



# SSH

From Gentoo Wiki

**SSH** (**Secure SHell**) is an encrypted terminal program that replaces the classic telnet (<http://en.wikipedia.org/wiki/telnet>) tool on Unix-like operating systems.

In addition to remote terminal access provided by the main **ssh** binary, the SSH suite of programs has grown to include other tools such as **scp** (**Secure Copy Program**) and **sftp** (**Secure File Transfer Protocol**).

Originally, SSH was not free. However, today the most popular and de-facto standard implementation of SSH is OpenBSD (<https://www.openbsd.org/>)'s OpenSSH, which comes pre-installed on Gentoo.

## Contents

- 1 Installation
  - 1.1 Check install
  - 1.2 USE flags
  - 1.3 Emerge
- 2 Configuration
  - 2.1 Create keys
  - 2.2 Server configuration
  - 2.3 Client configuration
  - 2.4 Passwordless authentication
    - 2.4.1 Client
    - 2.4.2 Server
    - 2.4.3 Single machine testing
  - 2.5 Intrusion prevention
- 3 Usage
  - 3.1 Services
    - 3.1.1 OpenRC

### Resources

 Home (<https://www.openssh.com/>)

 Wikipedia  
(<https://en.wikipedia.org/wiki/Secure%20Shell>)

 Open Hub  
(<https://www.openhub.net/p/openssh>)

- 3.1.2 systemd
- 3.2 Escape sequences
- 4 Troubleshooting
  - 4.1 Death of long-lived connections
  - 4.2 X11 forwarding, not forwarding, or tunneling
- 5 See also
- 6 External resources

# Installation


## Check install

Most deployments of Gentoo Linux will already have OpenSSH installed on the system. This can be checked by running the `ssh` command. If it is installed a usage statement should be printed:

```
user $ ssh
```

If no usage statement is printed `ssh` is either corrupted or not installed. It is also possible that a user is simply rebuilding OpenSSH to [ ] include a new USE configuration. Whatever the case, proceed on to view possible USE settings.

## USE flags

USE flags for net-misc/openssh (https://packages.gentoo.org/packages/net-misc/openssh)  Port of OpenBSD's free S...		
X (https://packages.gentoo.org/useflags/X)	Add support for X11	global
x509 (https://packages.gentoo.org/useflags/X509)	Adds support for X.509 certificate authentication	local
bindist (https://packages.gentoo.org/useflags/bindist)	Disable EC/RC5 algorithms in OpenSSL for patent reasons.	local
debug (https://packages.gentoo.org/useflags/debug)	Enable extra debug codepaths, like asserts and extra output. If you want to get meaningful backtraces see https://wiki.gentoo.org/wiki/Project:Quality_Assurance/Backtraces	global
hpn (https://packages.gentoo.org/useflags/hpn)	Enable high performance ssh	local

More information about USE flags  
(/wiki/Handbook:AMD64/Working/USE)

Data provided by the Gentoo Package Database (<https://packages.gentoo.org>) · Last update: 2018-02-01 02:28

## Emerge

After changing the necessary USE flags, do not forget to install (or rebuild) OpenSSH:

```
root # emerge --ask --changed-use net-misc/openssh
```

## Configuration

### Create keys

In order to provide a secure shell, cryptographic keys are used to manage the encryption, decryption, and hashing functionalities offered by SSH.

On the first start of the SSH service, system keys will be generated. Keys can be (re)generated using the **ssh-keygen** command.

To generate the key used for SSH protocol version 1 (which usually is not enabled anymore; it has been deprecated in favor of protocol version 2) use:

```
root # /usr/bin/ssh-keygen -t rsa1 -b 1024 -f /etc/ssh/ssh_host_key -N ""
```

To generate the keys for SSH protocol version 2 (DSA and RSA algorithms):

```
root # /usr/bin/ssh-keygen -t dsa -f /etc/ssh/ssh_host_dsa_key -N ""
```

```
root # /usr/bin/ssh-keygen -t rsa -f /etc/ssh/ssh_host_rsa_key -N ""
```

### Server configuration

The SSH server is usually configured in the `/etc/ssh/sshd_config` file, though it is also possible to perform further configuration in OpenRC's `/etc/conf.d/sshd`, including changing the location of the configuration file. For detailed information on how to configure the server see the *sshd\_config* man page (/wiki/Man\_page).

Users should study Sven's OpenSSH guide ([https://dev.gentoo.org/~swift/docs/security\\_benchmarks/openssh.html](https://dev.gentoo.org/~swift/docs/security_benchmarks/openssh.html)) for a security focused configuration.

### Client configuration

The **ssh** client and related programs (**scp**, **sftp**, etc.) can be configured using the following files:

- `~/.ssh/config`
- `/etc/ssh/ssh_config`

For more information read the `ssh_config` manual:

```
user $ man ssh_config
```

## Passwordless authentication

Handy for git (/wiki/Git) server management.

### Client

On the client, if not already done, create a key pair. This can be done by running the following command (of course, **not entering** a passphrase):

```
user $ ssh-keygen -t rsa
```

### Server

Make sure an account for the user exists on the server, and then place the clients' `id_rsa.pub` file into the server's `~/.ssh/authorized_keys` file in the user's home directory. This can be done by running the following command **on the client computer** (here, the user's passphrase on the server needs to be entered):

```
user $ ssh-copy-id <server>
```

Afterwards a passwordless login should be possible doing

```
user $ ssh <server>
```

```
larry@<server>
```

Then on the server, the file `/etc/ssh/sshd_config` should be set to `PasswordAuthentication no`.

### Single machine testing

The above procedure can be tested out locally:

```
user $ ssh-keygen -t rsa
```

```
user $ mv ~/.ssh/id_rsa.pub ~/.ssh/authorized_keys
```

```
user $ ssh localhost
```

## Intrusion prevention

SSH is a commonly attacked service. Tools such as `sshguard` (/wiki/Sshguard) and `fail2ban` (/wiki/Fail2ban) monitor logs and black list remote users who have repeatedly attempted, yet failed to login. Utilize them as needed to secure a frequently attacked system.

## Usage

### Services

#### OpenRC

Add the OpenSSH daemon to the default runlevel:

```
root # rc-update add sshd default
```

Start the sshd daemon with:

```
root # rc-service sshd start
```

The OpenSSH server can be controlled like any other OpenRC (/wiki/OpenRC)-managed service:

```
root # rc-service sshd start
```

```
root # rc-service sshd stop
```

```
root # rc-service sshd restart
```

#### Note

Active SSH connections to the server remain unaffected when issuing **rc-service sshd restart**.

## systemd

To have the OpenSSH daemon start when the system starts:

```
root # systemctl enable sshd.service
```

Created symlink from /etc/systemd/system/multi-user.target.wants/sshd.service to /usr/lib64/systemd/system/sshd.service.


To start the OpenSSH daemon now:

```
root # systemctl start sshd.service
```

To check if the service has started:

```
root # systemctl status sshd.service
```

## Escape sequences

During an active SSH session, pressing the tilde (  ) key starts an escape sequence. Enter the following for a list of options:

```
ssh> ~?
```

## Troubleshooting

There are 3 different levels of debug modes that can help troubleshooting issues. With the `-v` option SSH prints debugging messages about its progress. This is helpful in debugging connection, authentication, and configuration problems. Multiple `-v` options increase the verbosity. Maximum verbosity is three levels deep.

```
user $ ssh example.org -v
```

```
user $ ssh example.org -vv
```

```
user $ ssh example.org -vvv
```

## Death of long-lived connections

Many internet access devices perform Network Address Translation (NAT (</index.php?title=NAT&action=edit&redlink=1>)), a process that enables devices on a private network such as that typically found in a home or business place to access foreign networks, such as the internet, despite only having a single IP address on that network. Unfortunately, not all NAT devices are created equal, and some of them incorrectly close long-lived, occasional-use TCP connections such as those used by SSH. This is generally observable as a sudden inability to interact with the remote server, even though the **ssh** client program has not exited.

In order to resolve the issue, OpenSSH clients and servers can be configured to send a 'keep alive', or invisible message aimed at maintaining and confirming the live status of the link:

- To enable keep alive *for all clients connecting to your local server*, set `ClientAliveInterval 30` (or some other value, in seconds) within the `/etc/ssh/sshd_config` file.
- To enable keep alive *for all servers connected to by your local client*, set `ServerAliveInterval 30` (or some other value, in seconds) within the `/etc/ssh/ssh_config` file.

## X11 forwarding, not forwarding, or tunneling

**Problem:** After having made the necessary changes to the configuration files for permitting X11 forwarding, it is discovered X applications are executing on the server and are not being forwarded to the client.

**Solution:** What is likely occurring during SSH login into the remote server or host, the `DISPLAY` variable is either being unset or is being set *after* the SSH session sets it.

Test for this scenario perform the following after logging in remotely:

```
user $ echo $DISPLAY
```

```
localhost:10.0
```

The output should be something similar to `localhost:10.0` or `localhost2.local:10.0` using server side `X11UseLocalhost no` setting. If the usual `:0.0` is not displayed, check to make sure the `DISPLAY` variable within `~/.bash_profile` is not being unset or re-initializing. If it is, remove or comment out any custom initialization of the `DISPLAY` variable to prevent the code in `~/.bash_profile` from executing during a SSH login:

```
user $ ssh -t larry@localhost2 bash --noprofile
```

Be sure to substitute `larry` in the command above with the proper username.

A trick that works to complete this task would be to define an alias within the users' `~/.bashrc` file.

## See also

- [Securing the SSH service \(/wiki/Security\\_Handbook/Securing\\_services#ssh\)](/wiki/Security_Handbook/Securing_services#ssh) in the Gentoo Security Handbook

- Keychain (/wiki/Keychain) — This document describes how to use SSH shared keys along with the keychain program.
- autossh (/wiki/Autossh) — a command that detects when **SSH** connections drop and automatically reconnects them.
- SCP (/wiki/SCP) — an interactive file transfer program, similar to the **copy** command, that copies files over an encrypted SSH transport.
- SFTP (/wiki/SFTP) — an interactive file transfer program, similar to ftp, which performs all operations over an encrypted **SSH** transport.
- SSHFS (/wiki/SSHFS) — a secure shell client used to mount remote filesystems to local machines.
- Gentoo Handbook — Installation — Starting the SSH daemon  
(/wiki/Handbook:AMD64/Installation/Media#Optional:\_Starting\_the\_SSH\_daemon)
- Sakaki's\_EFI\_Install\_Guide/Setting\_Up\_Networking\_and\_Connecting\_via\_ssh#Connecting\_via\_ssh\_and\_Using\_screen  
(/wiki/Sakaki%27s\_EFI\_Install\_Guide/Setting\_Up\_Networking\_and\_Connecting\_via\_ssh#Connecting\_via\_ssh\_and\_Using\_screen)

## External resources

- Securing OpenSSH ([https://dev.gentoo.org/~swift/docs/security\\_benchmarks/openssh.html](https://dev.gentoo.org/~swift/docs/security_benchmarks/openssh.html)) - Gentoo developer documentation.
- net-misc/connect (<https://packages.gentoo.org/packages/net-misc/connect>) — SSH Proxy Command -- connect.c  
(<https://bitbucket.org/gotoh/connect/wiki/Home>)
- <https://lonesysadmin.net/2011/11/08/ssh-escape-sequences-aka-kill-dead-ssh-sessions/amp/> (<https://lonesysadmin.net/2011/11/08/ssh-escape-sequences-aka-kill-dead-ssh-sessions/amp/>) - A blog entry on escape sequences.
- <https://hackaday.com/2017/10/18/practical-public-key-cryptography/> (<https://hackaday.com/2017/10/18/practical-public-key-cryptography/>) - Practical public key cryptography (Hackaday).
- SSH on wiki.archlinux.org (<https://wiki.archlinux.org/index.php/SSH>)

Retrieved from "<http://wiki.gentoo.org/index.php?title=SSH&oldid=695206> (<http://wiki.gentoo.org/index.php?title=SSH&oldid=695206>)"

Categories (/wiki/Special:Categories): Documents containing Metadata (/wiki/Category:Documents\_containing\_Metadata)

| SSH (/wiki/Category:SSH) | Server (/wiki/Category:Server) | Daemons (/wiki/Category:Daemons) | Authentication (/wiki/Category:Authentication)

- This page was last modified on 20 December 2017, at 06:35.

© 2001–2018 Gentoo Foundation, Inc.

Gentoo is a trademark of the Gentoo Foundation, Inc. The contents of this document, unless otherwise expressly stated, are licensed under the CC-BY-SA-3.0 (<https://creativecommons.org/licenses/by-sa/3.0/>) license. The Gentoo Name and Logo Usage Guidelines (<https://www.gentoo.org/inside-gentoo/foundation/name-logo-guidelines.html>) apply.