

How To Set Up a Firewall with UFW on Ubuntu 16.04

Posted December 22, 2016  464.8k

FIREWALL

NETWORKING

SECURITY

UBUNTU

UBUNTU 16.04


72

By: Hazel Virdó

Not using **Ubuntu 16.04**? Choose a different version:

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 16.04.

Prerequisites

To follow this tutorial, you will need:

- One Ubuntu 16.04 server with a `sudo` non-root user, which you can set up by following Steps 1-3 in the [Initial Server Setup with Ubuntu 16.04](#) tutorial.

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

Step 1 — Using IPv6 with UFW (Optional)

This tutorial is written with IPv4 in mind, but will work for IPv6 as well as long as you enable it. If your Ubuntu server has IPv6 enabled, ensure that UFW is configured to support IPv6 so that it will manage firewall rules for IPv6 in addition to IPv4. To do this, open the UFW configuration with `nano` or your favorite editor.

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

Then make sure the value of `IPV6` is `yes` . It should look like this:

```
/etc/default/ufw excerpt  
  
...  
IPV6=yes  
...
```

Save and close the file. Now, when UFW is enabled, it will be configured to write both IPv4 and IPv6 firewall rules. However, before enabling UFW, we will want to ensure that your firewall is configured to allow you to connect via SSH. Let's start with setting the default policies.

Step 2 — Setting Up Default Policies

If you're just getting started with your firewall, the first rules to define are your default 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 cloud server would not be able to connect, while any application within the server would be able to reach the outside world.

Let's set your UFW rules back to the defaults so we can be sure that you'll be able to follow along with this tutorial. To set the defaults used by UFW, use these commands:

```
$ sudo ufw default deny incoming  
$ sudo ufw default allow outgoing
```

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 we enabled our UFW firewall now, it would deny all incoming connections. This means that we will need to create rules that explicitly allow legitimate incoming connections — SSH or HTTP connections, for example — if we want our 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.

To configure your server to allow incoming SSH connections, you can use this command:

```
$ sudo ufw allow ssh
```

This will create firewall rules that will allow all connections on port `22` , which is the port that the SSH daemon listens on by default. UFW knows what SSH and a number of other service names mean because they're listed as services in the `/etc/services` file.

However, we can actually write the equivalent rule by specifying the port instead of the service name. For example, this command works the same as the one above:

```
$ sudo ufw allow 22
```

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
```

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

Step 4 — Enabling UFW

To enable UFW, use this command:

```
$ sudo ufw enable
```

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

The firewall is now active. Feel free to 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

At this point, you should allow all of the other connections that your server needs to respond to. The connections that you should allow depends your specific needs. Luckily, you already know how to write rules that allow connections based on a service name or port; we already did this for SSH on port 22. You can also do this for:

- HTTP on port 80, which is what unencrypted web servers use, using `sudo ufw allow http` or `sudo ufw allow 80`
- HTTPS on port 443, which is what encrypted web servers use, using `sudo ufw allow https` or `sudo ufw allow 443`
- FTP on port 21, which is used for unencrypted file transfers (which you probably shouldn't use anyway), using `sudo ufw allow ftp` or `sudo ufw allow 21/tcp`

There are several others ways to allow other connections, aside from specifying a port or known service.

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 simply allows both protocols, which is OK in most cases.

Specific IP Addresses

When working with UFW, you can also specify IP addresses. For example, if you want to allow connections from a specific IP address, such as a work or home IP address of `15.15.15.51`, you need to specify `from`, then the IP address:

```
$ sudo ufw allow from 15.15.15.51
```

You can also specify a specific 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 `15.15.15.51` to connect to port `22` (SSH), use this command:

```
$ sudo ufw allow from 15.15.15.51 to any port 22
```

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 `15.15.15.1` to `15.15.15.254` you could use this command:

```
$ sudo ufw allow from 15.15.15.0/24
```

Likewise, you may also specify the destination port that the subnet `15.15.15.0/24` is allowed to connect to. Again, we'll use port `22` (SSH) as an example:

```
$ sudo ufw allow from 15.15.15.0/24 to any port 22
```

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 `eth1`.

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
```

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
```

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 change your default incoming policy to **allow** (which isn't recommended in the interest of security), 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 described above, replacing **allow** with **deny**.

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

```
$ sudo ufw deny http
```

Or if you want to deny all connections from 15.15.15.51 you could use this command:

```
$ sudo ufw deny from 15.15.15.51
```

Now let's take a look at how to delete rules.

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 specify which rules to delete: by rule number or by the actual rule (similar to how the rules were specified when they were created). We'll start with the **delete by rule number** method because it is easier, compared to writing the actual rules to delete, if you're new to UFW.

By Rule Number

If you're using the rule number to delete firewall rules, the first thing you'll want to do is get a list of 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 we decide that we want to delete rule 2, the one that allows port 80 (HTTP) connections, we can specify it in a UFW delete command like this:

```
$ sudo ufw delete 2
```

This would show a confirmation prompt 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.

By Actual Rule

The alternative to rule numbers is to specify the actual rule to delete. For example, if you want to remove the `allow http` rule, you could write it like this:

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

You could also specify the rule by `allow 80`, instead of by service name:

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

This method will delete both IPv4 and IPv6 rules, 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:

```
$ sudo ufw disable
```

Any rules that you created with UFW will no longer be active. You can always run `sudo ufw enable` if you need to activate it later.

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
```

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

Conclusion

Your firewall should now be configured to allow (at least) SSH connections. Be sure to allow any other incoming connections that your server, 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.

By: Hazel Virdó

♡ Upvote (72)

📌 Subscribe

🔗 Share



We just made it easier for you to deploy faster.

[TRY FREE](#)

Related Tutorials

How To Migrate Iptables Firewall Rules to a New Server

How To Secure Nginx with NAXSI on Ubuntu 16.04

How To Set Up a Firewall with UFW on Debian 9

How To Set Up a Firewall with UFW on Ubuntu 18.04

How To Secure Nginx with Let's Encrypt on FreeBSD

16 Comments

Leave a comment...

Log In to Comment

^ [newbie](#) December 27, 2016



0 I have a small question

suppose i blocked all incoming requests on ssh accept my home ip like this

```
sudo ufw allow from 15.15.15.51 to any port ssh
```

all other incoming ip to ssh will be blocked? what if my ip got changed suddenly and i didnt had chance to whitelist the new ip before the change? will be completely locked down?

^ [newbie](#) December 27, 2016



0 and another thing, this one is a bit silly though,
all the http requests are redirecting to https in nginx through 301 redirect, do i still need to have port 80 open in ufw?

thanks in advance.

^  [aaronellington](#) December 29, 2016

¹ Yes, you will still need port 80 open otherwise browsers will not get the 301 response.

^  [aaronellington](#) December 29, 2016

¹ You could always use your console access if it's DO Droplet, but either way you will need another way to access it if your IP changes.

If you do not have a static IP I do not suggest doing this.

^  [newbie](#) December 30, 2016

⁰ Really appreciate your both replies.


^  [paulk](#) January 10, 2017

⁰ HI

I think you might be wrong on the default rules. UFW, there are no default rules, you have to define them, so on install all ports are open and outgoing is open, so is incorrect to say all ports are blocked on install.

^  [MikeX](#) June 29, 2017

⁰ I registered just to thank you for this great tutorial on ufw. All others I found were either too trivial, not showing the stuff I needed, or too verbose and confusing. Yours is perfect and explains everything I needed to know.

^  [olivebay84](#) August 9, 2017

⁰ why it is not mentioned if those rules are persistent or not?

^  [rochie94](#) November 18, 2017

⁰ As a beginner this is amazing, thank you

^  [greengablesfan33](#) December 10, 2017

⁰ Is there a tutorial on how to allow certain applications you installed manually through the sudo sh dev where you add an additional package? I tried installing Icecast2 but it said, error establishing connection socket to port 8000. I tried with both localhost and my IP address of the server, but it didn't work. If it is an ippables issue, how do I fix this?

^  [atiqul](#) January 8, 2018

⁰ After Flow the step Step 4 — Enabling UFW
I found this error

Command may disrupt existing ssh connections. Proceed with operation (y/n)? y
ERROR: problem running ufw-init
modprobe: ERROR: ../libkmod/libkmod.c:514 lookupbuiltinfile() could not open builtin file '/lib/modules/4.4.0-042stab120.16/modules.builtin.bin'
modprobe: FATAL: Module nfconntrackftp not found in directory /lib/modules/4.4.0-042stab120.16
modprobe: ERROR: ../libkmod/libkmod.c:514 lookupbuiltinfile() could not open builtin file '/lib/modules/4.4.0-042stab120.16/modules.builtin.bin'
modprobe: FATAL: Module nfnetlink not found in directory /lib/modules/4.4.0-042stab120.16
modprobe: ERROR: ../libkmod/libkmod.c:514 lookupbuiltinfile() could not open builtin file '/lib/modules/4.4.0-042stab120.16/modules.builtin.bin'
modprobe: FATAL: Module nfconntracknetbiosns not found in directory /lib/modules/4.4.0-042stab120.16
iptables-restore: line 4 failed
iptables-restore: line 77 failed
iptables-restore: line 31 failed
sysctl: permission denied on key 'net.ipv4.tcp sack'

Problem loading ipv6 (skipping)
Problem running '/etc/ufw/before.rules'
Problem running '/etc/ufw/user.rules'

What can i do now please can anybody answer me...

^ [romanemperor6](#) January 23, 2018

0 Hi,

I followed the tutorial and allowed port 22, but still can't connect via putty anymore.
Is there a known problem for this or did I do something wrong?

^ [JasMineBen](#) March 21, 2018

0 Were you able to connect via putty again? I have the same problem.

^ [abhi5459](#) May 13, 2018

0 Thank you! This was very helpful.

^ [osrsbots](#) November 1, 2018

0 If I'm using DigitalOcean's firewall on a droplet, do I need to use UFW as well? I'm assuming not, but would that be best practice somehow, to have double security?

^ [mmahdali125](#) January 2, 2019

0 I think it
sudo ufw allow from 15.15.15.0/24
should be
sudo ufw allow from 15.15.15.0/254 ?

as per
15.15.15.1 to 15.15.15.254 you



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



Copyright © 2019 DigitalOcean™ Inc.

[Community](#) [Tutorials](#) [Questions](#) [Projects](#) [Tags](#) [Newsletter](#) [RSS](#) 

[Distros & One-Click Apps](#) [Terms, Privacy, & Copyright](#) [Security](#) [Report a Bug](#) [Write for DOnations](#) [Shop](#)