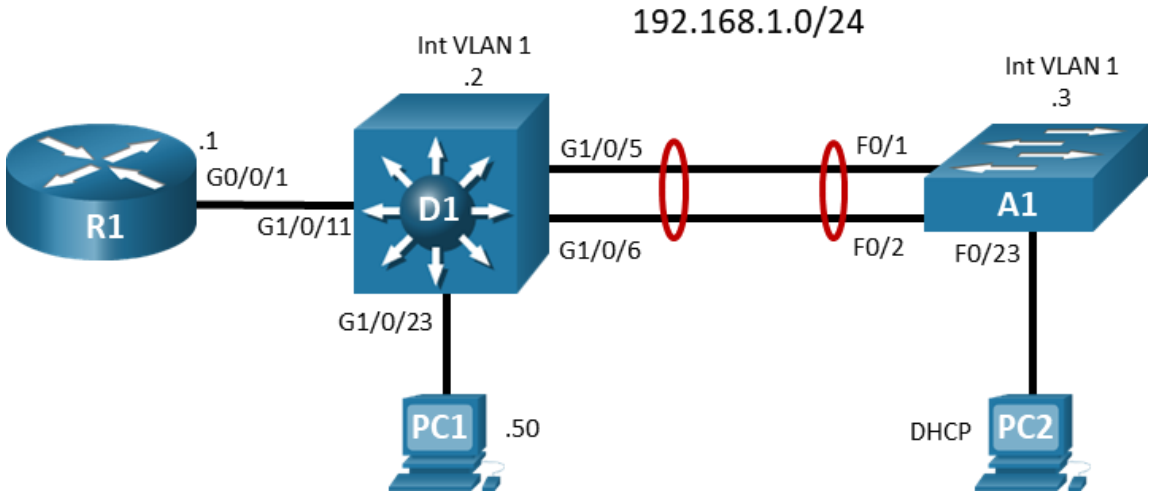
Lab -Configure Protections for Passwords and Terminal Lines

Topology



Addressing Table

Device	Interface	IP Address	Subnet Mask
R1	G0/0/1	192.168.1.1	255.255.255.0
D1	VLAN 1	192.168.1.2	255.255.255.0
A1	VLAN 1	192.168.1.3	255.255.255.0
PC 1	NIC	192.168.1.50	255.255.255.0
PC 2	NIC	DHCP	

Objectives

Part 1: Build the Network and Configure Basic Device Settings

Part 2: Explore Password Protection Options

Part 3: Configure and Verify Terminal Line Protection Options

Background / Scenario

Securing your network devices starts at the most basic level -- physical security. Physical security establishes restricted physical access to the devices. Most of the time you will connect to devices from a remote location. This makes securing remote access extremely important. In this lab, you will explore several different methods of protecting both local and remote access to your devices.

Note: This lab is an exercise in configuring options available for passwords and remote access protection and does not necessarily reflect network troubleshooting best practices.

Note: The routers used with CCNP hands-on labs are Cisco 4221 with Cisco IOS XE Release 16.9.4 (universalk9 image). The switches used in the labs are Cisco Catalyst 3650s with Cisco IOS XE Release 16.9.4 (universalk9 image) and Cisco Catalyst 2960s with Cisco IOS Release 15.2(2) (lanbasek9 image). Other routers, switches, and Cisco IOS versions can be used. Depending on the model and Cisco IOS version, the commands available and the output produced might vary from what is shown in the labs. Refer to the Router Interface Summary Table at the end of the lab for the correct interface identifiers.

Note: Make sure that the switches have been erased and have no startup configurations. If you are unsure, contact your instructor.

Required Resources

- 1 Router (Cisco 4221 with Cisco IOS XE Release 16.9.4 universal image or comparable)
- 1 Switch (Cisco 3650 with Cisco IOS XE Release 16.9.4 universal image or comparable)
- 1 Switch (Cisco 2960 with Cisco IOS Release 15.2(2) lanbasek9 image or comparable)
- 2 PCs (Operating system of choice with terminal emulation program installed)
- Console cables to configure the Cisco IOS devices via the console ports
- Ethernet cables as shown in the topology

Part 1: Build the Network and Configure Basic Device Settings

In Part 1, you will set up the network topology and configure basic settings and interface addressing on routers.

Step 1: Cable the network as shown in the topology.

Attach the devices as shown in the topology diagram, and cable as necessary.

Step 2: Configure basic settings for each device.

a. Console into each device, enter global configuration mode, and apply the basic settings. The startup configurations for each device are provided below.

Router R1

```
hostname R1
no ip domain lookup
banner motd # R1, Password and Terminal Protection #
line con 0
 exec-timeout 0 0
 logging synchronous
 exit
interface g0/0/1
 ip address 192.168.1.1 255.255.255.0
 no shutdown
 exit
ip dhcp excluded-address 192.168.1.1 192.168.1.5
ip dhcp pool HOST_ADDRESSING
 network 192.168.1.0 255.255.255.0
 default-router 192.168.1.1
 exit
end
```

Switch D1

```
hostname D1
no ip domain lookup
banner motd # D1, Password and Terminal Protection #
line con 0
 exec-timeout 0 0
 logging synchronous
 exit
interface vlan 1
 ip address 192.168.1.2 255.255.255.0
 no shutdown
 exit
ip default-gateway 192.168.1.1
interface g1/0/23
 spanning-tree portfast
 switchport mode access
 no shutdown
 exit
interface g1/0/11
 spanning-tree portfast
 switchport mode access
 no shutdown
 exit
interface range g1/0/5-6
 switchport mode trunk
 channel-group 1 mode active
 no shutdown
 exit
interface range g1/0/1-4, g1/0/7-10, g1/0/12-22, g1/0/24, g1/1/1-4
 shutdown
 exit
end
```

Switch A1

```
hostname A1
no ip domain lookup
banner motd # A1, Password and Terminal Protection #
line con 0
 exec-timeout 0 0
 logging synchronous
 exit
interface vlan 1
 ip address 192.168.1.3 255.255.255.0
 no shutdown
 exit
```

```
ip default-gateway 192.168.1.1
interface range f0/1-2
 switchport mode trunk
 channel-group 1 mode active
 no shutdown
 exit
interface range f0/3-24, g0/1-2
 shutdown
 exit
interface f0/23
 switchport mode access
 spanning-tree portfast
 no shutdown
 exit
end
```

b. Set the clock on each device to UTC time.

c. Save the running configuration to startup-config.

d. Verify that PC 2 receives an IP address via DHCP.

e. Verify that D1, A1, PC 1 and PC 2 can ping R1 interface G0/0/1.

Part 2: Explore Password Protection Options

Cisco device configuration files are simply text files. They are commonly backed up onto various media and stored in different, hopefully secure, places. Contained in the configuration, whether stored as a file or displayed on screen, are the various passwords used to access the device. An unauthorized individual having access to these files or output could use the information to gain access to the devices.

Note: There are many instances where passwords are used to protect protocol operations, but those are beyond the scope of this lab. This lab is focused on CLI access to the network device.

Step 1: Examine the password command.

a. The password command or parameter is used in several different ways. The most basic of these is configuring a password for the console line.

```
R1(config)# line con 0
R1(config-line)# password ?
  0     Specifies an UNENCRYPTED password will follow
  7     Specifies a HIDDEN password will follow
  LINE  The UNENCRYPTED (cleartext) line password
```

b. The options presented at this point might be confusing. The default is 0, which is to say “the string that follows is unencrypted”. You are not even required to use the zero; you just supply the password string in its unencrypted form. The 7 option is used when you are configuring an already-obfuscated password. In this example, we will use the plaintext string cisco123.

```
R1(config-line)# password cisco123
```

c. Now that the password is entered, add the **login** command, which configures the device to allow someone to log in if they supply the configured password. Without the **login** command, the password does not function.

```
R1(config-line)# login
```

d. Test the login requirement by exiting from the console and then **logout** of the router. Then attempt to reconnect to the router. You should be prompted for a password and denied if the password you supply is incorrect.

```
R1(config-line)# end
R1# logout

R1 con0 is now available

Press RETURN to get started.

 R1, Protect Passwords and Terminal Lines

User Access Verification

Password: <entered incorrect password and pressed return>
Password: <entered correct password, cisco123>
R1>
```

e. The problem with the **password** command is that the password string is stored in plaintext. Go into privileged EXEC mode and issue the command **show run | section line con** and examine the output. As you can see, the password string is available for all to see in plaintext. Excluding using the **secret** keyword, this default behavior applies to any system or user **password** command.

```
R1> enable
R1# show run | section line con
line con 0
 exec-timeout 0 0
 password cisco123
 logging synchronous
 login
 transport input none
 stopbits 1
```

Step 2: Secure passwords with type 7 encryption.

a. Type 7 encryption is a very weak encryption that is easily reversible. When configured, type 7 encryption is applied to all plaintext passwords, including system and user passwords that include the **password** command. Enable type 7 encryption for all plaintext passwords as shown.

```
R1# configure terminal
R1(config)# service password-encryption
R1(config)# exit
```

b. When the **service password-encryption** is enabled, all plaintext entries are a bit more complex.

```
R1# show run | include password
service password-encryption
 password 7 060506324F41584B56
```

However, do not let the apparent complexity mislead you. A string encrypted to type 7 is not considered secure. Search the internet for “Cisco Type 7 Password Cracker” and you will find many sites and even mobile apps that will reverse the encryption. Try one and you will see that the type 7 string is quickly and easily changed to plaintext.

Type 7 encryption is the most protection the local device can provide to a terminal line. To provide any more security, you must use a local login database, covered next, or use Authentication, Authorization and Accounting (AAA), which is covered in another lab.

Step 3: Secure passwords with type 5 encryption.

Type 5 encryption uses a 128-bit MD5 hash to secure the password. The device creates a hash out of the configuration entry, and then subsequent login attempts subject the supplied password string to the same hashing algorithm. If the hashes match, then the password string entered was the same as that stored on the device. Use the **secret** parameter to create a type 5 password. You should be familiar with the secret parameter from creating an enable secret password. In this case, our goal is to secure logins to the device, so we use the secret parameter as a part of the **username** command.

a. On all your devices, configure a local user named admin with the password string cisco123 using type 5 encryption.

```
R1(config)# username admin secret cisco123
```

b. Examine the output of **show run | include username** and you will see that the string is marked as a type 5 and looks very different from the type 7 passwords you saw earlier.

```
R1# show run | include username
username admin secret 5 $1$Ft4s$VepVuV73QfKNBlR4wMXff1
```

c. To use this enhanced password, issue the command **login local** in the console line configuration. This instructs the device to prompt for a username and password and compare the responses to the local database.

```
R1(config)# line con 0
R1(config-line)# login local
R1(config-line)# end
```

d. Now completely disconnect and attempt to log in again, you will see that you are now prompted for a username followed by a password.

```
R1, Protect Passwords and Terminal Lines


User Access Verification


Username: admin
Password: <enter correct password, cisco123>
R1>
```

Do not let the apparent complexity of a type 5 password mislead you. A string encrypted to type 5 is not considered secure. Search the internet for “Cisco Type 5 Password Cracker” and you will find many sites that will reverse the encryption. Try one and you will see that the type 5 string is quickly and easily changed to plaintext.

Step 4: Secure passwords with type 9 encryption.

With type 7 and type 5 passwords being so easy to circumvent, Cisco added support for type 8 and type 9 passwords. Type 8 passwords use Password-Based Key Derivation Function 2 (PBKDF2) with sha256 as the hashing function, while type 9 uses the SCRYPT hashing function, which also uses sha256, but in a different manner. Both types are considered uncrackable. SCRYPT is more computationally intense.

To use a type 9 password, add the **algorithm-type** parameter to your command before the secret parameter.

a. Create a privileged EXEC password using a type 9 password. Set the password string as **cisco123**.

```
R1(config)# enable algorithm-type scrypt secret cisco123
```

b. Reconfigure the admin account you configured earlier to use a type 9 password. Use the same password string, cisco123.

```
R1(config)# username admin algorithm-type scrypt secret cisco123
```

c. To see the results of these entries, issue the command **show run | include secret 9**. As you can see, the output strings are much longer than type 7 or type 5.

```
R1# show run | include secret 9
enable secret 9 $9$o.TVpUa5sYEeTU$WAA3GDD6u7GAK19Wcnh5hH325RLlG2H5EHA2ALY.GqU
username admin secret 9
$9$r5fycGQrMSV.7k$W49OJ3RnrybJjPLLpgKpqwaSja52GiKMYEQCdhwyxsg
```

Part 3: Configure and Verify Terminal Line Protection Options

Now that the passwords are secure, consider the protocols used to remotely access the management plane. The default remote access protocols are Telnet, which is unsecure, and Secure Shell (SSH). SSH is a protocol which provides a secure remote access connection to network devices. Communication between the client and server is encrypted in both SSH version 1 and 2. However, implement SSH version 2 when possible because it uses a more enhanced security encryption algorithm.

Step 1: Configure SSH support.

a. SSH uses crypto keys. Devices running IOS XE will generate self-signed certificates at startup. These certificates will not match our implementation, and so they need to be removed.

Note: Do not do this in a production environment without a clear understanding of the crypto systems being used. It may result in a device being unreachable and inaccessible.

Use the **crypto key zeroize** global configuration command to clear all existing keys.

```
R1(config)# crypto key zeroize
% All keys will be removed.
% All router certs issued using these keys will also be removed.
Do you really want to remove these keys? [yes/no]: yes
R1(config)#
*Feb 11 16:24:57.826: %CRYPTO_ENGINE-5-KEY_DELETED: A key named  has been
removed from key storage
*Feb 11 16:24:57.826: %CRYPTO_ENGINE-5-KEY_DELETED: A key named  has been
removed from key storage
*Feb 11 16:24:57.827: %SSH-5-DISABLED: SSH 1.99 has been disabled
```

b. To configure SSH on a Cisco device, at a minimum, you will need to configure a hostname, domain name, correct time, and a username database for authentication. In this lab, you have already configured a hostname, set the clock, and configured a user, admin. To complete the basic SSH configuration, you will configure a domain name for all the devices. Use the **ip domain name** command to set the domain name to CCNP.EIGHT. The IOS syntax has been updated from the **ip domain-name** to **ip domain name** command.

Note: The domain name used here is obviously made up. If your devices must be integrated into your organization's DNS infrastructure, then you must use your organization's domain name.

```
R1(config)# ip domain name CCNP.EIGHT
```

c. After meeting the prerequisites, generate a crypto key. Keys can be of varying sizes, but 2048 bits is considered a safe size. The command to do this will vary slightly between different devices and operating systems.

```
R1(config)# crypto key generate rsa modulus 2048
```

```
The name for the keys will be: R1.CCNP.EIGHT

% The key modulus size is 2048 bits
% Generating 2048 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 2 seconds)

R1(config)#
*Feb 11 16:27:28.354: %CRYPTO_ENGINE-5-KEY_ADDITION: A key named
R1.CCNP.EIGHT has been generated or imported by crypto-engine
*Feb 11 16:27:28.355: %SSH-5-ENABLED: SSH 1.99 has been enabled
*Feb 11 16:27:28.543: %CRYPTO_ENGINE-5-KEY_ADDITION: A key named
R1.CCNP.EIGHT.server has been generated or imported by crypto-engine
```

d. The console output shows that SSH 1.99 has been enabled, which means that the device supports connections using both SSH version 1 and SSH version 2. SSH version 1 has some flaws in implementation, so it should be avoided. To restrict the device support to just SSH version 2, issue the command **ip ssh version 2**.

```
R1(config)# ip ssh version 2
```

e. You should also restrict the period of time the authentication process is allowed to take and how many authentication attempts a user is given using the **ip ssh time-out** and **ip ssh authentication-retries** commands. The timeout value governs how long the device will wait for the authentication process (username and password) to complete. The default is 120 seconds. Restricting this time helps to ensure availability. Set it to 60 seconds or less. The authentication retry value, which defaults to 3, dictates how many attempts a user gets to put the correct password in before they are disconnected. Setting it to 2 is a common practice.

```
R1(config)# ip ssh time-out 30
R1(config)# ip ssh authentication-retries 2
```

f. Now configure the vty lines to access incoming SSH connections with the **transport input ssh** command. This command disables all support for Telnet on the vty lines.

```
R1(config)# line vty 0 4
R1(config-line)# transport input ssh
```

g. Lastly, configure the vty lines to query the local user database using the command **login local**.

```
R1(config-line)# login local
```

Test your SSH configuration by attempting to SSH from PC 1 and PC 2 to R1. Both attempts should be successful. Issue the **show ssh** command to view both established SSH sessions to R1.

```
R1# show ssh
Connection Version Mode  Encryption  Hmac          State               Username
0          2.0     IN    aes256-ctr  hmac-sha2-256 Session started       admin
0          2.0     OUT   aes256-ctr  hmac-sha2-256 Session started       admin
1          2.0     IN    aes256-ctr  hmac-sha2-256 Session started       admin
1          2.0     OUT   aes256-ctr  hmac-sha2-256 Session started       admin
```

Step 2: Add Timers to the vty lines.

With the default configuration, after a user is connected, the system will not disconnect the session until there has been 10 minutes without activity. This is an old default setting that needs to be changed to help keep your system secure.

a. Change the default inactivity timer to a shorter period using the **exec-timeout [minutes] {seconds}** command. Different organizations have different requirements for this value. For our purposes, set the exec-timeout on the vty lines to 3 minutes.

```
R1(config)# line vty 0 4
R1(config-line)# exec-timeout 3 0
```

b. On certain devices (the router only in our case) you can give the connected user a warning that they are about to be disconnected using the **logout-warning [seconds]** command. Configure a logout warning for 1 minute.

```
R1(config-line)# logout-warning 60
```

c. On certain devices (the router only in our case), you have the option to use an absolute timeout, which disconnects the user no matter if they are active or not. For our purposes, configure this value to be 15 minutes using the **absolute-timeout [minutes]** command.

```
R1(config-line)# absolute-timeout 15
```

d. Test these tweaks to your SSH configuration by attempting to SSH from PC 2 to one of the devices and delay your password response to see what happens. Try again and put in the wrong password until it disconnects you. Log in one more time and let the terminal line remain inactive until you get disconnected.

Step 3: Restrict Access to the vty lines at Layer 3.

Up to this point you have configured and secured access to the terminal lines by creating encrypted passwords, eliminating Telnet in favor of SSH version 2, and tuned SSH timers to help keep things secure. But none of this controls who can attempt to connect to your devices. To do this, configure and apply an access list.

a. Create a standard access list, either numbered or named, that permits only the host at 192.168.1.50 to access the vty lines.

```
R1(config)# ip access-list standard VTY-CONTROL
R1(config-std-nacl)# permit host 192.168.1.50
R1(config-std-nacl)# deny any log
R1(config-std-nacl)# exit
R1(config)#
```

b. Apply the access list to the vty lines using the **access-class [name | number] in** command.

```
R1(config)# line vty 0 4
R1(config-line)# access-class VTY-CONTROL in
R1(config-line)# exit
```

c. Test your ACL by attempting to connect to R1 from PC 2. The connection attempt should fail, while connection attempts from PC 1 should succeed.

d. As a challenge, configure D1 and A1 with secure passwords and SSH support for the terminal lines with restricted access.

Reflection Questions

1. What additional security can be configured on a Cisco device when implementing SSH?

2. How can you implement type 9 passwords using scrypt and avoid using the console and vty line passwords with type 7 encryption?

Router Interface Summary Table

Router Model	Ethernet Interface #1	Ethernet Interface #2	Serial Interface #1	Serial Interface #2
1800	Fast Ethernet 0/0 (F0/0)	Fast Ethernet 0/1 (F0/1)	Serial 0/0/0 (S0/0/0)	Serial 0/0/1 (S0/0/1)
1900	Gigabit Ethernet 0/0 (G0/0)	Gigabit Ethernet 0/1 (G0/1)	Serial 0/0/0 (S0/0/0)	Serial 0/0/1 (S0/0/1)
2801	Fast Ethernet 0/0 (F0/0)	Fast Ethernet 0/1 (F0/1)	Serial 0/1/0 (S0/1/0)	Serial 0/1/1 (S0/1/1)
2811	Fast Ethernet 0/0 (F0/0)	Fast Ethernet 0/1 (F0/1)	Serial 0/0/0 (S0/0/0)	Serial 0/0/1 (S0/0/1)
2900	Gigabit Ethernet 0/0 (G0/0)	Gigabit Ethernet 0/1 (G0/1)	Serial 0/0/0 (S0/0/0)	Serial 0/0/1 (S0/0/1)
4221	Gigabit Ethernet 0/0/0 (G0/0/0)	Gigabit Ethernet 0/0/1 (G0/0/1)	Serial 0/1/0 (S0/1/0)	Serial 0/1/1 (S0/1/1)
4300	Gigabit Ethernet 0/0/0 (G0/0/0)	Gigabit Ethernet 0/0/1 (G0/0/1)	Serial 0/1/0 (S0/1/0)	Serial 0/1/1 (S0/1/1)

Note: To find out how the router is configured, look at the interfaces to identify the type of router and how many interfaces the router has. There is no way to effectively list all the combinations of configurations for each router class. This table includes identifiers for the possible combinations of Ethernet and Serial interfaces in the device. The table does not include any other type of interface, even though a specific router may contain one. An example of this might be an ISDN BRI interface. The string in parenthesis is the legal abbreviation that can be used in Cisco IOS commands to represent the interface.