



TUTORIAL

How To Set Up a Firewall with UFW on Ubuntu 18.04

Ubuntu Security DigitalOcean Networking Firewall Ubuntu 18.04

By [Brian Boucheron](#) and [Erika Heide](#)

Last Validated on July 29, 2021 • Originally Published on July 5, 2018 855.8k

English

Not using Ubuntu 18.04?

Choose a different version or distribution.

Ubuntu 18.04

Introduction

UFW, or Uncomplicated Firewall, is an interface to `iptables` that is geared towards simplifying the process of configuring a firewall. While `iptables` is a solid and flexible tool, it can be difficult for beginners to learn how to use it to properly configure a firewall. If you're looking to get started securing your network, and you're not sure which tool to use, UFW may be the right choice for you.

This tutorial will show you how to set up a firewall with UFW on Ubuntu 18.04.

Prerequisites

To follow this tutorial, you will need:

- One Ubuntu 18.04 server with a `sudo` non-root user, which you can set up by following Steps 1–3 in the [Initial Server Setup with Ubuntu 18.04](#) tutorial.

UFW is installed by default on Ubuntu. If it has been uninstalled for some reason, you can install it with `sudo apt install ufw`.

Step 1 — Making Sure IPv6 is Enabled

In recent versions of Ubuntu, IPv6 is enabled by default. In practice that means most firewall rules added to the server will include both an IPv4 and an IPv6 version, the latter identified by `v6` within the output of UFW's status command. To make sure IPv6 is enabled, you can check your UFW configuration file at `/etc/default/ufw`. Open this file using `nano` or your favorite command line editor:

```
$ sudo nano /etc/default/ufw
```

Then make sure the value of `IPv6` is set to `yes`. It should look like this:

/etc/default/ufw excerpt

```
IPv6=yes
```

Save and close the file. If you're using `nano`, you can do that by typing `CTRL+X`, then `Y` and `ENTER` to confirm.

When UFW is enabled in a later step of this guide, it will be configured to write both IPv4 and IPv6 firewall rules.

Step 2 — Setting Up Default Policies

If you're just getting started with UFW, a good first step is to check your default firewall policies. These rules control how to handle traffic that does not explicitly match any other rules.

By default, UFW is set to deny all incoming connections and allow all outgoing connections. This means anyone trying to reach your server would not be able to connect, while any application

Sign up for our newsletter Get the latest tutorials on SysAdmin and open source topics.



Sign Up

To set the default UFW incoming policy to `deny`, run:

```
$ sudo ufw default deny incoming
```

Output

```
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
```

To set the default UFW outgoing policy to `allow`, run:

```
$ sudo ufw default allow outgoing
```

Output

```
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
```

These commands set the defaults to deny incoming and allow outgoing connections. These firewall defaults alone might suffice for a personal computer, but servers typically need to respond to incoming requests from outside users. We'll look into that next.

Step 3 — Allowing SSH Connections

If you were to enable your UFW firewall now, it would deny all incoming connections. This means that you'll need to create rules that explicitly allow legitimate incoming connections — SSH or HTTP connections, for example — if you want your server to respond to those types of requests. If you're using a cloud server, you will probably want to allow incoming SSH connections so you can connect to and manage your server.

Allowing the OpenSSH UFW Application Profile

Upon installation, most applications that rely on network connections will register an application profile within UFW, which enables users to quickly allow or deny external access to a service. You can check which profiles are currently registered in UFW with:

```
$ sudo ufw app list
```

Output

```
Available applications:
  OpenSSH
```

To enable the OpenSSH application profile, run:

```
$ sudo ufw allow OpenSSH
```

Output

```
Rule added
Rule added (v6)
```

This will create firewall rules to allow all connections on port `22`, which is the port that the SSH daemon listens on by default.

Allowing SSH by Service Name

Another way to configure UFW to allow incoming SSH connections is by referencing its service name: `ssh`.

SCROLL TO TOP

```
$ sudo ufw allow ssh
```

Sign up for our newsletter Get the latest tutorials on SysAdmin and open source topics.



Sign Up

Allowing SSH by Port Number

Alternatively, you can write the equivalent rule by specifying the port instead of the application profile or service name. For example, this command works the same as the previous examples:

```
$ sudo ufw allow 22
```

Output

Rule added

Rule added (v6)

If you configured your SSH daemon to use a different port, you will have to specify the appropriate port. For example, if your SSH server is listening on port `2222`, you can use this command to allow connections on that port:

```
$ sudo ufw allow 2222
```

Output

Rule added

Rule added (v6)

Now that your firewall is configured to allow incoming SSH connections, you can enable it.

Step 4 — Enabling UFW

Your firewall should now be configured to allow SSH connections. To verify which rules were added so far, even when the firewall is still disabled, you can use:

```
$ sudo ufw show added
```

Output

Added user rules (see 'ufw status' for running firewall):
ufw allow OpenSSH

After confirming you have a rule to allow incoming SSH connections, you can enable the firewall with:

```
$ sudo ufw enable
```

Output

Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup

You will receive a warning that says the command may disrupt existing SSH connections. You already set up a firewall rule that allows SSH connections, so it should be fine to continue. Respond to the prompt with `y` and hit `ENTER`.

The firewall is now active. Run the `sudo ufw status verbose` command to see the rules that are set. The rest of this tutorial covers how to use UFW in more detail, like allowing or denying different kinds of connections.

Step 5 — Allowing Other Connections

SCROLL TO TOP

At this point, you should allow all of the other connections that your server needs to respond to.
The connections that you should allow depend on your specific needs. You already know how to

Sign up for our newsletter Get the latest tutorials on SysAdmin and open source topics.



Enter your email address

Sign Up

or `sudo ufw allow 443`

- Apache with both HTTP and HTTPS, using `sudo ufw allow 'Apache Full'`
- Nginx with both HTTP and HTTPS, using `sudo ufw allow 'Nginx Full'`

Don't forget to check which application profiles are available for your server with `sudo ufw app list`.

There are several other ways to allow connections, aside from specifying a port or known service name. We'll see some of these next.

Specific Port Ranges

You can specify port ranges with UFW. Some applications use multiple ports, instead of a single port.

For example, to allow X11 connections, which use ports `6000 - 6007`, use these commands:

```
$ sudo ufw allow 6000:6007/tcp
$ sudo ufw allow 6000:6007/udp
```

When specifying port ranges with UFW, you must specify the protocol (`tcp` or `udp`) that the rules should apply to. We haven't mentioned this before because not specifying the protocol automatically allows both protocols, which is OK in most cases.

Specific IP Addresses

When working with UFW, you can also specify IP addresses within your rules. For example, if you want to allow connections from a specific IP address, such as a work or home IP address of `203.0.113.4`, you need to use the `from` parameter, providing then the IP address you want to allow:

```
$ sudo ufw allow from 203.0.113.4
```

Output
Rule added

You can also specify a port that the IP address is allowed to connect to by adding `to any port` followed by the port number. For example, If you want to allow `203.0.113.4` to connect to port `22` (SSH), use this command:

```
$ sudo ufw allow from 203.0.113.4 to any port 22
```

Output
Rule added

Subnets

If you want to allow a subnet of IP addresses, you can do so using CIDR notation to specify a netmask. For example, if you want to allow all of the IP addresses ranging from `203.0.113.1` to `203.0.113.254` you could use this command:

```
$ sudo ufw allow from 203.0.113.0/24
```

Output
Rule added

SCROLL TO TOP

Likewise, you may also specify the destination port that the subnet `203.0.113.0/24` is allowed to connect to. Again, we'll use port `22` (SSH) as an example:

Sign up for our newsletter Get the latest tutorials on SysAdmin and open source topics.



Enter your email address

Sign Up

Connections to a Specific Network Interface

If you want to create a firewall rule that only applies to a specific network interface, you can do so by specifying “allow in on” followed by the name of the network interface.

You may want to look up your network interfaces before continuing. To do so, use this command:

```
$ ip addr
```

Output Excerpt

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
. . .
3: eth1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default
. . .
```

The highlighted output indicates the network interface names. They are typically named something like `eth0` or `enp3s2`.

So, if your server has a public network interface called `eth0`, you could allow HTTP traffic (port `80`) to it with this command:

```
$ sudo ufw allow in on eth0 to any port 80
```

Output

```
Rule added
Rule added (v6)
```

Doing so would allow your server to receive HTTP requests from the public internet.

Or, if you want your MySQL database server (port `3306`) to listen for connections on the private network interface `eth1`, for example, you could use this command:

```
$ sudo ufw allow in on eth1 to any port 3306
```

Output

```
Rule added
Rule added (v6)
```

This would allow other servers on your private network to connect to your MySQL database.

Step 6 — Denying Connections

If you haven't changed the default policy for incoming connections, UFW is configured to deny all incoming connections. Generally, this simplifies the process of creating a secure firewall policy by requiring you to create rules that explicitly allow specific ports and IP addresses through.

However, sometimes you will want to deny specific connections based on the source IP address or subnet, perhaps because you know that your server is being attacked from there. Also, if you want to change your default incoming policy to **allow** (which is not recommended), you would need to create **deny** rules for any services or IP addresses that you don't want to allow connections for.

To write **deny** rules, you can use the commands previously described, replacing **allow** with **deny**.

For example, to deny HTTP connections, you could use this command:

SCROLL TO TOP

```
$ sudo ufw deny http
```

Sign up for our newsletter Get the latest tutorials on SysAdmin and open source topics.



Sign Up

```
$ sudo ufw deny from 203.0.113.4
```

Output

```
Rule added
```

In some cases, you may also want to block outgoing connections from the server. To deny all users from using a port on the server, such as port 25 for SMTP traffic, you can use `deny out` followed by the port number:

```
$ sudo ufw deny out 25
```

Output

```
Rule added
```

```
Rule added (v6)
```

This will block all outgoing SMTP traffic on the server.

Step 7 — Deleting Rules

Knowing how to delete firewall rules is just as important as knowing how to create them. There are two different ways to specify which rules to delete: by rule number or by its human-readable denomination (similar to how the rules were specified when they were created).

Deleting a UFW Rule By Number

To delete a UFW rule by its number, first you'll want to obtain a numbered list of all your firewall rules. The UFW status command has an option to display numbers next to each rule, as demonstrated here:

```
$ sudo ufw status numbered
```

Numbered Output:

```
Status: active
```

To	Action	From
--	-----	----
[1] 22	ALLOW IN	15.15.15.0/24
[2] 80	ALLOW IN	Anywhere

If you decide that you want to delete rule number **2**, the one that allows port 80 (HTTP) connections, you can specify it in a UFW delete command like this:

```
$ sudo ufw delete 2
```

Output

```
Deleting:
```

```
allow 80
```

```
Proceed with operation (y|n)? y
```

```
Rule deleted
```

This will prompt for a confirmation then delete rule 2, which allows HTTP connections. Note that if you have IPv6 enabled, you would want to delete the corresponding IPv6 rule as well.

Deleting a UFW Rule By Name

SCROLL TO TOP

Instead of using rule numbers, you may also refer to a rule by its human readable denomination, which is based on the type of rule (typically `allow` or `deny`) and the service name or port number

Sign up for our newsletter Get the latest tutorials on SysAdmin and open source topics.



Enter your email address

Sign Up

```
Output
Rule deleted
Rule deleted (v6)
```

The `delete` command works the same way for rules that were created referencing a service by its name or port. For example, if you previously set a rule to allow HTTP connections with `sudo ufw allow http`, this is how you could delete said rule:

```
$ sudo ufw delete allow http
```

```
Output
Rule deleted
Rule deleted (v6)
```

Because service names are interchangeable with port numbers when specifying rules, you could also refer to the same rule as `allow 80`, instead of `allow http`:

```
$ sudo ufw delete allow 80
```

```
Output
Rule deleted
Rule deleted (v6)
```

When deleting UFW rules by name, both IPv4 and IPv6 rules are deleted if they exist.

Step 8 — Checking UFW Status and Rules

At any time, you can check the status of UFW with this command:

```
$ sudo ufw status verbose
```

If UFW is disabled, which it is by default, you'll see something like this:

```
Output
Status: inactive
```

If UFW is active, which it should be if you followed Step 3, the output will say that it's active and it will list any rules that are set. For example, if the firewall is set to allow SSH (port `22`) connections from anywhere, the output might look something like this:

```
Output
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To Action From
--
22/tcp ALLOW IN Anywhere
```

Use the `status` command if you want to check how UFW has configured the firewall.

Step 9 — Disabling or Resetting UFW (optional)

If you decide you don't want to use UFW, you can disable it with this command:

SCROLL TO TOP

```
$ sudo ufw disable
```

Sign up for our newsletter Get the latest tutorials on SysAdmin and open source topics.



Sign Up

If you already have UFW rules configured but you decide that you want to start over, you can use the reset command:

```
$ sudo ufw reset
```

Output

```
Resetting all rules to installed defaults. This may disrupt existing ssh
connections. Proceed with operation (y|n)? y
Backing up 'user.rules' to '/etc/ufw/user.rules.20210729_170353'
Backing up 'before.rules' to '/etc/ufw/before.rules.20210729_170353'
Backing up 'after.rules' to '/etc/ufw/after.rules.20210729_170353'
Backing up 'user6.rules' to '/etc/ufw/user6.rules.20210729_170353'
Backing up 'before6.rules' to '/etc/ufw/before6.rules.20210729_170353'
Backing up 'after6.rules' to '/etc/ufw/after6.rules.20210729_170353'
```

This will disable UFW and delete any rules that were previously defined. This should give you a fresh start with UFW. Keep in mind that the default policies won't change to their original settings, if you modified them at any point.

Conclusion

Your firewall is now configured to allow (at least) SSH connections. Be sure to allow any other incoming connections that your server requires, while limiting any unnecessary connections, so your server will be functional and secure.

To learn about more common UFW configurations, check out the [UFW Essentials: Common Firewall Rules and Commands](#) tutorial.

Was this helpful?

Yes

No



[Report an issue](#)

About the authors



Brian Boucheron

Senior Technical Writer at DigitalOcean



Erika Heidi

Senior Technical Writer at DigitalOcean. Dev/Ops passionate about open source, PHP, and Linux.

Still looking for an answer?



Ask a question



Search for more help

RELATED

SCROLL TO TOP

Sign Up

Join Now

How To Use MongoDB Access Control

[Tutorial](#)

What is a DDoS Attack?

[Tutorial](#)

Comments

5 Comments

B *I*



Leave a comment...

Sign In to Comment

[sworks](#) October 27, 2018

1

Thank you so much. Very helpful guide.

[Reply](#) [Report](#)[pierretempel](#) February 20, 2019

4

Before enabling the firewall and potentially disrupting services, double check your configuration using `ufw show added`. This command exists, because `ufw status` only works on an active firewall.

[Reply](#) [Report](#)[ekaharo](#) October 29, 2019

0

Very thorough....thanks

[Reply](#) [Report](#)[mikejnpc](#) March 27, 2021

0

Hello... do i need to create a user for UFW to function? or can it be done as root...

I installed UFW but when i enable

Error occurred at line: 1

Try ``iptables-restore -h`` or ``iptables-restore -help`` for more information.`sysctl: setting key "net.ipv4.conf.all.acceptredirects": Read-only file system``sysctl: setting key "net.ipv4.conf.default.acceptredirects": Read-only file system``sysctl: setting key "net.ipv6.conf.all.acceptredirects": Read-only file system``sysctl: setting key "net.ipv6.conf.default.acceptredirects": Read-only file system``sysctl: setting key "net.ipv4.icmp_echo_ignore_broadcasts": Read-only file system``sysctl: setting key "net.ipv4.icmp_ignore_bogus_error_responses": Read-only file system``sysctl: setting key "net.ipv4.icmp_echo_ignore_all": Read-only file system``sysctl: setting key "net.ipv4.conf.all.log_martians": Read-only file system``sysctl: setting key "net.ipv4.conf.default.log_martians": Read-only file system`

Problem loading ipv6 (skipping)

Problem running `/etc/ufw/before.rules`Problem running `/etc/ufw/after.rules`Problem running `/etc/ufw/user.rules`and then if i try a work around `"iptables -A INPUT -p udp -dport 10555 -j ACCEPT "`

iptables v1.6.1: can't initialize iptables table `filter': Permission denied (you must be root)
Perhaps iptables or your kernel needs to be upgraded.

Sign up for our newsletter Get the latest tutorials on SysAdmin and open source topics.



Enter your email address

Sign Up

```
sudo ufw allow from 192.168.51.1/80 to any port 3080
```

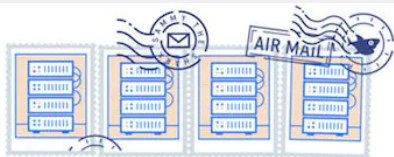
But I want to mention TCP in that command. Is it possible?

Why it throws me an error? Error says, "ERROR: Bad source address".

[Reply](#) [Report](#)

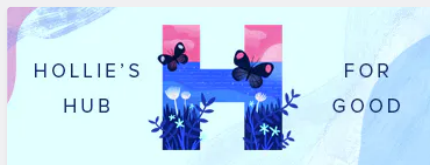


This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.



GET OUR BIWEEKLY NEWSLETTER

Sign up for Infrastructure as a Newsletter.



HOLLIE'S HUB FOR GOOD

Working on improving health and education, reducing inequality, and spurring economic growth? We'd like to help.



BECOME A CONTRIBUTOR

You get paid; we donate to tech nonprofits.

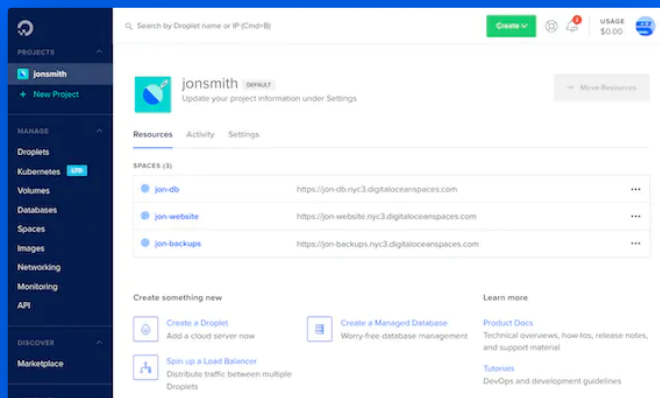
Featured on [Community](#) [Kubernetes Course](#) [Learn Python 3](#) [Machine Learning in Python](#) [Getting started with Go](#) [Intro to Kubernetes](#)

[DigitalOcean Products](#) [Virtual Machines](#) [Managed Databases](#) [Managed Kubernetes](#) [Block Storage](#) [Object Storage](#) [Marketplace](#) [VPC](#) [Load Balancers](#)

Welcome to the developer cloud

DigitalOcean makes it simple to launch in the cloud and scale up as you grow – whether you're running one virtual machine or ten thousand.

[Learn More](#)





[About](#)
[Leadership](#)
[Blog](#)

[Pricing](#)
[Products Overview](#)
[Droplets](#)

[Tutorials](#)
[Q&A](#)
[Tools and Integrations](#)

[Get Support](#)
[Trouble Signing In?](#)
[Sales](#)

Sign up for our newsletter Get the latest tutorials on SysAdmin and open source topics.



Enter your email address

Sign Up

[API Documentation](#)
[Documentation](#)
[Release Notes](#)

[Research Program](#)
[Open Source](#)
[Code of Conduct](#)

SCROLL TO TOP