CONTENTS

CHAPTER 1

Configuring Secure Shell 1

Finding Feature Information 1
Prerequisites for Configuring SSH 2
Restrictions for Configuring SSH 2
Information About Secure Shell (SSH) 3

SSH Server 3
SSH Integrated Client 3
RSA Authentication Support 3

How to Configure SSH 4

Configuring an SSH Server 4
Invoking an SSH Client 5

Troubleshooting Tips 6

Configuration Examples for SSH 6

Example SSH on a Cisco 7200 Series Router 6
Example SSH on a Cisco 7500 Series Router 7
Example SSH on a Cisco 12000 Series Router 9
Example: Verifying SSH 10

Additional References 11

Feature Information for Configuring Secure Shell 12

CHAPTER 2

Reverse SSH Enhancements 13

Finding Feature Information 13
Prerequisites for Reverse SSH Enhancements 13
Restrictions for Reverse SSH Enhancements 14
Information About Reverse SSH Enhancements 14

Reverse Telnet 14
Reverse SSH 14

How to Configure Reverse SSH Enhancements 14
CHAPTER 3

Secure Shell Version 2 Support 25
Finding Feature Information 25
Prerequisites for Secure Shell Version 2 Support 26
Restrictions for Secure Shell Version 2 Support 26
Information About Secure Shell Version 2 Support 26
  Secure Shell Version 2 26
  Secure Shell Version 2 Enhancements 27
  Secure Shell Version 2 Enhancements for RSA Keys 27
  SNMP Trap Generation 28
  SSH Keyboard Interactive Authentication 29
How to Configure Secure Shell Version 2 Support 29
  Configuring a Device for SSH Version 2 Using a Hostname and Domain Name 29
  Configuring a Device for SSH Version 2 Using RSA Key Pairs 31
  Configuring the Cisco SSH Server to Perform RSA-Based User Authentication 32
  Configuring the Cisco IOS SSH Client to Perform RSA-Based Server Authentication 34
  Starting an Encrypted Session with a Remote Device 36
    Troubleshooting Tips 37
  Enabling Secure Copy Protocol on the SSH Server 37
Verifying the Status of the Secure Shell Connection 39
Verifying the Secure Shell Status 41
Monitoring and Maintaining Secure Shell Version 2 42
Configuration Examples for Secure Shell Version 2 Support 45
  Example: Configuring Secure Shell Version 1 45
  Example: Configuring Secure Shell Version 2 45
  Example: Configuring Secure Shell Versions 1 and 2 45
  Example: Starting an Encrypted Session with a Remote Device 45
  Example: Configuring Server-Side SCP 45
  Example: Setting an SNMP Trap 46
Examples: SSH Keyboard Interactive Authentication 46
  Example: Enabling Client-Side Debugs 46
  Example: Enabling ChPass with a Blank Password Change 47
  Example: Enabling ChPass and Changing the Password on First Login 47
  Example: Enabling ChPass and Expiring the Password After Three Logins 47
  Example: SNMP Debugging 48
Examples: SSH Debugging Enhancements 48
Additional References for Secure Shell Version 2 Support 49
Feature Information for Secure Shell Version 2 Support 50
Chapter 1

Configuring Secure Shell

The Secure Shell (SSH) feature is an application and a protocol that provides a secure replacement to the Berkeley r-tools. The protocol secures sessions using standard cryptographic mechanisms, and the application can be used similarly to the Berkeley rexec and rsh tools. Two versions of SSH are available: SSH Version 1 and SSH Version 2. Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only. For information about SSH Version 2, see the “Secure Shell Version 2 Support” feature module.

Finding Feature Information, page 1
Prerequisites for Configuring SSH, page 2
Restrictions for Configuring SSH, page 2
Information About Secure Shell (SSH), page 3
How to Configure SSH, page 4
Configuration Examples for SSH, page 6
Additional References, page 11
Feature Information for Configuring Secure Shell, page 12

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and 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 module.

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 Configuring SSH

Note

Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

- Download the required image on the device. The Secure Shell (SSH) server requires an IPsec (Data Encryption Standard [DES] or 3DES) encryption software image; the SSH client requires an IPsec (DES or 3DES) encryption software image. For information about downloading a software image, see the Loading and Managing System Images Configuration Guide.

- Configure a hostname and host domain for your device by using the hostname and ip domain-name commands in global configuration mode.

- Generate a Rivest, Shamir, and Adleman (RSA) key pair for your device. This key pair automatically enables SSH and remote authentication when the crypto key generate rsa command is entered in global configuration mode.

Note

To delete the RSA key pair, use the crypto key zeroize rsa global configuration command. Once you delete the RSA key pair, you automatically disable the SSH server.

- Configure user authentication for local or remote access. You can configure authentication with or without authentication, authorization, and accounting (AAA). For more information, see the Authentication, Authorization, and Accounting Configuration Guide.

Restrictions for Configuring SSH

Note

Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

- The Secure Shell (SSH) server and SSH client are supported on Data Encryption Standard (DES) (56-bit) and 3DES (168-bit) data encryption software images only. In DES software images, DES is the only encryption algorithm available. In 3DES software images, both DES and 3DES encryption algorithms are available.

- Execution shell is the only application supported.

- The login banner is not supported in Secure Shell Version 1. It is supported in Secure Shell Version 2.
Information About Secure Shell (SSH)

Note
Unless otherwise noted, the term "SSH" denotes "SSH Version 1" only.

SSH Server

Note
Unless otherwise noted, the term "SSH" denotes "SSH Version 1" only.

The Secure Shell (SSH) Server feature enables an SSH client to make a secure, encrypted connection to a Cisco device. This connection provides functionality that is similar to that of an inbound Telnet connection. Before SSH, security was limited to Telnet security. SSH allows a strong encryption to be used with the Cisco software authentication. The SSH server in Cisco software works with publicly and commercially available SSH clients.

SSH Integrated Client

Note
Unless otherwise noted, the term "SSH" denotes "SSH Version 1" only.

The Secure Shell (SSH) Integrated Client feature is an application that runs over the SSH protocol to provide device authentication and encryption. The SSH client enables a Cisco device to make a secure, encrypted connection to another Cisco device or to any other device running the SSH server. This connection provides functionality similar to that of an outbound Telnet connection except that the connection is encrypted. With authentication and encryption, the SSH client allows for secure communication over an unsecured network.

The SSH client in Cisco software works with publicly and commercially available SSH servers. The SSH client supports the ciphers of Data Encryption Standard (DES), 3DES, and password authentication. User authentication is performed like that in the Telnet session to the device. The user authentication mechanisms supported for SSH are RADIUS, TACACS+, and the use of locally stored usernames and passwords.

Note
The SSH client functionality is available only when the SSH server is enabled.

RSA Authentication Support

Rivest, Shamir, and Adleman (RSA) authentication available in Secure Shell (SSH) clients is not supported on the SSH server for Cisco software by default. For more information about RSA authentication support, see the "Configuring a Router for SSH Version 2 Using RSA Pairs" section of the "Secure Shell Version 2 Support" module.
How to Configure SSH

Configuring an SSH Server

Note Unless otherwise noted, the term "SSH" denotes "SSH Version 1" only.

SUMMARY STEPS

1. enable
2. configure terminal
3. ip ssh {timeout seconds | authentication-retries integer}
4. ip ssh rekey {time time | volume volume}
5. exit
6. show ip ssh

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>Example: Device&gt; enable</td>
<td>• Enter your password if prompted.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example: Device# configure terminal</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> ip ssh {timeout seconds</td>
<td>authentication-retries integer}</td>
</tr>
<tr>
<td>Example: Device(config)# ip ssh timeout 30</td>
<td>• This command can also be used to establish the number of password prompts provided to the user. The number is the lower of the following two values:</td>
</tr>
<tr>
<td></td>
<td>• Value proposed by the client using the ssh -o numberofpasswordprompt command.</td>
</tr>
<tr>
<td></td>
<td>• Value configured on the device using the ip ssh authentication-retries integer command, plus one.</td>
</tr>
</tbody>
</table>
### Configuring Secure Shell

#### Invoking an SSH Client

**Note**

Unless otherwise noted, the term "SSH" denotes "SSH Version 1" only.

Perform this task to invoke the Secure Shell (SSH) client. The SSH client runs in user EXEC mode and has no specific configuration tasks.

**SUMMARY STEPS**

1. **enable**
2. **ssh -l username -vrf vrf-name ip-address**

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Device&gt; enable</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ssh -l username -vrf vrf-name ip-address</td>
<td>Invokes the SSH client to connect to an IP host or address in the specified virtual routing and forwarding (VRF) instance.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Device# ssh -l user1 -vrf vrf1 192.0.2.1</td>
<td></td>
</tr>
</tbody>
</table>
Troubleshooting Tips

Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

- If your Secure Shell (SSH) configuration commands are rejected as illegal commands, you have not successfully generated an Rivest, Shamir, and Adleman (RSA) key pair for your device. Make sure that you have specified a hostname and domain. Then use the **crypto key generate rsa** command to generate an RSA key pair and enable the SSH server.

- When configuring the RSA key pair, you might encounter the following error messages:
  - No hostname specified.
    You must configure a hostname for the device using the **hostname** global configuration command. See the “IPsec and Quality of Service” module for more information.
  - No domain specified.
    You must configure a host domain for the device using the **ip domain-name** global configuration command. See the “IPsec and Quality of Service” module for more information.

- The number of allowable SSH connections is limited to the maximum number of vty’s configured for the device. Each SSH connection uses a vty resource.

- SSH uses either local security or the security protocol that is configured through AAA on your device for user authentication. When configuring Authentication, Authorization, and Accounting (AAA), you must ensure that AAA is disabled on the console for user authentication. AAA authorization is disabled on the console by default. If AAA authorization is enabled on the console, disable it by configuring the **no aaa authorization console** command during the AAA configuration stage.

Configuration Examples for SSH

Example SSH on a Cisco 7200 Series Router

In the following example, SSH is configured on a Cisco 7200 with a timeout that is not to exceed 60 seconds and no more than 2 authentication retries. Before the SSH server feature is configured on the router, TACACS+ is specified as the method of authentication.

```
hostname Router72K
aaa new-model
aaa authentication login default tacacs+
aaa authentication login aaa7200kw none
enable password password
username username1 password 0 password1
username username2 password 0 password2
ip subnet-zero
no ip domain-lookup
```
ip domain-name cisco.com
! Enter the ssh commands.
ip ssh timeout 60
ip ssh authentication-retries 2
controller E1 2/0
center E1 2/1
interface Ethernet1/0
ip address 192.168.110.2 255.255.255.0 secondary
ip address 192.168.109.2 255.255.255.0
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
no keepalive
no cdp enable
interface Ethernet1/1
no ip address
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
shutdown
no cdp enable
interface Ethernet1/2
no ip address
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
shutdown
no cdp enable
no ip classless
ip route 192.168.1.0 255.255.255.0 10.1.10.1
ip route 192.168.9.0 255.255.255.0 10.1.1.1
ip route 192.168.10.0 255.255.255.0 10.1.1.1
map-list atm
ip 10.1.10.1 atm-vc 7 broadcast
no cdp run
tacacs-server host 192.168.109.216 port 9000
tacacs-server key cisco
radius-server host 192.168.109.216 auth-port 1650 acct-port 1651
radius-server key cisco
line con 0
exec-timeout 0 0
login authentication aaa7200kw
transport input none
line aux 0
line vty 0 4
password password
end

Example SSH on a Cisco 7500 Series Router

In the following example, SSH is configured on a Cisco 7500 with a timeout that is not to exceed 60 seconds and no more than 5 authentication retries. Before the SSH server feature is configured on the router, RADIUS is specified as the method of authentication.

hostname Router75K
aaa new-model
aaa authentication login default radius
aaa authentication login aaa7500kw none
enable password password

username username1 password 0 password1
username username2 password 0 password2
ip subnet-zero
no ip cef
no ip domain-lookup
ip domain-name cisco.com
! Enter ssh commands.
ip ssh timeout 60
ip ssh authentication-retries 5
controller E1 3/0
channel-group 0 timeslots 1
controller E1 3/1
channel-group 0 timeslots 1
channel-group 1 timeslots 2
interface Ethernet0/0/0
no ip address
no ip directed-broadcast
no ip route-cache distributed
shutdown
interface Ethernet0/0/1
no ip address
no ip directed-broadcast
no ip route-cache distributed
shutdown
interface Ethernet0/0/2
no ip address
no ip directed-broadcast
no ip route-cache distributed
shutdown
interface Ethernet0/0/3
no ip address
no ip directed-broadcast
no ip route-cache distributed
shutdown
interface Ethernet1/0
ip address 192.168.110.2 255.255.255.0 secondary
ip address 192.168.109.2 255.255.255.0
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
interface Ethernet1/1
ip address 192.168.109.2 255.255.255.0
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
shutdown
interface Ethernet1/2
no ip address
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
interface Ethernet1/3
no ip address
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
shutdown
interface Ethernet1/4
no ip address
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
shutdown
interface Ethernet1/5
no ip address
no ip directed-broadcast
no ip route-cache
no ip mroute-cache
shutdown
interface Serial2/0
ip address 10.1.1.2 255.0.0.0
no ip directed-broadcast
Example SSH on a Cisco 12000 Series Router

In the following example, SSH is configured on a Cisco 12000 with a timeout that is not to exceed 60 seconds and no more than two authentication retries. Before the SSH server feature is configured on the router, TACACS+ is specified as the method of authentication.

```
hostname Router12K
aaa new-model
aaa authentication login default tacacs+ local
aaa authentication login aaa12000kw local
enable password password
username username1 password 0 password1
username username2 password 0 password2
redundancy
main-cpu
auto-sync startup-config
ip subnet-zero
no ip domain-lookup
ip domain-name cisco.com
! Enter ssh commands.
ip ssh timeout 60
ip ssh authentication-retries 2
interface ATM0/0
no ip address
no ip directed-broadcast
no ip route-cache cef
shutdown
interface POS1/0
ip address 10.100.100.2 255.255.255.0
no ip directed-broadcast
encapsulation ppp
no ip route-cache cef
no keepalive
crc 16
do cdp enable
interface POS1/1
no ip address
no ip directed-broadcast
no ip route-cache cef
shutdown
crc 32
interface POS1/2
no ip address
no ip directed-broadcast
no ip route-cache cef
```
shutdown
crc 32

interface POS1/3
no ip address
no ip directed-broadcast
no ip route-cache cef
shutdown
crc 32

interface POS2/0
ip address 10.1.1.1 255.255.255.0
no ip directed-broadcast
encapsulation ppp
no ip route-cache cef
crc 16

interface Ethernet0
ip address 172.17.110.91 255.255.255.224
no ip directed-broadcast

router ospf 1
network 0.0.0.0 255.255.255.255 area 0.0.0.0
ip classless
ip route 0.0.0.0 0.0.0.0 172.17.110.65

logging trap debugging
tacacs-server host 172.17.116.138
tacacs-server key cisco

radius-server host 172.17.116.138 auth-port 1650 acct-port 1651
radius-server key cisco

line con 0
eexec-timeout 0 0
login authentication aal2000kw
transport input none
line aux 0
line vty 0 4

no scheduler max-task-time
no exception linecard slot 0 sge-registers
no exception linecard slot 1 sge-registers
no exception linecard slot 2 sge-registers
no exception linecard slot 3 sge-registers
no exception linecard slot 4 sge-registers
no exception linecard slot 5 sge-registers
no exception linecard slot 6 sge-registers
end

Example: Verifying SSH

Note

Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

To verify that the Secure Shell (SSH) server is enabled and to display the version and configuration data for your SSH connection, use the show ip ssh command. The following example shows that SSH is enabled:

Device# show ip ssh
SSH Enabled - version 1.5
Authentication timeout: 120 secs; Authentication retries: 3
The following example shows that SSH is disabled:

Device# show ip ssh
%SSH has not been enabled
To verify the status of your SSH server connections, use the show ssh command. The following example shows the SSH server connections on the device when SSH is enabled:

Device# show ssh
Connection    Version   Encryption State    Username
0 1.5 3DES Session Started guest
The following example shows that SSH is disabled:

Device# show ssh
%No SSH server connections running.

Additional References

Related Documents

<table>
<thead>
<tr>
<th>Related Topic</th>
<th>Document Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS commands</td>
<td>Cisco IOS Master Command List, All Releases</td>
</tr>
<tr>
<td>Authentication, authorization, and accounting (AAA)</td>
<td>Authentication, Authorization, and Accounting Configuration Guide</td>
</tr>
<tr>
<td>IPsec</td>
<td>&quot;IPsec and Quality of Service&quot; module</td>
</tr>
<tr>
<td>SSH Version 2</td>
<td>“Secure Shell Version 2 Support” module</td>
</tr>
<tr>
<td>Downloading a software image</td>
<td>Loading and Managing System Images Configuration Guide</td>
</tr>
</tbody>
</table>

Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
<td><a href="http://www.cisco.com/cisco/web/support/index.html">http://www.cisco.com/cisco/web/support/index.html</a></td>
</tr>
</tbody>
</table>
Feature Information for Configuring Secure Shell

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 Configuring Secure Shell

<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Releases</th>
<th>Feature Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure Shell</td>
<td>Cisco IOS 15.0(2)SE</td>
<td>The Secure Shell (SSH) feature is an application and a protocol that provides a secure replacement to the Berkeley r-tools. The protocol secures sessions using standard cryptographic mechanisms, and the application can be used similarly to the Berkeley rexec and rsh tools. Two versions of SSH are available: SSH Version 1 and SSH Version 2. This document describes SSH Version 1. This document also includes information about the Secure Shell SSH Version 1 Integrated Client feature and the Secure Shell SSH Version 1 Server Support feature. Both features are part of the Secure Shell functionality.</td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 15.2(1)E</td>
<td></td>
</tr>
</tbody>
</table>
Reverse SSH Enhancements

The Reverse SSH Enhancements feature, which is supported for SSH Version 1 and 2, provides an alternative way to configure reverse Secure Shell (SSH) so that separate lines do not need to be configured for every terminal or auxiliary line on which SSH must be enabled. This feature also eliminates the rotary-group limitation.

- Finding Feature Information, page 13
- Prerequisites for Reverse SSH Enhancements, page 13
- Restrictions for Reverse SSH Enhancements, page 14
- Information About Reverse SSH Enhancements, page 14
- How to Configure Reverse SSH Enhancements, page 14
- Configuration Examples for Reverse SSH Enhancements, page 20
- Additional References, page 21
- Feature Information for Reverse SSH Enhancements, page 23

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and 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 module.

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 Reverse SSH Enhancements

- SSH must be enabled.
- The SSH client and server must be running the same version of SSH.
Restrictions for Reverse SSH Enhancements

- The `-I` keyword and `userid : {number} \{ip-address\}` delimiter and arguments are mandatory when configuring the alternative method of Reverse SSH for console access.

Information About Reverse SSH Enhancements

Reverse Telnet

Reverse telnet allows you to telnet to a certain port range and connect to terminal or auxiliary lines. Reverse telnet has often been used to connect a Cisco device that has many terminal lines to the consoles of other Cisco devices. Telnet makes it easy to reach the device console from anywhere simply by telnet to the terminal server on a specific line. This telnet approach can be used to configure a device even if all network connectivity to that device is disconnected. Reverse telnet also allows modems that are attached to Cisco devices to be used for dial-out (usually with a rotary device).

Reverse SSH

Reverse telnet can be accomplished using SSH. Unlike reverse telnet, SSH provides for secure connections. The Reverse SSH Enhancements feature provides you with a simplified method of configuring SSH. Using this feature, you no longer have to configure a separate line for every terminal or auxiliary line on which you want to enable SSH. The previous method of configuring reverse SSH limited the number of ports that can be accessed to 100. The Reverse SSH Enhancements feature removes the port number limitation. For information on the alternative method of configuring reverse SSH, see How to Configure Reverse SSH Enhancements, on page 14.

How to Configure Reverse SSH Enhancements

Configuring Reverse SSH for Console Access

To configure reverse SSH console access on the SSH server, perform the following steps.
### SUMMARY STEPS

1. `enable`
2. `configure terminal`
3. `line line-number ending-line-number`
4. `no exec`
5. `login authentication listname`
6. `transport input ssh`
7. `exit`
8. `exit`
9. `ssh -l userid : {number} {ip-address}`

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device&gt; enable</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device# configure terminal</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> line line-number ending-line-number</td>
<td>Identifies a line for configuration and enters line configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device# line 1 3</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> no exec</td>
<td>Disables EXEC processing on a line.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device(config-line)# no exec</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> login authentication listname</td>
<td>Defines a login authentication mechanism for the lines.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device(config-line)# login authentication default</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The authentication method must use a username and password.
## Command or Action

<table>
<thead>
<tr>
<th>Step 6</th>
<th>transport input ssh</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example:</td>
<td></td>
<td>Defines which protocols to use to connect to a specific line of the device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- The <code>ssh</code> keyword must be used for the Reverse SSH Enhancements feature.</td>
</tr>
</tbody>
</table>

### Step 7

| exit | Exits line configuration mode. |

### Step 8

| exit | Exits global configuration mode. |

### Step 9

| ssh -l userid : {number} {ip-address} | Specifies the user ID to use when logging in on the remote networking device that is running the SSH server. |

### Note

The `userid` argument and `:rotary {number} {ip-address}` delimiter and arguments are mandatory when configuring the alternative method of Reverse SSH for modem access.

---

### Configuring Reverse SSH for Modem Access

To configure Reverse SSH for modem access, perform the steps shown in the “SUMMARY STEPS” section below.

In this configuration, reverse SSH is being configured on a modem used for dial-out lines. To get any of the dial-out modems, you can use any SSH client and start a SSH session as shown (in Step 10) to get to the next available modem from the rotary device.
## SUMMARY STEPS

1. `enable`
2. `configure terminal`
3. `line line-number ending-line-number`
4. `no exec`
5. `login authentication listname`
6. `rotary group`
7. `transport input ssh`
8. `exit`
9. `exit`
10. `ssh -l userid :rotary {number} {ip-address}`

## DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 1** enable | Enables privileged EXEC mode.  
  - Enter your password if prompted. |
| **Example:** Device> enable | |
| **Step 2** configure terminal | Enters global configuration mode. |
| **Example:** Device# configure terminal | |
| **Step 3** line line-number ending-line-number | Identifies a line for configuration and enters line configuration mode. |
| **Example:** Device# line 1 200 | |
| **Step 4** no exec | Disables EXEC processing on a line. |
| **Example:** Device(config-line)# no exec | |
| **Step 5** login authentication listname | Defines a login authentication mechanism for the lines.  
  **Note** The authentication method must use a username and password. |
<p>| <strong>Example:</strong> Device(config-line)# login authentication default | |</p>
<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 6</strong></td>
<td><strong>rotary group</strong></td>
</tr>
<tr>
<td>Example:</td>
<td>Device(config-line)# rotary 1</td>
</tr>
<tr>
<td></td>
<td>Defines a group of lines consisting of one or more virtual terminal lines or one auxiliary port line.</td>
</tr>
<tr>
<td><strong>Step 7</strong></td>
<td><strong>transport input ssh</strong></td>
</tr>
<tr>
<td>Example:</td>
<td>Device(config-line)# transport input ssh</td>
</tr>
<tr>
<td></td>
<td>Defines which protocols to use to connect to a specific line of the device.</td>
</tr>
<tr>
<td></td>
<td>• The ssh keyword must be used for the Reverse SSH Enhancements feature.</td>
</tr>
<tr>
<td><strong>Step 8</strong></td>
<td><strong>exit</strong></td>
</tr>
<tr>
<td>Example:</td>
<td>Device(config-line)# exit</td>
</tr>
<tr>
<td></td>
<td>Exits line configuration mode.</td>
</tr>
<tr>
<td><strong>Step 9</strong></td>
<td><strong>exit</strong></td>
</tr>
<tr>
<td>Example:</td>
<td>Device(config)# exit</td>
</tr>
<tr>
<td></td>
<td>Exits global configuration mode.</td>
</tr>
<tr>
<td><strong>Step 10</strong></td>
<td><strong>ssh -l userid :rotary {number} {ip-address}</strong></td>
</tr>
<tr>
<td>Example:</td>
<td>Device# ssh -l lab:rotary1 router.example.com</td>
</tr>
<tr>
<td></td>
<td>Specifies the user ID to use when logging in on the remote networking device that is running the SSH server.</td>
</tr>
<tr>
<td></td>
<td>• userid --User ID.</td>
</tr>
<tr>
<td></td>
<td>• : --Signifies that a port number and terminal IP address will follow the userid argument.</td>
</tr>
<tr>
<td></td>
<td>• number --Terminal or auxiliary line number.</td>
</tr>
<tr>
<td></td>
<td>• ip-address --Terminal server IP address.</td>
</tr>
<tr>
<td><strong>Note</strong></td>
<td>The userid argument and :rotary {number} {ip-address} delimiter and arguments are mandatory when configuring the alternative method of Reverse SSH for modem access.</td>
</tr>
</tbody>
</table>

**Troubleshooting Reverse SSH on the Client**

To troubleshoot the reverse SSH configuration on the client (remote device), perform the following steps.

**SUMMARY STEPS**

1. enable
2. debug ip ssh client
DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| Step 1 | enable | Enables privileged EXEC mode.  
  • Enter your password if prompted. |
| Example: | Device> enable | |
| Step 2 | debug ip ssh client | Displays debugging messages for the SSH client. |
| Example: | Device# debug ip ssh client | |

Troubleshooting Reverse SSH on the Server

To troubleshoot the reverse SSH configuration on the terminal server, perform the following steps. The steps may be configured in any order or independent of one another.

SUMMARY STEPS

1. enable
2. debug ip ssh
3. show ssh
4. show line

DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| Step 1 | enable | Enables privileged EXEC mode.  
  • Enter your password if prompted. |
<p>| Example: | Device&gt; enable | |
| Step 2 | debug ip ssh | Displays debugging messages for the SSH server. |
| Example: | Device# debug ip ssh | |</p>
<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 3</strong></td>
<td>show ssh</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>Device# show ssh</td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>show line</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>Device# show line</td>
</tr>
</tbody>
</table>

### Configuration Examples for Reverse SSH Enhancements

#### Example Reverse SSH Console Access

The following configuration example shows that reverse SSH has been configured for console access for terminal lines 1 through 3:

**Terminal Server Configuration**

```
line 1 3
no exec
login authentication default
transport input ssh
```

**Client Configuration**

The following commands configured on the SSH client will form the reverse SSH session with lines 1, 2, and 3, respectively:

```
ssh -l lab:1 router.example.com
ssh -l lab:2 router.example.com
ssh -l lab:3 router.example.com
```

#### Example Reverse SSH Modem Access

The following configuration example shows that dial-out lines 1 through 200 have been grouped under rotary group 1 for modem access:

```
line 1 200
no exec
login authentication default
rotary 1
transport input ssh
exit
```
The following command shows that reverse SSH will connect to the first free line in the rotary group:

```
ssh -l lab:rotary1 router.example.com
```

## Additional References

### Related Documents

<table>
<thead>
<tr>
<th>Related Topic</th>
<th>Document Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS commands</td>
<td>Cisco IOS Master Commands List, All Releases</td>
</tr>
<tr>
<td>Configuring Secure Shell</td>
<td>Secure Shell Configuration Guide</td>
</tr>
<tr>
<td>Security commands</td>
<td>Cisco IOS Security Command Reference</td>
</tr>
</tbody>
</table>

### Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
<td><a href="http://www.cisco.com/cisco/web/support/index.html">http://www.cisco.com/cisco/web/support/index.html</a></td>
</tr>
</tbody>
</table>

### Related Documents

<table>
<thead>
<tr>
<th>Related Topic</th>
<th>Document Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS commands</td>
<td>Cisco IOS Master Commands List, All Releases</td>
</tr>
<tr>
<td>Configuring Secure Shell</td>
<td>Secure Shell Configuration Guide</td>
</tr>
<tr>
<td>Security commands</td>
<td>Cisco IOS Security Command Reference</td>
</tr>
</tbody>
</table>
Standards

<table>
<thead>
<tr>
<th>Standards</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>No new or modified standards are supported by this feature.</td>
<td>--</td>
</tr>
</tbody>
</table>

MIBs

<table>
<thead>
<tr>
<th>MIBs</th>
<th>MIBs Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: <a href="http://www.cisco.com/go/mibs">http://www.cisco.com/go/mibs</a></td>
</tr>
</tbody>
</table>

RFCs

<table>
<thead>
<tr>
<th>RFCs</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>--</td>
</tr>
</tbody>
</table>

Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
<td><a href="http://www.cisco.com/cisco/web/support/index.html">http://www.cisco.com/cisco/web/support/index.html</a></td>
</tr>
</tbody>
</table>
Feature Information for Reverse SSH Enhancements

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 2: Feature Information for Reverse SSH Enhancements

<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Releases</th>
<th>Feature Information</th>
</tr>
</thead>
</table>
| Reverse SSH Enhancements | Cisco IOS 15.2(1)E | The Reverse SSH Enhancements feature, which is supported for SSH Version 1 and 2, provides an alternative way to configure reverse Secure Shell (SSH) so that separate lines do not need to be configured for every terminal or auxiliary line on which SSH must be enabled. This feature also eliminates the rotary-group limitation. The following command was introduced: **ssh**.
Secure Shell Version 2 Support

The Secure Shell Version 2 Support feature allows you to configure Secure Shell (SSH) Version 2. (SSH Version 1 support was implemented in an earlier Cisco software release.) SSH runs on top of a reliable transport layer and provides strong authentication and encryption capabilities. 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 allows for the secure transfer of files.

- Finding Feature Information, page 25
- Prerequisites for Secure Shell Version 2 Support, page 26
- Restrictions for Secure Shell Version 2 Support, page 26
- Information About Secure Shell Version 2 Support, page 26
- How to Configure Secure Shell Version 2 Support, page 29
- Configuration Examples for Secure Shell Version 2 Support, page 45
- Additional References for Secure Shell Version 2 Support, page 49
- Feature Information for Secure Shell Version 2 Support, page 50

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and 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 module.

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

- Before configuring SSH, ensure that the required image is loaded on your device. The SSH server requires you to have a k9 (Triple Data Encryption Standard [3DES]) software image depending on your release.
- You have to use a SSH remote device that supports SSH Version 2 and connect to a Cisco device.
- SCP relies on authentication, authorization, and accounting (AAA) to function correctly. Therefore, AAA must be configured on the device to enable the secure copy protocol on the SSH Server.

**Note**

The SSH Version 2 server and the SSH Version 2 client are supported on your Cisco software, depending on your release. (The SSH client runs both the SSH Version 1 protocol and the SSH Version 2 protocol. The SSH client is supported in both k8 and k9 images depending on your release.)

For more information about downloading a software image, refer to the *Configuration Fundamentals Configuration Guide*.

Restrictions for Secure Shell Version 2 Support

- Secure Shell (SSH) servers and SSH clients are supported in Triple Data Encryption Standard (3DES) software images.
- Execution Shell, remote command execution, and Secure Copy Protocol (SCP) are the only applications supported.
- Rivest, Shamir, and Adleman (RSA) key generation is an SSH server-side requirement. Devices that act as SSH clients need not generate RSA keys.
- The RSA key pair size must be greater than or equal to 768 bits.
- The following features are not supported:
  - Port forwarding
  - Compression

Information About Secure Shell Version 2 Support

Secure Shell Version 2


The configuration for the SSH Version 2 server is similar to the configuration for SSH Version 1. The `ip ssh version` command defines the SSH version to be configured. 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.
SSH Version 1 is a protocol that has never been defined in a standard. If you do not want your device 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 enables an SSH connection using the Rivest, Shamir, and Adleman (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). This behavior still exists, but by using the `ip ssh rsa keypair-name` command, you can overcome this 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 hostname and a domain name, which was required in SSH Version 1 of the Cisco software.

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

---

**Secure Shell Version 2 Enhancements**

The SSH Version 2 Enhancements feature includes a number of additional capabilities such as supporting Virtual Routing and Forwarding (VRF)-Aware SSH, SSH debug enhancements, and Diffie-Hellman (DH) group exchange support.

The VRF-Aware SSH feature is supported depending on your release.

The Cisco SSH implementation has traditionally used 768-bit modulus, but with an increasing need for higher key sizes to accommodate DH Group 14 (2048 bits) and Group 16 (4096 bits) cryptographic applications, a message exchange between the client and the server to establish the favored DH group becomes necessary. The `ip ssh dh min size` command configures the modulus size on the SSH server. In addition to this, the `ssh` command was extended to add VRF awareness to the 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 was enhanced by modifying SSH debug commands. The `debug ip ssh` command was extended to simplify the debugging process. Before the simplification of the debugging process, 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.

**Secure Shell Version 2 Enhancements for RSA Keys**

Cisco SSH Version 2 supports keyboard-interactive and password-based authentication methods. The SSH Version 2 Enhancements for RSA Keys feature also supports RSA-based public key authentication for the client and the server.

User authentication—RSA-based user authentication uses a private/public key pair associated with each user for authentication. The user must generate a private/public key pair on the client and configure a public key on the Cisco SSH server to complete the authentication.
An SSH user trying to establish credentials provides an encrypted signature using the private key. The signature and the user’s public key are sent to the SSH server for authentication. The SSH server computes a hash over the public key provided by the user. The hash is used to determine if the server has a matching entry. If a match is found, an RSA-based message verification is performed using the public key. Hence, the user is authenticated or denied access based on the encrypted signature.

Server authentication—While establishing an SSH session, the Cisco SSH client authenticates the SSH server by using the server host keys available during the key exchange phase. SSH server keys are used to identify the SSH server. These keys are created at the time of enabling SSH and must be configured on the client.

For server authentication, the Cisco SSH client must assign a host key for each server. When the client tries to establish an SSH session with a server, the client receives the signature of the server as part of the key exchange message. If the strict host key checking flag is enabled on the client, the client checks if it has the host key entry corresponding to the server. If a match is found, the client tries to validate the signature by using the server host key. If the server is successfully authenticated, the session establishment continues; otherwise, it is terminated and displays a “Server Authentication Failed” message.

**Note**

Storing public keys on a server uses memory; therefore, the number of public keys configurable on an SSH server is restricted to ten users, with a maximum of two public keys per user.

**Note**

RSA-based user authentication is supported by the Cisco server, but Cisco clients cannot propose public key as an authentication method. If the Cisco server receives a request from an open SSH client for RSA-based authentication, the server accepts the authentication request.

**Note**

For server authentication, configure the RSA public key of the server manually and configure the `ip ssh stricthostkeycheck` command on the Cisco SSH client.

### SNMP Trap Generation

Depending on your release, Simple Network Management Protocol (SNMP) traps are generated automatically when an SSH session terminates if the traps have been enabled and SNMP debugging has been enabled. For information about enabling SNMP traps, see the “Configuring SNMP Support” module in the *SNMP Configuration Guide*.

**Note**

When you configure 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. For an example of an SNMP trap generation configuration, see the "**" section.

You must also enable 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. For an example of SNMP debugging, see the “**Example: SNMP Debugging**” section.

SSH Keyboard Interactive Authentication

The SSH Keyboard Interactive Authentication feature, also known as Generic Message Authentication for SSH, is a method that can be used to implement different types of authentication mechanisms. Basically, any currently supported authentication method that requires only user input can be performed with this feature. The feature is automatically enabled.

The following methods are supported:

- Password
- SecurID and hardware tokens printing a number or a string in response to a challenge sent by the server
- Pluggable Authentication Module (PAM)
- S/KEY (and other One-Time-Pads)

For examples of various scenarios in which the SSH Keyboard Interactive Authentication feature has been automatically enabled, see the “Examples: SSH Keyboard Interactive Authentication, on page 46” section.

How to Configure Secure Shell Version 2 Support

Configuring a Device for SSH Version 2 Using a Hostname and Domain Name

**SUMMARY STEPS**

1. enable
2. configure terminal
3. hostname name
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]
8. exit

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>- Enter your password if prompted.</td>
</tr>
<tr>
<td>Device&gt; enable</td>
<td></td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
</tr>
<tr>
<td><strong>Step 2</strong> configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> Device# configure terminal</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> hostname <em>name</em></td>
<td>Configures a hostname for your device.</td>
</tr>
<tr>
<td><strong>Example:</strong> Device(config)# hostname cisco7200</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> ip domain-name <em>name</em></td>
<td>Configures a domain name for your device.</td>
</tr>
<tr>
<td><strong>Example:</strong> cisco7200(config)# ip domain-name example.com</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> crypto key generate rsa</td>
<td>Enables the SSH server for local and remote authentication.</td>
</tr>
<tr>
<td><strong>Example:</strong> cisco7200(config)# crypto key generate rsa</td>
<td></td>
</tr>
<tr>
<td><strong>Step 6</strong> ip ssh [time-out <em>seconds</em></td>
<td>(Optional) Configures SSH control variables on your device.</td>
</tr>
<tr>
<td></td>
<td>authentication-retries <em>integer</em>]</td>
</tr>
<tr>
<td><strong>Example:</strong> cisco7200(config)# ip ssh time-out 120</td>
<td></td>
</tr>
<tr>
<td><strong>Step 7</strong> ip ssh version [1</td>
<td>(Optional) Specifies the version of SSH to be run on your device.</td>
</tr>
<tr>
<td></td>
<td>2]</td>
</tr>
<tr>
<td><strong>Example:</strong> cisco7200(config)# ip ssh version 1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 8</strong> exit</td>
<td>Exits global configuration mode and enters privileged EXEC mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> cisco7200(config)# exit</td>
<td>• Use <strong>no hostname</strong> command to return to the default host.</td>
</tr>
</tbody>
</table>
Configuring a Device for SSH Version 2 Using RSA Key Pairs

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
7. exit

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>Example: Device&gt; enable</td>
<td>• Enter your password if prompted.</td>
</tr>
<tr>
<td><strong>Step 2</strong> configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example: Device# configure terminal</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> ip ssh rsa keypair-name keypair-name</td>
<td>Specifies the RSA key pair to be used for SSH.</td>
</tr>
<tr>
<td>Example: Device(config)# ip ssh rsa keypair-name sshkeys</td>
<td>Note A Cisco device can have many RSA key pairs.</td>
</tr>
<tr>
<td><strong>Step 4</strong> crypto key generate rsa usage-keys label key-label modulus modulus-size</td>
<td>Enables the SSH server for local and remote authentication on the device.</td>
</tr>
<tr>
<td>Example: Device(config)# crypto key generate rsa usage-keys label sshkeys modulus 768</td>
<td>• For SSH Version 2, the modulus size must be at least 768 bits.</td>
</tr>
<tr>
<td><strong>Step 5</strong> ip ssh [time-out seconds</td>
<td>authentication-retries integer]</td>
</tr>
<tr>
<td></td>
<td>Note To delete the RSA key pair, use the crypto key zeroize rsa command. When you delete the RSA key pair, you automatically disable the SSH server.</td>
</tr>
</tbody>
</table>
### Configuring the Cisco SSH Server to Perform RSA-Based User Authentication

**SUMMARY STEPS**

1. `enable`
2. `configure terminal`
3. `hostname name`
4. `ip domain-name name`
5. `crypto key generate rsa`
6. `ip ssh pubkey-chain`
7. `username username`
8. `key-string`
9. `key-hash key-type key-name`
10. `end`

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>Example: Device&gt; enable</td>
<td>• Enter your password if prompted.</td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Step 2</strong> configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example: Device# configure terminal</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> hostname name</td>
<td>Specifies the hostname.</td>
</tr>
<tr>
<td>Example: Device(config)# hostname host1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> ip domain-name name</td>
<td>Defines a default domain name that the Cisco software uses to complete unqualified hostnames.</td>
</tr>
<tr>
<td>Example: host1(config)# ip domain-name name1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> crypto key generate rsa</td>
<td>Generates RSA key pairs.</td>
</tr>
<tr>
<td>Example: host1(config)# crypto key generate rsa</td>
<td></td>
</tr>
<tr>
<td><strong>Step 6</strong> ip ssh pubkey-chain</td>
<td>Configures SSH-RSA keys for user and server authentication on the SSH server and enters public-key configuration mode.</td>
</tr>
<tr>
<td>Example: host1(config)# ip ssh pubkey-chain</td>
<td>• The user authentication is successful if the RSA public key stored on the server is verified with the public or the private key pair stored on the client.</td>
</tr>
<tr>
<td><strong>Step 7</strong> username username</td>
<td>Configures the SSH username and enters public-key user configuration mode.</td>
</tr>
<tr>
<td>Example: host1(conf-ssh-pubkey)# username user1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 8</strong> key-string</td>
<td>Specifies the RSA public key of the remote peer and enters public-key data configuration mode.</td>
</tr>
<tr>
<td>Example: host1(conf-ssh-pubkey-user)# key-string</td>
<td>Note You can obtain the public key value from an open SSH client; that is, from the .ssh/id_rsa.pub file.</td>
</tr>
<tr>
<td><strong>Step 9</strong> key-hash key-type key-name</td>
<td>(Optional) Specifies the SSH key type and version.</td>
</tr>
<tr>
<td>Example: host1(conf-ssh-pubkey-data)# key-hash ssh-rsa key1</td>
<td>• The key type must be ssh-rsa for the configuration of private public key pairs.</td>
</tr>
<tr>
<td></td>
<td>• This step is optional only if the key-string command is configured.</td>
</tr>
<tr>
<td></td>
<td>• You must configure either the key-string command or the key-hash command.</td>
</tr>
</tbody>
</table>
### Configuring the Cisco IOS SSH Client to Perform RSA-Based Server Authentication

#### SUMMARY STEPS

1. `enable`
2. `configure terminal`
3. `hostname name`
4. `ip domain-name name`
5. `crypto key generate rsa`
6. `ip ssh pubkey-chain`
7. `server server-name`
8. `key-string`
9. `exit`
10. `key-hash key-type key-name`
11. `end`
12. `configure terminal`
13. `ip ssh stricthostkeycheck`

#### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 1**
| `enable`  | Enables privileged EXEC mode. |
| **Example:**
<p>| <code>Device&gt;</code> <code>enable</code>  | Enter your password if prompted. |</p>
<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 2 configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device# configure terminal</td>
<td></td>
</tr>
<tr>
<td>Step 3 hostname name</td>
<td>Specifies the hostname.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device(config)# hostname host1</td>
<td></td>
</tr>
<tr>
<td>Step 4 ip domain-name name</td>
<td>Defines a default domain name that the Cisco software uses to complete unqualified hostnames.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>host1(config)# ip domain-name name1</td>
<td></td>
</tr>
<tr>
<td>Step 5 crypto key generate rsa</td>
<td>Generates RSA key pairs.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>host1(config)# crypto key generate rsa</td>
<td></td>
</tr>
<tr>
<td>Step 6 ip ssh pubkey-chain</td>
<td>Configures SSH-RSA keys for user and server authentication on the SSH server and enters public-key configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>host1(config)# ip ssh pubkey-chain</td>
<td></td>
</tr>
<tr>
<td>Step 7 server server-name</td>
<td>Enables the SSH server for public-key authentication on the device and enters public-key server configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>host1(conf-ssh-pubkey)# server server1</td>
<td></td>
</tr>
<tr>
<td>Step 8 key-string</td>
<td>Specifies the RSA public-key of the remote peer and enters public key data configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>Note You can obtain the public key value from an open SSH client; that is, from the .ssh/id_rsa.pub file.</td>
</tr>
<tr>
<td>host1(conf-ssh-pubkey-server)# key-string</td>
<td></td>
</tr>
<tr>
<td>Step 9 exit</td>
<td>Exits public-key data configuration mode and enters public-key server configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>host1(conf-ssh-pubkey-data)# exit</td>
<td></td>
</tr>
<tr>
<td>Step 10 key-hash key-type key-name</td>
<td>(Optional) Specifies the SSH key type and version.</td>
</tr>
</tbody>
</table>
### Starting an Encrypted Session with a Remote Device

The device with which you want to connect must support a Secure Shell (SSH) server that has an encryption algorithm that is supported in Cisco software. Also, you need not enable your device. SSH can be run in disabled mode.

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td>• The key type must be ssh-rsa for the configuration of private/public key pairs.</td>
</tr>
<tr>
<td>host1(conf-ssh-pubkey-server)# key-hash</td>
<td>• This step is optional only if the <strong>key-string</strong> command is configured.</td>
</tr>
<tr>
<td>ssh-rsa key1</td>
<td>• You must configure either the <strong>key-string</strong> command or the <strong>key-hash</strong> command.</td>
</tr>
<tr>
<td><strong>Note</strong></td>
<td>You can use a hashing software to compute the hash of the public key string, or you can copy the hash value from another Cisco device. Entering the public key data using the <strong>key-string</strong> command is the preferred way to enter the public key data for the first time.</td>
</tr>
<tr>
<td><strong>Step 11</strong></td>
<td><strong>end</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>Exits public-key server configuration mode and returns to privileged EXEC mode.</td>
</tr>
<tr>
<td>host1(conf-ssh-pubkey-server)# end</td>
<td></td>
</tr>
<tr>
<td><strong>Step 12</strong></td>
<td><strong>configure terminal</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>host1# configure terminal</td>
<td></td>
</tr>
<tr>
<td><strong>Step 13</strong></td>
<td><strong>ip ssh stricthostkeycheck</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>Ensures that server authentication takes place.</td>
</tr>
<tr>
<td>host1(config)# ip ssh stricthostkeycheck</td>
<td>• The connection is terminated in case of a failure.</td>
</tr>
<tr>
<td></td>
<td>• Use <strong>no hostname</strong> command to return to the default host.</td>
</tr>
</tbody>
</table>

**Starting an Encrypted Session with a Remote Device**

The device with which you want to connect must support a Secure Shell (SSH) server that has an encryption algorithm that is supported in Cisco software. Also, you need not enable your device. SSH can be run in disabled mode.
### SUMMARY STEPS

1. `ssh [-v {1 | 2} | -c {aes128-ctr | aes192-ctr | aes256-ctr | aes128-cbc | aes192-cbc | aes256-cbc | 3des | aes192-cbc | aes256-cbc | aes128-ctr | aes192-ctr | aes256-ctr | aes128-cbc | aes192-cbc | aes256-cbc | hmac-md5-96 | hmac-sha1-96}] [-l user-id | -l user-id:vrf-name number ip-address ip-address | -l user-id:rotary number ip-address | -m {hmac-md5-128 | hmac-md5-96 | hmac-sha1-160 | hmac-sha1-96}] [-o numberofpasswordprompts n | -p port-num] ip-addr | hostname [command | -vrf]

### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> `ssh [-v {1</td>
<td>2}</td>
</tr>
</tbody>
</table>

**Example:**

```
Device# ssh -v 2 -c aes256-ctr -m hmac-sha1-96 -l user2 10.76.82.24
```

### Troubleshooting Tips

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

### Enabling Secure Copy Protocol on the SSH Server

**Note**

The following task configures the server-side functionality for SCP. This task shows a typical configuration that allows the device to securely copy files from a remote workstation.
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 scpserverenable
10. exit
11. debug ip scp

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>Example: Device&gt; enable</td>
<td></td>
</tr>
<tr>
<td>• Enter your password if prompted.</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example: Device# configure terminal</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> aaa new-model</td>
<td>Enables the AAA access control model.</td>
</tr>
<tr>
<td>Example: Device(config)# aaa new-model</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> aaa authentication login default local</td>
<td>Sets AAA authentication at login to use the local username database for authentication.</td>
</tr>
<tr>
<td>Example: Device(config)# aaa authentication login default local</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> aaa authorization exec default local</td>
<td>Sets the parameters that restrict user access to a network, runs the authorization to determine if the user ID is allowed to run an EXEC shell, and specifies that the system must use the local database for authorization.</td>
</tr>
<tr>
<td>Example: Device(config)# aaa authorization exec default local</td>
<td></td>
</tr>
<tr>
<td><strong>Step</strong></td>
<td><strong>Command or Action</strong></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td><strong>Step 6</strong></td>
<td><code>username name privilege privilege-level password password</code></td>
</tr>
<tr>
<td><strong>Step 7</strong></td>
<td><code>ip ssh time-out seconds</code></td>
</tr>
<tr>
<td><strong>Step 8</strong></td>
<td><code>ip ssh authentication-retries integer</code></td>
</tr>
<tr>
<td><strong>Step 9</strong></td>
<td><code>ip scpserverenable</code></td>
</tr>
<tr>
<td><strong>Step 10</strong></td>
<td><code>exit</code></td>
</tr>
<tr>
<td><strong>Step 11</strong></td>
<td><code>debug ip scp</code></td>
</tr>
</tbody>
</table>

### Verifying the Status of the Secure Shell Connection

**SUMMARY STEPS**

1. `enable`
2. `show ssh`
3. `exit`
### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>Device&gt; enable</code></td>
</tr>
<tr>
<td></td>
<td>• Enter your password if prompted.</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>Displays the status of SSH server connections.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>Device# show ssh</code></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>Exits privileged EXEC mode and returns to user EXEC mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>Device# exit</code></td>
</tr>
</tbody>
</table>

### Examples

The following sample output from the `show ssh` command displays status of various SSH Version 1 and Version 2 connections for Version 1 and Version 2 connections:

```
Device# show ssh
Connection  Version Encryption State  Username
  0        1.5  3DES  Session started  lab
%No SSHv1 server connections running.
```

The following sample output from the `show ssh` command displays status of various SSH Version 1 and Version 2 connections for a Version 2 connection with no Version 1 connection:

```
Device# show ssh
Connection  Version 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.
```

The following sample output from the `show ssh` command displays status of various SSH Version 1 and Version 2 connections for a Version 1 connection with no Version 2 connection:

```
Device# show ssh
Connection  Version Encryption  Hmac  State  Username
  0       1.5  3DES  Session started  lab
%No SSHv1 server connections running.
```
Verifying the Secure Shell Status

SUMMARY STEPS

1. enable
2. show ip ssh
3. exit

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td><strong>enable</strong>&lt;br&gt;&lt;br&gt;Example: Device&gt; enable&lt;br&gt;&lt;br&gt;Enables privileged EXEC mode.&lt;br&gt;• Enter your password if prompted.</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><strong>show ip ssh</strong>&lt;br&gt;&lt;br&gt;Example: Device# show ip ssh&lt;br&gt;&lt;br&gt;Displays the version and configuration data for SSH.</td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td><strong>exit</strong>&lt;br&gt;&lt;br&gt;Example: Device# exit&lt;br&gt;&lt;br&gt;Exits privileged EXEC mode and returns to user EXEC mode.</td>
</tr>
</tbody>
</table>

Examples

The following sample output from the `show ip ssh` command displays the version of SSH that is enabled, the authentication timeout values, and the number of authentication retries for Version 1 and Version 2 connections:

```
Device# show ip ssh
SSH Enabled - version 1.99
Authentication timeout: 120 secs; Authentication retries: 3
```

The following sample output from the `show ip ssh` command displays the version of SSH that is enabled, the authentication timeout values, and the number of authentication retries for a Version 2 connection with no Version 1 connection:

```
```
Device# show ip ssh
SSH Enabled - version 2.0
Authentication timeout: 120 secs; Authentication retries: 3
------------------------------------------------------------------------
The following sample output from the `show ip ssh` command displays the version of SSH that is enabled, the authentication timeout values, and the number of authentication retries for a Version 1 connection with no Version 2 connection:

Device# 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

SUMMARY STEPS

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

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td></td>
</tr>
<tr>
<td>enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device&gt; enable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Enter your password if prompted.</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td></td>
</tr>
<tr>
<td>debug ip ssh</td>
<td>Enables debugging of SSH.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device# debug ip ssh</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td></td>
</tr>
<tr>
<td>debug snmp packet</td>
<td>Enables debugging of every SNMP packet sent or received by the device.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device# debug snmp packet</td>
<td></td>
</tr>
</tbody>
</table>
Example

The following sample output from the `debug ip ssh` command shows the connection is an SSH Version 2 connection:

```
Device# 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: sshReceive: 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: sshReceive: 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: received packet type 21
00:33:55: SSH2 1: SSH2_MSG_NEWKEYS received
00:33:55: SSH2 1: newkeys: mode 0
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
00:33:55: SSH2 1: sshReceive: 48 bytes received
00:33:55: SSH2 1: input: packet len 32
00:33:55: SSH2 1: partial packet 16, need 16, maclen 16
00:33:55: SSH2 1: MAC #3 ok
00:33:55: SSH2 1: input: padlen 10
00:33:55: SSH2 1: received packet type 5
00:33:55: SSH2 1: send: len 32 (includes padlen 10)
00:33:55: SSH2 1: done calc MAC out #3
00:33:55: SSH2 1: sshReceive: 64 bytes received
00:33:55: SSH2 1: input: packet len 48
00:33:55: SSH2 1: partial packet 16, need 32, maclen 16
00:33:55: SSH2 1: MAC #4 ok
00:33:55: SSH2 1: input: padlen 9
00:33:55: SSH2 1: received packet type 50
00:33:55: SSH2 1: send: len 32 (includes padlen 13)
00:33:55: SSH2 1: done calc MAC out #4
00:34:04: SSH2 1: sshReceive: 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: sshReceive: 16 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 #6 ok
00:34:04: SSH2 1: input: padlen 6
```
Secure Shell Configuration Guide, Cisco IOS Release 15E

Monitoring and Maintaining Secure Shell Version 2

Secure Shell Version 2 Support

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: 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-reg 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
00:34:04: 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 #11 ok
00:34:04: SSH2 1: input: padlen 17
00:34:04: SSH2 1: received packet type 94
00:34:04: SSH2 1: send: len 32 (includes padlen 17)
00:34:04: SSH2 1: done calc MAC out #8
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 #12 ok
00:34:04: SSH2 1: input: padlen 17
00:34:04: SSH2 1: received packet type 94
00:34:04: SSH2 1: send: len 32 (includes padlen 17)
00:34:04: SSH2 1: done calc MAC out #9
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 #13 ok
00:34:04: SSH2 1: input: padlen 17
00:34:04: SSH2 1: received packet type 94
00:34:04: SSH2 1: send: len 32 (includes padlen 17)
00:34:04: SSH2 1: done calc MAC out #10
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 #14 ok
00:34:04: SSH2 1: input: padlen 17
00:34:04: SSH2 1: received packet type 94
00:34:04: SSH2 1: send: len 32 (includes padlen 17)
00:34:04: SSH2 1: done calc MAC out #11
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 #15 ok
00:34:04: SSH2 1: input: padlen 17
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: pty-req request
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 #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
00:34:04: 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 #11 ok
00:34:04: SSH2 1: input: padlen 17
00:34:04: SSH2 1: received packet type 94
00:34:04: SSH2 1: send: len 32 (includes padlen 17)
00:34:04: SSH2 1: done calc MAC out #8
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 #12 ok
00:34:04: SSH2 1: input: padlen 17
00:34:04: SSH2 1: received packet type 94
00:34:04: SSH2 1: send: len 32 (includes padlen 17)
00:34:04: SSH2 1: done calc MAC out #9
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 #13 ok
00:34:04: SSH2 1: input: padlen 17
00:34:04: SSH2 1: received packet type 94
00:34:04: SSH2 1: send: len 32 (includes padlen 17)
00:34:04: SSH2 1: done calc MAC out #10
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 #14 ok
00:34:04: SSH2 1: input: padlen 17
00:34:04: SSH2 1: received packet type 94
00:34:04: SSH2 1: send: len 32 (includes padlen 17)
00:34:04: SSH2 1: done calc MAC out #11
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 #15 ok
00:34:04: SSH2 1: input: padlen 17
Configuration Examples for Secure Shell Version 2 Support

Example: Configuring Secure Shell Version 1

Device# configure terminal
Device(config)# ip ssh version 1

Example: Configuring Secure Shell Version 2

Device# configure terminal
Device(config)# ip ssh version 2

Example: Configuring Secure Shell Versions 1 and 2

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

Example: Starting an Encrypted Session with a Remote Device

Device# ssh -v 2 -c aes256-cbc -m hmac-sha1-160 -l sship 10.76.82.24

Example: Configuring Server-Side SCP

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

Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication login default local
Device(config)# aaa authorization exec default local
Device(config)# username samplename privilege 15 password password1
Device(config)# ip ssh time-out 120
Device(config)# ip ssh authentication-retries 3
Device(config)# ip scp server enable
Example: Setting an SNMP Trap

The following example shows that an SNMP trap is set. The trap notification is generated automatically when the SSH session terminates. In the example, a.b.c.d is the IP address of the SSH client. For an example of SNMP trap debug output, see the "Example: SNMP Debugging, on page 48" section.

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

Examples: SSH Keyboard Interactive Authentication

Example: Enabling Client-Side Debugs

The following example shows that the client-side debugs are turned on, and the maximum number of prompts is six (three for the SSH keyboard interactive authentication method and three for the password authentication method).

```
Password:
Password:
Password:
Password:
Password:
Password: cisco123
Last login: Tue Dec 6 13:15:21 2005 from 10.76.248.213
user1@courier:~>
exit
logout
[Connection to 10.76.248.200 closed by foreign host]
Device1# debug ip ssh client
SSH Client debugging is on
Device1# ssh -l lab 10.1.1.3
Password:
Password:
Password:
Password: lab
Device2>
```

```
*Nov 17 12:50:53.199: SSH0: sent protocol version id SSH-1.99-Cisco-1.25
*Nov 17 12:50:53.199: SSH CLIENT0: protocol version id is - SSH-1.99-Cisco-1.25
*Nov 17 12:50:53.199: SSH CLIENT0: sent protocol version id SSH-1.99-Cisco-1.25
*Nov 17 12:50:53.199: SSH CLIENT0: protocol version exchange successful
*Nov 17 12:50:53.203: SSH0: protocol version id is - SSH-1.99-Cisco-1.25
*Nov 17 12:50:53.335: SSH CLIENT0: key exchange successful and encryption on
*Nov 17 12:50:53.335: SSH2 CLIENT 0: using method keyboard-interactive
Password:
Password:
Password:
*Nov 17 12:51:01.887: SSH2 CLIENT 0: using method password authentication
Password:
Password: lab
Device2>
```

```
*Nov 17 12:51:11.407: SSH2 CLIENT 0: SSH2_MSG_USERAUTH_SUCCESS message received
*Nov 17 12:51:11.407: SSH CLIENT0: user authenticated
*Nov 17 12:51:11.407: SSH2 CLIENT 0: pty-req request sent
*Nov 17 12:51:11.411: SSH2 CLIENT 0: shell request sent
*Nov 17 12:51:11.411: SSH CLIENT0: session open
```
**Example: Enabling ChPass with a Blank Password Change**

In the following example, the ChPass feature is enabled, and a blank password change is accomplished using the SSH Keyboard Interactive Authentication method. A TACACS+ access control server (ACS) is used as the back-end AAA server.

```
Device1# ssh -l cisco 10.1.1.3
Password:
Old Password: cisco
New Password: cisco123
Re-enter New password: cisco123
Device2> exit
[Connection to 10.1.1.3 closed by foreign host]
```

**Example: Enabling ChPass and Changing the Password on First Login**

In the following example, the ChPass feature is enabled and TACACS+ ACS is used as the back-end server. The password is changed on the first login using the SSH keyboard interactive authentication method.

```
Device1# ssh -l cisco 10.1.1.3
Password: cisco
Your password has expired. Enter a new one now.
New Password: cisco123
Re-enter New password: cisco123
Device2> exit
[Connection to 10.1.1.3 closed by foreign host]
Device1# ssh -l cisco 10.1.1.3
Password: cisco1
Your password has expired. Enter a new one now.
New Password: cisco
Re-enter New password: cisco12
The New and Re-entered passwords have to be the same. Try again.
New Password: cisco
Re-enter New password: cisco
Device2>
```

**Example: Enabling ChPass and Expiring the Password After Three Logins**

In the following example, the ChPass feature is enabled and TACACS+ ACS is used as the back-end AAA server. The password expires after three logins using the SSH keyboard interactive authentication method.

```
Device# ssh -l cisco. 10.1.1.3
Password: cisco
Device2> exit
[Connection to 10.1.1.3 closed by foreign host]
```
Example: SNMP Debugging

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

```
Device1# debug snmp packet
SNMP packet debugging is on
Device1# ssh -l lab 10.0.0.2
Password:
Device2> exit
[Connection to 10.0.0.2 closed by foreign host]
Device1# ssh -l lab 10.0.0.2
Password: cisco
Your password has expired.
Enter a new one now.
New Password: cisco123
Re-enter New password: cisco123
Device2>
```

Examples: SSH Debugging Enhancements

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

```
Device1# 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 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: 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: SSH2: Session terminated normally

The following is sample output from the `debug ip ssh packet` command. The output provides debugging information about the SSH packet.

```
Device# 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: 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
```

### Additional References for Secure Shell Version 2 Support

**Related Documents**

<table>
<thead>
<tr>
<th>Related Topic</th>
<th>Document Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco IOS commands</td>
<td>Cisco IOS Master Command List, All Releases</td>
</tr>
</tbody>
</table>
### Related Topic

<table>
<thead>
<tr>
<th>AAA</th>
<th>Security Configuration Guide: Securing User Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hostname and host domain configuration tasks</td>
<td></td>
</tr>
<tr>
<td>Secure shell configuration tasks</td>
<td></td>
</tr>
<tr>
<td>Downloading a software image</td>
<td>Configuration Fundamentals Configuration Guide</td>
</tr>
<tr>
<td>Configuration fundamentals</td>
<td></td>
</tr>
<tr>
<td>IPsec configuration tasks</td>
<td>Security Configuration Guide: Secure Connectivity</td>
</tr>
<tr>
<td>SNMP traps configuration tasks</td>
<td>SNMP Configuration Guide</td>
</tr>
</tbody>
</table>

### Standards

<table>
<thead>
<tr>
<th>Standards</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>IETF Secure Shell Version 2 Draft Standards</td>
<td>Internet Engineering Task Force website</td>
</tr>
</tbody>
</table>

### Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
<td><a href="http://www.cisco.com/cisco/web/support/index.html">http://www.cisco.com/cisco/web/support/index.html</a></td>
</tr>
</tbody>
</table>

## 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](http://www.cisco.com/go/cfn). An account on Cisco.com is not required.
### Table 3: Feature Information for Secure Shell Version 2 Support

<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Releases</th>
<th>Feature Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure Shell Version 2 Support</td>
<td>Cisco IOS 15.2(1)E</td>
<td>The Secure Shell Version 2 Support feature allows you to configure Secure Shell (SSH) Version 2 (SSH Version 1 support was implemented in an earlier Cisco IOS software release). SSH runs on top of a reliable transport layer and provides strong authentication and encryption capabilities. SSH version 2 also supports AES counter-based encryption mode. The following commands were introduced or modified: <code>debug ip ssh</code>, <code>ip ssh min dh size</code>, <code>ip ssh rsa keypair-name</code>, <code>ip ssh version</code>, <code>ssh</code>.</td>
</tr>
<tr>
<td>Secure Shell Version 2 Client and Server Support</td>
<td>Cisco IOS 15.2(1)E</td>
<td>The Cisco IOS image was updated to provide for the automatic generation of SNMP traps when an SSH session terminates.</td>
</tr>
<tr>
<td>SSH Keyboard Interactive Authentication</td>
<td>Cisco IOS 15.2(1)E</td>
<td>The SSH Keyboard Interactive Authentication feature, also known as Generic Message Authentication for SSH, is a method that can be used to implement different types of authentication mechanisms. Basically, any currently supported authentication method that requires only user input can be performed with this feature.</td>
</tr>
<tr>
<td>Secure Shell Version 2 Enhancements</td>
<td>Cisco IOS 15.2(1)E</td>
<td>The Secure Shell Version 2 Enhancements feature includes a number of additional capabilities such as support for VRF-aware SSH, SSH debug enhancements, and DH Group 14 and Group 16 exchange support. The following commands were introduced or modified: <code>debug ip ssh</code>, <code>ip ssh dh min size</code>.</td>
</tr>
<tr>
<td>Feature Name</td>
<td>Releases</td>
<td>Feature Information</td>
</tr>
<tr>
<td>--------------</td>
<td>----------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Secure Shell Version 2 Enhancements for RSA Keys.</td>
<td>Cisco IOS 15.2(1)E</td>
<td>The Secure Shell Version 2 Enhancements for RSA Keys feature includes a number of additional capabilities to support RSA key-based user authentication for SSH and SSH server host key storage and verification. The following commands were introduced or modified: ip ssh pubkey-chain, ip ssh stricthostkeycheck.</td>
</tr>
</tbody>
</table>