34. Device Security

ccnacookbook.com

PASSWORDS

Three kinds of passwords are normally stored in plaintext in the running config and startup config:

PURPOSE	SYNTAX
Password on console or vty (telnet/ssh) lines	`R5(config-line)# password cisco`
Enable password for privileged exec mode	`R5(config)# enable password class`
User login account	`R5(config)# username fred password flintstone`

To optionally encrypt these passwords within the config so a human can't simply read them,

```
R5(config)# service password-encryption
```

In addition to encrypting the passwords, IOS adds a flag, "7," to indicate that the passwords are encrypted ("0" or a missing flag means they're in plaintext). Changing back to "no service password encryption" won't actually decrypt your passwords, but you can reenter them if you really want plaintext passwords in your "show running-config."

BEFORE	AFTER

`no service password-encryption`
`enable password class`
`username fred password 0 flintstone`
`line con 0`
`password cisco`
	`service password-encryption`
	`enable password 7 01100A054818`
	`username fred password 7 045D070F01355F5A061700`
	`line con 0`
	`password 7 094F471A1A0A`

Unfortunately, these passwords *can* be easily decrypted by someone who has access to the encrypted version, so Cisco added "secret" passwords, discussed next.

! Passwords, as discussed above, are of historical and test-taking interest only. They are not secure and should never actually be used. Use secret passwords, below, instead.

SECRET PASSWORDS

Hash—a mathematical transformation of a value that will always give the same result for the same input, but cannot be decrypted. You can think of it as a one-way or "trap-door" encryption. A good hash function will also be highly unlikely to give the same result for two different inputs. A hash is sometimes referred to as the "fingerprint" of its input. The impossibility of decrypting a hash is sometimes referred to as "computationally difficult."

Secret passwords in IOS store only the hash of a password. When a user enters a password, that attempt is hashed and the two hashes are compared. The stored (correct) password isn't, doesn't need to be, and can't be decrypted. Note: you still need good passwords, since a hacker can quite simply generate a table of hashes for all the predictable passwords. In fact, a good hacker would recognize the hashes for "cisco" and "class" by sight. IOS flags secret passwords as type "5."

Secret passwords are drop-in replacements for enable and user accounts.

```
R5(config)# enable secret mySecret
R5(config)# username barney secret rubble
```

If you have both a "password" and a "secret" configured, IOS will use the secret and ignore the normal password (it'll be considered wrong). Modern versions of IOS won't even allow you to enter the same password as "secret" that you already have configured as "password." On username commands, the non-secret password will actually be replaced, not just ignored.

Telnet / SSH access isn't allowed until a password is configured on the vty lines, which by default there isn't.

By default, Cisco happens to use the MD5 (Message Digest 5) hash, which is also considered insecure. Newer versions of IOS give you two other hash choices when you configure the enable secret password. This whole mess is summarized in the table below.

COMMAND	TAG	ALGORITHM
`enable password cisco`	0	(NONE)
`enable password cisco + service password-encryption`	7	

`enable secret cisco`
`enable algorithm-type md5 secret cisco`
`username barney algorithm-type md5 secret rubble`
	5	MD-5

Disavowed; do not use. Worse than MD5. Removed from later versions of IOS.	4	SHA-256
`enable algorithm-type sha-256 secret cisco` `username barney algorithm-type sha-256 secret rubble`	8	SHA-256
`enable algorithm-type scrypt secret cisco` `username barney algorithm-type scrypt secret rubble`	9	SHA-256

The difference between 8 and 9 is that type 9 (scrypt) is designed to be either slow or a memory hog depending on the implementation, preventing quick and easy brute force attacks.

BONUS — HACKING TYPE 7 ENCRYPTION

To see a type-7 encrypted password in plain-text, simply add it to a key-chain in its type-7 form and then "show" the key chain. IOS happily decrypts the password for your convenience. There are also plenty of offline, even web-based, tools to decrypt type-7 passwords.

```
R6(config)# do sho run | inclu passwor
service password-encryption
enable password 7 0822404F1A0A
username fred password 7 01150A0D551F151B2E424B

R6(config)# key chain foo
R6(config-keychain)# key 1
R6(config-keychain-key)# key-string 7 0822404F1A0A
R6(config-keychain-key)# key 2
R6(config-keychain-key)# key-string 7 01150A0D551F151B2E424B
R6# show key chain
Key-chain foo:
    key 1 -- text "class"
        accept lifetime (always valid) - (always valid) [valid now]
        send lifetime (always valid) - (always valid) [valid now]
    key 2 -- text "flintstone"
        accept lifetime (always valid) - (always valid) [valid now]
        send lifetime (always valid) - (always valid) [valid now]
```

BANNERS

Three banners are provided, allowing you to fine-tune what message is displayed before and after login and over console and telnet versus SSH.

Banner	Before / After Login	Purpose
Message Of The Day (MOTD)	Varies	Temporary informational messages (scheduled downtime, etc.)
Login	Before	Pre-login security warnings (Authorized Use Only)
Exec	After	Anything admins need to know that outsiders shouldn't, like the location of the device

Order—the login message always precedes a request for a password and the exec banner always follows login. The difference is the message of the day. Because telnet and console connections are considered somewhat trusted, they display the MOTD first, even before the login prompt. On SSH connections, the MOTD is displayed immediately following a successful password.

Connection	Banner Order			
Console / Telnet	MOTD	Login Banner	(user login)	Exec Banner
SSH	Login Banner	(user login)	MOTD	Exec Banner

Configuration—the default banner type is "motd." The next character following the (optional) banner type is a delimiter that tells IOS when you're done typing a multi-line banner. You get to choose the delimiter to be some character that's *not* part of the message.

```
R5(config)# banner motd @
Enter TEXT message.  End with the character '@'.
################################################
  The internet will be shut down for cleaning
  Friday, April 1. Please complete your thumb-
  drive training and safety check before then.
################################################@

R5(config)#banner login NO TRESPASSING
                    Here the first character after the name of the banner is an "N"
                    The resulting banner is "O TRESPASSI"
R5(config)# banner exec #You are now logged in.#
R5(config)# banner !Please don't post joke MOTDs!
                    Because the MOTD is the default banner, this will replace the MOTD above.
```

SECURING SWITCHPORTS

Cisco switches work out of the box, so best practice is to tighten the security on any unused ports, especially if those ports lead to wall jacks somewhere in the building. For unused ports:

```
S5(config)# interface range fa0/1 -24
S5(config-if-range)# shutdown
S5(config-if-range)# switchport mode access
S5(config-if-range)# switchport access vlan 99
                    Vlan 99 is an unused "dead-end" vlan
S5(config-if-range)# switchport trunk native vlan 99
```

VTY LINE ACLS

An ACL can be applied on a VTY line using the directive "access-class" instead of "access-group" (used with normal interfaces). To keep them straight, I just remember that in the labs for a Cisco "class," we do a lot of telnetting.

Standard ACLs look at the source address of a packet, so when applied to a VTY in the inbound direction, the ACL will control which devices an administrator is allowed to log in from—very normal. Interestingly, when applied outbound, a standard ACL looks at the destination address and can restrict where someone on the router can telnet *to*.

```
line vty 0 15
 access-class AdminSSH in
```

FIREWALLS

ASA (Cisco Adaptive Security Appliance)—a separate firewall device.

Like a router ACL, a firewall can

- Filter based on source & destination IP address
- Filter based on source & destination TCP & UDP ports (corresponding to applications)

A firewall can also

- Inspect HTTP URIs (sometimes called URLs)
- Intelligently track a flow of packets, rather than filtering one packet at a time
- Be stateful—make future decisions based on past events

The ability to be stateful is the important difference. For example, one packet is traffic, one million is a denial of service attack.

Zones—a configuration that allows you to control *which* devices can initiate conversations to which others, based on the zones each is in. For example, it's fine for web browsers in the company to initiate conversations and receive replies from servers outside the company. It's a huge problem if outsiders are initiating conversations inbound.

DMZ (DeMilitarized Zone)—If your company has a web server that should receive non-reply traffic from outside, that would be in a separate zone which allows such things.