

// Tutorial //

# UFW Essentials: Common Firewall Rules and Commands

Published on August 20, 2015 · Updated on December 15, 2021

Security   Ubuntu   Firewall   Networking



By [Mitchell Anicas](#) and [Erika Heidi](#)

English



## Introduction

UFW (**un**complicated **fire**wall) is a firewall configuration tool that runs on top of `iptables`, included by default within Ubuntu distributions. It provides a streamlined interface for configuring common firewall use cases via the command line.

This cheat sheet-style guide provides a quick reference to common UFW use cases and commands, including examples of how to allow and block services by port, network interface, and source IP address.

## How To Use This Guide

- This guide is in cheat sheet format with self-contained command-line snippets.
- Jump to any section that is relevant to the task you are trying to complete.

- When you see highlighted text in this guide's commands, keep in mind that this text should refer to IP addresses from your own network.

Remember that you can check your current UFW ruleset with `sudo ufw status` or `sudo ufw status verbose`.

## Verify UFW Status

To check if `ufw` is enabled, run:

```
sudo ufw status
```

Copy

Output

```
Status: inactive
```

The output will indicate if your firewall is active or not.

## Enable UFW

If you got a `Status: inactive` message when running `ufw status`, it means the firewall is not yet enabled on the system. You'll need to run a command to enable it.

By default, when enabled UFW will block external access to all ports on a server. In practice, that means if you are connected to a server via SSH and enable `ufw` before allowing access via the SSH port, you'll be disconnected. Make sure you follow the section on [how to enable SSH access](#) of this guide before enabling the firewall if that's your case.

To enable UFW on your system, run:

```
sudo ufw enable
```

Copy

You'll see output like this:

Output

```
Firewall is active and enabled on system startup
```

To see what is currently blocked or allowed, you may use the `verbose` parameter when running `ufw status`, as follows:

```
sudo ufw status
```

[Copy](#)

### Output

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

## Disable UFW

If for some reason you need to disable UFW, you can do so with the following command:

```
sudo ufw disable
```

[Copy](#)

Be aware that this command will fully disable the firewall service on your system.

## Block an IP Address

To block all network connections that originate from a specific IP address, run the following command, replacing the highlighted IP address with the IP address that you want to block:

```
sudo ufw deny from 203.0.113.100
```

[Copy](#)

### Output

```
Rule added
```

In this example, `from 203.0.113.100` specifies a **source** IP address of “203.0.113.100”.

If you run `sudo ufw status` now, you’ll see the specified IP address listed as denied:

### Output

```
Status: active

To Action From
--
Anywhere DENY 203.0.113.100
```

All connections, coming in or going out, are blocked for the specified IP address.

## Block a Subnet

If you need to block a full subnet, you may use the subnet address as `from` parameter on the `ufw deny` command. This would block all IP addresses in the example subnet `203.0.113.0/24`:

```
sudo ufw deny from 203.0.113.0/24
```

Copy

### Output

```
Rule added
```

## Block Incoming Connections to a Network Interface

To block incoming connections from a specific IP address to a specific network interface, run the following command, replacing the highlighted IP address with the IP address you want to block:

```
sudo ufw deny in on eth0 from 203.0.113.100
```

Copy

### Output

```
Rule added
```

The `in` parameter tells `ufw` to apply the rule only for **incoming** connections, and the `on eth0` parameter specifies that the rule applies only for the `eth0` interface. This might be useful if you have a system with several network interfaces (including virtual ones) and you need to block external access to some of these interfaces, but not all.

## Allow an IP Address

To allow all network connections that originate from a specific IP address, run the following command, replacing the highlighted IP address with the IP address that you want to allow access:

```
sudo ufw allow from 203.0.113.101
```

Copy

### Output

```
Rule added
```

If you run `sudo ufw status` now, you'll see output similar to this, showing the word `ALLOW` next to the IP address you just added.

### Output

Status: active

To	Action	From
--	-----	----
...		
Anywhere	ALLOW	203.0.113.101

You can also allow connections from a whole subnet by providing the corresponding subnet mask for a host, such as `203.0.113.0/24`.

## Allow Incoming Connections to a Network Interface

To allow incoming connections from a specific IP address to a specific network interface, run the following command, replacing the highlighted IP address with the IP address you want to allow:

```
sudo ufw allow in on eth0 from 203.0.113.102
```

Copy

### Output

Rule added

The `in` parameter tells `ufw` to apply the rule only for **incoming** connections, and the `on eth0` parameter specifies that the rule applies only for the `eth0` interface.

If you run `sudo ufw status` now, you'll see output similar to this:

### Output

Status: active

To	Action	From
--	-----	----
...		
Anywhere on eth0	ALLOW	203.0.113.102

## Delete UFW Rule

To delete a rule that you previously set up within UFW, use `ufw delete` followed by the rule (`allow` or `deny`) and the target specification. The following example would delete a rule previously set to allow all connections from an IP address of `203.0.113.101`:

```
sudo ufw delete allow from 203.0.113.101
```

Copy

### Output

```
Rule deleted
```

Another way to specify which rule you want to delete is by providing the rule ID. This information can be obtained with the following command:

```
sudo ufw status numbered
```

Copy

### Output

```
Status: active
```

	To	Action	From
	--	-----	----
[ 1 ]	Anywhere	DENY IN	203.0.113.100
[ 2 ]	Anywhere on eth0	ALLOW IN	203.0.113.102

From the output, you can see that there are two active rules. The first rule, with highlighted values, denies all connections coming from the IP address `203.0.113.100`. The second rule allows connections on the `eth0` interface coming in from the IP address `203.0.113.102`.

Because by default UFW already blocks all external access unless explicitly allowed, the first rule is redundant, so you can remove it. To delete a rule by its ID, run:

```
sudo ufw delete 1
```

Copy

You will be prompted to confirm the operation and to make sure the ID you're providing refers to the correct rule you want to delete.

### Output

```
Deleting:
  deny from 203.0.113.100
Proceed with operation (y|n)? y
Rule deleted
```

If you list your rules again with `sudo ufw status`, you'll see that the rule was removed.

## List Available Application Profiles

Upon installation, applications that rely on network communications will typically set up a UFW profile that you can use to allow connection from external addresses. This is often the same as running `ufw allow from`, with the advantage of providing a shortcut that abstracts the specific port numbers a service uses and provides a user-friendly nomenclature to referenced services.

To list which profiles are currently available, run the following:

```
sudo ufw app list
```

Copy

If you installed a service such as a web server or other network-dependent software and a profile was not made available within UFW, first make sure the service is enabled. For remote servers, you'll typically have OpenSSH readily available:

#### Output

```
Available applications:
  OpenSSH
```

## Enable Application Profile

To enable a UFW application profile, run `ufw allow` followed by the name of the application profile you want to enable, which you can obtain with a `sudo ufw app list` command. In the following example, we're enabling the OpenSSH profile, which will allow all incoming SSH connections on the default SSH port.

```
sudo ufw allow "OpenSSH"
```

Copy

#### Output

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

Remember to quote profile names that consist of multiple words, such as `Nginx HTTPS`.

## Disable Application Profile

To disable an application profile that you had previously set up within UFW, you'll need to remove its corresponding rule. For example, consider the following output from `sudo ufw status`:

```
sudo ufw status
```

Copy

## Output

Status: active

To	Action	From
--	-----	----
OpenSSH	ALLOW	Anywhere
Nginx Full	ALLOW	Anywhere
OpenSSH (v6)	ALLOW	Anywhere (v6)
Nginx Full (v6)	ALLOW	Anywhere (v6)

This output indicates that the `Nginx Full` application profile is currently enabled, allowing any and all connections to the web server both via HTTP as well as via HTTPS. If you'd want to only allow HTTPS requests from and to your web server, you'd have to first enable the most restrictive rule, which in this case would be `Nginx HTTPS`, and then disable the currently active `Nginx Full` rule:

```
sudo ufw allow "Nginx HTTPS"
sudo ufw delete allow "Nginx Full"
```

Copy

Remember you can list all available application profiles with `sudo ufw app list`.

## Allow SSH

When working with remote servers, you'll want to make sure that the SSH port is open to connections so that you are able to log in to your server remotely.

The following command will enable the OpenSSH UFW application profile and allow all connections to the default SSH port on the server:

```
sudo ufw allow OpenSSH
```

Copy

## Output

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

Although less user-friendly, an alternative syntax is to specify the exact port number of the SSH service, which is typically set to `22` by default:

```
sudo ufw allow 22
```

Copy

## Output



Rule added  
Rule added (v6)

## Allow Incoming SSH from Specific IP Address or Subnet

To allow incoming connections from a specific IP address or subnet, you'll include a `from` directive to define the source of the connection. This will require that you also specify the destination address with a `to` parameter. To lock this rule to SSH only, you'll limit the `proto` (protocol) to `tcp` and then use the `port` parameter and set it to `22`, SSH's default port.

The following command will allow only SSH connections coming from the IP address `203.0.113.103`:

```
sudo ufw allow from 203.0.113.103 proto tcp to any port 22
```

Copy

### Output

Rule added

You can also use a subnet address as `from` parameter to allow incoming SSH connections from an entire network:

```
sudo ufw allow from 203.0.113.0/24 proto tcp to any port 22
```

Copy

### Output

Rule added

## Allow Incoming Rsync from Specific IP Address or Subnet

The [Rsync](#) program, which runs on port `873`, can be used to transfer files from one computer to another.

To allow incoming `rsync` connections from a specific IP address or subnet, use the `from` parameter to specify the source IP address and the `port` parameter to set the destination port `873`. The following command will allow only Rsync connections coming from the IP address `203.0.113.103`:

```
sudo ufw allow from 203.0.113.103 to any port 873
```

Copy

### Output

Rule added

To allow the entire `203.0.113.0/24` subnet to be able to `rsync` to your server, run:

```
sudo ufw allow from 203.0.113.0/24 to any port 873
```

Copy

#### Output

Rule added

## Allow Nginx HTTP / HTTPS

Upon installation, the Nginx web server sets up a few different UFW profiles within the server. Once you have Nginx installed and enabled as a service, run the following command to identify which profiles are available:

```
sudo ufw app list | grep Nginx
```

Copy

#### Output

```
Nginx Full
Nginx HTTP
Nginx HTTPS
```

To enable both HTTP and HTTPS traffic, choose `Nginx Full`. Otherwise, choose either `Nginx HTTP` to allow only HTTP or `Nginx HTTPS` to allow only HTTPS.

The following command will allow both HTTP and HTTPS traffic on the server (ports `80` and `443`):

```
sudo ufw allow "Nginx Full"
```

Copy

#### Output

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

## Allow Apache HTTP / HTTPS

Upon installation, the Apache web server sets up a few different UFW profiles within the server. Once you have Apache installed and enabled as a service, run the following command to identify which

profiles are available:

```
sudo ufw app list | grep Apache
```

Copy

#### Output

```
Apache
Apache Full
Apache Secure
```

To enable both HTTP and HTTPS traffic, choose `Apache Full`. Otherwise, choose either `Apache` for HTTP or `Apache Secure` for HTTPS.

The following command will allow both HTTP and HTTPS traffic on the server (ports `80` and `443`):

```
sudo ufw allow "Nginx Full"
```

Copy

#### Output

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

## Allow All Incoming HTTP (port `80`)

Web servers, such as Apache and Nginx, typically listen for HTTP requests on port `80`. If your default policy for incoming traffic is set to drop or deny, you'll need to create a UFW rule to allow external access on port `80`. You can use either the port number or the service name (`http`) as a parameter to this command.

To allow all incoming HTTP (port `80`) connections, run:

```
sudo ufw allow http
```

Copy

#### Output

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

An alternative syntax is to specify the port number of the HTTP service:

```
sudo ufw allow 80
```

Copy

### Output

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

## Allow All Incoming HTTPS (port 443)

HTTPS typically runs on port 443 . If your default policy for incoming traffic is set to drop or deny, you'll need to create a UFW rule to allow external access on port 443 . You can use either the port number or the service name ( `https` ) as a parameter to this command.

To allow all incoming HTTPS (port 443 ) connections, run:

```
sudo ufw allow https
```

Copy

### Output

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

An alternative syntax is to specify the port number of the HTTPS service:

```
sudo ufw allow 443
```

Copy

### Output

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

## Allow All Incoming HTTP and HTTPS

If you want to allow both HTTP and HTTPS traffic, you can create a single rule that allows both ports. This usage requires that you also define the protocol with the `proto` parameter, which in this case should be set to `tcp` .

To allow all incoming HTTP and HTTPS (ports 80 and 443 ) connections, run:

```
sudo ufw allow proto tcp from any to any port 80,443
```

Copy

### Output

Rule added  
Rule added (v6)

## Allow MySQL Connection from Specific IP Address or Subnet

MySQL listens for client connections on port 3306 . If your MySQL database server is being used by a client on a remote server, you'll need to create a UFW rule to allow that access.

To allow incoming MySQL connections from a specific IP address or subnet, use the `from` parameter to specify the source IP address and the `port` parameter to set the destination port 3306 .

The following command will allow the IP address 203.0.113.103 to connect to the server's MySQL port:

```
sudo ufw allow from 203.0.113.103 to any port 3306
```

Copy

Output

Rule added

To allow the entire 203.0.113.0/24 subnet to be able to connect to your MySQL server, run:

```
sudo ufw allow from 203.0.113.0/24 to any port 3306
```

Copy

Output

Rule added

## Allow PostgreSQL Connection from Specific IP Address or Subnet

PostgreSQL listens for client connections on port 5432 . If your PostgreSQL database server is being used by a client on a remote server, you need to be sure to allow that traffic.

To allow incoming PostgreSQL connections from a specific IP address or subnet, specify the source with the `from` parameter, and set the port to 5432 :

```
sudo ufw allow from 203.0.113.103 to any port 5432
```

Copy

Output

Rule added

To allow the entire `203.0.113.0/24` subnet to be able to connect to your PostgreSQL server, run:

```
sudo ufw allow from 203.0.113.0/24 to any port 5432
```

Copy

#### Output

Rule added

## Block Outgoing SMTP Mail

Mail servers, such as Sendmail and Postfix, typically use port `25` for SMTP traffic. If your server shouldn't be sending outgoing mail, you may want to block that kind of traffic. To block outgoing SMTP connections, run:

```
sudo ufw deny out 25
```

Copy

#### Output

Rule added

Rule added (v6)

This configures your firewall to **drop** all outgoing traffic on port `25`. If you need to reject outgoing connections on a different port number, you can repeat this command and replace `25` with the port number you want to block.

## Conclusion

UFW is a powerful tool that can greatly improve the security of your servers when properly configured. This reference guide covers some common UFW rules that are often used to configure a firewall on Ubuntu.

Most of the commands in this guide can be adapted to fit different use cases and scenarios, by changing parameters such as the source IP address and/or destination port. For more detailed information about each command parameter and available modifiers, you can use the `man` utility to check UFW's manual:

```
man ufw
```

Copy

The [official UFW page on Ubuntu's documentation](#) is another resource you can use as reference for more advanced use cases and examples.