Chapter 43

Secure Shell

Introduction

This chapter describes the Secure Shell protocol, support for Secure Shell on the router, how to configure the router to act both as a Secure Shell server and client, and how to use Secure Shell to manage the router.

Secure management is increasingly important in modern networks, as the ability to easily and effectively manage routers and the requirement for security are two almost universal requirements. Traditionally, routers are managed using either remote terminal sessions via the Telnet protocol or SNMP. Both methods have serious security problems. They are protected by plaintext reusable passwords that are vulnerable to wiretapping and password guessing, which could result in sensitive information, even the passwords themselves, traversing the network as plaintext.

The Secure Shell (SSH) protocol is similar to the Telnet and rlogin protocols in providing encrypted and strongly authenticated remote login sessions. SSH provides sessions between a host running a Secure Shell server and a machine with a Secure Shell client.

The AR400 router implements both a Secure Shell server and a Secure Shell client to enable network managers to securely—with the benefit of cryptographic authentication and encryption—manage the routers over an insecure network:

- Secure Shell replaces Telnet for remote terminal sessions. Secure Shell is strongly authenticated and encrypted.
- Remote command execution allows manager commands to be sent to a router securely and conveniently, without requiring a terminal session on the router.
- A network manager logged on to an AR400 router may connect to another router or a secure host using Secure Shell.

Secure Shell on the Router

The AR400 router implementation of the Secure Shell protocol is compatible with the Secure Shell protocol described in the following Internet Draft:

The SSH (Secure Shell) Remote Login Protocol, T. Ylonen, 15 November 1995.

The protocol depends on other technologies, including DES encryption and RSA public key encryption, that may require licencing in some regions, and may be subject to ITAR export controls.

The following Secure Shell options and features are supported:

- Inbound Secure Shell connections (server mode) and outbound Secure Shell connections (client mode).
- RSA public keys with lengths of 512–2048 bits. Keys are stored in a format compatible with other Secure Shell implementations, and mechanisms are provided to copy keys to and from the router.
- Single DES and Triple-DES (3DES) encryption for Secure Shell sessions. 3DES encryption requires a special feature licence.
- Remote non-interactive shell allows arbitrary router commands to be sent securely to the router, possibly automatically.

All functions work whether a Mini Accelerator Card (MAC) card is installed. If a MAC card is installed, it improves performance.

The following Secure Shell options and features are **not** supported:

- IDEA or Blowfish encryption.
- Compression of Secure Shell traffic.
- Non-encrypted Secure Shell sessions. The purpose of Secure Shell is to provide encrypted and strongly-authenticated terminal sessions so the Secure Shell server refuses to operate in non-encrypted mode.
- Tunnelling of TCP/IP traffic. Instead IP encryption should be implemented using Security Associations.

Log messages are generated to record the following events:

- The addition of a Secure Shell user.
- The deletion of a Secure Shell user.
- The modification of a Secure Shell user.
- The Secure Shell server has been enabled.
- The Secure Shell server has been disabled.
- A Secure Shell connection has been accepted.
- A Secure Shell connection has been rejected.
- A Secure Shell connection has been disconnected.

The Secure Shell implementation is functionally similar to existing implementations on various platforms, such as ssh-1.2.26 (UNIX) and Datafellows F-Secure SSH (Windows), and interoperates with these implementations.

Configuring Secure Shell

When the Secure Shell server is enabled, connections from Secure Shell clients can be accepted. If a firewall policy exists on the Secure Shell server, you must add a rule that allows access to the Secure Shell server on port 22 before Secure Shell connections are accepted. When the Secure Shell server is disabled, connections from Secure Shell clients are rejected.

To add a firewall rule that accepts Secure Shell connections, use the **add firewall policy rule** command on page 41-43 of Chapter 41, Firewall.

To enable or disable the Secure Shell server, use the commands:

```
enable ssh server hostkey=keyid serverkey=keyid
   [expirytime=hours] [logintimeout=seconds]

disable ssh server
```

To modify the Secure Shell server configuration, use the command:

```
set ssh server [hostkey=keyId] [serverkey=keyId]
   [expirytime=hours] [logintimeout=seconds]
```

To display the current status and configuration of the Secure Shell server, use the command:

```
show ssh [counter]
```

The status of all Secure Shell sessions currently active on the router, including both outbound sessions to another host and inbound sessions into the router, can be monitored using the command:

```
show ssh sessions
```

The Secure Shell server accepts only connections from registered users. When there are no registered users, the router does not accept a Secure Shell connections even when the Secure Shell server is enabled. The following commands add and delete registered users:

```
add ssh user=username {password=password|keyid=id}
   [ipaddress=ipadd] [mask=mask]

delete ssh user=username
```

To modify the configuration for an existing registered user, use the command:

```
set ssh user=username {password=password|keyid=id}
   [ipaddress=ipadd] [mask=mask]
```

To temporarily disable and enable a registered user, use the commands:

```
disable ssh user=username

enable ssh user=username
```

A user that is disabled cannot make Secure Shell connections to the Secure Shell server. If a Secure Shell user makes 5 consecutive incorrect logins, their status changes to disabled. To display a list of registered users and their current status, use the command:

```
show ssh user[=username]
```

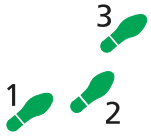
To initiate a Secure Shell connection from the router to a remote router or Secure Shell host, use the command:

```
ssh ipadd user=username {password=password|keyid=id}
   [command=string]
```

Secure Shell encryption key management is implemented by the ENCO module. RSA public keys can be imported and exported to/from the single-line ASCII format used by all Secure Shell implementations.

Configuration Example

The following example illustrates the steps to create encryption keys, configure the Secure Shell server, and register users to make Secure Shell connections to the router.



To configure Secure Shell

1. Create a Security Officer login on the server.

A Security Officer must be configured before putting the router in security mode. Add a user with Security Officer privilege to the User Authentication Database:

```
add user=login-name login={true|false|on|off|yes|no}
   password=password privilege=securityofficer
   [other-user-parameters]
```

See the **add user** command on page 1-65 of Chapter 1, Operation.

2. Enable security mode.

To store the Secure Shell configuration and other security information, such as encryption keys when the router is restarted, enable the security mode as follows:

```
enable system security_mode
```

When the router is in security mode, only a user with Security Officer privilege can configure security-sensitive aspects of it. See "Normal Mode and Security Mode" on page 1-6 of Chapter 1, Operation.

Sensitive data, such as encryption keys, created on a router that is not in security mode is destroyed when the router is restarted. Enable security mode before creating encryption keys.

3. Create encryption keys.

Two RSA private keys are required to enable the Secure Shell server. The first, is the *host key*, which is the router's own RSA key. The recommended length of this key is 1024 bits. The second key, the *server key*, is a randomly created key, which is re-generated after the specified timeout. This key must be 128 bits shorter than the host key, but should be at least 512 bits.

Before creating keys, first define a user with Security Officer privileges. Defining a security officer lets that user enable security mode when logging in. The security officer definition process must be carried out on all routers that use the keys (i.e., on the head office router and the remote office router).

```
create enco key=0 type=rsa length=1024
   description="Host Key" form=ssh

create enco key=1 type=rsa length=768
   description="Server Key" form=ssh
```

To verify the key creation, use the command:

```
show enco key
```

4. Optionally add a firewall rule that accepts Secure Shell connections.

If you have a firewall enabled on the Secure Shell server, you must add a rule that accepts Secure Shell connections to the server. For example, to add Secure Shell access over port 22 with a public IP address of 200.200.200.1, a private IP address of 192.168.1.1, a public interface of ppp0, and a remote IP address of 200.200.200.5, use the command:

```
add firewall policy=main rule=1 action=allow interface=ppp0
   protocol=tcp port=22 ipaddress=192.168.1.1
   gblip=200.200.200.1 gblport=22 remote=200.200.200.5
```

5. Enable the Secure Shell server.

Once the required host and server keys have been generated, the Secure Shell server can be enabled using the command:

```
enable ssh server hostkey=0 serverkey=1 expirytime=1
   logintimeout=60
```

In this case, the server key expiry time has been set to 1 hour. Because routers are always busy (unlike a UNIX box), it is recommended that the expiry time be set to ensure that RSA keys are re-generated in off-peak times. One method of doing this is to set the expiry time to 0 (never) and then use the router script facility to destroy and recreate the key.

To verify the SSH server status, use the command:

```
show ssh
```

6. Create SSH users.

There are two ways to authenticate an SSH session: password authentication and RSA authentication.

To assign password authentication.

(With a password of "secret" to a user called "John".)

```
add ssh user=john password=secret
```

To assign RSA authentication.

Create an RSA public key for the user. This RSA public key must be the public part of the RSA private key that the user sends when opening the SSH session.

To create the public key on the server you must:

(a) Create the public key file. This can be done in a number of ways by using a variety of SSH client programs. It can also be done on the router. In this example, we are creating the public key on the client router. (The remote router in this example is called the server router.)

First, create a private key on the **client** router:

```
create enco key=5 type=rsa length=768 description="Example
   private key"
```

The length of the key is flexible, it does not have to be 768 bits.

Then, from the private key, create a public key in a file

```
create enco key=5 type=rsa file=user.key form=ssh
```

The key number entered in this command must be the same as that used in the previous command because the purpose of this command is to create a public key from the private key. The file name must have a ".key" extension.

Once the key file has been created, it can be uploaded to a TFTP server or zmodem server so that it can be loaded onto the SSH server:

```
upload server=ipadd file=user.key
```

or

```
upload po=0 method=zmo file=user.key
```

(b) Load the public key file onto the **server** router.

```
load server=ipadd destination=flash file=key-file-name
```

or

```
load po=0 meth=zmo destination=flash
```

(c) Create an ENCO key from this file on the **server** router:

```
create enco key=7 type=rsa file=user.key
   description="user's public key" form=ssh
```

Now that the public key is created on the server, it is possible to create the SSH user:

```
add ssh user=name keyid=7
```

Other SSH user parameters.

An IP address or subnet can also be added to force the user to login from that address. If the Secure Shell user is also a member of the router's User Authentication Database, then the user has the associated privileges.

To require the user "John" to attach from the network IP 192.168.2.0, use the command:

```
set ssh user=john ipaddress=192.168.2.0 MASK=255.255.255.0
```

Verify the list of registered Secure Shell users:

```
show ssh user
```

7. Create an outgoing SSH terminal session.

Use the authentication method (password or RSA) that matches the method from the previous step to create users.

Password authentication.

To create an outgoing terminal session using password authentication, follow this procedure:

A user "John" has been configured with password authentication (password "secret") on the server router 192.168.2.5. To use Secure Shell to login to the remote router as user "John", on the **client** router, use the command:

```
ssh 192.168.2.5 user=john password=secret
```

The first time that you make an SSH connection from a client router to a server router using password authentication, the server router sends the public part of its host key to the client router. Because the client router has not seen that host key before, it presents the message:

```
Host key not recognised - saved as ssh.key

SSH. Session closed.
```

At this point it is necessary to create an ENCO key from the saved public key file.

To create the ENCO key, use the command:

```
create enco key=6 type=rsa file=ssh.key
```

```
    description="B host key" form=ssh
```

Then you can successfully create an SSH connection to the server router:

```
ssh 192.168.2.5 user=john PASSWORD=secret
```

RSA authentication.

To create the outgoing terminal session using RSA authentication, follow this procedure:

A user "Admin" has been configured with an RSA public key (matching RSA private key 5 on the local router) on the server router 192.168.2.5. To use Secure Shell to login to the server router as user "Admin", on the client router, use the command:

```
ssh 192.168.2.5 user=Admin keyid=5
```

Executing commands on the remote router.

It is possible to make an SSH connection that simply connects to the server, executes a command on the server, and disconnects again.

```
ssh 192.168.2.5 user=john password=secret
    command="act scr=test.scp"
```

In order to execute commands that require manager or security officer privilege, the server router must be configured with a manager or security officer user corresponding to the SSH user.

For example, having created the SSH user john_passwd, it is necessary to also create john_passwd in the main user database:

```
add user=john_passwd password=secret privilege=manager
```

The password configured for this user is not important; it does not have to match the password for the SSH user. The SSH user john_rsa can have a corresponding manager user created:

```
add user=john_rsa password=any_old_thing privilege=manager
```

Command Reference

This section describes the commands available on the router to enable, configure, control, and monitor the Secure Shell client and server.

The shortest valid command is denoted by capital letters in the Syntax section. See "Conventions" on page xcv of Preface in the front of this manual for details of the conventions used to describe command syntax. See Appendix A, Messages for a complete list of messages and their meanings.

add ssh user

Syntax

```
ADD SSH USER=username {PASSword=password|KEYid=key-id}
   [IPaddress=ipadd|ipv6add] [MASk=mask]
```

where:

- *username* is a character string 1 to 15 characters long. Valid characters are uppercase and lowercase letters, and decimal digits (0–9). The string may not contain spaces.
- *password* is a character string 1 to 31 characters long. Valid characters are any printable character. If the string contains spaces, put double quotes around it.
- *key-id* is a decimal key ID.
- *ipadd* is an IP address in dotted decimal notation.
- *ipv6add* is a valid IPv6 address.
- *mask* is an IP network mask in dotted decimal notation.

Description

This command adds an entry to the list of users who are permitted to connect to the router via Secure Shell. If the user list is empty, no Secure Shell connections are allowed. If the Secure Shell user is also a member of the router user database, then that user has the associated privileges.

The USER parameter specifies the username that the user must supply when connecting to the router using a Secure Shell client.

The PASSWORD parameter specifies the password that the user must supply when connecting to the router using a Secure Shell client.

The KEY parameter specifies the RSA key to be used to authenticate the user. This parameter enables Secure Shell RSA authentication. The specified key must exist.

The IPADDRESS and MASK parameters specify the range of IP addresses from which the named user may connect. By default, connections are accepted from any IP address.

For security reasons, when security mode is enabled, this command can be issued only by a user with Security Officer privilege. If security mode is disabled, sensitive data, for example encryption keys created on a router that is not in security mode, is destroyed when the router is restarted.

See the **add user** command on page 1-65 of Chapter 1, Operation and **enable system security_mode** command on page 1-93 of Chapter 1, Operation.

Example To add a Secure Shell user with password authentication and a specified IP address, use the command:

```
add ssh user=john pass=secret ip=192.168.2.1
   mas=255.255.255.0
```

To add a Secure Shell user with RSA authentication, use the command:

```
add ssh user=john key=5
```

Related Commands
delete ssh user
set ssh user
show ssh user

delete ssh user

Syntax `DELete SSH USER=`*`username`*

where *username* is a character string 1 to 15 characters long. Valid characters are uppercase and lowercase letters, and decimal digits (0–9). The string may not contain spaces.

Description This command deletes the specified user from the list of users permitted to connect to the router via Secure Shell.

For security reasons, when security mode is enabled, this command can be issued only by a user with Security Officer privilege. If security mode is disabled, sensitive data, for example encryption keys created on a router that is not in security mode, is destroyed when the router is restarted.

See the **add user** command on page 1-65 of Chapter 1, Operation and **enable system security_mode** command on page 1-93 of Chapter 1, Operation.

Example To delete a Secure Shell user use the command:

```
del ssh user=john
```

Related Commands
add ssh user
set ssh user
show ssh user

disable ssh server

Syntax `DISable SSH SERVer`

Description This command disables the Secure Shell server. When the Secure Shell server is disabled, connections from Secure Shell clients are not accepted.

The Secure Shell server is disabled by default. Secure Shell sessions may be initiated from the router to another host, but inbound connections are not accepted.

For security reasons, when security mode is enabled, this command can be issued only by a user with Security Officer privilege. If security mode is disabled, sensitive data, for example encryption keys created on a router that is not in security mode, is destroyed when the router is restarted.

See the **add user** command on page 1-65 of Chapter 1, Operation and **enable system security_mode** command on page 1-93 of Chapter 1, Operation.

Example To disable the Secure Shell server, use the command:

```
dis ssh serve
```

Related Commands
enable ssh server
set ssh server
show ssh

disable ssh user

Syntax `DISable SSH USER=`*`username`*

where *username* is a character string 1 to 15 characters long. Valid characters are uppercase and lowercase letters, and digits (0–9). The string may not contain spaces.

Description This command disables a Secure Shell user so that they cannot login. By default, users are enabled when they are added to the list of permitted users.

For security reasons, when security mode is enabled, this command can be issued only by a user with Security Officer privilege. If security mode is disabled, sensitive data, for example encryption keys created on a router that is not in security mode, is destroyed when the router is restarted.

See the **add user** command on page 1-65 of Chapter 1, Operation and **enable system security_mode** command on page 1-93 of Chapter 1, Operation.

Example To disable a Secure Shell user, use the command:

```
dis ssh user
```

Related Commands
add ssh user
delete ssh user
enable ssh user
set ssh user
show ssh user

enable ssh server

Syntax

```
ENAble SSH SERVer HOSTkey=key-id SERVERkey=key-id
   [EXPirytime=hours] [LOGintimeout=seconds]
```

where:

- *key-id* is a decimal key ID.
- *hours* is the time in hours.
- *seconds* is the time in seconds.

Description

This command enables the Secure Shell server. When the Secure Shell server is enabled, connections from Secure Shell clients are accepted. If a firewall is configured on the router, you must add a rule that accepts Secure Shell connections before you can enable the Secure Shell server.

The HOSTKEY parameter specifies the key that must be used for the router host key. The specified key must exist.

The SERVERKEY parameter specifies the key that must be used for the Secure Shell server key. The specified key must exist.

The EXPIRYTIME parameter specifies the time in hours after which the Secure Shell server key expires and is regenerated. If 0 is specified, the key does not expire. The default is 0.

The LOGINTIMEOUT parameter specifies how long the server waits before disconnecting an un-authenticated client. The default is 1 minute.

By default the Secure Shell server is disabled. Secure Shell sessions may be initiated from the router to another host but inbound connections are not accepted.

For security reasons, when security mode is enabled, this command can be issued only by a user with Security Officer privilege. If security mode is disabled, sensitive data, for example encryption keys created on a router that is not in security mode, is destroyed when the router is restarted.

See the **add user** command on page 1-65 of Chapter 1, Operation and **enable system security_mode** command on page 1-93 of Chapter 1, Operation.

Example

To enable the Secure Shell server, use the command:

```
ena ssh serve host=0 server=1
```

Related Commands

disable ssh server
set ssh server
show ssh

enable ssh user

Syntax `ENAble SSH USER=`*`username`*

where *username* is a character string 1 to 15 characters long. Valid characters are uppercase and lowercase letters, and digits (0–9). The string may not contain spaces.

Description This command enables a Secure Shell user. By default, users are enabled when they are added to the list of permitted users. However, if a user fails to login five times in a row, the user is disabled. This command enables the user again.

For security reasons, when security mode is enabled, this command can be issued only by a user with Security Officer privilege. If security mode is disabled, sensitive data, for example encryption keys created on a router that is not in security mode, is destroyed when the router is restarted.

See the **add user** command on page 1-65 of Chapter 1, Operation and **enable system security_mode** command on page 1-93 of Chapter 1, Operation.

Example To enable a Secure Shell user who has been disabled, use the command:

```
ena ssh user=john
```

Related Commands

add ssh user
delete ssh user
disable ssh user
set ssh user
show ssh user

set ssh server

Syntax

```
SET SSH SERVer [HOSTkey=key-id] [SERVERkey=key-id]
   [EXPirytime=hours] [IDLEtimeout=0..4294967295]
   [LOGintimeout=seconds] [MAXSessions=0..30]
```

where:

- *key-id* is a decimal key ID.
- *hours* is the time in hours.
- *seconds* is the time in seconds.

Description This command modifies the configuration of the Secure Shell server.

The HOSTKEY parameter specifies the key for the router host. The specified key must exist.

The SERVERKEY parameter specifies the key for the Secure Shell server. The specified key must exist.

The EXPIRYTIME parameter specifies the time in hours after which the Secure Shell server key expires. If 0 is specified, the key does not expire. The default is 0.

The IDLETIMEOUT parameter specifies a period of time, in seconds, for the Telnet server's idle timer. If the specified time period lapses since the last time a Telnet session received data from the remote client, the session is terminated; this applies from the moment that the Telnet session becomes established, regardless of whether the user has logged in or not. If zero is specified, the idle timer remains off, and the session must be explicitly terminated. The default is zero.

If the Telnet server idle timeout period is modified while there are established Telnet sessions, the idle timers for those sessions are reset so that they use the new timeout value. Any idle time accumulated by those sessions prior to the issuing of the set command is lost.

The LOGINTIMEOUT parameter specifies the length of time the server waits before disconnecting an un-authenticated client. The default is 1 minute.

By default the Secure Shell server is disabled. Secure Shell sessions may be initiated from the router to another host, but inbound connections are not accepted.

For security reasons, when security mode is enabled, this command can be issued only by a user with Security Officer privilege. If security mode is disabled, sensitive data, for example encryption keys created on a router that is not in security mode, is destroyed when the router is restarted.

See the **add user** command on page 1-65 of Chapter 1, Operation and **enable system security_mode** command on page 1-93 of Chapter 1, Operation.

The MAXSESSIONS parameter specifies the number of concurrent Telnet sessions that are supported by the local switch. Once this limit is reached, any subsequent session requests are rejected. The session limit cannot be set below the number of currently established Telnet sessions. The default, is 30.

Example To set the Secure Shell server key expiry time to 1 hour, use the command:

```
set ssh serve exp=1
```

Related Commands

disable ssh server
enable ssh server
show ssh

set ssh user

Syntax

```
SET SSH USER=username {PASSword=password|KEYid=key-id}
   [IPaddress=ipadd|ipv6add] [MASk=mask]
```

where:

- *username* is a character string from 1 to 15 characters long. Valid characters are uppercase and lowercase letters, and decimal digits (0–9). The string may not contain spaces.
- *password* is a character string from 1 to 31 characters long. Valid characters are any printable character. If the string contains spaces, put double quotes around it.
- *key-id* is a decimal key ID.
- *ipadd* is an IP address in dotted decimal notation.
- *ipv6add* is a valid IPv6 address.
- *mask* is an IP network mask in dotted decimal notation.

Description

This command modifies an entry in the list of users that are permitted to connect to the router via Secure Shell. The user must already exist.

The USER parameter specifies the username that the user must supply when connecting to the router using a Secure Shell client.

The PASSWORD parameter specifies the password that the user must supply when connecting to the router using a Secure Shell client.

The KEY parameter specifies the RSA key to authenticate the user. This parameter enables Secure Shell RSA authentication. The specified key must exist.

The IPADDRESS and MASK parameters specify the range of IP addresses from which the named user may connect. By default, connections are accepted from any IP address.

For security reasons, when security mode is enabled, this command can be issued only by a user with Security Officer privilege. If security mode is disabled, sensitive data, for example encryption keys created on a router that is not in security mode, is destroyed when the router is restarted.

See the **add user** command on page 1-65 of Chapter 1, Operation and **enable system security_mode** command on page 1-93 of Chapter 1, Operation.

Example

To change a Secure Shell user to RSA authentication or to change their key, use the command:

```
set ssh user=john key=5
```

Related Commands

add ssh user
delete ssh user
show ssh user

show ssh

Syntax `SHow SSH [COUnter]`

Description This command displays the status and configuration of the Secure Shell client and server.

If COUNTER is not specified, the current configuration of the Secure Shell server is displayed (Figure 43-1, Table 43-1 on page 43-16).

If COUNTER is specified, counters for the Secure Shell client and server are displayed (Figure 43-2 on page 43-17, Table 43-2 on page 43-18).

Figure 43-1: Example output from the **show ssh** command

```
Secure Shell Configuration
-------------------------------------------------------------
Version..................... 1.5
Server Enabled.............. TRUE
Port........................ 22
Host Key ID................. 0
Host Key Bits............... 1024
Server Key ID............... 1
Server Key Bits............. 768
Server Key Expiry(hours).. 0
Login Timeout (secs)....... 60
Authentication Available.. Password,RSA
Ciphers Available.......... DES,3DES
Services Available......... Shell,Cmd
```

Table 43-1: Parameters in the output of the **show ssh** command

Parameter	Meaning
Version	The compatible version of the Secure Shell protocol.
Server Enabled	Whether the Secure Shell server is enabled.
Port	The TCP port where the Secure Shell server listens for connections. The default is port 22.
Host Key ID	The ENCO key ID for the router's host RSA key. The Secure Shell server cannot operate without this key.
Host Key Bits	The length of the router host key. This must be from 768 to 2048 bits.
Server Key ID	The ENCO key ID for the Secure Shell server RSA key. The Secure Shell server cannot operate without this key.
Server Key Bits	The length of the Secure Shell server key. This must be at least 128 bits less than the host key (minimum 512 bits).
Server Key Expiry (hours)	The time in hours before the Secure Shell server key expires and is regenerated.
Login Timeout (secs)	The time in seconds when a client must authenticate itself before it is disconnected.
Authentication Available	A list of available authentication methods; either Password or RSA.
Ciphers Available	A list of the ciphers available; either DES or 3DES.
Services Available	A list of the available Secure Shell services; either Shell or Cmd.

Figure 43-2: Example output from the **show ssh counter** command

```
Secure Shell Counters:

  inOctets                     10407  outOctets                    16371
  rxPkt                          391  txPkt                          424
  rxPktCheckFail                   0  txPktFail                        0
  rxVersionID                     11  txVersionID                     11

  rxMSGDisconnect                  0  txMSGDisconnect                  3
  rxSMSGPublicKey                  0  txSMSGPublicKey                 11
  rxCMSGSessionKey                11  txCMSGSessionKey                 0
  rxCMSGUser                      11  txCMSGUser                       0
  rxCMSGAuthRhosts                 0  txCMSGAuthRhosts                 0
  rxCMSGAuthRSA                   10  txCMSGAuthRSA                    0
  rxSMSGAuthRSAChallenge           0  txSMSGAuthRSAChallenge           3
  rxCMSGAuthRSAResponse            3  txCMSGAuthRSAResponse            0
  rxCMSGAuthPassword               8  txCMSGAuthPassword               0
  rxCMSGAuthRhostsRSA              0  txCMSGAuthRhostsRSA              0
  rxSMSGSuccess                    0  txSMSGSuccess                   19
  rxSMSGFailure                    0  txSMSGFailure                   25
  rxCMSGReqCompression             0  txCMSGReqCompression             0
  rxCMSGReqX11Forwarding           0  txCMSGReqX11Forwarding           0
  rxCMSGReqPortForwarding          0  txCMSGReqPortForwarding          0
  rxCMSGReqAgentForwarding         0  txCMSGReqAgentForwarding         0
  rxCMSGReqPty                     4  txCMSGReqPty                     0
  rxCMSGWindowSize                 0  txCMSGWindowSize                 0
  rxCMSGExecShell                  4  txCMSGExecShell                  0
  rxCMSGExecCmd                    0  txCMSGExecCmd                    0
  rxCMSGStdInData                326  txCMSGStdInData                  0
  rxSMSGStdOutData                 0  txSMSGStdOutData               349
  rxSMSGStdErrData                 0  txSMSGStdErrData                 0
  rxCMSGEOF                        0  txCMSGEOF                        0
  rxSMSGExitStatus                 0  txSMSGExitStatus                 3
  rxCMSGExitConfirmation           3  txCMSGExitConfirmation           0
  rxUnSupportedMsg                 0
  rxUnknownMsg                     0

  encodeSKSuccess                  0  decodeSKSuccess                 11
  encodeSKFail                     0  decodeSKFail                     0
  getHostKeyFail                   0  getServerKeyFail                 0
  serverKeyReGenerated             0

  encodeRSAChallengeGood           3  decodeRSAChallengeGood           0
  encodeRSAChallengeFail           0  decodeRSAChallengeFail           0
  getUserKeyFail                   0

  encoEncodeConfigured            11  encoDecodeConfigured            11
  encoEncodeConfigureFail          0  encoDecodeConfigureFail          0
  encoEncodeDetached               0  encoDecodeDetached               0
  encoEncodeDead                   0  encoDecodeDead                   0
  encoEncodeStart                402  encoDecodeStart                369
  encoEncoded                    402  encoDecoded                    369
  encoEncodeFail                   0  encoDecodeFail                   0
  encoEncodeResetDone              0  encoDecodeResetDone              0
  encoEncodeResetFail              0  encoDecodeResetFail              0
  encoEncodeDiscard                0  encoDecodeDiscard                0
```

Table 43-2: Parameters in the output of the **show ssh counter** command

Parameter	Meaning
inOctets	The number of octets received by the router.
outOctets	The number of octets transmitted by the router.
rxPkt	The number of packets received by the router.
txPkt	The number of packets transmitted by the router.
rxPktCheckFail	The number of packets received by the router that contained an invalid checksum.
txPktFail	The number of packets not transmitted by the router due to an error.
rxVersionID	The number of version identification messages received by the router.
txVersionID	The number of version identification messages transmitted by the router.
rxMSGDisconnect	The number of SSH_MSG_DISCONNECT messages received by the router, to terminate a session.
txMSGDisconnect	The number of SSH_MSG_DISCONNECT messages transmitted by the router, to terminate a session.
rxSMSGPublicKey	The number of SSH_SMSG_PUBLIC_KEY messages received by the client, containing the remote server's host key, server public key, supported ciphers and supported authentication methods.
txSMSGPublicKey	The number of SSH_SMSG_PUBLIC_KEY messages transmitted by the server, containing the server's host key, server public key, supported ciphers and supported authentication methods.
rxCMSGSessionKey	The number of SSH_CMSG_SESSION_KEY messages received by the server, containing the remote client's selected cipher, a copy of the 64-bit cookie sent by the server, the client's protocol flags and a session key encrypted with the server's host key and server key.
txCMSGSessionKey	The number of SSH_CMSG_SESSION_KEY messages transmitted by the client, containing the client's selected cipher, a copy of the 64-bit cookie sent by the remote server, the client's protocol flags and a session key encrypted with the remote server's host key and server key.
rxCMSGUser	The number of SSH_CMSG_USER messages received by the server, containing a user name to login.
txCMSGUser	The number of SSH_CMSG_USER messages transmitted by the client containing a user name to login.
rxCMSGAuthRhosts	The number of SSH_CMSG_AUTH_RHOSTS messages received by the server, containing the remote client's user name for .rhosts authentication.
txCMSGAuthRhosts	The number of SSH_CMSG_AUTH_RHOSTS messages tranmitted by the client, containing the client's user name for .rhosts authentication.
rxCMSGAuthRSA	The number of SSH_CMSG_AUTH_RSA messages received by the server, containing a remote client's public key for RSA authentication.

Table 43-2: Parameters in the output of the **show ssh counter** command (continued)

Parameter	Meaning
txCMSGAuthRSA	The number of SSH_CMSG_AUTH_RSA messages transmitted by the client, containing the public key for RSA authentication.
rxSMSGAuthRSAChallenge	The number of SSH_SMSG_AUTH_RSA_CHALLENGE messages received by the client, containing an encrypted challenge from the remote server.
txSMSGAuthRSAChallenge	The number of SSH_SMSG_AUTH_RSA_CHALLENGE messages transmitted by the server, containing an encrypted challenge for the remote client.
rxCMSGAuthRSAResponse	The number of SSH_CMSG_AUTH_RSA_RESPONSE messages received by the server from a remote client, containing the response to a challenge.
txCMSGAuthRSAResponse	The number of SSH_CMSG_AUTH_RSA_RESPONSE messages transmitted by the client to a remote server, containing the response to a challenge.
rxCMSGAuthPassword	The number of SSH_CMSG_AUTH_PASSWORD messages received by the server, containing the remote client's plaintext password.
txCMSGAuthPassword	The number of SSH_CMSG_AUTH_PASSWORD messages transmitted by the client, containing the client's plaintext password.
rxCMSGAuthRhostsRSA	The number of SSH_CMSG_AUTH_RHOSTS_RSA messages received by the server, containing the remote client's user name and public host key for .rhosts and RSA authentication.
txCMSGAuthRhostsRSA	The number of SSH_CMSG_AUTH_RHOSTS_RSA messages transmitted by the client, containing the client's user name and public host key for .rhosts and RSA authentication.
rxSMSGSuccess	The number of SSH_SMSG_SUCCESS messages received by the client indicating a successful request.
txSMSGSuccess	The number of SSH_SMSG_SUCCESS messages transmitted by the server indicating a successful request.
rxSMSGFailure	The number of SSH_SMSG_FAILURE messages received by the client indicating a failed request.
txSMSGFailure	The number of SSH_SMSG_FAILURE messages transmitted by the server indicating a failed request.
rxCMSGReqCompression	The number of SSH_CMSG_REQUEST_COMPRESSION messages received by the server, requesting compression for the connection.
txCMSGReqCompression	The number of SSH_CMSG_REQUEST_COMPRESSION messages transmitted by the client, requesting compression for the connection.
rxCMSGReqX11Forwarding	The number of SSH_CMSG_X11_REQUEST_FORWARDING messages received by the server, requesting forwarding of X11 connections.
txCMSGReqX11Forwarding	The number of SSH_CMSG_X11_REQUEST_FORWARDING messages transmitted by the client, requesting forwarding of X11 connections.

Table 43-2: Parameters in the output of the **show ssh counter** command (continued)

Parameter	Meaning
rxCMSGReqPortForwarding	The number of SSH_CMSG_PORT_FORWARD_REQUEST messages received by the server, requesting forwarding of a TCP/IP port.
txCMSGReqPortForwarding	The number of SSH_CMSG_PORT_FORWARD_REQUEST messages transmitted by the client, requesting forwarding of a TCP/IP port.
rxCMSGReqAgentForwarding	The number of SSH_CMSG_AGENT_REQUEST_FORWARDING messages received by the server, requesting forwarding of the connection to the authentication agent.
txCMSGReqAgentForwarding	The number of SSH_CMSG_AGENT_REQUEST_FORWARDING messages transmitted by the client, requesting forwarding of the connection to the authentication agent.
rxCMSGReqPty	The number of SSH_CMSG_REQUEST_PTY messages received by the server, requesting a pseudo terminal device be allocated for this session.
txCMSGReqPty	The number of SSH_CMSG_REQUEST_PTY messages transmitted by the client, requesting a pseudo terminal device be allocated for this session.
rxCMSGWindowSize	The number of SSH_CMSG_WINDOW_SIZE messages transmitted by the client, specifying a new size for the client's window.
txCMSGWindowSize	The number of SSH_CMSG_WINDOW_SIZE messages received by the server, specifying a new size for the remote client's window.
rxCMSGExecShell	The number of SSH_CMSG_EXEC_SHELL messages received by the server, to create an interactive terminal session.
txCMSGExecShell	The number of SSH_CMSG_EXEC_SHELL messages transmitted by the client, to create an interactive terminal session.
rxCMSGExecCmd	The number of SSH_CMSG_EXEC_CMD messages received by the server, containing a command to execute.
txCMSGExecCmd	The number of SSH_CMSG_EXEC_CMD messages transmitted by the client, containing a command to execute.
rxCMSGStdInData	The number of SSH_CMSG_STDIN_DATA messages received by the client, containing data written to stdout by applications on the remote server.
txCMSGStdInData	The number of SSH_SMSG_STDOUT_DATA messages transmitted by the client, containing data from the client to be written to stdin on the remote server.
rxSMSGStdOutData	The number of SSH_SMSG_STDOUT_DATA messages received by the server, containing data from the remote client to be written to stdin on the server.
txSMSGStdOutData	The number of SSH_SMSG_STDOUT_DATA messages transmitted by the server, containing data written to stdout by applications on the server.

Table 43-2: Parameters in the output of the **show ssh counter** command (continued)

Parameter	Meaning
rxSMSGStdErrData	The number of SSH_SMSG_STDERR_DATA messages received by the client, containing data written to stderr by applications on the remote server.
txSMSGStdErrData	The number of SSH_SMSG_STDERR_DATA messages transmitted by the server, containing data written to stderr by applications on the server.
rxCMSGEOF	The number of SSH_CMSG_EOF messages received by the server from the remote client, indicating end of input from the remote client
txCMSGEOF	The number of SSH_CMSG_EOF messages transmitted by the client to the remote server, indicating end of input from the client
rxSMSGExitStatus	The number of SSH_SMSG_EXITSTATUS messages received by the client, indicating that the shell or command has exited.
txSMSGExitStatus	The number of SSH_SMSG_EXITSTATUS messages transmitted by the server, indicating that the shell or command has exited.
rxCMSGExitConfirmation	The number of SSH_CMSG_EXIT_CONFIRMATION messages received by the server in response to a SSH_SMSG_EXITSTATUS message.
txCMSGExitConfirmation	The number of SSH_CMSG_EXIT_CONFIRMATION messages transmitted by the client in response to a SSH_SMSG_EXITSTATUS message.
rxUnSupportedMsg	The number of requests for unsupported options received by the server.
rxUnknownMsg	The number of requests for unrecognised options received by the server.
encodeSKSuccess	The number of double RSA encryptions of a session key.
decodeSKSuccess	The number of double RSA decryptions of a session key.
encodeSKFail	The number of failed attempts to encode a session key.
decodeSKFail	The number of failed attempts to decode a session key.
getHostKeyFail	The number of failed attempts to get the host key from the ENCO module.
getServerKeyFail	The number of failed attempts to get the server key from the ENCO module.
serverKeyReGenerated	The number of times a new random server key was created.
encodeRSAChallengeGood	The number of times the server encrypted a random challenge for RSA authentication.
decodeRSAChallengeGood	The number of times the server decrypted a random challenge for RSA authentication.
encodeRSAChallengeFail	The number of times the server failed to encrypt an RSA challenge.
decodeRSAChallengeFail	The number of times the server failed to decrypt an RSA challenge.
getUserKeyFail	The number of times the server failed to get the user RSA key from the ENCO module.

Table 43-2: Parameters in the output of the **show ssh counter** command (continued)

Parameter	Meaning
encoEncodeConfigured	The number of times an encryption channel was configured for a session.
encoDecodeConfigured	The number of times an decryption channel was configured for a session.
encoEncodeConfigureFail	The number of times the server failed to configure an encryption channel.
encoDecodeConfigureFail	The number of times the server failed to configure a decryption channel.
encoEncodeDetached	The number of times the encryption channel of a session was destroyed.
encoDecodeDetached	The number of times the decryption channel of a session was destroyed.
encoEncodeDead	The number of times the encryption engine has failed.
encoDecodeDead	The number of times the decryption engine has failed.
encoEncodeStart	The number of times an encode job was started on an ENCO channel.
encoDecodeStart	The number of times a decode job was started on an ENCO channel.
encoEncoded	The number of times an encode job completed.
encoDecoded	The number of times a decode job completed.
encoEncodeFail	The number of times an encode job failed to complete.
encoDecodeFail	The number of times a decode job failed to complete.
encoEncodeResetDone	The number of times an encryption channel has been reset.
encoDecodeResetDone	The number of times a decryption channel has been reset.
encoEncodeResetFail	The number of times the server failed to reset an encryption channel.
encoDecodeResetFail	The number of times the server failed to reset an decryption channel.
encoEncodeDiscard	The number of times an encode job has been discarded by the ENCO module.
encoDecodeDiscard	The number of times a decode job has been discarded by the ENCO module.

Example To display the configuration of the Secure Shell server, use the command:

```
sh ssh
```

To display the Secure Shell counters, use the command:

```
sh ssh cou
```

Related Commands

enable ssh server
disable ssh server
set ssh server
show ssh sessions

show ssh sessions

Syntax `SHow SSH SESSions`

Description This command displays the status of all Secure Shell sessions currently active on the router, including both outbound sessions to another host and inbound sessions into the router.

Figure 43-3: Example output from the **show ssh sessions** command

```
ID  Type    Dir   Peer Address     User             State    Octets In/Out
-------------------------------------------------------------------------
01  Listen  In    0.0.0.0                           Initial
02  Listen  In    ::                                Initial
03  Shell   In    192.168.2.5      MANAGER          Open     00162435/01524172
04  Shell   Out   192.168.100.264  JOHN             Open     00000016/00017254
05  Cmd     In    10.5.3.66        MANAGER          Open     00000182/00006243
06  -       In    172.17.99.4      -                Authen   00000088/00000002
07  -       Out   192.168.1.1      ROUTERB          Request  00000018/00000012
```

Table 43-3: Parameters in the output of the **show ssh sessions** command

Parameter	Meaning
ID	A unique identifier for each Secure Shell session.
Type	The type of Secure Shell connection; one of Listen (listen session for incoming connections), Shell (interactive shell connection) or Cmd (non-interactive remote command execution).
Dir	The direction of the connection; either In (connection made from a client to the Secure Shell server) or Out (connection made from the router to a remote Secure Shell server).
Peer Address	The IP address of the remote end of the Secure Shell connection.
User	The username under which the connection is made, or "-" if user authentication has not yet taken place. For inbound sessions, the specified username identifies a user account on the router. For outbound sessions, it identifies a user on the remote host.
State	The current state of the Secure Shell connection: Initial - connection being initiated Starting - host-to-host authentication underway Authen - user authentication in progress Request - request of session type in progress Open - session in progress
Octets In/Out	The number of octets sent and received on the connection.

Example To display currently active Secure Shell sessions, use the command:

```
sh ssh sess
```

Related Commands

enable ssh server
disable ssh server
set ssh server
show ssh

show ssh user

Syntax `SHow SSH USER[=`*`username`*`]`

where *username* is a character string 1 to 15 characters long. Valid characters are uppercase and lowercase letters, and digits (0–9). The string may not contain spaces.

Description This command displays information about the users allowed to make connections to the Secure Shell server.

If a username is not specified, summary information about all users is displayed (Figure 43-4, Table 43-2 on page 43-18).

If a username is specified, detailed information about the specified user is displayed (Figure 43-5 on page 43-24, Table 43-5 on page 43-25).

Figure 43-4: Example output from the **show ssh user** command

```
Secure Shell User List

User     IpAddr                          Authentication   KeyId   Status
-------------------------------------------------------------------------
test4    fe80:230:84ff:fe0e:263e         Password           0     enabled
test2    fe80:230:84ff:fe0e:263d         Password           0     enabled
secoff   0.0.0.0                         RSA                5     enabled
800      0.0.0.0                         RSA                4     enabled
admin    0.0.0.0                         RSA                7     disabled
john     192.168.2.1                     Password           0     enabled
```

Table 43-4: Parameters in the output of the **show ssh user** command

Parameter	Meaning
User	The username for the user.
IpAddr	The IP address from which the user may login.
Authentication	The type of authentication required for the user; either Password or RSA.
KeyId	The key ID of the ENCO key used for RSA authentication.
Status	Whether the status of the user is enabled or disabled.

Figure 43-5: Example output from the **show ssh user** command for a specific user

```
User..................... john
Status................... Enabled
Authorisation method..... Password
RSA key ID............... 0
Shell.................... Yes
IpAddress................ 192.168.2.1
Mask..................... 255.255.255.255
Failed Logins............ 0
```

Table 43-5: Parameters in the output of the **show ssh user** command for a specific user

Parameter	Meaning
User	The username for the user.
Status	Whether the status of the user is enabled or disabled.
Authorisation method	The type of authentication required for the user; either Password or RSA.
RSA key ID	The key ID of the ENCO key used for RSA authentication.
Shell	Whether the user has Shell access.
IpAddress	The IP address from which the user may login.
Mask	The network mask for the IP address.
Failed Logins	The total number of failed login attempts for the user.

Example To display the Secure Shell user list, use the command:

```
sh ssh user
```

To display specific information about the Secure Shell user named "john", use the command:

```
sh ssh user=john
```

Related Commands

add ssh user
delete ssh user
disable ssh user
enable ssh user
set ssh user

ssh

Syntax

```
SSH {ipadd|ipv6add[%interface]|host} USER=username
   {PASSword=password|KEYid=key-id} [COMmand=string]
```

where:

- *ipadd* is an IP address in dotted decimal notation.
- *ipv6add* is a valid IPv6 address.
- *interface* is the interface the SSH request is sent out, for a request to SSH to an IPv6 link-local address e.g. eth0.
- *host* is a full-domain name of a host, a host nickname created with the **add ip host** command on page 14-76 of Chapter 14, Internet Protocol (IP), or a host name in the same domain.
- *username* is a character string 1 to 15 characters long. Valid characters are uppercase and lowercase letters, and digits (0–9). The string may not contain spaces.
- *password* is a character string 1 to 31 characters long. Valid characters are any printable character. If the string contains spaces, put double quotes around it.

- *key-id* is a decimal key ID.
- *string* is a character string 1 to 80 characters long. Valid characters are any printable characters. If the string contains spaces, put double quotes around it.

Description This command initiates a Secure Shell connection to the specified host.

A Secure Shell request to an IPv6 link-local address requires interface information as well as the address, because a single link-local address can belong to several interfaces. To SSH to a link-local address, specify the interface out which the SSH request is sent, as well as the address. This interface is the interface, on the router from which the SSH request originates, that is connected to the required destination interface, Figure 15-2 on page 15-19 in Chapter 15, Internet Protocol Version 6 (IPv6). For example:

```
ssh fe80:7c27%vlan1 user=Admin password-18Again
```

The USER parameter specifies the name of a user on the remote host.

The PASSWORD parameter specifies the password for Secure Shell password authentication.

The KEY parameter specifies the RSA key that is to be used to authenticate the user. This parameter enables Secure Shell RSA authentication. The specified key must exist on both the client (RSA Private Key) and the server (RSA Public Key).

The COMMAND parameter specifies a command to be executed on the remote host or router. If this parameter is present, the command is executed on the remote system and the connection is closed. If the COMMAND parameter is not specified, an interactive terminal session is created.

Examples To connect to the remote host with IP address 172.16.1.5 as user "Admin" with password "l8Again", use the command:

```
ssh 172.16.1.5 user=Admin pass=l8Again
```

Related Commands **show ssh sessions**