



# **IPTABLES**

## **Quick Guide**

BEU CYBER

Mohamed Ben Lakhdim

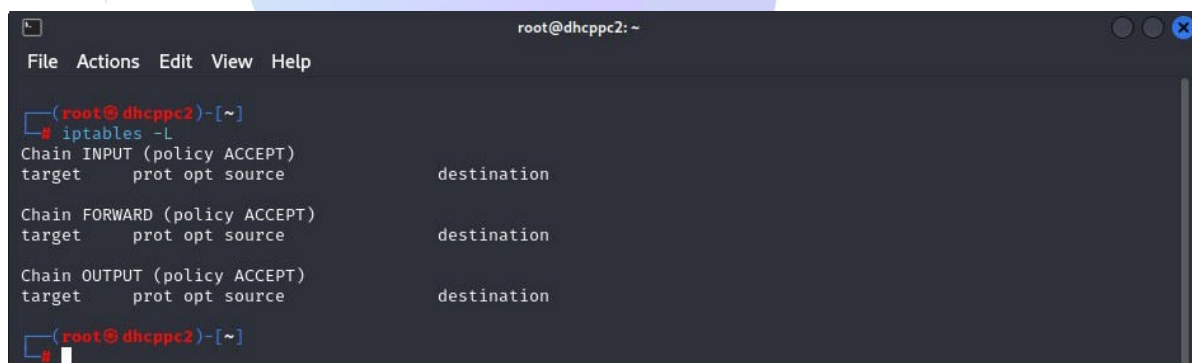
<https://www.linkedin.com/mwlite/in/mohamed-ben-lakhdim-b50a0125a>  
<https://www.linkedin.com/company/beu-cyber>

# 1. Definition

IPTABLES is packet filtering firewall on LINUX that allows you to control traffic coming in and out of your network. It's like a security guard that only lets people through based on your set of rules. Iptables is a powerful tool that can be used to protect your network or devices from potential security threats.

Iptables is composed of three chains:

- 1.INPUT: The INPUT chain is used to handle the incoming traffic to the server. This chain specifies the rules for the incoming packets that are destined for the server.
- 2.OUTPUT: The OUTPUT chain is used to handle the outgoing traffic from the server. This chain specifies the rules for the packets that are generated by the server and are going out.
- 3.FORWARD: The FORWARD chain is used to handle the traffic that is passing through the server. This chain specifies the rules for the packets that are not destined for the server and are just being forwarded by the server.

A terminal window titled 'root@dhcppc2: ~' showing the output of the 'iptables -L' command. The output lists three chains: INPUT, FORWARD, and OUTPUT, all with a policy of ACCEPT. Each chain has a target rule for 'destination' with protocol, option, and source fields.

```
root@dhcppc2: ~  
File Actions Edit View Help  
(root@dhcppc2)~  
# iptables -L  
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  
(root@dhcppc2)~  
#
```

As you can see the policy is set to **ACCEPT** that means we accept every package, and we don't filter anything which may be dangerous on our device or network

## 2. HOW TO USE IPTABLES

To use iptables you should know some principles:

- know your system!
- know your enemy (we will cover this in the last paragraph)
- Protection is key.

Before starting to use iptables and to prevent to lose all your progress you must keep a backup the best way to do it is by adding a timestamp to the backup file `/sbin/iptables-save > /root/iptables-works-`date +%F`` So, if you did something that prevents your system from working and trust me you probably will all you have to do is a simple command to restore it.

`/sbin/iptables-save < \ /root/iptables-works-"date"`

So now we can start to experiment on our iptables there is some basic commands you should know:

`iptables -A` is used to assert a rule as the last in table

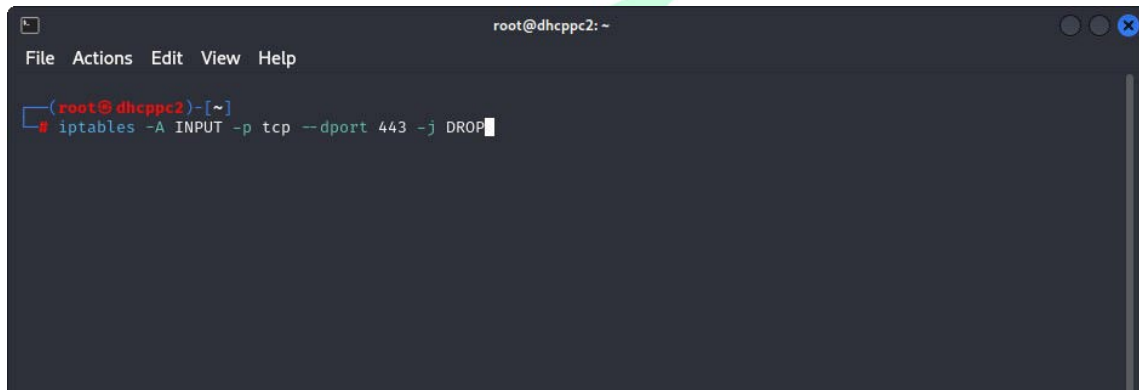
`iptables -I` is used to insert as the first rule in table

`iptables -F` is used to delete all rules from the tables

`iptables -L` is used to list all the rules in the tables

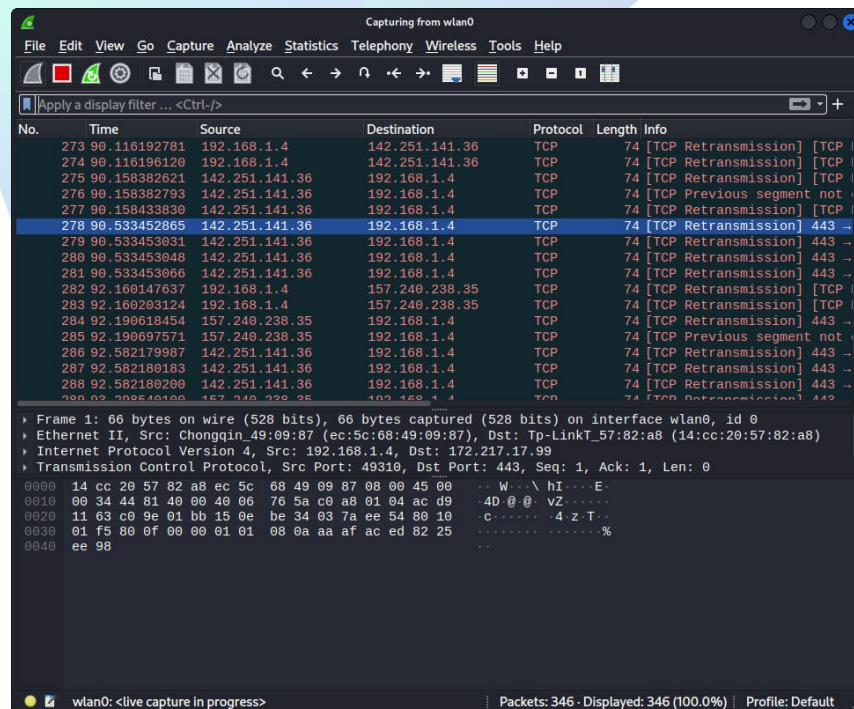
`iptables -D` is used to delete a specific rule from the table.

So, let's write our first command in iptables and see what's going to happen: **iptables -A INPUT -p tcp --dport 443 -j DROP**



```
root@dhcppc2: ~  
File Actions Edit View Help  
(root@dhcppc2)-[~]  
* iptables -A INPUT -p tcp --dport 443 -j DROP
```

What this command does is to drop all TCP protocol packages which destination port is 443. Let's see what happens if we try to enter a site.



All packages just got dropped. And we cannot reach our website.

Let's imagine a scenario where you think an Ip is malicious, and you are trying to stop him from reaching your network!

```
iptables -I INPUT -s <ipadress> -j DROP
```

Now let's imagine that you have some maintenance, and you need to drop all traffic of UDP/TCP for only a certain time, so it doesn't get disrupted.

```
iptables -I INPUT -p tcp -m time --timestart <hh:mm> --  
timestop <hh:mm> -j DROP
```

```
iptables -I INPUT -p udp -m time --timestart <hh:mm> --  
timestop <hh:mm> -j DROP
```

There is a really important option in iptables that can limit the access of Ip addresses the two options are limit and connlimit "connlimit" sets a limit on the number of connections that can come from a single IP address at the same time.

"limit" sets a limit on the speed at which incoming traffic from a specific IP address can come in. This can be measured in either the number of packets or the amount of data per second.

These limits will protect the system from being overwhelmed by too much traffic and is a good defense against Dos attacks.

### 3. HOW TO DEFEND YOUR SYSTEM

We will begin this part by explaining the two tactics of defending your system with iptables :

“whitelisting”: is only allowing traffic from specific set of Ip addresses. Only the traffic of these addresses is allowed to reach the system.

“blacklisting” : is the process of explicitly blocking incoming traffic from a specific set of IP addresses. Traffic from these IP addresses is prevented from reaching the system .

A combination of two tactics may be the best way to defend your system.

First of all you should put specific rules at the top of your tables and avoid putting generic rules at the top so you don't lock yourself out for example don't put rules like this one at the top `iptables -I INPUT -p tcp --dport 443 -j DROP` and try to specify as you can in your commands. And don't forget to whitelist your Ip address at the top of your table this a very effective method to not get locked out `iptables -I INPUT -s <ipaddress> -j ACCEPT.`

So now let's write a basic set of rules and try understand each one and what it does and then we try to see if it really protects our system lets set default policy as DROP

```
iptables --policy INPUT DROP
```

```
iptables --policy OUTPUT DROP
```

```
iptables --policy FORWARD DROP
```



```
root@dhcpc2: ~  
File Actions Edit View Help  
  
(root@dhcpc2)-[~]  
# iptables -L  
Chain INPUT (policy DROP)  
target      prot opt source      destination  
  
Chain FORWARD (policy DROP)  
target      prot opt source      destination  
  
Chain OUTPUT (policy DROP)  
target      prot opt source      destination  
  
(root@dhcpc2)-[~]  
#
```

As we can see all policy are marked as DROP

now we start to whitelist all connections we are going to accept , remember we should only let the connections that let the service function effectively not more nor less

If a connection is established or related, we are going to let it go through.

```
iptables -I INPUT 1 -m state --state RELATED,ESTABLISHED  
-j ACCEPT
```

```
iptables -I OUTPUT 1 -m state --state  
RELATED,ESTABLISHED -j ACCEPT
```

This a basic command to allow loop back connection

```
iptables -A OUTPUT -i lo -j ACCEPT
```

```
iptables -A INPUT -i lo -j ACCEPT
```

This rule allows connection to DHCP so we can get an Ip address subnet mask ...

```
iptables -A OUTPUT -p udp --dport 67:68 --sport 67:68 -j ACCEPT
```

This rule will allow inbound of SSH so you can make changes on the system.

```
iptables -A INPUT -p tcp -m tcp --dport 22 -m state --state NEW -j ACCEPT
```

This command will allow you to send email

```
iptables -A INPUT -p tcp -m tcp --dport 25 -m state --state NEW -j
```

```
iptables -A OUTPUT -p tcp -m tcp --dport 25 -m state --state NEW -j ACCEPT
```

This rule is used to allow the DNS lookups

```
iptables -A OUTPUT -p udp -m udp --dport 53 -j ACCEPT
```

This rule will let you send ping packages

```
iptables -A OUTPUT -p icmp -j ACCEPT
```

This rule is for the NTP or Network Time Protocol which is critical for the functioning of the system

```
iptables -A OUTPUT -p udp --dport 123 --sport 123 -j ACCEPT
```

This the last rule of our example we allow outbound of http and https



```
iptables -A OUTPUT -p tcp -m tcp --dport 443 -m state --state  
NEW -j
```

```
iptables -A OUTPUT -p tcp -m tcp --dport 80 -m state --state  
NEW -j
```

This set of rules is basic but it's really secure for someone's laptop that doesn't want to get his security compromised. It needs more modifications but for a start its decent it lets little things in and little things out.

So now let's see how to defend our system against special attacks:

Let's test our system security by doing Nmap scans on it so we can see what happens.

First test is a ping scan let's see what's the result will be

```
Starting Nmap 7.91 ( https://nmap.org ) at 2023-02-26 15:10 EST  
Nmap scan report for 192.168.1.2  
Host is down.  
  
Nmap done: 1 IP address (0 hosts up) scanned in 0.14 seconds
```

As you can see, after the scan the host is down because in our rules, we don't let the ping package come in. Let's see some other types of scans.

There is a famous Nmap scan called null scan which Nmap sends TCP packets with no flags set let's do a scan on port 22.

```
Host is up (0.043s latency).  
PORT      STATE      SERVICE  
22/tcp    open      ssh  
  
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
```

**iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP**

this is how we use the iptables to just stop certain TCP connections that seems suspicious.

One of the most dangerous attacks is a Dos(Denial-of-service) attacks that tries to flood our system with traffic or requests so we have a trick to stop it using limit module :

**iptables -A INPUT -p tcp --syn -m limit --limit 20/s -j ACCEPT**

so we just limit the connections to 20 connections per second .