15.2.3 Configure User Security and Restrictions

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

Configure User Security and Restrictions 0:00-0:14

In this demonstration, we're going to talk about configuring user restrictions. We're going to look at this topic from three different aspects. First of all, password aging, and then setting login limits, and then finally setting user limits with the ulimit command.

Password Aging 0:15-2:24

Let's begin by talking about password aging. In today's security environment, you need to be very careful that you configure your passwords to expire after a set period of time. This is called password aging. The key thing to remember here is that the longer a user has the same password, the more likely that it's going to be compromised.

To prevent this, you need to configure aging for your user passwords. This is done with the chage command. Let's take a look at the chage man page. The syntax for using chage is to enter the chage command, followed by a series of options, then the user account that you want to apply those options to.

Some of the more useful options you can use are listed right here. First of all, we have the -m option, which specifies the minimum number of days between password changes.

You also have the -M option that specifies the maximum number of days between password changes.

We also have the -w option that specifies the number of warning days a user is going to get before the password change is required. Go ahead and 'exit' out of the man page.

Let's work through an example. I'm going to enter 'chage' here at the shell prompt as my root user. Then I'm going to specify the '-M' option to set the maximum number of days between password changes. Let's set that to '60', 2 months.

For many organizations, that would be way too long. A lot of organizations force password changes every 30 days. Let's specify '-w' to specify that a warning is given seven days before the password is about to expire, so the user has plenty of time to make the change before the password actually expires.

Then we have to specify who we want to apply this change to. Let's apply it to our 'ksanders' user account. The password is set to expire for the ksanders user in 60 days. Let's use the 'tail' command to verify this by viewing the '/etc/shadow' file.

Look at the very end of the file. We take a look here at the ksanders user account. We can see here that the maximum number of days between password changes is 60 and the ksanders user will get seven warning days before a password is about to expire to go and change their password.

User Limits 2:24-5:15

The next thing we need to discuss is configuring user limits. Understand that on Linux you can configure limits for how many times a user may log in, how much CPU time they use, how much memory they can use, and so on. This is done using the pam_limits PAM module. PAM stands for Pluggable Authentication Module.

We configure these limits in the /etc/security/limits.conf file. The syntax for this file is shown here. First, we specify the domain. The domain, as noted here, can be a user or it could be a group.

If you want to use a group, you have to put the @ sign in front of the group name to indicate it's a group, or you can use a wild card--such as listed here--to apply it to everybody. Scroll down a little bit so we can see more.

Next we specify the type. As noted here, the type can have two different values. We can either specify a hard limit that can never be exceeded, or we can specify a soft limit that can be exceeded but just temporarily.

Then we specify the item. This is the particular thing that's going to be limited. As you can see down here, we have lots of different options for what we can limit what the item can be.

We can specify core to limit the size of the core dump files. We can specify data to configure a maximum data size and memory for the user's programs. We can specify fsize to restrict the maximum file size. We can specify nofile. We can specify the maximum number of open files.

We can specify rss to set the maximum resident set size and memory.

We can specify stack to set the maximum stack size. We can specify cpu and set the amount of CPU time that can be used by a single process in minutes. We can specify nproc to specify the number of concurrent processes that will be allowed. Or we can come down here and specify maxlogins to specify the maximum number of simultaneous logins that we allow for this user.

We can also specify priority if we want to, to specify the priority to run user processes with, for this particular user account. Once we specify the item, we scroll back here, and we then specify the value for that item. This is the limit that we're going to configure.

With this in mind, let's go down here and let's create a new limit. We're going to configure a limit for our ksanders user. I'll enter 'ksanders' for our domain. Next we want to specify a 'hard' limit and the item we want to limit is our 'cpu' time. This is the maximum amount of CPU time that a single process run by this user is allowed to consume. We have to specify how much that is. We do that in minutes. We'll specify '10' minutes. Let's press Escape, and we'll 'exit', which saves the changes to the file and exits our editor. To apply the change, we would then have to reboot the system.

'ulimit' Command 5:15-7:37

The last thing we're going to look at the in this demonstration is the ulimit command. To be honest, I don't really care much for the ulimit command. It's not as useful, in my opinion, as the limits.conf file that we just looked at.

The ulimit command does allow you to configure limits on system resources on a per-user basis, much like we did in the limits.conf file. However, be aware that any limits that you configure with ulimit will affect programs that are launched only from within the shell prompt.

If, on the other hand, the user were to come over here and launch a graphical application on the desktop, then the limits you specify with ulimit aren't applied. That's why I don't really care for it that much.

The syntax for ulimit is to enter 'ulimit', followed by the options you want to use, then the limit that you want to specify. There are many different limits you can configure with ulimit. Take a look at the ulimit man page to see a full list of what you can do.

For our purposes today, let's just run 'ulimit -a' to, first of all, view the current limits for my user account. You can see what they are right here. Let's 'clear' this screen. Let's run that command again so it's a little clearer as to what we're looking at. You can also use ulimit to set a limit, as well as view them. Let's set a basic limit.

Let's suppose we want to set a soft limit of 100 concurrent process for my user account. To do this, we would enter 'ulimit-s' to set a soft limit. If we wanted to, we could use -h to set a hard limit. Then we have to specify which limit we want to use. We'll use the '-u' option to set the maximum number of processes available to the user.

How did I know that that's the right option to use? I looked at the man page. That's what you should do as well. Then we have to set that to a specific value. Let's enter '100'. My user account can only have a maximum of 100 concurrent processes. Enter and the limit is applied.

If we run 'ulimit -a' again, we see that my max user processes has dropped from 7922 down to 100, which in the real world, depending on how the system is being used, would not be anywhere near enough. That's essentially how you use ulimit.

Basically, you use ulimit to constrain what users can do so you don't end up with one user hogging up all the system resources and not allowing other users on the system to access them.

That's it for this demonstration.

Summary 7:38-7:45

In this demonstration we talked about how to set up user restrictions on Linux. We first looked at password aging. We looked at login limits and we ended this demonstration by looking at the ulimit command.

Copyright © 2022 TestOut Corporation All rights reserved.