upcloud.com

Configuring iptables on Ubuntu 14.04

https://www.upcloud.com/support/configuring-iptables-on-ubuntu-14-04/

The user-space application program iptables allows configuring the tables provided by the Linux kernel firewall, as well as the chains and rules it stores. The kernel module currently used for iptables only applies to IPv4 traffic, to configure firewall rules for IPv6 connections instead use ip6tables, which responds to the same command structures as iptables.

Listing current rules

Ubuntu servers do not implement any restrictions by default, but for future reference, check the current iptable rules using the following command

```
sudo iptables -L
```

This will print out a list of three chains, *input*, *forward* and *output*, like the empty rules table example output below.

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

The chain names indicate which traffic the rules in each list will be applied to, *input* is for any connections coming to your cloud server, *output* is any leaving traffic and *forward* for any pass through. Each chain also has its *policy* setting which determines how the traffic is handled if it doesn’t match any specific rules, by default it’s set to *accept*.

Adding rules

Firewalls can commonly be configured in one of two ways, either set the default rule to accept and then block any unwanted traffic with specific rules, or by using the rules to define allowed traffic and blocking everything else. The latter is often the recommended approach, as it allows pre-emptively blocking traffic, rather than having to reactively reject connections that should not be attempting to access your cloud server.

To begin using iptables, you should first add the rules for allowed inbound traffic for the services you require. Iptables can track the state of the connection, so use the command below to allow established connections to continue.

```
sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
```

You can check that the rule was added using the same *sudo iptables -L* as before.

Next allow traffic to a specific port to enable SSH connections with the following

```
sudo iptables -A INPUT -p tcp --dport ssh -j ACCEPT
```

The *ssh* in the command translates to the port number 22, which the protocol uses by default. The same command structure can be used to allow traffic to other ports as well. To enable access to an HTTP web server, use the following command

```
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

After adding all the allowed rules you require, change the input policy to drop.

```
sudo iptables -P INPUT DROP
```

The same policy rules can be defined to other chains as well by entering the chain name and selecting either DROP or ACCEPT.

Saving and restoring rules

Now if you were to restart your cloud server all of these iptables configurations would be wiped. To prevent this, save the rules to a file.

```
sudo iptables-save > /etc/iptables/rules.v4
```

You can then simply restore the saved rules by reading the file you saved with

```
# Overwrite the current rules
sudo iptables-restore < /etc/iptables/rules.v4
# Add the new rules keeping the current ones
sudo iptables-restore -n < /etc/iptables/rules.v4
```

You can automate the restore process at reboot by installing an additional package for iptables which takes over the loading of the saved rules. To this with the following command

```
sudo apt-get install iptables-persistent
```

After the installation the initial setup will ask to save the current rules for IPv4 and IPv6, just select *Yes* and press enter for both.

If you make further changes to your iptables rules, remember to save them again using the same command as above. The iptables-persistent looks for the files *rules.v4* and *rules.v6* under */etc/iptables*.

These are just a few simple commands you can use with iptables, which is capable of much more. Read on to check on some of the other options available for more advanced control over iptable rules.

Advanced rule setup

As per basic firewall behaviour, the rules are read in the order they are listed on each chain, which means you’ll need to put the rules in the correct order. Appending new rules adds them to the end of the list. You can add new rules to a specific position of the list by inserting them using *iptables -I <index>*

-command, where the <*index*> is the order number you wish to insert the rule. To know which index number to enter, use the following command.

```
sudo iptables -L --line-numbers

Chain INPUT (policy DROP)
 num target prot opt source   destination
 1   ACCEPT all  --  anywhere anywhere ctstate RELATED,ESTABLISHED
 2   ACCEPT tcp  --  anywhere anywhere tcp dpt:ssh
 3   ACCEPT tcp  --  anywhere anywhere tcp dpt:http
```

The number at the beginning of each rule line indicates the position in the chain. To insert a new rule above a specific existing rule, simply use the index number of that existing rule. For example to insert a new rule to the top of the chain, use the following command with index number 1.

```
sudo iptables -I INPUT 1 -p tcp --dport 80 -j ACCEPT
```

If you wish to remove an existing rule from a certain chain, use the delete command with the parameter *-D*. The easiest way to select the rule for delete is to use the index numbers explained above. For example to delete the second rule on the input chain, use this command

```
sudo iptables -D INPUT 2
```

It’s also possible to flush all rules of a specific chain or even the whole iptables using the *-F* -parameter. This is useful if you suspect iptables is interfering with your attempted network traffic, or you simply wish to start configuring again from a clean table. Remember to save the rules to a file before flushing the table.

```
# Clear input chain
sudo iptables -F INPUT
# Flush the whole iptables
sudo iptables -F
```

With the iptable flushed, your server could be vulnerable to attacks. Make sure to secure your system with an alternative method while disabling iptables even temporarily.