Secure Shell Version 2 Support

Last Updated: January 16, 2012

The Secure Shell Version 2 Support feature allows you to configure Secure Shell (SSH) Version 2. SSH runs on top of a reliable transport layer and provides strong authentication and encryption capabilities. Currently, the only reliable transport that is defined for SSH is TCP. SSH provides a means to securely access and securely execute commands on another computer over a network. The Secure Copy Protocol (SCP) feature that is provided with SSH also allows for the secure transfer of files.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Secure Shell Version 2 Support

Prior to configuring SSH, perform the following task:

- Download the required image on your router. The SSH server requires you to have a k9 (Triple Data Encryption Standard [3DES]) software image from downloaded on your router.

Restrictions for Secure Shell Version 2 Support

- SSH servers and SSH clients are supported in k9 software images.
- Execution Shell, remote command execution, and Secure Copy Protocol (SCP) are the only applications supported.
- Rivest, Shamir, and Adelman (RSA) key generation is an SSH server side requirement. Routers that act as SSH clients do not need to generate RSA keys.
- The RSA key-pair size must be greater than or equal to 768.
- The following functionality is not supported:
    - RSA user authentication (in the SSH server or SSH client for Cisco IOS XE software)
    - Public key authentication
    - SSH server strict host key check
    - Port forwarding
    - Compression

Information About Secure Shell Version 2 Support

- Secure Shell Version 2, page 2
- Secure Shell Version 2 Enhancements, page 3
- SNMP Trap Generation, page 3

Secure Shell Version 2

The Secure Shell Version 2 Support feature allows you to configure SSH Version 2.

The configuration for the SSH Version 2 server is similar to the configuration for SSH Version 1. The command **ip ssh version** has been introduced so that you may define which version of SSH that you want to configure. If you do not configure this command, SSH by default runs in compatibility mode; that is, both SSH Version 1 and SSH Version 2 connections are honored.

Note SSH Version 1 is a protocol that has never been defined in a standard. If you do not want your router to fall back to the undefined protocol (Version 1), you should use the **ip ssh version** command and specify Version 2.

The **ip ssh rsa keypair-name** command was also introduced so that you can enable a SSH connection using RSA keys that you have configured. Previously, SSH was linked to the first RSA keys that were generated (that is, SSH was enabled when the first RSA key pair was generated). The behavior still exists, but by using the **ip ssh rsa keypair-name** command, you can overcome that behavior. If you configure the **ip ssh rsa keypair-name** command with a key-pair name, SSH is enabled if the key pair exists, or SSH will be enabled if the key pair is generated later. If you use this command to enable SSH, you are not forced to configure a host name and a domain name, which was required in SSH Version 1 of the Cisco IOS XE software.

Note The login banner is supported in Secure Shell Version 2, but it is not supported in Secure Shell Version 1.

Secure Shell Version 2 Enhancements

The Secure Shell Version 2 Enhancements include a number of additional capabilities such as supporting VRF aware SSH, SSH debug enhancements, and Diffie-Hellman group exchange support.

The Cisco IOS XE SSH implementation has traditionally used 768 bit modulus but with an increasing need for higher key sizes to accommodate Diffie-Hellman (DH) Group 14 (2048 bits) and Group 16 (4096 bits) cryptographic applications a message exchange between the client and server to establish the favored DH group becomes necessary. The **ip ssh dh min size**command was introduced so you can configure modulus size on the SSH server. In addition to this the **ssh**command was extended to add VRF awareness to SSH client side functionality through which the VRF instance name in the client is provided with the IP address to look up the correct routing table and establish a connection.

Debugging has been enhanced by modifying SSH debug commands. The **debug ip ssh** command has been extended to allow you to simplify the debugging process. Previously this command printed all debug messages related to SSH regardless of what was specifically required. The behavior still exists, but if you configure the **debug ip ssh** command with a keyword messages are limited to information specified by the keyword.

SNMP Trap Generation

Simple Network Management Protocol (SNMP) traps will be generated automatically when an SSH session terminates if the traps have been enabled and SNMP debugging has been turned on. For information about enabling SNMP traps, see the Configuring SNMP Support feature module.

Note When configuring the **snmp-server host** command, the IP address must be the address of the PC that has the SSH (telnet) client and that has IP connectivity to the SSH server. See Example Setting an SNMP Trap, page 15 for more information.

You must also turn on SNMP debugging using the **debug snmp packet** command to display the traps. The trap information includes information such as the number of bytes sent and the protocol that was used for the SSH session. See Example SNMP Debugging, page 16 for more information.

How to Configure Secure Shell Version 2 Support

- Configuring a Router for SSH Version 2 Using a Host Name and Domain Name, page 4
- Configuring a Router for SSH Version 2 Using RSA Key Pairs, page 5
- Starting an Encrypted Session with a Remote Device, page 6
- Enabling Secure Copy Protocol on the SSH Server, page 7
- Verifying the Status of the Secure Shell Connection Using the show ssh Command, page 9
- Verifying the Secure Shell Status Using the show ip ssh Command, page 11
- Monitoring and Maintaining Secure Shell Version 2, page 12

Configuring a Router for SSH Version 2 Using a Host Name and Domain Name

To configure your router for SSH Version 2 using a host name and domain name, perform the following steps. You may also configure SSH Version 2 by using the RSA key pair configuration. See Configuring a Router for SSH Version 2 Using RSA Key Pairs, page 5 for more information.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **hostname** *hostname*
4. **ip domain-name** *name*
5. **crypto key generate rsa**
6. **ip ssh** [**time-out** *seconds* | **authentication-retries** *integer*]
7. **ip ssh version** [**1** | **2**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**hostname** *hostname* **Example:** `Router (config)# hostname cisco 7200`	Configures a host name for your router.
Step 4	**ip domain-name** *name* **Example:** `Router (config)# ip domain-name example.com`	Configures a domain name for your router.

	Command or Action	Purpose
Step 5	**crypto key generate rsa** **Example:** `Router (config)# crypto key generate rsa`	Enables the SSH server for local and remote authentication.
Step 6	**ip ssh** [**time-out** *seconds* | **authentication-retries** *integer*] **Example:** `Router (config)# ip ssh time-out 120`	(Optional) Configures SSH control variables on your router.
Step 7	**ip ssh version** [**1** | **2**] **Example:** `Router (config)# ip ssh version 1`	(Optional) Specifies the version of SSH to be run on your router.

Configuring a Router for SSH Version 2 Using RSA Key Pairs

To enable SSH Version 2 without configuring a host name or domain name, perform the following steps. SSH Version 2 will be enabled if the key pair that you configure already exists or if it is generated later. You may also configure SSH Version 2 by using the host name and domain name configuration. See Configuring a Router for SSH Version 2 Using a Host Name and Domain Name, page 4 for more information.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ssh rsa keypair-name** *keypair-name*
4. **crypto key generate rsa usage-keys label** *key-label* **modulus** *modulus-size*
5. **ip ssh** [**time-out** *seconds* | **authentication-retries** *integer*]
6. **ip ssh version 2**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip ssh rsa keypair-name** *keypair-name* **Example:** `Router (config)# ip ssh rsa keypair-name sshkeys`	Specifies which RSA keypair to use for SSH usage. **Note** A router can have many RSA key pairs.
Step 4	**crypto key generate rsa usage-keys label** *key-label* **modulus** *modulus-size* **Example:** `Router (config)# crypto key generate rsa usage-keys label sshkeys modulus 768`	Enables the SSH server for local and remote authentication on the router. For SSH Version 2, the modulus size must be at least 768 bits. **Note** To delete the RSA key-pair, use the **crypto key zeroize rsa** command. After you have deleted the RSA key-pair, you automatically disable the SSH server.
Step 5	**ip ssh** [**time-out** *seconds* | **authentication-retries** *integer*] **Example:** `Router (config)# ip ssh time-out 120`	Configures SSH control variables on your router.
Step 6	**ip ssh version 2** **Example:** `Router (config)# ip ssh version 2`	Specifies the version of SSH to be run on a router.

Starting an Encrypted Session with a Remote Device

To start an encrypted session with a remote networking device, perform the following step. (You do not have to enable your router. SSH can be run in disabled mode.)

Note The device you wish to connect with must support a SSH server that has an encryption algorithm that is supported in Cisco IOS XE software.

SUMMARY STEPS

1. **ssh** [**-v** {**1** | **2**}][**-c** {**3des** | **aes128-cbc** | **aes192-cbc** | **aes256-cbc**}] [**-m** {**hmac-md5** | **hmac-md5-96** | **hmac-sha1** | **hmac-sha1-96**}] [**l** *userid*] [**-o numberofpasswordprompts** *n*] [**-p** *port-num*]{*ip-addr* | *hostname*} [*command*]

DETAILED STEPS

	Command or Action	Purpose

Step 1	**ssh** [**-v** {**1** | **2**}][**-c** {**3des** | **aes128-cbc** | **aes192-cbc** | **aes256-cbc**}] [**-m** {**hmac-md5** | **hmac-md5-96** | **hmac-sha1** | **hmac-sha1-96**}] [**l** *userid*] [**-o numberofpasswordprompts** *n*] [**-p** *port-num*]{*ip-addr* | *hostname*} [*command*]
	
	Example:
	`Router# ssh -v 2 -c aes256-cbc -m hmac-sha1-96 -l user2 10.76.82.24`
	
	Example:
	
	Example:
	`Or`
	
	Example:
	`The above example adheres to the SSH Version 2 conventions. A more natural and common way to start a session is by linking the username with the hostname. For example, the following configuration example provides an end result that is identical to that of the above example:`
	
	Example:
	`Router# ssh -v 2 -c aes256-cbc -m hmac-sha1-96 user2@10.76.82.24`
		Starts an encrypted session with a remote networking device.

- Troubleshooting Tips, page 7

Troubleshooting Tips

The **ip ssh version** command can be used for troubleshooting your SSH configuration. By changing versions, you can determine which SSH version has a problem.

Enabling Secure Copy Protocol on the SSH Server

To configure server-side functionality for SCP, perform the following steps. This example shows a typical configuration that allows the router to securely copy files from a remote workstation.

SCP relies on AAA authentication and authorization to function correctly. Therefore AAA must be configured on the router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa new-model**
4. **aaa authentication login default local**
5. **aaa authorization exec default local**
6. **username** *name* **privilege** *privilege-level* **password** *password*
7. **ip ssh time-out** *seconds*
8. **ip ssh authentication-retries** *integer*
9. **ip scp server enable**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Router(config)# aaa new-model`	Enables the authentication, authorization, and accounting (AAA) access control model.
Step 4	**aaa authentication login default local** **Example:** `Router(config)# aaa authentication login default local`	Sets authentication, authorization, and accounting (AAA) authentication at login to use the local username database for authentication.

	Command or Action	Purpose
Step 5	**aaa authorization exec default local** **Example:** `Router(config)# aaa authorization exec default local`	Sets the parameters that restrict user access to a network; runs the authorization to determine if the user ID allowed to run an EXEC shell; and specifies that the system uses the local database for authorization.
Step 6	**username** *name* **privilege** *privilege-level* **password** *password* **Example:** `Router(config)# username samplename privilege 15 password password1`	Establishes a username-based authentication system, specifies the username, the privilege level, and an unencrypted password.
Step 7	**ip ssh time-out** *seconds* **Example:** `Router(config)# ip ssh time-out 120`	Sets the time interval (in seconds) that the router waits for the SSH client to respond.
Step 8	**ip ssh authentication-retries** *integer* **Example:** `Router(config)# ip ssh authentication-retries 3`	Sets the number of authentication attempts after which the interface is reset.
Step 9	**ip scp server enable** **Example:** `Router (config)# ip scp server enable`	Enables the router to securely copy files from a remote workstation.

- Troubleshooting Tips, page 9

Troubleshooting Tips

To troubleshoot SCP authentication problems, use the **debug ip scp**command.

Verifying the Status of the Secure Shell Connection Using the show ssh Command

To display the status of the SSH connection on your router, use the **show ssh** command.

SUMMARY STEPS

1. **enable**
2. **show ssh**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ssh** **Example:** `Router# show ssh`	Displays the status of SSH server connections.

Examples

Version 1 and Version 2 Connections

Version 2 Connection with No Version 1

Version 1 Connection with No Version 2

The following sample output from the **show ssh** command display status about various SSH Version 1 and Version 2 connections.

```
-----------------------------------------------------------------------
Router# show ssh
Connection      Version Encryption      State                   Username
 0              1.5     3DES            Session started         lab
Connection Version Mode Encryption  Hmac                State
Username
1          2.0     IN   aes128-cbc  hmac-md5     Session started       lab
1          2.0     OUT  aes128-cbc  hmac-md5     Session started       lab
-------------------------------------------------------------------------
```

```
-------------------------------------------------------------------------
Router# show ssh
Connection Version Mode Encryption  Hmac                State
Username
1          2.0     IN   aes128-cbc  hmac-md5     Session started       lab
1          2.0     OUT  aes128-cbc  hmac-md5     Session started       lab
%No SSHv1 server connections running.
-------------------------------------------------------------------------
```

```
-------------------------------------------------------------------------
Router# show ssh
Connection      Version Encryption      State                   Username
 0              1.5     3DES            Session started         lab
```

```
%No SSHv2 server connections running.
-------------------------------------------------------------------------
```

Verifying the Secure Shell Status Using the show ip ssh Command

To verify your SSH configuration, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **show ip ssh**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip ssh** **Example:** `Router# show ip ssh`	Displays the version and configuration data for SSH.

Examples

Version 1 and Version 2 Connections

Version 2 Connection with No Version 1

Version 1 Connection with No Version 2

The following examples from the **show ip ssh** command display the version of SSH that is enabled, the authentication timeout values, and the number of authentication retries.

```
-----------------------------------------------------------------------
router# show ip ssh
3d06h: %SYS-5-CONFIG_I: Configured from console by consoleh
SSH Enabled - version 1.99
Authentication timeout: 120 secs; Authentication retries: 3
-----------------------------------------------------------------------
```

```
------------------------------------------------------------------------
Router# show ip ssh
SSH Enabled - version 2.0
Authentication timeout: 120 secs; Authentication retries: 3
------------------------------------------------------------------------
```

```
------------------------------------------------------------------------
Router# show ip ssh
3d06h: %SYS-5-CONFIG_I: Configured from console by console
```

```
SSH Enabled - version 1.5
Authentication timeout: 120 secs; Authentication retries: 3
------------------------------------------------------------------------
```

Monitoring and Maintaining Secure Shell Version 2

To display debug messages about the SSH connections, use the **debug ip ssh** command.

SUMMARY STEPS

1. **enable**
2. **debug ip ssh**
3. **debug snmp packet**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug ip ssh** **Example:** `Router# debug ip ssh`	Displays debugging messages for SSH.
Step 3	**debug snmp packet** **Example:** `Router# debug snmp packet`	Displays information about every SNMP packet sent or received by the router.

Example

The following output from the **debug ip ssh** command shows that the digit 2 keyword has been assigned, signifying that it is an SSH Version 2 connection.

```
Router# debug ip ssh
00:33:55: SSH1: starting SSH control process
00:33:55: SSH1: sent protocol version id SSH-1.99-Cisco-1.25
00:33:55: SSH1: protocol version id is - SSH-2.0-OpenSSH_2.5.2p2
00:33:55: SSH2 1: send: len 280 (includes padlen 4)
00:33:55: SSH2 1: SSH2_MSG_KEXINIT sent
00:33:55: SSH2 1: ssh_receive: 536 bytes received
00:33:55: SSH2 1: input: packet len 632
00:33:55: SSH2 1: partial packet 8, need 624, maclen 0
00:33:55: SSH2 1: ssh_receive: 96 bytes received
00:33:55: SSH2 1: partial packet 8, need 624, maclen 0
00:33:55: SSH2 1: input: padlen 11
00:33:55: SSH2 1: received packet type 20
00:33:55: SSH2 1: SSH2_MSG_KEXINIT received
00:33:55: SSH2: kex: client->server aes128-cbc hmac-md5 none
```

```
00:33:55: SSH2: kex: server->client aes128-cbc hmac-md5 none
00:33:55: SSH2 1: expecting SSH2_MSG_KEXDH_INIT
00:33:55: SSH2 1: ssh_receive: 144 bytes received
00:33:55: SSH2 1: input: packet len 144
00:33:55: SSH2 1: partial packet 8, need 136, maclen 0
00:33:55: SSH2 1: input: padlen 5
00:33:55: SSH2 1: received packet type 30
00:33:55: SSH2 1: SSH2_MSG_KEXDH_INIT received
00:33:55: SSH2 1: signature length 111
00:33:55: SSH2 1: send: len 384 (includes padlen 7)
00:33:55: SSH2: kex_derive_keys complete
00:33:55: SSH2 1: send: len 16 (includes padlen 10)
00:33:55: SSH2 1: newkeys: mode 1
00:33:55: SSH2 1: SSH2_MSG_NEWKEYS sent
00:33:55: SSH2 1: waiting for SSH2_MSG_NEWKEYS
00:33:55: SSH2 1: ssh_receive: 16 bytes received
00:33:55: SSH2 1: input: packet len 16
00:33:55: SSH2 1: partial packet 8, need 8, maclen 0
00:33:55: SSH2 1: input: padlen 10
00:33:55: SSH2 1: newkeys: mode 0
00:33:55: SSH2 1: received packet type 2100:33:55: SSH2 1: SSH2_MSG_NEWKEYS received
00:33:56: SSH2 1: ssh_receive: 48 bytes received
00:33:56: SSH2 1: input: packet len 32
00:33:56: SSH2 1: partial packet 16, need 16, maclen 16
00:33:56: SSH2 1: MAC #3 ok
00:33:56: SSH2 1: input: padlen 10
00:33:56: SSH2 1: received packet type 5
00:33:56: SSH2 1: send: len 32 (includes padlen 10)
00:33:56: SSH2 1: done calc MAC out #3
00:33:56: SSH2 1: ssh_receive: 64 bytes received
00:33:56: SSH2 1: input: packet len 48
00:33:56: SSH2 1: partial packet 16, need 32, maclen 16
00:33:56: SSH2 1: MAC #4 ok
00:33:56: SSH2 1: input: padlen 9
00:33:56: SSH2 1: received packet type 50
00:33:56: SSH2 1: send: len 32 (includes padlen 13)
00:33:56: SSH2 1: done calc MAC out #4
00:34:04: SSH2 1: ssh_receive: 160 bytes received
00:34:04: SSH2 1: input: packet len 64
00:34:04: SSH2 1: partial packet 16, need 48, maclen 16
00:34:04: SSH2 1: MAC #5 ok
00:34:04: SSH2 1: input: padlen 13
00:34:04: SSH2 1: received packet type 50
00:34:04: SSH2 1: send: len 16 (includes padlen 10)
00:34:04: SSH2 1: done calc MAC out #5
00:34:04: SSH2 1: authentication successful for lab
00:34:04: SSH2 1: input: packet len 64
00:34:04: SSH2 1: partial packet 16, need 48, maclen 16
00:34:04: SSH2 1: MAC #6 ok
00:34:04: SSH2 1: input: padlen 6
00:34:04: SSH2 1: received packet type 2
00:34:04: SSH2 1: ssh_receive: 64 bytes received
00:34:04: SSH2 1: input: packet len 48
00:34:04: SSH2 1: partial packet 16, need 32, maclen 16
00:34:04: SSH2 1: MAC #7 ok
00:34:04: SSH2 1: input: padlen 19
00:34:04: SSH2 1: received packet type 90
00:34:04: SSH2 1: channel open request
00:34:04: SSH2 1: send: len 32 (includes padlen 10)
00:34:04: SSH2 1: done calc MAC out #6
00:34:04: SSH2 1: ssh_receive: 192 bytes received
00:34:04: SSH2 1: input: packet len 64
00:34:04: SSH2 1: partial packet 16, need 48, maclen 16
00:34:04: SSH2 1: MAC #8 ok
00:34:04: SSH2 1: input: padlen 13
00:34:04: SSH2 1: received packet type 98
00:34:04: SSH2 1: pty-req request
00:34:04: SSH2 1: setting TTY - requested: height 24, width 80; set: height 24,
width 80
00:34:04: SSH2 1: input: packet len 96
00:34:04: SSH2 1: partial packet 16, need 80, maclen 16
00:34:04: SSH2 1: MAC #9 ok
00:34:04: SSH2 1: input: padlen 11
```

```
00:34:04: SSH2 1: received packet type 98
00:34:04: SSH2 1: x11-req request
00:34:04: SSH2 1: ssh_receive: 48 bytes received
00:34:04: SSH2 1: input: packet len 32
00:34:04: SSH2 1: partial packet 16, need 16, maclen 16
00:34:04: SSH2 1: MAC #10 ok
00:34:04: SSH2 1: input: padlen 12
00:34:04: SSH2 1: received packet type 98
00:34:04: SSH2 1: shell request
00:34:04: SSH2 1: shell message received
00:34:04: SSH2 1: starting shell for vty
00:34:04: SSH2 1: send: len 48 (includes padlen 18)
00:34:04: SSH2 1: done calc MAC out #7
00:34:07: SSH2 1: ssh_receive: 48 bytes received
00:34:07: SSH2 1: input: packet len 32
00:34:07: SSH2 1: partial packet 16, need 16, maclen 16
00:34:07: SSH2 1: MAC #11 ok
00:34:07: SSH2 1: input: padlen 17
00:34:07: SSH2 1: received packet type 94
00:34:07: SSH2 1: send: len 32 (includes padlen 17)
00:34:07: SSH2 1: done calc MAC out #8
00:34:07: SSH2 1: ssh_receive: 48 bytes received
00:34:07: SSH2 1: input: packet len 32
00:34:07: SSH2 1: partial packet 16, need 16, maclen 16
00:34:07: SSH2 1: MAC #12 ok
00:34:07: SSH2 1: input: padlen 17
00:34:07: SSH2 1: received packet type 94
00:34:07: SSH2 1: send: len 32 (includes padlen 17)
00:34:07: SSH2 1: done calc MAC out #9
00:34:07: SSH2 1: ssh_receive: 48 bytes received
00:34:07: SSH2 1: input: packet len 32
00:34:07: SSH2 1: partial packet 16, need 16, maclen 16
00:34:07: SSH2 1: MAC #13 ok
00:34:07: SSH2 1: input: padlen 17
00:34:07: SSH2 1: received packet type 94
00:34:07: SSH2 1: send: len 32 (includes padlen 17)
00:34:07: SSH2 1: done calc MAC out #10
00:34:08: SSH2 1: ssh_receive: 48 bytes received
00:34:08: SSH2 1: input: packet len 32
00:34:08: SSH2 1: partial packet 16, need 16, maclen 16
00:34:08: SSH2 1: MAC #14 ok
00:34:08: SSH2 1: input: padlen 17
00:34:08: SSH2 1: received packet type 94
00:34:08: SSH2 1: send: len 32 (includes padlen 17)
00:34:08: SSH2 1: done calc MAC out #11
00:34:08: SSH2 1: ssh_receive: 48 bytes received
00:34:08: SSH2 1: input: packet len 32
00:34:08: SSH2 1: partial packet 16, need 16, maclen 16
00:34:08: SSH2 1: MAC #15 ok
00:34:08: SSH2 1: input: padlen 17
00:34:08: SSH2 1: received packet type 94
00:34:08: SSH2 1: send: len 32 (includes padlen 16)
00:34:08: SSH2 1: done calc MAC out #12
00:34:08: SSH2 1: send: len 48 (includes padlen 18)
00:34:08: SSH2 1: done calc MAC out #13
00:34:08: SSH2 1: send: len 16 (includes padlen 6)
00:34:08: SSH2 1: done calc MAC out #14
00:34:08: SSH2 1: send: len 16 (includes padlen 6)
00:34:08: SSH2 1: done calc MAC out #15
00:34:08: SSH1: Session terminated normally
```

Configuration Examples for Secure Shell Version 2 Support

- Example Configuring Secure Shell Version 1, page 15
- Example ConfiguringSecureShellVersion2, page 15
- Example Configuring Secure Shell Versions 1 and 2, page 15
- Example Starting an Encrypted Session with a Remote Device, page 15

Example Configuring Secure Shell Version 1

```
Router# configure terminal
Router (config)# ip ssh version 1
Router (config)# end
```

Example ConfiguringSecureShellVersion2

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# ip ssh version 2
Router(config)# end
```

Example Configuring Secure Shell Versions 1 and 2

```
Router# configure terminal
Router (config)# no ip ssh version
Router (config)# end
```

Example Starting an Encrypted Session with a Remote Device

```
Router# ssh -v 2 -c aes256-cbc -m hmac-sha1-160 -l shaship 10.76.82.24
```

Example Configuring Server-Side SCP

The following example shows how to configure server-side functionality for SCP. This example also configures AAA authentication and Authorization on the router. This example uses a locally defined username and password.

```
Router# configure terminal
Router (config)# aaa new-model
Router (config)# aaa authentication login default local
Router (config)# aaa authorization exec default local
Router (config)# username samplename privilege 15 password password1
Router (config)# ip ssh time-out 120
Router (config)# ip ssh authentication-retries 3
Router (config)# ip scp server enable
Router (config)# end
```

Example Setting an SNMP Trap

The following shows that an SNMP trap has been set. The trap notification is generated automatically when the SSH session terminates. For an example of SNMP trap debug output, see the section "Example SNMP Debugging, page 16."

```
snmp-server
snmp-server host a.b.c.d public tty
```

Where a.b.c.d is the IP address of the SSH client.

Example SNMP Debugging

The following is sample output using the **debug snmp packet** command. The output provides SNMP trap information for an SSH session.

```
Router1# debug snmp packet
SNMP packet debugging is on
Router1# ssh -l lab 10.0.0.2
Password:
Router2# exit
[Connection to 10.0.0.2 closed by foreign host]
Router1#
*Jul 18 10:18:42.619: SNMP: Queuing packet to 10.0.0.2
*Jul 18 10:18:42.619: SNMP: V1 Trap, ent cisco, addr 10.0.0.1, gentrap 6, spectrap 1
local.9.3.1.1.2.1 = 6
tcpConnEntry.1.10.0.0.1.22.10.0.0.2.55246 = 4
ltcpConnEntry.5.10.0.0.1.22.10.0.0.2.55246 = 1015
ltcpConnEntry.1.10.0.0.1.22.10.0.0.2.55246 = 1056
ltcpConnEntry.2.10.0.0.1.22.10.0.0.2.55246 = 1392
local.9.2.1.18.2 = lab
*Jul 18 10:18:42.879: SNMP: Packet sent via UDP to 10.0.0.2
Router1#
```

Example SSH Debugging Enhancements

The following is sample output from the **debug ip ssh detail**command. The output provides debugging information regarding the SSH protocol and channel requests.

```
Router# debug ip ssh detail
00:04:22: SSH0: starting SSH control process
00:04:22: SSH0: sent protocol version id SSH-1.99-Cisco-1.25
00:04:22: SSH0: protocol version id is - SSH-1.99-Cisco-1.25
00:04:22: SSH2 0: SSH2_MSG_KEXINIT sent
00:04:22: SSH2 0: SSH2_MSG_KEXINIT received
00:04:22: SSH2:kex: client->server enc:aes128-cbc mac:hmac-sha1
00:04:22: SSH2:kex: server->client enc:aes128-cbc mac:hmac-sha1
00:04:22: SSH2 0: expecting SSH2_MSG_KEXDH_INIT
00:04:22: SSH2 0: SSH2_MSG_KEXDH_INIT received
00:04:22: SSH2: kex_derive_keys complete
00:04:22: SSH2 0: SSH2_MSG_NEWKEYS sent
00:04:22: SSH2 0: waiting for SSH2_MSG_NEWKEYS
00:04:22: SSH2 0: SSH2_MSG_NEWKEYS received
00:04:24: SSH2 0: authentication successful for lab
00:04:24: SSH2 0: channel open request
00:04:24: SSH2 0: pty-req request
00:04:24: SSH2 0: setting TTY - requested: height 24, width 80; set: height 24, width 80
00:04:24: SSH2 0: shell request
00:04:24: SSH2 0: shell message received
00:04:24: SSH2 0: starting shell for vty
00:04:38: SSH0: Session terminated normally
```

The following is sample output from the **debug ip ssh packet**command. The output provides debugging information regarding the ssh packet.

```
Router# debug ip ssh packet
00:05:43: SSH2 0: send:packet of  length 280 (length also includes padlen of 4)
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: input: total packet length of 280 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 24 bytes received
```

```
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 4 bytes
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: input: total packet length of 144 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 16 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 6 bytes
00:05:43: SSH2 0: signature length 143
00:05:43: SSH2 0: send:packet of  length 448 (length also includes padlen of 7)
00:05:43: SSH2 0: send:packet of  length 16 (length also includes padlen of 10)
00:05:43: SSH2 0: newkeys: mode 1
00:05:43: SSH2 0: ssh_receive: 16 bytes received
00:05:43: SSH2 0: input: total packet length of 16 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 8 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 10 bytes
00:05:43: SSH2 0: newkeys: mode 0
00:05:43: SSH2 0: ssh_receive: 52 bytes received
00:05:43: SSH2 0: input: total packet length of 32 bytes
00:05:43: SSH2 0: partial packet length(block size)16 bytes,needed 16 bytes, maclen 20
```

00:05:43: SSH2 0: MAC compared for #3 :ok

Where to Go Next

You have to use a SSH remote device that supports SSH Version 2, and you have to connect to a router.

Additional References

- Related Documents, page 17
- Standards, page 18
- MIBs, page 18
- RFCs, page 18
- Technical Assistance, page 18

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Security commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Security Command Reference*
Authentication, Authorization, and Accounting	Configuring Authentication , Configuring Authorization , and Configuring Accounting feature modules.
Configuring Secure Shell, a host name and host domain	Configuring Secure Shell feature module.

Related Topic	Document Title
Debugging commands	*Cisco IOS Debug Command Reference*
IPsec	Configuring Security for VPNs with IPsec feature module.
SNMP, configuring traps	Configuring SNMP Support feature module.

Standards

Standards	Title
Internet Engineering Task Force (IETF) Secure Shell Version 2 Draft Standards	http://www.ietf.org/ Internet Engineering Task Force website.

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
None	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Secure Shell Version 2 Support

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1 Feature Information for Secure Shell Version 2 Support

Feature Name	Releases	Feature Information
Secure Shell Version 2 Support	Cisco IOS XE Release 2.1	The Secure Shell Version 2 Support feature allows you to configure Secure Shell (SSH) Version 2. SSH runs on top of a reliable transport layer and provides strong authentication and encryption capabilities. In Cisco IOS Release 2.4, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. The following commands were introduced or modified by this feature: **debug ip ssh**, **ip ssh min dh size**, **ip ssh rsa keypair-name**, **ip ssh version**, **ssh**.
Secure Shell Version 2 Enhancements	Cisco IOS XE Release 2.1	The Secure Shell Version 2 Enhancements include a number of additional capabilities such as support for VRF aware SSH, SSH debug enhancements, and Diffie-Hellman group 14 and group 16 exchange support. In Cisco IOS Release 2.4, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers.