UFW Essentials: Common Firewall Rules and Commands

Posted August 20, 2015 778.6k FIREWALL SECURITY NETWORKING UBUNTU

By: Mitchell Anicas

Introduction

UFW is a firewall configuration tool for iptables that is included with Ubuntu by default. This cheat sheet-style guide provides a quick reference to UFW commands that will create iptables firewall rules are useful in common, everyday scenarios. This includes UFW examples of allowing and blocking various services by port, network interface, and source IP address.

How To Use This Guide

- If you are just getting started with using UFW to configure your firewall, check out our introduction to UFW
- Most of the rules that are described here assume that you are using the default UFW ruleset. That is, it is set to allow outgoing and deny incoming traffic, through the default policies, so you have to selectively allow traffic in
- Use whichever subsequent sections are applicable to what you are trying to achieve. Most sections are not predicated on any other, so you can use the examples below independently
- Use the Contents menu on the right side of this page (at wide page widths) or your browser's find function to locate the sections you need
- Copy and paste the command-line examples given, substituting the values in red with your own values

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

Block an IP Address

To block all network connections that originate from a specific IP address, `15.15.15.51` for example, run this command:

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

In this example, `from 15.15.15.51` specifies a **source** IP address of "15.15.15.51". If you wish, a subnet, such as `15.15.15.0/24`, may be specified here instead. The source IP address can be specified in any firewall rule, including an **allow** rule.

Block Connections to a Network Interface

To block connections from a specific IP address, e.g. `15.15.15.51`, to a specific network interface, e.g. `eth0`, use this command:

```
$ sudo ufw deny in on eth0 from 15.15.15.51
```

This is the same as the previous example, with the addition of `in on eth0`. The network interface can be specified in any firewall rule, and is a great way to limit the rule to a particular network.

Service: SSH

If you're using a cloud server, you will probably want to allow incoming SSH connections (port 22) so you can connect to and manage your server. This section covers how to configure your firewall with various SSH-related rules.

Allow SSH

To allow all incoming SSH connections run this command:

```
$ sudo ufw allow ssh
```

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

```
$ sudo ufw allow 22
```

Allow Incoming SSH from Specific IP Address or Subnet

To allow incoming SSH connections from a specific IP address or subnet, specify the source. For example, if you want to allow the entire `15.15.15.0/24` subnet, run this command:

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

Allow Incoming Rsync from Specific IP Address or Subnet

Rsync, 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, specify the source IP address and the destination port. For example, if you want to allow the entire `15.15.15.0/24` subnet to be able to rsync to your server, run this command:

```
$ sudo ufw allow from 15.15.15.0/24 to any port 873
```

Service: Web Server

Web servers, such as Apache and Nginx, typically listen for requests on port 80 and 443 for HTTP and HTTPS connections, respectively. If your default policy for incoming traffic is set to drop or deny, you will want to create rules that will allow your server to respond to those requests.

Allow All Incoming HTTP

To allow all incoming HTTP (port 80) connections run this command:

```
$ sudo ufw allow http
```

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

```
$ sudo ufw allow 80
```

Allow All Incoming HTTPS

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

```
$ sudo ufw allow https
```

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

```
$ sudo ufw allow 443
```

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. To allow all incoming HTTP and HTTPS (port 443) connections run this command:

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

Note that you need to specify the protocol, with `proto tcp`, when specifying multiple ports.

Service: MySQL

MySQL listens for client connections on port 3306. If your MySQL database server is being used by a client on a remote server, you need to be sure to allow that traffic.

Allow MySQL from Specific IP Address or Subnet

To allow incoming MySQL connections from a specific IP address or subnet, specify the source. For example, if you want to allow the entire `15.15.15.0/24` subnet, run this command:

```
$ sudo ufw allow from 15.15.15.0/24 to any port 3306
```

Allow MySQL to Specific Network Interface

To allow MySQL connections to a specific network interface—say you have a private network interface `eth1`, for example—use this command:

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

Service: PostgreSQL

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.

PostgreSQL from Specific IP Address or Subnet

To allow incoming PostgreSQL connections from a specific IP address or subnet, specify the source. For example, if you want to allow the entire `15.15.15.0/24` subnet, run this command:

```
$ sudo ufw allow from 15.15.15.0/24 to any port 5432
```

The second command, which allows the outgoing traffic of **established** PostgreSQL connections, is only necessary if the `OUTPUT` policy is not set to `ACCEPT`.

Allow PostgreSQL to Specific Network Interface

To allow PostgreSQL connections to a specific network interface—say you have a private network interface `eth1`, for example—use this command:

```
$ sudo ufw allow in on eth1 to any port 5432
```

The second command, which allows the outgoing traffic of **established** PostgreSQL connections, is only necessary if the `OUTPUT` policy is not set to `ACCEPT`.

Service: Mail

Mail servers, such as Sendmail and Postfix, listen on a variety of ports depending on the protocols being used for mail delivery. If you are running a mail server, determine which protocols you are using and allow the appropriate types of traffic. We will also show you how to create a rule to block outgoing SMTP mail.

Block Outgoing SMTP Mail

If your server shouldn't be sending outgoing mail, you may want to block that kind of traffic. To block outgoing SMTP mail, which uses port 25, run this command:

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

This configures your firewall to **drop** all outgoing traffic on port 25. If you need to service by its port number, instead of port 25, simply replace it.

Allow All Incoming SMTP

To allow your server to respond to SMTP connections, port 25, run this command:

```
$ sudo ufw allow 25
```

Note: It is common for SMTP servers to use port 587 for outbound mail.

Allow All Incoming IMAP

To allow your server to respond to IMAP connections, port 143, run this command:

```
$ sudo ufw allow 143
```

Allow All Incoming IMAPS

To allow your server to respond to IMAPS connections, port 993, run this command:

```
$ sudo ufw allow 993
```

Allow All Incoming POP3

To allow your server to respond to POP3 connections, port 110, run this command:

```
$ sudo ufw allow 110
```

Allow All Incoming POP3S

To allow your server to respond to POP3S connections, port 995, run this command:

```
$ sudo ufw allow 995
```

Conclusion

That should cover many of the commands that are commonly used when using U[illegible] firewall. Of course, UFW is a very flexible tool so feel free to mix and match the co[illegible]