SSH Hardening Guides

Below are guides to hardening SSH on various systems. Note that following them may not result in a perfect auditing score, as not all packaged SSH server versions support the required options. However, these instructions will result in the best possible score.

These guides were inspired by this document (which is now out-dated).

Server Guides:

- Amazon Linux 2023
- Debian 11 (Bullseye)
- Debian 12 (Bookworm)
- Rocky Linux 9
- Ubuntu 20.04 LTS
- Ubuntu 22.04 LTS
- Ubuntu 24.04 LTS

Client Guides:

- Amazon Linux 2023
- Debian 12 (Bookworm)
- Rocky Linux 9
- Ubuntu 20.04 LTS / Linux Mint 20
- Ubuntu 22.04 LTS / Linux Mint 21
- Ubuntu 24.04 LTS / Linux Mint 22

Community-Developed Guides:

The following guides have been written by the community. They have not been officially tested, and are not officially supported:

- ArubaOS Switch (AOS S) 16.11
- Dropbear 2022.83
- Fortinet FortiOS
- FreeBSD (various versions)

- Mac OS 13 (Ventura) & 14 (Sonoma)
- Mikrotik RouterOS
- OPNsense 20.7 and newer
- Proxmox-VE-7.3-6
- Synology DSM
- Void Linux

Instructions for submitting a hardening guide can be found here.

Legacy Guides:

- Debian Buster (Debian 10)
- OpenBSD 6.2
- pfSense 2.4
- RedHat Enterprise Linux 7 / CentOS 7
- RedHat Enterprise Linux 8 / CentOS 8
- Ubuntu 14.04 LTS
- Ubuntu 16.04 LTS (Server)
- Ubuntu 16.04 LTS / Linux Mint 18 (Client)
- Ubuntu 18.04 LTS
- Ubuntu 18.04 LTS / Linux Mint 19 (Client)
- Ubuntu Core 16
- Ubuntu Core 18

Ubuntu 24.04 LTS Server

Last modified: October 1, 2024

Change Log:

Copy commands to clipboard

October 1, 2024: Added RequiredRSASize directive to enforce a minimum of 3072-bit user and host-based authentication keys.

April 29, 2024: Initial revision. In comparison to Ubuntu 22.04 LTS guide, the following changes were made: 1.) For key exchanges, diffie-hellman-group18-sha512 and diffie-hellman-group-exchange-sha256 were prioritized over diffie-hellman-group16-sha512 due to greater security strength; GSS algorithms were prioritized over their non-GSS equivalents in order to match the client guide, 2.) For ciphers, 256-bit AES ciphers were prioritized over 192 and 128-bit AES ciphers due to their increased resistence against quantum computing attacks (previously, weaker GCM ciphers had priority over CTR ciphers), 3.) The HostbasedAcceptedAlgorithms and PubkeyAcceptedAlgorithms settings are now the same as HostKeyAlgorithms setting, 4.) The hmac-sha2-512-etm@openssh.com MAC was increased in priority due to its increased resistence against quantum computing attacks, and 5.) The ED25519 host keys were given priority over RSA host keys due to their greater efficiency.

Note: all commands below are to be executed as the *root* user.

1. **Re-generate the ED25519 and RSA keys**

```
rm /etc/ssh/ssh_host_*
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
```

2. ## Remove small Diffie-Hellman moduli

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv /etc/ssh/moduli.safe /etc/ssh/moduli
```

3. ## Enable the ED25519 and RSA keys

Enable the ED25519 and RSA *HostKey* directives in the */etc/ssh/sshd_config* file:

```
echo -e "\nHostKey /etc/ssh/ssh_host_ed25519_key\nHostKey
/etc/ssh/ssh_host_rsa_key" >> /etc/ssh/sshd_config
```

4. ## Restrict supported key exchange, cipher, and MAC algorithms

```
echo -e "# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms sntrup761x25519-
sha512@openssh.com,gss-curve25519-sha256-,curve25519-
sha256,curve25519-sha256@libssh.org,diffie-hellman-group18-
sha512,diffie-hellman-group-exchange-sha256,gss-group16-
sha512-,diffie-hellman-group16-sha512\n\nCiphers chacha20-
poly1305@openssh.com,aes256-gcm@openssh.com,aes256-
ctr,aes192-ctr,aes128-gcm@openssh.com,aes128-ctr\n\nMACs
hmac-sha2-512-etm@openssh.com,hmac-sha2-256-
etm@openssh.com,umac-128-etm@openssh.com\n\nRequiredRSASize
3072\n\nHostKeyAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-sha2-
512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256\n\nCASignatureAlgorithms sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\nGSSAPIKexAlgorithms gss-curve25519-sha256-,gss-
group16-sha512-\n\nHostbasedAcceptedAlgorithms sk-ssh-
ed25519-cert-v01@openssh.com,ssh-ed25519-cert-
v01@openssh.com,rsa-sha2-512-cert-v01@openssh.com,rsa-sha2-
256-cert-v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-
ed25519,rsa-sha2-512,rsa-sha2-
256\n\nPubkeyAcceptedAlgorithms sk-ssh-ed25519-cert-
```

```
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-sha2-
512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256" > /etc/ssh/sshd_config.d/ssh-
audit_hardening.conf
```

5. ## Restart OpenSSH server

```
service ssh restart
```

6. ## Implement connection rate throttling

Connection rate throttling is needed in order to protect against the DHEat denial-of-service attack. A complete and flexible solution is to use iptables to allow up to 10 connections every 10 seconds from any one source address. An alternate solution is to set OpenSSH's PerSourceMaxStartups directive to 1 (note, however, that this can cause incomplete results during ssh-audit scans, as well as other client failures when bursts of connections are made).

```
iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m
recent --set
iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m
recent --update --seconds 10 --hitcount 10 -j DROP
ip6tables -I INPUT -p tcp --dport 22 -m state --state NEW -
m recent --set
ip6tables -I INPUT -p tcp --dport 22 -m state --state NEW -
m recent --update --seconds 10 --hitcount 10 -j DROP
```

Enable persistence of the iptables rules across server reboots:

```
DEBIAN_FRONTEND=noninteractive apt install -q -y netfilter-
persistent iptables-persistent
service netfilter-persistent save
```

Note: Because of a bug in OpenSSH, 2048-bit DH moduli will still be used in some limited circumstances. Only a maximum score of 95% is possible.

Ubuntu 22.04 LTS Server

Last modified: April 22, 2024

Change Log:

Copy commands to clipboard

October 1, 2024: Re-ordered host keys to prioritize ED25519 due to efficiency. Re-ordered cipher list to prioritize larger key sizes as a countermeasure to quantum attacks.

April 22, 2024: added connection throttling instructions to counteract the DHEat denial-of-service attack.

Note: all commands below are to be executed as the *root* user.

1. ## Re-generate the RSA and ED25519 keys

```
rm /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. ## Enable the ED25519 and RSA keys

   Enable the ED25519 and RSA *HostKey* directives in the */etc/ssh/sshd_config* file:

```
echo -e "\nHostKey /etc/ssh/ssh_host_ed25519_key\nHostKey
/etc/ssh/ssh_host_rsa_key" >> /etc/ssh/sshd_config
```

3. ## Remove small Diffie-Hellman moduli

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv /etc/ssh/moduli.safe /etc/ssh/moduli
```

4. ## Restrict supported key exchange, cipher, and MAC algorithms

```
echo -e "# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms sntrup761x25519-
sha512@openssh.com,curve25519-sha256,curve25519-
sha256@libssh.org,gss-curve25519-sha256-,diffie-hellman-
group16-sha512,gss-group16-sha512-,diffie-hellman-group18-
sha512,diffie-hellman-group-exchange-sha256\n\nCiphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-
gcm@openssh.com,aes128-ctr\n\nMACs hmac-sha2-256-
etm@openssh.com,hmac-sha2-512-etm@openssh.com,umac-128-
etm@openssh.com\n\nHostKeyAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-sha2-
512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256\n\nCASignatureAlgorithms sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
```

```
256\n\nGSSAPIKexAlgorithms gss-curve25519-sha256-,gss-
group16-sha512-\n\nHostbasedAcceptedAlgorithms sk-ssh-
ed25519-cert-v01@openssh.com,ssh-ed25519-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512-cert-v01@openssh.com,rsa-sha2-512,rsa-sha2-256-
cert-v01@openssh.com,rsa-sha2-
256\n\nPubkeyAcceptedAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512-cert-
v01@openssh.com,rsa-sha2-512,rsa-sha2-256-cert-
v01@openssh.com,rsa-sha2-256" > /etc/ssh/sshd_config.d/ssh-
audit_hardening.conf
```

5. ## Implement connection rate throttling

   Connection rate throttling is needed in order to protect against the DHEat denial-of-service attack. A complete and flexible solution is to use iptables to allow up to 10 connections every 10 seconds from any one source address. An alternate solution is to set OpenSSH's PerSourceMaxStartups directive to 1 (note, however, that this can cause incomplete results during ssh-audit scans, as well as other client failures when bursts of connections are made).

   ```
   iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m
   recent --set
   iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m
   recent --update --seconds 10 --hitcount 10 -j DROP
   ip6tables -I INPUT -p tcp --dport 22 -m state --state NEW -
   m recent --set
   ip6tables -I INPUT -p tcp --dport 22 -m state --state NEW -
   m recent --update --seconds 10 --hitcount 10 -j DROP
   ```

   Enable persistence of the iptables rules across server reboots:

   ```
   DEBIAN_FRONTEND=noninteractive apt install -q -y netfilter-
   persistent iptables-persistent
   service netfilter-persistent save
   ```

6. ## Restart OpenSSH server

   ```
   service ssh restart
   ```

Note: Because of a bug in OpenSSH, 2048-bit DH moduli will still be used in some limited circumstances. Only a maximum score of 95% is possible.

Ubuntu 20.04 LTS Server

Last modified: April 24, 2024

Change Log:

Copy commands to clipboard

April 24, 2024: added connection throttling instructions to counteract the DHEat denial-of-service attack.

Note: all commands below are to be executed as the *root* user.

1. ## Re-generate the RSA and ED25519 keys

```
rm /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. ## Remove small Diffie-Hellman moduli

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv /etc/ssh/moduli.safe /etc/ssh/moduli
```

3. ## Enable the RSA and ED25519 keys

Enable the RSA and ED25519 *HostKey* directives in the */etc/ssh/sshd_config* file:

```
sed -i 's/^\#HostKey \/etc\/ssh\/ssh_host_\
(rsa\|ed25519\)_key$/HostKey \/etc\/ssh\/ssh_host_\1_key/g'
/etc/ssh/sshd_config
```

4. ## Restrict supported key exchange, cipher, and MAC algorithms

```
echo -e "\n# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms curve25519-sha256,curve25519-
sha256@libssh.org,diffie-hellman-group16-sha512,diffie-
hellman-group18-sha512,diffie-hellman-group-exchange-
sha256\nCiphers chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes192-
ctr,aes128-ctr\nMACs hmac-sha2-256-etm@openssh.com,hmac-
sha2-512-etm@openssh.com,umac-128-
etm@openssh.com\nHostKeyAlgorithms ssh-ed25519,ssh-ed25519-
cert-v01@openssh.com,sk-ssh-ed25519@openssh.com,sk-ssh-
ed25519-cert-v01@openssh.com,rsa-sha2-256,rsa-sha2-512,rsa-
```

```
sha2-256-cert-v01@openssh.com,rsa-sha2-512-cert-
v01@openssh.com" > /etc/ssh/sshd_config.d/ssh-
audit_hardening.conf
```

5. **Restart OpenSSH server**

```
service ssh restart
```

6. **Implement connection rate throttling**

Connection rate throttling is needed in order to protect against the DHEat denial-of-service attack. A complete and flexible solution is to use iptables to allow up to 10 connections every 10 seconds from any one source address. An alternate solution is to set OpenSSH's PerSourceMaxStartups directive to 1 (note, however, that this can cause incomplete results during ssh-audit scans, as well as other client failures when bursts of connections are made).

```
iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m
recent --set
iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m
recent --update --seconds 10 --hitcount 10 -j DROP
ip6tables -I INPUT -p tcp --dport 22 -m state --state NEW -
m recent --set
ip6tables -I INPUT -p tcp --dport 22 -m state --state NEW -
m recent --update --seconds 10 --hitcount 10 -j DROP
```

Enable persistence of the iptables rules across server reboots:

```
DEBIAN_FRONTEND=noninteractive apt install -q -y netfilter-
persistent iptables-persistent
service netfilter-persistent save
```

Note: Because of a bug in OpenSSH, 2048-bit DH moduli will still be used in some limited circumstances. Only a maximum score of 95% is possible.

Ubuntu 18.04 LTS Server

Last modified: February 8, 2020

Copy commands to clipboard

Note: all commands below are to be executed as the *root* user.

1. **Re-generate the RSA and ED25519 keys**

```
rm /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. **Remove small Diffie-Hellman moduli**

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv /etc/ssh/moduli.safe /etc/ssh/moduli
```

3. **Disable the DSA and ECDSA host keys**

Comment out the DSA and ECDSA *HostKey* directives in the */etc/ssh/sshd_config* file:

```
sed -i 's/^HostKey \/etc\/ssh\/ssh_host_\
(dsa\|ecdsa\)_key$/\#HostKey \/etc\/ssh\/ssh_host_\1_key/g'
/etc/ssh/sshd_config
```

4. **Restrict supported key exchange, cipher, and MAC algorithms**

```
echo -e "\n# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms curve25519-sha256,curve25519-
sha256@libssh.org,diffie-hellman-group16-sha512,diffie-
hellman-group18-sha512,diffie-hellman-group-exchange-
sha256\nCiphers chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes192-
ctr,aes128-ctr\nMACs hmac-sha2-256-etm@openssh.com,hmac-
sha2-512-etm@openssh.com,umac-128-
etm@openssh.com\nHostKeyAlgorithms ssh-ed25519,ssh-ed25519-
cert-v01@openssh.com" >> /etc/ssh/sshd_config
```

5. **Restart OpenSSH server**

```
service ssh restart
```

Note: Because of a bug in OpenSSH, 2048-bit DH moduli will still be used in some limited circumstances. Only a maximum score of 95% is possible.

Ubuntu 16.04 LTS Server

Last modified: July 6, 2020

Note: all commands below are to be executed as the *root* user.

Copy commands to clipboard

1. **Re-generate ED25519 key**

```
rm /etc/ssh/ssh_host_*
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. **Remove small Diffie-Hellman moduli**

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv /etc/ssh/moduli.safe /etc/ssh/moduli
```

3. **Disable the RSA, DSA, and ECDSA host keys**

   Comment out the RSA, DSA and ECDSA *HostKey* directives in the */etc/ssh/sshd_config* file:

```
sed -i 's/^HostKey \/etc\/ssh\/ssh_host_\
(rsa\|dsa\|ecdsa\)_key$/\#HostKey
\/etc\/ssh\/ssh_host_\1_key/g' /etc/ssh/sshd_config
```

4. **Restrict supported key exchange, cipher, and MAC algorithms**

```
echo -e "\n# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms curve25519-sha256@libssh.org,diffie-
hellman-group-exchange-sha256\nCiphers chacha20-
poly1305@openssh.com,aes256-gcm@openssh.com,aes128-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-ctr\nMACs
hmac-sha2-256-etm@openssh.com,hmac-sha2-512-
etm@openssh.com,umac-128-etm@openssh.com" >>
/etc/ssh/sshd_config
```

5. **Restart OpenSSH server**

```
service ssh restart
```

Note: Because of a bug in OpenSSH, 2048-bit DH moduli will still be used in some limited circumstances. Only a maximum score of 95% is possible.

Ubuntu 14.04 LTS Server

Last modified: October 17, 2017

Copy commands to clipboard

Note: all commands below are to be executed as the *root* user.

1. **Re-generate the RSA and ED25519 keys**

```
rm /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. **Remove small Diffie-Hellman moduli**

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv /etc/ssh/moduli.safe /etc/ssh/moduli
```

3. **Disable the DSA and ECDSA host keys**

   Comment out the DSA and ECDSA *HostKey* directives in the */etc/ssh/sshd_config* file:

```
sed -i 's/^HostKey \/etc\/ssh\/ssh_host_\
(dsa\|ecdsa\)_key$/\#HostKey \/etc\/ssh\/ssh_host_\1_key/g'
/etc/ssh/sshd_config
```

4. **Restrict supported key exchange, cipher, and MAC algorithms**

```
echo -e "\n# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms curve25519-sha256@libssh.org,diffie-
hellman-group-exchange-sha256\nCiphers chacha20-
poly1305@openssh.com,aes256-gcm@openssh.com,aes128-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-ctr\nMACs
hmac-sha2-256-etm@openssh.com,hmac-sha2-512-
etm@openssh.com,umac-128-etm@openssh.com" >>
/etc/ssh/sshd_config
```

5. **Restart OpenSSH server**

```
service ssh restart
```

Note: Because of a bug in OpenSSH, 2048-bit DH moduli will still be used in

some limited circumstances. Only a maximum score of 95% is possible.

Ubuntu Core 18 Server

Last modified: October 6, 2019

Copy commands to clipboard

Note: all commands below are to be executed as the *root* user.

1. ## Re-generate the RSA and ED25519 keys

   Note: It is highly recommended that you run the *ssh-keygen* commands below on another host. Some IoT devices do not have good entropy sources to generate sufficient keys with!

   ```
   ssh-keygen -t rsa -b 4096 -f ssh_host_rsa_key -N ""
   ssh-keygen -t ed25519 -f ssh_host_ed25519_key -N ""
   ```

   Be sure to upload the following 4 files to the target device's */etc/ssh* directory:
   - ssh_host_ed25519_key
   - ssh_host_ed25519_key.pub
   - ssh_host_rsa_key
   - ssh_host_rsa_key.pub

2. ## Remove small Diffie-Hellman moduli

   ```
   awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
   mv /etc/ssh/moduli.safe /etc/ssh/moduli
   ```

3. ## Restrict supported key exchange, cipher, and MAC algorithms

   ```
   echo -e "\n# Only enable RSA and ED25519 host
   keys.\nHostKey /etc/ssh/ssh_host_rsa_key\nHostKey
   /etc/ssh/ssh_host_ed25519_key\n\n# Restrict key exchange,
   cipher, and MAC algorithms, as per sshaudit.com\n#
   hardening guide.\nKexAlgorithms curve25519-
   sha256,curve25519-sha256@libssh.org,diffie-hellman-group16-
   sha512,diffie-hellman-group18-sha512,diffie-hellman-group-
   exchange-sha256\nCiphers chacha20-
   poly1305@openssh.com,aes256-gcm@openssh.com,aes128-
   gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-ctr\nMACs
   hmac-sha2-256-etm@openssh.com,hmac-sha2-512-
   ```

```
etm@openssh.com,umac-128-etm@openssh.com" >>
/etc/ssh/sshd_config
```

4. **Restart OpenSSH server**

```
service ssh reload
```

Note: Because of a bug in OpenSSH, 2048-bit DH moduli will still be used in some limited circumstances. Only a maximum score of 95% is possible.

Ubuntu Core 16 Server

Last modified: October 17, 2017

Note: all commands below are to be executed as the *root* user.

Copy commands to clipboard

1. **Re-generate the RSA and ED25519 keys**

   Note: It is highly recommended that you run the *ssh-keygen* commands below on another host. Some IoT devices do not have good entropy sources to generate sufficient keys with!

```
ssh-keygen -t rsa -b 4096 -f ssh_host_rsa_key -N ""
ssh-keygen -t ed25519 -f ssh_host_ed25519_key -N ""
```

   Be sure to upload the following 4 files to the target device's */etc/ssh* directory:
   - ssh_host_ed25519_key
   - ssh_host_ed25519_key.pub
   - ssh_host_rsa_key
   - ssh_host_rsa_key.pub

2. **Remove small Diffie-Hellman moduli**

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv /etc/ssh/moduli.safe /etc/ssh/moduli
```

3. **Restrict supported key exchange, cipher, and MAC algorithms**

```
sed -i 's/^MACs \(.*\)$/\#MACs \1/g' /etc/ssh/sshd_config
echo -e "\n# Restrict MAC algorithms, as per sshaudit.com
hardening guide.\nMACs hmac-sha2-512-etm@openssh.com,hmac-
```

```
sha2-256-etm@openssh.com,umac-128-etm@openssh.com" >>
/etc/ssh/sshd_config
```

4. **Restart OpenSSH server**

```
service ssh reload
```

Note: Because of a bug in OpenSSH, 2048-bit DH moduli will still be used in some limited circumstances. Only a maximum score of 95% is possible.

Debian Bookworm (Debian 12)

Last modified: October 1, 2024

Change Log:

Copy commands to clipboard

October 1, 2024: Re-ordered host keys to prioritize ED25519 due to efficiency. Re-ordered cipher list to prioritize larger key sizes as a countermeasure to quantum attacks.

April 24, 2024: added connection throttling instructions to counteract the DHEat denial-of-service attack.

Note: all commands below are to be executed as the *root* user.

1. **Re-generate the RSA and ED25519 keys**

```
rm /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. **Enable the ED25519 and RSA keys**

Enable the ED25519 and RSA *HostKey* directives in the */etc/ssh/sshd_config* file:

```
echo -e "\nHostKey /etc/ssh/ssh_host_ed25519_key\nHostKey
/etc/ssh/ssh_host_rsa_key" >> /etc/ssh/sshd_config
```

3. **Remove small Diffie-Hellman moduli**

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv /etc/ssh/moduli.safe /etc/ssh/moduli
```

4. **Restrict supported key exchange, cipher, and MAC algorithms**

```
echo -e "# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening guide.\n
KexAlgorithms sntrup761x25519-
sha512@openssh.com,curve25519-sha256,curve25519-
sha256@libssh.org,gss-curve25519-sha256-,diffie-hellman-
group16-sha512,gss-group16-sha512-,diffie-hellman-group18-
sha512,diffie-hellman-group-exchange-sha256\n\nCiphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-
gcm@openssh.com,aes128-ctr\n\nMACs hmac-sha2-256-
etm@openssh.com,hmac-sha2-512-etm@openssh.com,umac-128-
etm@openssh.com\n\nHostKeyAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-sha2-
512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256\n\nRequiredRSASize
3072\n\nCASignatureAlgorithms sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\nGSSAPIKexAlgorithms gss-curve25519-sha256-,gss-
group16-sha512-\n\nHostbasedAcceptedAlgorithms sk-ssh-
ed25519-cert-v01@openssh.com,ssh-ed25519-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512-cert-v01@openssh.com,rsa-sha2-512,rsa-sha2-256-
cert-v01@openssh.com,rsa-sha2-
256\n\nPubkeyAcceptedAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512-cert-
v01@openssh.com,rsa-sha2-512,rsa-sha2-256-cert-
v01@openssh.com,rsa-sha2-256\n\n" >
/etc/ssh/sshd_config.d/ssh-audit_hardening.conf
```

5. ## Restart OpenSSH server

```
service ssh restart
```

6. ## Implement connection rate throttling

Connection rate throttling is needed in order to protect against the DHEat denial-of-service attack. A complete and flexible solution is to use iptables to allow up to 10 connections every 10 seconds from any one source address. An alternate solution is to set OpenSSH's PerSourceMaxStartups directive to 1 (note, however, that this can cause incomplete results during ssh-audit scans, as well as other client failures when bursts of connections are made).

```
DEBIAN_FRONTEND=noninteractive apt install -q -y iptables
netfilter-persistent iptables-persistent
```

```
iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m
recent --set
iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m
recent --update --seconds 10 --hitcount 10 -j DROP
ip6tables -I INPUT -p tcp --dport 22 -m state --state NEW -
m recent --set
ip6tables -I INPUT -p tcp --dport 22 -m state --state NEW -
m recent --update --seconds 10 --hitcount 10 -j DROP
```

Enable persistence of the iptables rules across server reboots:

```
service netfilter-persistent save
```

Debian Bullseye (Debian 11)

Last modified: September 17, 2021

Copy commands to clipboard

Note: all commands below are to be executed as the *root* user.

1. ## Re-generate the RSA and ED25519 keys

```
rm -f /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. ## Enable the RSA and ED25519 keys

Enable the RSA and ED25519 *HostKey* directives in the */etc/ssh/sshd_config* file:

```
sed -i 's/^\#HostKey \/etc\/ssh\/ssh_host_\
(rsa\|ed25519\)_key$/HostKey \/etc\/ssh\/ssh_host_\1_key/g'
/etc/ssh/sshd_config
```

3. ## Remove small Diffie-Hellman moduli

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv -f /etc/ssh/moduli.safe /etc/ssh/moduli
```

4. ## Restrict supported key exchange, cipher, and MAC algorithms

```
echo -e "\n# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms curve25519-sha256,curve25519-
sha256@libssh.org,diffie-hellman-group16-sha512,diffie-
hellman-group18-sha512,diffie-hellman-group-exchange-
sha256\nCiphers chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes192-
ctr,aes128-ctr\nMACs hmac-sha2-256-etm@openssh.com,hmac-
sha2-512-etm@openssh.com,umac-128-
etm@openssh.com\nHostKeyAlgorithms ssh-ed25519,ssh-ed25519-
cert-v01@openssh.com,sk-ssh-ed25519@openssh.com,sk-ssh-
ed25519-cert-v01@openssh.com,rsa-sha2-256,rsa-sha2-512,rsa-
sha2-256-cert-v01@openssh.com,rsa-sha2-512-cert-
v01@openssh.com" > /etc/ssh/sshd_config.d/ssh-
audit_hardening.conf
```

5. **Restart OpenSSH server**

```
service ssh restart
```

Debian Buster (Debian 10)

Last modified: September 17, 2021

Copy commands to clipboard

Note: all commands below are to be executed as the *root* user.

1. **Re-generate the RSA and ED25519 keys**

```
rm -f /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. **Enable the RSA and ED25519 keys**

   Enable the RSA and ED25519 *HostKey* directives in the */etc/ssh/sshd_config* file:

```
sed -i 's/^\#HostKey \/etc\/ssh\/ssh_host_\
(rsa\|ed25519\)_key$/HostKey \/etc\/ssh\/ssh_host_\1_key/g'
/etc/ssh/sshd_config
```

3. **Remove small Diffie-Hellman moduli**

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv -f /etc/ssh/moduli.safe /etc/ssh/moduli
```

4. **Restrict supported key exchange, cipher, and MAC algorithms**

```
echo -e "\n# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms curve25519-sha256,curve25519-
sha256@libssh.org,diffie-hellman-group16-sha512,diffie-
hellman-group18-sha512,diffie-hellman-group-exchange-
sha256\nCiphers chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes192-
ctr,aes128-ctr\nMACs hmac-sha2-256-etm@openssh.com,hmac-
sha2-512-etm@openssh.com,umac-128-
etm@openssh.com\nHostKeyAlgorithms ssh-ed25519,ssh-ed25519-
cert-v01@openssh.com,rsa-sha2-256,rsa-sha2-512,rsa-sha2-
256-cert-v01@openssh.com,rsa-sha2-512-cert-v01@openssh.com"
>> /etc/ssh/sshd_config
```

5. **Restart OpenSSH server**

```
service ssh restart
```

Rocky Linux 9

Last modified: October 1, 2024

Change Log:

Copy commands to clipboard

October 1, 2024: Re-ordered host keys to prioritize ED25519 due to efficiency. Re-ordered cipher list to prioritize larger key sizes as a countermeasure to quantum attacks.

April 24, 2024: added connection throttling instructions to counteract the DHEat denial-of-service attack.

Note: all commands below are to be executed as the *root* user.

1. **Re-generate the RSA and ED25519 keys**

```
rm -f /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. **Enable the ED25519 and RSA keys**

Enable the ED25519 and RSA *HostKey* directives in the */etc/ssh/sshd_config* file:

```
echo -e "\nHostKey /etc/ssh/ssh_host_ed25519_key\nHostKey
/etc/ssh/ssh_host_rsa_key" >> /etc/ssh/sshd_config
```

3. ## Remove small Diffie-Hellman moduli

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv -f /etc/ssh/moduli.safe /etc/ssh/moduli
```

4. ## Restrict supported key exchange, cipher, and MAC algorithms

```
echo -e "# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms sntrup761x25519-
sha512@openssh.com,curve25519-sha256,curve25519-
sha256@libssh.org,gss-curve25519-sha256-,diffie-hellman-
group16-sha512,gss-group16-sha512-,diffie-hellman-group18-
sha512,diffie-hellman-group-exchange-sha256\n\nCiphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-
gcm@openssh.com,aes128-ctr\n\nMACs hmac-sha2-256-
etm@openssh.com,hmac-sha2-512-etm@openssh.com,umac-128-
etm@openssh.com\n\nHostKeyAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-sha2-
512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256\n\nRequiredRSASize
3072\n\nCASignatureAlgorithms sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\nGSSAPIKexAlgorithms gss-curve25519-sha256-,gss-
group16-sha512-\n\nHostbasedAcceptedAlgorithms sk-ssh-
ed25519-cert-v01@openssh.com,ssh-ed25519-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512-cert-v01@openssh.com,rsa-sha2-512,rsa-sha2-256-
cert-v01@openssh.com,rsa-sha2-
256\n\nPubkeyAcceptedAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-sha2-
512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256\n\n" > /etc/crypto-policies/back-
ends/opensshserver.config
```

5. ## Restart OpenSSH server

```
systemctl restart sshd
```

6. **Implement connection rate throttling**

   Connection rate throttling is needed in order to protect against the DHEat denial-of-service attack. A complete and flexible solution is to use iptables/firewalld to allow up to 10 connections every 10 seconds from any one source address. An alternate solution is to set OpenSSH's PerSourceMaxStartups directive to 1 (note, however, that this can cause incomplete results during ssh-audit scans, as well as other client failures when bursts of connections are made).

   ```
   firewall-cmd --permanent --direct --add-rule ipv4 filter
   INPUT 0 -p tcp --dport 22 -m state --state NEW -m recent --
   set
   firewall-cmd --permanent --direct --add-rule ipv4 filter
   INPUT 1 -p tcp --dport 22 -m state --state NEW -m recent --
   update --seconds 10 --hitcount 10 -j DROP
   firewall-cmd --permanent --direct --add-rule ipv6 filter
   INPUT 0 -p tcp --dport 22 -m state --state NEW -m recent --
   set
   firewall-cmd --permanent --direct --add-rule ipv6 filter
   INPUT 1 -p tcp --dport 22 -m state --state NEW -m recent --
   update --seconds 10 --hitcount 10 -j DROP
   ```

   Reload firewalld to enable new rules:

   ```
   systemctl reload firewalld
   ```

RedHat Enterprise Linux 8 Server / CentOS 8 Server

Last modified: October 20, 2020 Copy commands to clipboard

Note: all commands below are to be executed as the *root* user.

1. **Re-generate the RSA and ED25519 keys**

   ```
   rm -f /etc/ssh/ssh_host_*
   ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
   ""
   ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
   ""
   chgrp ssh_keys /etc/ssh/ssh_host_ed25519_key
   /etc/ssh/ssh_host_rsa_key
   ```

```
chmod g+r /etc/ssh/ssh_host_ed25519_key
/etc/ssh/ssh_host_rsa_key
```

2. **Remove small Diffie-Hellman moduli**

   ```
   awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
   mv -f /etc/ssh/moduli.safe /etc/ssh/moduli
   ```

3. **Disable ECDSA host key**

   Comment out the ECDSA *HostKey* directive in the */etc/ssh/sshd_config* file:

   ```
   sed -i 's/^HostKey
   \/etc\/ssh\/ssh_host_ecdsa_key$/\#HostKey
   \/etc\/ssh\/ssh_host_ecdsa_key/g' /etc/ssh/sshd_config
   ```

4. **Restrict supported key exchange, cipher, and MAC algorithms**

   ```
   cp /etc/crypto-policies/back-ends/opensshserver.config
   /etc/crypto-policies/back-ends/opensshserver.config.orig
   echo -e "CRYPTO_POLICY='-oCiphers=chacha20-
   poly1305@openssh.com,aes256-gcm@openssh.com,aes128-
   gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-ctr -
   oMACs=hmac-sha2-256-etm@openssh.com,hmac-sha2-512-
   etm@openssh.com,umac-128-etm@openssh.com -
   oGSSAPIKexAlgorithms=gss-curve25519-sha256- -
   oKexAlgorithms=curve25519-sha256,curve25519-
   sha256@libssh.org,diffie-hellman-group16-sha512,diffie-
   hellman-group18-sha512,diffie-hellman-group-exchange-sha256
   -oHostKeyAlgorithms=ssh-ed25519,ssh-ed25519-cert-
   v01@openssh.com,rsa-sha2-256,rsa-sha2-512 -
   oPubkeyAcceptedKeyTypes=ssh-ed25519,ssh-ed25519-cert-
   v01@openssh.com,rsa-sha2-256,rsa-sha2-512'" > /etc/crypto-
   policies/back-ends/opensshserver.config
   ```

5. **Restart OpenSSH server**

   ```
   systemctl restart sshd.service
   ```

RedHat Enterprise Linux 7 Server / CentOS 7 Server

Last modified: February 18, 2024

Note: all commands below are to be executed as the *root* user.

Copy commands to clipboard

1. ## Disable automatic re-generation of RSA & ECDSA keys

```
mkdir -p /etc/systemd/system/sshd-keygen.service.d
cat << EOF > /etc/systemd/system/sshd-keygen.service.d/ssh-
audit.conf
[Unit]
ConditionFileNotEmpty=
ConditionFileNotEmpty=!/etc/ssh/ssh_host_ed25519_key
EOF
systemctl daemon-reload
```

2. ## Re-generate the ED25519 key

```
rm -f /etc/ssh/ssh_host_*
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
chgrp ssh_keys /etc/ssh/ssh_host_ed25519_key
chmod g+r /etc/ssh/ssh_host_ed25519_key
```

3. ## Remove small Diffie-Hellman moduli

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv -f /etc/ssh/moduli.safe /etc/ssh/moduli
```

4. ## Disable the RSA, DSA, and ECDSA host keys

   Comment out the RSA, DSA, and ECDSA *HostKey* directives in the */etc/ssh/sshd_config* file:

```
sed -i 's/^HostKey \/etc\/ssh\/ssh_host_\
(rsa\|dsa\|ecdsa\)_key$/\#HostKey
\/etc\/ssh\/ssh_host_\1_key/g' /etc/ssh/sshd_config
```

5. ## Restrict supported key exchange, cipher, and MAC algorithms

```
echo -e "\n# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms curve25519-sha256,curve25519-
sha256@libssh.org,diffie-hellman-group18-sha512,diffie-
hellman-group16-sha512,diffie-hellman-group-exchange-
sha256\nCiphers chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes192-
```

```
ctr,aes128-ctr\nMACs hmac-sha2-256-etm@openssh.com,hmac-
sha2-512-etm@openssh.com,umac-128-etm@openssh.com" >>
/etc/ssh/sshd_config
```

6. **Restart OpenSSH server**

```
systemctl restart sshd.service
```

Amazon Linux 2023

Last modified: October 1, 2024

Change Log:

Copy commands to clipboard

October 1, 2024: Re-ordered host keys to prioritize ED25519 due to efficiency. Re-ordered cipher list to prioritize larger key sizes as a countermeasure to quantum attacks.

April 22, 2024: added connection throttling instructions to counteract the DHEat denial-of-service attack.

March 15, 2024: Initial revision.

Note: all commands below are to be executed as the *root* user.

1. **Re-generate the RSA and ED25519 keys**

```
rm -f /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. **Enable the ED25519 and RSA keys**

Enable the ED25519 and RSA *HostKey* directives in the */etc/ssh/sshd_config* file:

```
echo -e "\nHostKey /etc/ssh/ssh_host_ed25519_key\nHostKey
/etc/ssh/ssh_host_rsa_key" >> /etc/ssh/sshd_config
```

3. **Remove small Diffie-Hellman moduli**

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv -f /etc/ssh/moduli.safe /etc/ssh/moduli
```

4. **Restrict supported key exchange, cipher, and MAC algorithms**

```
echo -e "# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms sntrup761x25519-
sha512@openssh.com,curve25519-sha256,curve25519-
sha256@libssh.org,gss-curve25519-sha256-,diffie-hellman-
group16-sha512,gss-group16-sha512-,diffie-hellman-group18-
sha512,diffie-hellman-group-exchange-sha256\n\nCiphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-
gcm@openssh.com,aes128-ctr\n\nMACs hmac-sha2-256-
etm@openssh.com,hmac-sha2-512-etm@openssh.com,umac-128-
etm@openssh.com\n\nHostKeyAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-sha2-
512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256\n\nCASignatureAlgorithms sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\nGSSAPIKexAlgorithms gss-curve25519-sha256-,gss-
group16-sha512-\n\nHostbasedAcceptedAlgorithms sk-ssh-
ed25519-cert-v01@openssh.com,ssh-ed25519-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512-cert-v01@openssh.com,rsa-sha2-512,rsa-sha2-256-
cert-v01@openssh.com,rsa-sha2-
256\n\nPubkeyAcceptedAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-sha2-
512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256\n\n" > /etc/crypto-policies/back-
ends/opensshserver.config
```

5. ## Restart OpenSSH server

   ```
   systemctl restart sshd
   ```

6. ## Implement connection rate throttling

   Connection rate throttling is needed in order to protect against the DHEat denial-of-service attack. A complete and flexible solution is to use iptables to allow up to 10 connections every 10 seconds from any one source address. An alternate solution is to set OpenSSH's PerSourceMaxStartups directive to 1 (note, however, that this can cause incomplete results during ssh-audit scans, as well as other client failures when bursts of connections are made).

   ```
   dnf install -y iptables
   iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m
   recent --set
   ```

```
iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m
recent --update --seconds 10 --hitcount 10 -j DROP
ip6tables -I INPUT -p tcp --dport 22 -m state --state NEW -
m recent --set
ip6tables -I INPUT -p tcp --dport 22 -m state --state NEW -
m recent --update --seconds 10 --hitcount 10 -j DROP
```

Enable persistence of the iptables rules across server reboots:

```
dnf install -y iptables-services
iptables-save > /etc/sysconfig/iptables
ip6tables-save > /etc/sysconfig/ip6tables
systemctl enable iptables
systemctl enable ip6tables
systemctl start iptables
systemctl start ip6tables
```

pfSense 2.4

Last modified: October 17, 2017

Note: all commands below are to be executed as the *root* user.

Copy commands to clipboard

1. ## Re-generate the RSA and ED25519 keys

```
rm /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. ## Remove small Diffie-Hellman moduli

```
awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe
mv -f /etc/ssh/moduli.safe /etc/ssh/moduli
```

3. ## Restrict supported key exchange, cipher, and MAC algorithms

```
sed -i.bak 's/^MACs \(.*\)$/\#MACs \1/g'
/etc/ssh/sshd_config && rm /etc/ssh/sshd_config.bak
echo "" | echo "MACs hmac-sha2-512-etm@openssh.com,hmac-
sha2-256-etm@openssh.com,umac-128-etm@openssh.com" >>
/etc/ssh/sshd_config
```

4. **Restart OpenSSH server**

```
service sshd onerestart
```

Note: Because of a bug in OpenSSH, 2048-bit DH moduli will still be used in some limited circumstances. Only a maximum score of 95% is possible.

OpenBSD 6.2 Server

Last modified: October 20, 2020

Note: all commands below are to be executed as the *root* user.

Copy commands to clipboard

1. **Re-generate the RSA and ED25519 keys**

```
rm /etc/ssh/ssh_host_*
ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N
""
ssh-keygen -t ed25519 -f /etc/ssh/ssh_host_ed25519_key -N
""
```

2. **Create custom Diffie-Hellman groups**

```
ssh-keygen -G /etc/ssh/moduli -b 3072
```

Note: This will likely take some time to complete.

3. **Disable the DSA and ECDSA host keys**

```
echo -e "\n# Restrict host keys to ED25519 and RSA
only.\nHostKeyAlgorithms ssh-ed25519\n" >>
/etc/ssh/sshd_config
```

4. **Restrict supported key exchange, cipher, and MAC algorithms**

```
echo -e "# Restrict key exchange, cipher, and MAC
algorithms, as per sshaudit.com\n# hardening
guide.\nKexAlgorithms curve25519-sha256,curve25519-
sha256@libssh.org,diffie-hellman-group16-sha512,diffie-
hellman-group18-sha512,diffie-hellman-group-exchange-
sha256\nCiphers chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes192-
ctr,aes128-ctr\nMACs hmac-sha2-256-etm@openssh.com,hmac-
```

```
sha2-512-etm@openssh.com,umac-128-etm@openssh.com" >>
/etc/ssh/sshd_config
```

5. **Restart OpenSSH server**

```
kill -HUP `cat /var/run/sshd.pid`
```

Note: Because of a bug in OpenSSH, 2048-bit DH moduli will still be used in some limited circumstances. Only a maximum score of 95% is possible.

Amazon Linux 2023 Client *Last modified: October 1, 2024*

Change Log: Copy commands to clipboard

October 1, 2024: Re-ordered cipher list to prioritize larger key sizes as a countermeasure to quantum attacks.

April 22, 2024: added connection throttling instructions to counteract the DHEat denial-of-service attack.

March 15, 2024: Initial revision.

- **Run the following in a terminal to harden the SSH client for the local user:**

```
mkdir -p -m 0700 ~/.ssh; echo -e "\nHost *\n Ciphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-
gcm@openssh.com,aes128-ctr\n\n KexAlgorithms
sntrup761x25519-sha512@openssh.com,gss-curve25519-
sha256-,curve25519-sha256,curve25519-sha256@libssh.org,gss-
group16-sha512-,diffie-hellman-group16-sha512,diffie-
hellman-group18-sha512,diffie-hellman-group-exchange-
sha256\n\n MACs hmac-sha2-256-etm@openssh.com,hmac-sha2-
512-etm@openssh.com,umac-128-etm@openssh.com\n\n
HostKeyAlgorithms sk-ssh-ed25519-cert-v01@openssh.com,ssh-
ed25519-cert-v01@openssh.com,rsa-sha2-512-cert-
v01@openssh.com,rsa-sha2-256-cert-v01@openssh.com,sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\n CASignatureAlgorithms sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\n GSSAPIKexAlgorithms gss-curve25519-sha256-,gss-
group16-sha512-\n\n HostbasedAcceptedAlgorithms sk-ssh-
ed25519-cert-v01@openssh.com,ssh-ed25519-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512-cert-v01@openssh.com,rsa-sha2-512,rsa-sha2-256-
```

```
cert-v01@openssh.com,rsa-sha2-256\n\n
PubkeyAcceptedAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512-cert-
v01@openssh.com,rsa-sha2-512,rsa-sha2-256-cert-
v01@openssh.com,rsa-sha2-256\n\n" >> ~/.ssh/config
```

Debian 12 Client *Last modified: October 1, 2024*

Change Log: Copy commands to clipboard

October 1, 2024: Added RequiredRSASize directive to enforce a minimum of 3072-bit user and host-based authentication keys. Re-ordered cipher list to prioritize larger key sizes as a countermeasure to quantum attacks.

March 15, 2024: Initial revision.

- **Run the following in a terminal to harden the SSH client for the local user:**

```
mkdir -p -m 0700 ~/.ssh; echo -e "\nHost *\n Ciphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-
gcm@openssh.com,aes128-ctr\n\n KexAlgorithms
sntrup761x25519-sha512@openssh.com,gss-curve25519-
sha256-,curve25519-sha256,curve25519-sha256@libssh.org,gss-
group16-sha512-,diffie-hellman-group16-sha512,diffie-
hellman-group18-sha512,diffie-hellman-group-exchange-
sha256\n\n MACs hmac-sha2-256-etm@openssh.com,hmac-sha2-
512-etm@openssh.com,umac-128-etm@openssh.com\n\n
RequiredRSASize 3072\n\n HostKeyAlgorithms sk-ssh-ed25519-
cert-v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-
sha2-512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256\n\n CASignatureAlgorithms sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\n GSSAPIKexAlgorithms gss-curve25519-sha256-,gss-
group16-sha512-\n\n HostbasedAcceptedAlgorithms sk-ssh-
ed25519-cert-v01@openssh.com,ssh-ed25519-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512-cert-v01@openssh.com,rsa-sha2-512,rsa-sha2-256-
cert-v01@openssh.com,rsa-sha2-256\n\n
PubkeyAcceptedAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512-cert-
```

```
v01@openssh.com,rsa-sha2-512,rsa-sha2-256-cert-
v01@openssh.com,rsa-sha2-256\n\n" >> ~/.ssh/config
```

Rocky Linux 9 Client

Last modified: October 1, 2024

Change Log:

Copy commands to clipboard

October 1, 2024: Added RequiredRSASize directive to enforce a minimum of 3072-bit user and host-based authentication keys. Re-ordered cipher list to prioritize larger key sizes as a countermeasure to quantum attacks.

March 15, 2024: Initial revision.

- **Run the following in a terminal to harden the SSH client for the local user:**

```
mkdir -p -m 0700 ~/.ssh; echo -e "\nHost *\n Ciphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-
gcm@openssh.com,aes128-ctr\n\n KexAlgorithms
sntrup761x25519-sha512@openssh.com,gss-curve25519-
sha256-,curve25519-sha256,curve25519-sha256@libssh.org,gss-
group16-sha512-,diffie-hellman-group16-sha512,diffie-
hellman-group18-sha512,diffie-hellman-group-exchange-
sha256\n\n MACs hmac-sha2-256-etm@openssh.com,hmac-sha2-
512-etm@openssh.com,umac-128-etm@openssh.com\n\n
RequiredRSASize 3072\n\n HostKeyAlgorithms sk-ssh-ed25519-
cert-v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-
sha2-512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256\n\n CASignatureAlgorithms sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\n GSSAPIKexAlgorithms gss-curve25519-sha256-,gss-
group16-sha512-\n\n HostbasedAcceptedAlgorithms sk-ssh-
ed25519-cert-v01@openssh.com,ssh-ed25519-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512-cert-v01@openssh.com,rsa-sha2-512,rsa-sha2-256-
cert-v01@openssh.com,rsa-sha2-256\n\n
PubkeyAcceptedAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512-cert-
v01@openssh.com,rsa-sha2-512,rsa-sha2-256-cert-
v01@openssh.com,rsa-sha2-256\n\n" >> ~/.ssh/config
```

Ubuntu 24.04 LTS Client / Linux Mint 22 Client

Last modified: October 1, 2024

Copy commands to clipboard

Change Log:

October 1, 2024: Added RequiredRSASize directive to enforce a minimum of 3072-bit user and host-based authentication keys.

April 29, 2024: Initial revision. In comparison to Ubuntu 22.04 LTS Client guide, the following changes were made: 1.) For key exchanges, diffie-hellman-group18-sha512 and diffie-hellman-group-exchange-sha256 were prioritized over diffie-hellman-group16-sha512 due to greater security strength, 2.) For ciphers, 256-bit AES ciphers were prioritized over 192 and 128-bit AES ciphers due to their increased resistence against quantum computing attacks (previously, weaker GCM ciphers had priority over CTR ciphers), 3.) The HostbasedAcceptedAlgorithms and PubkeyAcceptedAlgorithms settings are now the same as HostKeyAlgorithms setting, and 4.) The hmac-sha2-512-etm@openssh.com MAC was increased in priority due to its increased resistence against quantum computing attacks.

- **Run the following in a terminal to harden the SSH client for the local user:**

```
mkdir -p -m 0700 ~/.ssh; echo -e "\nHost *\n Ciphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-
gcm@openssh.com,aes128-ctr\n\n KexAlgorithms
sntrup761x25519-sha512@openssh.com,gss-curve25519-
sha256-,curve25519-sha256,curve25519-
sha256@libssh.org,diffie-hellman-group18-sha512,diffie-
hellman-group-exchange-sha256,gss-group16-sha512-,diffie-
hellman-group16-sha512\n\n MACs hmac-sha2-512-
etm@openssh.com,hmac-sha2-256-etm@openssh.com,umac-128-
etm@openssh.com\n\n RequiredRSASize 3072\n\n
HostKeyAlgorithms sk-ssh-ed25519-cert-v01@openssh.com,ssh-
ed25519-cert-v01@openssh.com,rsa-sha2-512-cert-
v01@openssh.com,rsa-sha2-256-cert-v01@openssh.com,sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\n CASignatureAlgorithms sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\n GSSAPIKexAlgorithms gss-curve25519-sha256-,gss-
group16-sha512-\n\n HostbasedAcceptedAlgorithms sk-ssh-
ed25519-cert-v01@openssh.com,ssh-ed25519-cert-
v01@openssh.com,rsa-sha2-512-cert-v01@openssh.com,rsa-sha2-
256-cert-v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-
ed25519,rsa-sha2-512,rsa-sha2-256\n\n
PubkeyAcceptedAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,rsa-sha2-
512-cert-v01@openssh.com,rsa-sha2-256-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512,rsa-sha2-256\n\n" >> ~/.ssh/config
```

Ubuntu 22.04 LTS Client / Linux Mint 21 Client

Last modified: October 1, 2024

Copy commands to clipboard

Change Log:

October 1, 2024: Re-ordered cipher list to prioritize larger key sizes as a countermeasure to quantum attacks.

- **Run the following in a terminal to harden the SSH client for the local user:**

```
mkdir -p -m 0700 ~/.ssh; echo -e "\nHost *\n Ciphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-
gcm@openssh.com,aes128-ctr\n\n KexAlgorithms
sntrup761x25519-sha512@openssh.com,gss-curve25519-
sha256-,curve25519-sha256,curve25519-sha256@libssh.org,gss-
group16-sha512-,diffie-hellman-group16-sha512,diffie-
hellman-group18-sha512,diffie-hellman-group-exchange-
sha256\n\n MACs hmac-sha2-256-etm@openssh.com,hmac-sha2-
512-etm@openssh.com,umac-128-etm@openssh.com\n\n
HostKeyAlgorithms sk-ssh-ed25519-cert-v01@openssh.com,ssh-
ed25519-cert-v01@openssh.com,rsa-sha2-512-cert-
v01@openssh.com,rsa-sha2-256-cert-v01@openssh.com,sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\n CASignatureAlgorithms sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512,rsa-sha2-
256\n\n GSSAPIKexAlgorithms gss-curve25519-sha256-,gss-
group16-sha512-\n\n HostbasedAcceptedAlgorithms sk-ssh-
ed25519-cert-v01@openssh.com,ssh-ed25519-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,ssh-ed25519,rsa-
sha2-512-cert-v01@openssh.com,rsa-sha2-512,rsa-sha2-256-
cert-v01@openssh.com,rsa-sha2-256\n\n
PubkeyAcceptedAlgorithms sk-ssh-ed25519-cert-
v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,sk-ssh-
ed25519@openssh.com,ssh-ed25519,rsa-sha2-512-cert-
v01@openssh.com,rsa-sha2-512,rsa-sha2-256-cert-
v01@openssh.com,rsa-sha2-256\n\n" >> ~/.ssh/config
```

Ubuntu 20.04 LTS Client / Linux Mint 20 Client

Last modified: October 20, 2020

- **Run the following in a terminal to harden the SSH client for the local user:**

Copy commands to clipboard

```
mkdir -p -m 0700 ~/.ssh; echo -e "\nHost *\n Ciphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes192-
ctr,aes128-ctr\n KexAlgorithms curve25519-
sha256,curve25519-sha256@libssh.org,diffie-hellman-group16-
sha512,diffie-hellman-group18-sha512,diffie-hellman-group-
exchange-sha256\n MACs hmac-sha2-256-etm@openssh.com,hmac-
sha2-512-etm@openssh.com,umac-128-etm@openssh.com\n
HostKeyAlgorithms ssh-ed25519,ssh-ed25519-cert-
v01@openssh.com,sk-ssh-ed25519@openssh.com,sk-ssh-ed25519-
cert-v01@openssh.com,rsa-sha2-256,rsa-sha2-256-cert-
v01@openssh.com,rsa-sha2-512,rsa-sha2-512-cert-
v01@openssh.com\n" >> ~/.ssh/config
```

Ubuntu 18.04 LTS Client / Linux Mint 19 Client

Last modified: October 20, 2020

Copy commands to clipboard

- **Run the following in a terminal to harden the SSH client for the local user:**

```
mkdir -p -m 0700 ~/.ssh; echo -e "\nHost *\n Ciphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes192-
ctr,aes128-ctr\n KexAlgorithms curve25519-
sha256,curve25519-sha256@libssh.org,diffie-hellman-group16-
sha512,diffie-hellman-group18-sha512,diffie-hellman-group-
exchange-sha256\n MACs hmac-sha2-256-etm@openssh.com,hmac-
sha2-512-etm@openssh.com,umac-128-etm@openssh.com\n
HostKeyAlgorithms ssh-ed25519,ssh-ed25519-cert-
v01@openssh.com,rsa-sha2-256,rsa-sha2-512\n" >>
~/.ssh/config
```

Ubuntu 16.04 LTS Client / Linux Mint 18 Client

Last modified: October 20, 2020

Copy commands to clipboard

- **Run the following in a terminal to harden the SSH client for the local user:**

```
mkdir -p -m 0700 ~/.ssh; echo -e "\nHost *\n Ciphers
chacha20-poly1305@openssh.com,aes256-
gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes192-
ctr,aes128-ctr\n KexAlgorithms curve25519-
sha256@libssh.org,diffie-hellman-group-exchange-sha256\n
MACs hmac-sha2-256-etm@openssh.com,hmac-sha2-512-
etm@openssh.com,umac-128-etm@openssh.com\n
HostKeyAlgorithms ssh-ed25519,ssh-ed25519-cert-
v01@openssh.com,rsa-sha2-256,rsa-sha2-512\n" >>
~/.ssh/config
```
