

# How to Use Two-Factor Authentication with Sudo and SSH on Linux with Google Authenticator

Learn how to enhance Linux security by implementing two-factor authentication with Google Authenticator for both Sudo and SSH access on your Linux system.

# Contents

01	Introduction	3
02	Prerequisites	3
03	1. Install the Google Authenticator PAM Module	3
04	2. Generate Your 2FA Code	4
05	3. Configure SSH	5
06	4. Configure 2FA for Sudo	7
07	Recover From a 2FA Lockout	8
08	More Information	9

# Introduction

---

Two-factor authentication, or 2FA, confirms a user's identity via two different factors: something they know and something they have. Also known as multi-factor authentication, two-step verification, and two-step authentication, 2FA enhances security. This guide explains how to use two-factor authentication with sudo and SSH on Linux with Google Authenticator by configuring the Google Authenticator PAM module.

## Prerequisites

---

Before you proceed with this guide, you need a deployed Linux cloud server. We recommend the following best practices:

- Create a sudo user
- [Update the server](#)
- [Configure a firewall](#)
- Configure SSH to allow access [with an SSH key](#).

You will also need a two-factor authenticator app. Popular choices are Google Authenticator, Microsoft Authenticator, Authy, and FreeOTP.

## 1. Install the Google Authenticator PAM Module

---

Log in to your Linux server as a non-root user with sudo access. Follow the instructions for your Linux distribution to install the Google Authenticator PAM module and configure SSH Google Authenticator for enhanced two-factor authentication security.

### Ubuntu, Debian, and other apt-based Distributions

Install with apt.

```
$ sudo apt-get install libpam-google-authenticator
```

## Fedora

Install with yum.

```
$ sudo yum install google-authenticator
```

## CentOS, Rocky Linux, Alma Linux, VzLinux, and other RHEL Clones

Install the EPEL Repository.

```
$ sudo yum install epel-release
```

Install with yum.

```
$ sudo yum install google-authenticator
```

## 2. Generate Your 2FA Code

Note: Each user connecting to the server will perform these steps.

Run the Google Authenticator setup program. You can run the program without command-line options for an interactive setup, or use the following options:

```
$ google-authenticator -t -f -d -w 3 -e 10 -r 3 -R 30
```

These options explained:

- -t : Use TOTP verification
- -f : Write the configuration to `~/.google_authenticator`
- -d : Do not allow reuse of previously used tokens.

- `-w 3` : The window size of allowed tokens. By default, tokens expire every 30 seconds. A window size of 3 allows authentication with the token before and after the current token for clock skew.
- `-e 10` : Generate 10 emergency backup codes
- `-r 3 -R 30` : Rate-limit. Allow 3 logins every 30 seconds.

Use `--help` for more options.

The program will update your configuration files and display several values:

- A QR code. You can scan this code with most authenticator apps.
- A secret key. Enter this key in your app if you cannot scan the QR code.
- The initial verification code, which will expire in 30 seconds.
- A list of 10 one-time use emergency codes.

Important: Follow the instructions in your 2FA application to create a new entry with the QR code or secret key. Store your emergency codes in a secure location. If you need to reset your code, rerun the program.

## 3. Configure SSH

These steps disable password authentication. Public/private SSH keys are required for login, and 2FA will be enabled.

1. Edit your SSH PAM configuration file.

```
$ sudo nano /etc/pam.d/sshd
```

2. Add the following line to the bottom of the file. The `nullok` option allows users that have not yet generated a 2FA code to log in, while codes are required if the user has followed Step 2 above. This option is useful during rollout. After all users have generated codes, you can remove the `nullok` option to require 2FA for everyone.

```
auth required pam_google_authenticator.so nullok
```

3. Disable user password authentication. Comment out the following line by adding `#` to the beginning.

```
# @include common-auth
```

4. Save and close the file.
5. Edit the SSH daemon configuration file.

```
$ sudo nano /etc/ssh/sshd_config
```

6. Find the line for `ChallengeResponseAuthentication` and set its value to `yes`.

```
ChallengeResponseAuthentication yes
```

7. Verify the following options are set as shown, or add them if they don't exist.

```
PasswordAuthentication no  
PubkeyAuthentication yes  
KbdInteractiveAuthentication yes  
AuthenticationMethods publickey,keyboard-interactive
```

8. Save and close the file.
9. Restart the SSH service daemon.

```
$ sudo systemctl restart ssh
```

10. Log out of the server.
11. Log in to the server with your SSH key. The server will request a verification code. Enter the code generated by your authenticator app.

Recovery tip: If there is an error in the SSH configuration and you are unable to log in, use the Vultr web console.

## 4. Configure 2FA for Sudo

Configure `sudo` to require 2FA codes by following these steps.

1. Edit `/etc/pam.d/common-auth`.

```
$ sudo nano /etc/pam.d/common-auth
```

2. Add these lines to the bottom of the file. The `nullok` option allows users that have not yet generated a 2FA code to use sudo, while codes are required if the user has followed Step 2 above. This option is useful during rollout. After all users have generated codes, you can remove the `nullok` option to require 2FA for everyone. See [this note for more information](#).

```
auth required pam_google_authenticator.so nullok  
auth required pam_permit.so
```

3. Save and exit the file.

The 2FA option takes effect immediately. If a user has configured 2FA in Step 2 above, `sudo` will require 2FA codes in addition to the user password. Once all users have configured 2FA, you can remove the `nullok` option.

Note: Users that have NOPASSWD set in the sudoers file will not receive the 2FA challenge.

# Recover From a 2FA Lockout

## Emergency Backup

If you lose access to your authenticator app, use one of your emergency backup codes. The codes are for one-time use only.

## SSH Lockout

If you are locked out from SSH, you can use the Vultr web console. The configuration in this guide does not require 2FA for console access.

## Disable 2FA for a specific user

To disable 2FA for a specific user:

1. Log in as root.
2. Delete the `.google_authenticator` file in the user's home directory.

```
$ sudo rm /home/user/.google_authenticator
```

## Disable 2FA for all users

To disable 2FA for all users:

1. Edit `/etc/ssh/sshd_config`.
2. Locate the following line:

```
AuthenticationMethods publickey,keyboard-interactive
```

3. Remove the `keyboard-interactive` method:

```
AuthenticationMethods publickey
```

4. Save and close the file, then restart the SSH daemon:

```
$ sudo systemctl restart ssh
```

## More Information

---

Links to download popular client apps:

- [Google Authenticator](#)
- [Microsoft Authenticator](#)
- [Authy](#)
- [FreeOTP](#)

Learn more about the [Google Authenticator PAM module](#) on GitHub.



VULTR

