THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS.
CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: http://www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

© 2015 Cisco Systems, Inc. All rights reserved.
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
### Contents

- **SSH Keyboard Interactive Authentication** 33
- **How to Configure Secure Shell Version 2 Support** 33
  - Configuring a Device for SSH Version 2 Using a Hostname and Domain Name 33
  - Configuring a Device for SSH Version 2 Using RSA Key Pairs 35
  - Configuring the Cisco SSH Server to Perform RSA-Based User Authentication 36
  - Configuring the Cisco IOS SSH Client to Perform RSA-Based Server Authentication 38
- **Starting an Encrypted Session with a Remote Device** 40
  - Troubleshooting Tips 41
- **Enabling Secure Copy Protocol on the SSH Server** 41
- **Verifying the Status of the Secure Shell Connection** 43
- **Verifying the Secure Shell Status** 45
- **Monitoring and Maintaining Secure Shell Version 2** 46
- **Configuration Examples for Secure Shell Version 2 Support** 49
  - Example: Configuring Secure Shell Version 1 49
  - Example: Configuring Secure Shell Version 2 49
  - Example: Configuring Secure Shell Versions 1 and 2 49
  - Example: Starting an Encrypted Session with a Remote Device 49
  - Example: Configuring Server-Side SCP 49
  - Example: Setting an SNMP Trap 50
- **Examples: SSH Keyboard Interactive Authentication** 50
  - Example: Enabling Client-Side Debugs 50
  - Example: Enabling ChPass with a Blank Password Change 51
  - Example: Enabling ChPass and Changing the Password on First Login 51
  - Example: Enabling ChPass and Expiring the Password After Three Logins 51
  - Example: SNMP Debugging 52
  - Examples: SSH Debugging Enhancements 52
- **Additional References for Secure Shell Version 2 Support** 53
- **Feature Information for Secure Shell Version 2 Support** 54

---

**CHAPTER 5**

- **SSH Terminal-Line Access** 57
  - Finding Feature Information 57
  - Prerequisites for SSH Terminal-Line Access 57
  - Restrictions for SSH Terminal-Line Access 58
  - Information About SSH Terminal-Line Access 58
  - Overview of SSH Terminal-Line Access 58
How to Configure SSH Terminal-Line Access 59
Configuring SSH Terminal-Line Access 59
Verifying SSH Terminal-Line Access 61
Configuration Examples for SSH Terminal-Line Access 61
Example SSH Terminal-Line Access Configuration 61
Example SSH Terminal-Line Access for a Console Serial Line Ports Configuration 61
Additional References 62
Feature Information for SSH Terminal-Line Access 63

CHAPTER 6
X.509v3 Certificates for SSH Authentication 65
Finding Feature Information 65
Prerequisites for X.509v3 Certificates for SSH Authentication 66
Restrictions for X.509v3 Certificates for SSH Authentication 66
Information About X.509v3 Certificates for SSH Authentication 66
Digital certificates 66
Server and user authentication using X.509v3 66
How to Configure X.509v3 Certificates for SSH Authentication 67
Configuring IOS SSH Server to Use Digital Certificates for Sever Authentication 67
Configuring IOS SSH Server to Verify User's Digital Certificate for User Authentication 68
Verifying Configuration for Server and User Authentication Using Digital Certificates 70
Configuration Examples for X.509v3 Certificates for SSH Authentication 71
Example: Configuring IOS SSH Server to Use Digital Certificates for Sever Authentication 71
Example: Configuring IOS SSH Server to Verify User's Digital Certificate for User Authentication 71
Additional References for X.509v3 Certificates for SSH Authentication 71
Feature Information for X.509v3 Certificates for SSH Authentication 72

CHAPTER 7
SSH Algorithms for Common Criteria Certification 75
Finding Feature Information 75
Information About SSH Algorithms for Common Criteria Certification 76
SSH Algorithms for Common Criteria Certification 76
Cisco IOS SSH Server Algorithms 76
Cisco IOS SSH Client Algorithms  76
How to Configure SSH Algorithms for Common Criteria Certification  77
   Configuring an Encryption Key Algorithm for a Cisco IOS SSH Server and Client  77
      Troubleshooting Tips  78
   Configuring a MAC Algorithm for a Cisco IOS SSH Server and Client  78
      Troubleshooting Tips  79
   Configuring a Host Key Algorithm for a Cisco IOS SSH Server  80
      Troubleshooting Tips  81
Verifying SSH Algorithms for Common Criteria Certification  81
Configuration Examples For SSH Algorithms for Common Criteria Certification  82
   Example: Configuring Encryption Key Algorithms for a Cisco IOS SSH Server  82
   Example: Configuring Encryption Key Algorithms for a Cisco IOS SSH Client  82
   Example: Configuring MAC Algorithms for a Cisco IOS SSH Server  82
   Example: Configuring MAC Algorithms for a Cisco IOS SSH Client  82
   Example: Configuring Host Key Algorithms for a Cisco IOS SSH Server  83
Additional References for SSH Algorithms for Common Criteria Certification  83
Feature Information for SSH Algorithms for Common Criteria Certification  84
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></td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>enable</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong> Device&gt; enable</td>
<td>• Enter your password if prompted.</td>
</tr>
</tbody>
</table>

| **Step 2** | Enters global configuration mode. |
| configure terminal | |
| **Example:** Device# configure terminal | |

| **Step 3** | Configures Secure Shell (SSH) control parameters. |
| ip ssh {timeout seconds | authentication-retries integer} | |
| **Example:** Device(config)# ip ssh timeout 30 | |

**Note**
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:

• Value proposed by the client using the ssh -o numberofpasswordprompt command.

• Value configured on the device using the ip ssh authentication-retries integer command, plus one.
### 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>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> 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>Example: 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 connections 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
```
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
no 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
```
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><em>Cisco IOS Master Command List, All Releases</em></td>
</tr>
<tr>
<td>Authentication, authorization, and accounting (AAA)</td>
<td><em>Authentication, Authorization, and Accounting Configuration Guide</em></td>
</tr>
<tr>
<td>IPsec</td>
<td>“IPsec and Quality of Service” 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><em>Loading and Managing System Images Configuration Guide</em></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</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 12.0(5)S</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</td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 15.1(1)SY</td>
<td>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>
</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 21

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 \(-l\) keyword and user_id \{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><strong>Example:</strong></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><strong>Example:</strong></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><strong>Example:</strong></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><strong>Example:</strong></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><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>Device(config-line)# login authentication default</td>
<td></td>
</tr>
<tr>
<td><strong>Note</strong></td>
<td>The authentication method must use a username and password.</td>
</tr>
</tbody>
</table>
### Command or Action | Purpose
--- | ---
**Step 6** | transport input ssh
**Example:**
Device(config-line)# transport input ssh
Defines which protocols to use to connect to a specific line of the device.
- The `ssh` keyword must be used for the Reverse SSH Enhancements feature.

**Step 7** | exit
**Example:**
Device(config-line)# exit
Exits line configuration mode.

**Step 8** | exit
**Example:**
Device(config)# exit
Exits global configuration mode.

**Step 9** | ssh -l userid : {number} {ip-address}
**Example:**
Device# ssh -l lab:1 router.example.com
Specifies the user ID to use when logging in on the remote networking device that is running the SSH server.
- `userid` --User ID.
- `:` --Signifies that a port number and terminal IP address will follow the userid argument.
- `{number}` --Terminal or auxiliary line number.
- `{ip-address}` --Terminal server IP address.

**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>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td><strong>enable</strong></td>
</tr>
<tr>
<td>Example:</td>
<td><strong>Device&gt; enable</strong></td>
</tr>
<tr>
<td></td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td></td>
<td>• Enter your password if prompted.</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><strong>configure terminal</strong></td>
</tr>
<tr>
<td>Example:</td>
<td><strong>Device# configure terminal</strong></td>
</tr>
<tr>
<td></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td><strong>line line-number ending-line-number</strong></td>
</tr>
<tr>
<td>Example:</td>
<td><strong>Device# line 1 200</strong></td>
</tr>
<tr>
<td></td>
<td>Identifies a line for configuration and enters line configuration mode.</td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td><strong>no exec</strong></td>
</tr>
<tr>
<td>Example:</td>
<td><strong>Device(config-line)# no exec</strong></td>
</tr>
<tr>
<td></td>
<td>Disables EXEC processing on a line.</td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td><strong>login authentication listname</strong></td>
</tr>
<tr>
<td>Example:</td>
<td><strong>Device(config-line)# login authentication default</strong></td>
</tr>
<tr>
<td></td>
<td>Defines a login authentication mechanism for the lines.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> The authentication method must use a username and password.</td>
</tr>
<tr>
<td>Command or Action</td>
<td>Purpose</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
</tr>
<tr>
<td><strong>Step 6</strong></td>
<td></td>
</tr>
<tr>
<td><code>rotary group</code></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>Example:</strong></td>
<td></td>
</tr>
<tr>
<td><code>Device(config-line)# rotary 1</code></td>
<td></td>
</tr>
</tbody>
</table>

| **Step 7**        |         |
| `transport input ssh` | Defines which protocols to use to connect to a specific line of the device. |
| **Example:**      |         |
| `Device(config-line)# transport input ssh` | It must be used for the Reverse SSH Enhancements feature. |

| **Step 8**        |         |
| `exit`            | Exits line configuration mode. |
| **Example:**      |         |
| `Device(config-line)# exit` |         |

| **Step 9**        |         |
| `exit`            | Exits global configuration mode. |
| **Example:**      |         |
| `Device(config)# exit` |         |

| **Step 10**       | Specifies the user ID to use when logging in on the remote networking device that is running the SSH server. |
| `ssh -l userid :rotary {number} {ip-address}` |         |
| **Example:**      |         |
| `Device# ssh -l lab:rotary1 router.example.com` |         |
| `userid` --User ID. | |
| `: --Signifies that a port number and terminal IP address will follow the `userid` argument. | |
| `number` --Terminal or auxiliary line number. | |
| `ip-address` --Terminal server IP address. | |

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

---

**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 1</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Enter your password if prompted.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device&gt; enable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 2</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>debug ip ssh client</td>
<td>Displays debugging messages for the SSH client.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device# debug ip ssh client</td>
</tr>
</tbody>
</table>

---

**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 1</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Enter your password if prompted.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device&gt; enable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 2</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>debug ip ssh</td>
<td>Displays debugging messages for the SSH server.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device# debug ip ssh</td>
</tr>
<tr>
<td>Step 3</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 4</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>show line</td>
<td>Displays parameters of a terminal line.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Device# show line</td>
<td></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

```plaintext
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:

```plaintext
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:

```plaintext
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>
</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>
</tr>
<tr>
<td>Link</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](http://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>
<tbody>
<tr>
<td>Reverse SSH Enhancements</td>
<td>Cisco IOS 12.2(33)SRD Cisco IOS 12.2(33)SXI Cisco IOS 12.3(11)T</td>
<td>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: <code>ssh</code>.</td>
</tr>
</tbody>
</table>
CHAPTER 3

Secure Copy

The Secure Copy (SCP) feature provides a secure and authenticated method for copying device configurations or device image files. SCP relies on Secure Shell (SSH), an application and protocol that provide a secure replacement for the Berkeley r-tools suite (Berkeley university's own set of networking applications). This document provides the procedure to configure a Cisco device for SCP server-side functionality.

- Finding Feature Information, page 23
- Prerequisites for Secure Copy, page 23
- Information About Secure Copy, page 24
- How to Configure Secure Copy, page 24
- Configuration Examples for Secure Copy, page 26
- Additional References, page 27
- Feature Information for Secure Copy, page 27
- Glossary, page 28

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 Copy

- Before enabling Secure Copy (SCP), you must correctly configure Secure Shell (SSH), authentication, and authorization on the device.
Because SCP relies on SSH for its secure transport, the device must have a Rivest, Shamir, and Adelman (RSA) key pair.

Information About Secure Copy

How Secure Copy Works

The behavior of Secure Copy (SCP) is similar to that of remote copy (RCP), which comes from the Berkeley r-tools suite (Berkeley university’s own set of networking applications), except that SCP relies on Secure Shell (SSH) for security. In addition, SCP requires that authentication, authorization, and accounting (AAA) authorization be configured so that the device can determine whether the user has the correct privilege level.

SCP allows a user with appropriate authorization to copy any file that exists in the Cisco IOS File System (IFS) to and from a device by using the copy command. An authorized administrator may also perform this action from a workstation.

Note

Enable the SCP option while using the pscp.exe file with the Cisco software.

How to Configure Secure Copy

Configuring Secure Copy

To configure a Cisco device for Secure Copy (SCP) server-side functionality, perform the following steps.

SUMMARY STEPS

1. enable
2. configure terminal
3. aaa new-model
4. aaa authentication login {default | list-name} method1 [ method2... ]
5. aaa authorization {network | exec | commands level | reverse-access | configuration} {default | list-name} [method1 [ method2... ]]
6. username name [privilege level] password encryption-type encrypted-password
7. ip scp server enable
8. exit
9. show running-config
10. 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>Sets AAA authentication at login.</td>
</tr>
<tr>
<td>Example: Device(config)# aaa new-model</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> aaa authentication login {default</td>
<td>list-name} method1 [ method2... ]</td>
</tr>
<tr>
<td>Example: Device(config)# aaa authentication login default group tacacs+</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> aaa authorization {network</td>
<td>exec</td>
</tr>
<tr>
<td>Example: Device(config)# aaa authorization exec default group tacacs+</td>
<td></td>
</tr>
<tr>
<td>Note: The <strong>exec</strong> keyword runs authorization to determine if the user is allowed to run an EXEC shell; therefore, you must use the <strong>exec</strong> keyword when you configure SCP.</td>
<td></td>
</tr>
<tr>
<td><strong>Step 6</strong> username name [privilege level] password encryption-type encrypted-password</td>
<td>Establishes a username-based authentication system.</td>
</tr>
<tr>
<td>Example: Device(config)# username superuser privilege 2 password 0 superpassword</td>
<td></td>
</tr>
<tr>
<td>Note: You may omit this step if a network-based authentication mechanism, such as TACACS+ or RADIUS, has been configured.</td>
<td></td>
</tr>
<tr>
<td><strong>Step 7</strong> ip scp server enable</td>
<td>Enables SCP server-side functionality.</td>
</tr>
<tr>
<td>Example: Device(config)# ip scp server 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 8</strong></td>
<td><strong>exit</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>Device(config)# exit</td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>Exits global configuration mode and returns to privileged EXEC mode.</td>
</tr>
<tr>
<td><strong>Step 9</strong></td>
<td><strong>show running-config</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>Device# show running-config</td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>(Optional) Displays the SCP server-side functionality.</td>
</tr>
<tr>
<td><strong>Step 10</strong></td>
<td><strong>debug ip scp</strong></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>Device# debug ip scp</td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>(Optional) Troubleshoots SCP authentication problems.</td>
</tr>
</tbody>
</table>

**Configuration Examples for Secure Copy**

**Example: Secure Copy Configuration Using Local Authentication**

The following example shows how to configure the server-side functionality of Secure Copy (SCP). This example uses a locally defined username and password.

```
! AAA authentication and authorization must be configured properly in order for SCP to work.
aaa new-model
aaa authentication login default local
aaa authorization exec default local
username user1 privilege 15 password 0 lab
! SSH must be configured and functioning properly.
ip scp server enable
```

**Example: Secure Copy Configuration Using Network-Based Authentication**

The following example shows how to configure the server-side functionality of Secure Copy (SCP) using a network-based authentication mechanism:

```
! AAA authentication and authorization must be configured properly in order for SCP to work.
aaa new-model
aaa authentication login default group tacacs+
aaa authorization exec default group tacacs+
! SSH must be configured and functioning properly.
ip scp server enable
```
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>Secure Shell Version 1 and 2 support</td>
<td>Secure Shell Configuration Guide</td>
</tr>
<tr>
<td>Authentication and authorization commands</td>
<td>Cisco IOS Security Command Reference: Commands A to C</td>
</tr>
<tr>
<td>Configuring authentication and authorization</td>
<td>Authentication, Authorization, and Accounting 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 Secure Copy**

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 Copy

<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Releases</th>
<th>Feature Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure Copy</td>
<td>Cisco IOS 12.0(21)S</td>
<td>The Secure Copy (SCP) feature provides a secure and authenticated method for copying device configurations or device image files. SCP relies on Secure Shell (SSH), an application and protocol that provide a secure replacement for the Berkeley r-tools suite. The following commands were introduced or modified: <code>debug ip scp</code>, <code>ip scp server enable</code>.</td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 12.2(2)T</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 12.2(25)S</td>
<td></td>
</tr>
</tbody>
</table>

Glossary

AAA—authentication, authorization, and accounting. A framework of security services that provide the method for identifying users (authentication), for remote access control (authorization), and for collecting and sending security server information used for billing, auditing, and reporting (accounting).

RCP—remote copy. Relies on Remote Shell (Berkeley r-tools suite) for security; RCP copies files such as device images and startup configurations to and from devices.

SCP—secure copy. Relies on SSH for security; SCP support allows secure and authenticated copying of anything that exists in the Cisco IOS File System (IFS). SCP is derived from RCP.

SSH—Secure Shell. An application and protocol that provide a secure replacement for the Berkeley r-tools suite. The protocol secures the sessions using standard cryptographic mechanisms, and the application can be used similar to the Berkeley rexec and rsh tools. SSH Version 1 is implemented in the Cisco software.
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 29
- Prerequisites for Secure Shell Version 2 Support, page 30
- Restrictions for Secure Shell Version 2 Support, page 30
- Information About Secure Shell Version 2 Support, page 30
- How to Configure Secure Shell Version 2 Support, page 33
- Configuration Examples for Secure Shell Version 2 Support, page 49
- Additional References for Secure Shell Version 2 Support, page 53
- Feature Information for Secure Shell Version 2 Support, page 54

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 50" 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></td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>enable</td>
<td>Enter your password if prompted.</td>
</tr>
</tbody>
</table>

Example:

Device> enable
<table>
<thead>
<tr>
<th>Step 2</th>
<th><strong>Command or Action</strong></th>
<th><strong>Purpose</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>configure terminal</code></td>
<td>Enters global configuration mode.</td>
</tr>
</tbody>
</table>

**Example:**
```
Device# configure terminal
```

<table>
<thead>
<tr>
<th>Step 3</th>
<th><strong>Command or Action</strong></th>
<th><strong>Purpose</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>hostname name</code></td>
<td>Configures a hostname for your device.</td>
</tr>
</tbody>
</table>

**Example:**
```
Device(config)# hostname cisco7200
```

<table>
<thead>
<tr>
<th>Step 4</th>
<th><strong>Command or Action</strong></th>
<th><strong>Purpose</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>ip domain-name name</code></td>
<td>Configures a domain name for your device.</td>
</tr>
</tbody>
</table>

**Example:**
```
cisco7200(config)# ip domain-name example.com
```

<table>
<thead>
<tr>
<th>Step 5</th>
<th><strong>Command or Action</strong></th>
<th><strong>Purpose</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>crypto key generate rsa</code></td>
<td>Enables the SSH server for local and remote authentication.</td>
</tr>
</tbody>
</table>

**Example:**
```
cisco7200(config)# crypto key generate rsa
```

<table>
<thead>
<tr>
<th>Step 6</th>
<th><strong>Command or Action</strong></th>
<th><strong>Purpose</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>`ip ssh [time-out seconds</td>
<td>authentication-retries integer]`</td>
</tr>
</tbody>
</table>

**Example:**
```
cisco7200(config)# ip ssh time-out 120
```

<table>
<thead>
<tr>
<th>Step 7</th>
<th><strong>Command or Action</strong></th>
<th><strong>Purpose</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>`ip ssh version [1</td>
<td>2]`</td>
</tr>
</tbody>
</table>

**Example:**
```
cisco7200(config)# ip ssh version 1
```

<table>
<thead>
<tr>
<th>Step 8</th>
<th><strong>Command or Action</strong></th>
<th><strong>Purpose</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>exit</code></td>
<td>Exits global configuration mode and enters privileged EXEC mode.</td>
</tr>
</tbody>
</table>

**Example:**
```
cisco7200(config)# exit
```

- Use **no hostname** command to return to the default host.
### 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: <code>Device&gt; enable</code></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: <code>Device# configure terminal</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> <code>ip ssh rsa keypair-name keypair-name</code></td>
<td>Specifies the RSA key pair to be used for SSH.</td>
</tr>
<tr>
<td>Example: <code>Device(config)# ip ssh rsa keypair-name sshkeys</code></td>
<td>A Cisco device can have many RSA key pairs.</td>
</tr>
<tr>
<td><strong>Step 4</strong> <code>crypto key generate rsa usage-keys label key-label modulus modulus-size</code></td>
<td>Enables the SSH server for local and remote authentication on the device.</td>
</tr>
<tr>
<td>Example: <code>Device(config)# crypto key generate rsa usage-keys label sshkeys modulus 768</code></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>
</tbody>
</table>

Secure Shell Configuration Guide, Cisco IOS Release 15S
### 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:</td>
<td></td>
</tr>
<tr>
<td>Device&gt; enable</td>
<td></td>
</tr>
<tr>
<td><strong>Step 6</strong> ip ssh version 2</td>
<td>Specifies the version of SSH to be run on the device.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device(config)# ip ssh version 2</td>
<td></td>
</tr>
<tr>
<td><strong>Step 7</strong> exit</td>
<td>Exits global configuration mode and enters privileged EXEC mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>Device(config)# exit</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></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><code>configure terminal</code></td>
<td>Example: Device# configure terminal</td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>Specifies the hostname.</td>
</tr>
<tr>
<td><code>hostname name</code></td>
<td>Example: Device(config)# hostname host1</td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>Defines a default domain name that the Cisco software uses to complete unqualified hostnames.</td>
</tr>
<tr>
<td><code>ip domain-name name</code></td>
<td>Example: host1(config)# ip domain-name name1</td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td>Generates RSA key pairs.</td>
</tr>
<tr>
<td><code>crypto key generate rsa</code></td>
<td>Example: host1(config)# crypto key generate rsa</td>
</tr>
<tr>
<td><strong>Step 6</strong></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><code>ip ssh pubkey-chain</code></td>
<td>Example: host1(config)# ip ssh pubkey-chain</td>
</tr>
<tr>
<td><strong>Step 7</strong></td>
<td>Specifies the RSA public key of the remote peer and enters public-key data configuration mode.</td>
</tr>
<tr>
<td><code>username username</code></td>
<td>Example: host1(conf-ssh-pubkey)# username user1</td>
</tr>
<tr>
<td><strong>Step 8</strong></td>
<td>Specifies the RSA public key of the remote peer and enters public-key data configuration mode.</td>
</tr>
<tr>
<td><code>key-string</code></td>
<td>Example: host1(conf-ssh-pubkey-user)# key-string</td>
</tr>
<tr>
<td><strong>Step 9</strong></td>
<td>(Optional) Specifies the SSH key type and version.</td>
</tr>
<tr>
<td><code>key-hash key-type key-name</code></td>
<td>Example: host1(conf-ssh-pubkey-data)# key-hash ssh-rsa key1</td>
</tr>
</tbody>
</table>
You can use a hashing software to compute the hash of the public key string, or you can also copy the hash value from another Cisco device. Entering the public key data using the **key-string** command is the preferred way to enter the public key data for the first time.

**Step 10**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>end</td>
<td>Exits public-key data configuration mode and returns to privileged EXEC mode.</td>
</tr>
</tbody>
</table>

**Example:**

```text
host1(conf-ssh-pubkey-data)# end
```

## 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>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td>enable</td>
<td>• Enter your password if prompted.</td>
</tr>
</tbody>
</table>

**Example:**

```text
Device> enable
```
<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<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 name</td>
<td>Specifies the hostname.</td>
</tr>
<tr>
<td><strong>Example:</strong> 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><strong>Example:</strong> 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><strong>Example:</strong> 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><strong>Example:</strong> host1(config)# ip ssh pubkey-chain</td>
<td></td>
</tr>
<tr>
<td><strong>Step 7</strong> 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><strong>Example:</strong> host1(conf-ssh-pubkey)# server server1</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><strong>Example:</strong> host1(conf-ssh-pubkey-server)# key-string</td>
<td><strong>Note</strong> 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> exit</td>
<td>Exits public-key data configuration mode and enters public-key server configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> host1(conf-ssh-pubkey-data)# exit</td>
<td></td>
</tr>
<tr>
<td><strong>Step 10</strong> 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

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Example:**
host1(conf-ssh-pubkey-server)# key-hash
ssh-rsa key1 | • The key type must be ssh-rsa for the configuration of private/public key pairs.
• This step is optional only if the **key-string** command is configured.
• You must configure either the **key-string** command or the **key-hash** command. |

**Note**: 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 **key-string** command is the preferred way to enter the public key data for the first time.

<table>
<thead>
<tr>
<th>Step 11</th>
<th>end</th>
<th>Exits public-key server configuration mode and returns to privileged EXEC mode.</th>
</tr>
</thead>
</table>
| **Example:**
host1(conf-ssh-pubkey-server)# end | |

<table>
<thead>
<tr>
<th>Step 12</th>
<th>configure terminal</th>
<th>Enters global configuration mode.</th>
</tr>
</thead>
</table>
| **Example:**
host1# configure terminal | |

<table>
<thead>
<tr>
<th>Step 13</th>
<th>ip ssh stricthostkeycheck</th>
<th>Ensures that server authentication takes place.</th>
</tr>
</thead>
</table>
| **Example:**
host1(config)# ip ssh stricthostkeycheck | • The connection is terminated in case of a failure.
• Use **no hostname** command to return to the default host. |

### Starting an Encrypted Session with a Remote Device

**Note**: 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 | aes128-cbc | aes192-cbc | aes256-cbc} | -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 scp server enable
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></td>
<td>enable</td>
</tr>
<tr>
<td>Example:</td>
<td>Device&gt; enable</td>
</tr>
<tr>
<td></td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td></td>
<td>• Enter your password if prompted.</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>configure terminal</td>
</tr>
<tr>
<td>Example:</td>
<td>Device# configure terminal</td>
</tr>
<tr>
<td></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>aaa new-model</td>
</tr>
<tr>
<td>Example:</td>
<td>Device(config)# aaa new-model</td>
</tr>
<tr>
<td></td>
<td>Enables the AAA access control model.</td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>aaa authentication login default local</td>
</tr>
<tr>
<td>Example:</td>
<td>Device(config)# aaa authentication login default local</td>
</tr>
<tr>
<td></td>
<td>Sets AAA authentication at login to use the local username database for authentication.</td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td>aaa authorization exec default local</td>
</tr>
<tr>
<td>Example:</td>
<td>Device(config)# aaa authorization exec default local</td>
</tr>
<tr>
<td></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>
</tbody>
</table>
### Command or Action

<table>
<thead>
<tr>
<th>Step 6</th>
<th><code>username name privilege privilege-level password password</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td><code>Device(config)# username samplename privilege 15 password password1</code></td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>Establishes a username-based authentication system, and specifies the username, privilege level, and an unencrypted password. <strong>Note</strong> The minimum value for the <code>privilege-level</code> argument is 15. A privilege level of less than 15 results in the connection closing.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 7</th>
<th><code>ip ssh time-out seconds</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td><code>Device(config)# ip ssh time-out 120</code></td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>Sets the time interval (in seconds) that the device waits for the SSH client to respond.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 8</th>
<th><code>ip ssh authentication-retries integer</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td><code>Device(config)# ip ssh authentication-retries 3</code></td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>Sets the number of authentication attempts after which the interface is reset.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 9</th>
<th><code>ip scp server enable</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td><code>Device(config)# ip scp server enable</code></td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>Enables the device to securely copy files from a remote workstation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 10</th>
<th><code>exit</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td><code>Device(config)# exit</code></td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>Exits global configuration mode and returns to privileged EXEC mode.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 11</th>
<th><code>debug ip scp</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td><code>Device# debug ip scp</code></td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>(Optional) Provides diagnostic information about SCP authentication problems.</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>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td><code>enable</code></td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Device&gt; enable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Enter your password if prompted.</td>
</tr>
<tr>
<td>Step 2</td>
<td><code>show ssh</code></td>
<td>Displays the status of SSH server connections.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Device# show ssh</td>
<td></td>
</tr>
<tr>
<td>Step 3</td>
<td><code>exit</code></td>
<td>Exits privileged EXEC mode and returns to user EXEC mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Device# exit</td>
<td></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
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
```

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  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.
```

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
1           2.0        IN          aes128-cbc  hmac-md5  Session started  lab
```
### 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>
</table>
| **Step 1**

  - **enable**
  - **Example:**
    
    Device> enable

  - Enables privileged EXEC mode.
  - Enter your password if prompted.

| **Step 2**

  - **show ip ssh**
  - **Example:**
    
    Device# show ip ssh

  - Displays the version and configuration data for SSH.

| **Step 3**

  - **exit**
  - **Example:**
    
    Device# exit

  - Exits privileged EXEC mode and returns to user EXEC mode.

**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

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><code>enable</code></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></td>
<td></td>
</tr>
<tr>
<td><code>debug ip ssh</code></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><code>debug snmp packet</code></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: SSH1: 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 1: kex: client->server aes128-cbc hmac-md5 none
00:33:55: SSH2 1: 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 1: 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: newkeys: mode 0
00:33:55: SSH2 1: received packet type 21
00:33:55: SSH2 1: SSH2_MSG_NEWKEYS 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
00:33:55: 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: ssh receive: 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
00:33:56: 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: 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 15S

Secure Shell Version 2 Support

Monitoring and Maintaining Secure Shell Version 2

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-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
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 authentication 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 52" 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: 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: 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: lab
Device2>
```

```
*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# debug snmp packet
*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
tcpConnEntry.5.10.0.0.1.22.10.0.0.2.55246 = 1015
tcpConnEntry.1.10.0.0.1.22.10.0.0.2.55246 = 1056
tcpConnEntry.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
```

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: SSH0: 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: 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: ssh_receive: 16 bytes received
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

<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 | Document Title
--- | ---
AAA Hostname and host domain configuration tasks Secure shell configuration tasks | Security Configuration Guide: Securing User Services
Downloading a software image Configuration fundamentals | Configuration Fundamentals Configuration Guide
IPsec configuration tasks | Security Configuration Guide: Secure Connectivity
SNMP traps configuration tasks | SNMP Configuration Guide

Standards

<table>
<thead>
<tr>
<th>Standards</th>
<th>Description</th>
</tr>
</thead>
</table>
| IETF Secure Shell Version 2 Draft Standards | Internet Engineering Task Force website

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 4: 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 12.2(11)T</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></td>
<td>Cisco IOS 12.2(25)S</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 12.3(4)T</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 15.3(2)S</td>
<td></td>
</tr>
<tr>
<td>Secure Shell Version 2 Client and Server Support</td>
<td>Cisco IOS 12.0(32)SY</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></td>
<td>Cisco IOS 12.3(7)JA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 12.4(17)</td>
<td></td>
</tr>
<tr>
<td>SSH Keyboard Interactive Authentication</td>
<td>Cisco IOS 12.2(33)SXH3</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></td>
<td>Cisco IOS 12.4(18)</td>
<td></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</td>
<td>Cisco IOS 12.2(50)SY</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. In Cisco IOS 15.1(2)S, support was added for the Cisco 7600 series router. Note Only the VRF-aware SSH feature is supported in Cisco IOS Release 12.2(50)SY. The following commands were introduced or modified: <code>debug ip ssh</code>, <code>ip ssh dh min size</code>.</td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 12.4(20)T</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 15.1(2)S</td>
<td></td>
</tr>
<tr>
<td>Secure Shell Version 2 Enhancements for RSA Keys</td>
<td>Cisco IOS 15.0(1)M</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: <code>ip ssh pubkey-chain</code>, <code>ip ssh stricthostkeycheck</code>.</td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 15.1(1)S</td>
<td></td>
</tr>
</tbody>
</table>
CHAPTER 5

SSH Terminal-Line Access

The SSH Terminal-Line Access feature provides users secure access to tty (text telephone) lines. tty allows the hearing- and speech-impaired to communicate by using a telephone to type messages.

- Finding Feature Information, page 57
- Prerequisites for SSH Terminal-Line Access, page 57
- Restrictions for SSH Terminal-Line Access, page 58
- Information About SSH Terminal-Line Access, page 58
- How to Configure SSH Terminal-Line Access, page 59
- Configuration Examples for SSH Terminal-Line Access, page 61
- Additional References, page 62
- Feature Information for SSH Terminal-Line Access, page 63

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 SSH Terminal-Line Access

Download the required image to your router. The secure shell (SSH) server requires the router to have an IPSec (Data Encryption Standard (DES) or 3DES) encryption software image from Cisco IOS Release 12.1(1)T or a later release. The SSH client requires the router to have an IPSec (DES or 3DES) encryption software image from Cisco IOS Release 12.1(3)T or a later release. See the Cisco IOS Configuration Fundamentals Configuration Guide, Release 12.4T for more information on downloading a software image.
The SSH server requires the use of a username and password, which must be defined through the use of a local username and password, TACACS+, or RADIUS.

Note
The SSH Terminal-Line Access feature is available on any image that contains SSH.

Restrictions for SSH Terminal-Line Access

**Console Server Requirement**
To configure secure console server access, you must define each line in its own rotary and configure SSH to use SSH over the network when user want to access each of those devices.

**Memory and Performance Impact**
Replacing reverse Telnet with SSH may reduce the performance of available tty lines due to the addition of encryption and decryption processing above the vty processing. (Any cryptographic mechanism uses more memory than a regular access.)

Information About SSH Terminal-Line Access

**Overview of SSH Terminal-Line Access**
Cisco IOS supports reverse Telnet, which allows users to Telnet through the router--via a certain port range--to connect them to tty (asynchronous) lines. Reverse Telnet has allowed users to connect to the console ports of remote devices that do not natively support Telnet. However, this method has provided very little security because all Telnet traffic goes over the network in the clear. The SSH Terminal-Line Access feature replaces reverse Telnet with SSH. This feature may be configured to use encryption to access devices on the tty lines, which provide users with connections that support strong privacy and session integrity.

SSH is an application and a protocol that provides secure replacement for the suite of Berkeley r-tools such as rsh, rlogin, and rep. (Cisco IOS supports rlogin.) The protocol secures the sessions using standard cryptographic mechanisms, and the application can be used similarly to the Berkeley rexec and rsh tools. Currently two versions of SSH are available: SSH Version 1 and SSH Version 2. Only SSH Version 1 is implemented in the Cisco IOS software.

The SSH Terminal-Line Access feature enables users to configure their router with secure access and perform the following tasks:

- Connect to a router that has multiple terminal lines connected to consoles or serial ports of other routers, switches, or devices.
- Simplify connectivity to a router from anywhere by securely connecting to the terminal server on a specific line.
- Allow modems attached to routers to be used for dial-out securely.
- Require authentication of each of the lines through a locally defined username and password, TACACS+, or RADIUS.
The session slot command that is used to start a session with a module requires Telnet to be accepted on the virtual tty (vty) lines. When you restrict vty lines only to SSH, you cannot use the command to communicate with the modules. This applies to any Cisco IOS device where the user can telnet to a module on the device.

How to Configure SSH Terminal-Line Access

Configuring SSH Terminal-Line Access

Perform this task to configure a Cisco router to support reverse secure Telnet.

SSH must already be configured on the router.

SUMMARY STEPS

1. enable
2. configure terminal
3. line line-number [ending-line-number]
4. no exec
5. login {local | authentication listname}
6. rotary group
7. transport input {all | ssh}
8. exit
9. ip ssh port portnum rotary group

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></td>
<td>Router&gt; enable</td>
</tr>
<tr>
<td><strong>Step 2</strong> configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td>Router# configure terminal</td>
</tr>
</tbody>
</table>
### Configuring SSH Terminal-Line Access

<table>
<thead>
<tr>
<th>Step 3</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>line</strong> line-number [ending-line-number]</td>
<td>Identifies a line for configuration and enters line configuration mode.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Router(config)# line 1 200</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** For router console configurations, each line must be defined in its own rotary, and SSH must be configured to listen in on each rotary. An authentication method requiring a username and password must be configured for each line. This may be done through the use of a local username and password stored on the router, through the use of TACACS+, or through the use of RADIUS. Neither Line passwords nor the enable password are sufficient to be used with SSH.

<table>
<thead>
<tr>
<th>Step 4</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>no exec</strong></td>
<td>Disables exec processing on each of the lines.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Router(config-line)# no exec</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 5</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>login</strong> {local</td>
<td>authentication listname}</td>
<td>Defines a login authentication mechanism for the lines.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Router(config-line)# login authentication default</td>
<td>The authentication method must utilize a username and password.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 6</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>rotary</strong> group</td>
<td>Defines a group of lines consisting of one or more lines.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Router(config-line)# rotary 1</td>
<td>All rotaries used must be defined, and each defined rotary must be used when SSH is enabled.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 7</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>transport</strong> input {all</td>
<td>ssh}</td>
<td>Defines which protocols to use to connect to a specific line of the router.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Router(config-line)# transport input ssh</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 8</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>exit</strong></td>
<td>Exits line configuration mode.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Router(config-line)# exit</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 9</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ip ssh port</strong> portnum rotary group</td>
<td>Enables secure network access to the tty lines.</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Router(config)# ip ssh port 2000 rotary 1</td>
<td>• Use this command to connect the portnum argument with the rotary group argument, which is associated with a line or group of lines.</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The group argument must correspond with the rotary group number chosen in Step 6.
Verifying SSH Terminal-Line Access

To verify that this functionality is working, you can connect to a router using an SSH client.

Configuration Examples for SSH Terminal-Line Access

Example SSH Terminal-Line Access Configuration

The following example shows how to configure the SSH Terminal-Line Access feature on a modem used for dial-out on lines 1 through 200. To get any of the dial-out modems, use any SSH client and start an SSH session to port 2000 of the router to get to the next available modem from the rotary.

```
line 1 200
no exec
login authentication default
rotary 1
transport input ssh
exit
ip ssh port 2000 rotary 1
```

Example SSH Terminal-Line Access for a Console Serial Line Ports Configuration

The following example shows how to configure the SSH Terminal-Line Access feature to access the console or serial line interface of various devices. For this type of access, each line is put into its own rotary, and each rotary is used for a single port. In this example, lines 1 through 3 are used; the port (line) mappings of the configuration are shown in the table below.

*Table 5: Port (line) Configuration Mappings*

<table>
<thead>
<tr>
<th>Line Number</th>
<th>SSH Port Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2001</td>
</tr>
<tr>
<td>2</td>
<td>2002</td>
</tr>
<tr>
<td>3</td>
<td>2003</td>
</tr>
</tbody>
</table>

```
line 1
no exec
login authentication default
rotary 1
transport input ssh
line 2
```
no exec
login authentication default
rotary 2
transport input ssh
line 3
no exec
login authentication default
rotary 3
transport input ssh
ip ssh port 2001 rotary 1 3

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>SSH</td>
<td>Cisco IOS Security Configuration Guide: Securing User Services</td>
</tr>
<tr>
<td>SSH commands</td>
<td>Cisco IOS Security Command Reference</td>
</tr>
<tr>
<td>Dial Technologies</td>
<td>Cisco IOS Dial Technologies Configuration Guide</td>
</tr>
<tr>
<td>Dial commands</td>
<td>Cisco IOS Dial Technologies Command Reference</td>
</tr>
<tr>
<td>Downloading a software image</td>
<td>Cisco IOS Configuration Fundamentals Configuration Guide</td>
</tr>
</tbody>
</table>

Standards

<table>
<thead>
<tr>
<th>Standard</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

MIBs

<table>
<thead>
<tr>
<th>MIB</th>
<th>MIBs Link</th>
</tr>
</thead>
<tbody>
<tr>
<td></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>RFC</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</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>
<tr>
<td>download documentation, software, and tools. Use these resources to install</td>
<td></td>
</tr>
<tr>
<td>and configure the software and to troubleshoot and resolve technical issues</td>
<td></td>
</tr>
<tr>
<td>with Cisco products and technologies. Access to most tools on the Cisco</td>
<td></td>
</tr>
<tr>
<td>Support and Documentation website requires a Cisco.com user ID and password.</td>
<td></td>
</tr>
</tbody>
</table>

**Feature Information for SSH Terminal-Line Access**

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 6: Feature Information for SSH Terminal-Line Access

<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Releases</th>
<th>Feature Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSH Terminal-Line Access</td>
<td>Cisco IOS 12.2(4)JA</td>
<td>The SSH Terminal-Line Access feature provides users secure access to tty (text telephone) lines. tty allows the hearing- and speech-impaired to communicate by using a telephone to type messages. This feature was introduced in Cisco IOS Release 12.2(4)JA. This feature was integrated into Cisco IOS Release 12.2(15)T. This feature was integrated into Cisco IOS Release 12.2(6th)S. The following command was introduced or modified: <code>ip ssh port</code>.</td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 12.2(15)T</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cisco IOS 12.2(6th)S</td>
<td></td>
</tr>
</tbody>
</table>
X.509v3 Certificates for SSH Authentication

The X.509v3 Certificates for SSH Authentication feature uses the X.509v3 digital certificates in server and user authentication at the secure shell (SSH) server side.

This module describes how to configure server and user certificate profiles for a digital certificate.

- Finding Feature Information, page 65
- Prerequisites for X.509v3 Certificates for SSH Authentication, page 66
- Restrictions for X.509v3 Certificates for SSH Authentication, page 66
- Information About X.509v3 Certificates for SSH Authentication, page 66
- How to Configure X.509v3 Certificates for SSH Authentication, page 67
- Configuration Examples for X.509v3 Certificates for SSH Authentication, page 71
- Additional References for X.509v3 Certificates for SSH Authentication, page 71
- Feature Information for X.509v3 Certificates for SSH Authentication, page 72

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 X.509v3 Certificates for SSH Authentication

- The X.509v3 Certificates for SSH Authentication feature introduces the **ip ssh server algorithm authentication** command to replace the **ip ssh server authenticate user** command. If you use the **ip ssh server authenticate user** command, the following deprecation message is displayed.
  Warning: SSH command accepted but this CLI will be deprecated soon. Please move to new CLI "ip ssh server algorithm authentication". Please configure "default ip ssh server authenticate user" to make CLI ineffective.

  - Use the **default ip ssh server authenticate user** command to remove the **ip ssh server authenticate user** command from effect. The IOS secure shell (SSH) server then starts using the **ip ssh server algorithm authentication** command.

Restrictions for X.509v3 Certificates for SSH Authentication

- The X.509v3 Certificates for SSH Authentication feature implementation is applicable only on the IOS secure shell (SSH) server side.

- IOS SSH server supports only the x509v3-ssh-rsa algorithm based certificate for server and user authentication on the IOS SSH server side.

Information About X.509v3 Certificates for SSH Authentication

Digital certificates

The validity of the authentication depends upon the strength of the linkage between the public signing key and the identity of the signer. Digital certificates in the X.509v3 format (RFC5280) are used to provide identity management. A chain of signatures by a trusted root certification authority and its intermediate certificate authorities binds a given public signing key to a given digital identity.

Public key infrastructure (PKI) trustpoint helps manage the digital certificates. The association between the certificate and the trustpoint helps track the certificate. The trustpoint contains information about the certificate authority (CA), different identity parameters, and the digital certificate. Multiple trustpoints can be created to associate with different certificates.

Server and user authentication using X.509v3

For server authentication, the IOS secure shell (SSH) server sends its own certificate to the SSH client for verification. This server certificate is associated with the trustpoint configured in the server certificate profile (ssh-server-cert-profile-server configuration mode).

For user authentication, the SSH client sends the user's certificate to the IOS SSH server for verification. The SSH server validates the incoming user certificate using public key infrastructure (PKI) trustpoints configured in the server certificate profile (ssh-server-cert-profile-user configuration mode).

By default, certificate-based authentication is enabled for server and user at the IOS SSH server end.
How to Configure X.509v3 Certificates for SSH Authentication

Configuring IOS SSH Server to Use Digital Certificates for Server Authentication

SUMMARY STEPS

1. enable
2. configure terminal
3. ip ssh server algorithm hostkey {x509v3-ssh-rsa [ssh-rsa] | ssh-rsa [x509v3-ssh-rsa]}
4. ip ssh server certificate profile
5. server
6. trustpoint sign PKI-trustpoint-name
7. ocsp-response include
8. end

DETAILED STEPS

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td>• Enter your password if prompted.</td>
</tr>
<tr>
<td></td>
<td>Device&gt; enable</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 2</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Device# configure terminal</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 3</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ip ssh server algorithm hostkey {x509v3-ssh-rsa [ssh-rsa]</td>
<td>ssh-rsa [x509v3-ssh-rsa]}</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa</td>
<td></td>
</tr>
</tbody>
</table>

**Note** The IOS SSH server must have at least one configured host key algorithm:

- ssh-rsa – public key based authentication
- x509v3-ssh-rsa – certificate-based authentication

<table>
<thead>
<tr>
<th>Step 4</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ip ssh server certificate profile</td>
<td>Configures server certificate profile and user certificate profile and enters SSH certificate profile configuration mode.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Device(config)# ip ssh server certificate profile</td>
<td></td>
</tr>
</tbody>
</table>
Configuring IOS SSH Server to Verify User's Digital Certificate for User Authentication

### SUMMARY STEPS

1. enable
2. configure terminal
3. ip ssh server algorithm authentication {publickey | keyboard | password}
4. ip ssh server algorithm publickey {x509v3-ssh-rsa | ssh-rsa | ssh-rsa [x509v3-ssh-rsa]}
5. ip ssh server certificate profile
6. user
7. trustpoint verify PKI-trustpoint-name
8. ocsp-response required
9. end
## 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** `ip ssh server algorithm authentication {publickey | keyboard | password}` | Defines the order of user authentication algorithms. Only the configured algorithm is negotiated with the secure shell (SSH) client.  |
| **Note** The IOS SSH server must have at least one configured user authentication algorithm.  
**Note** To use the certificate method for user authentication, the `publickey` keyword must be configured.  
**Note** The `ip ssh server algorithm authentication` command replaces the `ip ssh server authenticate user` command. |  |
| **Example:**  
  Device(config)# ip ssh server algorithm authentication publickey |  |
| **Step 4** `ip ssh server algorithm publickey {x509v3-ssh-rsa | ssh-rsa | x509v3-ssh-rsa}` | Defines the order of public key algorithms. Only the configured algorithm is accepted by the SSH client for user authentication.  |
| **Note** The IOS SSH client must have at least one configured public key algorithm:  
  * ssh-rsa – public-key-based authentication  
  * x509v3-ssh-rsa – certificate-based authentication |  |
| **Example:**  
  Device(config)# ip ssh server algorithm publickey x509v3-ssh-rsa |  |
| **Step 5** `ip ssh server certificate profile` | Configures server certificate profile and user certificate profile and enters SSH certificate profile configuration mode.  |
| **Example:**  
  Device(config)# ip ssh server certificate profile |  |
| **Step 6** `user` | Configures user certificate profile and enters SSH server certificate profile user configuration mode.  |
| **Example:**  
  Device(ssh-server-cert-profile)# user |  |
| **Step 7** `trustpoint verify PKI-trustpoint-name` | Configures the public key infrastructure (PKI) trustpoint that is used to verify the incoming user certificate.  |
| **Note** Configure multiple trustpoints by executing the same command multiple times. A maximum of 10 trustpoints can be configured. |  |
| **Example:**  
  Device(ssh-server-cert-profile-user)# trustpoint verify trust2 |  |
### Command or Action | Purpose
--- | ---
**Step 8** | 
ocsp-response required
**Example:**
Device(ssh-server-cert-profile-user)#
ocsp-response required
(Optional) Mandates the presence of the Online Certificate Status Protocol (OCSP) response with the incoming user certificate.
**Note** By default the "no" form of this command is configured and the user certificate is accepted without an OCSP response.

**Step 9** | end
**Example:**
Device(ssh-server-cert-profile-user)#
end
Exits SSH server certificate profile user configuration mode and enters privileged EXEC mode.

---

## Verifying Configuration for Server and User Authentication Using Digital Certificates

### SUMMARY STEPS

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

### DETAILED STEPS

**Step 1**
**enable**
Enables privileged EXEC mode.
- Enter your password if prompted.

**Example:**
Device> enable

**Step 2**
**show ip ssh**
Displays the currently configured authentication methods. To confirm the use of certificate-based authentication, ensure that the x509v3-ssh-rsa algorithm is the configured host key algorithm.

**Example:**
Device# show ip ssh
SSH Enabled - version 1.99
Authentication methods:publickey, keyboard-interactive, password
Authentication Publickey Algorithms:x509v3-ssh-rsa, ssh-rsa
Hostkey Algorithms: x509v3-ssh-rsa, ssh-rsa
Authentication timeout: 120 secs; Authentication retries: 3
Minimum expected Diffie Hellman key size: 1024 bits

Configuration Examples for X.509v3 Certificates for SSH Authentication

Example: Configuring IOS SSH Server to Use Digital Certificates for Server Authentication

Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa
Device(config)# ip ssh server certificate profile
Device(ssh-server-cert-profile)# server
Device(ssh-server-cert-profile-server)# trustpoint sign trust1
Device(ssh-server-cert-profile-server)# exit

Example: Configuring IOS SSH Server to Verify User's Digital Certificate for User Authentication

Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm authentication publickey
Device(config)# ip ssh server algorithm publickey x509v3-ssh-rsa
Device(config)# ip ssh server certificate profile
Device(ssh-server-cert-profile)# user
Device(ssh-server-cert-profile-user)# trustpoint verify trust2
Device(ssh-server-cert-profile-user)# end

Additional References for X.509v3 Certificates for SSH Authentication

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>
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>
<thead>
<tr>
<th>Feature Name</th>
<th>Releases</th>
<th>Feature Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>X.509v3 Certificates for SSH</td>
<td>Cisco IOS 15.5(1)S</td>
<td>The X.509v3 Certificates for SSH Authentication feature uses the X.509v3 digital</td>
</tr>
<tr>
<td>Authentication</td>
<td>Cisco IOS 15.5(2)T</td>
<td>certificates in server and user authentication at the secure shell (SSH) server side.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The following commands were introduced or modified: <code>ip ssh server algorithm hostkey</code>, <code>ip ssh server algorithm authentication</code>, and <code>ip ssh server certificate profile</code>.</td>
</tr>
</tbody>
</table>
SSH Algorithms for Common Criteria Certification

The SSH Algorithms for Common Criteria Certification feature provides the list and order of the algorithms that are allowed for Common Criteria Certification. This module describes how to configure the encryption, Message Authentication Code (MAC), and host key algorithms for a secure shell (SSH) server and client so that SSH connections can be limited on the basis of the allowed algorithms list.

- Finding Feature Information, page 75
- Information About SSH Algorithms for Common Criteria Certification, page 76
- How to Configure SSH Algorithms for Common Criteria Certification, page 77
- Configuration Examples For SSH Algorithms for Common Criteria Certification, page 82
- Additional References for SSH Algorithms for Common Criteria Certification, page 83
- Feature Information for SSH Algorithms for Common Criteria Certification, page 84

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.
Information About SSH Algorithms for Common Criteria Certification

SSH Algorithms for Common Criteria Certification

A Secure Shell (SSH) configuration enables a Cisco IOS SSH server and client to authorize the negotiation of only those algorithms that are configured from the allowed list. If a remote party tries to negotiate using only those algorithms that are not part of the allowed list, the request is rejected and the session is not established.

Cisco IOS SSH Server Algorithms

Cisco IOS secure shell (SSH) servers support the encryption algorithms (Advanced Encryption Standard Counter Mode [AES-CTR], AES Cipher Block Chaining [AES-CBC], Triple Data Encryption Standard [3DES]) in the following order:

1. aes128-ctr
2. aes192-ctr
3. aes256-ctr
4. aes128-cbc
5. 3des-cbc
6. aes192-cbc
7. aes256-cbc

Cisco IOS SSH servers support the Message Authentication Code (MAC) algorithms in the following order:

1. hmac-sha1
2. hmac-sha1-96

Cisco IOS SSH servers support the host key algorithms in the following order:

1. x509v3-ssh-rsa
2. ssh-rsa

Cisco IOS SSH Client Algorithms

Cisco IOS secure shell (SSH) clients support the encryption algorithms (Advanced Encryption Standard counter mode [AES-CTR], AES Cipher Block Chaining [AES-CBC], Triple Data Encryption Standard [3DES]) in the following order:

1. aes128-ctr
Cisco IOS SSH clients support the Message Authentication Code (MAC) algorithms in the following order:

1. hmac-sha1
2. hmac-sha1-96

Cisco IOS SSH clients support only one host key algorithm and do not need a CLI configuration:

- ssh-rsa

How to Configure SSH Algorithms for Common Criteria Certification

Configuring an Encryption Key Algorithm for a Cisco IOS SSH Server and Client

SUMMARY STEPS

1. enable
2. configure terminal
3. ip ssh {server | client} algorithm encryption {aes128-ctr | aes192-ctr | aes256-ctr | aes128-cbc | 3des-cbc | aes192-cbc | aes256-cbc}
4. end

DETAILED STEPS

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Enter your password if prompted.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Device&gt; enable</td>
<td></td>
</tr>
</tbody>
</table>
### Configuring a MAC Algorithm for a Cisco IOS SSH Server and Client

**SUMMARY STEPS**

1. enable
2. configure terminal
3. `ip ssh {server | client} algorithm mac {hmac-sha1 | hmac-sha1-96}`
4. `end`

---

**Troubleshooting Tips**

If you try to disable the last encryption algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All encryption algorithms cannot be disabled
```
## 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><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>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><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>Device# configure terminal</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> ip ssh {server</td>
<td>client} algorithm mac {hmac-sha1</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>Device(config)# ip ssh server algorithm mac hmac-sha1 hmac-sha1-96</td>
<td>Note: The Cisco IOS SSH server and client must have at least one configured Hashed Message Authentication Code (HMAC) algorithm.</td>
</tr>
<tr>
<td>Device(config)# ip ssh client algorithm mac hmac-sha1 hmac-sha1-96</td>
<td>Note: To disable one algorithm from the previously configured algorithm list, use the no form of this command. To disable more than one algorithm, use the no form of this command multiple times with different algorithm names.</td>
</tr>
<tr>
<td></td>
<td>Note: For default configuration, use the default form of this command as shown below:</td>
</tr>
<tr>
<td></td>
<td>Device(config)# ip ssh server algorithm mac hmac-sha1 hmac-sha1-96</td>
</tr>
<tr>
<td></td>
<td>Device(config)# ip ssh client algorithm mac hmac-sha1 hmac-sha1-96</td>
</tr>
<tr>
<td><strong>Step 4</strong> end</td>
<td>Exits global configuration mode and returns to privileged EXEC mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>Device(config)# end</td>
<td></td>
</tr>
</tbody>
</table>

## Troubleshooting Tips

If you try to disable the last MAC algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All mac algorithms cannot be disabled
```
# Configuring a Host Key Algorithm for a Cisco IOS SSH Server

## SUMMARY STEPS

1. enable
2. configure terminal
3. `ip ssh server algorithm hostkey {x509v3-ssh-rsa|ssh-rsa}`
4. 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></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 server algorithm hostkey {x509v3-ssh-rsa</td>
<td>ssh-rsa}`</td>
</tr>
<tr>
<td>Example: Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa ssh-rsa</td>
<td></td>
</tr>
<tr>
<td><strong>Note</strong> The Cisco IOS SSH server must have at least one configured host key algorithm:</td>
<td></td>
</tr>
<tr>
<td>• x509v3-ssh-rsa—X.509v3 certificate-based authentication</td>
<td></td>
</tr>
<tr>
<td>• ssh-rsa—Public-key-based authentication</td>
<td></td>
</tr>
<tr>
<td><strong>Note</strong> To disable one algorithm from the previously configured algorithm list, use the <code>no</code> form of this command. To disable more than one algorithm, use the <code>no</code> form of this command multiple times with different algorithm names.</td>
<td></td>
</tr>
<tr>
<td><strong>Note</strong> For default configuration, use the default form of this command as shown below:</td>
<td></td>
</tr>
<tr>
<td>Example: Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa ssh-rsa</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> end</td>
<td>Exits global configuration mode and returns to privileged EXEC mode.</td>
</tr>
<tr>
<td>Example: Device(config)# end</td>
<td></td>
</tr>
</tbody>
</table>
**Troubleshooting Tips**

If you try to disable the last host key algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All hostkey algorithms cannot be disabled
```

---

**Verifying SSH Algorithms for Common Criteria Certification**

**SUMMARY STEPS**

1. `enable`
2. `show ip ssh`

**DETAILED STEPS**

**Step 1**

`enable`

Enables privileged EXEC mode.

- Enter your password if prompted.

**Example:**

```
Device> enable
```

**Step 2**

`show ip ssh`

Displays configured Secure Shell (SSH) encryption, host key, and Message Authentication Code (MAC) algorithms.

**Example:**

The following sample output from the `show ip ssh` command shows the encryption algorithms configured in the default order:

```
Device# show ip ssh
```

The following sample output from the `show ip ssh` command shows the MAC algorithms configured in the default order:

```
Device# show ip ssh
MAC Algorithms: hmac-sha1 hmac-sha1-96
```
The following sample output from the `show ip ssh` command shows the host key algorithms configured in the default order:

```
Device# show ip ssh
Hostkey Algorithms: x509v3-ssh-rsa, ssh-rsa
```

### Configuration Examples For SSH Algorithms for Common Criteria Certification

#### Example: Configuring Encryption Key Algorithms for a Cisco IOS SSH Server

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm encryption aes128-ctr aes192-ctr aes256-ctr aes128-cbc
                      3des-cbc aes192-cbc aes256-cbc
Device(config)# end
```

#### Example: Configuring Encryption Key Algorithms for a Cisco IOS SSH Client

```
Device> enable
Device# configure terminal
Device(config)# ip ssh client algorithm encryption aes128-ctr aes192-ctr aes256-ctr aes128-cbc
                      3des-cbc aes192-cbc aes256-cbc
Device(config)# end
```

#### Example: Configuring MAC Algorithms for a Cisco IOS SSH Server

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm mac hmac-shal hmac-shal-96
Device(config)# end
```

#### Example: Configuring MAC Algorithms for a Cisco IOS SSH Client

```
Device> enable
Device# configure terminal
Device(config)# ip ssh client algorithm mac hmac-shal hmac-shal-96
Device(config)# end
```
Example: Configuring Host Key Algorithms for a Cisco IOS SSH Server

Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa ssh-rsa
Device(config)# end

Additional References for SSH Algorithms for Common Criteria Certification

<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>
| Security commands | • Cisco IOS Security Command Reference: Commands A to C  
 | | • Cisco IOS Security Command Reference: Commands D to L  
 | | • Cisco IOS Security Command Reference: Commands M to R  
 | | • Cisco IOS Security Command Reference: Commands S to Z |
| SSH authentication | “Secure Shell-Configuring User Authentication Methods” chapter in the Secure Shell Configuration Guide |
| X.509v3 digital certificates in server and user authentication | “X.509v3 Certificates for SSH Authentication” chapter in the Secure Shell Configuration Guide |
Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support 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 SSH Algorithms for Common Criteria Certification

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 8: Feature Information for SSH Algorithms for Common Criteria Certification

<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Releases</th>
<th>Feature Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSH Algorithms for Common Criteria Certification</td>
<td>Cisco IOS 15.5(2)T Cisco IOS 15.5(2)S</td>
<td>The SSH Algorithms for Common Criteria Certification feature provides the list and order of the algorithms that are allowed for Common Criteria Certification. This module describes how to configure the encryption, Message Authentication Code (MAC), and host key algorithms for a secure shell (SSH) server and client so that SSH connections can be limited on the basis of the allowed algorithms list. The following commands were introduced by this feature: ip ssh {server</td>
</tr>
</tbody>
</table>