iptables Tutorial

Saurabh Barjatiya
IIIT Hyderabad

24^{th} July, 2009

Contents

1 Introduction

It has become necessary to use firewall on our personal computers and servers to reduce the available ports for attack. '*iptables*' is default firewall that comes with most modern GNU/Linux distributions. 'iptables' is very powerful firewall and it can be customized to suit our needs very easily.

For defining rules 'iptables' uses concepts of tables and chains. We can add or remove rules to chains. Chains can be added or removed from tables. There are few default tables and chains which cannot be removed. The default tables that come with iptables are filter, nat, mangle and raw[1]. (Note: Table and chain names are case-sensitive). Since this is introductory tutorial we will look only at filter table.

In filter table there are three pre-defined chains INPUT, FORWARD and OUTPUT. As the name suggest filter table is used to filter traffic coming to system(INPUT), being forwarded by system possibly to other network (FORWARD) and going out from system (OUTPUT). Since most of the time since PCs and servers have only one interface connected to network, we usually reject everything on FORWARD chain. Also it is assumed in most cases that outgoing connections from the PC can be trusted as we or our applications have initiated the connection. Therefore, we do not add any rules to OUTPUT chain in normal scenario.

Hence most of the firewall rules that we add go to INPUT chain of filter table.

2 Rule processing

'iptables' follows rules in particular order. So it is important for us to specify rules in correct order so that firewall functions as desired. Like most firewalls 'iptables' reads rules in chains starting from first rule. As soon a rules is matched action specified for that rules is performed. If action is final action like ACCEPT, DROP, REJECT, etc then firewall stops processing rules for that packet. However if rules does not matches or the action is to go to other chain, mark packet, etc. then firewall continues processing.

So if we want firewall to reject connections from all IPs except a few then we have to add rules like

1. Allow packets from first special IP
2. Allow packets from other special IP

[1]The tutorial has been written as per iptables version iptables v1.3.5. But most of the things written in this tutorial should be valid for almost all versions of iptables.

3. Reject all packets

On the other hand if we want to accept connections from all IPs except a few bad IPs then we can add rules like

1. Drop packets from first bad IP
2. Drop packets from other bad IP
3. Accept all packets

It is important to mention here that unlike most firewalls (esp. hardware firewalls) if no rule matches the packet the iptables defaults to accepting the packet. Hence if there are no firewall rules then iptables will accept all incoming and outgoing packets.

3 Rule syntax

There are various options that can be specified in iptables firewall rules. If we specify more than one option then options get AND'ed together, that is only if all the criteria specified in the rule match then action specified in the rule is carried out. Note that even if one criteria fails to match then action will not be taken and processing from next rule in chain (if present) will continue. Some very common options that we can use to define iptables rules are:

`-A <Chain> :` This option implies that rules must be appended (added to end) of specified chain. Hence rules added using -A become last rule in the chain.

`-I <Chain> :` This option implies that rules must be inserted at top of the rules list. Hence rules added using -I become first rule in the chain.

`-D <Chain> :` This option can be used to delete rule from the chain. 'iptables' rules can be addeded / removed on the fly without stopping firewall. Hence this option is very useful if we want to remove some rule from already running firewall without restarting the firewall.

`-s <Source IP> :` This option can be used to specify source IP of the packet. Only if source IP address of packet matches IP address specified in this rule some action would be taken, else processing will continue.

`-d <Destination IP> :` Similar to -s for specifying source IP address, this option can be used to specify destination IP of the packet.

`-p <Protocol> :` This can be used to specify transport layer protocol running over IP protocol. The most common protocol options that are used in firewall rules are udp and tcp.

`-j <Target> :` This is used to specify target of the rule, that is what action should be taken if all criteria specified in rule are satisfied. The most common targets used are ACCEPT, REJECT and DROP.

`-i <Interface> :` This is used to specify interface from which packet has entered the machine.

`-o <Interface> :` This is used to test against interface from which packet would leave the machine, if it is allowed to go.

Please refer to man page of iptables which can be seen using command '`man iptables`', which should work on most distributions. The man page will contain list of valid tables, chains, options, targets, etc. for the version of iptables installed on your system.

4 iptables modules

Since there can be too many parameters that one would want to match again headers / contents of the packet, the options which are less common are grouped in iptables modules. Modules also help in grouping of similar options. Many modules come with default iptables installation. Some important ones are discussed here.

4.1 tcp

Since the default options allow us only to match only source and destination address of the packet, we still need way to match packets against source and destination ports. To match packets against such specific parameters like source and destination ports, we can use protocol modules which allow us to specify ports we want to match. Hence whenever we want to specify such specific parameters we have to tell iptables to load modules which provides us those parameters.

For example, tcp module provides us options of matching source and destination port numbers. So in iptables rule before we can use '`--sport <source port>`' or '`--dport < destination port >`' we must specify to load tcp module using '`-m tcp`'. After we specified that we are going to use tcp module, we can specify the specific source or destination port or both in the rule.

4.2 udp

Very similar to tcp module we also have udp module. Once we include udp module using '`-m udp`', we can specify source and destination udp ports using options '`--sport <Source Port>`' and '`--dport <Destination Port>`'.

4.3 state

Like tcp and udp module that allow us to specify parameters like source and destination port, we have module named state that allows us to specify state of the TCP connection. Since TCP is connection based protocols, a TCP connection can be in various different states. To apply firewall rules to TCP connections based on the state of the connection, we can use state module. This module allows us to specify state of the connection using '`--state <Connection State>`', where valid connection states are `NEW`, `RELATED` and `ESTABLISHED`. Generally we either specify state as `NEW` or we specify `RELATED`, `ESTABLISHED` together to specify an already running connection using syntax '`--state RELATED, ESTABLISHED`'.

5 iptables Rules

Some sample IP tables rules are:

- `-A INPUT -j ACCEPT`
  This rule can be used to accept all incoming packets.
- `-A INPUT -s 10.10.10.10 -j REJECT`
  This rule can be used to reject all packets from source IP address 10.10.10.10.
- `-A OUTPUT -d 10.20.20.20 -j REJECT`
  This rule can be used to block all outgoing packets to IP address 10.20.20.20.
- `-A INPUT -p tcp --dport 22 -j ACCEPT`
  This rule can be used to accept all connections to destination port 22. This can be used to specify that incoming SSH connections are allowed.
- `-A FORWARD -j REJECT`
  This rule can be used to block all packets which are supposed to be forwarded. Note that IP forwarding is disabled on most Linux distributions by default. Even if we enable IP forwarding and block the

forwarind using firewall then all packets will be blocked from being forwarded. Both OS and iptables should be configured to allow packet forwarding then only packets will get forwarded from one interface to another.

- `-A INPUT -m state --state RELATED, ESTABLISHED -j ACCEPT`
  This rule can be used to accept packets of all running connections. Hence this rule will allow all packets except the ones which start new connection or are unrelated to any packet that firewall has seen so far.

- `-A INPUT -m tcp --dport 22 -m state --state NEW -j ACCEPT`
  This rule can be used to accept all new incoming connections on port 22. This is used to illustrate that we can use more than one iptables module in same rule.

6 Sample iptables configurations

Few complete iptables configurations are discussed in this section. The configurations are intentionally kept small and simple for learning purposes.

6.1 Allow only SSH

To allow only input SSH connections and loopback connections, we can use following configuration.

```
-A INPUT -p tcp -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT
-A FORWARD -j REJECT
```

6.2 Allow only HTTP/HTTPS

To allow only input HTTP/HTTPS connections and loopback connections, we can use following configuration.

```
-A INPUT -p tcp -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 443 -j ACCEPT
-A INPUT -j REJECT
-A FORWARD -j REJECT
```