EXEC Timeout

Packet Lab

What is ‘EXEC timeout’

To allow access to your Cisco devices you can configure a number of ‘lines’. Examples of these lines are the console port (line con 0), virtual ports for remote access (vty line), and the auxiliary port (line aux 0). When a user ends their session with the ‘exit’ or ‘logout’ command IOS (usually!) clears that line so that other users can access it. But what happens if a user goes home for the night and forgets to disconnect their session?

Since there are a limited number of lines, you need a mechanism to make lines available if a user has not used the line for a period of time. That’s where the ‘EXEC timeout’ comes into play.

exec-timeout

exec-timeout - To set the interval that the EXEC command interpreter waits until user input is detected, use the **exec-timeout** line configuration command. Use the **no** form of this command to remove the timeout definition.

exec-timeout *minutes* [*seconds*]
no exec-timeout

Syntax Description
minutes - Integer that specifies the number of minutes.
seconds - (Optional) Additional time intervals in seconds.

Default - 10 minutes

If no input is detected during the interval, the EXEC facility resumes the current connection. If no connections exist, the EXEC facility returns the terminal to the idle state and disconnects the incoming session.

That's Cisco's convoluted way of saying "if the user doesn't type something within the exec-timeout period, then the router will kick his sorry ass off."

exec-timeout

There are a couple of things that you need to understand about the EXEC timeout:

This is an ‘idle’ timer and not an ‘absolute’ timer. I personally feel that this command should have been called ‘idle timeout’ as that is what it really is. Don’t confuse the functionality of the ‘exec-timeout’ command with that of the **absolute-timeout** command. If a user inputs anything within the EXEC timeout interval, then that session is kept alive. The **absolute-timeout** command (covered in another lesson) will boot a user from the line after a period of time, regardless of whether or not they are idle.

The more devious among you may already have plans to mess with your co-workers by setting the EXEC timeout to zero with the **exec-timeout 0 0** command. While I applaud your evil nature, this will do exactly the opposite of what it looks like it should do. You can disable the EXEC timeout with either of the following commands: **exec-timeout 0 0** or **no exec-timeout**. Don’t despair, **exec-timeout 0 1** is still available for messing with co-workers.

Due to the goofy value ranges for this command, **exec-timeout 1 30** is the same as **exec-timeout 0 90**.

exec-timeout

```
r1(config)# line con 0
r1(config-line)#exec-timeout ?
  <0-35791>  Timeout in minutes

r1(config-line)#exec-timeout 0 ?
  <0-2147483>  Timeout in seconds
  <cr>

r1(config-line)#exec-timeout 0 59

r1(config-line)#exec-timeout 0 90
line con 0
 exec-timeout 1 30

r1(config-line)#no exec-timeout
line con 0
 exec-timeout 0 0
```

Verification

show terminal and **show line x** are the commands you'll use for verification. Unfortunately, both commands produce a ton of output.

```
r2#show terminal
Line 0, Location: "", Type: ""
Length: 24 lines, Width: 80 columns
Baud rate (TX/RX) is 9600/9600, no parity, 2 stopbits, 8 databits
Status: PSI Enabled, Ready, Active, Automore On
Capabilities: none
Modem state: Ready
Modem hardware state: CTS* noDSR  DTR RTS
Special Chars: Escape  Hold  Stop  Start  Disconnect  Activation
                ^^x    none   -     -       none
Timeouts:      Idle EXEC    Idle Session   Modem Answer  Session   Dispatch
                never         never                        none      not set
                            Idle Session Disconnect Warning
                              never
                            Login-sequence User Response
                             00:00:30
                            Autoselect Initial Wait
                              not set
<--output truncated -->
```

Verification

```
r2#show line 66
   Tty Typ     Tx/Rx    A Modem  Roty AccO AccI   Uses   Noise  Overruns   Int
    66 VTY              -    -      -    -    -      0       0     0/0       -

Line 66, Location: "", Type: ""
Length: 24 lines, Width: 80 columns
Baud rate (TX/RX) is 9600/9600
Status: No Exit Banner
Capabilities: none
Modem state: Idle
Special Chars: Escape  Hold  Stop  Start  Disconnect  Activation
                ^^x    none   -     -       none
Timeouts:      Idle EXEC    Idle Session   Modem Answer  Session   Dispatch
               00:10:00        never                        none     not set
                            Idle Session Disconnect Warning
                              never
                            Login-sequence User Response
                             00:00:30
                            Autoselect Initial Wait
                              not set
<--output truncated -->
```

Verification

show terminal will show you the terminal settings for the line you are currently connected on, while **show line x** will show the terminal settings for a specific line.

To reduce the amount of output you can Grep with **| section Timeouts** or **| begin Timeouts**

```
r2#show terminal | section Timeouts
Timeouts:      Idle EXEC    Idle Session   Modem Answer  Session   Dispatch
                never         never                        none     not set
                            Idle Session Disconnect Warning
                              never
                            Login-sequence User Response
                             00:00:30
                            Autoselect Initial Wait
                              not set
```

EXEC Timeout As A Security Feature

You can use the EXEC timeout as a very rudimentary security feature in that it will disconnect an idle user’s session, thereby potentially reducing the risk of an engineer walking away from his computer with an active session still up. If you are logged into a device and take off for that rock star 2 hour lunch and don’t lock you computer, someone could potentially walk by and do a little network configuration (**write erase** followed by **reload** is good) while you’re out. At least with the EXEC timeout set, there’s an additional (minimal) layer of security in play.

What Value Should I Use For The EXEC Timeout?

There really is no “one size fits all” answer to this question. The default EXEC timeout of 10 minutes is generally good, but I prefer to set it to 30 minutes. There are a lot of times when I am working on a device and need to do something else (research, slap a coworker, take a piss) and 10 minutes seems to go by fast. If I don’t get back to something within 30 minutes, then it probably was not that important in the first place.

One thing that I would recommend that you avoid is disabling the EXEC timeout feature. While I often turn this off for labs, I would never disable this feature in production. Especially on the console port. It doesn’t make any sense to implement authentication on a console port and then disable the EXEC timeout. If the engineer forgets to logout and simple disconnects the console cable, then any goofball could just plug into the console port and start configuring the network…with your credentials.

Packet Lab

Summary

By setting the EXEC timeout, you’re telling the Cisco device to automatically disconnect users on a line once they have been idle for the duration of the EXEC timeout value. This feature is configured on a line-by-line basis. This feature allows the Cisco device to clear lines so that they are available to other users. It also acts as a simple security mechanism so that an unattended terminal session can timeout before it becomes an open doorway to creative network reconfiguration.