CYBERSECURITY SEMINARS Google.org, 1st Cycle

6th Assignment - Firewalls

Full Name: Klajdi Cami

Exercise 1 for Firewall

Which of the aforementioned rules is activated to check requests according to the network topology in each of the following scenarios?

1. A TCP/IP packet requesting access from the Internet to the Web server for HTTP

2. A TCP/IP packet requesting access from Host2 to Host1 for HTTP

3. A TCP/IP packet requesting access from the DBserver to the Web server for HTTP

4. A TCP/IP packet requesting access from the Internet to the DBserver for SQL

5. A TCP/IP packet requesting access from Host2 to the DBserver for SQL as incoming traffic from the Internet

6. A TCP/IP packet requesting access from Host1 to the Web Server for HTTP

Answer:

Rule	Προέλευση	Προορισμός	Θύρα/Υπηρεσία	Ενέργεια
1	Εσωτερικό δίκτυο	Οπουδήποτε	όλες	allow
2	Host2	DBserver	1433 / SQL	allow
3	Οπουδήποτε	Web server	80 / HTTP	allow
4	Οπουδήποτε	Οπουδήποτε	όλες	deny

Image 1: *Firewall Rules*

1. A TCP/IP packet requesting access from the Internet to the Web server for HTTP

According to the firewall rules, Rule #3 applies, which allows incoming traffic from anywhere to the Web Server, exclusively for the HTTP service (port 80).

2. A TCP/IP packet requesting access from Host2 to Host1 for HTTP

Rule #2 allows incoming traffic from Host2 but only to the DBServer and exclusively for the SQL service on port 1433.

Rule #3 allows incoming traffic to the Web Server only for the HTTP service on port 80.

There is no rule explicitly allowing communication from Host2 to Host1 for HTTP.

Rule #4 denies all traffic not covered by the previous rules.

Thus, the packet will be rejected, as there is no rule explicitly permitting communication between Host2 and Host1 for HTTP.

3. A TCP/IP packet requesting access from the DBServer to the Web server for HTTP

Rule #3 allows incoming traffic from anywhere to the Web Server, but only for the HTTP service (port 80).

Although the DBServer is part of the internal network, it is still covered by the term “Anywhere” as a source.

The request is destined for the Web Server and concerns port 80, thus meeting the conditions of Rule #3.

Therefore, the packet will be allowed by Rule #3, as communication from the DBServer to the Web Server via HTTP (port 80) is permitted.

4. A TCP/IP packet requesting access from the Internet to the DBServer for SQL

Rule #2 allows incoming traffic to the DBServer, but only from Host2 and only for the SQL service (port 1433).

This particular request originates from the Internet, not from Host2.

There is no other rule allowing incoming traffic from the Internet to the DBServer.

Rule #4 states that all packets not matching any of the above rules are rejected.

Therefore, the packet will be rejected by Rule #4, as there is no rule allowing incoming traffic from the Internet to the DBServer for SQL (port 1433).

5. A TCP/IP packet requesting access from Host2 to the DBServer for SQL as incoming traffic from the Internet

Rule #2 allows incoming traffic only from Host2 to the DBServer, exclusively for the SQL service (port 1433).

This specific request originates from Host2, the destination is the DBServer, and the requested service is SQL (port 1433), which exactly matches Rule #2.

Therefore, the packet will be allowed by Rule #2, as communication from Host2 to the DBServer for SQL (port 1433) is explicitly permitted.

6. A TCP/IP packet requesting access from Host1 to the Web Server for HTTP

Rule #3 allows incoming traffic from anywhere to the Web Server, but only for the HTTP service (port 80).

Host1 belongs to the local network, but since the rule allows traffic from anywhere, the request is covered.

The destination port is 80 (HTTP), which matches the rule.

Thus, the packet will be allowed by Rule #3, as the request meets all conditions (origin from anywhere, destination to the Web Server, use of port 80 for HTTP).

Exercise 2 for IPTABLES

Blocking a Specific IP Address

1. Open the two Ubuntu virtual machines you have created.

2. Apply a rule on one of the two machines to block communication between them based on their IP addresses.

3. Confirm using the ping command that there is no connectivity between the two machines.

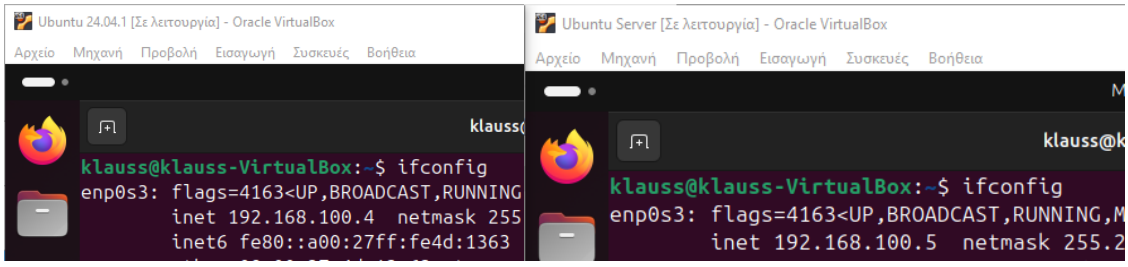
4. Remove the rule from the list and confirm that connectivity between them has been restored.

Answer:

First, we determine the IP addresses of the two systems using the command:

ifconfig

In our case, the client has the IP address **192.168.100.4**, and the server has the IP address **192.168.100.5**.



Then, we check the connection between the two machines. This can be easily done by executing the following command from one of the two systems (e.g., from the client):
ping -c 4 192.168.100.5

We observe that there is connectivity.

```
klauss@klauss-VirtualBox:~$ ping -c 4 192.168.100.5
PING 192.168.100.5 (192.168.100.5) 56(84) bytes of data.
64 bytes from 192.168.100.5: icmp_seq=1 ttl=64 time=0.774 ms
64 bytes from 192.168.100.5: icmp_seq=2 ttl=64 time=0.761 ms
64 bytes from 192.168.100.5: icmp_seq=3 ttl=64 time=0.669 ms
64 bytes from 192.168.100.5: icmp_seq=4 ttl=64 time=0.783 ms

--- 192.168.100.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3287ms
rtt min/avg/max/mdev = 0.669/0.746/0.783/0.045 ms
klauss@klauss-VirtualBox:~$
```

On the server side, to block communication with the client, we execute the following command:

sudo iptables -A INPUT –s 192.168.100.4 –j DROP

```
klauss@klauss-VirtualBox:~$ sudo iptables -A INPUT -s 192.168.100.4 -j DROP
klauss@klauss-VirtualBox:~$ sudo iptables -v --list
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

    0     0 DROP       all  --  any    any     192.168.100.4        anywhere


Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

To confirm that there is no connectivity between the client and the server, we run the ping command again from the client side:

ping -c 4 192.168.100.5

We observe that the system sent 4 packets but did not receive any response, which means that the connection has indeed been blocked.

```
klauss@klauss-VirtualBox:~$ ping -c 4 192.168.100.5
PING 192.168.100.5 (192.168.100.5) 56(84) bytes of data.

--- 192.168.100.5 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3497ms
```

To restore the connection, we execute the following command:
sudo iptables –D INPUT 1

This will delete the rule that prevents the connection.

```
klauss@klauss-VirtualBox:~$ sudo iptables -v --list
Chain INPUT (policy ACCEPT 33 packets, 5496 bytes)
 pkts bytes target     prot opt in     out     source               destination

    9   745 DROP       all  --  any    any     192.168.100.4        anywhere


Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

klauss@klauss-VirtualBox:~$ sudo iptables -D INPUT 1
klauss@klauss-VirtualBox:~$ sudo iptables -v --list
Chain INPUT (policy ACCEPT 36 packets, 5745 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

If we run the ping command again from the client, we see that the connection has been restored successfully:

```
klauss@klauss-VirtualBox:~$ ping -c 4 192.168.100.5
PING 192.168.100.5 (192.168.100.5) 56(84) bytes of data.
64 bytes from 192.168.100.5: icmp_seq=1 ttl=64 time=0.455 ms
64 bytes from 192.168.100.5: icmp_seq=2 ttl=64 time=0.699 ms
64 bytes from 192.168.100.5: icmp_seq=3 ttl=64 time=0.822 ms
64 bytes from 192.168.100.5: icmp_seq=4 ttl=64 time=0.762 ms

--- 192.168.100.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3508ms
rtt min/avg/max/mdev = 0.455/0.684/0.822/0.139 ms
```

Blocking a Specific MAC Address

1. Open the two Ubuntu virtual machines you have created.

2. Apply a rule on one of the two machines to block communication between them based on the MAC address.

3. Confirm using the ping command that there is no connectivity between the two machines.

4. Remove the rule from the list and confirm that connectivity between them has been restored.

Answer:

First, we execute the command:

ifconfig

on the client side to find the system's MAC address, which in this case is **08:00:27:4d:13:63**.

```
klauss@klauss-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.100.4  netmask 255.255.255.0  broadcast 192.168.100.255
        inet6 fe80::a00:27ff:fe4d:1363  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:4d:13:63  txqueuelen 1000  (Ethernet)
```

Then, we execute the following command on the server side:

sudo iptables -A INPUT -m mac --mac-source 08:00:27:4d:13:63 -j DROP

This command blocks the connection based on the MAC address.

```
klauss@klauss-VirtualBox:~$ sudo iptables -A INPUT -m mac --mac-source 08:00:27:4d:13:6
3 -j DROP
[sudo] password for klauss:
klauss@klauss-VirtualBox:~$ sudo iptables -v --list
Chain INPUT (policy ACCEPT 66 packets, 11699 bytes)
 pkts bytes target     prot opt in     out     source               destination

    0     0 DROP       all  --  any    any     anywhere             anywhere
  MAC 08:00:27:4d:13:63

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

We confirm from the client side that there is no connectivity by running the command:

ping -c 4 192.168.100.5

```
klauss@klauss-VirtualBox:~$ ping -c 4 192.168.100.5
PING 192.168.100.5 (192.168.100.5) 56(84) bytes of data.

--- 192.168.100.5 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3201ms
```

To remove the rule from the server, we execute:

sudo iptables –D INPUT 1

```
klauss@klauss-VirtualBox:~$ sudo iptables -v --list
Chain INPUT (policy ACCEPT 66 packets, 11699 bytes)
 pkts bytes target     prot opt in     out     source               destination

    0     0 DROP       all  --  any    any     anywhere             anywhere
  MAC 08:00:27:4d:13:63

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

klauss@klauss-VirtualBox:~$ sudo iptables -D INPUT 1
klauss@klauss-VirtualBox:~$ sudo iptables -v --list
Chain INPUT (policy ACCEPT 74 packets, 13364 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

Finally, we confirm that the connection has been restored.

```
klauss@klauss-VirtualBox:~$ ping -c 4 192.168.100.5
PING 192.168.100.5 (192.168.100.5) 56(84) bytes of data.
64 bytes from 192.168.100.5: icmp_seq=1 ttl=64 time=1.50 ms
64 bytes from 192.168.100.5: icmp_seq=2 ttl=64 time=0.706 ms
64 bytes from 192.168.100.5: icmp_seq=3 ttl=64 time=1.37 ms
64 bytes from 192.168.100.5: icmp_seq=4 ttl=64 time=0.813 ms

--- 192.168.100.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3026ms
rtt min/avg/max/mdev = 0.706/1.098/1.502/0.343 ms
```

Blocking a Specific URL Address

1. Open the Ubuntu virtual machine you have created.

2. Open the browser and confirm that you can access the website **in.gr**.

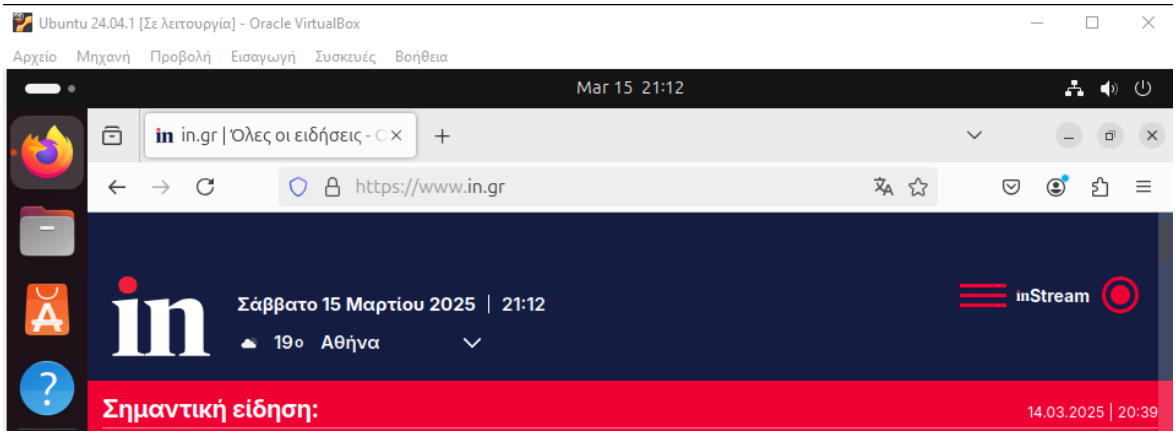
3. Apply a rule to block incoming traffic from this specific website.

4. Confirm through the browser that no data is received from the website.

5. Remove the rule from the list and confirm that connectivity to the website has been restored.

Answer:

First, we observe that we can initially access the website normally.



To block incoming traffic from the specific website, we execute:

sudo iptables -A OUTPUT -p tcp -m string --string "URL_HERE" -- algo kmp -j DROP

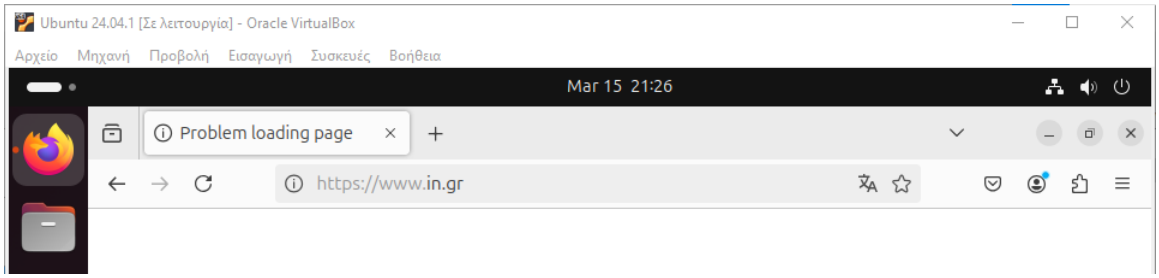
```
klauss@klauss-VirtualBox:~$ sudo iptables -A OUTPUT -p tcp -m string --string "www.in.g
r" --algo kmp -j DROP
klauss@klauss-VirtualBox:~$ sudo iptables -v --list
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain OUTPUT (policy ACCEPT 4195 packets, 532K bytes)
 pkts bytes target     prot opt in     out     source               destination

    0     0 DROP       tcp  --  any    any     anywhere             anywhere
  STRING match  "www.in.gr" ALGO name kmp
```

We confirm through the browser that no data is received from the specific website.



Finally, we remove this rule by executing the command:

sudo iptables -D OUTPUT 1

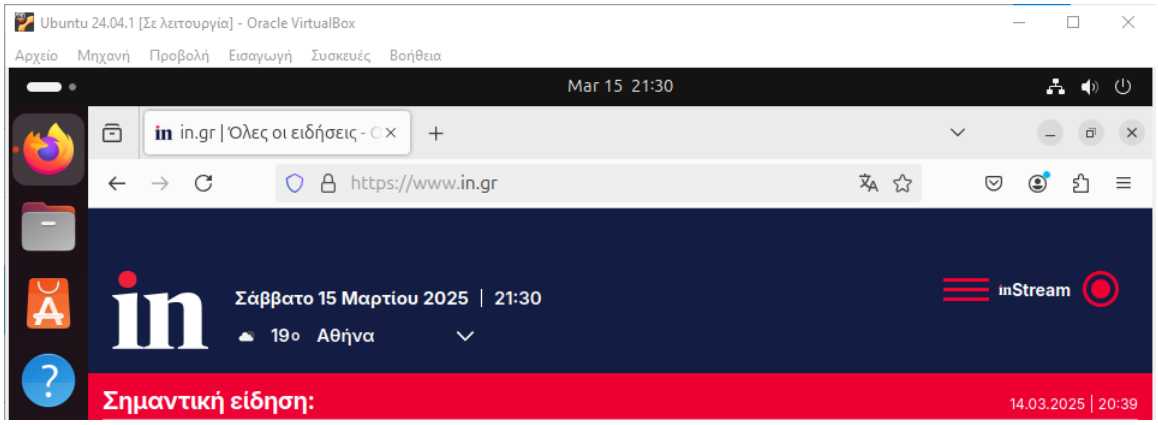
```
klauss@klauss-VirtualBox:~$ sudo iptables -D OUTPUT 1
klauss@klauss-VirtualBox:~$ sudo iptables -v --list
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain OUTPUT (policy ACCEPT 5335 packets, 683K bytes)
 pkts bytes target     prot opt in     out     source               destination

klauss@klauss-VirtualBox:~$
```

and confirm that connectivity to the specific website has been restored.



Blocking a Specific Destination Port

1. Open the Ubuntu virtual machine you have created.

2. Open the browser and confirm that you can connect to a website using an HTTPS connection.

3. Close the browser.

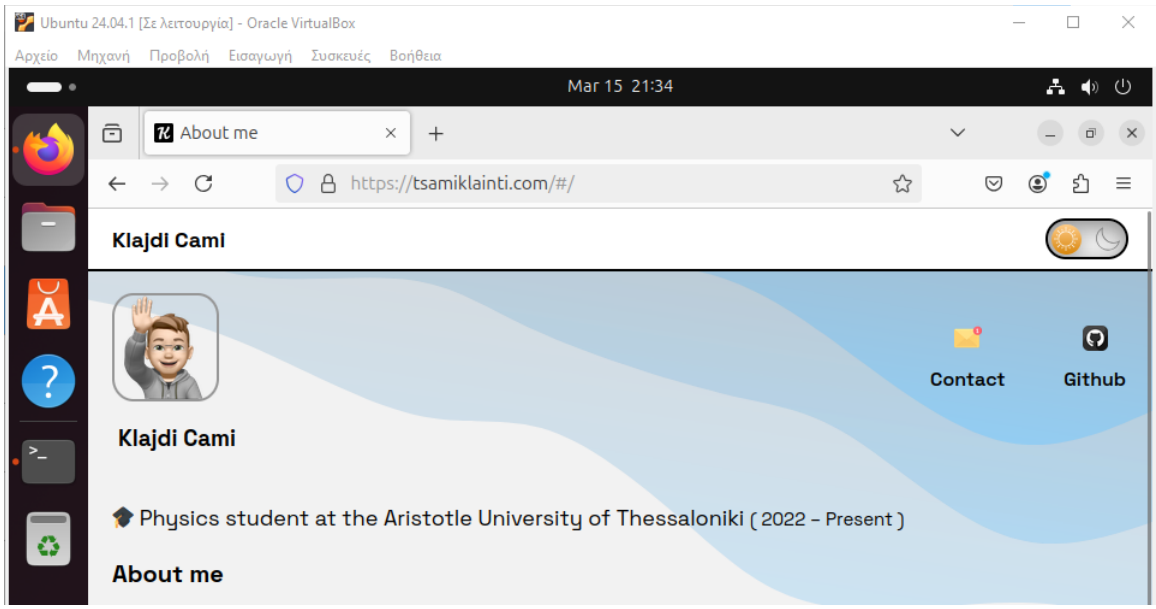
4. Apply a rule to block outgoing HTTPS traffic with destination port 443.

5. Confirm through the browser that you cannot connect to any website, as HTTPS traffic on port 443 has been blocked—meaning no website using HTTPS can be accessed.

6. Remove the rule from the list and confirm that connectivity has been restored to the specific website or any other website using the HTTPS protocol and port 443.

Answer:

First, we open the browser and confirm that we can connect to a website using HTTPS.



To block outgoing HTTPS traffic with destination port 443, we execute the following command:

sudo iptables -A OUTPUT -p tcp --dport 443 -j DROP

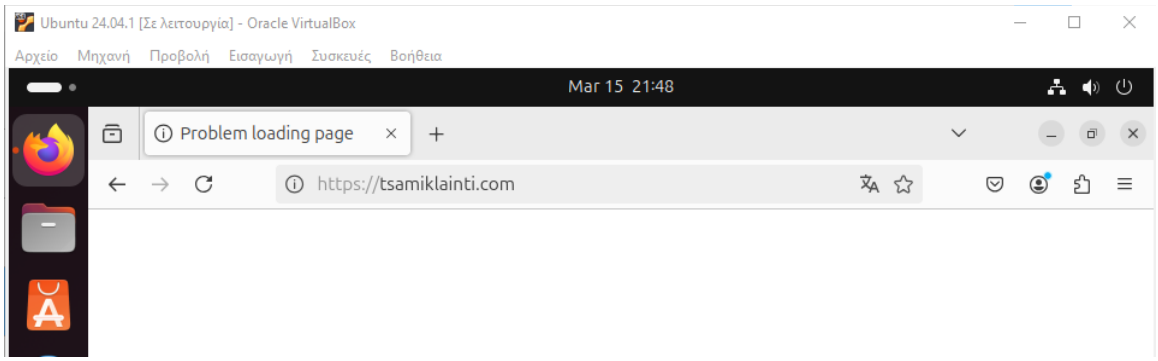
```
klauss@klauss-VirtualBox:~$ sudo iptables -A OUTPUT -p tcp --dport 443 -j DROP
klauss@klauss-VirtualBox:~$ sudo iptables -v --list
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 9233 packets, 1138K bytes)
 pkts bytes target     prot opt in     out     source               destination

    0     0 DROP       tcp  --  any    any     anywhere             anywhere
  tcp dpt:https
```

We confirm through the browser that we cannot connect to any website since outgoing HTTPS traffic on destination port 443 has been blocked.



We remove the rule from the list by executing the command:

sudo iptables -D OUTPUT 1

```
klauss@klauss-VirtualBox:~$ sudo iptables -v --list
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 10586 packets, 1334K bytes)
 pkts bytes target     prot opt in     out     source               destination

klauss@klauss-VirtualBox:~$
```

And we confirm that connectivity to the specific website has been restored.