

# How to Install Caddy Webserver on Debian 12

Learn how to install Caddy webserver on Debian 12 with our step-by-step guide. Get your web server up and running quickly with this modern, efficient solution.

---

# Contents

01	Introduction	3
02	Prerequisites	3
03	Install Caddy	3
04	Manage the Caddy System Service	6
05	Create a New Caddy Virtual Host	7
06	Secure the Caddy Webserver	9
07	Set Up Firewall Rules	10
08	Conclusion	12

# Introduction

Caddy is an open-source webserver that enables the deliver of dynamic web applications with automatic TLS/SSL termination. Caddy can work as a webserver, reverse proxy, load balancer, or API gateway to securely deliver applications or services using site block configurations.

This article explains how to install the Caddy webserver on Debian 12.

## Prerequisites

Before you begin:

- Deploy a [Debian 12 server](#) on Vultr.
- Create a [domain name A record pointing to the server IP address](#).
- Securely [access the server using SSH](#) as a [non-root sudo user](#).
- [Update the server](#).

## Install Caddy

Caddy is not available in the Debian 12 package repositories by default but can be installed from source files or through the latest repository information on your server. Follow the steps below to download the latest Caddy repository information and install the application on your server.

1. Update the server's package index.

```
CONSOLE
```

```
$ sudo apt update
```

2. Install all necessary dependency packages.

```
CONSOLE
```

```
$ sudo apt install -y debian-keyring debian-archive-keyring  
apt-transport-https curl
```

### 3. Import the Caddy repository GPG key.

CONSOLE

```
$ curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/  
gpg.key' | sudo gpg --dearmor -o /usr/share/keyrings/caddy-  
stable-archive-keyring.gpg
```

### 4. Add the Caddy repository to your APT sources.

CONSOLE

```
$ curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/  
debian.deb.txt' | sudo tee /etc/apt/sources.list.d/caddy-  
stable.list
```

### 5. Update the server's package index again.

CONSOLE

```
$ sudo apt update
```

### 6. Install Caddy.

CONSOLE

```
$ sudo apt install caddy
```

### 7. View the installed Caddy version on your server.

CONSOLE

```
$ caddy version
```

You output should be similar to the one below.

```
v2.8.4 h1:q3pe0wpBj10cHFZ3n/1nL4V4bxBrYoSoab7rL9BMYNk=
```

8. Allow incoming connections to the HTTP port `80` through the firewall.

CONSOLE

```
$ sudo ufw allow 80
```

9. Restart the firewall to apply changes.

CONSOLE

```
$ sudo ufw reload
```

10. Access your Server IP using a web browser such as Chrome and verify that Caddy delivers the default web application page.

```
http://SERVER-IP
```



# Manage the Caddy System Service

Follow the steps below to manage Caddy as a system service on your server.

1. Enable the Caddy system service to automatically start at boot.

CONSOLE

```
$ sudo systemctl enable caddy
```

2. Start the Caddy service.

CONSOLE

```
$ sudo systemctl start caddy
```

3. View the Caddy service status and verify that it's running.

CONSOLE

```
$ sudo systemctl status caddy
```

Your output should look like the one below:

```
● caddy.service - Caddy
   Loaded: loaded (/lib/systemd/system/caddy.service; enabled; preset: enabled)
   Active: active (running) since Mon 2024-07-22 16:59:15 UTC; 2h 54min ago
     Docs: https://caddyserver.com/docs/
  Main PID: 7579 (caddy)
    Tasks: 7 (limit: 4637)
   Memory: 18.7M
      CPU: 202ms
   CGroup: /system.slice/caddy.service
           └─7579 /usr/bin/caddy run --environ --config /etc/caddy/Caddyfile
```

# Create a New Caddy Virtual Host

Caddy stores configuration files in the `/etc/caddy` directory by default and supports Caddyfile configurations from any location on your server. Follow the steps below to create a new Caddy virtual host to serve web application files from the `/var/www/example.com` directory on your server.

1. Create a new `/var/www/example.com` web application files directory.

CONSOLE

```
$ sudo mkdir -p /var/www/example.com
```

2. Create a new HTML application file `index.html`.

CONSOLE

```
$ sudo nano /var/www/example.com/index.html
```

3. Add the following contents to the `index.html` file.

HTML

```
<html>
<head>
  <title>Greetings from Vultr!</title>
</head>
<body>
  <h1 align="center">Hello world!, Greetings from Vultr</
h1>
  <h1 align="center">Success! The Caddy server is working!
</h1>
</body>
</html>
```

Save and close the file.

4. Switch to the Caddy configuration files directory.

CONSOLE

```
$ cd /etc/caddy/
```

5. Back up the default Caddyfile configuration.

CONSOLE

```
$ sudo mv Caddyfile Caddyfile.default
```

6. Create a new Caddyfile configuration.

CONSOLE

```
$ sudo nano Caddyfile
```

7. Add the following configurations to the file. Replace `example.com` with your actual domain.

INI

```
example.com {  
    tls admin@example.com  
    root * /var/www/example.com  
    file_server {  
        index index.html  
    }  
    log {  
        output file /var/log/caddy/example.log  
        format console  
    }  
}
```

Save and exit the file.

The above Caddy configuration creates a new virtual host and delivers web application files using your domain `example.com`. Within the configuration:

- `example.com`: Defines a new virtual host entry for your domain or IP address.

- `tls`: Specifies the email address to associate with Let's Encrypt SSL certificate generation requests.
- `root`: Specifies the directory where your web application files are stored.
- `file_server`: Activates the file server for your web application. Within the directive, `index` specifies the default file to serve when your domain is accessed.
- `log`: Enables logging of access and error details to a specific file such as `/var/log/caddy/example.log`.

## 8. Test the Caddy configuration for errors.

### CONSOLE

```
$ caddy validate
```

### Output:

```
2024/07/25 17:21:59.158 INFO    http.auto_https enabling automatic HTTP->HTTPS
redirects      {"server_name": "srv0"}
2024/07/25 17:21:59.158 INFO    tls.cache.maintenance  started background
certificate maintenance {"cache": "0xc0000ef200"}
2024/07/25 17:21:59.158 INFO    tls.cache.maintenance  stopped background
certificate maintenance {"cache": "0xc0000ef200"}
Valid configuration
```

## 9. Reload the Caddy service to apply the configuration changes.

### CONSOLE

```
$ sudo systemctl reload caddy
```

# Secure the Caddy Webserver

Caddy automatically generates and renews trusted Let's Encrypt SSL certificates for all site blocks with domain values. Follow the steps below to

secure access to the Caddyfile configuration and allow network connections to all necessary network connection ports.

1. Grant the Caddy user full privileges to the `/etc/caddy` directory.

CONSOLE

```
$ sudo chown -R caddy:caddy /etc/caddy
```

2. Grant the Caddy user full permissions to the Caddyfile while disabling access for other system users.

CONSOLE

```
$ sudo chmod 660 /etc/caddy/Caddyfile
```

3. Long list the `/etc/caddy` directory to verify the permission changes.

CONSOLE

```
$ ls -l /etc/caddy/
```

Output:

```
total 8
-rw-rw---- 1 caddy caddy 244 Jul 25 17:18 Caddyfile
-rw-r--r-- 1 caddy caddy 769 Jun  2 12:07 Caddyfile.default
```

## Set Up Firewall Rules

Caddy uses the HTTP port `80` and HTTPS port `443` to serve web application files on your server. Follow the steps below to configure the default firewall to allow connections to all network ports required by the Caddy webserver.

1. View the UFW status and verify that it's active.

CONSOLE

```
$ sudo ufw status
```

If the status is `inactive`, allow the SSH port `22` and enable UFW.

CONSOLE

```
$ sudo ufw allow 22 && sudo ufw enable
```

2. Allow incoming connections to the HTTPS port `443`.

CONSOLE

```
$ sudo ufw allow 443
```

3. Reload the firewall to apply changes.

CONSOLE

```
$ sudo ufw reload
```

4. View the firewall status.

CONSOLE

```
$ sudo ufw status
```

5. Access your domain using a browser such as Chrome to verify that Caddy serves your virtual host's web application files.

```
https://example.com
```



**Hello World!, Greetings from Vultr**  
**Success! The Caddy server is working!**

## Conclusion

---

You have installed Caddy on a Debian 12 server and managed the application service to securely deliver web applications. Caddy can work as a webserver or a reverse proxy to securely deliver applications or services. For more information and configuration options, visit the [Caddy documentation](#).



VULTR

