

SSHD Configuration Mode Commands

The Secure Shell Configuration Mode is used to manage the SSH server options for the current context.

Important

- Use the **ssh generate key** command in Context Configuration Mode to generate the sshd keys before you can configure the sshd server.
- The global limit for utility service connections, including SSH, Telnet, and SFTP, is 201. This includes any inter-card communication that uses SSH.

Command Modes

Exec > Global Configuration > Context Configuration > SSH Configuration

configure > context *context_name* **> server sshd**

Entering the above command sequence results in the following prompt:

```
[local]host_name(config-sshd)#
```

Important The commands or keywords/variables that are available are dependent on platform type, product version, and installed license(s).

- allowusers add, on page 2
- authorized-key, on page 3
- challenge-response-authentication, on page 4
- ciphers, on page 5
- client-alive-countmax, on page 7
- client-alive-interval, on page 8
- do show, on page 9
- end, on page 9
- exit, on page 10
- listen, on page 10
- macs, on page 11
- max servers, on page 12
- subsystem, on page 13

allowusers add

Specifies and controls which users can access SSH services.

Product

All

Privilege

Security Administrator

Command Modes

Exec > Global Configuration > Context Configuration > SSH Configuration

configure > context *context_name* **> server sshd**

Entering the above command sequence results in the following prompt:

`[local]host_name(config-sshd)#`

Syntax Description

`[ default | no ] allowusers add` *`user_list`*

default

Unrestricted access for all users.

no

Removes the list of user name patterns resulting in unresticted access by all users.

user_list

Specifies a list of user name patterns, separated by spaces, as an alphanumeric string of 1 through 999 characters. If the pattern takes the form 'USER' then login is restricted for that user. If the pattern is in the format 'USER@IP_ADDRESS' then USER and IP address are separately checked, restricting logins to those users from that particular iIP address.

The following limits apply to the *user_string*:

- The maximum length of this string is 3000 bytes including spaces.
- The maximum number of allowusers, which is counted by spaces, is 256, which is consistent with the limit from OpenSSH.

Important If you exceed either of the above limits, an error message is displayed. The message prompts you to use a regular expression pattern to shorten the string, or remove all the allowusers with **no allowusers add** or **default allowusers add** and re-configure.

Important For more details about how to create complex rules, see the OpenSSH sshd_config man page. add - Add more users to the list of user name patterns.

Usage Guidelines

Use this command to specify and control which users can access SSH services.

Access to a service may be restricted to users having a legitimate need. This restriction applies on a white-list basis: only explicitly allowed users shall connect to a host via SSH and possibly from a specified source IP addresses. Under OpenSSH, the AllowUsers directive of sshd_config specifies a list of SSH authorized users and groups.

Example

The following command specifies an AllowUsers list of four users:

```
allowusers add user1 user2@10.1.1.1 user3@10.1.1.2 user4
```

authorized-key

Sets or removes a user name having authorized keys for access to the sshd server in the current context.

Product

All

Privilege

Security Administrator, Administrator

Command Modes

Exec > Global Configuration > Context Configuration > SSH Configuration

configure > context *context_name* **> server sshd**

Entering the above command sequence results in the following prompt:

```
[local]host_name(config-sshd)#
```

Syntax Description

authorized-key username *user_name* **host** *host_ip* **[type { v2-dsa | v2-rsa }]**

default

Resets the parameter to the default value.

username *user_name*

Sets a username as having authorized keys for access to the sshd server. Specifies the username as an alphanumeric string of 1 through 255 characters.

host *host_ip*

Associates an SSH host having the authorization keys for the username as an host IP address in IPv4 dotted decimal or IPv6 colon-separated-hexadecimal notation.

[type { v2-dsa | v2-rsa }]

Specifies which type of SSH authorization key will be accepted instead of all key types. The options are: **v2-dsa** (SSHv2 Digital Signature Algorithm), or **v2-rsa** (SSHv2 Rivest, Shamir and Adleman).

Usage Guidelines

Use this command to set a username with authorized keys for access to the sshd server within the current context.

Usernames should be created using the **nopassword** option to prevent bypassing of the sshd keys (**administrator** command in Context Configuration mode).



Important Only 10 sshd authorization-keys can be configured per context.

Example

The following command specifies that username *dbailey* with authorization keys at host IP address *209.165.200.225* can access the system with all types of authorization keys:

```
authorized-key username dbailey host 209.165.200.225
```

challenge-response-authentication

The challenge-response-authentication option under SSHD configuration is used to enable the Keyboard Interactive Authentication method.

Product

All

Privilege

Security Administrator, Administrator

Command Modes

Exec > Global Configuration > Context Configuration > SSH Configuration

configure > **context** *context_name* > **server sshd**

Entering the above command sequence results in the following prompt:

```
[local]host_name(config-sshd)#
```

Syntax Description

[no] challenge-response-authentication

no

Disables the challenge-response-authentication parameter.

challenge-response-authentication

Enables the challenge-response-authentication in SSHD server configuration.

Note

1. The challenge-response-authentication option is only supported from release 21.28.
2. Enabling challenge-response-authentication is only advised in certain special cases. For example, when the TACACS server chooses to display a specific prompt to the user. Unless there is a very specific reason, challenge-response-authentication must not be enabled. Contact your Cisco representative before enabling this option.
3. Even though it is not explicity restricted, customers are strongly advised not to enable challenge-response-authentication for any products other than legacy PGW, SGW, and SAEGW.
4. To use Keyboard Interactive Authentication method, challenge-response-authentication must be enabled under the context which owns the IP address that is used for SSH login.
5. The challenge-response-authentication has no effect on SSH logins through the console.
6. The challenge-response-authentication is an SSHD option, and it doesn't affect logins for telnet or FTP.
7. The user responses must be of size less than 128 bytes.

Important We envisage challenge-response-authentication being used in conjunction with TAC special cases. The TACACS server can send up to 511 characters in AUTHEN-RE and those characters shall be passed to the end user who is trying to login. If the len field is 512 bytes or above, following error message is shown to the user and TAC as expected: ERROR: Enter any key to proceed.

8. When challenge-response-authentication is enabled, the user has 60 seconds to respond to the prompt.

ciphers

Configures the cipher priority list in sshd for SSH symmetric encryption. It changes the cipher option for that context.

Product

All

Privilege

Security Administrator, Administrator

Command Modes

Exec > Global Configuration > Context Configuration > SSH Configuration

configure > context *context_name* **> server sshd**

Entering the above command sequence results in the following prompt:

```
[local]host_name(config-sshd)#
```

Syntax Description

[default] ciphers *algorithm*

default

Release 20.x to 21.15 (Normal build only)

Resets the value of *algorithm* in a Normal build to:

```
blowfish-cbc,3des-cbc,aes128-cbc,aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com,chacha20-poly1305@openssh.com
```

Resets the value of *algorithm* in a Trusted build as follows:

```
aes256-ctr,aes192-ctr,aes128-ctr
```

Release 21.16 onwards: Post OpenSSH to CiscoSSH Upgrade and Migration

Default Algorithms in a Normal Build:

```
aes256-ctr,aes192-ctr,aes128-ctr,aes256-gcm@openssh.com,aes128-gcm@openssh.com,chacha20-poly1305@openssh.com
```

Available Algorithms in a Normal Build:

```
aes256-ctr,aes192-ctr,aes128-ctr,aes256-gcm@openssh.com,aes128-gcm@openssh.com,chacha20-poly1305@openssh.com,aes128-cbc
```

Default and Available Algorithms in Trusted Builds:

```
aes256-ctr,aes192-ctr,aes128-ctr
```

Note There is no change in the default and configurable Ciphers for Trusted builds.

algorithm

Specifies the algorithm to be used as a single string of comma-separated variables (no spaces) in priority order from those shown below:

- **blowfish-cbc** – symmetric-key block cipher, Cipher Block Chaining, CBC

Note This algorithm is removed post the OpenSSH to CiscoSSH upgrade and migration.

- **3des-cbc** – Triple Data Encryption Standard, CBC
- **aes128-cbc** – Advanced Encryption Standard, 128-bit key size, CBC
- **aes128-ctr** –Advanced Encryption Standard, 128-bit key size, Counter-mode encryption, CTR
- **aes192-ctr** – Advanced Encryption Standard, 192-bit key size, CTR
- **aes256-ctr** – Advanced Encryption Standard, 256-bit key size, CTR
- **aes128-gcm@openssh.com** – Advanced Encryption Standard, 128-bit key size, Galois Counter Mode [GCM], OpenSSH
- **aes256-gcm@openssh.com** – Advanced Encryption Standard, 256-bit key size, GCM, OpenSSH
- **chacha20-poly1305@openssh.com** – ChaCha20 symmetric cipher, Poly1305 cryptographic Message Authentication Code [MAC], OpenSSH

algorithm is a string of 1 through 511 alphanumeric characters.



Important For release 20.0 and higher Trusted builds, only the AES128-CTR, AES-192-CTR and AES-256CTR ciphers are available.

Usage Guidelines Use this command to configure the cipher priority list in sshd for SSH symmetric encryption.

Example

The following command sets the supported SSH algorithms and their priority.

```
ciphers blowfish-cbc,aes128-cbc,aes128-ctr,aes192-ctr,aes256-ctr
```

client-alive-countmax

Sets the number of client-alive messages which may be sent without sshd receiving any messages back from the SSH client. If this threshold is reached while the client-alive messages are being sent, sshd disconnects the SSH client thus terminating the session.

Product All

Privilege Security Administrator, Administrator

Command Modes Exec > Global Configuration > Context Configuration > SSH Configuration

configure > **context** *context_name* > **server sshd**

Entering the above command sequence results in the following prompt:

```
[local]host_name(config-sshd)#
```

Syntax Description **[default | no] client-alive-countmax** *count_number*

default

Sets the default value for this parameter to 3.



Important For higher security, Cisco recommends at least a client-alive-countmax of 2 and client-alive-interval of 5. Smaller session logout values may lead to occasional ssh session logouts. Adjust values to balance security and user friendliness.

no

Disables the client-alive-countmax parameter.

count_number

Specifies the number of times a client-alive message will be sent as an integer from 1 through 3. The messages are sent following the expiry of each client-alive interval. Default = 3

Unresponsive SSH clients will be disconnected when the maximum number of client-alive-intervals have expired.

Usage Guidelines

Use this command to set the number of client-alive messages which may be sent without sshd receiving any messages back from the SSH client. If this threshold is reached while client-alive messages are being sent, sshd will disconnect the SSH client, terminating the session. The client-alive messages are sent through the encrypted channel and, therefore, are not spoofable. The client-alive mechanism is valuable when the client or server depend on knowing when a connection has become inactive.



Important This parameter applies to SSH protocol version 2 only.

Example

The following command sets the SSH client-alive-countmax to 2.

```
client-alive-countmax 2
```

client-alive-interval

Sets a timeout interval in seconds after which if no data has been received from the SSH client, sshd sends a message through the encrypted channel to request a response from the client.

Product

All

Privilege

Security Administrator, Administrator

Command Modes

Exec > Global Configuration > Context Configuration > SSH Configuration

configure > context *context_name* **> server sshd**

Entering the above command sequence results in the following prompt:

```
[local]host_name(config-sshd)#
```

Syntax Description

`[ default | no ] client-alive-interval` *seconds*

default

Sets the client-alive-interval to 15 seconds.



Important For higher security, Cisco recommends at least a client-alive-interval of 5 and client-alive-countmax of 2. Smaller session logout values may lead to occasional ssh session logouts. Adjust values to balance security and user friendliness.

no

Disables the client-alive-interval parameter.

seconds

Specifies the amount of time in seconds that sshd waits to receive a response from the SSH client as an integer from 1 through 15. Default = 15

Usage Guidelines

Use this command to set a timeout interval in seconds after which if no data has been received from the client, sshd sends a message through the encrypted channel to request a response from the client. The number of times that the message is sent is determined by the client-alive-countmax parameter. The approximate amount of time before sshd disconnects an SSH client disconnect = client-alive-countmax X client-alive-interval.



Important This parameter applies to SSH protocol version 2 only.

Example

The following command sets the SSH client-alive-interval to 5 seconds.

```
client-alive-interval 5
```

do show

Executes all **show** commands while in Configuration mode.

Product All

Privprivilege Security Administrator, Administrator

Syntax Description

```
do show
```

Usage Guidelines

Use this command to run all Exec mode **show** commands while in Configuration mode. It is not necessary to exit the Config mode to run a **show** command.

The pipe character | is only available if the command is valid in the Exec mode.



Caution There are some Exec mode **show** commands which are too resource intensive to run from Config mode. These include: **do show support collection**, **do show support details**, **do show support record** and **do show support summary**. If there is a restriction on a specific **show** command, the following error message is displayed:

```
Failure: Cannot execute 'do show support' command from Config mode.
```

end

Exits the current configuration mode and returns to the Exec mode.

Product All

Privilege Security Administrator, Administrator

Syntax Description **end**

Usage Guidelines Use this command to return to the Exec mode.

exit

Exits the current mode and returns to the parent configuration mode.

Product All

Privilege Security Administrator, Administrator

Syntax Description **exit**

Usage Guidelines Use this command to return to the parent configuration mode.

listen

Configures the SSH server in the current context to only listen for connections from the interface with the specified IP address. The default behavior is to listen on all interfaces.

Product All

Privilege Security Administrator, Administrator

Command Modes Exec > Global Configuration > Context Configuration > SSH Configuration

configure > context *context_name* **> server sshd**

Entering the above command sequence results in the following prompt:

```
[local]host_name(config-sshd)#
```

Syntax Description **listen** *ip_address*
no listen

no

Disable listening for a specific interface address and enable listening on all interfaces.

ip_address

Enables listening only on the interface with the specified IP address. *ip_address* must be entered using IPv4 dotted-decimal notation.

Usage Guidelines Use this command to configure the SSH server for the current context to only listen for connections from the interface with the specified IP address. Only one IP address may be set for listening.

Example

The following command specifies that the Server should only listen for connections in the interface with the IP address of *192.168.0.10*:

```
listen 192.168.0.10
```

macs

Configures the MAC algorithm priority list in sshd for SSH symmetric encryption. It changes the MAC algorithm for that context.

Product

All

Privilege

Security Administrator, Administrator

Command Modes

Exec > Global Configuration > Context Configuration > SSH Configuration

configure > context *context_name* **> server sshd**

Entering the above command sequence results in the following prompt:

```
[local]host_name(config-sshd)#
```

Syntax Description

macs*algorithm*

default macs

default

Release 20.x to 21.15

Resets the value of *algorithm* in a Normal build and Trusted build to:

```
hmac-sha2-512-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha1-etm@openssh.com,hmac-sha2-512,
hmac-sha2-256,hmac-sha1
```

Available algorithms in a Normal build are:

```
hmac-sha2-512-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha1-etm@openssh.com,hmac-sha2-512,
hmac-sha2-256,hmac-sha1,umac-128-etm@openssh.com,umac-128@openssh.com,umac-64-etm@openssh.com,umac-64@openssh.com
```

Available algorithms in a Trusted build are:

```
hmac-sha2-512-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha1-etm@openssh.com,hmac-sha2-512,
hmac-sha2-256,hmac-sha1
```

Release 21.16 onwards: Post OpenSSH to CiscoSSH Upgrade and Migration

Default and Available Algorithms in Normal Builds:

```
hmac-sha2-512-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha1-etm@openssh.com,hmac-sha2-512,
hmac-sha2-256,hmac-sha1
```

Default Algorithms in Trusted Builds:

```
hmac-sha2-512,hmac-sha2-256,hmac-sha1
```

Available Algorithms in Trusted Builds:

```
hmac-sha2-512,hmac-sha2-256,hmac-sha1
```



Note hmac-sha2-512-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha1-etm@openssh.com are removed from the Trusted builds.

algorithm

- Specifies the algorithms to be used as a single string of comma-separated variables (no spaces) in priority order (left to right) from those listed as follows:
  - HMAC = hash-based message authentication code
  - SHA2 = Secure Hash Algorithm 2
  - SHA1 = Secure Hash Algorithm 1
  - ETM = Encrypt-Then-MAC
  - UMAC = message authentication code based on universal hashing

algorithm is a string of 1 through 511 alphanumeric characters.

Usage Guidelines Use this command to configure the priority of MAC algorithms in sshd for SSH symmetric encryption.

Example

The following command sets the supported MAC algorithms and their priority.

```
MACs
hmac-sha2-512-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha1-etm@openssh.com,hmac-sha2-512,hmac-sha2-256,hmac-sha1
```

max servers

Configures the maximum number of SSH servers that can be started within any 60-second interval. If this limit is reached, the system waits two minutes before trying to start any more servers.

Product All

Privilege Security Administrator, Administrator

Command Modes Exec > Global Configuration > Context Configuration > SSH Configuration

configure > context *context_name* **> server sshd**

Entering the above command sequence results in the following prompt:

```
[local]host_name(config-sshd)#
```

Syntax Description **max servers** *number*

number

Default: 40

Specifies the maximum number of servers that can be spawned in any 60-second interval. *number* must be an integer from 1 through 100.

In 16.0 and later releases, this range is increased to 1-4000 to support the Stranded CDR feature. For more information on this feature, see the "**gtpp push-to-active url**" CLI command in the Global Configuration mode.

Usage Guidelines

Set the number of servers to tune the system response as a heavily loaded system may need more servers to support the incoming requests.

The converse would be true as well in that a system can benefit by reducing the number of servers such that telnet services do not cause excessive system impact to other services.

Example

```
max servers 50
```

subsystem

Configures the system to perform file transfers using Secure FTP (SFTP) over ssh v2. Administrators must be configured with the FTP attribute privilege to issue this command. This command also supports creation of SFTP subsystem root directories with access privileges. Administrators can assign an SFTP subsystem to local users.

Product

All

Privilege

Security Administrator, Administrator

Command Modes

Exec > Global Configuration > Context Configuration > SSH Configuration

configure > context *context_name* **> server sshd**

Entering the above command sequence results in the following prompt:

```
[local]host_name(config-sshd)#
```

Syntax Description

```
subsystem { cli | sftp [ name sftp_name root-dir pathname mode { read-only |
 readwrite } ] }
no
subsystem { cli | sftp }
no subsystem sftp name stfp_name
```

no

Disables the SFTP ssh file transfer method or access to the CLI via ssh or a specified SFTP subsystem.

Important An SFTP subsystem can only be removed if the subsystem is not currently assigned to any local user.

cli

Default: Enabled

Configures the SSH system for the current context to allow access to the CLI.

sftp

Default: Disabled

Enables the SSH system for the current context to perform file transfers using Secure FTP (SFTP) over ssh v2.

name *sftp_name*

Assigns a name for this SFTP subsystem. *sftp_name* is an alphanumeric string that uniquely identifies this subsystem.

root-dir *pathname*

Specifies the root directory to which SFTP files can be transferred. Options include:

- /hd-raid/records/cdr
- /flash

mode { read-only | readwrite }

Specifies the SFTP transfer mode. Options include:

- read-only
- read-write

Usage Guidelines

Use this command to enable or disable file transfers using SFTP over an ssh v2 tunnel.

You can also create multiple SFTP subsystems with an associated pathname and access privilege (read-only or read-write). When creating a local user, an administrator can assign the user an SFTP subsystem. If the user is not an administrator, he or she will only be able to access the subsystem with read-only privilege. The SFTP subsystem directory becomes the SFTP user's root directory with associated access privileges.

Also use this command to enable or disable access to the CLI over an SSH connection.

Example

The following command enables SFTP for the current context:

```
subsystem sftp
```

The following command disables access to the CLI through an SSH session for the current context:

```
no subsystem cli
```

The following command creates an SFTP subsystem for CDR records with read-write privileges:

```
subsystem sftp name cdr-rw-server root-dir /hd-raid/records/cdr mode
readwrite
```