Search guide

How to Secure Your Server

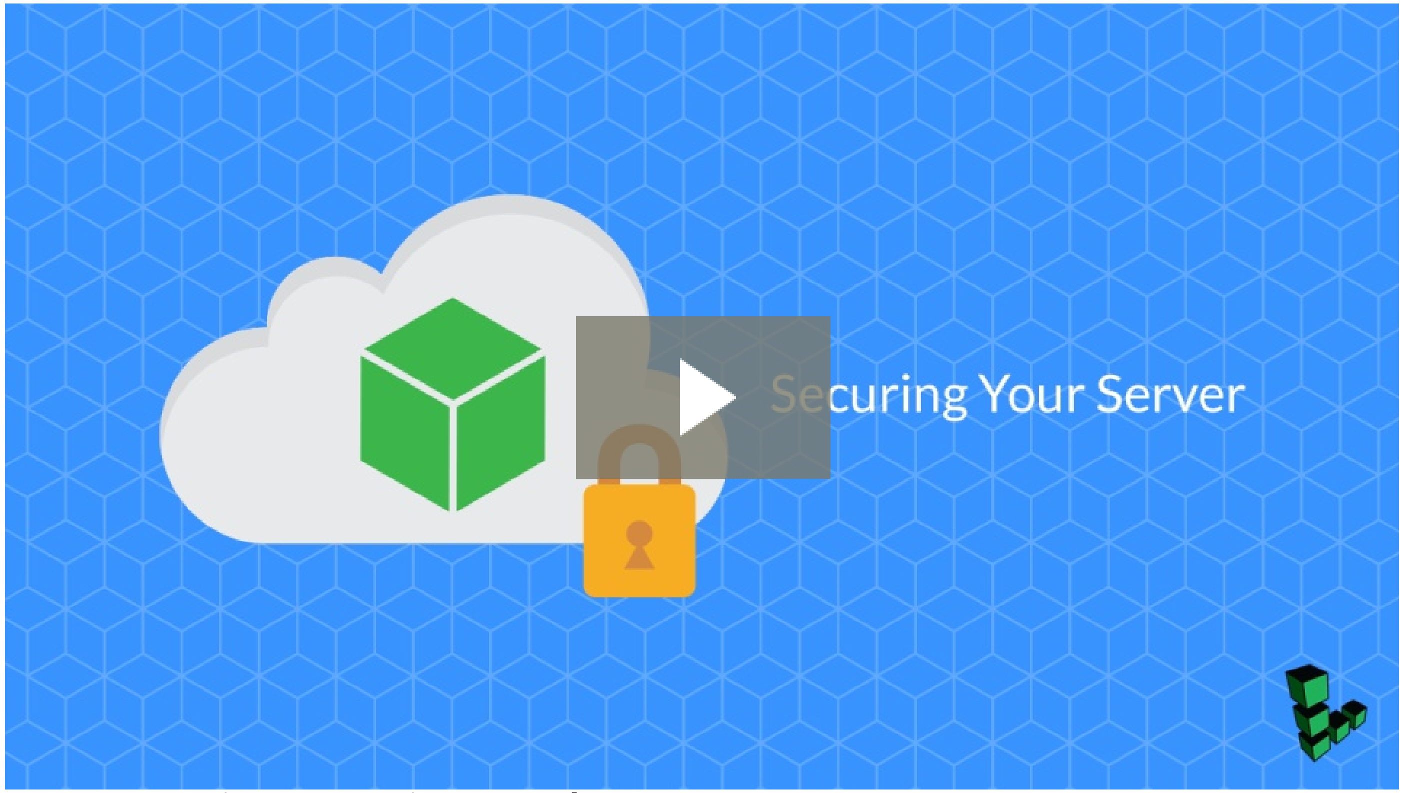
Updated Tuesday, May 28, 2019 by Jared Kobos

Written by Linode

Contribute on GitHub Report an Issue | View File | Edit File

In the Getting Started guide, you learned how to deploy a Linux distribution, boot your Linode and perform basic administrative tasks. Now it’s time to harden your Linode against unauthorized access.

RSS feed



Automatic Security Updates

There are arguments for and against automatic updates on servers. Fedora’s Wiki has a good breakdown of the pros and cons, but the risk of automatic updates will be minimal if you limit them to security updates. Not all package managers make that easy or possible, though.

The practicality of automatic updates is something you must judge for yourself because it comes down to what *you* do with your Linode. Bear in mind that automatic updates apply only to packages sourced from repositories, not self-compiled applications. You may find it worthwhile

Monthly Guides Update

Email address

Sign Up

to have a test environment that replicates your production server. Updates can be applied there and reviewed for issues before being applied to the live environment.

- CentOS uses *yum-cron* for automatic updates.
- Debian and Ubuntu use *unattended upgrades*.
- Fedora uses *dnf-automatic*.

Add a Limited User Account

Up to this point, you have accessed your Linode as the `root` user, which has unlimited privileges and can execute *any* command–even one that could accidentally disrupt your server. We recommend creating a limited user account and using that at all times. Administrative tasks will be done using `sudo` to temporarily elevate your limited user's privileges so you can administer your server.

Note
Not all Linux distributions include `sudo` on the system by default, but all the images provided by Linode have sudo in their package repositories. If you get the output `sudo: command not found`, install sudo before continuing.

To add a new user, first log in to your Linode via SSH.

CentOS / Fedora

1. Create the user, replacing `example_user` with your desired username, and assign a password:

   ```
   useradd example_user && passwd example_user
   ```

2. Add the user to the `wheel` group for sudo privileges:

   ```
   usermod -aG wheel example_user
   ```

   > **Caution**
   > In CentOS 6 a wheel group is disabled by default for sudo access. You must to configure it manually. Type from root: `/usr/sbin/visudo`. Then find the line `# %wheel` and uncomment this line. To began typing in vi, press `a`. To save and exit press `Escape`, then type `:w` (press enter), `:q` (press enter)

Ubuntu

1. Create the user, replacing `example_user` with your desired username. You'll then be asked to assign the user a password:

   ```
   adduser example_user
   ```

2. Add the user to the `sudo` group so you'll have administrative privileges:

```
adduser example_user sudo
```

Debian

A standard Debian Server installation does not include `sudo` by default, but Linode packages it in our Debian images. If you don’t already have `sudo`, you’ll need to install it before going further.

1. Create the user, replacing `example_user` with your desired username. You’ll then be asked to assign the user a password:

   ```
   adduser example_user
   ```

2. Add the user to the `sudo` group so you’ll have administrative privileges:

   ```
   adduser example_user sudo
   ```

3. After creating your limited user, disconnect from your Linode:

   ```
   exit
   ```

4. Log back in as your new user. Replace `example_user` with your username, and the example IP address with your Linode’s IP address:

   ```
   ssh example_user@203.0.113.10
   ```

Now you can administer your Linode from your new user account instead of `root`. Nearly all superuser commands can be executed with `sudo` (example: `sudo iptables -L -nv`) and those commands will be logged to `/var/log/auth.log`.

Harden SSH Access

By default, password authentication is used to connect to your Linode via SSH. A cryptographic key-pair is more secure because a private key takes the place of a password, which is generally much more difficult to brute-force. In this section we'll create a key-pair and configure the Linode to not accept passwords for SSH logins.

Create an Authentication Key-pair

1. This is done on your local computer, **not** your Linode, and will create a 4096-bit RSA key-pair. During creation, you will be given the option to encrypt the private key with a passphrase. This means that it cannot be used without entering the passphrase, unless you save it to your local desktop's keychain manager. We suggest you use the key-pair with a passphrase, but you can leave this field blank if you don't want to use one.

   Linux / OS X

   > **Caution**

> If you’ve already created an RSA key-pair, this command will overwrite it, potentially locking you out of other systems. If you’ve already created a key-pair, skip this step. To check for existing keys, run `ls ~/.ssh/id_rsa*`.

```
ssh-keygen -b 4096
```

Press **Enter** to use the default names `id_rsa` and `id_rsa.pub` in `/home/your_username/.ssh` before entering your passphrase.

Windows

This can be done using PuTTY as outlined in our guide: Use Public Key Authentication with SSH.

2. Upload the public key to your Linode. Replace `example_user` with the name of the user you plan to administer the server as, and `203.0.113.10` with your Linode’s IP address.

Linux

From your local computer:

```
ssh-copy-id example_user@203.0.113.10
```

OS X

On your Linode (while signed in as your limited user):

```
mkdir -p ~/.ssh && sudo chmod -R 700 ~/.ssh/
```

From your local computer:

```
scp ~/.ssh/id_rsa.pub example_user@203.0.113.10:~/.ssh/authorized_k
```

Note
`ssh-copy-id` is available in Homebrew if you prefer it over SCP. Install with `brew install ssh-copy-id`.

Windows

- **Option 1**: This can be done using WinSCP. In the login window, enter your Linode's public IP address as the hostname, and your non-root username and password. Click *Login* to connect.

  Once WinSCP has connected, you'll see two main sections. The section on the left shows files on your local computer and the section on the right shows files on your Linode. Using the file explorer on the left, navigate to the file where you've saved your public key, select the public key file, and click *Upload* in the toolbar above.

  You'll be prompted to enter a path where you'd like to place the file on your Linode. Upload the file to

`/home/example_user/.ssh/authorized_keys`, replacing `example_user` with your username.

- **Option 2:** Copy the public key directly from the PuTTY key generator into the terminal emulator connected to your Linode (as a non-root user):

  ```
  mkdir ~/.ssh; nano ~/.ssh/authorized_keys
  ```

  The above command will open a blank file called `authorized_keys` in a text editor. Copy the public key into the text file, making sure it is copied as a single line exactly as it was generated by PuTTY. Press **CTRL+X**, then **Y**, then **Enter** to save the file.

Finally, you'll want to set permissions for the public key directory and the key file itself:

```
sudo chmod -R 700 ~/.ssh && chmod 600 ~/.ssh/authorized_keys
```

These commands provide an extra layer of security by preventing other users from accessing the public key directory as well as the file itself. For more information on how this works, see our guide on how to modify file permissions.

3. Now exit and log back into your Linode. If you specified a passphrase for your private key, you'll need to enter it.

SSH Daemon Options

1. **Disallow root logins over SSH.** This requires all SSH connections be by non-root users. Once a limited user account is connected, administrative privileges are accessible either by using `sudo` or changing to a root shell using `su -`.

   /etc/ssh/sshd_config

   ```
   # Authentication:
   ...
   PermitRootLogin no
   ```

2. **Disable SSH password authentication.** This requires all users connecting via SSH to use key authentication. Depending on the Linux distribution, the line `PasswordAuthentication` may need to be added, or uncommented by removing the leading `#`.

   /etc/ssh/sshd_config

   ```
   # Change to no to disable tunnelled clear text passwords
   PasswordAuthentication no
   ```

   Note
   You may want to leave password authentication enabled if you connect to your Linode from many different computers. This will allow you to authenticate with a password instead of generating and uploading a key-pair for every device.

3. **Listen on only one internet protocol.** The SSH daemon listens for incoming connections over both IPv4 and IPv6 by default. Unless you need to SSH into your Linode using both protocols, disable whichever you do not need. *This does not disable the protocol system-wide, it is only for the SSH daemon.*

   Use the option:

   - `AddressFamily inet` to listen only on IPv4.
   - `AddressFamily inet6` to listen only on IPv6.

   The `AddressFamily` option is usually not in the `sshd_config` file by default. Add it to the end of the file:

   ```
   echo 'AddressFamily inet' | sudo tee -a /etc/ssh/sshd_config
   ```

4. Restart the SSH service to load the new configuration.

   If you're using a Linux distribution which uses systemd (CentOS 7, Debian 8, Fedora, Ubuntu 15.10+)

   ```
   sudo systemctl restart sshd
   ```

   If your init system is SystemV or Upstart (CentOS 6, Debian 7, Ubuntu 14.04):

   ```
   sudo service sshd restart
   ```

Use Fail2Ban for SSH Login Protection

Fail2Ban is an application that bans IP addresses from logging into your server after too many failed login attempts. Since legitimate logins usually take no more than three tries to succeed (and with SSH keys, no more than one), a server being spammed with unsuccessful logins indicates attempted malicious access.

Fail2Ban can monitor a variety of protocols including SSH, HTTP, and SMTP. By default, Fail2Ban monitors SSH only, and is a helpful security deterrent for any server since the SSH daemon is usually configured to run constantly and listen for connections from any remote IP address.

For complete instructions on installing and configuring Fail2Ban, see our guide: Securing Your Server with Fail2ban.

Remove Unused Network-Facing Services

Most Linux distributions install with running network services which listen for incoming connections from the internet, the loopback interface, or a combination of both. Network-facing services which are not needed should be removed from the system to reduce the attack surface of both running processes and installed packages.

Determine Running Services

To see your Linode’s running network services:

```
sudo ss -atpu
```

The following is an example of the output given by `ss`, and shows that the SSH daemon (sshd) is listening and connected. Note that because distributions run different services by default, your output will differ.

```
Netid State Recv-Q Send-Q Local Address:Port Peer Address:Port
tcp LISTEN 0 128 *:ssh *:* users:(("sshd",pid=3675,fd=3))
tcp ESTAB 0 208 203.0.113.1:ssh 198.51.100.2:54820 users:(("sshd",pid=
3698,fd=3))
tcp LISTEN 0 128 :::ssh :::* users:(("sshd",pid=3675,fd=4))
```

TCP

See the **Peer Address:Port** column of the `ss` readout. The process `sshd` is listening on `*:*`, which translates into any incoming IPv4 address to any port, and over any network interface. The next line shows an established SSH connection from IP address 198.51.100.2 via ephemeral port 54820. The last line, `:::*` denotes the `sshd` process listening for any incoming SSH connections over IPv6 to any port, and again over any network interface.

UDP

UDP sockets are *stateless*, meaning they are either open or closed and every process's connection is independent of those which occurred before and after. This is in contrast to TCP connection states such as *LISTEN*, *ESTABLISHED* and *CLOSE_WAIT*. The `ss` output above shows no UDP connections.

Determine Which Services to Remove

A basic TCP and UDP nmap scan of your Linode without a firewall enabled would show SSH and possibly other services listening for incoming connections. By configuring a firewall you can filter those ports to your requirements. Ideally, the unused services should be disabled.

You will likely be administering your server primarily through an SSH connection, so that service needs to stay. As mentioned above, RSA keys and Fail2Ban can help protect SSH. System services like `chronyd`, `systemd-resolved`, and `dnsmasq` are usually listening on localhost and only occasionally contacting the outside world. Services like this are part of your operating system and will cause problems if removed and not properly substituted.

However, some services are unnecessary and should be removed unless you have a specific need for them. Some examples could be Exim, Apache and RPC.

Uninstall the Listening Services

How to remove the offending packages will differ depending on your distribution's package manager.

Arch

```
sudo pacman -Rs package_name
```

CentOS

```
sudo yum remove package_name
```

Debian / Ubuntu

```
sudo apt purge package_name
```

Fedora

```
sudo dnf remove package_name
```

Run `ss -atup` again to verify that the unwanted services are no longer running.

Configure a Firewall

Using a *firewall* to block unwanted inbound traffic to your Linode provides a highly effective security layer. By being very specific about the traffic you allow in, you can prevent intrusions and network mapping. A best practice is to allow only the traffic you need, and deny everything else. See our documentation on some of the most common firewall applications:

- Iptables is the controller for netfilter, the Linux kernel's packet filtering framework. Iptables is included in most Linux distributions by default.
- FirewallD is the iptables controller available for the CentOS / Fedora family of distributions.
- UFW provides an iptables frontend for Debian and Ubuntu.

Common Lockout Recovery Steps

If for whatever reason you find yourself locked out of your Linode after putting your security controls into place, there are still a number of ways that you can regain access to your Linode.

- Access your Linode through our out-of-band Lish console to regain access to the internals of your Linode without relying on SSH.
- If you need to re-enable password authentication and/or root login over ssh to your Linode, you can do this by reversing the following sections of this file to reflect these changes

```
/etc/ssh/sshd_config
```

```
# Authentication:
...
PermitRootLogin yes
...
PasswordAuthentication yes
```

From there, you just need to restart SSH.

If you're using a Linux distribution which uses systemd (CentOS 7, Debian 8, Fedora, Ubuntu 15.10+)

```
sudo systemctl restart sshd
```

If your init system is SystemV or Upstart (CentOS 6, Debian 7, Ubuntu 14.04):

```
sudo service sshd restart
```

- If you need to remove your public key from your Linode, you can enter the following command:

```
rm ~/.ssh/authorized_keys
```

You can then replace your key by re-following the Create an Authentication Key-pair section of this guide.

Next Steps

These are the most basic steps to harden any Linux server, but further security layers will depend on its intended use. Additional techniques can include application configurations, using intrusion detection or installing a form of access control.

Now you can begin setting up your Linode for any purpose you choose. We have a library of documentation to assist you with a variety of topics ranging from migration from shared hosting to enabling two-factor authentication to hosting a website.

Join our Community

Find answers, ask questions, and help others.

comments powered by Disqus

Write for Linode.

We're always expanding our docs. If you like to help people, can write, and have expertise in a Linux or cloud infrastructure topic, learn how you can contribute to our library.

Get started in the Linode Cloud today.

Create an Account