# 10.5.8 Synchronize Time with NTP

Click one of the buttons to take you to that part of the video.

Synchronize Time with NTP 0:00-0:33

In this demonstration we're going to look at synchronizing the system time using the ntp protocol. Let's go ahead and configure ntp on this system. We're going to use the 'su -' command first to switch to our root user account. The first thing I'm going to do is use the 'rpm' command with the '-qi' option to see whether or not the ntp package has already been installed on this system, and it has not. With some distributions, it will be installed by default, others it won't. With this fedora system, ntp is not installed by default.

---

ntp Package 0:34-0:50

What I am going to do is use the 'yum' command to install the ntp package. At this point, we've identified which packages need to be installed, along with all the dependent packages, and it's prompting me as to whether I want to go ahead and download and install them, and I'm going to say yes, please do.

---

daemon 0:51-1:01

At this point, the ntp package has been installed on my system. I now have an ntpd daemon that I can run in order to synchronize network time.

---

ntp.conf 1:02-5:07

Before we do that, we have to tell it which time providers we want it to synchronize time with, and that is done by editing the '/etc/ntp.conf' file. If we scroll down here, we can see the different servers that it's configured by default to synchronize time with.

The syntax for configuring the ntp.conf file is to simply specify 'server', space, and then either the IP address or the DNS name of the time provider that you want to synchronize time with. As you can see, when we installed ntp on this system, four different time providers were configured by default.

These are time providers that exist out on the internet and they're public time providers, meaning that they offer time synchronization over the internet to basically anybody who wants to synchronize time with them. The first server is 0.fedora.pool.ntp.org. The second server is 1.fedora.pool.ntp.org, 2.fedora.pool.ntp.org, and 3.fedora.pool.ntp.org.

The first thing you notice here is the fact that you can synchronize time with multiple different time providers instead of just one. And that's useful in situations where maybe if this one is not available for some reason, it can synchronize time with the others as well.

There is something very important you need to understand about these servers, and that is the fact that these URLs do not point to a single physical server. You will see that these URLs point to pool.ntp.org. These are part of the pool.ntp.org project, as indicated up here. Basically, pool.ntp.org is a big virtual cluster of lots of time providers all over the world that provide public time synchronization services over the internet. pool.ntp.org is designed to provide load balancing.

Back in the old days when I first started using ntp, we would go out on the internet and we would find a list of different public ntp time providers and we would configure these server directives with the IP address or DNS name of those providers. This could cause a problem on occasion if lots of people around the world decided to use the same ntp time providers. Those servers could get overloaded.

Using pool.ntp.org, however, these URLs don't point to the same server all the time. Instead, they point to a random time server within the pool of available time providers, and the servers that these URLs point to change every single hour.

This provides load balancing, and it prevents one time server from being overloaded with public time synchronization requests, and it also makes configuration really easy on our end of things because I don't have to remember lots of different addresses for lots of different time providers out on the internet. I just remember pool.ntp.org and we're ready to go.

You could specify additional servers if you wanted to. We'll talk about that more in just a minute. For our purposes here, however, let's just go with the default time configuration set up here--where we go out and query these four different time providers and get time from them. We'll exit without making any change to the file.

Before we start the ntpd daemon, we need to actually get our local system time--here on this system--relatively close to the time on the time servers that we're going to be connecting to.

Here is the key thing that you must understand when working with ntp, and that is the fact that if your local time is more than 17 minutes off from the time server, ntp declares it to be insane time and it will never synchronize. It is so far out of synchronization that ntp doesn't want to deal with it.

And this is a common mistake made by new system administrators. They get everything configured, they turn on the ntp daemon, and time never syncs and they can't figure out why. It's because their local system time was so far off from the time provider that ntp just basically gives up and says, "Hey I can't deal with this, you're way off."

To prevent this, you use the 'ntpdate' command.

---

## ntpdate Command 5:08-8:19

You go out and perform a quick, one-time sync with the time server to get the two devices within this 17-minute window. ntpdate is not like the netdate command. It will not perform an exact synchronization immediately. It will just kind of gradually nudge the time on your local system and get it close to the time provided by the time server.

Before we can run ntpdate, we have to make sure that the ntpd daemon is not actually running on the system, because you can't run ntpdate if that daemon is running.

Let's use the 'systemctl' command, 'status ntpd' and we're good to go. It is currently inactive; it is not running. We can go ahead and run 'ntpdate', and then we specify the URL of the system we want to synchronize time with.

Let's just use 'pool.ntp.org', and that will randomly pick one of the available servers in the time provider pool, and it will perform a quick one-time update of our local time through the time provider to kind of get the two in the same ballpark. Okay.

The ntpdate command went out, got the time from a random server within this pool of time providers, and it said we were just a little bit off, not very much at all. The times were already pretty close. They were actually close enough that the ntpd daemon would have been able to synchronize time without running ntpdate, but it's always a good practice to do this first just in case.

Be aware that sometimes if the time were off a lot, you can actually run the ntpdate command multiple times to gradually nudge your system time closer and closer and closer to the time on the time provider. There have been occasions where I have had to run two or three or four times to get the time close enough to where I felt comfortable.

Because our time is really, really close already, we can just go ahead and start the ntp daemon--systemctl. We'll run the 'systemctl' command again, and instead of using status we'll run 'start'. And now let's check the status to make sure everything loaded properly and we're good to go.

The status is active, the daemon is running, and down here we can see the different interfaces that the ntp daemon is listening on. You can see that it actually is listening on all of our configured IP addresses. It is listening on the IPv4 address on our loopback interface. It is listening on the IPv4 address on our Ethernet interface. It is listening on the IPv6 loopback interface, and it is also listening on the IPv6 address on our standard Ethernet interface as well.

If we wanted to tighten this up a little bit, we could actually go into our ntp.conf file and specify exactly what interface we wanted to listen on instead of all of them, because there really is no reason to be listening on our loopback interface, because no time provider is going to be sending network time over the ntp protocol to our loopback interface. For our purposes today, this will work just fine.

---

## ntpq 8:20-10:58

At this point we can use the ntpq command to see how well time is being synchronized between our local system and the time providers we configured in the ntp.conf file. We enter 'ntpq -p'. And when we do, we see the actual DNS names of the servers that we're synchronizing time with.

Remember I said a minute ago that servers referenced by the URLs in pool.ntp.org change every hour. So if I were to come back and run this command again later in the day, I would actually see different server names listed, because we would rotate and start synchronizing with a different set of ntp servers.

The key thing you want to notice over here are our delay, offset, and jitter parameters. At first glance you might be a little concerned, because it might appear that we're way off, such as 7 seconds off or 10 seconds off, with a fairly substantial amount of jitter. These are actually measured in milliseconds, not whole seconds, so we're actually 7 milliseconds off from this server, 4 milliseconds off from this server, 10 milliseconds off from this server. And you know what, as far as I'm concerned, being a few milliseconds off doesn't hurt anything.

At this point, time is synchronized. The ntp daemon on this system will continually poll all of the servers specified in our ntp.conf file throughout the day, and will keep time synced up between the two. Here's a cool thing that you can do with ntp. Understand that ntpd daemon running on the system is a time client, but it also is a time server at the same time.

If I had a network of 1,000 computers, I could go through each computer and configure them to go out and get time from pool.ntp.org. And that would be fine, it works, but it creates unnecessary network traffic, and also places a little bit of load on the time providers out on the network--especially if everybody in the world were to configure things in that same way.

A better way to do things is to pick a limited number of computers on your network, configure them to go out and talk to the servers out on the internet, but then configure the rest of the computers in your network to point to these computers.

To do that, you would just go back into your 'ntp.conf' file and you would remove these four lines that were added by default. And then specify something, such as server '10.0.0.104' or whatever the IP address is of the system that you've configured to go out on the internet, and get its time. Let me go ahead and 'exit' out of this file without making that change and we're good to go.

---

Summary 10:59-11:26

That's it for this demonstration. In this demo we talked about synchronizing time using the ntp protocol. We first installed the ntp package on the system. We then performed a quick one-time update using the ntpdate command. We then configured the ntp daemon's configuration file, ntp.conf. We enabled the daemon. We then verified the status of the ntp protocol using the ntpq command. And then we ended this demonstration by talking about the fact that the ntp daemon can function as both a client and as a server at the same time.

---