Boson®

ROUTE Curriculum

300-101

Labs powered by

NetSim®
NETWORK SIMULATOR®

Boson

ROUTE

300-101 Curriculum

LM20170830/BV1.0

Boson® NetSim® NETWORK SIMULATOR®

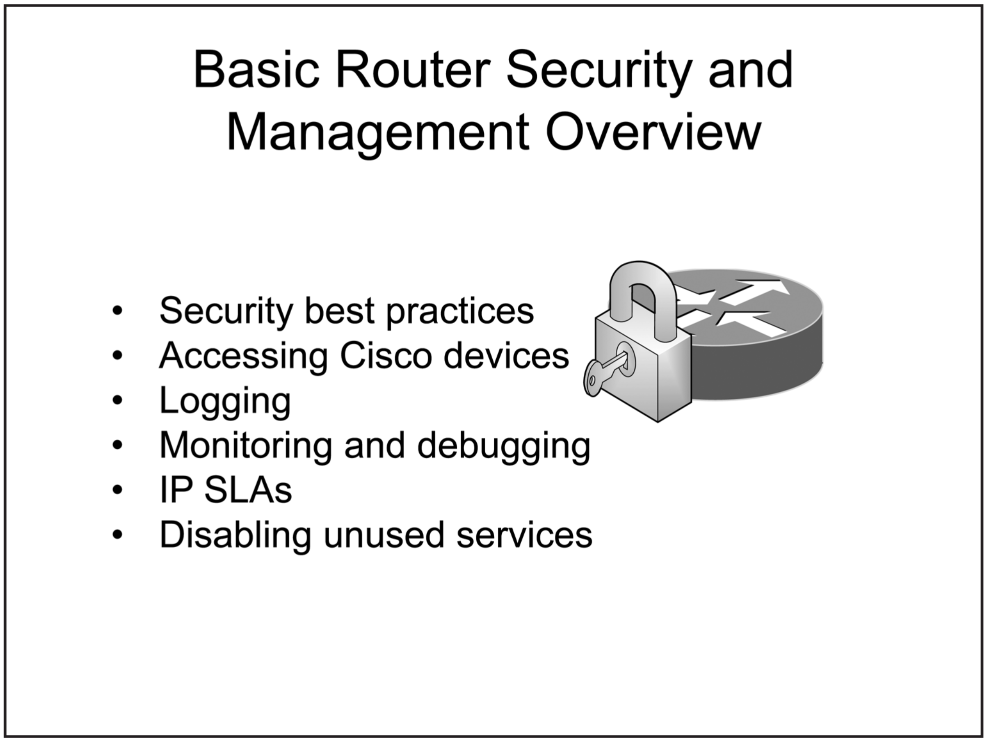
25 Century Blvd., Ste. 500, Nashville, TN 37214 | Boson.com

The labs referenced in this book have been printed in the Boson Lab Guide, which is included with the purchase of the curriculum. These labs can be performed with real Cisco hardware or in the Boson NetSim Network Simulator version 11 or later. To learn more about the benefits of using NetSim or to purchase the software, please visit www.boson.com/netsim.

Content in these modules is available in the full version of the curriculum. Please visit www.boson.com for more information.

Module 1

Basic Router Security and Management



Basic Router Security and Management Overview

- Security best practices
- Accessing Cisco devices
- Logging
- Monitoring and debugging
- IP SLAs
- Disabling unused services

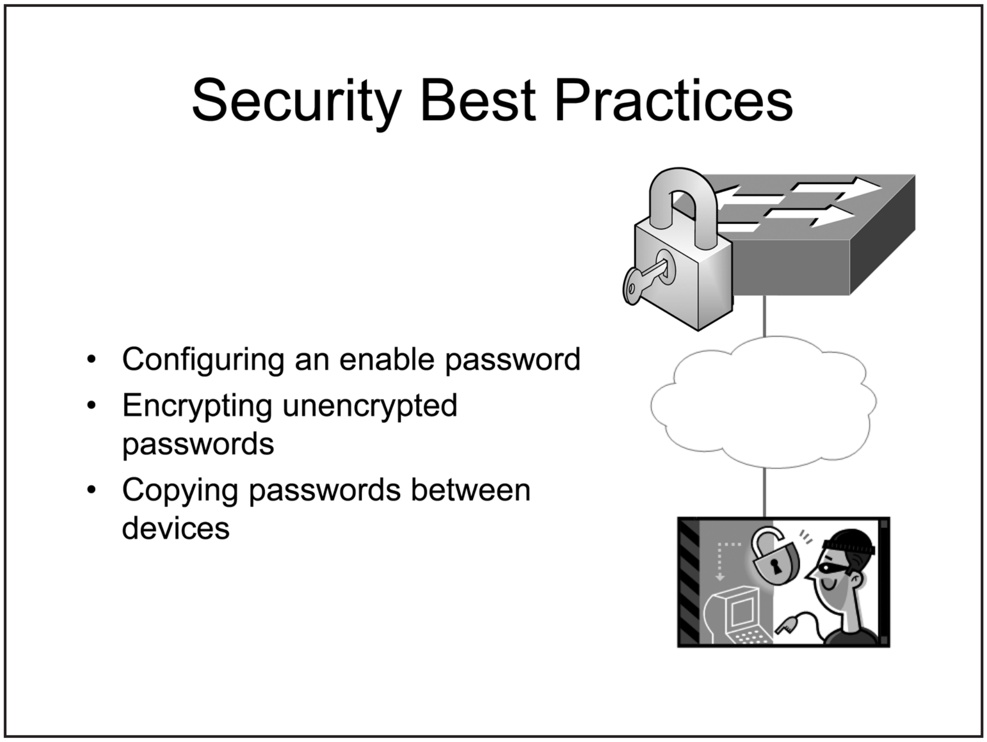
Overview

Network systems face a number of threats from both internal and external sources. This module will explore some of the methods that administrators can use to secure routers. You will also be exposed to several network management and monitoring tools that you can use to detect network problems before they become serious.

Objectives

After completing this module, you should have the basic knowledge required to complete all of the following tasks:

- Learn about security best practices, including how to configure passwords.
- Understand how to access and require authentication for Cisco devices.
- Understand various logging methods that are available for Cisco devices.
- Understand, configure, and verify Network Time Protocol (NTP).
- Understand and configure logging.
- Understand the capabilities and drawbacks of **debug** commands.
- Understand and configure Simple Network Management Protocol (SNMP) versions 1, 2c, and 3.
- Understand, configure, and use NetFlow.
- Understand and configure IP Service Level Agreements (SLAs).
- Learn how to disable or replace unused services.



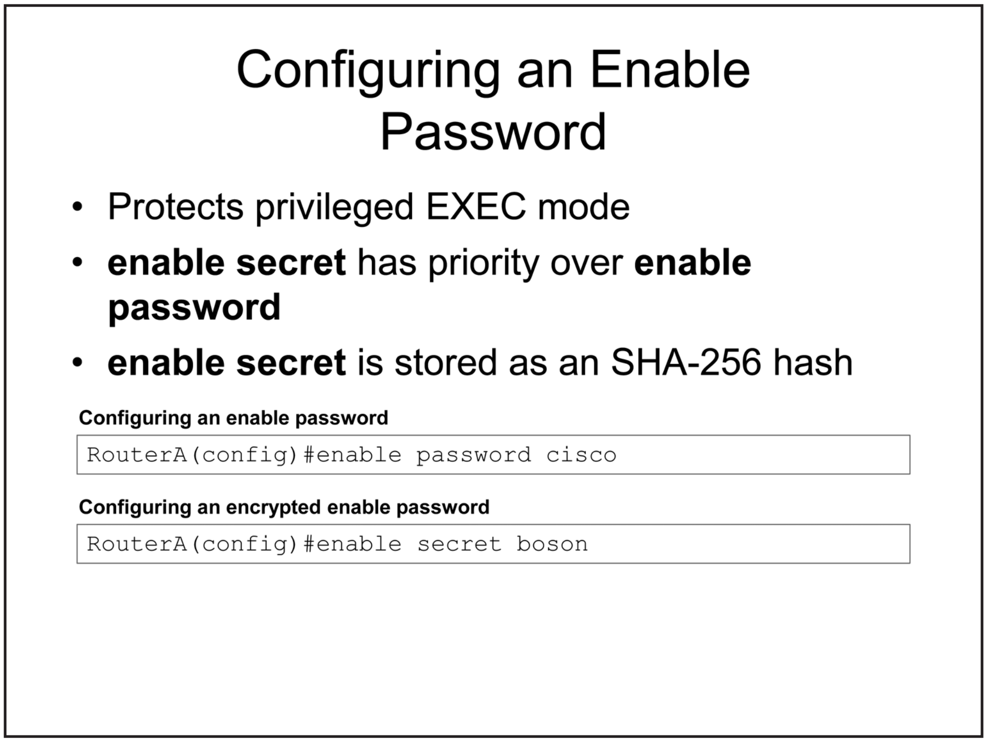
Security Best Practices

- Configuring an enable password
- Encrypting unencrypted passwords
- Copying passwords between devices

Security Best Practices

Security best practices, at their core, involve limiting access to the router or the network to which it is connected. There are several mechanisms that can be used to secure administrative access to a Cisco device. At the simplest level, the device should require authentication for access to the console or any of its terminal lines. Ideally, the device should accept only encrypted connections so that passwords and other configuration information are not transmitted in clear text. If possible, user names and passwords should be assigned to allow only specific users access to the device.

This section covers how to configure and encrypt an enable password as well as how to copy passwords between devices.



Configuring an Enable Password

- Protects privileged EXEC mode
- **enable secret** has priority over **enable password**
- **enable secret** is stored as an SHA-256 hash

Configuring an enable password

```
RouterA(config)#enable password cisco
```

Configuring an encrypted enable password

```
RouterA(config)#enable secret boson
```

Configuring an Enable Password

By default, privileged EXEC mode on a Cisco device provides unrestricted access to every available IOS command. The simplest way to prevent an authorized user from accessing these commands is to assign an enable password. If an enable password is configured, the device will issue a password prompt when a user attempts to access privileged EXEC mode. You can issue the **enable password** *password* command to configure an enable password. By default, the **enable password** command stores an unencrypted password in the device's configuration file, as shown in the following sample output from the **show running-config** command:

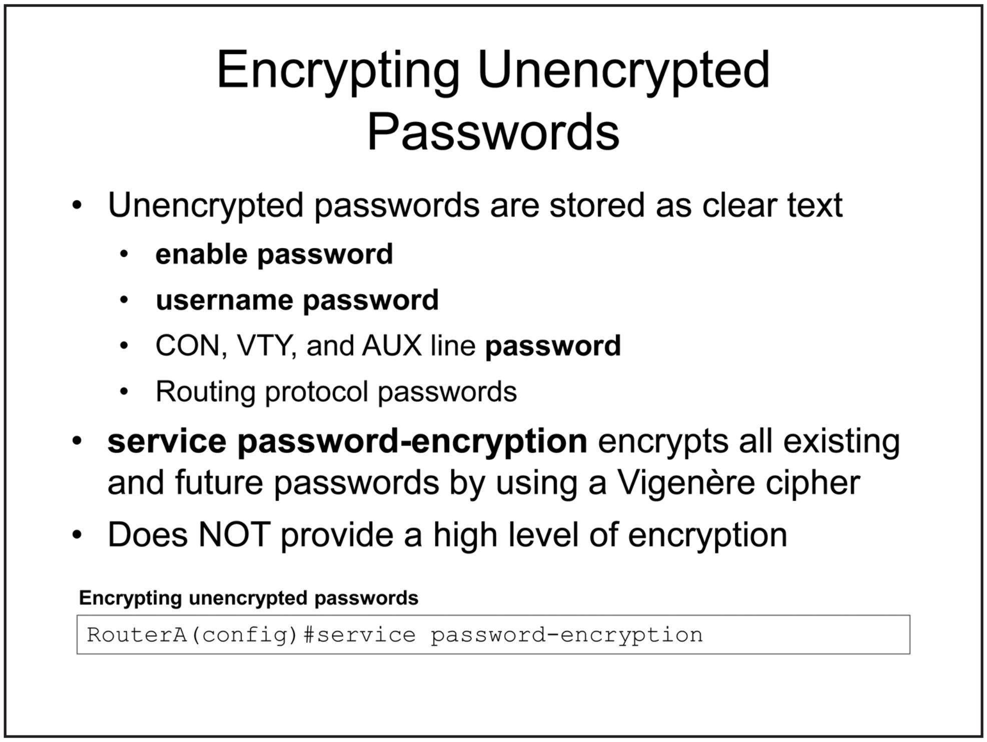
```
enable password cisco
```

By contrast, the **enable secret** command stores an encrypted password in the device's configuration file using an Secure Hash Algorithm (SHA)-256 hash. The syntax for the **enable secret** command is **enable secret** [**level** *level*] {*password* | [*encryption-type*] *encrypted-password*}, where *password* is a string of characters. The optional **level** parameter specifies the privilege level for which the encrypted password should be used. If both the **enable password** and **enable secret** commands are in the running configuration of a Cisco device, the device will ignore the password associated with the **enable password** command. The following sample output from the **show running-config** command shows both a password that has been encrypted by the **enable secret** command and an unencrypted password from the **enable password** command:

```
enable secret 5 $1$7nZu$J8bk/JkdJ8rPJUGKNk3im/
enable password cisco
```

Many password commands in the running configuration use an integer to indicate the type of encryption that was used to protect the password. For example, a 0 indicates that the password is unencrypted, a 5 indicates the password is a Message Digest 5 (MD5) hash, and a 7 indicates that the password was encrypted using Cisco's original password algorithm. The following sample output from the **show running-config** command shows each of these types of passwords:

```
enable secret 5 $1$7nZu$J8bk/JkdJ8rPJUGKNk3im/
enable password 7 130019130900013A2A373B243A3017
username tester password 0 testerpassword
```



Encrypting Unencrypted Passwords

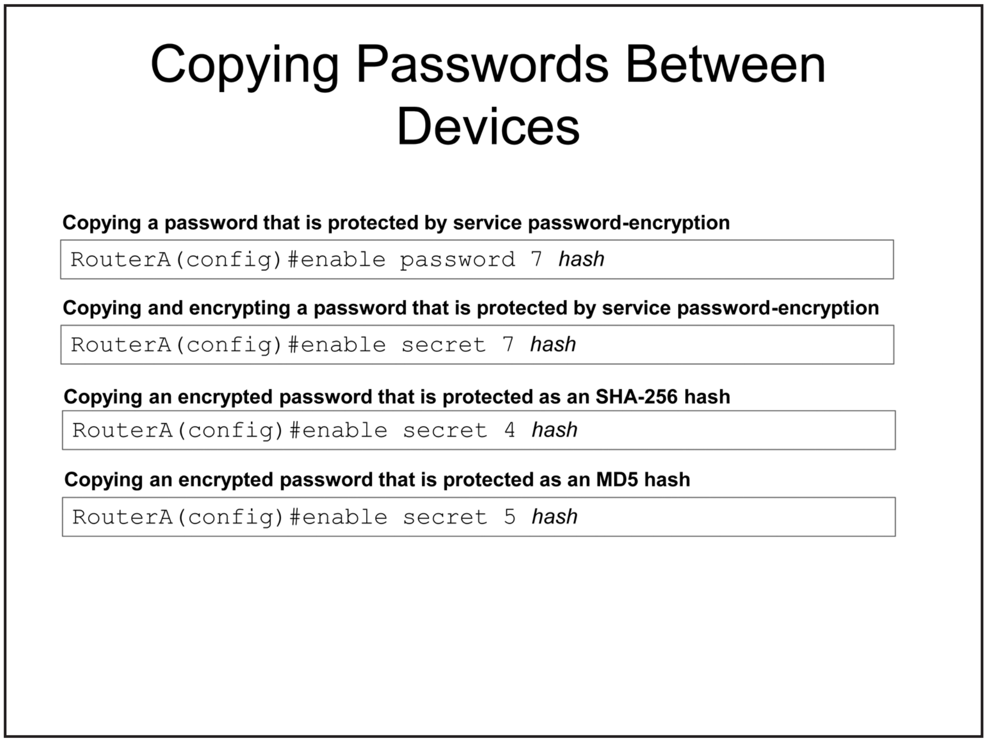
- Unencrypted passwords are stored as clear text
  - **enable password**
  - **username password**
  - CON, VTY, and AUX line **password**
  - Routing protocol passwords
- **service password-encryption** encrypts all existing and future passwords by using a Vigenère cipher
- Does NOT provide a high level of encryption

Encrypting unencrypted passwords

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

Encrypting Unencrypted Passwords

Except for the password you create by issuing the **enable secret** command, all Cisco passwords are stored as clear text by default, including the **enable** password, **username** passwords, line passwords, and routing protocol passwords. When the **service password-encryption** command is issued, existing and future enable, console, and virtual terminal (VTY) passwords are encrypted by using a Vigenère cipher. This cipher does not provide a high level of encryption, but it is better than storing clear-text passwords. Issuing the **no service password-encryption** command causes future passwords to be stored as clear text but will not decrypt existing passwords.



Copying Passwords Between Devices

Copying a password that is protected by service password-encryption

```
RouterA(config)#enable password 7 hash
```

Copying and encrypting a password that is protected by service password-encryption

```
RouterA(config)#enable secret 7 hash
```

Copying an encrypted password that is protected as an SHA-256 hash

```
RouterA(config)#enable secret 4 hash
```

Copying an encrypted password that is protected as an MD5 hash

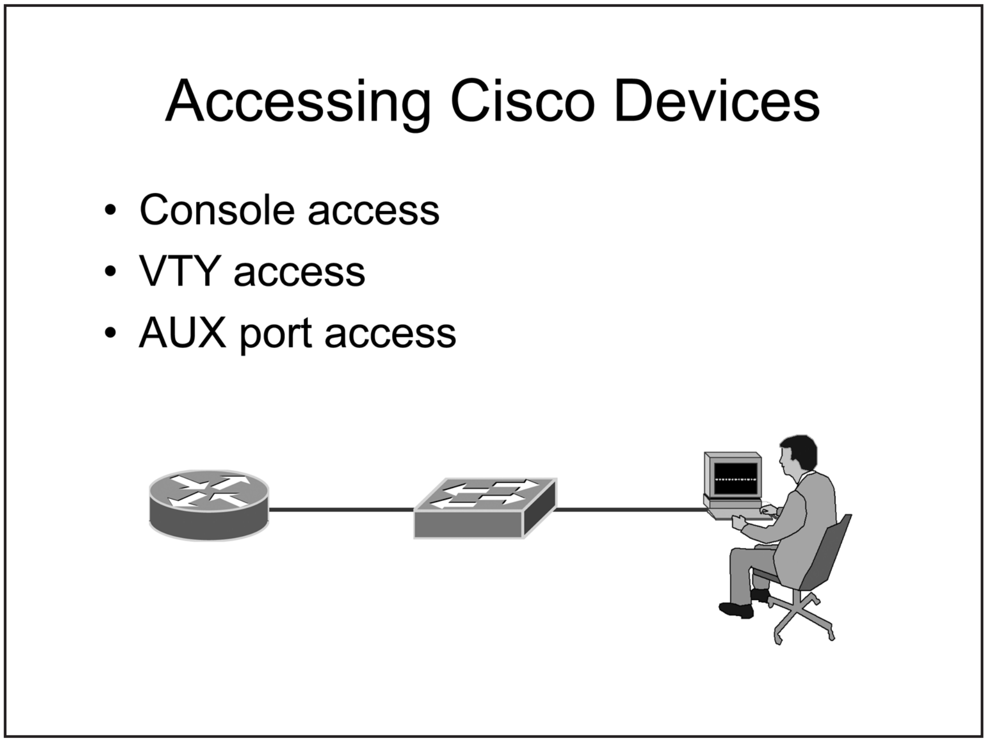
```
RouterA(config)#enable secret 5 hash
```

Copying Passwords Between Devices

You can copy encrypted passwords between devices even if you do not know the clear-text translation of the password simply by copying the hash. For example, consider the following router output:

```
enable password 7 130019130900013A2A373B243A3017
```

This password has been encrypted by the **service password-encryption** command. If you issue the **enable password 7 130019130900013A2A373B243A3017** command on another router, the enable password will be the same on both routers. This method also works for passwords that have been hashed by SHA-256 or MD5 by indicating the appropriate encryption type in the command.

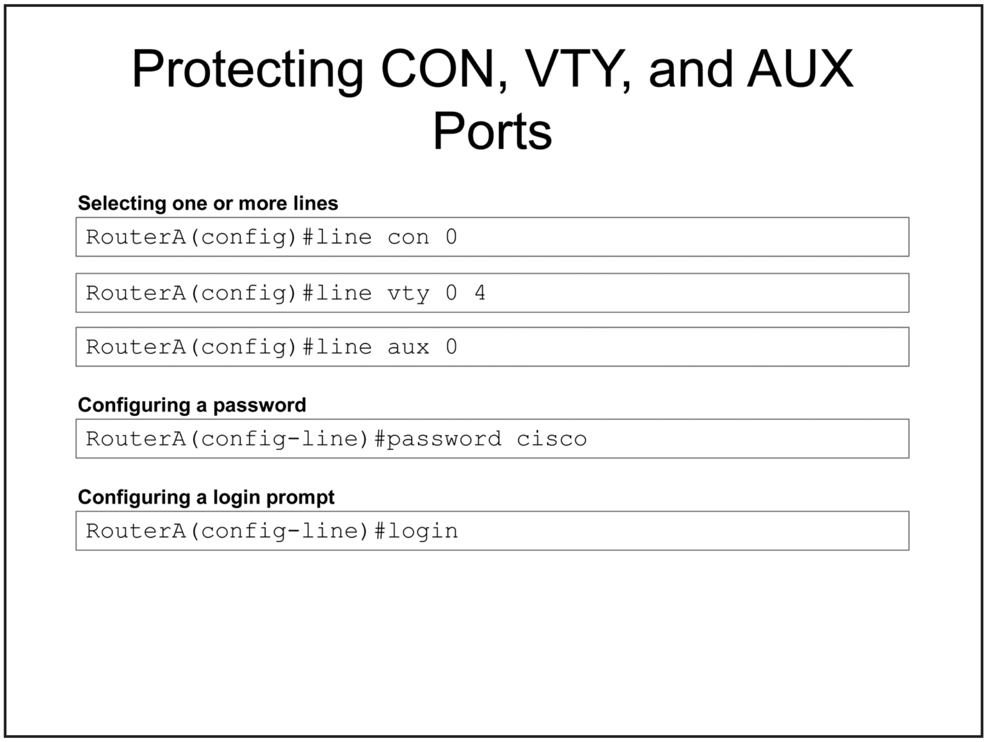


Accessing Cisco Devices

- Console access
- VTY access
- AUX port access

Accessing Cisco Devices

You can add another layer of authentication by requiring a password for console (CON) access, VTY access, or auxiliary (AUX) port access. If you configure an **enable password** and a line password, a user would be required to issue two passwords to make configuration changes to a device. For example, an administrator who is accessing the device by using Telnet might need to first issue the VTY password to connect to the device and then issue the **enable password** command to make configuration changes. The **enable password** should never be the same as the line password.



Protecting CON, VTY, and AUX Ports

Selecting one or more lines

```
RouterA(config)#line con 0
```

```
RouterA(config)#line vty 0 4
```

```
RouterA(config)#line aux 0
```

Configuring a password

```
RouterA(config-line)#password cisco
```

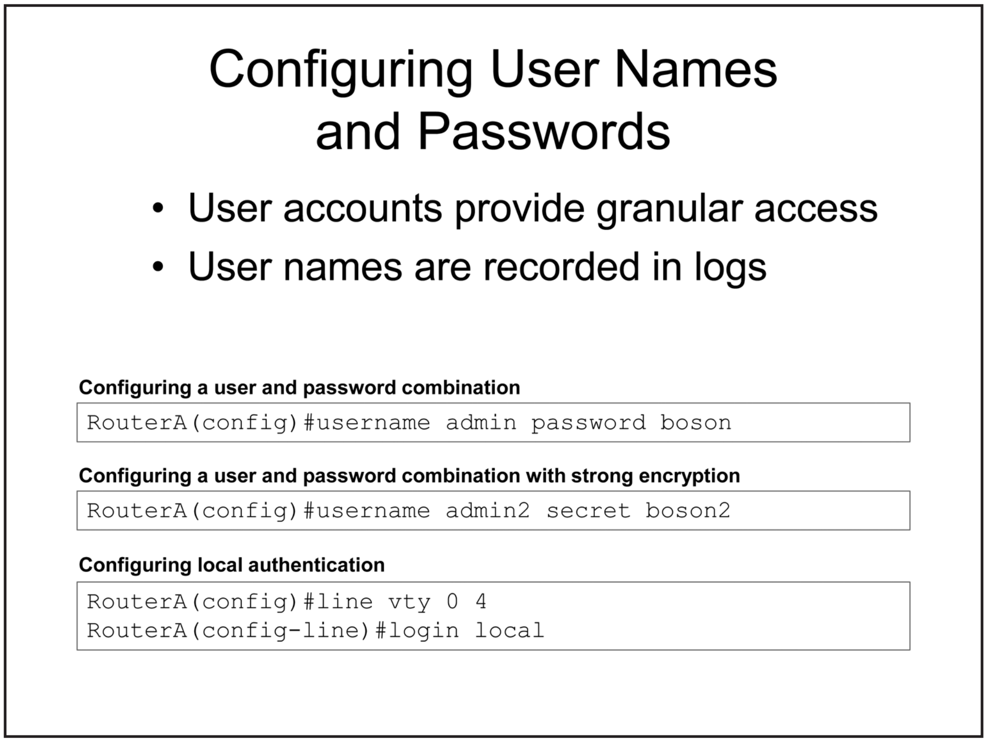
Configuring a login prompt

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

Protecting CON, VTY, and AUX Ports

You can issue the **password** command from line configuration mode to specify a password for the CON port, one or more VTY lines, or the AUX line. Once a password has been configured, you can then issue the **login** command to specify that a user must authenticate before access to the device is granted. If the **login** command is issued and a password has not been configured, the device will be inaccessible through the point of access until a password has been configured.

Line passwords, like most passwords configured on Cisco devices, are not encrypted by default. Because the passwords are stored in an unencrypted state, anyone who can access the configuration files will have access to the stored passwords. This is of even greater concern if the configuration files are stored remotely, such as on a Trivial File Transfer Protocol (TFTP) server.



Configuring User Names and Passwords

- User accounts provide granular access
- User names are recorded in logs

Configuring a user and password combination

```
RouterA(config)#username admin password boson
```

Configuring a user and password combination with strong encryption

```
RouterA(config)#username admin2 secret boson2
```

Configuring local authentication

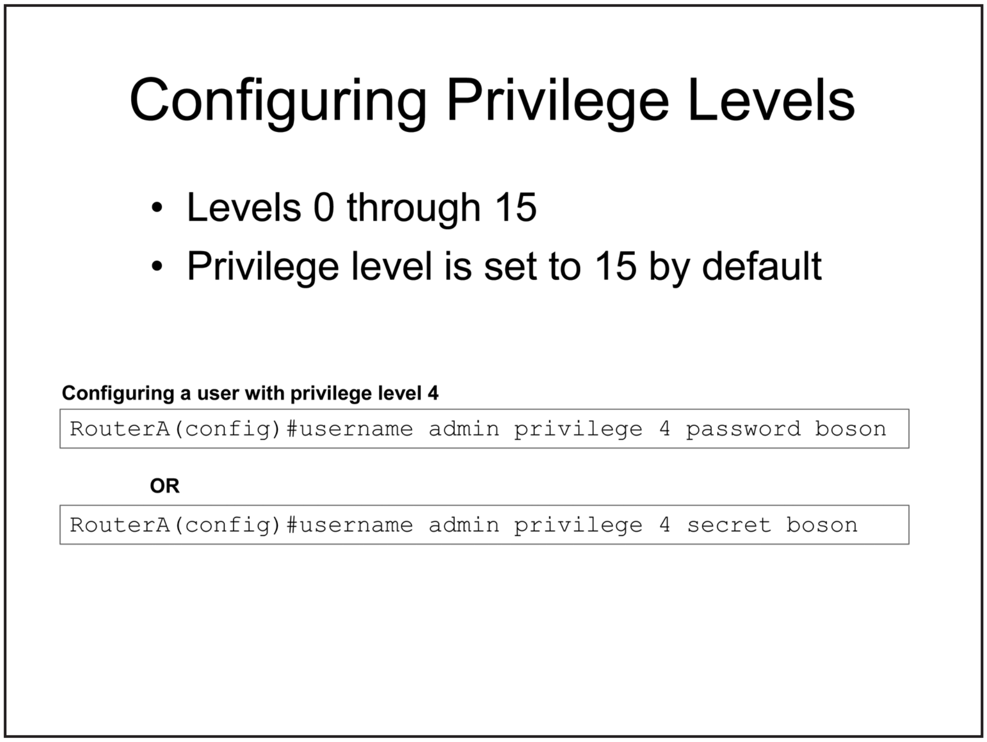
```
RouterA(config)#line vty 0 4
RouterA(config-line)#login local
```

Configuring User Names and Passwords

You can configure a Cisco device to require both a user name and a password for authentication. User accounts provide an additional level of granularity for both logging and access control. The account information can be stored locally on the device or remotely, such as on a Remote Authentication Dial-In User Service (RADIUS) server. You should issue the **username** command to specify the user name for a particular account. The syntax of the **username** command is **username** *user-name* **password** *password*, where *user-name* is the account name and password is the associated password. Alternatively, you can issue the **username** command with the **secret** keyword instead of the **password** keyword. The **secret** keyword specifies that the password string should be encrypted using strong encryption. If the **password** keyword is used and the password encryption service is not running, the password string will be stored as plain text. The following sample output from the **show running-config** command indicates a user name of *tester* with an unencrypted password and a user name of *secrettester* with a password that has been encrypted by the MD5 algorithm:

```
username tester password 0 testerpassword
username secrettester secret 5 $1$oaOa$9Gtp.JjvzsfesgRUq5zJW.
```

You can issue the **login local** command from line configuration mode to configure a Cisco device to use the local user database for authentication on a particular terminal line, or on the CON or AUX ports.



Configuring Privilege Levels

- Levels 0 through 15
- Privilege level is set to 15 by default

Configuring a user with privilege level 4

```
RouterA(config)#username admin privilege 4 password boson
```

OR

```
RouterA(config)#username admin privilege 4 secret boson
```

Configuring Privilege Levels

You can configure user accounts with privilege levels, which work the same way as **enable password** privilege levels do. You can create multiple privilege levels, from level 0 through level 15, to more granularly specify the commands that users can issue. Privilege level 15 is granted to a user if the privilege level has not been explicitly configured, which indicates that all commands are available to the user.

Privilege level 1 grants access similar to what you have before you issue the **enable** command. All lines (CON, AUX, and VTY) default to privilege level 1.

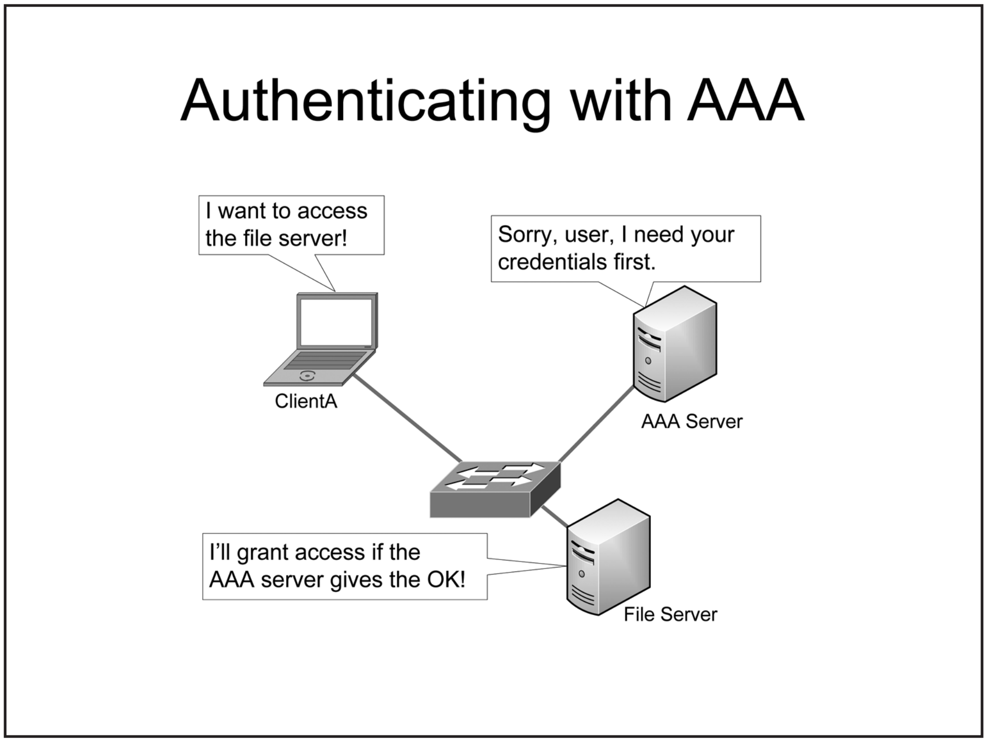
Privilege level 0 grants access to only the following commands:

- **disable**
- **enable**
- **exit**
- **help**
- **logout**

By default, all of the commands on a Cisco router are configured for privilege level 0, privilege level 1, or privilege level 15. A user can access any command at the user's privilege level and below.

You can change the privilege level for individual commands to granularly provide access or to increase security. However, the five commands at level 0 cannot be changed. The National Security Agency (NSA) recommends moving the following commands from level 1 to level 15:

- **connect**
- **telnet**
- **rlogin**
- **show ip access-lists**
- **show logging**



Authenticating with AAA

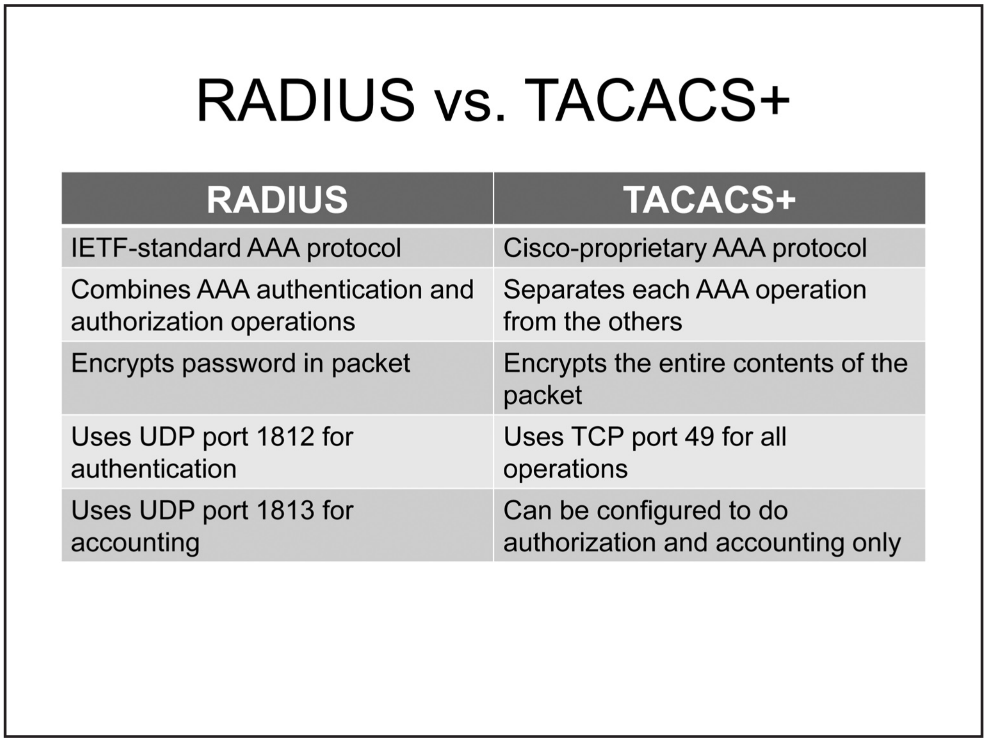
I want to access the file server!
Sorry, user, I need your credentials first.
ClientA
AAA Server
I'll grant access if the AAA server gives the OK!
File Server

Authenticating with AAA

Access controls use varying methods of Authentication, Authorization, and Accounting (AAA) to verify the identity of a user, prevent unauthorized access to sensitive data, and record user activity on a system. External AAA servers, such as a RADIUS server or a Terminal Access Controller Access-Control System Plus (TACACS+) server, take the burden of authentication off local devices by centralizing identity-based authentication for an entire network regardless of the network location of the device requesting the network service or the device hosting the service.

The following list defines the three phases of the AAA process:

- Authentication – the process of verifying a user's identity
- Authorization – the process of verifying the level of access configured for a user
- Accounting – the process of recording the use of resources



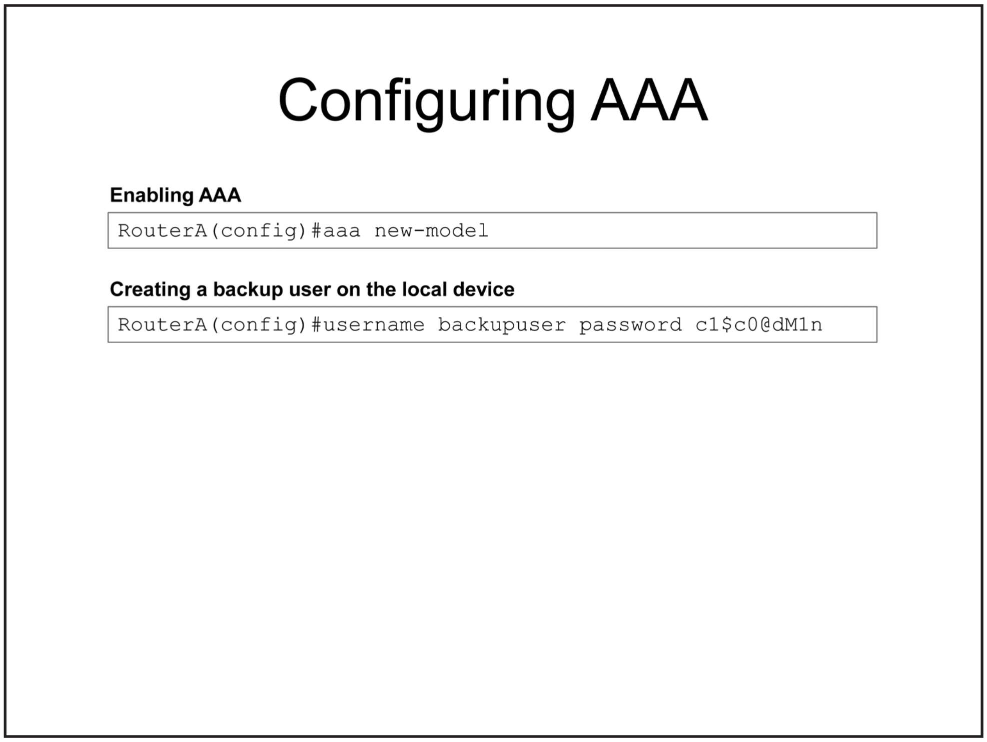
RADIUS vs. TACACS+

RADIUS	TACACS+
IETF-standard AAA protocol	Cisco-proprietary AAA protocol
Combines AAA authentication and authorization operations	Separates each AAA operation from the others
Encrypts password in packet	Encrypts the entire contents of the packet
Uses UDP port 1812 for authentication	Uses TCP port 49 for all operations
Uses UDP port 1813 for accounting	Can be configured to do authorization and accounting only

RADIUS vs. TACACS+

RADIUS is a standard AAA protocol created by the Internet Engineering Task Force (IETF). Compared to TACACS+, RADIUS has several limitations. For example, RADIUS encrypts only the password in Access-Request packets; it does not encrypt the entire contents of the packet like TACACS+ does. RADIUS, not TACACS+, uses User Datagram Protocol (UDP) for packet delivery. UDP port 1812 is used for authentication and authorization. UDP port 1813 is used for accounting. Some older RADIUS servers might use ports 1645 and 1646 instead. RADIUS is often used as the transport protocol for Extensible Authentication Protocol (EAP) devices, such as 802.1X-enabled wireless networks, because RADIUS is capable of encapsulating EAP.

TACACS+ is a Cisco-proprietary protocol used during AAA operations. Unlike RADIUS, TACACS+ provides more granular and flexible control over user access privileges. For example, the AAA operations are separated by TACACS+, whereas RADIUS combines the authentication and authorization services into a single function. Because TACACS+ separates these functions, administrators have more control over access to configuration commands. For example, TACACS+ can be used to provide all AAA functions, or to provide only authorization and accounting while allowing another service to perform authentication. In addition, TACACS+ encrypts the entire contents of packets, thus providing additional security. TACACS+ uses Transmission Control Protocol (TCP) port 49 for transport.



Configuring AAA

Enabling AAA

```
RouterA(config)#aaa new-model
```

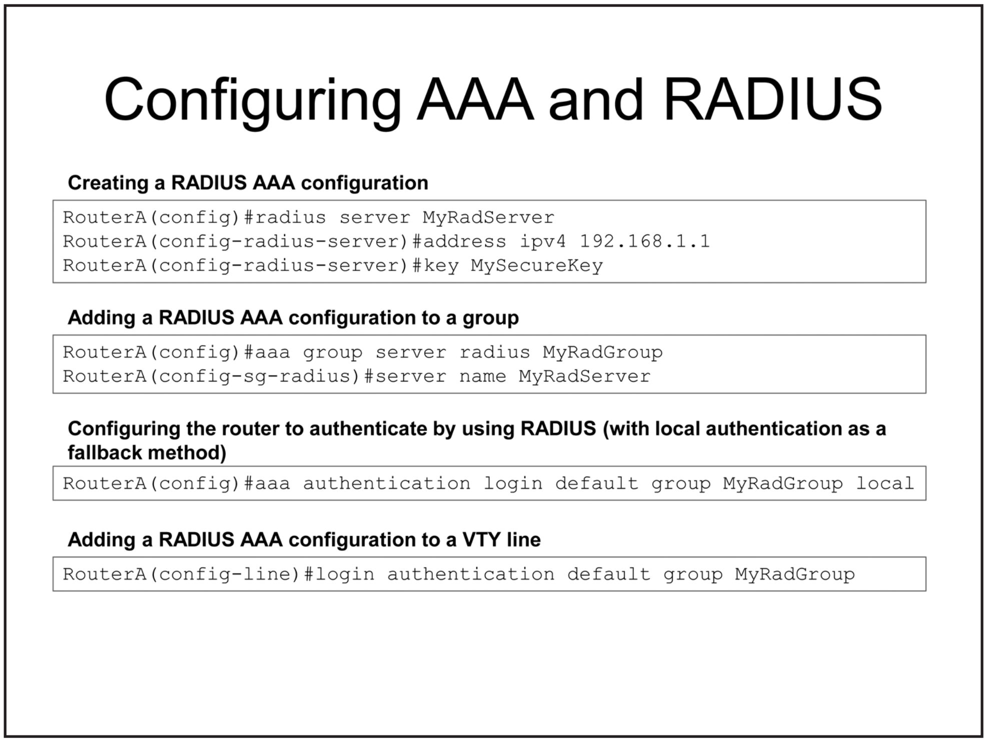
Creating a backup user on the local device

```
RouterA(config)#username backupuser password c1$c0@dM1n
```

Configuring AAA

As an alternative to local authentication, you could configure a RADIUS server or a TACACS+ server as the authentication source for a Cisco device. First, you would issue the **aaa new-model** command in global configuration mode. The **aaa new-model** command enables AAA services on the local device. Other AAA commands, such as the **aaa authentication** command, cannot be issued until the **aaa new-model** command has been issued on the device.

Next, you would issue the **username** command to create a local database user as a backup so that you do not become locked out of the device you are configuring. When the **aaa new-model** command is issued, local authentication is automatically applied to all interfaces and VTY lines but not to the CON line. Creating a backup local user therefore prevents you from becoming locked out of the console.



Configuring AAA and RADIUS

Creating a RADIUS AAA configuration

```
RouterA(config)#radius server MyRadServer
RouterA(config-radius-server)#address ipv4 192.168.1.1
RouterA(config-radius-server)#key MySecureKey
```

Adding a RADIUS AAA configuration to a group

```
RouterA(config)#aaa group server radius MyRadGroup
RouterA(config-sg-radius)#server name MyRadServer
```

Configuring the router to authenticate by using RADIUS (with local authentication as a fallback method)

```
RouterA(config)#aaa authentication login default group MyRadGroup local
```

Adding a RADIUS AAA configuration to a VTY line

```
RouterA(config-line)#login authentication default group MyRadGroup
```

Configuring AAA and RADIUS

After you have enabled AAA services and configured a local backup user, you can configure RADIUS on the Cisco device. After RADIUS is correctly configured, you should configure the device's AAA service to use RADIUS. You begin the process of configuring RADIUS by issuing the **radius server** *configuration-name* command in global configuration mode, where *configuration-name* is the name of the RADIUS server or configuration you want to create.

In RADIUS server configuration mode, you can assign the RADIUS server an IP address, authentication port number, and accounting port number by issuing the **address ipv4** *ip-address* [**auth-port** *port-number*] [**acct-port** *port-number*] command, where *ip-address* is the IP version 4 (IPv4) address you want the server to use. Both the **auth-port** keyword and the **acct-port** keyword are optional and can be used to configure the specific UDP ports on which the authentication and accounting services operate. If you do not specify authentication or accounting port numbers, the RADIUS server will use the default UDP port values of 1812 and 1813, respectively.

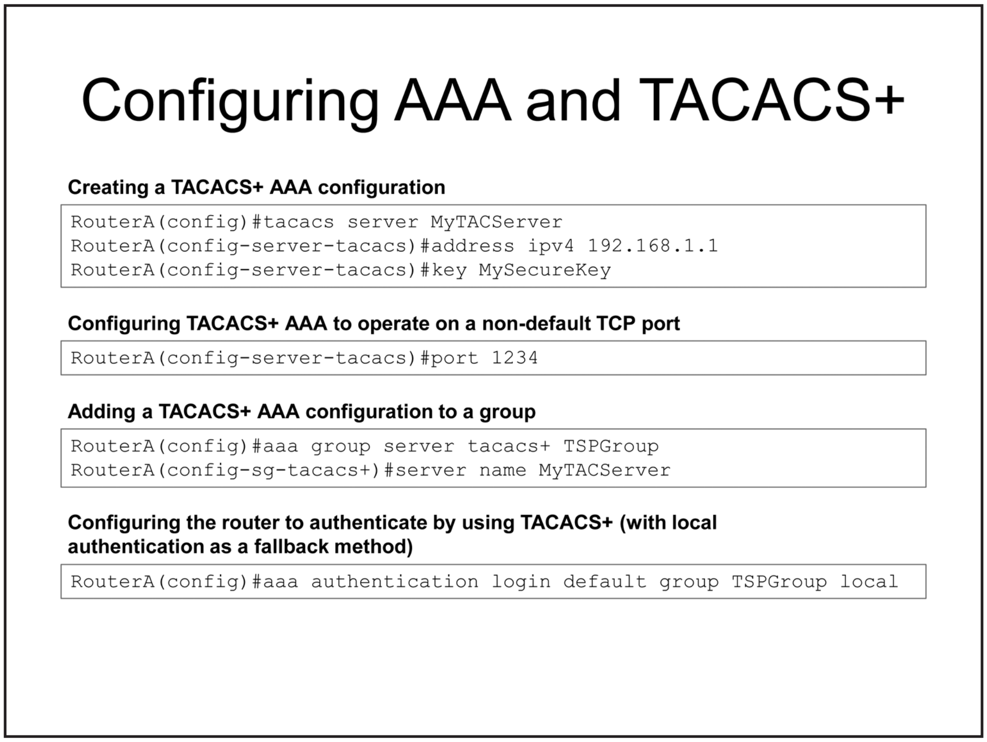
Next, you should issue the **key** *shared-key-string* command, where *shared-key-string* is a case-sensitive string that is used to verify the source and integrity of communications between the RADIUS server and a RADIUS client or a RADIUS proxy. This shared secret must match the shared secret that is configured on the RADIUS client in order for communications between clients and the server to be secure.

After the RADIUS server is configured, you should associate the server with a AAA RADIUS server group. The AAA server group will ultimately be what you configure AAA to use as an authentication service on the device. You can add more than one RADIUS server to a AAA RADIUS server group. Before you can associate a server with a group, you must either create the group or place the device into the group's configuration mode by issuing the **aaa group server radius** *group-name* command, where *group-name* is the name of the RADIUS server group you want to create or configure.

In RADIUS server group configuration mode, you should issue the **server name** *configuration-name* command to associate the RADIUS server you previously configured with the server group. The *configuration-name* parameter should be the same value that you issued for the **radius server** command when you first configured the RADIUS server.

Finally, you should issue the **aaa authentication login** command to configure the router to use AAA authentication. You can also issue the **login authentication** command in line configuration mode to configure the CON, VTY, or AUX line to use AAA authentication.

When configuring a RADIUS server to secure access to a Cisco device, you should ensure that the device will attempt to authenticate against the RADIUS server first. If the RADIUS server is not available or cannot authenticate, the device should use its local database as a fallback option. The **aaa authentication login default group MyRadGroup** local command achieves this. The **default** keyword specifies that the device should use AAA authentication by default. The **group** keyword followed by the **MyRadGroup** parameter specifies that the device should first attempt to use the AAA group named MyRadGroup for authentication. The **local** keyword specifies that the local user database should be used for authentication if the RADIUS server is unavailable.



Configuring AAA and TACACS+

Creating a TACACS+ AAA configuration

```
RouterA(config)#tacacs server MyTACServer
RouterA(config-server-tacacs)#address ipv4 192.168.1.1
RouterA(config-server-tacacs)#key MySecureKey
```

Configuring TACACS+ AAA to operate on a non-default TCP port

```
RouterA(config-server-tacacs)#port 1234
```

Adding a TACACS+ AAA configuration to a group

```
RouterA(config)#aaa group server tacacs+ TSPGroup
RouterA(config-sg-tacacs+)#server name MyTACServer
```

Configuring the router to authenticate by using TACACS+ (with local authentication as a fallback method)

```
RouterA(config)#aaa authentication login default group TSPGroup local
```

Configuring AAA and TACACS+

Although the two AAA services are unique, configuring a TACACS+ server on a Cisco device is similar to configuring a RADIUS server. After you have enabled AAA services and configured a local backup user, you can configure TACACS+ on the Cisco device. After TACACS+ is correctly configured, you should configure the device's AAA service to use TACACS+. You begin the process of configuring TACACS+ by issuing the **tacacs server** *configuration-name* command in global configuration mode, where *configuration-name* is the name of the TACACS+ server or configuration you want to create.

In TACACS+ server configuration mode, you can assign the TACACS+ server an IP address by issuing the **address ipv4** *ip-address* command, where *ip-address* is the IPv4 address you want the server to use. Unlike a RADIUS configuration, the TACACS+ configuration process uses a separate command for specifying a port other than the default TCP port of 49. If you do not configure the **port** command, the TACACS+ server will operate on its default TCP port. The **port** *port-number* command, where *port-number* is the TCP port number on which you want the server to operate, can be issued in TACACS+ server configuration mode.

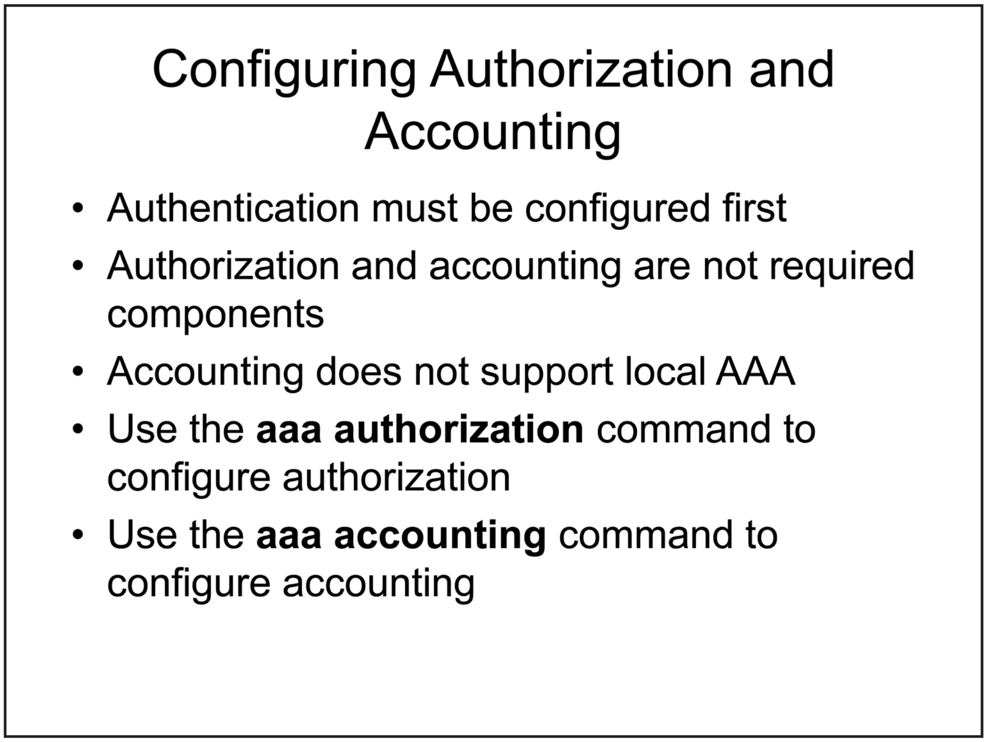
Next, you should issue the **key** *shared-key-string* command in TACACS+ server configuration mode, where *shared-key-string* is a case-sensitive string that is used to secure communications between the TACACS+ server and a TACACS+ client. Unlike RADIUS, this key is used to encrypt all communications between a TACACS+ server and a TACACS+ client, not just the password.

After the TACACS+ server is configured, you should associate the server with a AAA TACACS+ server group. The AAA server group will ultimately be what you configure AAA to use as an authentication service on the device. You can add more than one TACACS+ server to a AAA TACACS+ server group. Before you can associate a server with a group, you must either create the group or place the device into the group's configuration mode by issuing the **aaa group server tacacs+** *group-name* command, where *group-name* is the name of the TACACS+ server group you want to create or configure.

In TACACS+ server group configuration mode, you should issue the **server name** *configuration-name* command to associate the TACACS+ server you previously configured with the server group. The *configuration-name* parameter should be the same value that you issued for the **tacacs server** command when you first configured the TACACS+ server.

Finally, you should issue the **aaa authentication login** command to configure the router to use AAA authentication. You can also issue the **login authentication** command in line configuration mode to configure the CON, VTY, or AUX line to use AAA authentication.

When configuring a TACACS+ server to secure access to a Cisco device, you should ensure that the device will attempt to authenticate against the TACACS+ server first. If the TACACS+ server is not available or cannot authenticate, the device should use its local database to authenticate console and VTY connections. The **aaa authentication login default group TSPGroup local** command achieves this. The **default** keyword specifies that the device should use AAA authentication by default. The **group** keyword followed by the **TSPGroup** parameter specifies that the device should first attempt to use the AAA group named TSPGroup for authentication. The **local** keyword specifies that the local user database should be used for authentication if the TACACS+ server is unavailable.



Configuring Authorization and Accounting

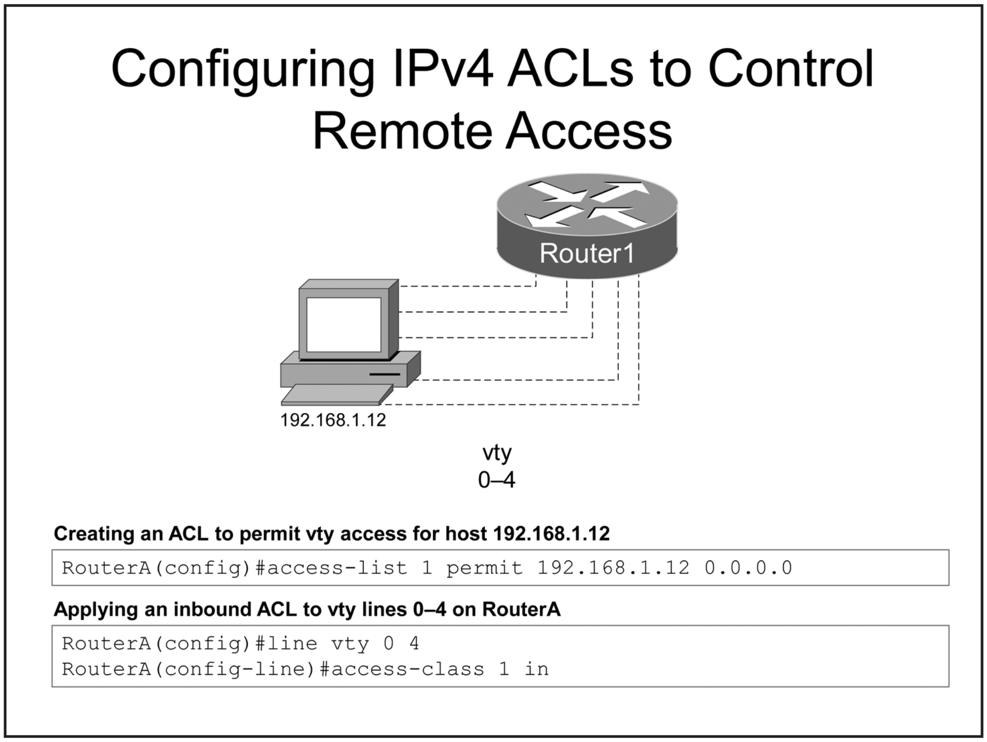
- Authentication must be configured first
- Authorization and accounting are not required components
- Accounting does not support local AAA
- Use the **aaa authorization** command to configure authorization
- Use the **aaa accounting** command to configure accounting

Configuring Authorization and Accounting

Authentication is a required AAA component; authorization and accounting are not. Authentication tests whether you are who you say you are. Authorization specifies what you are allowed to do and when you are allowed to do it. Accounting measures the network resources that you use and is often used for billing purposes, trend analysis, or capacity planning.

Authentication and authorization support local AAA. However, accounting cannot use the local user database and requires an external AAA server.

The **aaa authorization** and **aaa accounting** commands are configured similarly to the **aaa authentication** command. However, the **aaa authentication** command must be configured first.



Configuring IPv4 ACLs to Control Remote Access

Creating an ACL to permit vty access for host 192.168.1.12

```
RouterA(config)#access-list 1 permit 192.168.1.12 0.0.0.0
```

Applying an inbound ACL to vty lines 0–4 on RouterA

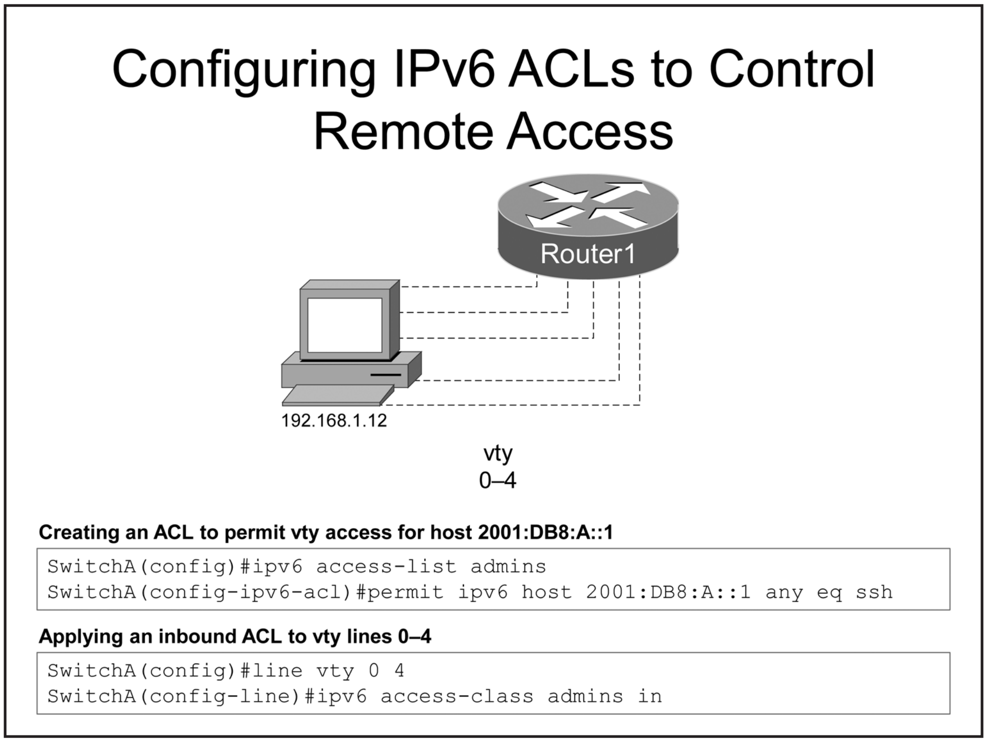
```
RouterA(config)#line vty 0 4
RouterA(config-line)#access-class 1 in
```

Configuring IPv4 ACLs to Control Remote Access

You can control remote access to a Cisco device by limiting the types of protocols that the device accepts and by restricting access to a particular range of IP addresses. Some Cisco devices have five VTY lines by default, numbered 0–4; others have 16 VTY lines, numbered 0–15. You might assume that an extended access control list (ACL) is necessary to limit VTY access to a specific set of protocols. However, you should use the **transport input** command, not an extended ACL, to filter incoming protocols. The **transport input all** command allows access by all supported VTY line protocols. The **transport input telnet** command prevents non-Telnet connections to the VTY lines. The **transport input none** command prevents all incoming connections to the VTY lines. Because you need to filter access to the VTY lines by only source address, a standard ACL is sufficient.

To apply an inbound ACL to a VTY line, you should issue the **access-class** *acl-number* **in** command from line configuration mode, where *acl-number* is the name or number of the ACL. To enter line configuration mode for VTY lines, you should issue the **line vty** *start-line end-line* command, where *start-line* and *end-line* indicate the range of VTY lines that you are configuring.

In the example above, the **access-list 1 permit 192.168.1.12 0.0.0.0** command permits all traffic from the host at 192.168.1.12; you could also issue the **access-list 1 permit host 192.168.1.12** command. The **line vty 0 4** command enters VTY line configuration mode for VTY lines **0** through **4**. The **access-class 1 in** command applies access list **1** to inbound traffic on the VTY lines.



Configuring IPv6 ACLs to Control Remote Access

Creating an ACL to permit vty access for host 2001:DB8:A::1

```
SwitchA(config)#ipv6 access-list admins
SwitchA(config-ipv6-acl)#permit ipv6 host 2001:DB8:A::1 any eq ssh
```

Applying an inbound ACL to vty lines 0–4

```
SwitchA(config)#line vty 0 4
SwitchA(config-line)#ipv6 access-class admins in
```

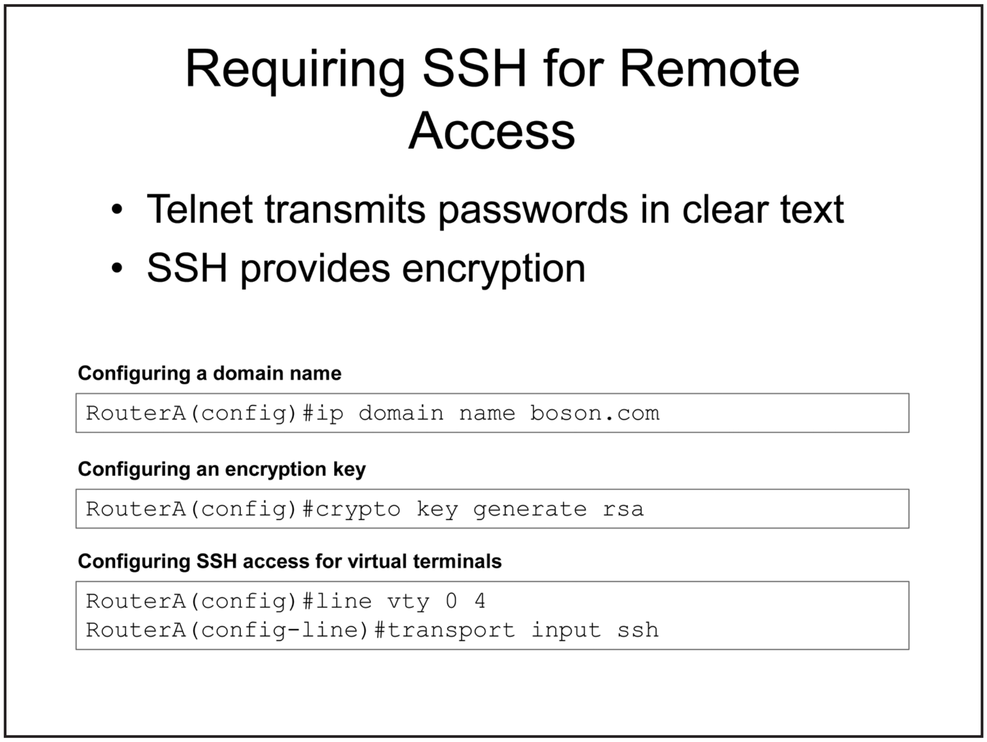
Configuring IPv6 ACLs to Control Remote Access

In many ways, IP version 6 (IPv6) ACLs work similarly to IPv4 ACLs in that you can configure IPv6 ACLs to control remote access to a given device. For example, IPv6 ACLs always have an implicit deny rule at the end. However, IPv6 ACLs do not support some features that IPv4 ACLs support and the method of configuring an IPv6 ACL is different.

First you should configure the IPv6 ACL by using the **ipv6 access-list** *access-list-name* command, where *access-list-name* is an ACL identifier string. This places the device into IPv6 ACL configuration mode. Unlike IPv4 ACLs, IPv6 ACLs support only named ACLs. Therefore, you cannot configure a standard, extended, or Media Access Control (MAC) ACL for IPv6.

In IPv6 ACL configuration mode, you should issue the **deny** | **permit** *protocol* {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* [*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* [*port-number*]] command to configure the ACL rules. For example, the **permit ipv6 host 2001:DB8:A::1 any eq ssh** command permits a single host that has been assigned the IPv6 address of 2001:DB8:A::1 to connect to any destination by using Secure Shell (SSH).

To apply an inbound IPv6 ACL to a VTY line, you should issue the **ipv6 access-class** *access-list-name* **in** command from line configuration mode, where *access-list-name* is the name of the ACL.



Requiring SSH for Remote Access

- Telnet transmits passwords in clear text
- SSH provides encryption

Configuring a domain name

```
RouterA(config)#ip domain name boson.com
```

Configuring an encryption key

```
RouterA(config)#crypto key generate rsa
```

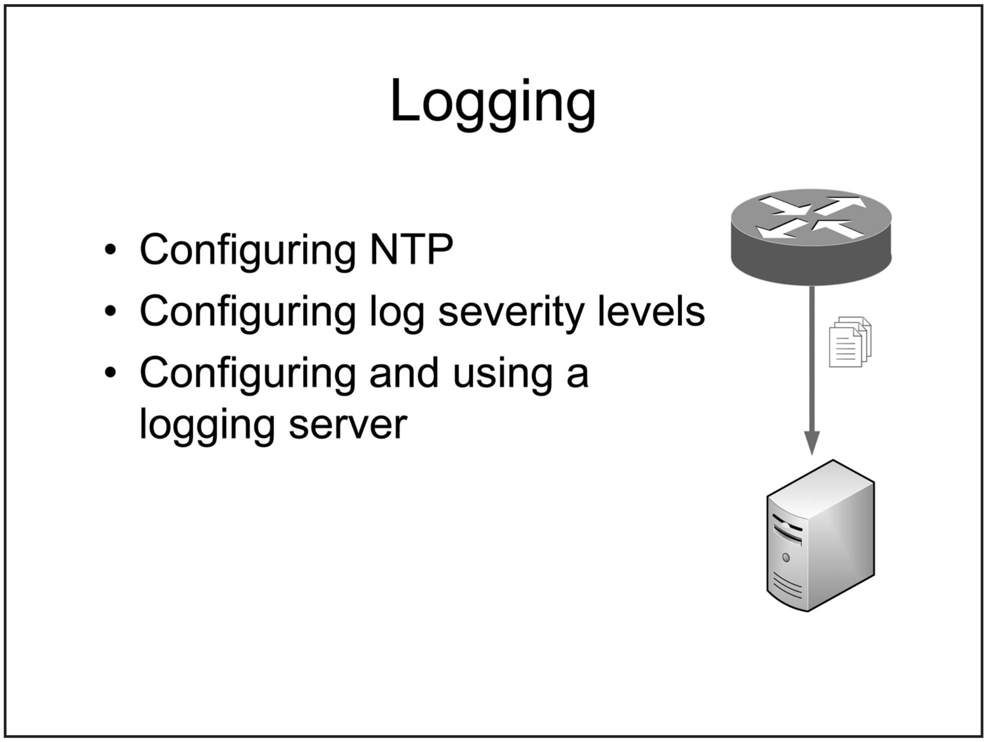
Configuring SSH access for virtual terminals

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

Requiring SSH for Remote Access

Because Telnet transmits information in clear text, it should not be used as a management protocol. Instead, all network devices should be configured to accept only encrypted connections, such as SSH. SSH requires the use of a user name and password for authentication. You can use the **transport input** command to specify the types of connections that a router or switch will accept. You can issue the **transport input ssh** command from terminal line configuration mode to configure a Cisco device to accept only SSH connections on the specified lines.

If SSH has not been previously configured, you should first configure SSH. To configure SSH, you must first have configured a host name and a domain name on the device. You can issue the **ip domain name** command to configure a domain name. After you have configured the host name and domain name, you can issue the **crypto key generate rsa** command to create an RSA encryption key for SSH.

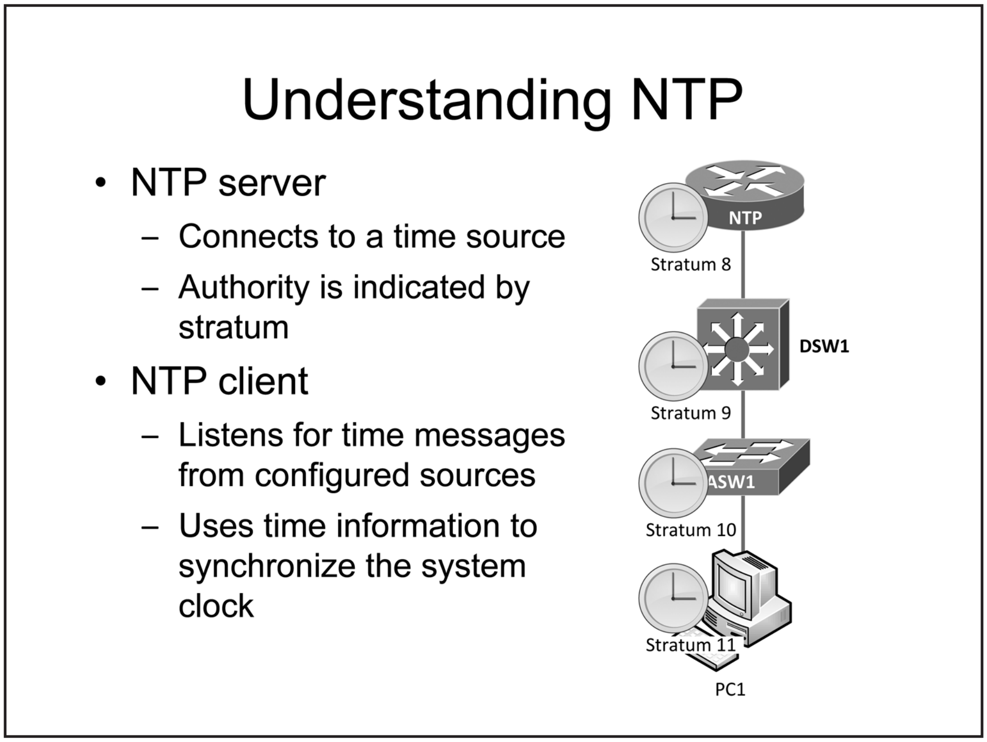


Logging

- Configuring NTP
- Configuring log severity levels
- Configuring and using a logging server

Logging

Logging provides a way to monitor a network and its devices so that an administrator can spot and address potential problems quickly. In this section, you will learn how to configure NTP, how to configure log severity levels, and how to configure and use a logging server.



Understanding NTP

- NTP server
  - Connects to a time source
  - Authority is indicated by stratum
- NTP client
  - Listens for time messages from configured sources
  - Uses time information to synchronize the system clock

Understanding NTP

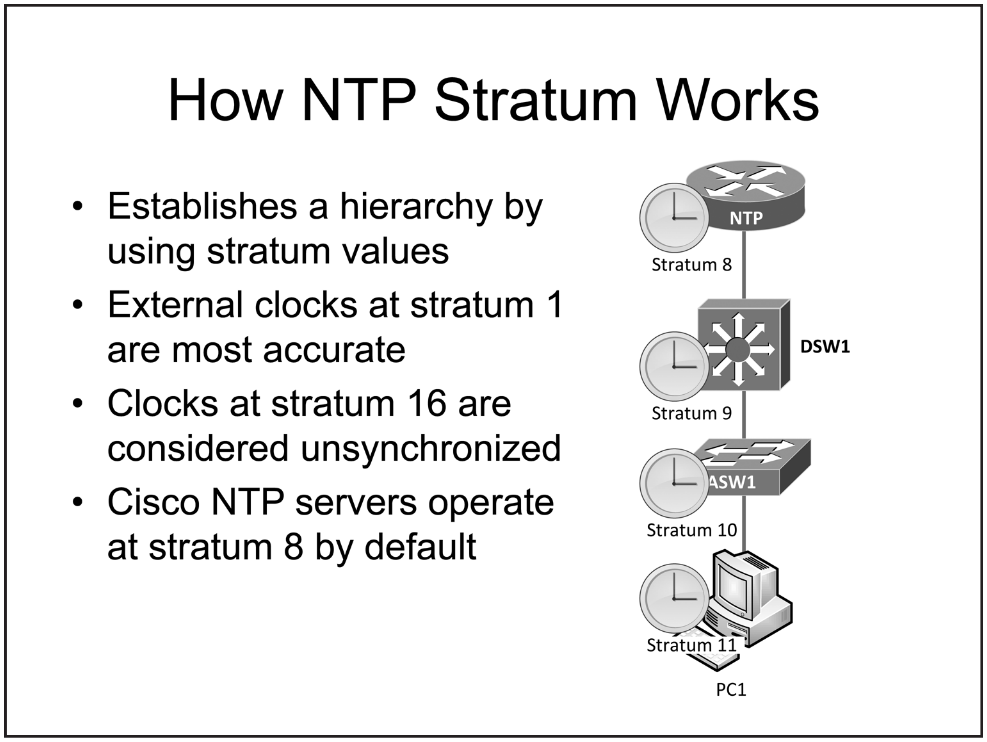
Synchronized time is an essential component of many network services, particularly those that provide event logging, authentication, or encryption. For example, Kerberos authentication can fail if there is too great a discrepancy between client and server system clocks.

Synchronized time increases the accuracy of activity analysis from network logs. Cisco devices can use NTP to synchronize the date and time on the switch to an internal network time server or to an external source. NTP requires no more than one packet per minute to keep Cisco devices synchronized to within one-millisecond precision.

To configure a switch to synchronize the date and time with a specific time server, you should issue the **ntp server** *ip-address* command in global configuration mode.

An NTP server receives its time information from a time source with a higher level of authority than itself. The authority of a particular time server is indicated by its stratum.

An NTP client listens for time messages from configured NTP servers and can be configured to form associations with multiple servers. If multiple servers are configured, the client will typically synchronize with the lowest stratum server unless there is too great of a time differential between that server and servers in the higher strata.



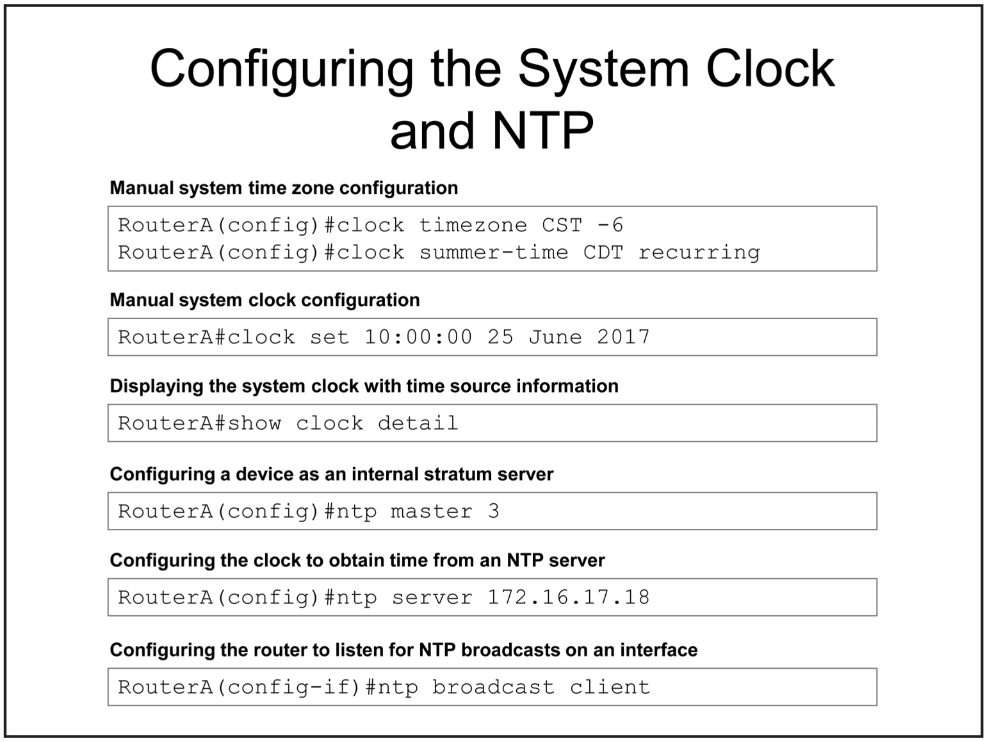
How NTP Stratum Works

- Establishes a hierarchy by using stratum values
- External clocks at stratum 1 are most accurate
- Clocks at stratum 16 are considered unsynchronized
- Cisco NTP servers operate at stratum 8 by default

How NTP Stratum Works

Lower stratum values indicate more authoritative time sources. Therefore, devices with higher stratum values will trust devices with lower stratum values.

An NTP server that connects directly to an authoritative, external time source, such as an atomic clock or global positioning system (GPS) unit, is considered a stratum 1 NTP server. Stratum 1 devices are considered the most authoritative sources of clock information in the NTP hierarchy. Stratum 1 NTP servers distribute their time information to devices in the higher strata. For example, stratum 2 devices function as NTP clients to stratum 1 devices and as NTP servers to devices in the higher strata.



Configuring the System Clock and NTP

Manual system time zone configuration

```
RouterA(config)#clock timezone CST -6
RouterA(config)#clock summer-time CDT recurring
```

Manual system clock configuration

```
RouterA#clock set 10:00:00 25 June 2017
```

Displaying the system clock with time source information

```
RouterA#show clock detail
```

Configuring a device as an internal stratum server

```
RouterA(config)#ntp master 3
```

Configuring the clock to obtain time from an NTP server

```
RouterA(config)#ntp server 172.16.17.18
```

Configuring the router to listen for NTP broadcasts on an interface

```
RouterA(config-if)#ntp broadcast client
```

Configuring the System Clock and NTP

Because NTP data is formatted as Coordinated Universal Time (UTC) time instead of the local time zone, you could optionally issue the **clock timezone** command to configure the local time zone so that time values are displayed relative to the local time zone. The syntax for the **clock timezone** command is **clock timezone** *zone hours-offset* [*minutes-offset*], where *zone* is the acronym of a standard time zone, such as **CST** for Central Standard Time, *hours-offset* is the number of hours of offset from UTC time, and *minutes-offset* is an optional number of minutes the time zone is offset from UTC. You can also issue the **clock summer-time** command to indicate whether the router should participate in Daylight Saving Time (DST).

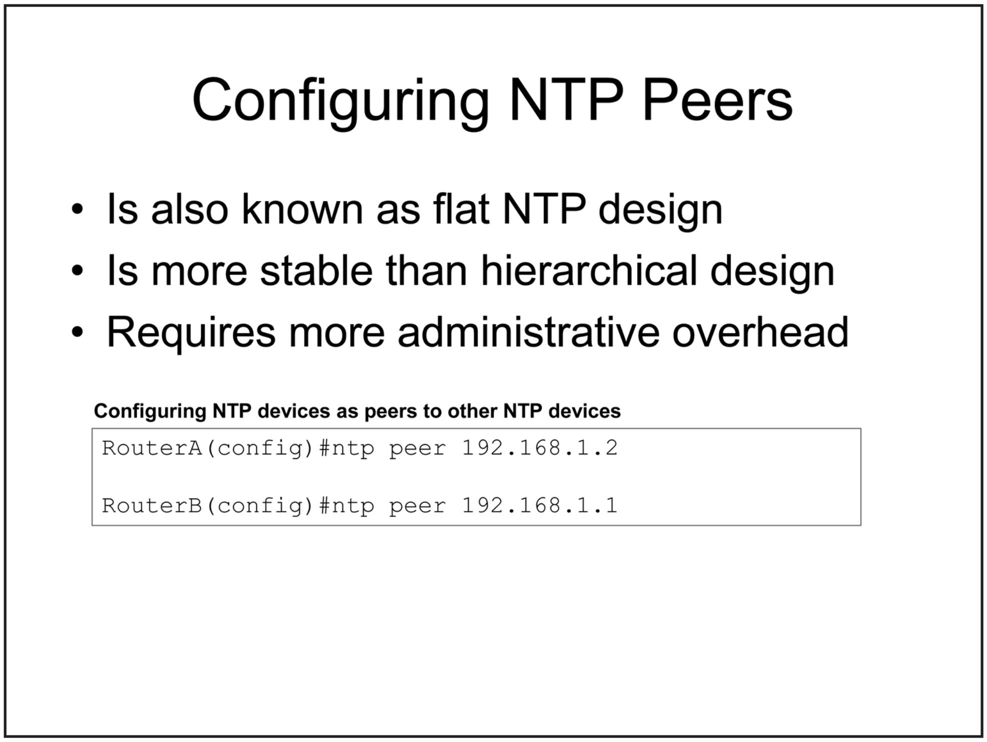
If a Cisco device cannot access a network time source or an external time source, you can set the internal clock manually by issuing the **clock set** command. The syntax for the **clock set** command is **clock set** *hh***:***mm***:***ss day month year* or **clock set** *hh***:***mm***:***ss month day year*.

You can display the system clock by issuing the **show clock** command. The **show clock detail** command will display the system clock along with time source information.

You can configure a Cisco device as an NTP server with the internal clock as its time source by issuing the **ntp master** [*stratum*] command. The **ntp master** command defaults to stratum 8, but you can specify a stratum as a parameter to change the default behavior if necessary.

Issuing the **ntp server** [*address*] command configures a Cisco device to function as an NTP static client for the specified time source and also as an NTP server for devices at higher strata. The stratum of the Cisco device will be one level higher than the stratum of the specified NTP server.

Issuing the **ntp broadcast client** command from interface configuration mode configures a Cisco device to listen on the interface for NTP broadcasts from NTP servers. The difference between a broadcast client and a static client is that a broadcast client can receive its time from any NTP server.



Configuring NTP Peers

- Is also known as flat NTP design
- Is more stable than hierarchical design
- Requires more administrative overhead

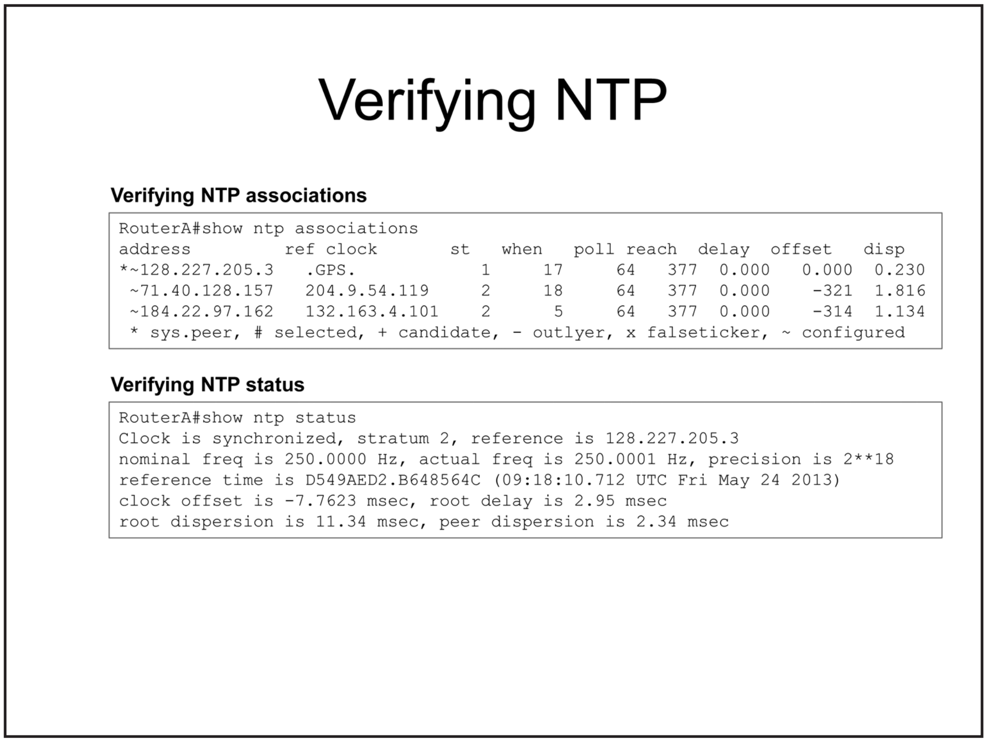
Configuring NTP devices as peers to other NTP devices

```
RouterA(config)#ntp peer 192.168.1.2

RouterB(config)#ntp peer 192.168.1.1
```

Configuring NTP Peers

Configuring a Cisco router as an NTP peer enables symmetric active mode on the router. A device in symmetric active mode attempts to mutually synchronize with another NTP host. When symmetric active mode is enabled, the host might synchronize the peer or it might be synchronized by the peer. This is also known as flat NTP design, where no device has priority over another. Since symmetric active mode has multiple devices that participate in mutual synchronization, it is more stable than a hierarchical client/server design. However, it requires more administrative overhead because each peer must be configured with the address of one or more other NTP peers. To configure devices as NTP peers, you should issue the **ntp peer** [*address*] command on each device.



Verifying NTP

Verifying NTP associations

```
RouterA#show ntp associations
address         ref clock       st   when   poll reach  delay  offset   disp
*~128.227.205.3   .GPS.            1     17     64   377  0.000   0.000  0.230
 ~71.40.128.157   204.9.54.119     2     18     64   377  0.000    -321  1.816
 ~184.22.97.162   132.163.4.101    2      5     64   377  0.000    -314  1.134
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
```

Verifying NTP status

```
RouterA#show ntp status
Clock is synchronized, stratum 2, reference is 128.227.205.3
nominal freq is 250.0000 Hz, actual freq is 250.0001 Hz, precision is 2**18
reference time is D549AED2.B648564C (09:18:10.712 UTC Fri May 24 2013)
clock offset is -7.7623 msec, root delay is 2.95 msec
root dispersion is 11.34 msec, peer dispersion is 2.34 msec
```

Verifying NTP

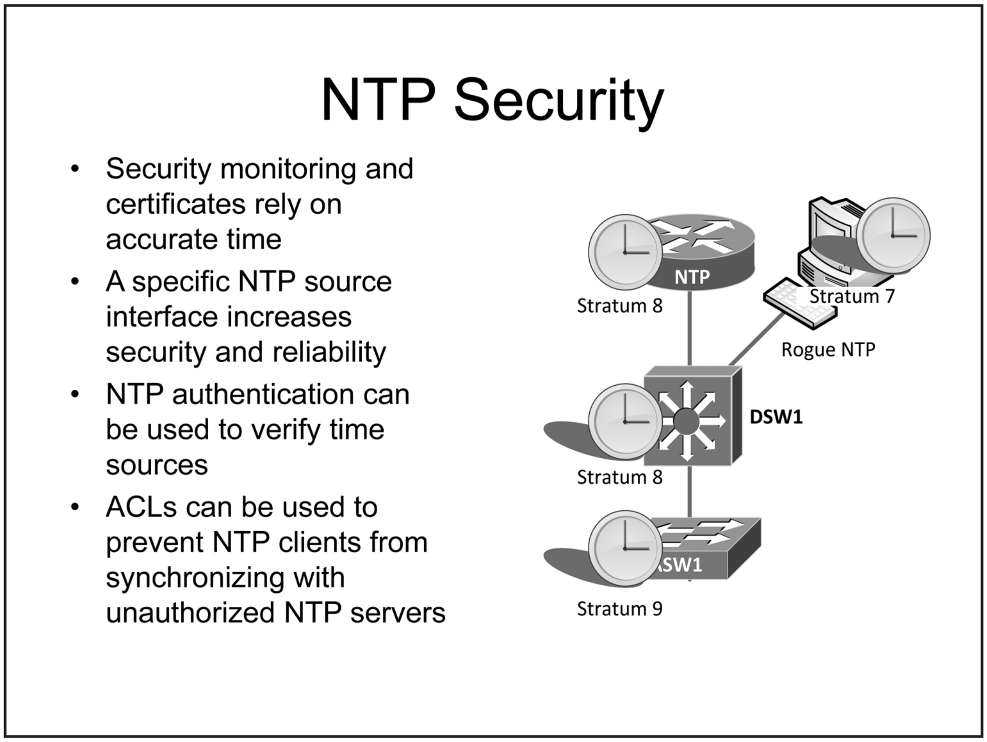
You can use the **show ntp associations** command to verify the NTP configuration on a Cisco device. The output of the **show ntp associations** command shows the IP addresses of configured NTP servers and their respective clock sources, strata, and reachability statistics. For example, in the following command output, the NTP server at IP address 128.227.205.3 is a stratum 1 server that uses a GPS time source as its time source:

```
address          ref clock       st   when   poll reach  delay  offset   disp
*~128.227.205.3   .GPS.           1     17     64   377  0.000   0.000  0.230
 ~71.40.128.157   204.9.54.119   2     18     64   377  0.000    -321  1.816
 ~184.22.97.162   132.163.4.101  2      5     64   377  0.000    -314  1.134
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
```

The asterisk (*) next to the IP address in the command output indicates that this server is the NTP master time source to which the Cisco device is synched. A tilde (~) next to an IP address indicates that the address was manually configured.

You can use the **show ntp status** command to verify operation of NTP on a Cisco device. The example command output indicates that the system has synchronized its clock with the NTP server at IP address 128.227.205.3 and that it is functioning as a stratum 2 NTP server for devices in higher-numbered strata:

```
Clock is synchronized, stratum 2, reference is 128.227.205.3
nominal freq is 250.0000 Hz, actual freq is 250.0001 Hz, precision is 2**18
reference time is D549AED2.B648564C (09:18:10.712 UTC Fri May 24 2013)
clock offset is -7.7623 msec, root delay is 2.95 msec
root dispersion is 11.34 msec, peer dispersion is 2.34 msec
```



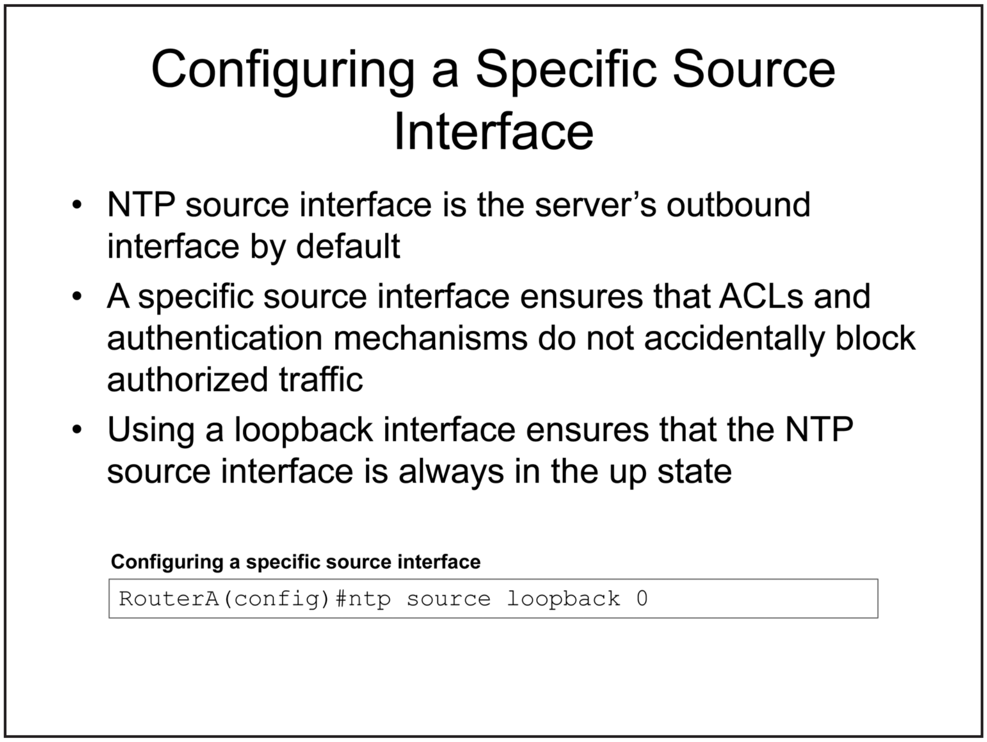
NTP Security

- Security monitoring and certificates rely on accurate time
- A specific NTP source interface increases security and reliability
- NTP authentication can be used to verify time sources
- ACLs can be used to prevent NTP clients from synchronizing with unauthorized NTP servers

NTP Security

Keeping accurate time across a network infrastructure is important for monitoring and security. Unsynchronized time can cause problems with certificate-based authentication methods. Network troubleshooting can become difficult or impossible without accurate timestamps.

There are several methods that you can use to prevent NTP clients from synchronizing with unauthorized NTP servers. First, you will learn about configuring a specific source interface to increase security and reliability. Next, you will learn how to configure NTP authentication. Finally, you will learn how to use ACLs to allow NTP synchronization only with certain devices.



Configuring a Specific Source Interface

- NTP source interface is the server's outbound interface by default
- A specific source interface ensures that ACLs and authentication mechanisms do not accidentally block authorized traffic
- Using a loopback interface ensures that the NTP source interface is always in the up state

Configuring a specific source interface

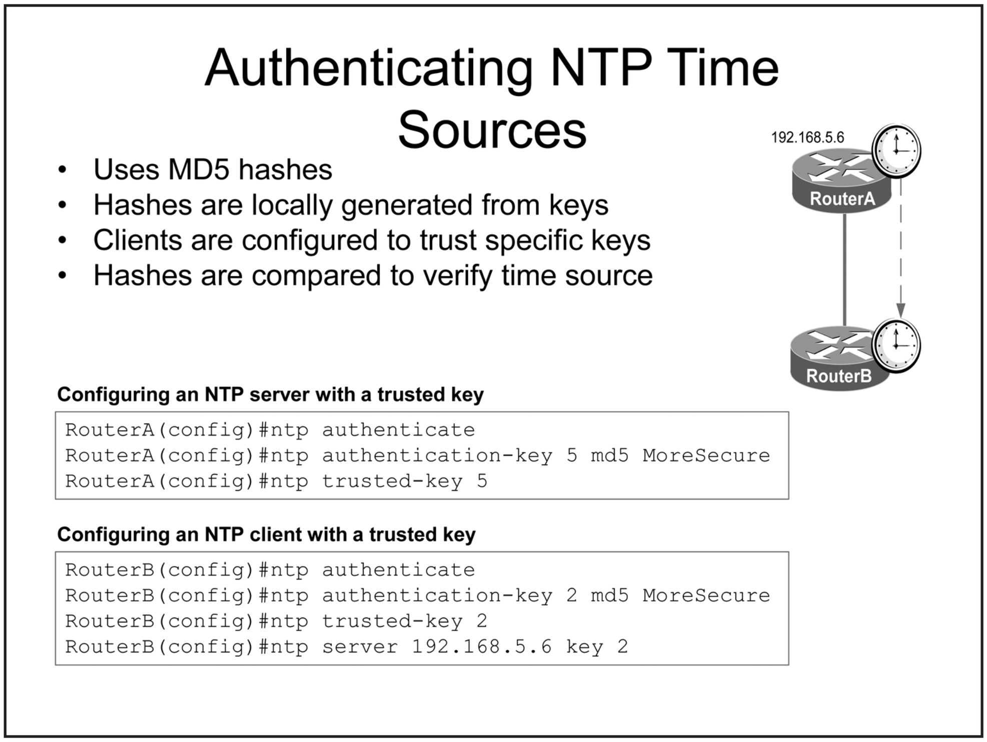
```
RouterA(config)#ntp source loopback 0
```

Configuring a Specific Source Interface

By default, when a Cisco device sends NTP packets through an interface, it uses that interface's IP address as the source IP address. You can change this behavior by configuring a specific source interface for NTP.

Creating a specific source interface for an NTP server ensures that NTP packets always come from the same IP address no matter what interface the NTP server uses to send the packets. This allows you to configure downstream NTP clients so that they will allow and authenticate only with that IP address.

To configure a specific source interface, issue the **ntp source** *interface-type interface-number* command from global configuration mode. After you issue this command, NTP packets will use the IP address of the specified interface no matter which interface the NTP packets are sent through. However, if the NTP source interface is down, NTP cannot use that interface's IP address. Therefore, Cisco recommends using a loopback interface as the NTP specific source interface to ensure that it is always in the up state.



Authenticating NTP Time Sources

- Uses MD5 hashes
- Hashes are locally generated from keys
- Clients are configured to trust specific keys
- Hashes are compared to verify time source

Configuring an NTP server with a trusted key

```
RouterA(config)#ntp authenticate
RouterA(config)#ntp authentication-key 5 md5 MoreSecure
RouterA(config)#ntp trusted-key 5
```

Configuring an NTP client with a trusted key

```
RouterB(config)#ntp authenticate
RouterB(config)#ntp authentication-key 2 md5 MoreSecure
RouterB(config)#ntp trusted-key 2
RouterB(config)#ntp server 192.168.5.6 key 2
```

Authenticating NTP Time Sources

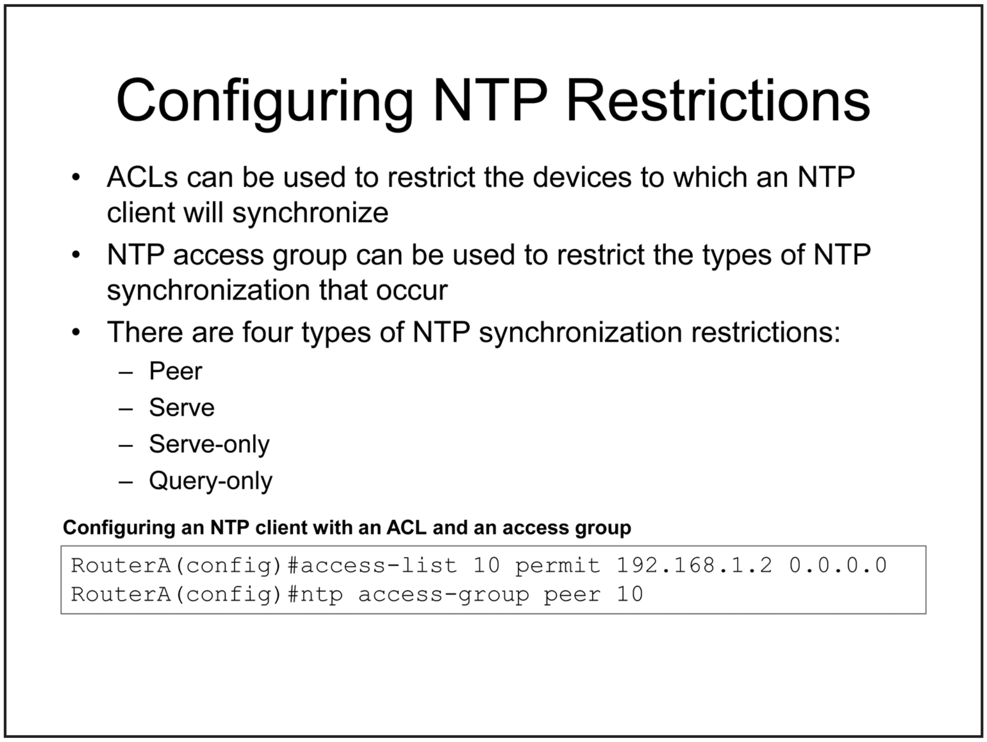
You can configure NTP authentication so that NTP clients will synchronize only with authenticated NTP servers. NTP authentication uses only MD5 hashes for authentication.

Configuring the NTP authentication key is done the same way on both NTP servers and NTP clients. First, you must enable NTP authentication on a device by issuing the **ntp authenticate** command from global configuration mode.

Next, create the authentication key by issuing the **ntp authentication-key** command from global configuration mode. The syntax for the **ntp authentication-key** command is **ntp authentication-key** *key-number* **md5** *key.* The *key-number* is a locally significant identifier, so it does not have to match on the NTP server and NTP client. The key can be any case-sensitive string of up to 32 characters.

Finally, configure NTP with the keys that it is allowed to use by issuing the **ntp trusted-key** command from global configuration mode. The syntax for the ntp trusted-key command is **ntp trusted-key** *key-number* [**-** *end-key-number*], where *key-number* is the key number that you used in the **ntp authentication-key** command. If you have configured several authentication keys, you can configure NTP to use all of them. For example, to configure NTP to use keys 1 through 3, you should issue the **ntp trusted-key 1 - 3** command.

After you have issued these commands on an NTP client, you must still configure the NTP client so that it authenticates the NTP server. To do so, issue the **ntp server** *ip-address* **key** *key-number* command from global configuration mode. The *ip-address* variable is the IP address of the NTP server; if you have created a specific NTP source interface on the NTP server, you should use the IP address of that interface. The *key-number* variable is the locally significant key number that you created on the NTP client, not the key number that you configured on the NTP server.



Configuring NTP Restrictions

- ACLs can be used to restrict the devices to which an NTP client will synchronize
- NTP access group can be used to restrict the types of NTP synchronization that occur
- There are four types of NTP synchronization restrictions:
  - Peer
  - Serve
  - Serve-only
  - Query-only

Configuring an NTP client with an ACL and an access group

```
RouterA(config)#access-list 10 permit 192.168.1.2 0.0.0.0
RouterA(config)#ntp access-group peer 10
```

Configuring NTP Restrictions

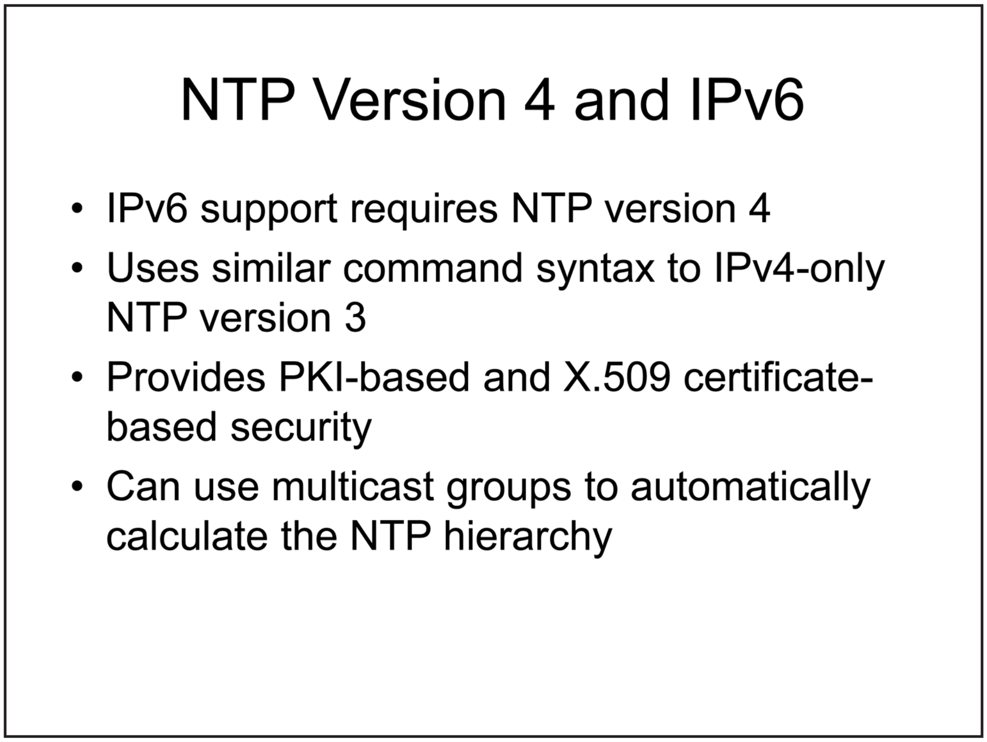
You can use ACLs to specify the devices to which an NTP client will synchronize. You can also configure restrictions on what kind of NTP messages are allowed.

After you have created an ACL, you should apply the ACL to NTP by issuing the **ntp access-group** command from global configuration mode. The syntax for the **ntp access-group** command is **ntp access-group** [**ipv4** | **ipv6**] {**peer** | **serve** | **serve-only** | **query-only**} {*acl-number* | *acl-name*} [**kod**].

There are four types of NTP synchronization restrictions:

- The **peer** keyword allows time requests, NTP control queries, and synchronization with the remote device.
- The **serve** keyword allows time requests and NTP control queries but does not allow synchronization.
- The **serve-only** keyword allows only time requests.
- The **query-only** keyword allows only NTP control queries.

NTP access groups support standard, extended, and named ACLs. The optional **kod** keyword can be used to send a Kiss-of-Death packet to any host that sends a packet that does not comply with the NTP policy.



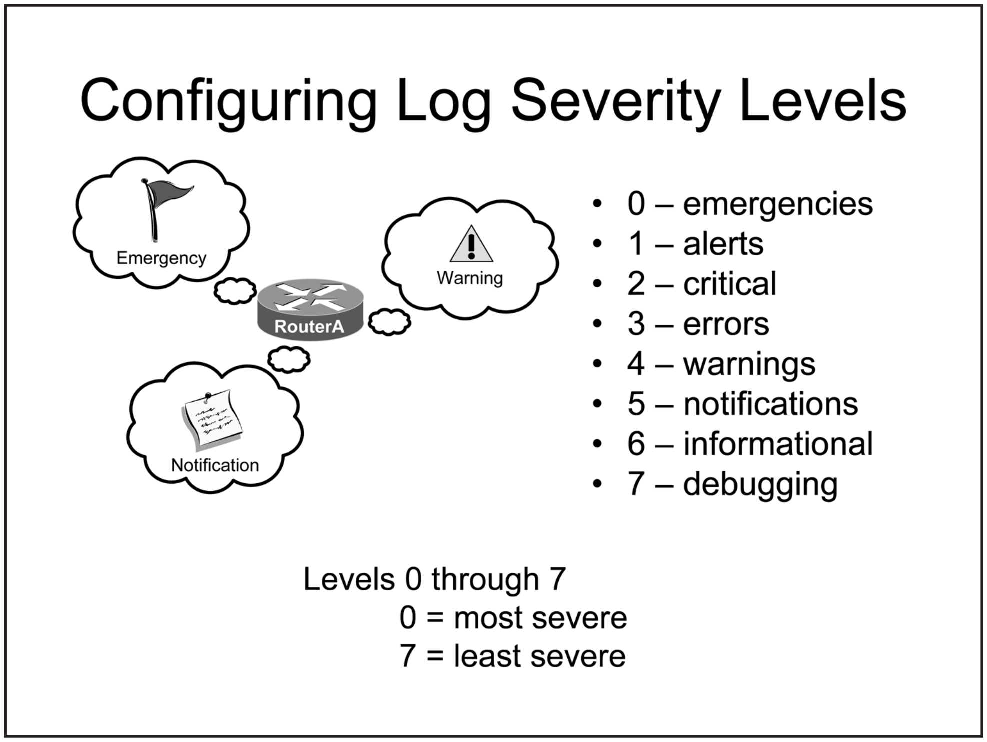
NTP Version 4 and IPv6

- IPv6 support requires NTP version 4
- Uses similar command syntax to IPv4-only NTP version 3
- Provides PKI-based and X.509 certificate-based security
- Can use multicast groups to automatically calculate the NTP hierarchy

NTP Version 4 and IPv6

NTP version 3 (NTPv3) supports only IPv4. NTP version 4 (NTPv4) is an extension of NTPv3 that supports both IPv4 and IPv6. NTPv4 is backward-compatible with NTPv3 and therefore uses similar command syntax.

NTPv4 increases security over NTPv3 by providing a security framework based on public-key cryptography and X.509 certificates. In addition, NTPv4 can use site-local IPv6 multicast groups to calculate and configure the NTP server hierarchy, providing the best accuracy and consuming the least bandwidth.



Emergency
Warning
RouterA
Notification

Configuring Log Severity Levels

- 0 – emergencies
- 1 – alerts
- 2 – critical
- 3 – errors
- 4 – warnings
- 5 – notifications
- 6 – informational
- 7 – debugging

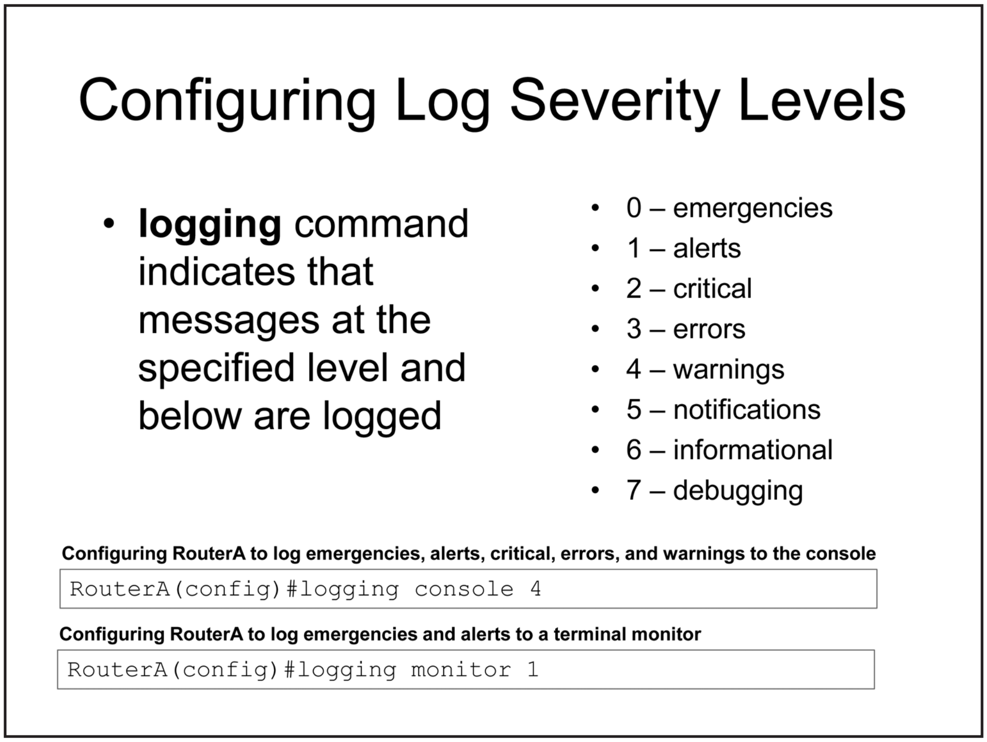
Levels 0 through 7
0 = most severe
7 = least severe

Configuring Log Severity Levels

Cisco log messages are divided into the following severity levels:

- 0 – emergencies
- 1 – alerts
- 2 – critical
- 3 – errors
- 4 – warnings
- 5 – notifications
- 6 – informational
- 7 – debugging

A lower severity level indicates a level of relative importance. For example, log messages with a severity level of 5 are considered have a higher severity level than log messages with a severity of 7.



Configuring Log Severity Levels

- **logging** command indicates that messages at the specified level and below are logged

- 0 – emergencies
- 1 – alerts
- 2 – critical
- 3 – errors
- 4 – warnings
- 5 – notifications
- 6 – informational
- 7 – debugging

Configuring RouterA to log emergencies, alerts, critical, errors, and warnings to the console

```
RouterA(config)#logging console 4
```

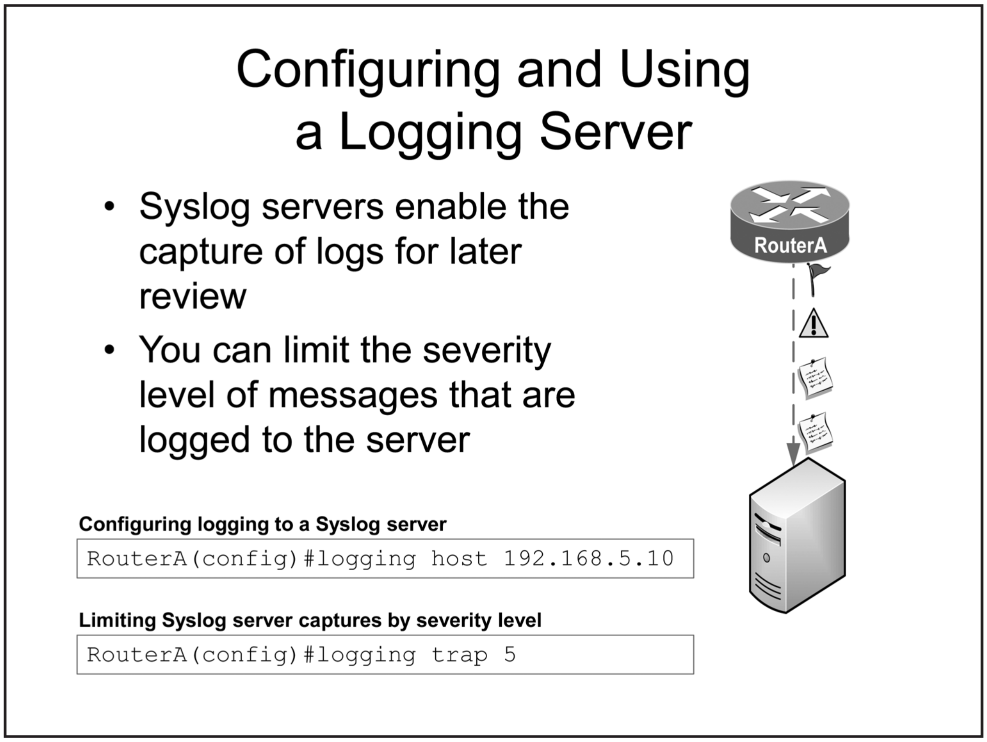
Configuring RouterA to log emergencies and alerts to a terminal monitor

```
RouterA(config)#logging monitor 1
```

By default, all messages are logged to the available logging locations on a Cisco device. However, you can specify the minimum log severity level for a particular logging location to filter extraneous logging messages.

Issuing the **logging console** *severity-level* command would specify that messages logged to the console must have a minimum severity level of *severity-level*. When the **logging console** command is issued with a *severity-level* parameter, messages with the specified severity level and all lower-numbered severity levels will be displayed by the console. For example, in the sample configuration above, the **logging console 4** command configures SwitchA to display log messages that are at levels 0 through 4: emergencies, alerts, critical, errors, and warnings.

Issuing the **logging monitor** *severity-level* command would specify that messages logged to local VTY sessions must have a minimum severity level of *severity-level*. When the **logging monitor** command is issued with a *severity-level* parameter, messages with the specified severity level and all lower-numbered severity levels will be displayed by the console. For example, in the sample configuration above, the **logging monitor 1** command configures SwitchA to display log messages that are at level 0 and level 1.



Configuring and Using a Logging Server

- Syslog servers enable the capture of logs for later review
- You can limit the severity level of messages that are logged to the server

Configuring logging to a Syslog server

```
RouterA(config)#logging host 192.168.5.10
```

Limiting Syslog server captures by severity level

```
RouterA(config)#logging trap 5
```

Configuring and Using a Logging Server

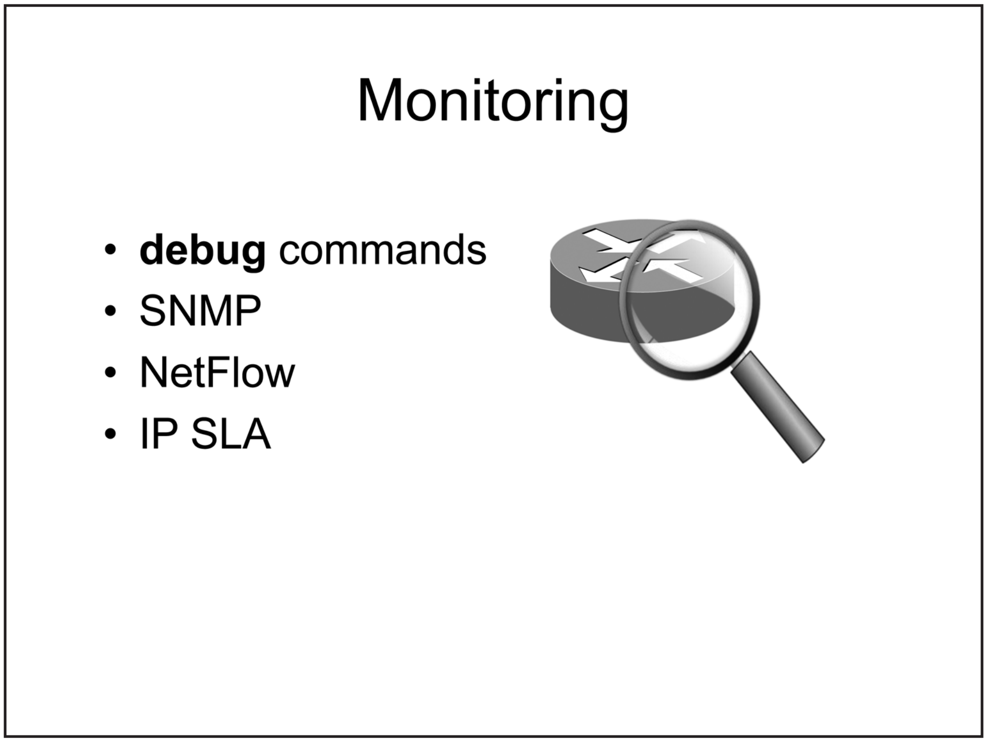
By default, message logging is enabled and sends messages to the console on Cisco routers and switches. This feature can be directly disabled by issuing the **no logging console** command. You can configure an additional destination for message logging by issuing the **logging host** *host-ip-address* command, where *host-ip-address* is the host name or IP address of a Syslog server. A Syslog server captures Syslog-formatted messages and stores them. In the sample configuration above, the **logging host 192.168.5.10** command configures SwitchA to send logging messages to the Syslog server that has been assigned the IP address of 192.168.5.10.

You can limit the severity level of messages sent to a Syslog server by issuing the **logging trap** *severity-level* command. In the sample configuration above, the **logging trap 5** command configures the switch to send to the server Syslog messages that are at levels 0 through 5: emergencies, alerts, critical, errors, warnings, and notifications.

Finally, you can verify the logging configuration of a Cisco router or switch by issuing the **show logging** command from privileged EXEC mode. The output of the **show logging** command will enable you to determine where logging occurs, such as on the console or to a Syslog server, the number of messages that have been logged, and the logging level that is configured for each logging device.

The following sample output reflects the remote logging settings in the previous example:

```
Trap logging: level notifications, 34 message lines logged
    Logging to 192.168.5.10  (udp port 514, audit disabled,
          link up),
          2 message lines logged,
              0 message lines rate-limited,
              0 message lines dropped-by-MD,
              xml disabled, sequence number disabled
              filtering disabled
        Logging Source-Interface:       VRF Name:
```

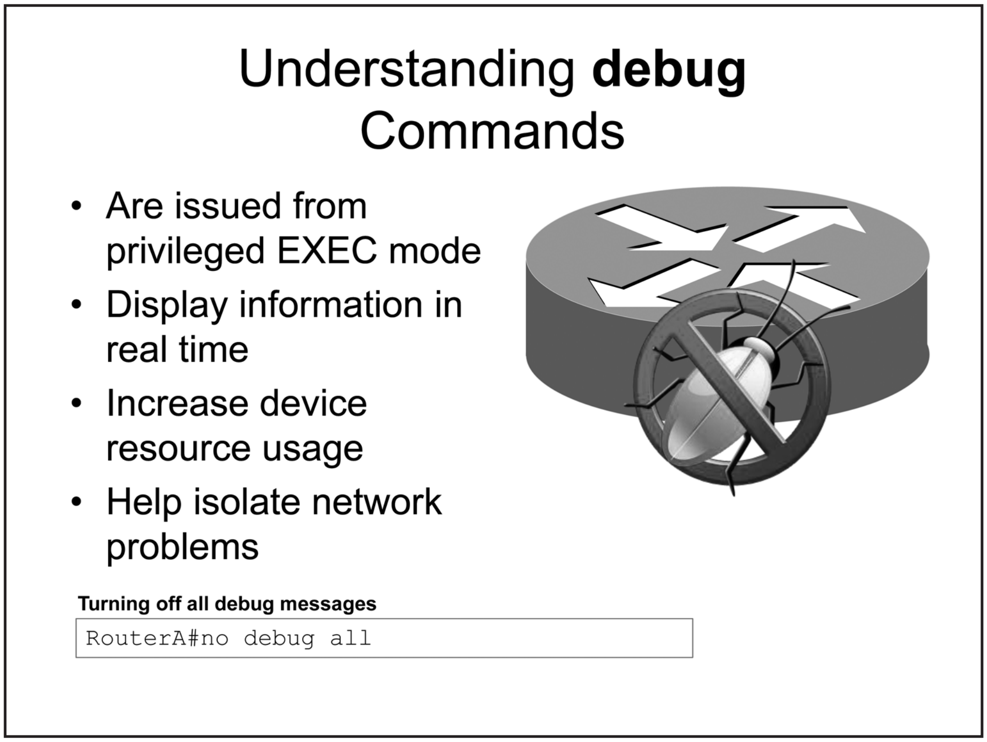


Monitoring

- **debug** commands
- SNMP
- NetFlow
- IP SLA

Monitoring

The more network devices you administer, the more difficult it is to monitor them. Fortunately, there are many tools that you can use to properly monitor and assess the devices on your network. In this section, you will learn about **debug** commands, SNMP, NetFlow, and IP SLA.



Understanding **debug** Commands

- Are issued from privileged EXEC mode
- Display information in real time
- Increase device resource usage
- Help isolate network problems

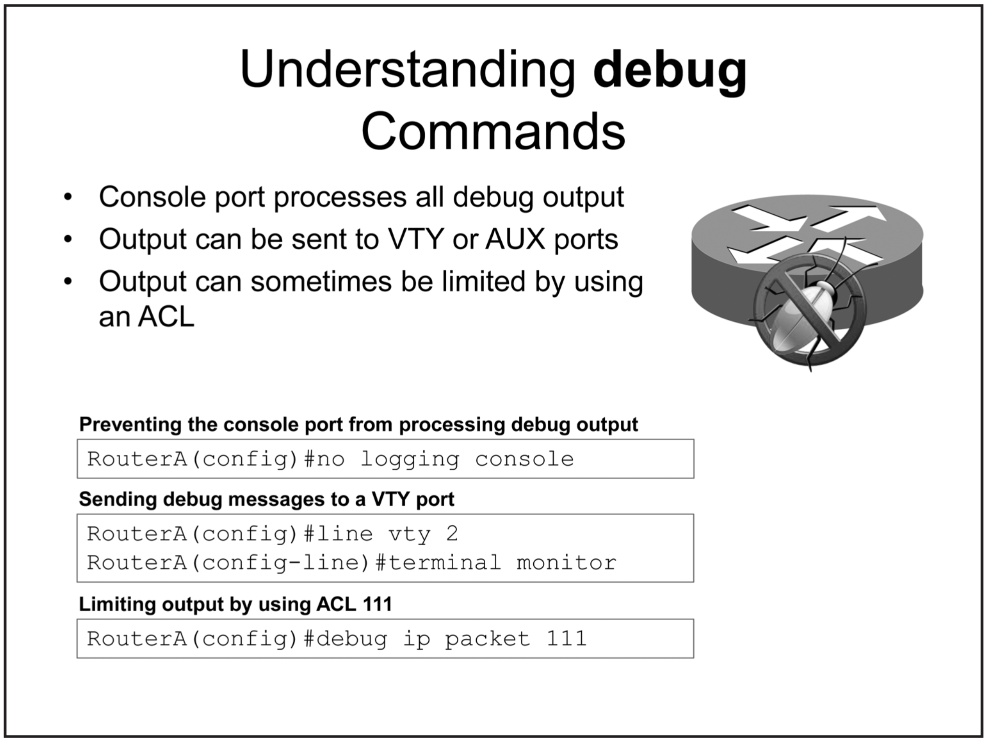
Turning off all debug messages

```
RouterA#no debug all
```

*Understanding **debug** Commands*

IOS **debug** commands enable an administrator to view traffic and information in real time, as it happens on the device. However, **debug** commands can significantly increase the use of device resources and can degrade performance. Therefore, you should use **debug** commands only to troubleshoot a problem, not to monitor normal network traffic. To see the effects of **debug** commands on a device, you can issue the **show processes** command, which displays a list of processes that are running on the device along with CPU utilization statistics.

You should typically issue **debug** commands from privileged EXEC mode. After you have gathered all the information you need from the output of a **debug** command, you can disable debugging by issuing the **no** form of the specific **debug** command or by issuing the **no debug all** command in privileged EXEC mode.



Understanding **debug** Commands

- Console port processes all debug output
- Output can be sent to VTY or AUX ports
- Output can sometimes be limited by using an ACL

Preventing the console port from processing debug output

```
RouterA(config)#no logging console
```

Sending debug messages to a VTY port

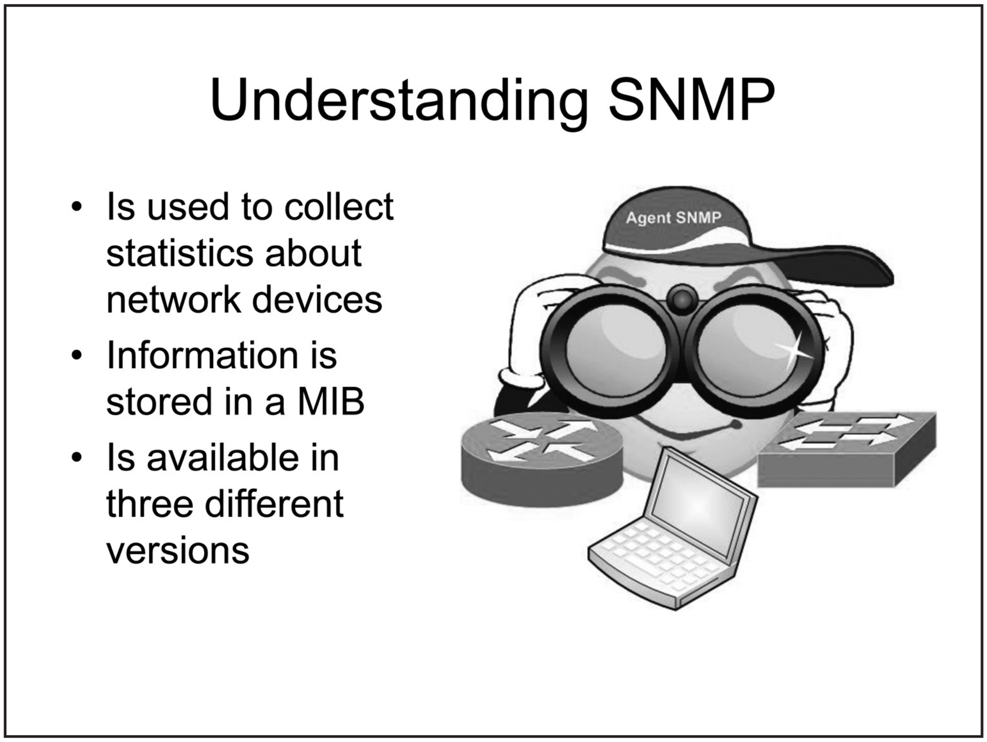
```
RouterA(config)#line vty 2
RouterA(config-line)#terminal monitor
```

Limiting output by using ACL 111

```
RouterA(config)#debug ip packet 111
```

You can take steps to minimize the effects of **debug** commands on device performance. For example, you can issue the **no logging console** command to disable the echoing of debugging output to the console. The console port processes all debug output, which increases the CPU load on the device. After you disable logging to the console, you can issue the **terminal monitor** command to display debugging output to the Telnet or SSH session you use to connect to the device.

You can also limit some debug output by using an ACL so that only packets that match the specified criteria are logged. For example, the **debug ip packet 111** command displays only packets that match ACL 111.



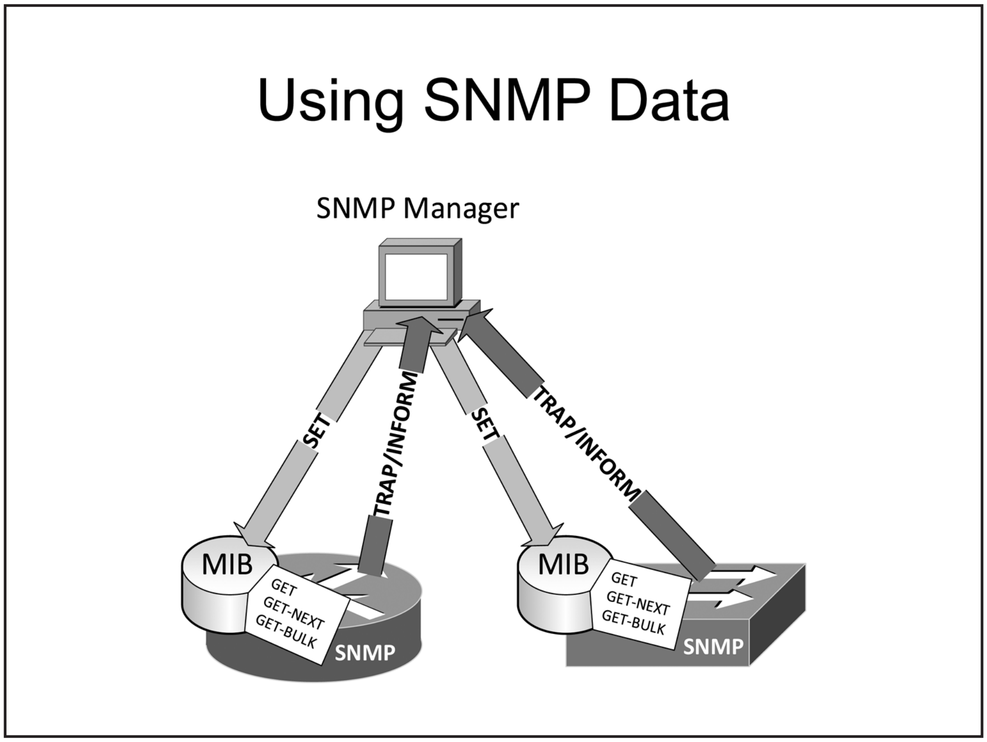
Understanding SNMP

- Is used to collect statistics about network devices
- Information is stored in a MIB
- Is available in three different versions

Understanding SNMP

SNMP is used to collect statistics about network devices. An SNMP agent reads and displays information from a hierarchical database of objects known as a management information base (MIB).

Three versions of SNMP are available to Cisco devices: SNMP version 1 (SNMPv1), SNMPv2c, and SNMPv3. This section covers the differences between these versions.



Using SNMP Data

Using SNMP Data

SNMP agents can be regularly polled over UDP by an SNMP manager, which in turn might be a component of a centralized network management system (NMS). The SNMP manager is thus responsible for collecting data from the local MIBs of SNMP agents and storing it over time. The data that the SNMP manager obtains from devices can be used to trigger NMS performance or security notifications.

There are several operations available for the SNMP agent to retrieve data from its MIB, depending on the version of SNMP that is in use. All versions of SNMP support the GET operation to retrieve data from the MIB. In addition, all versions of SNMP support the GET-NEXT operation, which the SNMP agent uses to obtain the next object from the MIB. However, only SNMPv2c and SNMPv3 support the GET-BULK operation, which is used by a management application to retrieve information in bulk from an MIB.

There are also several operations available for an SNMP manager to communicate with an SNMP agent. For example, an SNMP manager uses the SET operation to send information to an MIB from the SNMP manager. Both the TRAP operation and the INFORM operation are used by an SNMP agent to send information to an SNMP manager. A TRAP is triggered information that is sent from an SNMP agent to the SNMP manager. An INFORM is similar to a TRAP but contains an acknowledgment.

You must configure at least one **snmp-server host** command to complete an SNMP server traps configuration. Until a destination SNMP server is specified by issuing the **snmp-server host** *host-name* command, no notifications will be sent. In addition, you must configure at least one **snmp-server enable traps** command in order to configure a Cisco device to send SNMP traps to an SNMP server. When issued without additional parameters, the **snmp-server enable traps** command configures a Cisco device to send all SNMP notifications.

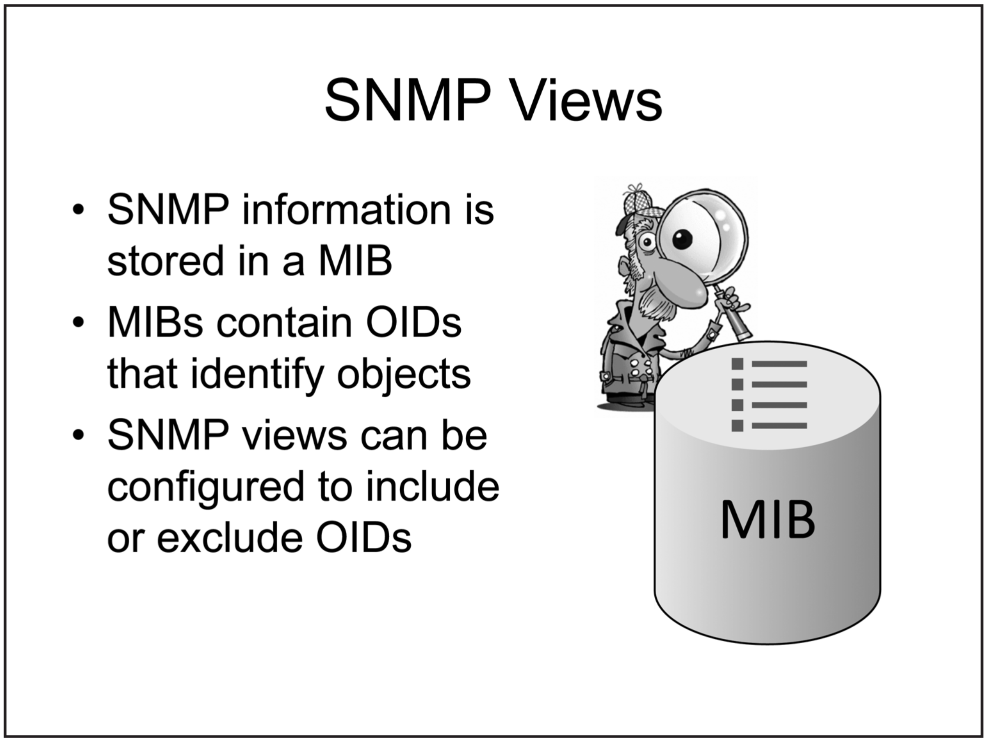
The syntax of the **snmp-server enable traps** command is **snmp-server enable traps** [*notification-type*], where *notification-type* is the type of trap or inform to enable. You can issue multiple **snmp-server enable traps** commands on a device. For example, to ensure that only critical digital signal processor (DSP) traps are sent to the SNMP server, you could issue the following commands:

```
snmp-server enable traps alarms 1
snmp-server enable traps dsp
```

The **snmp-server enable traps alarms 1** command ensures that only critical traps are sent. In addition, the **snmp-server enable traps dsp** command ensures that only DSP traps are sent. There are four alarm severity levels that can be specified:

- Critical
- Major
- Minor
- Informational

If no severity level is specified when traps are enabled, the default level is 4, which means that informational, minor, major, and critical traps are all sent.

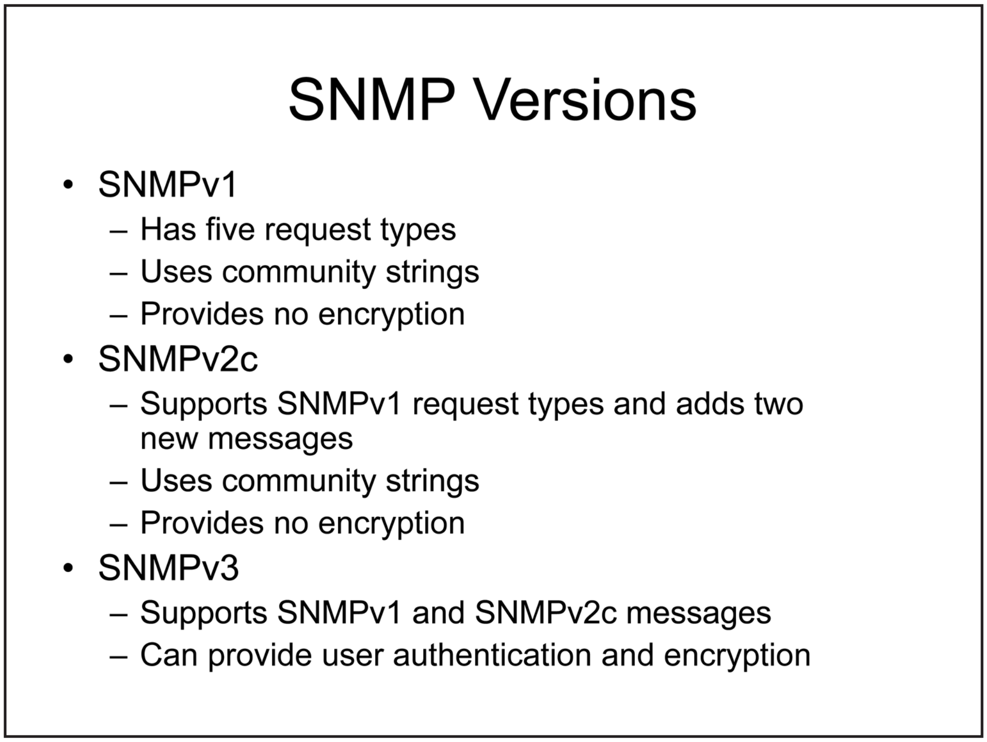


SNMP Views

- SNMP information is stored in a MIB
- MIBs contain OIDs that identify objects
- SNMP views can be configured to include or exclude OIDs

SNMP Views

SNMP information is stored in the MIB. The objects in the MIB are organized by using object IDs (OIDs), which are unique identifiers that are assigned to each object. To limit the information that can be accessed by a user, you can create an SNMP view. An SNMP view can be configured to include or exclude certain OID subtrees. However, SNMP views are supported only on SNMPv3.



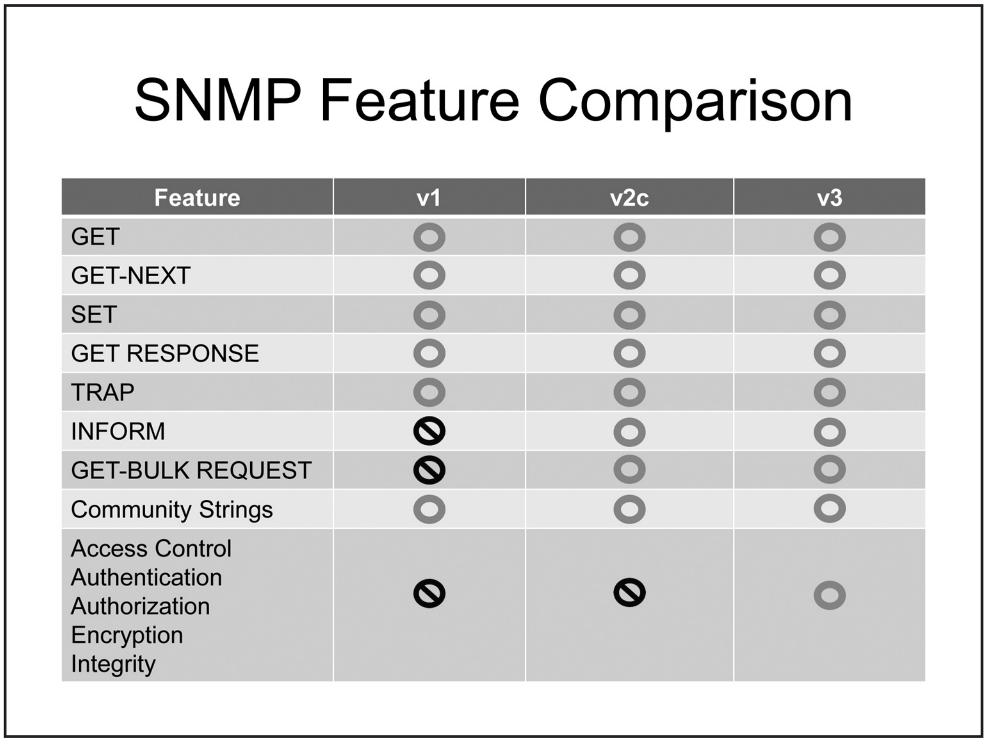
SNMP Versions

- SNMPv1
  - Has five request types
  - Uses community strings
  - Provides no encryption
- SNMPv2c
  - Supports SNMPv1 request types and adds two new messages
  - Uses community strings
  - Provides no encryption
- SNMPv3
  - Supports SNMPv1 and SNMPv2c messages
  - Can provide user authentication and encryption

SNMP Versions

SNMPv1 and SNMPv2c use community strings to provide authentication. In fact, the c in SNMPv2c stands for "community strings" to differentiate it from other attempts to add security in SNMPv2. Neither SNMPv1 nor SNMPv2c uses encryption; all data and community strings are sent in clear text. A malicious user can sniff an SNMP community string and use it to access and modify network devices. SNMPv3 is an enhancement to the SNMP protocol that uses encryption to provide confidentiality, integrity, and authentication.

SNMPv3, on the other hand, is capable of using encryption for strong authentication and confidential communication. SNMPv3 uses either MD5 or SHA to authenticate. In addition, SNMPv3 is capable of encrypting entire packets of information and verifying the integrity of information sent across the network.



SNMP Feature Comparison

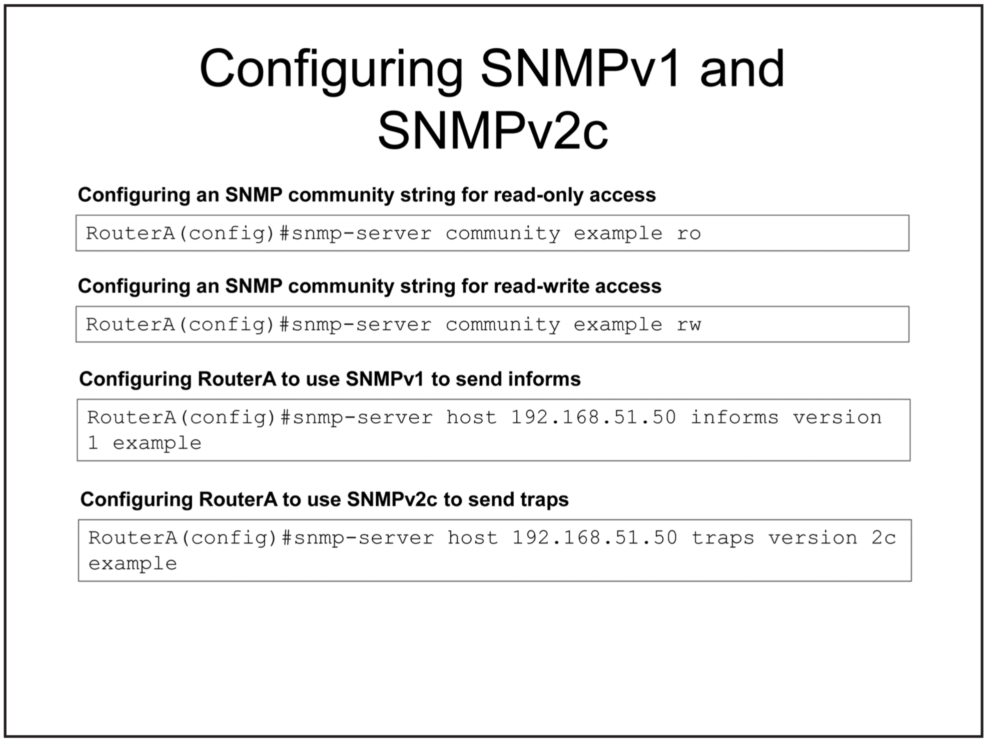
Feature	v1	v2c	v3
GET	○	○	○
GET-NEXT	○	○	○
SET	○	○	○
GET RESPONSE	○	○	○
TRAP	○	○	○
INFORM	⊘	○	○
GET-BULK REQUEST	⊘	○	○
Community Strings	○	○	○
Access Control Authentication Authorization Encryption Integrity	⊘	⊘	○

SNMP Feature Comparison

All three versions of SNMP support GET, GET-NEXT, SET, GET RESPONSE, and TRAP messages. SNMPv2c and SNMPv3 also include support for INFORM messages.

SNMPv2c and SNMPv3 both support a bulk retrieval operation known as GET-BULK. SNMPv1 is incapable of retrieving information from the MIB in bulk form.

All three versions of SNMP support community strings. SNMPv2c improved upon error handling and several other features of SNMPv1. However, only SNMPv3 provides support for access control, authentication, authorization, encryption, and message integrity.



Configuring SNMPv1 and SNMPv2c

Configuring an SNMP community string for read-only access

```
RouterA(config)#snmp-server community example ro
```

Configuring an SNMP community string for read-write access

```
RouterA(config)#snmp-server community example rw
```

Configuring RouterA to use SNMPv1 to send informs

```
RouterA(config)#snmp-server host 192.168.51.50 informs version
1 example
```

Configuring RouterA to use SNMPv2c to send traps

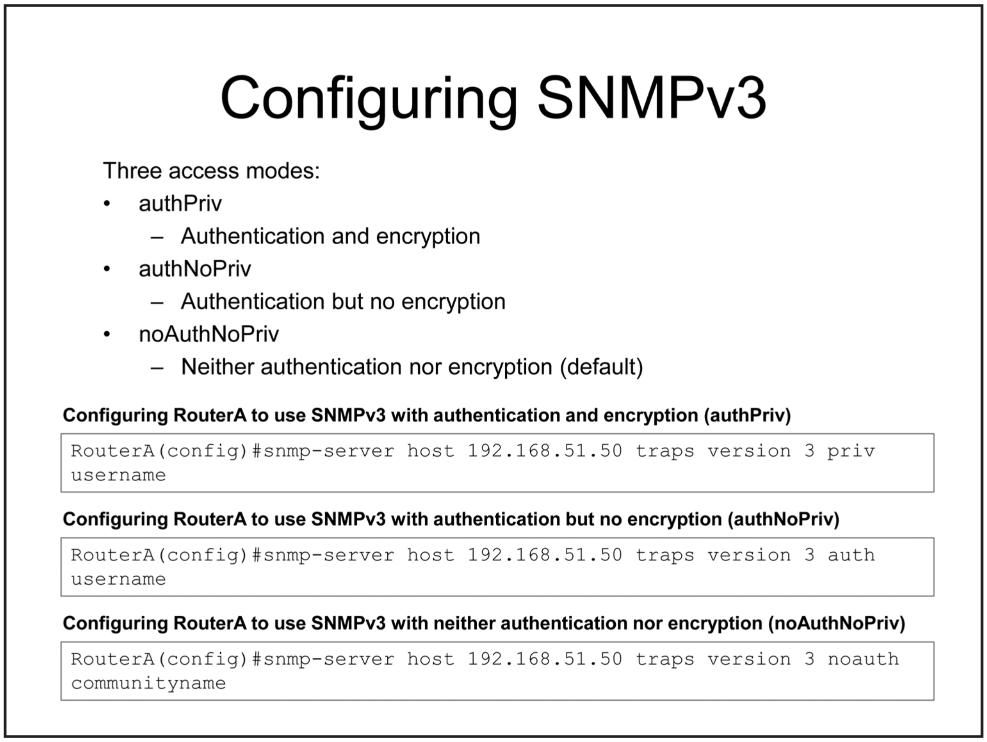
```
RouterA(config)#snmp-server host 192.168.51.50 traps version 2c
example
```

Configuring SNMPv1 and SNMPv2c

To configure SNMP access to a Cisco device, you should issue the **snmp-server community** *community-string* [**ro** | **rw**] command in global configuration mode, where *community-string* is the string of alphanumeric characters that you want to assign as the community string. The community string can be no more than 32 characters long and cannot contain the at (@) symbol. The **ro** keyword enables read-only access to the MIB. The **rw** keyword enables read-write access to the MIB.

The commands above display two ways to configure SNMP on a Cisco router. The first configures SNMP with a community string of `example` and read-only access. The second configures SNMP with a community string of `example` and read-write access. By default, all three versions of SNMP are available when SNMP is enabled on a Cisco SNMP management device. The SNMP agent can then be configured to connect to the management station by using a specific version of SNMP.

You can specify the IP address or host name of a recipient of an SNMP notification operation by issuing the **snmp-server host** {*ip-address* | *host-name*} command. By default, there is no recipient specified and this command is disabled. You can specify the type of request that is sent to the recipient by appending either the **informs** keyword or the **traps** keyword to the **snmp-server host** command. For example, the **snmp-server host 192.168.51.50 informs** command sends informs requests to the host at 192.168.51.50. You can also issue the command with the **version** keyword to specify the version of SNMP to use.



Configuring SNMPv3

Three access modes:

- authPriv
  - Authentication and encryption
- authNoPriv
  - Authentication but no encryption
- noAuthNoPriv
  - Neither authentication nor encryption (default)

Configuring RouterA to use SNMPv3 with authentication and encryption (authPriv)

```
RouterA(config)#snmp-server host 192.168.51.50 traps version 3 priv
username
```

Configuring RouterA to use SNMPv3 with authentication but no encryption (authNoPriv)

```
RouterA(config)#snmp-server host 192.168.51.50 traps version 3 auth
username
```

Configuring RouterA to use SNMPv3 with neither authentication nor encryption (noAuthNoPriv)

```
RouterA(config)#snmp-server host 192.168.51.50 traps version 3 noauth
communityname
```

Configuring SNMPv3

If you issue the **snmp-server host** command with version 3, you can also specify the **priv**, **auth**, or **noauth** keyword to configure the SNMPv3 access mode.

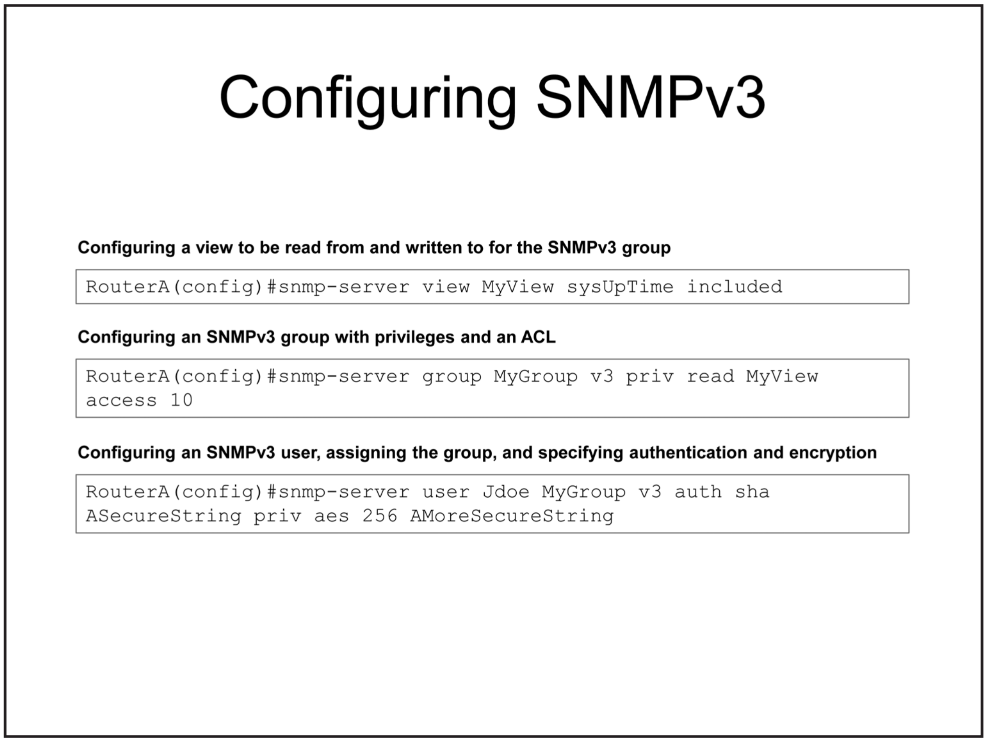
- Issuing the **priv** keyword configures SNMPv3 to use the authPriv access mode, which provides both authentication and encryption.
- Issuing the **auth** keyword configures SNMPv3 to use the authNoPriv access mode, which provides authentication but not encryption.
- Issuing the **noauth** keyword configures SNMPv3 to use the noAuthNoPriv access mode, which provides neither authentication nor encryption. This is equivalent to the access mode used by SNMPv1 and SNMPv2c.

The authPriv access mode authenticates by matching an MD5 or SHA hash of the user name. The authentication process is also encrypted by using either Data Encryption Standard (DES), Triple DES (3DES), or Advanced Encryption Standard (AES). The authPriv security level is the only SNMPv3 security level that can encrypt the authentication process, and it does so by using Cipher Block Chaining Data Encryption Standard (CBC-DES).

The authNoPriv access mode authenticates by matching Hash-based Message Authentication Code-SHA (HMAC-SHA) or HMAC-MD5 authentication strings. However, the authNoPriv security level does not provide encryption.

The noAuthNoPriv access mode authenticates by matching a user name sent as clear text. SNMPv1 and SNMPv2c match community strings instead of user names.

In addition to the security level, you can configure the **snmp-server host** command with a specific community string to use, UDP port on which to operate, and notification type to send.



Configuring SNMPv3

Configuring a view to be read from and written to for the SNMPv3 group

```
RouterA(config)#snmp-server view MyView sysUpTime included
```

Configuring an SNMPv3 group with privileges and an ACL

```
RouterA(config)#snmp-server group MyGroup v3 priv read MyView
access 10
```

Configuring an SNMPv3 user, assigning the group, and specifying authentication and encryption

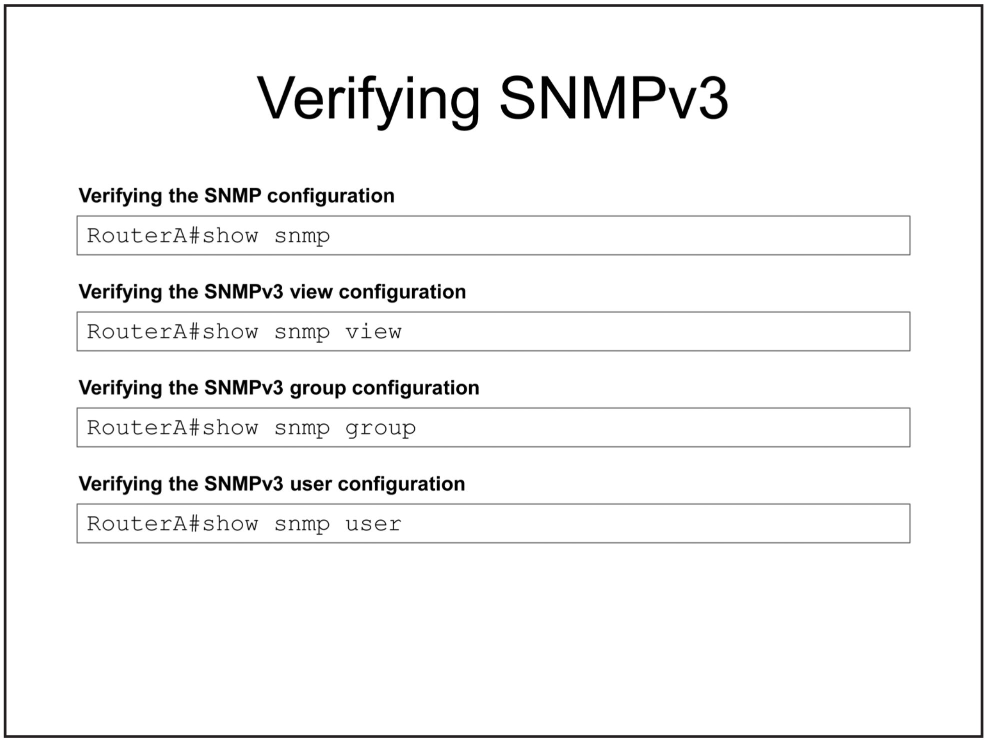
```
RouterA(config)#snmp-server user Jdoe MyGroup v3 auth sha
ASecureString priv aes 256 AMoreSecureString
```

To configure a view, you should issue the **snmp-server view** command. The syntax for the **snmp-server view** command is **snmp-server view** *view-name oid-tree* {**included** | **excluded**}. For example, the **snmp-server view MyView sysUpTime included** command creates or modifies an SNMP view named MyView and includes the sysUpTime OID subtree.

After you configure a view, you can configure an SNMPv3 group and authorize the view for the group by issuing the **snmp-server group** command. The syntax for the **snmp-server group** command is **snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**context** *context-name*] [**read** *read-view*] [**write** *write-view*] [**notify** *notify-view*] **access** [**ipv6** *named-access-list*] [*acl-number* | *acl-name*]]. For example, the **snmp-server group MyGroup v3 priv read MyView access 10** command creates an SNMP group named MyGroup, configures it for SNMPv3, enables authentication and encryption, provides read access to MyView, and restricts access to packets that match standard ACL 10.

You can then configure an SNMPv3 user and assign the user to a group by issuing the snmp-server user command. The syntax for the **snmp-server user** command is **snmp-server user** *user-name group-name* [**remote** *host* [**udp-port** *port*] [**vrf** *vrf-name*] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** [**ipv6** *named-access-list*] [**priv** {**des** | **3des** | **aes** {**128** | **192** | **256**}} *priv-password*] {*acl-number* | *acl-name*}]. For example, the **snmp-server user Jdoe MyGroup v3 auth sha ASecureString priv aes 256 AMoreSecureString** command creates a user named Jdoe, assigns Jdoe to MyGroup, configures it for SNMPv3, enables SHA authentication with the password ASecureString, and enables 256-bit AES encryption with the password AMoreSecureString.

Issuing the previous commands would provide Jdoe with read access to the sysUpTime subtree. Assigning subtrees to views, assigning views to groups, and assigning groups to users seems like a lot of work, but it provides a great deal of flexibility while still maintaining scalability. After all, if you have included multiple OID subtrees in a view and assigned multiple views to a group, you would not want to have to issue dozens of commands each time you create a new user. Instead, the next time you need to assign the same view or views to a new user, you can simply assign the new user to MyGroup.



Verifying SNMPv3

Verifying the SNMP configuration

```
RouterA#show snmp
```

Verifying the SNMPv3 view configuration

```
RouterA#show snmp view
```

Verifying the SNMPv3 group configuration

```
RouterA#show snmp group
```

Verifying the SNMPv3 user configuration

```
RouterA#show snmp user
```

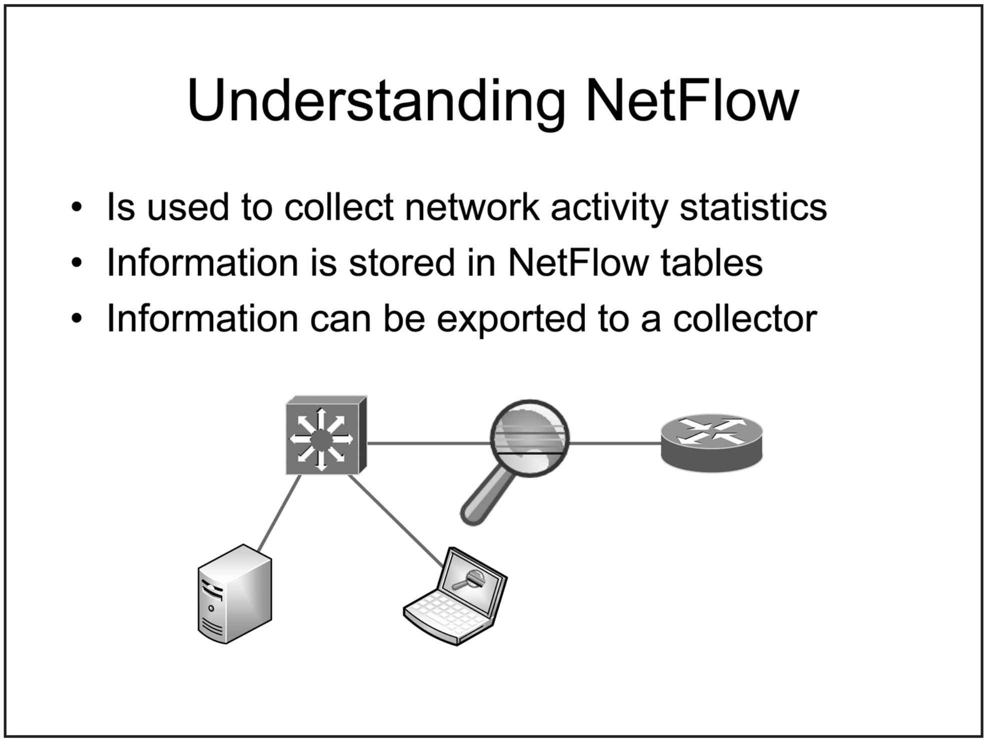
Verifying SNMPv3

Most SNMP configuration commands have an equivalent **show** command that can be used to verify the configuration. The **show snmp** command without additional keywords displays SNMP counter information, such as the number of requests and responses, in addition to the number of errors.

The **show snmp view** command displays the view names along with the corresponding included and excluded OID subtrees. It also displays whether the view is stored in volatile, nonvolatile, or permanent storage, as well as whether the view is active or nonactive.

The **show snmp group** command displays each SNMP group name, the corresponding SNMP version, and the associated read, write, and notify views.

The **show snmp user** command displays the associated group name, the authentication protocol, the encryption protocol, and active ACL for each SNMP user. It also displays whether the user information is stored in volatile, nonvolatile, or permanent storage.



Understanding NetFlow

- Is used to collect network activity statistics
- Information is stored in NetFlow tables
- Information can be exported to a collector

Understanding NetFlow

NetFlow is a Cisco feature that you can use to capture statistics about network traffic flows that pass through many Cisco routers and Layer 3–capable switches. Although Cisco considers a series of packets a *flow* if they share, at a minimum, the same source and destination IP addresses, a flow is defined as a series of packets that share the following characteristics:

- Source IP address
- Destination IP address
- Protocol number
- Source protocol port
- Destination protocol port
- Type of Service (ToS) bits
- Associated interface

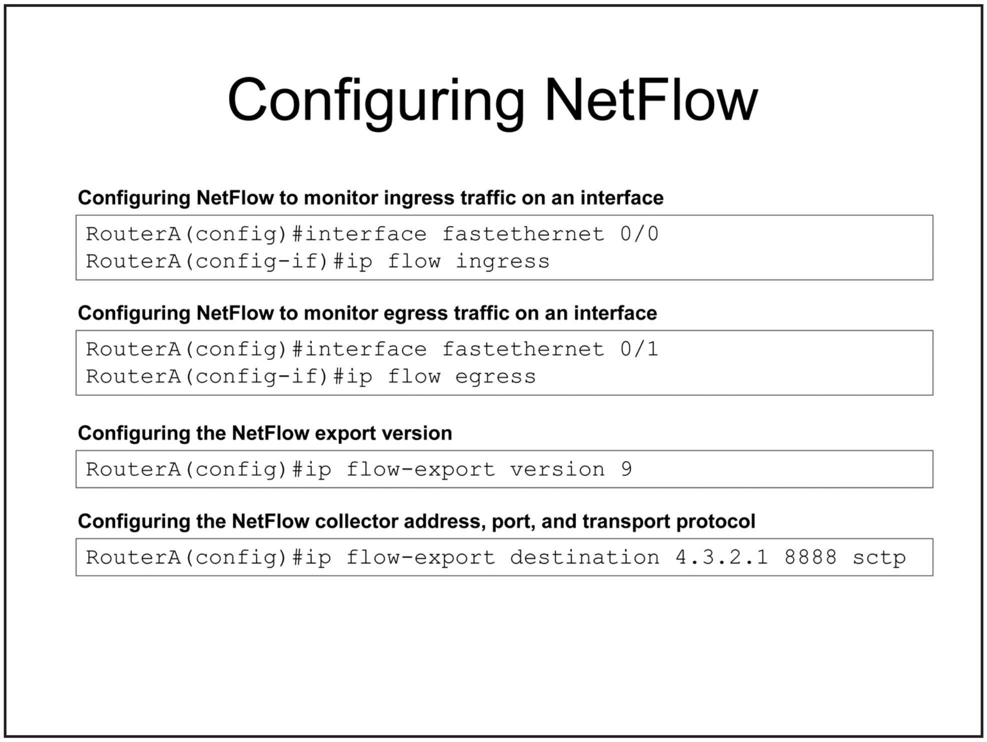
By default, the data gathered by NetFlow is stored locally in dedicated NetFlow tables on each configured device. You can access the information stored in the NetFlow tables of a device by issuing the appropriate NetFlow-related **show** commands from privileged EXEC mode. Alternatively, you can configure the device to export NetFlow statistics to a central location, which is referred to as a NetFlow collector.

Each record in the NetFlow table contains a considerable amount of information about any given flow. In addition to the data that defines the flow, each record can also include the following information:

- Next-hop router address
- Input and output interface numbers

- Number of packets transmitted
- Number of bytes transmitted
- Time stamp of the first and last packets
- Source and destination autonomous system numbers (ASNs)
- Source and destination subnet masks
- TCP flags

Because NetFlow data is collected over time, it is particularly suited for accounting, billing, and security applications.



Configuring NetFlow

Configuring NetFlow to monitor ingress traffic on an interface

```
RouterA(config)#interface fastethernet 0/0
RouterA(config-if)#ip flow ingress
```

Configuring NetFlow to monitor egress traffic on an interface

```
RouterA(config)#interface fastethernet 0/1
RouterA(config-if)#ip flow egress
```

Configuring the NetFlow export version

```
RouterA(config)#ip flow-export version 9
```

Configuring the NetFlow collector address, port, and transport protocol

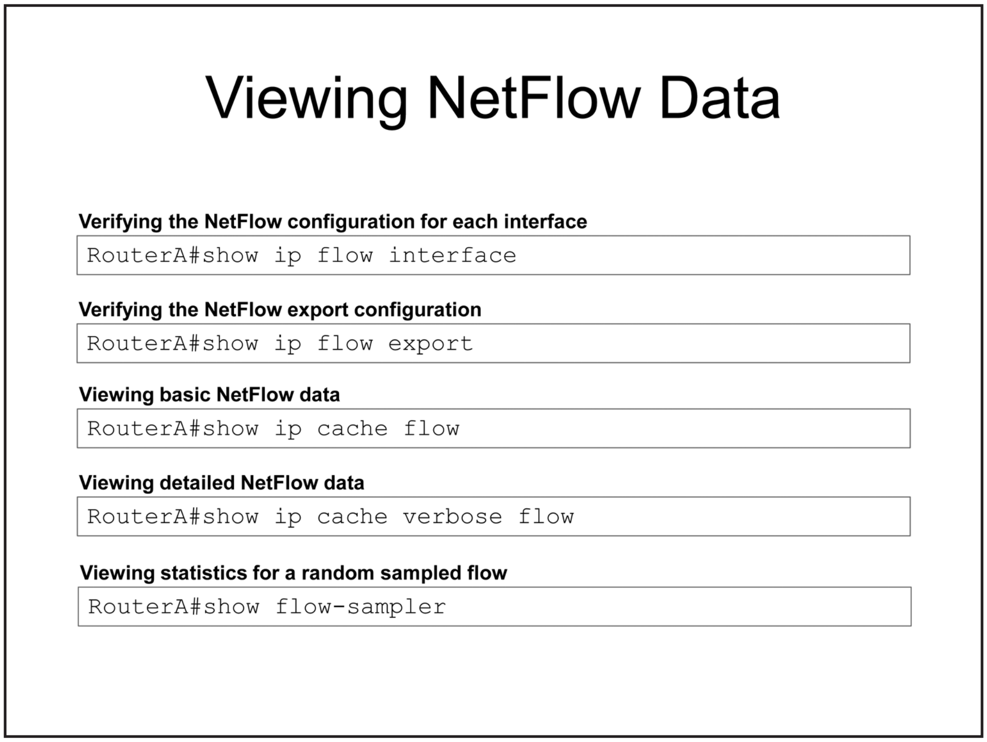
```
RouterA(config)#ip flow-export destination 4.3.2.1 8888 sctp
```

Configuring NetFlow

You can configure NetFlow to monitor either ingress or egress traffic on an interface. For example, you can issue the **ip flow ingress** command from interface configuration mode to enable NetFlow on a particular interface. The syntax of the **ip flow** command is **ip flow** {**egress** | **ingress**}, where the **ingress** keyword enables NetFlow for inbound traffic on the interface and the **egress** keyword enables NetFlow for outbound traffic on the interface.

Although NetFlow statistics are stored locally in NetFlow tables, you can export NetFlow data to an external device by using the **ip flow-export** command. You should first issue the **ip flow-export version** command to specify the record format for the exported NetFlow data. By default, NetFlow data is exported using version 1, but Cisco recommends that you change the export version to the highest version supported by your NetFlow collector. Most NetFlow devices support export versions 1, 5, and 9. For example, you can issue the **ip flow-export version 9** command to change the NetFlow export format to version 9.

Once you have specified a NetFlow export version, you should issue the **ip flow-export destination** command to specify the IP address and port number of the NetFlow collector. NetFlow records are exported as UDP datagrams by default, but some platforms support Stream Control Transmission Protocol (SCTP) as an alternate transport protocol. You can use the **sctp** keyword with the **ip flow-export destination** command to specify that SCTP should be used instead of UDP to transmit NetFlow data. For example, you could issue the **ip flow-export destination 4.3.2.1 8888 sctp** command to specify a NetFlow collector with an IP address of 4.3.2.1 that is listening for NetFlow data on SCTP port 8888.



Viewing NetFlow Data

Verifying the NetFlow configuration for each interface

```
RouterA#show ip flow interface
```

Verifying the NetFlow export configuration

```
RouterA#show ip flow export
```

Viewing basic NetFlow data

```
RouterA#show ip cache flow
```

Viewing detailed NetFlow data

```
RouterA#show ip cache verbose flow
```

Viewing statistics for a random sampled flow

```
RouterA#show flow-sampler
```

Viewing NetFlow Data

You can issue the **show ip flow interface** command to verify the basic NetFlow configuration for all interfaces on the device. In the sample output below, you can see that NetFlow is configured to monitor inbound flows on the FastEthernet 0/0 interface and outbound flows on the FastEthernet 0/1 interface:

```
RouterA#show ip flow interface
FastEthernet0/0
  ip flow ingress
FastEthernet0/1
  ip flow egress
```

If the device has been configured to export NetFlow data, you can issue the **show ip flow export** command to verify the NetFlow export format version and the IP address and port numbers of any configured NetFlow collectors. In the sample output below, you can see that NetFlow data is exported using the version 9 format and that two collectors have been configured. One collector has an IP address 1.2.3.4 and is listening on UDP port 9999, whereas the other collector has an IP address of 4.3.2.1 and is listening on SCTP port 8888:

```
RouterA#show ip flow export
Flow export v9 is enabled for main cache
  Export source and destination details :
  VRF ID : Default
    Destination(1)  1.2.3.4 (9999)
    Destination(2)  4.3.2.1 (8888) via SCTP
  Version 9 flow records
  418 flows exported in 4534 udp datagrams
  325 flows exported in 1864 sctp messages
  0 flows failed due to lack of export packet
  0 export packets were sent up to process level
  0 export packets were dropped due to no fib
  0 export packets were dropped due to adjacency issues
  0 export packets were dropped due to fragmentation failures
  0 export packets were dropped due to encapsulation fixup failures
```

You can issue the **show ip cache flow** command to view basic NetFlow data. The command output displays a variety of statistics including the number of flows for each protocol, the source and destination IP addresses for each flow, and the number of packets transmitted in each flow. In the sample output below, 39 packets have been transmitted between the 10.10.10.2 and 192.168.1.2 IP addresses:

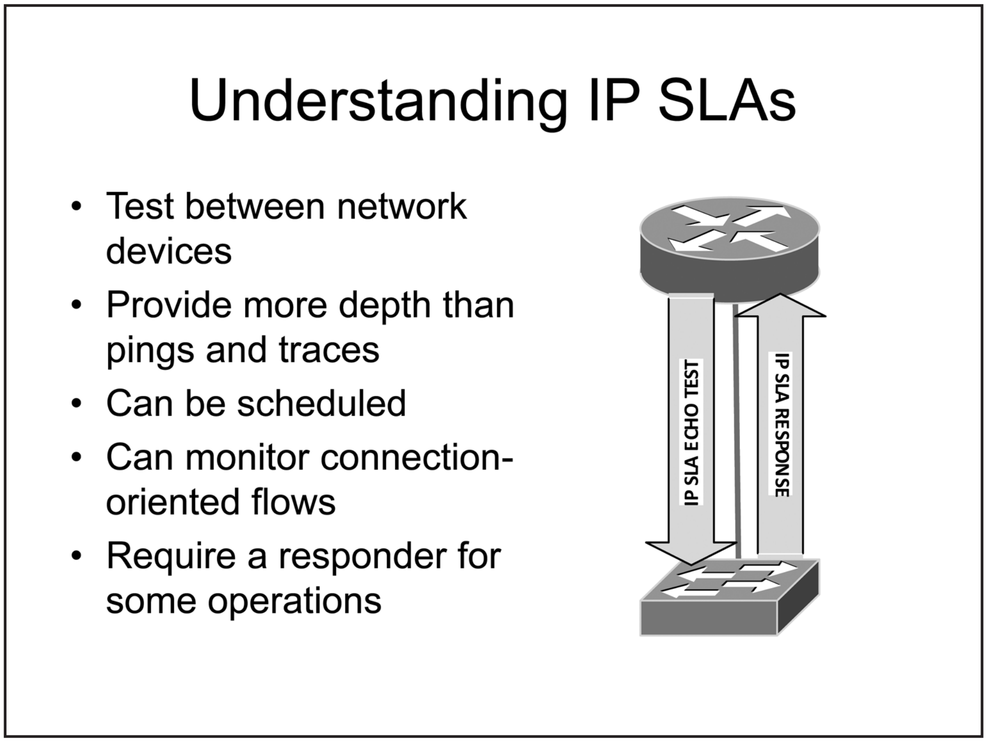
```
RouterA#show ip cache flow
IP packet size distribution (1103746 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .249 .694 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000
    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .000 .027 .000 .027 .000 .000 .000 .000 .000 .000
IP Flow Switching Cache, 278544 bytes
  35 active, 4061 inactive, 980 added
  2921778 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 21640 bytes
  0 active, 1024 inactive, 0 added, 0 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added
  last clearing of statistics never
Protocol Total    Flows   Packets Bytes  Packets Active(Sec) Idle(Sec)
-------- Flows     /Sec     /Flow  /Pkt     /Sec     /Flow     /Flow
TCP-WWW    83      0.0      1321    40      1.5    1200.1       0.8
TCP-NTP   127      0.0      1203    40      0.6    1200.1       0.7
TCP-other 337      0.0      1220    40      4.7    1201.4       0.8
UDP-TFTP   17      0.0      1213    28      0.5    1199.4       1.0
UDP-other 138      0.0      1117    28      2.1    1199.5       0.9
ICMP      125      0.0      1133   418      2.1    1199.4       0.8
Total:    915      0.0      1166    91     22.4    1799.6       0.8
SrcIf SrcIPaddress    DstIf         DstIPaddress    Pr SrcP DstP  Pkts
Fa0/0 10.10.10.2        Fa0/1       192.168.1.2     01 0000 0C01    39
Fa0/0 10.10.20.3        Fa1/1       192.168.2.5     11 0043 0043    18
Fa0/0 10.10.30.5        Fa1/0       192.168.4.7     11 0045 0045    42
```

For slightly more detailed NetFlow data, you can issue the **show ip cache verbose flow** command. The command output includes all of the same flow summary and protocol information; however, the detailed flow information is changed to include additional information such as the next-hop IP address, ToS flags, and ASNs:

```
RouterA#show ip cache verbose flow
IP packet size distribution (1103746 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
<output omitted>
SrcIf SrcIPaddress    DstIf          DstIPaddress    Pr TOS Flgs  Pkts
Port Msk AS           Port Msk AS    NextHop              B/Pk  Active
Fa0/0 10.10.10.2        Fa0/1      192.168.1.2   01 00  10     799
0000 /0  0            0C01 /0  0     0.0.0.0              28  1258.1
Fa0/0 10.10.20.3        Fa1/1      192.168.2.5      11 00  10     799
0043 /0  0            0043 /0  0     0.0.0.0              28  1258.0
Fa0/0 10.10.30.5        Fa1/0     192.168.4.7     11 00  10     799
0045 /0  0            0045 /0  0     0.0.0.0              28  1258.0
```

You can issue the **show flow-sampler** command to display statistical information to determine how many packets have been matched for a random sampling of packets. The following command output displays data from a NetFlow sampler named BSN, which has an ID of 1 and has matched five packets out of 100 in random sampling mode:

```
Sampler : BSN, id : 1, packets matched : 5, mode : random sampling mode
 sampling interval is : 100
```



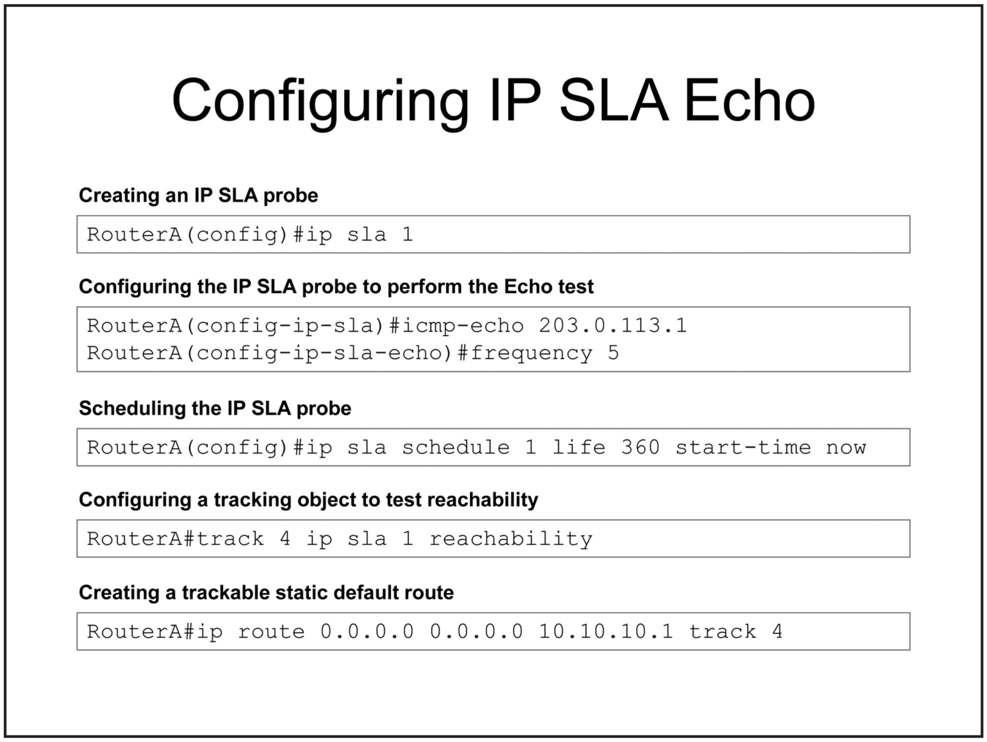
Understanding IP SLAs

- Test between network devices
- Provide more depth than pings and traces
- Can be scheduled
- Can monitor connection-oriented flows
- Require a responder for some operations

Understanding IP SLAs

IP SLA operations are a suite of tools that enable an administrator to analyze and troubleshoot IP networks. IP SLAs are more robust and can provide more detailed information about a problem than a standard ping or trace can provide. For example, the IP SLA Internet Control Message Protocol (ICMP) Echo test can be used to send Echo requests to remote devices and receive Echo replies to test availability, just as the **ping** command does. However, IP SLAs also support a variety of other tests that can measure network latency and response time, among other things. For example, the IP SLA ICMP round-trip latency operation can be used to determine whether end-to-end delays or slowness are occurring on Voice over IP (VoIP) calls.

IP SLAs can be used to perform a number of different tests, can be configured to run at scheduled intervals, and can be configured to monitor connection-oriented flows. In addition, you can gather more data and perform more complex operations by enabling an IP SLA responder.



Configuring IP SLA Echo

Creating an IP SLA probe

```
RouterA(config)#ip sla 1
```

Configuring the IP SLA probe to perform the Echo test

```
RouterA(config-ip-sla)#icmp-echo 203.0.113.1
RouterA(config-ip-sla-echo)#frequency 5
```

Scheduling the IP SLA probe

```
RouterA(config)#ip sla schedule 1 life 360 start-time now
```

Configuring a tracking object to test reachability

```
RouterA#track 4 ip sla 1 reachability
```

Creating a trackable static default route

```
RouterA#ip route 0.0.0.0 0.0.0.0 10.10.10.1 track 4
```

Configuring IP SLA Echo

In order to configure the IP SLA Echo operation, you must first create an IP SLA probe by issuing the **ip sla** *operation-number* command, where *operation-number* is an integer value that you want to use to identify the probe. Issuing this command also places the device into IP SLA configuration mode.

In IP SLA configuration mode, you can configure a variety of operations for the probe to perform. To specifically configure the IP SLA Echo operation, you should issue the **icmp-echo** *destination-ip-address* command, where *destination-ip-address* is the IP address of the remote device for which you want to test availability. You can also issue the **frequency** *seconds* command to define a rate in seconds at which the Echo operation will repeat.

After you have configured the probe, you must schedule it to run by issuing the **ip sla schedule** *operation-number* [**life** {**forever** | *seconds*}] [*start-time* {*hh*:*mm*[:*ss*] [*monthday* | *daymonth*] | **pending** | **now** | **after** *hh*:*mm*:*ss*}] [**ageout** *seconds*] [**recurring**] command, where operation-number is the number that you assigned the probe when you created it, from global configuration mode. If you issue the **ip sla schedule** *operation-number* command without parameters, the probe is enabled but placed into the pending state. You can adjust how the schedule behaves by issuing the optional keywords as follows:

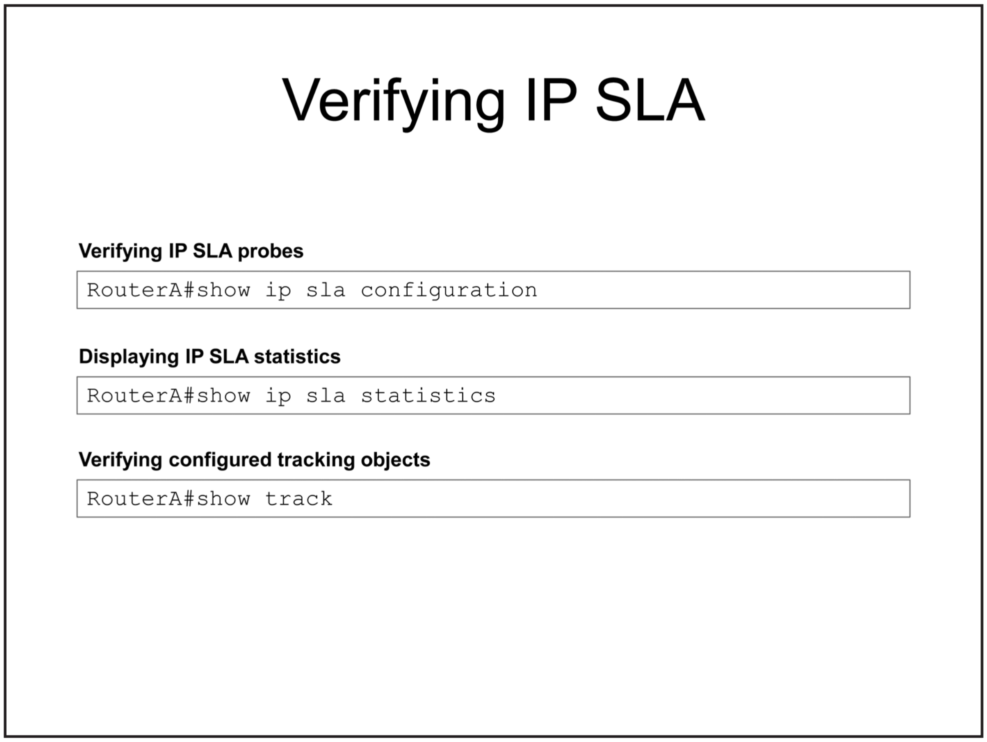
- **life** – If omitted, the default value is 3,600 seconds, or one hour. You can specify an alternate number of seconds or issue the keyword **forever** so that the SLA continues to run without end once it is started.
- **start-time** – If omitted, the start time defaults to **pending**. However, you can specify that the SLA process start at a given time, such as 11:30:30, or after a given time by issuing the **after** keyword prior to the time you specify. You can also specify a specific month and day, such as **January 1** or **1**

January, on which to start the SLA. Finally, you can specify that the SLA start immediately by issuing the **now** keyword.

- **ageout** – The number of seconds that should elapse before removing an operation from memory when it is not collecting information. By default, the ageout value is 0, which configures the schedule to never remove the operation from memory.
- **recurring** – When a start time and life have been specified, issuing this keyword ensures that the SLA starts and runs for the same duration every day.

To use the probe to test the reachability of a route, you must create a tracking object and map it to the probe by issuing the **track** *object-number* **ip sla** *operation-number* **reachability** command from global configuration mode. For example, the **track 4 ip sla 1 reachability** command creates tracking object 4 and links it to IP SLA probe 1.

You can then configure a tracked static route that will be used if the tracked object defined in the probe is reachable. To do so, add the **track** *object-number* keywords to the end of the static route. For example, the **ip route 0.0.0.0 0.0.0.0 10.10.10.1 track 4** command configures a router to use a static default route to 10.10.10.1 if tracking object 4 is reachable.



Verifying IP SLA

Verifying IP SLA probes

```
RouterA#show ip sla configuration
```

Displaying IP SLA statistics

```
RouterA#show ip sla statistics
```

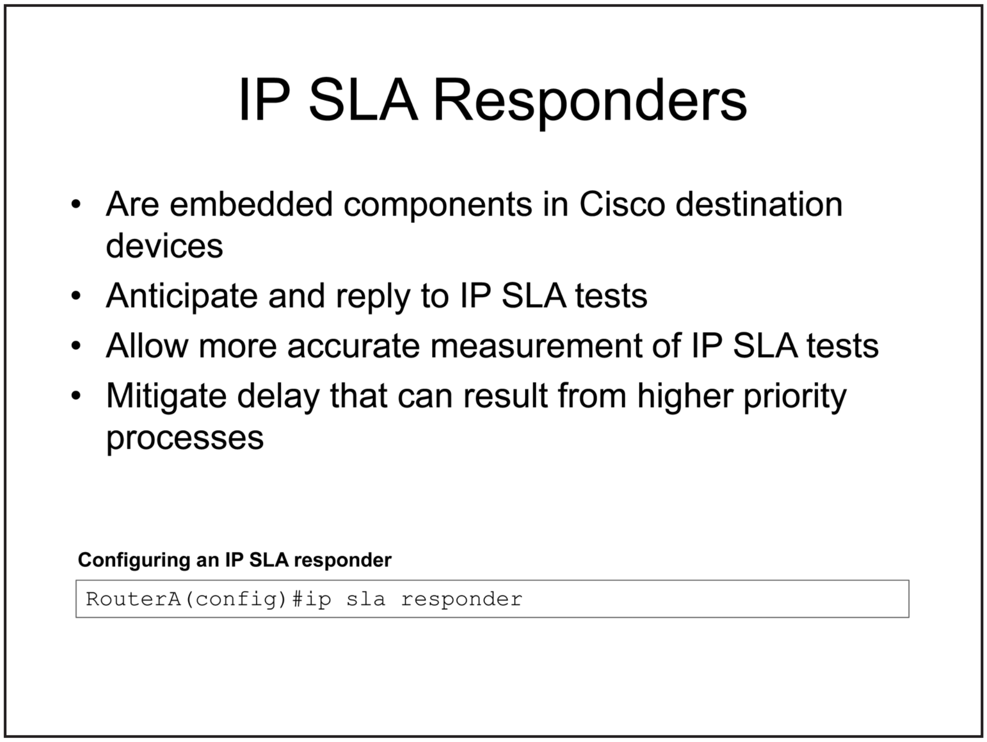
Verifying configured tracking objects

```
RouterA#show track
```

Verifying IP SLA

After IP SLA has been configured, you can display all the values with which a probe has been configured by issuing the **show ip sla configuration** command from privileged EXEC mode. You can examine the operational status and statistics associated with an IP SLA probe by issuing the **show ip sla statistics** [**aggregated**] command from privileged EXEC mode. The optional **aggregated** keyword causes the command to display aggregated statistical errors and distribution information.

You can verify configured tracking objects by issuing the **show track** command. The output of the **show track** command displays the mapping between the tracking object and the probe as well as the status of the tracking object.



IP SLA Responders

- Are embedded components in Cisco destination devices
- Anticipate and reply to IP SLA tests
- Allow more accurate measurement of IP SLA tests
- Mitigate delay that can result from higher priority processes

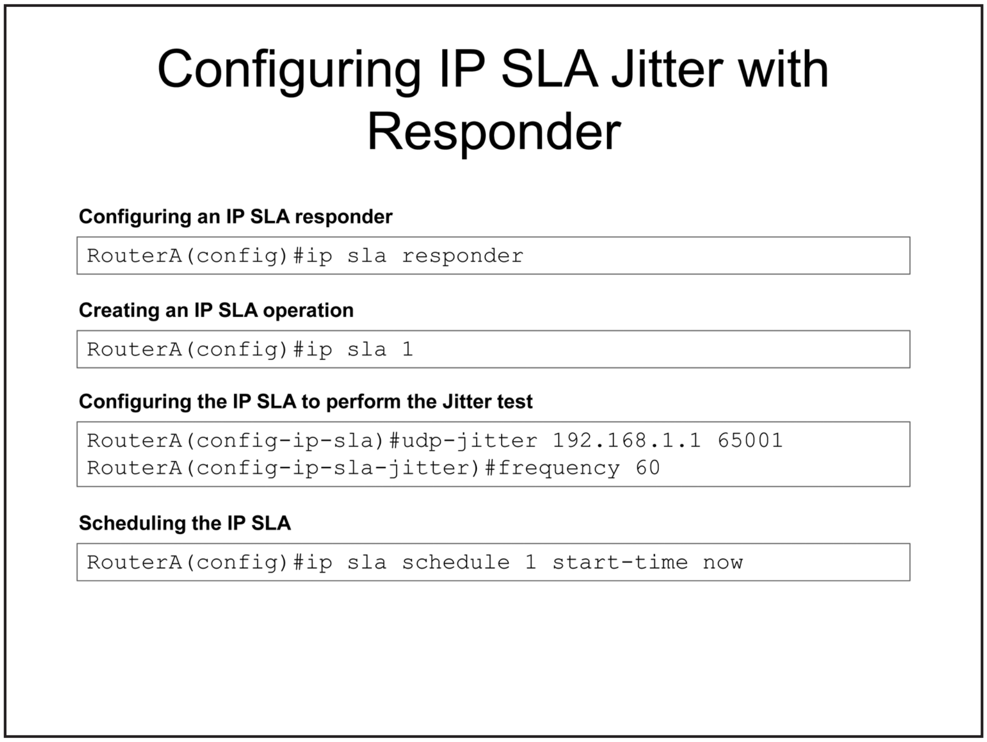
Configuring an IP SLA responder

```
RouterA(config)#ip sla responder
```

IP SLA Responders

To gather more robust data, the destination router for an IP SLA operation should have an IP SLA responder enabled. The IP SLA responder is an embedded component that provides accurate measurements without requiring dedicated probes. Although an IP SLA responder is not required for normal IP SLA operation, it can account for packet processing time on a destination router by adding timestamps to a packet when it enters or leaves a network interface. The IP SLA operations source router then considers the additional packet timestamps when reporting data about round-trip times on a network.

To enable an IP SLA responder for general operations, you should issue the **ip sla responder** command in global configuration mode on the destination router. To enable an IP SLA responder for TCP Connect operations, you should issue the **ip sla responder tcp-connect ipaddress** *ip-address* **port** *port-number* command. To enable an IP SLA responder for UDP Echo or Jitter operations, you should issue the **ip sla responder udp-echo ipaddress** *ip-address* **port** *port-number* command.



Configuring IP SLA Jitter with Responder

Configuring an IP SLA responder

```
RouterA(config)#ip sla responder
```

Creating an IP SLA operation

```
RouterA(config)#ip sla 1
```

Configuring the IP SLA to perform the Jitter test

```
RouterA(config-ip-sla)#udp-jitter 192.168.1.1 65001
RouterA(config-ip-sla-jitter)#frequency 60
```

Scheduling the IP SLA

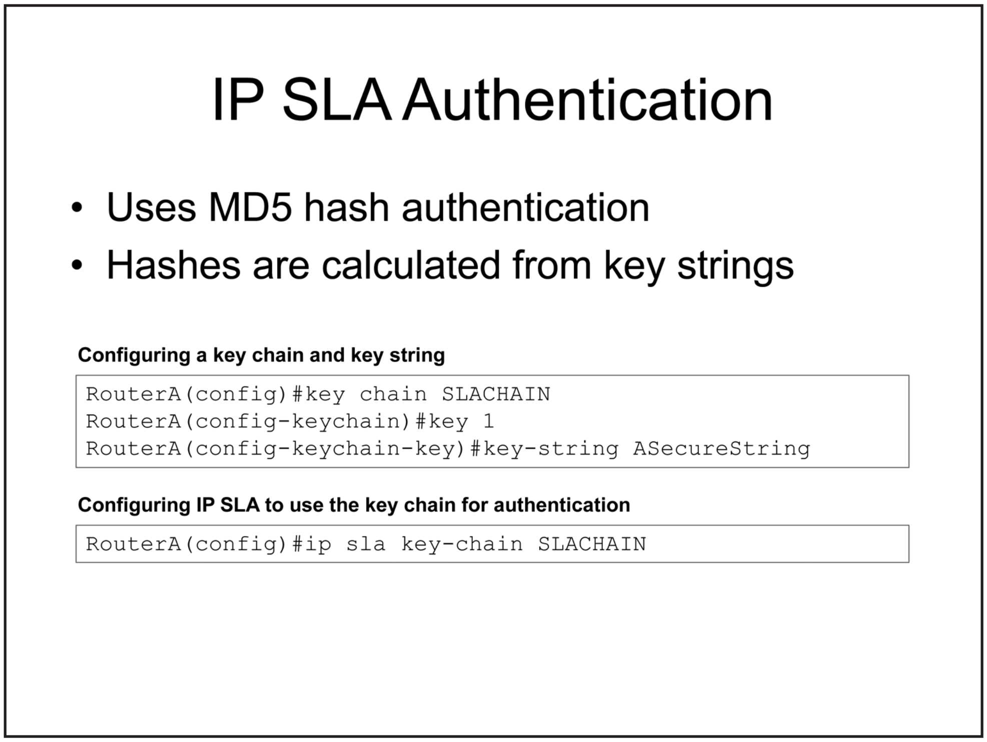
```
RouterA(config)#ip sla schedule 1 start-time now
```

Configuring IP SLA Jitter with Responder

Before you configure an IP SLA Jitter operation on a source device, you must enable the IP SLA Responder on the target device. To do so, you should issue the **ip sla responder** command or the **ip sla responder udp-echo ipaddress** *ip-address* **port** *port-number* command.

Configuring IP SLA Jitter is similar to configuring IP SLA Echo. You must first create an IP SLA by issuing the **ip sla** *operation-number* command. To specifically configure the IP SLA Jitter operation, you should issue the **udp-jitter** *destination-ip-address destination-port* command. You can also issue the **frequency** *seconds* command to define a rate in seconds at which the Jitter operation will repeat.

After you have configured the SLA, you must schedule the SLA to run by issuing the **ip sla schedule** *operation-number* [**life** {**forever** | *seconds*}] [*start-time* {*hh*:*mm*[:*ss*] [*monthday* | *daymonth*] | **pending** | **now** | **after** *hh*:*mm*:*ss*}] [**ageout** *seconds*] [**recurring**] command. If you issue the **ip sla schedule** *operation-number* command without parameters, the SLA is enabled but placed into the pending state.



IP SLA Authentication

- Uses MD5 hash authentication
- Hashes are calculated from key strings

Configuring a key chain and key string

```
RouterA(config)#key chain SLACHAIN
RouterA(config-keychain)#key 1
RouterA(config-keychain-key)#key-string ASecureString
```

Configuring IP SLA to use the key chain for authentication

```
RouterA(config)#ip sla key-chain SLACHAIN
```

IP SLA Authentication

You can also configure message authentication for IP SLA. With IP SLA message authentication, devices must authenticate with one another before they will accept IP SLA control messages. IP SLA uses MD5 for authentication.

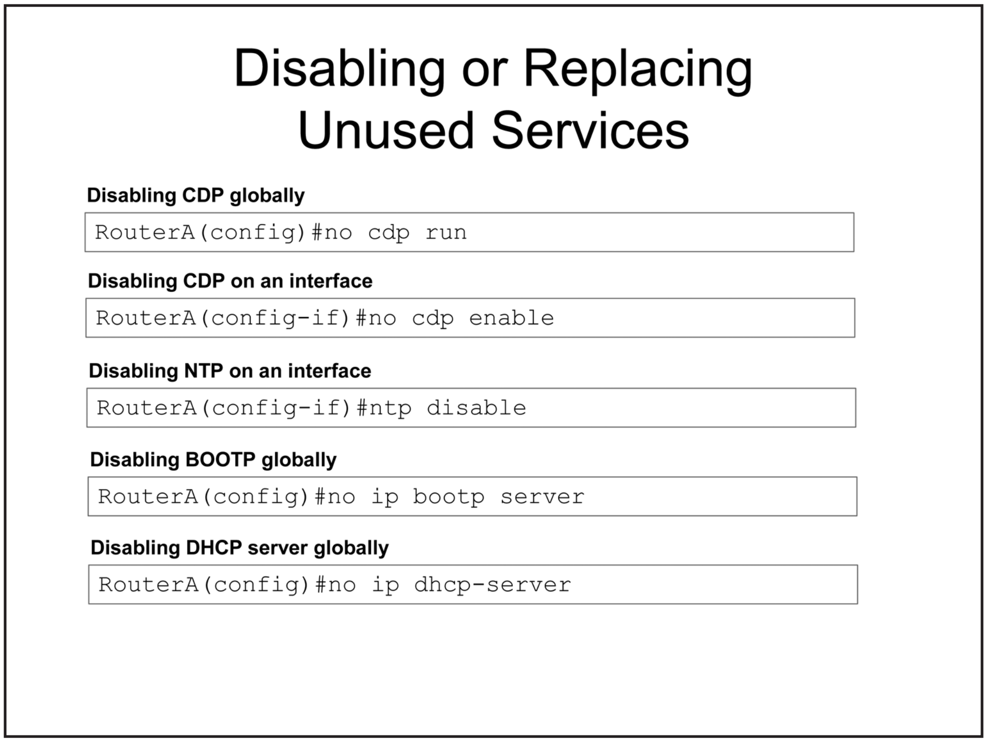
First, you must create a key chain by issuing the **key chain** *keychain-name* command. Issuing this also places the device into keychain configuration mode.

In keychain configuration mode, issue the **key** *key-number* command, which identifies a key on the key chain. Issuing this command places the device into keychain key configuration mode.

In keychain key configuration mode, issue the **key-string** *password* command. The key string is case-sensitive and must match on both devices.

The command sequence in the example above creates a key chain named SLACHAIN, establishes key 1, and creates the authentication string ASecureString.

After you have created the key chain and configure the key string, you must configure IP SLA to use that key chain for authentication. To do so, you should issue the **ip sla key-chain** *keychain-name* command, where *keychain-name* is the name you used in the original **key chain** command. For example, you would issue the **ip sla key-chain SLACHAIN** command to use the key chain in this scenario.



Disabling or Replacing Unused Services

Disabling CDP globally

```
RouterA(config)#no cdp run
```

Disabling CDP on an interface

```
RouterA(config-if)#no cdp enable
```

Disabling NTP on an interface

```
RouterA(config-if)#ntp disable
```

Disabling BOOTP globally

```
RouterA(config)#no ip bootp server
```

Disabling DHCP server globally

```
RouterA(config)#no ip dhcp-server
```

Disabling or Replacing Unused Services

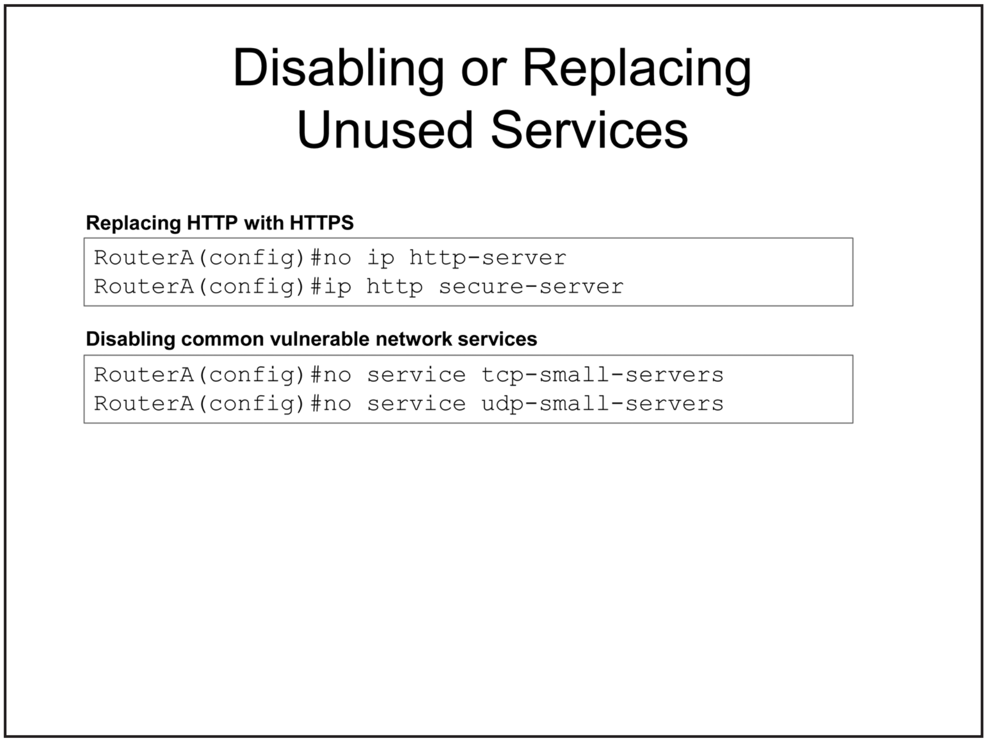
Any service that is enabled and is not required on a router is a security risk. In addition, many network services that are available on a Cisco router can communicate potentially sensitive data in clear text across the network. The simplest method of securing such network services on a Cisco switch is to disable and not use them.

You should consider disabling Cisco Discovery Protocol (CDP) on any device where it is not specifically required. Cisco recommends disabling CDP globally on a device if CDP is not required by your organization. CDP transmits unencrypted information about Cisco devices over the network. If CDP is required, then it should be enabled only on the interfaces that require it. To disable CDP globally on a switch, issue the **no cdp run** command in global configuration mode. To disable CDP on a specific interface, issue the **no cdp enable** command in interface configuration mode.

Cisco recommends disabling NTP on interfaces on which time information should never be received. A malicious user could connect a low stratum NTP device to the network and feed incorrect time information to networked devices. To disable the processing of NTP packets on an interface, issue the **ntp disable** command from interface configuration mode.

Cisco recommends disabling Bootstrap Protocol (BOOTP) on any devices on which it is not used. A malicious user can use BOOTP information to steal an IOS image. In addition, BOOTP is seldom used. To disable BOOTP, issue the **no ip bootp server** command from global configuration mode.

Although Dynamic Host Configuration Protocol (DHCP) is used much more often than BOOTP, DHCP is basically an extension of BOOTP and can be exploited similarly. To disable DHCP, issue the **no ip dhcp-server** command from global configuration mode.



Disabling or Replacing Unused Services

Replacing HTTP with HTTPS

```
RouterA(config)#no ip http-server
RouterA(config)#ip http secure-server
```

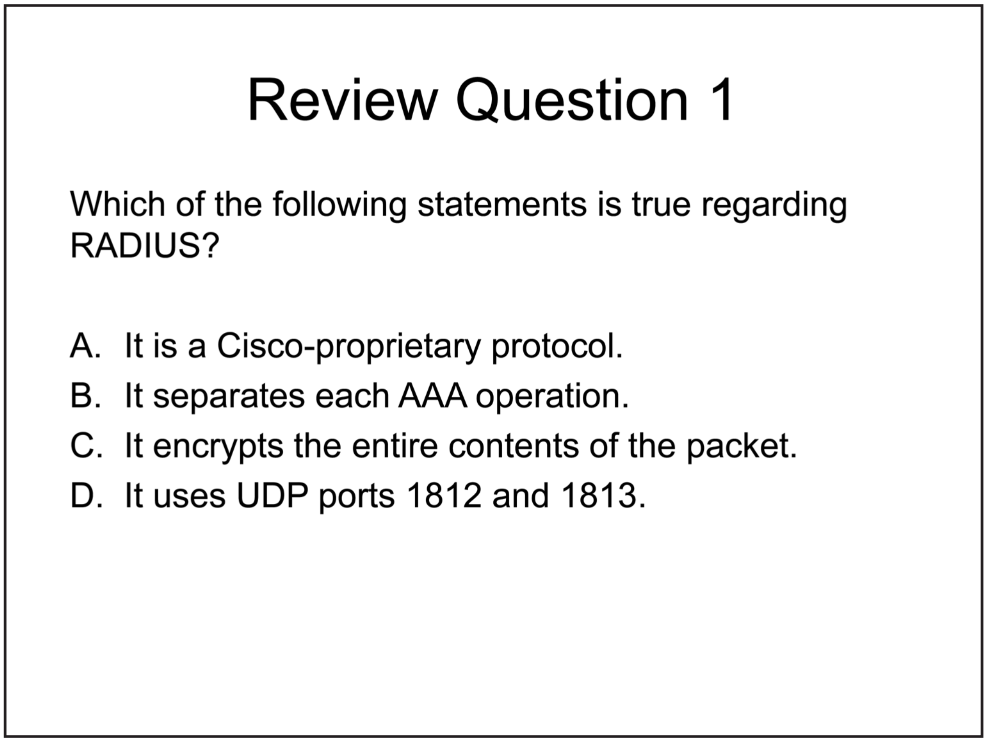
Disabling common vulnerable network services

```
RouterA(config)#no service tcp-small-servers
RouterA(config)#no service udp-small-servers
```

The default Hypertext Transfer Protocol (HTTP) server configuration on a Cisco switch sends authorization information in clear text across a network, making it vulnerable to sniffers. If you require HTTP access to the switch, you should implement a Secure HTTP (HTTPS) server. An HTTPS server encrypts standard HTTP traffic over Secure Sockets Layer (SSL). To replace an HTTP server with an HTTPS server on a Cisco switch, issue the **no ip http-server** command followed by the **ip http secure-server** command in global configuration mode. The **no ip http-server** command disables the HTTP server. The **ip http secure-server** command enables the HTTPS server.

Unnecessary TCP or UDP services that are enabled on a switch can create vulnerable access ports for attackers. You can issue the **no service** command to disable TCP or UDP services that are not required by your organization. The **no service tcp-small-servers** command and the **no service udp-small-servers** command disable the following TCP and UDP services: Echo, Chargen, Discard, and Daytime. Versions of IOS later than 11.2 have TCP small servers and UDP small servers disabled by default.

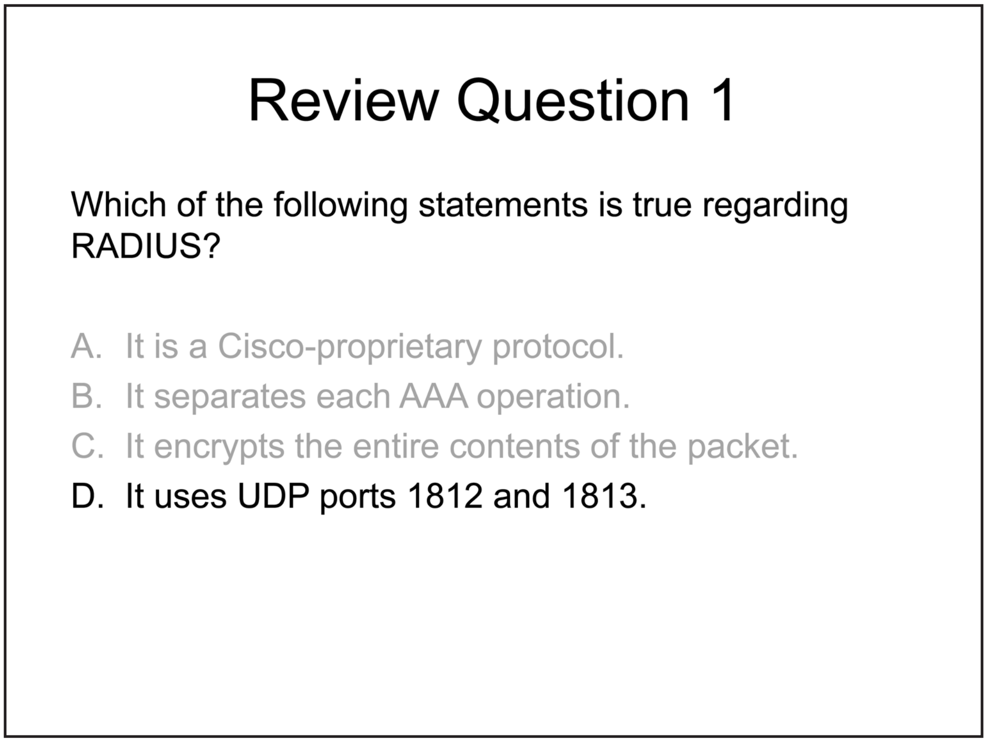
Module Notes



Review Question 1

Which of the following statements is true regarding RADIUS?

A. It is a Cisco-proprietary protocol.
B. It separates each AAA operation.
C. It encrypts the entire contents of the packet.
D. It uses UDP ports 1812 and 1813.



Review Question 1

Which of the following statements is true regarding RADIUS?

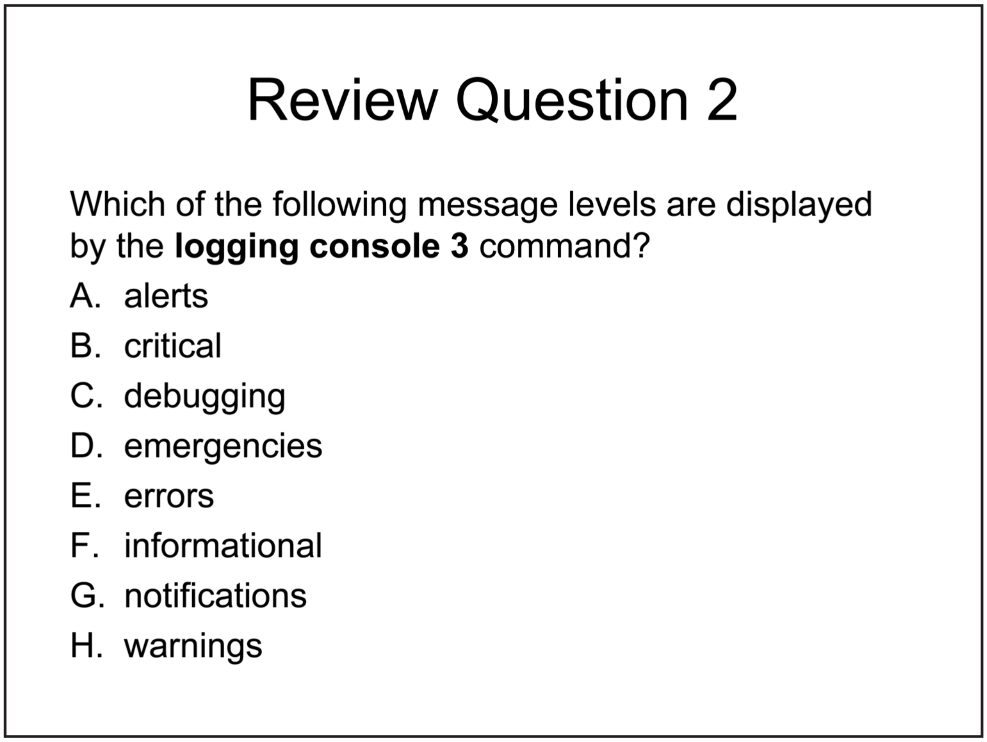
A. It is a Cisco-proprietary protocol.
B. It separates each AAA operation.
C. It encrypts the entire contents of the packet.
D. It uses UDP ports 1812 and 1813.

Remote Authentication Dial-In User Service (RADIUS) uses User Datagram Protocol (UDP) ports 1812 and 1813. UDP port 1812 is used for authentication and authorization. UDP port 1813 is used for accounting.

RADIUS is not a Cisco-proprietary protocol. RADIUS is an Internet Engineering Task Force (IETF) standard protocol that is defined in Request for Comments (RFC) 2865 and 2866. By contrast, Terminal Access Controller Access-Control System Plus (TACACS+) is a Cisco-proprietary protocol.

RADIUS does not separate each Authentication, Authorization, and Accounting (AAA) operation. The authentication and authorization functions of RADIUS are combined into a single function, which limits the flexibility that administrators have when configuring these functions. By contrast, TACACS+ separates the authentication, authorization, and accounting functions of AAA, which enables more granular control of access to resources.

RADIUS does not encrypt the entire contents of the packet; it encrypts only the password of the packet. By contrast, TACACS+ encrypts the entire body of the packet.



Review Question 2

Which of the following message levels are displayed by the **logging console 3** command?

A. alerts

B. critical

C. debugging

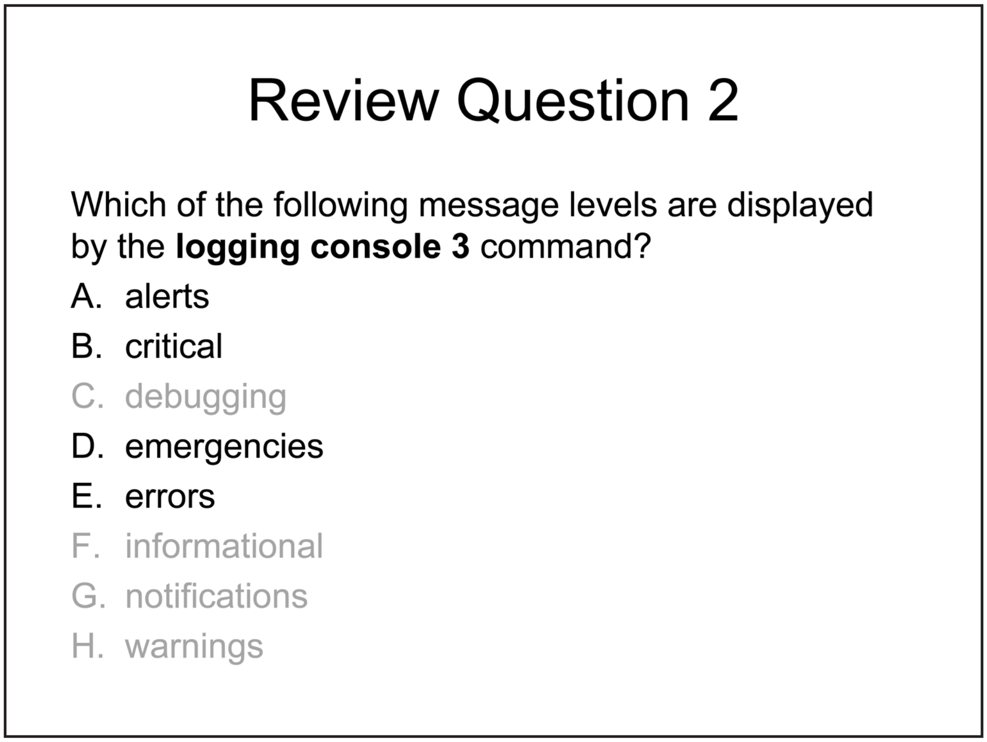
D. emergencies

E. errors

F. informational

G. notifications

H. warnings



Review Question 2

Which of the following message levels are displayed by the **logging console 3** command?

A. alerts

B. critical

C. debugging

D. emergencies

E. errors

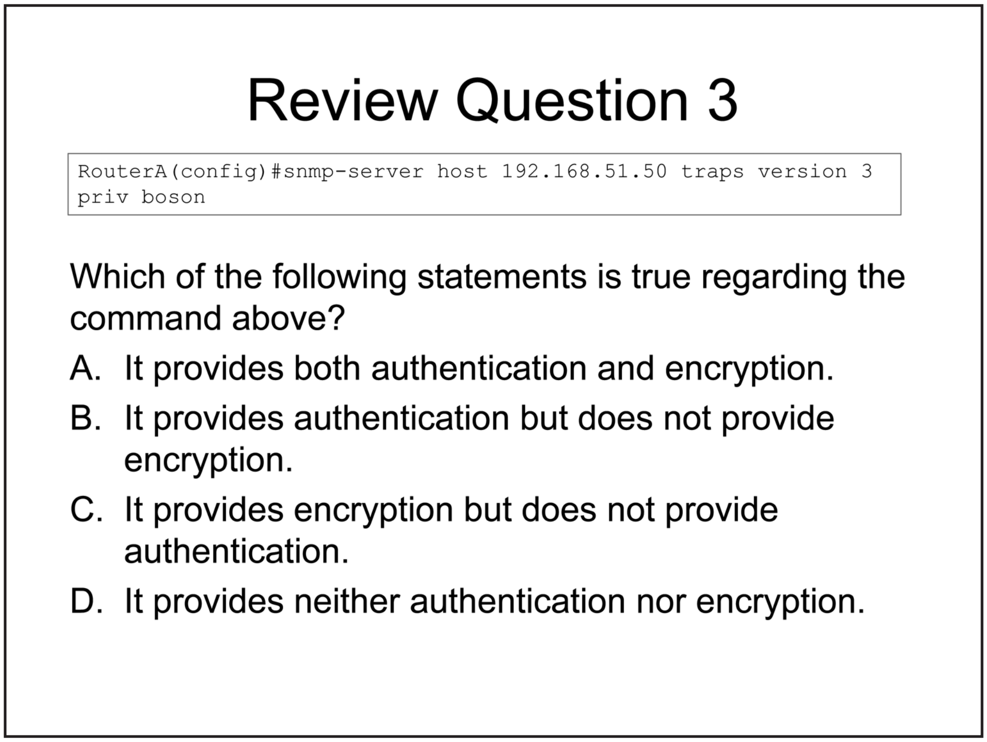
F. informational

G. notifications

H. warnings

The alerts, critical, emergencies, and errors message levels are displayed by the **logging console 3** command. The **logging console 3** command specifies that messages at severity level 3 and lower will be logged to the console. Cisco log messages are divided into the following severity levels:

- 0 – emergencies
- 1 – alerts
- 2 – critical
- 3 – errors
- 4 – warnings
- 5 – notifications
- 6 – informational
- 7 – debugging



Review Question 3

```
RouterA(config)#snmp-server host 192.168.51.50 traps version 3
priv boson
```

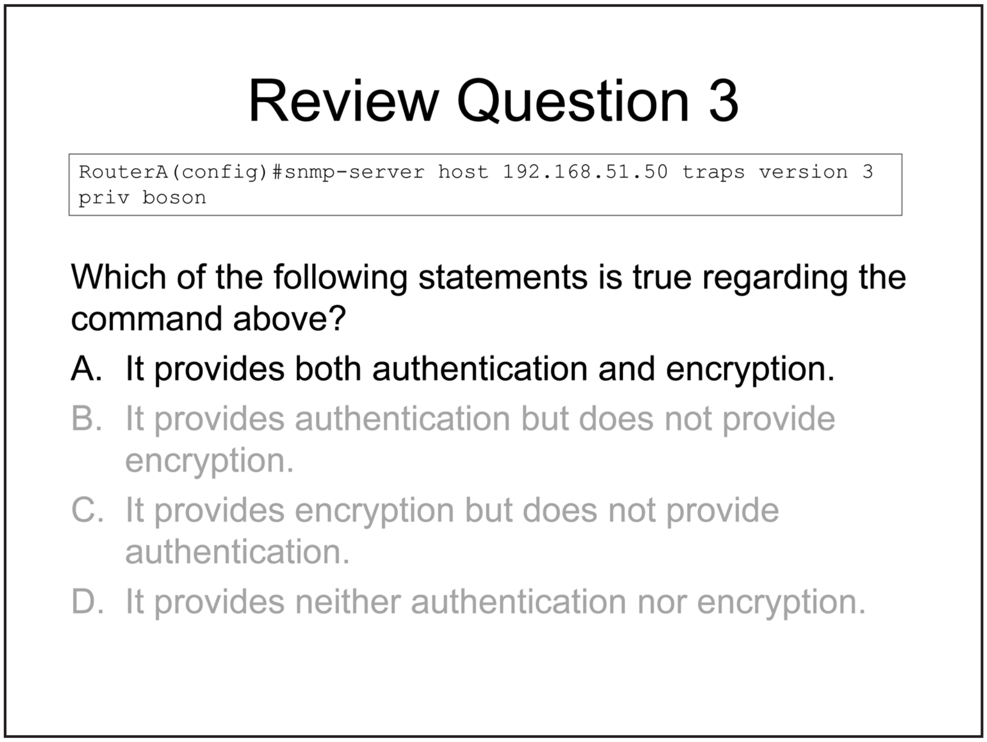
Which of the following statements is true regarding the command above?

A. It provides both authentication and encryption.

B. It provides authentication but does not provide encryption.

C. It provides encryption but does not provide authentication.

D. It provides neither authentication nor encryption.



Review Question 3

```
RouterA(config)#snmp-server host 192.168.51.50 traps version 3
priv boson
```

Which of the following statements is true regarding the command above?

A. It provides both authentication and encryption.

B. It provides authentication but does not provide encryption.

C. It provides encryption but does not provide authentication.

D. It provides neither authentication nor encryption.

The **snmp-server host 192.168.51.50 traps version 3 priv boson** command provides both authentication and encryption. When issuing the **snmp-server host** command with **version 3**, you can specify the **priv**, **auth**, or **noauth** keyword to configure the Simple Network Management Protocol version 3 (SNMPv3) access mode.

- Issuing the **priv** keyword configures SNMPv3 to use the authPriv access mode, which provides both authentication and encryption.
- Issuing the **auth** keyword configures SNMPv3 to use the authNoPriv access mode, which provides authentication but not encryption.
- Issuing the **noauth** keyword configures SNMPv3 to use the noAuthNoPriv access mode, which provides neither authentication nor encryption.

Lab Exercises

Module 1: Basic Router Security and Management

Lab 1.1 – Initial Configuration
Lab 1.2 – Router Remote Access via Telnet
Lab 1.3 – AAA Login Authentication and Exec Authorization
Lab 1.4 – Configuring SSH
Lab 1.5 – Configuring NTP
Lab 1.6 – Configuring NTP Authentication
Lab 1.7 – System Message Logging
Lab 1.8 – Basic Debugging
Lab 1.9 – Configuring Network Device Management
Lab 1.10 – NetFlow

Labs powered by NetSim® NETWORK SIMULATOR®

The labs referenced in this book have been printed in the Boson Lab Guide, which is available for purchase. To learn more about the Boson NetSim or to purchase and download the software, please visit www.boson.com/netsim-cisco-network-simulator.

Index

Certification Candidates

Boson Software's ExSim-Max practice exams are designed to simulate the complete exam experience. These practice exams have been written by in-house authors who have over 30 years combined experience writing practice exams. ExSim-Max is designed to simulate the live exam, including topics covered, question types, question difficulty, and time allowed, so you know what to expect. To learn more about ExSim-Max practice exams, please visit www.boson.com/exsim-max-practice-exams or contact Boson Software.

Organizational and Volume Customers

Boson Software's outstanding IT training tools serve the skill development needs of organizations such as colleges, technical training educators, corporations, and governmental agencies. If your organization would like to inquire about volume opportunities and discounts, please contact Boson Software at orgsales@boson.com.

Contact Information

E-Mail: support@boson.com
Phone: 877-333-EXAM (3926)
615-889-0121
Fax: 615-889-0122
Address: 25 Century Blvd., Ste. 500
Nashville, TN 37214

Boson.com

877.333.3926 support@boson.com