Now, you should be able to ssh to your server if needed.

Next, let's update our server to make sure it's got the latest packages.

`dnf update -y`

Now, we can log out, and back in as our non-root user. Now, I know this seems silly, but have faith. We'll need to do a lot of commands requiring root privileges, so we'll just su into the root user for the time being. Normally, however, it will benefit you to log in as a non-root user, and just work from the non-root user's shell.

`sudo su`

Enter your superuser password, and you should be at a root prompt.

Setup a Static IPv4 on our Server

These next set of commands will be used to set a static IP on our server. If your server already has a static IP, then you can skip ahead.

use the network manager cli (nmcli) command to get an initial look at your server's network info as it is now.

`nmcli`

```
eth0: connected to Wired connection 1
        "eth0"
        ethernet (veth), 00:16:3E:AB:D9:85, sw, mtu 1500
        ip4 default, ip6 default
        inet4 192.168.34.17/24
        route4 192.168.34.0/24 metric 100
        route4 default via 192.168.34.1 metric 100
```

Take note of the name of your connection. It's on the first line after the "connected to". IN my case it's called 'Wired connection 1'. You'll need this for the following commands.

We'll use the nmcli to set our values as follows: Replace my connection name with yours if needed, as well as the ip address you need for your server, and gateway, dns, etc addresses appropriately for your network.

```
$~ nmcli connection modify 'Wired connection 1' ipv4.addresses 192.168.34.17/24
$~ nmcli connection modify 'Wired connection 1' ipv4.gateway 192.168.34.1
$~ nmcli connection modify 'Wired connection 1' ipv4.dns 192.168.34.1
$~ nmcli connection modify 'Wired connection 1' ipv4.dns-search srv.world
$~ nmcli connection modify 'Wired connection 1' ipv4.method manual
$~ nmcli connection down 'Wired connection 1'; nmcli connection up 'Wired connection 1'
```

In the above commands, the `$\~` is not part of the command, please make sure ***not to*** copy that if you are copy / pasting the commands.

First we set the ipv4.addresses, then we set the ipv4.gateway, then ipv4.dns. Next, we tell the connection it is okay to search the world level for any requests. We also set the ipv4.method to 'manual' so it won't pull a dhcp address again int he future, and then we take the connection down, and bring it back up again to set our changes in place.

Now you can test your connection by pinging a machine on your LAN, and if you have internet, ping a site like `example.com`.

```
ping example.com
```

You can use CTRL + C to stop the ping when you're ready.

Set our Server's Hostname

For FreeIPA wee need a proper hostname and domain name setup on it. We can check what our current hostname is by using the hostnamectl comamnd.

```
hostnamectl
```

In my original output the "Static Hostname" was just 'freeipa'. I need to change this to be freeipa.fixitdelrio.local (and you should set yours to the domain you want for your FreeIPA system.

```
hostnamectl hostname freeipa.fixitdelrio.local
```

Now, recheck your hostnamectl output, and ensure it's updated properly.

We also need to modify our /etc/hsots file to have this proper hostname setup in it.

```
nano /etc/hosts
```

My initial file looks like:

```
127.0.1.1       freeipa
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
```

I want to remove the loopback line which starts with 127.0.1.1, and then add a couple of lines above it.

I'll be putting my FreeIPA server on my Netbird VPN so that I can use the server across the VPN from machines that aren't on my home LAN as well as those at home. IF you are only setting this up locally, then you may only need to add the local LAN IP you setup as your static IP earlier.

My final file looks like

```
100.75.10.13  freeipa.fixitdelrio.local freeipa
192.168.34.17 freeipa.fixitdelrio.local freeipa
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
```

Here, I've added two lines to the top of my /etc/hosts file. First, the line for my VPN address for this machine. On that line, I have the IPv4 address, the fqdn (fully qualified domain name), and the hostname portion alone.

```
100.75.10.13        freeipa.fixitdelrio.local        freeipa


ipv4                  fqdn                       hostname
```

The next line has the same format, except I entered the local static IPv4 address I setup for this server in the earlier nmcli steps.

Save this file with CTRL + O, then press Enter to confirm. Exit with CTRL + X.

Install the FreeIPA Server Package

Now, we'll install the FreeIPA server package to our server. Feel free to exit the root user if you prefer.

`sudo dnf install freeipa-server -y`

This process is going to install a lot of packages, and it will take some time, so let it run, and be patient. When it's complete, we can check the version by using the ipa command:

`ipa --version`

You should see the proper version information with no errors.

`VERSION: 4.12.2, API_VERSION: 2.254`

Configure the FreeIPA Server Install

We are about to start the configuration of the FreeIPA server. Depending on what you want from the server will depend on how your installation progresses. I did not setup the DNS or NTP server portions on my FreeIPA server. This can be done with tools I'm already running, so I chose not to use it. You, however, may want to do this, and if you do, there may be additional steps you need to take during the configuration.

In the command line, we'll enter:

```
sudo ipa-server-install
```

While the command says 'install', it really is more of a configuration at this point.

You'll be prompted for several bits of information during this process, just be patient, and read carefully what it's asking. Each prompt will have a default answer set in square brackets"[ ]". If your answer is the same as that given in square breackets, then you can simply press the Enter key to accept the defaults and move forward.

You'll be asked about DNS, as I said, no need for me to say 'yes' to this setup, and it is defaulted to '[no]', so I just press Enter.

You'll also be asked for the domain name of your server. If you've setup your hostname as I describe above, it should detect the domain portion, which in my case is "fixitdelrio.local", and that will be filled in the default option. If your's is not the default, then type in your server domain.

Next, you'll be prompted for the server's fully qualified domain name / hostname. Again, if you've filled in this on the hostname setup above, it should be detected and automatically set as the default. If so, feel free to just press Enter, and if not, then type it in.

You'll be prompted for the Kerberos Realm, which is your server's domain but in all capital letters. Mine is FIXITDELRIO.LOCAL for example. Agai, it should be detected and set as the default.

You'll be asked about time synchronization with chrony, I don't need this, and suggest you also answer 'no' to this, which should be the default. We can setup time sync later.

Once you're through all the information prompts, it will display a summary of what's about to be done, and this will prompt whether you are certain you want to proceed. It will also bedefaulted to [no]. Make sure to type "yes" at this prompt, then press Enter. The configuration will now proceed.

NOTE: This configuration takes a while, for me, it took about 7 minutes, so be patient, and let it run. There are several steps where it may seem stuck, but just let it run.

When it's complete, it will display a bit of useful information, so make sure to copy and save it somewhere while we complete our system setup.

Firewall Setup / Install

You may already have a firewall installed on your system, and if so, then we'll want to make sure the proper ports are open and available. If not, we'll install it real quick.

If you're following along and using CentOS, Fedora, Redhat, then we'll install FirewallD, but if you're on a Debian based system, you'll likely use UFW (uncomplicated firewall). I'll put the commands to install each, as well as the commands to set the ports needed for each.

Let's test to see if we have a firewall installed:

CentOS / Fedora / RedHat

```
sudo systemctl status firewalld
```

Debian / Ubuntu

```
sudo ufw status
```

If you get an error response for either command based on your distro of choice, then you likely need to install the firewall software.

Let's start with CentOS / Fedora / RedHat:

```
sudo dnf install firewalld
```

Once the install completes, we need to start the firewall running:

```
sudo systemctl start firewalld
```

and then enable it, so it will auto-start on reboot:

```
sudo systemctl enable firewalld
```

Now you can check the status again with

```
sudo systemctl status firewalld
```

You should now see it as 'active'.

If you are running Ubuntu / Debian, let's get ufw installed:

```
sudo apt update && sudo apt install ufw -y
```

After the install completes, the firewall should be running. You can check it again with

```
sudo ufw status
```

Set our Firewall Rules for FreeIPA Server

CentOS / Fedora / RedHat

```
sudo firewall-cmd --add-port={80,443,389,686,88,464,53}/tcp --permanent
```

```
sudo firewall-cmd --add-port={88,464,53,123}/udp --permanent
```

Now we can check our ports with

```
sudo firewall-cmd --list-ports
```

and you should see all of the ports we entered above listed now.

You can also check the service allowed with

```
sudo firewall-cmd --list-services
```

You should see SSH and a couple of others listed there as well.

Debian / Ubuntu

First , lt's make sure port 22 is open and allowed so we don't get cut off from our server when we neable the UFW. We can check the ufw status with

```
sudo ufw status
```

If you see 'inactive', then you'll know it's installed, but not enabled. We want to set all of our ports, then enable it so we don't disconnect and block our own access to the server.

```
sudo ufw allow 22,2222
```

Now, we'll add the ports we want to have open for TCP:

```
sudo ufw allow 80,443,389,636,88,464,53/tcp
```

```
sudo ufw allow 88,464,53,123/udp
```

Finally, let's make our firewall active:

```
sudo ufw enable
```

and now we can check our firewall with

```
sudo ufw status
```

Initialize our FreeIPA Admin

We need to initialize our FreeIPA Server admin account:

```
sudo kinit admin
```

Adding a User from the CLI

Now we can add a user to the system from the command line, but know that you can also add users from the Web GUI.

```
sudo ipa user-add <username> --password --homedir=/home/<username> --shell=/bin/bash
```

In the above command we tell FreeIPA to create a new user, replacing the placeholder with the actual username you want, then tell it to prompt for a password for the user, to create a home directory for the user, and that the user should have the BaSH shell when they login on machines using LDAP from FreeIPA.

When you press Enter, the system will prompt you for the user's first name, last name, password, and password confirmation.

IMPORTANT: Remember the password you give. You need to provide this to the user. The first time they log in, they'll be prompted to change this password. This is done as a security measure and it is intentional.

Editing User Details in the CLI

Again, it's useful to know that you can do a lot from the ipa command line interface commands. You can

- Edit a User's password

```
sudo ipa user-mod <username> --password <new password>
```

- Edit a User's email

```
sudo ipa user-mod <username> --email <updated email address>
```

- You can get more editing abilities from the documentation for RedHat IdM

The FreeIPA Web GUI

If you haven't already, and you are accessing your FreeIPA server from a different machine / VM, then you will want to add the /etc/hosts entries in your current desktop for the FreeIPA server you setup.

```
nano /etc/hosts
```

then add the entries for the IPv4 addresses to the end of this file, along with the fqdn and hostname for the ip(s) you will be using.

```
192.168.34.17    freeipa.fixitdelrio.local    freeipa
100.75.10.13     freeipa.fixitdelrio.local    freeipa
```

Save with CTRL + O, then press Enter to confirm, and exit the nano editor with CTRL + X.

Now open your favorite modern web browser and navigate to https://. In my case I went to https://freeipa.fixitdelriio.local

You'll be presented with a certificate warning. Just tell the browser it's okay and accept the certificate to move forward.

Once done, you'll be presented with a login screen. Now, you can login as the user you created above, but you really want to first login as your admin user created earlier on, as this user will have full administrative access in the FreeIPA web gui.

Once logged in as an admin, you can go to the various tabs and see all of the power you have. I go through a brief overview in my video, so make sure to check it out for more details.

First off, I recommend making your user that you created above, a full admin user, so you can use that user for anything you need in the Web GUI instead of the admin user.

Next, go through, and understand the Host Based Access Control, Roles, Groups, and Sudo Roles setup. This is where you'll get the most bang for your buck in FreeIPA. In a future video we'll be integrating Authentik with FreeIPA to get that full SSO functionality for ourselves and our users.

Add a FreeIPA Client Machine

Much of the necessary setup for the server is needed for the FreeIPA Client. We need to give our client a static IP address, a fully qualified domain name / hostname, and ensure we have SSH setup properly. Check the Instructions from the server portion and use the same methods for the client machine.

Configure the FreeIPA client

You should have the following already configured on the client before starting this process:

- Static IPv4 Address
- Fully Qualified Domain Name / Hostname in `/etc/hosts` and `/etc/hostname`
- The domain and IPv4 of the server setup in `/etc/hosts` so the client can reach it using it's FQDN.
- (optional) VPN installed and configured, with VPN information of server also setup in `etc/hosts`.

Install the FreeIPA Client

Next, we need to install the FreeIPA client on our client machine.

RedHat / CentOS / Fedora

```
sudo dnf install freeipa-client -y
```

Debian / Ubuntu

```
sudo apt update && sudo apt install freeipa-client oddjobs-mkhomeidr -y
```

Here we add the package `oddjobs-mkhomedir` as this allows FreeIPA to make a home directory when a new user logs into the system using their FreeIPA (LDAP) credentials.

During this install you'll be prompted to enter the Kerberos Realm. If you've setup the system already, as you should have, with hostname, fqdn, etc. thenit should be auto-filled with the proper realm info. The Realm, recall, will be your domain in all caps (e.g. EXAMPLE.LAN).

Next, you'll be prompted to enter the FQDN of the Kerberos Server for tht Realm, this is just the FQDN of your FreeIPA server (e.g. ipa.example.lan).

After that you'll be prompted for the Administrative server for the Kerberos Realm, and again is't just the FQDN of your FreeIPA Server (e.g. ipa.example.lan).

Once that is done, it's time to configure the FreeIPA Client software.

```
sudo ipa-client-install --mkhomedir --no-ntp
```

Only add the `--no-ntp` portion if you are not using the NTP server on your FreeIPA server.

If you, like me, are not using your FreeIPA server for DNS, then you'll be given a warning that DNS isn't setup for auto-discovery, and you'll be prompted for the FreeIPA server domain. This is just the somina portion (e.g. example.lan).

Next, you'll be prompted for the FreeIPA server name (e.g. ipa.example.lan)

And again, you'll see the DNS warning, and prompted to proceed with fixed values, and no DNS discover, type 'yes' and press Enter. Then you'll be shown a summary of informaiton for the configuration. Make sure it's all correct. Then type 'yes' again, and press Enter. The configuration will continue.

You need to provide a FreeIPA user with permissions to add the new system. You can use the Admin user, or create a new user with this permission through the Web GUI if you prefer.

The configuration will continue, and once complete, you can reboot the client, then you should be able to login with a user from FreeIPA instead of a local user.

Congratulations, you can now setup FreeIPA far and wide for all your users and devices.