By: finid

Subscribe

How to Configure the Linux Firewall for Docker Swarm on Ubuntu 16.04

12

Posted January 9, 2017 29.3k FIREWALL DOCKER SECURITY UBUNTU 16.04

Introduction

Docker Swarm is a feature of Docker that makes it easy to run Docker hosts and containers at scale. A Docker Swarm, or Docker cluster, is made up of one or more Dockerized hosts that function as *manager* nodes, and any number of *worker* nodes. Setting up such a system requires careful manipulation of the Linux firewall.

The network ports required for a Docker Swarm to function correctly are:

- TCP port `2376` for secure Docker client communication. This port is required for Docker Machine to work. Docker Machine is used to orchestrate Docker hosts.
- TCP port `2377`. This port is used for communication between the nodes of a Docker Swarm or cluster. It only needs to be opened on manager nodes.
- TCP and UDP port `7946` for communication among nodes (container network discovery).
- UDP port `4789` for overlay network traffic (container ingress networking).

> **Note:** Aside from those ports, port `22` (for SSH traffic) and any other ports needed for specific services to run on the cluster have to be open.

In this article, you'll learn how to configure the Linux firewall on Ubuntu 16.04 using the different firewall management applications available on all Linux distributions. Those firewall management applications are FirewallD, IPTables Tools, and UFW, the Uncomplicated Firewall. UFW is the default firewall application on Ubuntu distributions, including Ubuntu 16.04. While this tutorial covers three methods, each one delivers the same outcome, so you can choose the one you are most familiar with.

Before proceeding with this article, you should:

- Set up the hosts that make up your cluster, including at least one swarm manager and one swarm worker. You can follow the tutorial How To Provision and Manage Remote Docker Hosts with Docker Machine on Ubuntu 16.04 to set these up.

> **Note:** You'll notice that the commands (and all the commands in this article) are not prefixed with `sudo`. That's because it's assumed that you're logged into the server using the `docker-machine ssh` command after provisioning it using Docker Machine.

Method 1 — Opening Docker Swarm Ports Using UFW

If you just set up your Docker hosts, UFW is already installed. You just need to enable and configure it. Follow this guide to learn more about using UFW on Ubuntu 16.04.

Execute the following commands on the nodes that will function as Swarm managers:

```
$ ufw allow 22/tcp
$ ufw allow 2376/tcp
$ ufw allow 2377/tcp
$ ufw allow 7946/tcp
$ ufw allow 7946/udp
$ ufw allow 4789/udp
```

Afterwards, reload UFW:

```
$ ufw reload
```

If UFW isn't enabled, do so with the following command:

```
$ ufw enable
```

This might not be necessary, but it never hurts to restart the Docker daemon anytime you make changes to and restart the firewall:

```
systemctl restart docker
```

```
$ ufw allow 22/tcp
$ ufw allow 2376/tcp
$ ufw allow 7946/tcp
$ ufw allow 7946/udp
$ ufw allow 4789/udp
```

Afterwards, reload UFW:

```
$ ufw reload
```

If UFW isn't enabled, enable it:

```
$ ufw enable
```

Then restart the Docker daemon:

```
systemctl restart docker
```

That's all you need to do to open the necessary ports for Docker Swarm using UFW.

Method 2 — Opening Docker Swarm Ports Using FirewallD

FirewallD is the default firewall application on Fedora, CentOS and other Linux distributions that are based on them. But FirewallD is also available on other Linux distributions, including Ubuntu 16.04.

If you opt to use FirewallD instead of UFW, first uninstall UFW:

```
$ apt-get purge ufw
```

Then install FirewallD:

```
$ apt-get install firewalld
```

Verify that it's running:

If it's not running, start it:

```
$ systemctl start firewalld
```

Then enable it so that it starts on boot:

```
$ systemctl enable firewalld
```

On the node that will be a Swarm manager, use the following commands to open the necessary ports:

```
$ firewall-cmd --add-port=22/tcp --permanent
$ firewall-cmd --add-port=2376/tcp --permanent
$ firewall-cmd --add-port=2377/tcp --permanent
$ firewall-cmd --add-port=7946/tcp --permanent
$ firewall-cmd --add-port=7946/udp --permanent
$ firewall-cmd --add-port=4789/udp --permanent
```

Note: If you make a mistake and need to remove an entry, type:

`firewall-cmd --remove-port=port-number/tcp –permanent`.

Afterwards, reload the firewall:

```
$ firewall-cmd --reload
```

Then restart Docker.

```
$ systemctl restart docker
```

Then on each node that will function as a Swarm worker, execute the following commands:

```
$ firewall-cmd --add-port=22/tcp --permanent
$ firewall-cmd --add-port=2376/tcp --permanent
$ firewall-cmd --add-port=7946/tcp --permanent
$ firewall-cmd --add-port=7946/udp --permanent
$ firewall-cmd --add-port=4789/udp --permanent
```

```
$ firewall-cmd --reload
```

Then restart Docker.

```
systemctl restart docker
```

You've successfully used FirewallD to open the necessary ports for Docker Swarm.

Method 3 — Opening Docker Swarm Ports Using IPTables

To use IPtables on any Linux distribution, you'll have to first uninstall any other firewall utilities. If you're switching from FirewallD or UFW, first uninstall them.

Then install the `iptables-persistent` package, which manages the automatic loading of IPtables rules:

```
$ apt-get install iptables-persistent
```

Next, flush any existing rules using this command:

```
$ netfilter-persistent flush
```

Now you can add rules using the `iptables` utility. This first set of command should be executed on the nodes that will serve as Swarm managers.

```
$ iptables -A INPUT -p tcp --dport 22 -j ACCEPT
$ iptables -A INPUT -p tcp --dport 2376 -j ACCEPT
$ iptables -A INPUT -p tcp --dport 2377 -j ACCEPT
$ iptables -A INPUT -p tcp --dport 7946 -j ACCEPT
$ iptables -A INPUT -p udp --dport 7946 -j ACCEPT
$ iptables -A INPUT -p udp --dport 4789 -j ACCEPT
```

After you enter all of the commands, save the rules to disk:

```
$ netfilter-persistent save
```

Then restart Docker.

```
$ sudo systemctl restart docker
```

On the nodes that will function as Swarm workers, execute these commands:

```
$ iptables -A INPUT -p tcp --dport 22 -j ACCEPT
$ iptables -A INPUT -p tcp --dport 2376 -j ACCEPT
$ iptables -A INPUT -p tcp --dport 7946 -j ACCEPT
$ iptables -A INPUT -p udp --dport 7946 -j ACCEPT
$ iptables -A INPUT -p udp --dport 4789 -j ACCEPT
```

Save these new rules to disk:

```
$ netfilter-persistent save
```

Then restart Docker:

```
$ sudo systemctl restart docker
```

That's all it takes to open the necessary ports for Docker Swarm using IPTables. You can learn more about how these rules work in the tutorial How the Iptables Firewall Works.

If you wish to switch to FirewallD or UFW after using this method, the proper way to go about it is to first stop the firewall:

```
$ sudo netfilter-persistent stop
```

Then flush the rules:

```
$ sudo netfilter-persistent flush
```

Finally, save the now empty tables to disk:

```
$ sudo netfilter-persistent save
```

Then you can switch to UFW or FirewallD.

Conclusion

FirewallD, IPTables Tools and UFW are the three firewall management applications in the Linux world. You just learned how to use each to open the network ports needed to set up Docker Swarm. Which method you use is just a matter of personal preference, as they are all equally capable.

By: finid

Editor:
Brian Hogan

Related Tutorials

How To Build Docker Images and Host a Docker Image Repository with GitLab

Webinar Series: Getting Started with Kubernetes

Webinar Series: Building Containerized Applications

Webinar Series: Getting Started with Containers

How To Secure Web Server Infrastructure With DigitalOcean Cloud Firewalls Using Doctl

6 Comments

Leave a comment...

Log In to Comment

JKON *July 27, 2017*

0 Did you notice that when using ufw as a firewall and for example:

- You have blocked all the ports with UFW except 22
- You have docker container running with port binding "80:80"

The traffic will still go through to your docker container. Docker does some fancy stuff with iptables when running the containers. This does not seem very secure. Either you need to have your docker virtual networks carefully crafted or you need to bind the ports to 127.0.0.1:PORT:PORT, which I'm not sure how well with works in multi node swarm mode.

// JKON

danf37b25eb626a *October 3, 2017*

0 When I follow the instructions for setting up a swarm, the swarm works initially, but after a few minutes it becomes unreachable from docker cloud and from the docker for mac application. It says "Docker Cloud is receiving heartbeats, but cannot connect to the swarm
Resolution: Swarm is not publicly reachable, it may be behind a firewall or NAT"

Anything else I should try? This is my ufw configuration:

```
22/tcp                     ALLOW       Anywhere
2376/tcp                   ALLOW       Anywhere
2377/tcp                   ALLOW       Anywhere
7946/tcp                   ALLOW       Anywhere
7946/udp                   ALLOW       Anywhere
4789/udp                   ALLOW       Anywhere
6783/tcp                   ALLOW       Anywhere
6783/udp                   ALLOW       Anywhere
```

```
80/tcp                     ALLOW       Anywhere
8080/tcp                   ALLOW       Anywhere
9000/tcp                   ALLOW       Anywhere
9000/udp                   DENY        Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
2376/tcp (v6)              ALLOW       Anywhere (v6)
2377/tcp (v6)              ALLOW       Anywhere (v6)
7946/tcp (v6)              ALLOW       Anywhere (v6)
7946/udp (v6)              ALLOW       Anywhere (v6)
4789/udp (v6)              ALLOW       Anywhere (v6)
6783/tcp (v6)              ALLOW       Anywhere (v6)
6783/udp (v6)              ALLOW       Anywhere (v6)
2375/tcp (v6)              ALLOW       Anywhere (v6)
443/tcp (v6)               ALLOW       Anywhere (v6)
80/tcp (v6)                ALLOW       Anywhere (v6)
8080/tcp (v6)              ALLOW       Anywhere (v6)
9000/tcp (v6)              ALLOW       Anywhere (v6)
```

f858043b042ee57 *November 2, 2017*

0 I ran a series of "ufw allow" commands, restarted docker, checked "ufw status" and see the firewall rules there. When I look at the droplet in my DO dashboard, the Networking->Firewalls section says "You haven't applied any Firewalls to this Droplet yet." Is the droplet firewall something different?

mikkel418725b8766e1a77ce54 *January 28, 2018*

0 I have to log into EACH AND EVERY SERVER to configure the firewall ? What if I have 200 servers ? Or something I don't get here ?

tannerchung *February 8, 2018*

0 I'm having a tough time getting docker-machine to work properly, able to `docker-machine ssh` to a node but unable to use `docker-machine env`. And, my swarm is able to spin up containers on a worker, but unable to redirect traffic from a web browser to any services located only on a worker.

I'm now trying to add security group permissions based on this chart https://docs.docker.com/datacenter/ucp/2.2/guides/admin/install/system-requirements/#ports-used and the ports you suggested.. but can you help explain why 4789 is not on the list and why the other ports you mentioned aren't detailed with ingress and egress?

0 So docker-machine worked after I added `2376` TCP ingress to my Open Stack Security Group.

You mention that `2377` TCP is used between nodes only for managers but the docker docs say

```
managers, workers   in  TCP 2377 (configurable) Port for communication be
workers out TCP 2377 (configurable) Port for communication between swarm
```

mangers **and** workers for 2377.. is that right? nevermind about `4789` , it's on the list

```
managers, workers in, out UDP 4789 Port for overlay networking
```