

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ  
Τμήμα Πληροφορικής



Εργασία Μαθήματος **Ασφάλεια Δικτύων και Επικοινωνιών**

Αρ. Άσκησης – Τίτλος Άσκησης	<b>3η Άσκηση - Υλοποίηση πολιτικής ασφάλειας δικτύου με firewall και IDS</b>
Όνομα φοιτητή – Αρ. Μητρώου (όλων σε περίπτωση ομαδικής εργασίας)	Ανδριανόπουλος Βασίλειος ΜΠΚΕΔ2303
	Τσουτσουλιανούδης Γεώργιος ΜΠΚΕΔ2347
Ημερομηνία παράδοσης	01-03-2024



## Περιεχόμενα

Περιεχόμενα .....	2
1 Αρχικοποίηση συστημάτων .....	3
1.1 Τοπολογία .....	3
1.2 Παραμετροποίηση κίνησης του δικτύου .....	4
2 Παραμετροποίηση κανόνων στο iptables .....	7
2.1 Βασική αντιμετώπιση προβλημάτων δικτύου .....	8
2.2 Ενεργοποίηση της εταιρικής κίνησης δικτύου στο Internet .....	10
2.3 Υλοποίηση πολιτικής ασφάλειας για το εσωτερικό δίκτυο ITZ .....	10
2.4 Ανακατεύθυνση επισκεψιμότητας σε υπηρεσία web στο DMZ .....	12
2.5 Αυτόματη εκκίνηση κανόνων .....	15
2.6 Application Layer Firewall .....	16
3.1 Έλεγχος Κανόνων .....	23
4 Δοκιμή επιθέσεων και υλοποίηση κανόνων αποτροπής .....	27
4.1 Υλοποίηση επίθεσης .....	27
4.2 Αποτροπή επίθεσης .....	27
5 Ανίχνευση επιθέσεων με τη χρήση IDS .....	29
5.1 FTP BruteForce attack .....	34
5.2 HTTP FLOOD ATTACK .....	36



## 1 Αρχικοποίηση συστημάτων

### 1.1 Τοπολογία

Όπως αναφέρει και η εκφώνηση, το VM1 (Gateway) έχει 3 network interfaces. Πιο συγκεκριμένα το eth0, που είναι bridged με το οικιακό δίκτυο, το eth1 που είναι host-only και βρίσκεται στο ίδιο δίκτυο με το VM2 (ITZ) όπου είναι και αυτό host-only και το eth2 που είναι host-only και βρίσκεται στο ίδιο δίκτυο με το VM3 (Public Host) όπου είναι και αυτό host-only.

```
(root@kali)-[/etc/iptables]
# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.31.41 netmask 255.255.255.0 broadcast 192.168.31.255
    inet6 fe80::2106:513c:d1c1:b7fa prefixlen 64 scopeid 0<20<link>
    ether 08:00:27:21:b1:d0 txqueuelen 1000 (Ethernet)
    RX packets 1379303 bytes 1869059067 (1.7 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 247472 bytes 17136270 (16.3 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.3 netmask 255.255.0.0 broadcast 10.0.255.255
    inet6 fe80::42d6:9444:4900:2080 prefixlen 64 scopeid 0<20<link>
    ether 08:00:27:97:91:3c txqueuelen 1000 (Ethernet)
    RX packets 202 bytes 23072 (22.5 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 39 bytes 6381 (6.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.1.0.2 netmask 255.255.0.0 broadcast 10.1.255.255
    inet6 fe80::cce6:95f:500c:ec prefixlen 64 scopeid 0<20<link>
    ether 08:00:27:4b:ba:00 txqueuelen 1000 (Ethernet)
    RX packets 247477 bytes 17132740 (16.3 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1352803 bytes 3213370122 (2.9 GiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0<10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 2 bytes 140 (140.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2 bytes 140 (140.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Εικόνα 1: Network Interfaces of VM1 (Gateway)



```
(root@kali)-[/home/kali]
# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.6 netmask 255.255.0.0 broadcast 10.0.255.255
    inet6 fe80::e48a:8400:809f:e539 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:59:43:04 txqueuelen 1000 (Ethernet)
    RX packets 4 bytes 866 (866.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 22 bytes 3034 (2.9 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 4 bytes 240 (240.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4 bytes 240 (240.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Εικόνα 2: Network Interfaces of VM2

## 1.2 Παραμετροποίηση κίνησης του δικτύου

Παραμετροποιούμε το VM1, ώστε να παίζει το ρόλο του Gateway.

Αρχικά, ενεργοποιούμε το ip forwarding στο VM1. Ενεργοποιώντας τη ρύθμιση IP forwarding, το σύστημα μπορεί να δρομολογεί πακέτα μεταξύ διαφορετικών δικτύων ή υποδικτύων, επιτρέποντας την πρόσβαση σε πόρους από διαφορετικά δίκτυα ή τον έλεγχο της κυκλοφορίας των πακέτων.

```
(root@kali)-[/etc/iptables]
# sudo nano /etc/sysctl.conf
```

Εικόνα 3: Παραμετροποιούμε στο sysctl.conf αρχείο

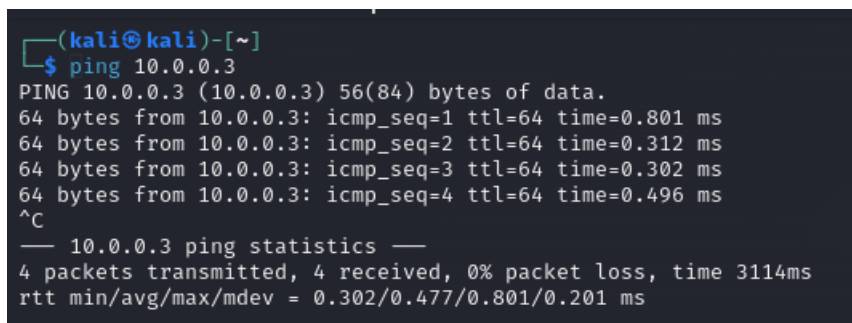


```
root@kali: /etc/iptables
File Actions Edit View Help
GNU nano 7.2 /etc/sysctl.conf
#
# /etc/sysctl.conf - Configuration file for setting system variables
# See /etc/sysctl.d/ for additional system variables.
# See sysctl.conf (5) for information.
#
#kernel.domainname = example.com
# Uncomment the following to stop low-level messages on console
#kernel.printk = 3 4 1 3
#####
# Functions previously found in netbase
#
# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1
# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
# Uncomment the next line to enable packet forwarding for IPv6
# Enabling this option disables Stateless Address Autoconfiguration
# based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1
[ Read 64 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^I Execute   ^C Location
^X Exit      ^R Read File ^N Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

Εικόνα 4: Κάνουμε uncommment την εντολή net.ipv4.ip\_forward=1

Αποθηκεύοντας την παραπάνω ρύθμιση, είμαστε βέβαιοι πως ο ρόλος του VM1 ως gateway θα είναι μόνιμος και ενεργός ακόμα και μετά από επανεκκίνηση του συστήματος.

Μεταβαίνοντας στο VM2, επιβεβαιώνουμε αρχικά πως το VM1 είναι ορατό στο VM2 και στη συνέχεια προχωράμε στην αλλαγή του default gateway ώστε να στέλνονται όλα τα πακέτα στο VM1 και εκείνο με τη σειρά του να τα κάνει forward.



```
(kali@kali)-[~]
$ ping 10.0.0.3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=0.801 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.312 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.302 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=0.496 ms
^C
— 10.0.0.3 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3114ms
rtt min/avg/max/mdev = 0.302/0.477/0.801/0.201 ms
```

Εικόνα 5: Κάνουμε ping το VM1 από το VM2



```
(kali㉿kali)-[~]  
$ ping 8.8.8.8  
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.  
^C  
— 8.8.8.8 ping statistics —  
5 packets transmitted, 0 received, 100% packet loss, time 4104ms  
  
(kali㉿kali)-[~]  
$ curl -I https://google.com/  
curl: (6) Could not resolve host: google.com
```

Εικόνα 6: Επιβεβαιώνουμε πως το VM2 δεν έχει πρόσβαση στο διαδίκτυο

Στη συνέχεια, ορίζουμε ως default gateway το VM1.

Η αλλαγή του gateway θα μπορούσε να γίνει και με την εντολή

`ip route add default via 10.0.0.3 dev eth0`

Ωστόσο, η συγκεκριμένη παραμετροποίηση αποθηκεύεται προσωρινά μέχρι την επομένη επανεκκίνηση του συστήματος, πράγμα που εμείς δε θέλουμε.

Για μόνιμη αλλαγή του gateway προχωράμε με τα παρακάτω βήματα.

```
(kali㉿kali)-[~]  
$ sudo nano /etc/network/interfaces  
[sudo] password for kali:
```

Εικόνα 7: Παραμετροποιούμε το αρχείο interfaces

```
kali@kali: ~  
File Actions Edit View Help  
GNU nano 7.2 /etc/network/interfaces *  
# This file describes the network interfaces available on your system  
# and how to activate them. For more information, see interfaces(5).  
  
source /etc/network/interfaces.d/*  
  
# The loopback network interface  
auto lo  
iface lo inet loopback  
  
auto eth0  
iface eth0 inet static  
    address 10.0.0.6  
    netmask 255.255.0.0  
    gateway 10.0.0.3
```

Εικόνα 8: Ρύθμιση στατικής διεύθυνση IP και αλλαγής gateway στην IP του VM1



Πραγματοποιούμε επανεκκίνηση στο σύστημα και στη συνέχεια επιβεβαιώνουμε πως ο default gateway άλλαξε επιτυχώς.

```
(kali㉿kali)-[~]  
$ ip route  
default via 10.0.0.3 dev eth0 onlink  
10.0.0.0/16 dev eth0 proto kernel scope link src 10.0.0.6
```

Εικόνα 9: Επιβεβαίωση αλλαγής default gateway

## 2 Παραμετροποίηση κανόνων στο iptables

Μεταβαίνουμε στο VM1 που παίζει τον ρόλο του Gateway

Αρχίζοντας με την παραμετροποίηση του iptables, προσαρμόζουμε τις ρυθμίσεις του έτσι ώστε από προεπιλογή να κάνει drop όλα τα πακέτα.

```
(root㉿kali)-[/etc/iptables]  
# iptables -P INPUT DROP  
  
(root㉿kali)-[/etc/iptables]  
# iptables -P FORWARD DROP  
  
(root㉿kali)-[/etc/iptables]  
# iptables -P OUTPUT DROP
```

Εικόνα 10: Από προεπιλογή DROP σε όλα τα chains

Επιβεβαιώνουμε πως το VM2 δεν μπορεί να στείλει icmp πακέτα στο VM1

```
(kali㉿kali)-[~]  
$ ping 10.0.0.3  
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.  
^C  
— 10.0.0.3 ping statistics —  
6 packets transmitted, 0 received, 100% packet loss, time 5105ms
```

Εικόνα 11: Επιβεβαιώνουμε στο VM2 πως τα πακέτα έγιναν drop

Το ίδιο μπορούμε να επιβεβαιώσουμε και στο VM1 από το iptables

```
(root@kali)-[/home/kali]
# iptables -n -L -v
Chain INPUT (policy DROP 11 packets, 786 bytes)
 pkts bytes target    prot opt in     out     source         destination
Chain FORWARD (policy DROP 9 packets, 644 bytes)
 pkts bytes target    prot opt in     out     source         destination
Chain OUTPUT (policy DROP 2 packets, 620 bytes)
 pkts bytes target    prot opt in     out     source         destination
```

Εικόνα 12: Επιβεβαιώνουμε στο VM1 πως τα πακέτα έγιναν drop

## 2.1 Βασική αντιμετώπιση προβλημάτων δικτύου

Ωστόσο, για να μπορούμε να αντιμετωπίσουμε τυχόν προβλήματα στο δίκτυο, επιτρέπουμε στα γειτονικά VM του δικτύου VM 1 και 2 να κάνουν ping μεταξύ τους.

Αρχίζουμε με το να επιτρέπεται στα συστήματα που βρίσκονται στο ITZ να κάνει ping το VM1

```
(root@kali)-[/home/kali]
# iptables -A INPUT -s 10.0/16 -p icmp --icmp-type echo-request -j ACCEPT

(root@kali)-[/home/kali]
# iptables -A OUTPUT -d 10.0/16 -p icmp --icmp-type echo-reply -j ACCEPT
```

Εικόνα 13: Επιτρέπουμε τα συστήματα στο ITZ δίκτυο να κάνουν ping το VM1

Εξηγώντας πιο αναλυτικά τις παραπάνω εντολές,

Με την εντολή -A INPUT, προσθέτουμε τον κανόνα στην αλυσίδα εισόδου (INPUT chain), που ρυθμίζει την ροή των εισερχομένων πακέτων.

Η παράμετρος -s 10.0/16 ελέγχει εάν η προέλευση του πακέτου είναι στο δίκτυο 10.0/16 του ITZ. Χρησιμοποιούμε το "/16" λόγω του netmask 255.255.0.0.

Με το -p icmp, ελέγχουμε αν το πρωτόκολλο είναι ICMP (ping).

Τέλος, με --icmp-type echo-request, εξετάζουμε εάν ο τύπος του ICMP πακέτου είναι echo-request. Το echo-request στέλνεται από έναν υπολογιστή που είναι ITZ προς το VM1 για να δοκιμάσει εάν είναι διαθέσιμος στο δίκτυο.

Αν όλα τα παραπάνω ισχύουν, τότε ο κανόνας αυτός αποδέχεται το ICMP πακέτο echo-request.





Ωστόσο, ο υπολογιστής του ITZ που έστειλε αυτό το πακέτο πρέπει να λάβει πίσω το ICMP πακέτο echo-reply.

Επομένως, ακολουθούμε τα ίδια βήματα αλλά για το OUTPUT chain, --icmp-type echo-reply και -d (destination) το δίκτυο του ITZ.

Τέλος, για να μπορεί το VM1 να κάνει Ping κάποιο σύστημα που βρίσκεται στο ITZ η λογική είναι παρόμοια.

```
(root@kali)-[/home/kali]
# iptables -A OUTPUT -d 10.0/16 -p icmp --icmp-type echo-request -j ACCEPT

(root@kali)-[/home/kali]
# iptables -A INPUT -s 10.0/16 -p icmp --icmp-type echo-reply -j ACCEPT
```

Εικόνα 14: Επιτρέπουμε το VM1 να κάνει ping στα συστήματα του ITZ δικτύου

```
(root@kali)-[/home/kali]
# ping 10.0.0.3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=0.242 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.287 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.267 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=0.281 ms
^C
— 10.0.0.3 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3082ms
rtt min/avg/max/mdev = 0.242/0.269/0.287/0.017 ms
```

Εικόνα 15: Επιβεβαιώνουμε στο VM2 ότι μπορεί να κάνει ping το VM1

```
(root@kali)-[/etc/iptables]
# ping 10.0.0.6
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
64 bytes from 10.0.0.6: icmp_seq=1 ttl=64 time=0.258 ms
64 bytes from 10.0.0.6: icmp_seq=2 ttl=64 time=0.574 ms
64 bytes from 10.0.0.6: icmp_seq=3 ttl=64 time=0.236 ms
64 bytes from 10.0.0.6: icmp_seq=4 ttl=64 time=0.425 ms
^C
— 10.0.0.6 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3068ms
rtt min/avg/max/mdev = 0.236/0.373/0.574/0.137 ms
```

Εικόνα 16: Επιβεβαιώνουμε στο VM1 ότι μπορεί να κάνει ping το VM2



## 2.2 Ενεργοποίηση της εταιρικής κίνησης δικτύου στο Internet

Για την ενεργοποίηση της εταιρικής κίνησης δικτύου στο Internet, αρκεί να προσθέσουμε την παρακάτω στο nat table

```
(root@kali)-[/etc/iptables]
# iptables -t nat -A POSTROUTING -s 10.0/16 -o eth0 -j MASQUERADE
```

Εικόνα 17: Προσθήκη του rule στο nat table

Αυτό σημαίνει ότι όταν τα πακέτα προέρχονται από το ITZ δίκτυο και πρόκειται να σταλούν μέσω του interface eth0, το iptables θα τροποποιήσει τη διεύθυνση IP προέλευσης των πακέτων έτσι ώστε να φαίνεται ότι προέρχονται από τον VM1 που τα προωθεί.

## 2.3 Υλοποίηση πολιτικής ασφάλειας για το εσωτερικό δίκτυο ITZ

Αρχικά, δημιουργούμε ένα καινούριο chain στο iptables, για να μπορούμε εύκολα να ομαδοποιήσουμε τα rules που αφορούν το ITZ.

```
(root@kali)-[/home/kali]
# iptables -N Itznet
```

Εικόνα 18: Δημιουργούμε το chain Itznet

```
(root@kali)-[/home/kali]
# iptables -A FORWARD -i l0 -s 10.0/16 -j ACCEPT
# iptables -A FORWARD -o l0 -d 10.0/16 -j ACCEPT
```

Εικόνα 19: Επιτρέπουμε το loopback

Στη πρώτη γραμμή, το interface εισόδου είναι το eth1 με source το ITZ δίκτυο, ενώ στη δεύτερη γραμμή, το interface εξόδου είναι το l0 με destination το ITZ δίκτυο.

```
(root@kali)-[/home/kali]
# iptables -A Itznet -s 10.0.0.0/16 -i eth1 -p udp -m udp --dport 53 -j ACCEPT
```

Εικόνα 20: Επιτρέπουμε την DNS κίνηση



Στη δεύτερη γραμμή, επιτρέπουμε την κίνηση από το ITZ που χρησιμοποιεί udp πρωτόκολο στη θύρα 53 (DNS)

```
(root@kali)-[/home/kali]
# iptables -A Itznet -s 10.0.0.0/16 -i eth1 -p tcp -m tcp --dport 443 -m state --state NEW -j ACCEPT

(root@kali)-[/home/kali]
# iptables -A Itznet -s 10.0.0.0/16 -i eth1 -p tcp -m tcp --dport 80 -m state --state NEW -j ACCEPT
```

**Εικόνα 21: Επιτρέπουμε την εξερχόμενη κίνηση προς HTTP/HTTPS**

Στους παραπάνω κανόνες, επιτρέπουμε τα εξερχόμενα πακέτα (με source το ITZ) που χρησιμοποιούν tcp πρωτόκολο στη θύρα 80 και 443 να πραγματοποιούν νέες συνδέσεις.

Στη συνέχεια προσθέτουμε τους παρακάτω κανόνες.

```
(root@kali)-[/home/kali]
# iptables -A Itznet -d 10.0.0.0/16 -i eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT

(root@kali)-[/home/kali]
# iptables -A Itznet -s 10.0.0.0/16 -i eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

**Εικόνα 22: Επιτρέπουμε τις Related και established συνδέσεις**

Οι παραπάνω κανόνες είναι αναγκαίοι για να μπορούμε να έχουμε ασφαλή και ολοκληρωμένη επικοινωνία με τις διαδικτυακές ιστοσελίδες (δηλ. port 80,443)

```
(root@kali)-[/home/kali]
# iptables -A Itznet -d 10.0.0.0/16 -i eth0 -j LOG --log-prefix "Suspicious: "
```

**Εικόνα 23: Κάνουμε logging τα υπόλοιπα πακέτα**

Οτιδήποτε δεν ικανοποιεί τους παραπάνω κανόνες, τα θεωρούμε ύποπτα και τα καταγράφουμε.

```
(root@kali)-[/home/kali]
# iptables -A Itznet -j DROP
```

**Εικόνα 24: Κάνουμε drop οποιοδήποτε άλλο πακέτο**

Τέλος, αφού ολοκληρωθεί η καταγραφή του πακέτου, γίνεται drop με τον παραπάνω κανόνα.

Μένει μόνο να ενεργοποιήσουμε την παραπάνω αλυσίδα με τις εντολές

```
iptables -A FORWARD -d 10.0.0.0/16 -i eth0 -j Itznet
iptables -A FORWARD -s 10.0.0.0/16 -i eth1 -j Itznet
```



## 2.4 Ανακατεύθυνση επισκεψιμότητας σε υπηρεσία web στο DMZ

Αρχικά, δημιουργούμε ένα καινούριο chain στο iptables, για να μπορούμε εύκολα να ομαδοποιήσουμε τα rules που αφορούν το DMZ.

```
(root@kali)-[/home/kali]
# iptables -N Dmznet
```

Εικόνα 25: Δημιουργούμε το Dmznet chain

```
(root@kali)-[/home/kali]
# iptables -A Dmznet -i eth0 -d 10.1/16 -m state --state NEW,ESTABLISHED -j ACCEPT
```

Εικόνα 26: Επιτρέπουμε τις εισερχόμενες συνδέσεις από το εξωτερικό δίκτυο

Με τον παραπάνω κανόνα, επιτρέπουμε μόνο τις new ή established εισερχόμενες συνδέσεις .

```
(root@kali)-[/home/kali]
# iptables -A DmzIn -i eth2 -s 10.1/16 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

Εικόνα 27: Επιτρέπουμε τις εξερχόμενες συνδέσεις από το DMZ δίκτυο

Με τον παραπάνω κανόνα, επιτρέπουμε μόνο τις related ή established εξερχόμενες συνδέσεις. Αυτό γιατί δε θα χρειαστεί ποτέ ο webserver να πραγματοποιήσει νέα σύνδεση με κάποιον στο εξωτερικό δίκτυο.

```
(root@kali)-[/home/kali]
# iptables -A FORWARD -j Dmznet
```

Εικόνα 28: Προωθούμε τα πακέτα στο Dmznet

Οποιαδήποτε σύνδεση δεν αφορά το ITZ θα προχωράει στο DMZ.

Ωστόσο, για να είναι προσβάσιμος ο webserver από το εξωτερικό δίκτυο, πρέπει να προσθέσουμε τον παρακάτω κανόνα στο iptables

```
(root@kali)-[/home/kali]
# iptables -A PREROUTING -d 192.168.31.41/32 -i eth0 -p tcp -m tcp --dport 80 -j DNAT --to-destination 10.1.0.3
```

Εικόνα 29: Επιτρέπουμε τη πρόσβαση του webserver από το εξωτερικό δίκτυο

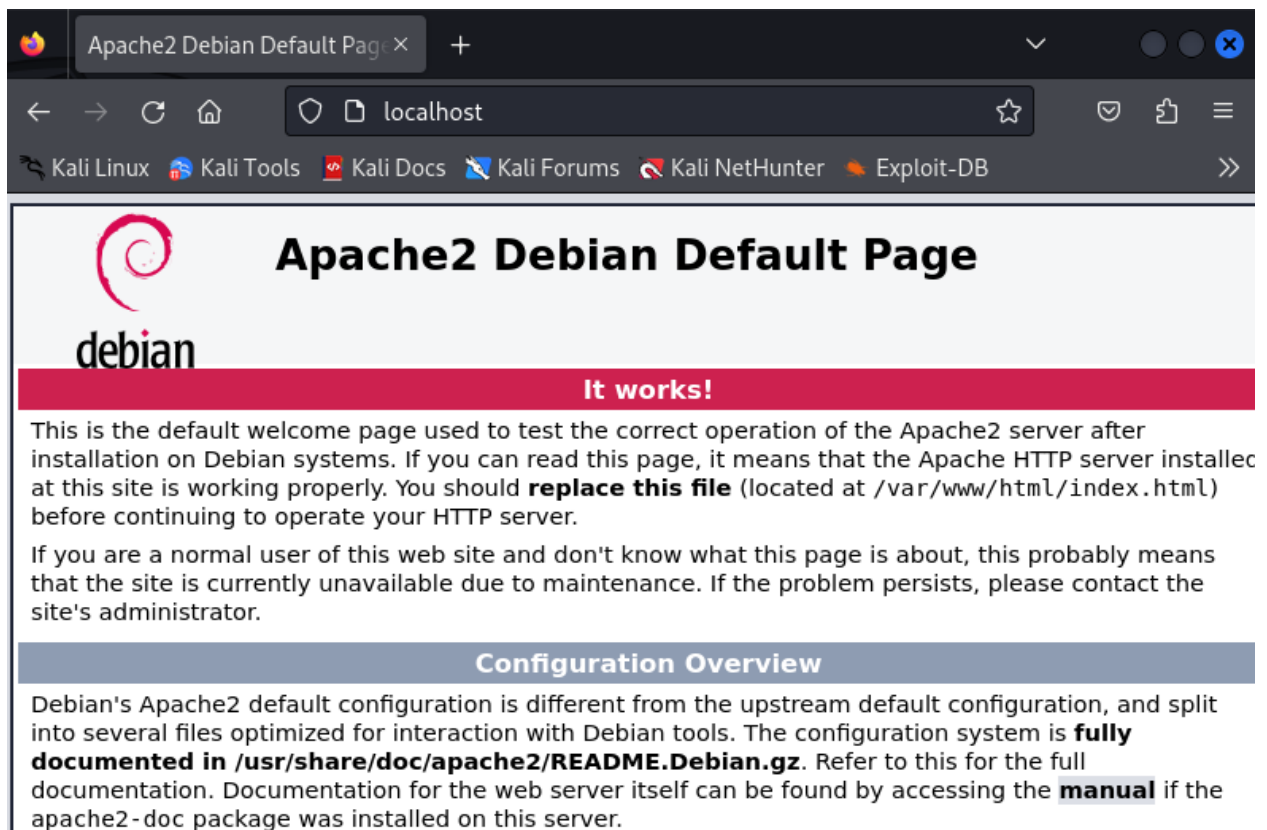


Ο παραπάνω κανόνας, ορίζει στα πακέτα που έρχονται από την ip του interface eth0 (192.168.31.41) του firewall να υποβάλλονται σε μετατροπή διεύθυνσης dnat (σε αυτή του webserver)

Για να ελέγξουμε τους παραπάνω κανόνες, σηκώνουμε έναν apache server στο kali 3

```
(root@kali)-[/home/kali]  
# service apache2 start
```

Εικόνα 30: Εκκίνηση apache server

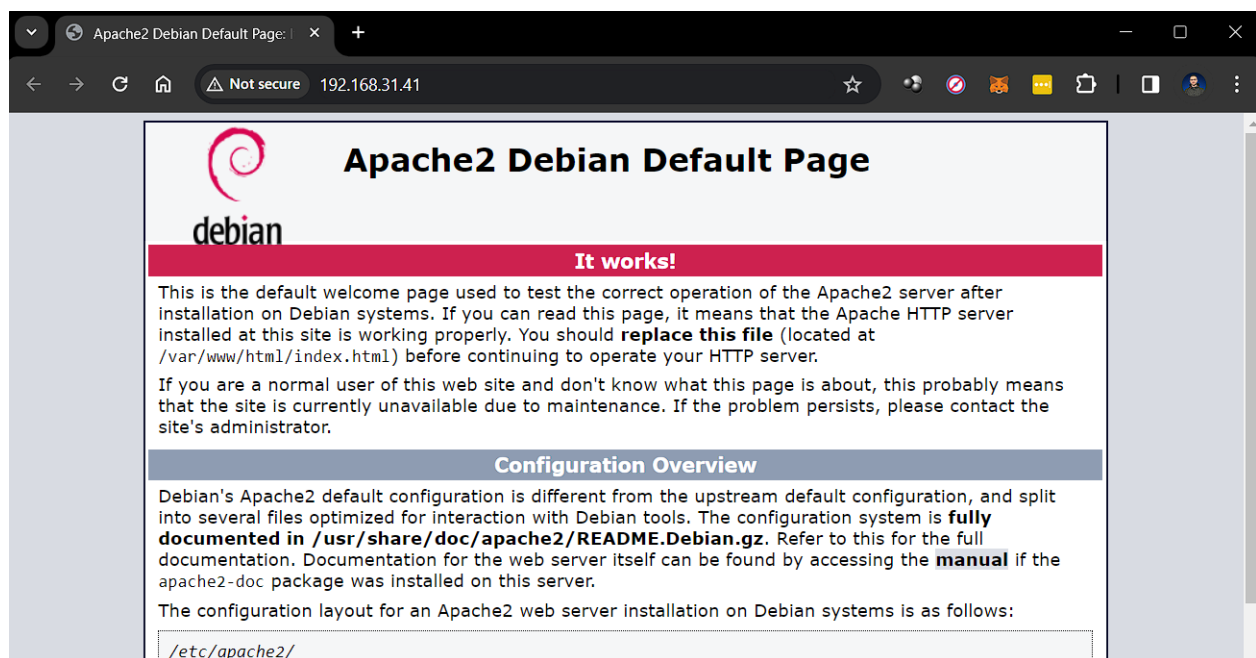


Εικόνα 31: Επιβεβαίωση πρόσβασης από το kali 3 μέσω του localhost

Μεταβαίνουμε στο Windows σύστημα που έχει τοπική διεύθυνση στο ίδιο subnet με το eth0 του firewall, και δοκιμάζουμε να μεταβούμε στον apache server, πληκτρολογώντας την ip του firewall (eth0)

```
C:\Users\Bill>curl -I http://192.168.31.41
HTTP/1.1 200 OK
Date: Mon, 29 Jan 2024 21:46:36 GMT
Server: Apache/2.4.58 (Debian)
Last-Modified: Thu, 30 Nov 2023 16:54:46 GMT
ETag: "29cd-60b61824dd4c0"
Accept-Ranges: bytes
Content-Length: 10701
Vary: Accept-Encoding
Content-Type: text/html
```

Εικόνα 32: 200 OK apache response από τα Windows



Εικόνα 33: Επιβεβαίωση πρόσβασης και από browser



## 2.5 Αυτόματη εκκίνηση κανόνων

Για την αυτόματη εκκίνηση κανόνων, θα χρησιμοποιήσουμε το Systemd. Ένα system και service manager του linux. Χρησιμοποιώντας το systemd μπορούμε να τρέξουμε ένα script μετά την εκκίνηση, το οποίο θα επαναφέρει τους κανόνες του firewall μας και θα το κάνει μόνιμο χωρίς να χρειαστεί να εγκαταστήσουμε κάποιο 3<sup>rd</sup> party app.

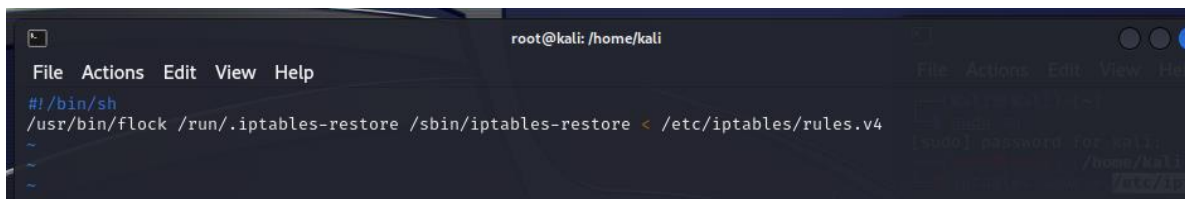
Πρώτα δημιουργούμε το script restore.sh που θα εκτελείτε κατά την έναρξη του linux

```
(root@kali)-[/home/kali]
# sudo vi /etc/iptables-persistent/restore.sh
```

Εικόνα 34: Δημιουργία restore.sh

Στη συνέχεια, πληκτρολογούμε την παρακάτω εντολή. Η εντολή αυτή χρησιμοποιεί το iptables-restore command μαζί με το full path των rules που έχουμε αποθηκεύσει με την εντολή

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



```
root@kali: /home/kali
File Actions Edit View Help
#!/bin/sh
/usr/bin/flock /run/.iptables-restore /sbin/iptables-restore < /etc/iptables/rules.v4
```

Εικόνα 35: Αποθήκευση εντολής

Θα χρειαστεί να δημιουργήσουμε ένα host αρχείο για το systemd service.

```
(root@kali)-[/home/kali]
# sudo vi /etc/systemd/system/iptables-persistent.service
```

Εικόνα 36: Δημιουργία systemd host file

Τέλος, ενεργοποιούμε το service.



```
(root@kali)-[/home/kali]
# sudo systemctl enable iptables-persistent.service
```

Εικόνα 36: Ενεργοποίηση του Service

## 2.6 Application Layer Firewall

Για τη προστασία του apache server, θα εγκαταστήσουμε το ModSecurity.

Ενεργοποιούμε τη σύνδεση του kali 3 στο διαδίκτυο και προχωράμε με την εγκατάσταση του πακέτου.

```
(root@kali)-[/home/kali]
# sudo apt install libapache2-mod-security2
```

Εικόνα 37: Εγκατάσταση του mod security

Στη συνέχεια ενεργοποιούμε το service και κάνουμε restart τον apache server.

```
(root@kali)-[/home/kali]
# sudo a2enmod security2
```

Εικόνα 38: Ενεργοποίηση του modsecurity

```
(root@kali)-[/home/kali]
# sudo systemctl restart apache2
```

Εικόνα 39: Επανεκκίνηση του apache server

Συνεχίζοντας, θα χρειαστεί να παραμετροποιήσουμε το configuration file του modsecurity ώστε να ελέγχει για πιθανές απειλές.

Μεταβαίνουμε στο αρχείο *security2.conf*



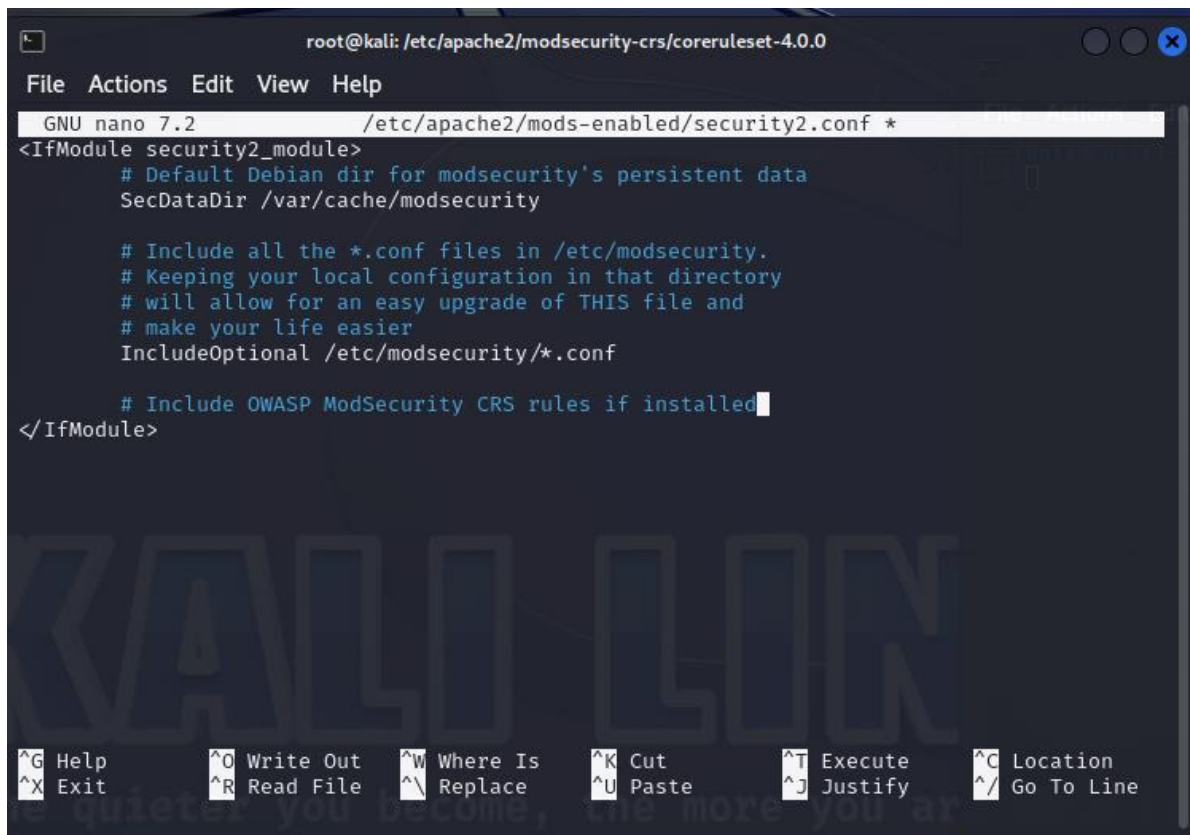
```
(root@kali)-[/home/kali]
# nano /etc/apache2/mods-enabled/security2.conf
```

Εικόνα 40: Επεξεργασία του configuration file

Στη συνέχεια, προσθέτουμε την παρακάτω εντολή,

```
IncludeOptional /etc/modsecurity/*.conf
```

Αυτό σημαίνει πως ο apache θα συμπεριλάβει όλα τα \*.conf αρχεία που βρίσκονται στο directory /etc/modsecurity



```
root@kali: /etc/apache2/modsecurity-crs/coreruleset-4.0.0
File Actions Edit View Help
GNU nano 7.2 /etc/apache2/mods-enabled/security2.conf *
<IfModule security2_module>
  # Default Debian dir for modsecurity's persistent data
  SecDataDir /var/cache/modsecurity

  # Include all the *.conf files in /etc/modsecurity.
  # Keeping your local configuration in that directory
  # will allow for an easy upgrade of THIS file and
  # make your life easier
  IncludeOptional /etc/modsecurity/*.conf

  # Include OWASP ModSecurity CRS rules if installed
</IfModule>

^K Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute   ^C Location
^X Exit      ^R Read File  ^\ Replace   ^U Paste      ^J Justify   ^_ Go To Line
```

Εικόνα 41: Συμπεριλαμβάνουμε όλα τα .conf αρχεία

Στη συνέχεια, μετανομάζουμε το αρχείο **modsecurity.conf-recommended** σε **modsecurity.conf**

```
(root@kali)-[/home/kali]
# sudo mv /etc/modsecurity/modsecurity.conf-recommended /etc/modsecurity/modsecurity.conf
```

Εικόνα 42: Μετανομάζουμε το modsecurity.conf-recommended

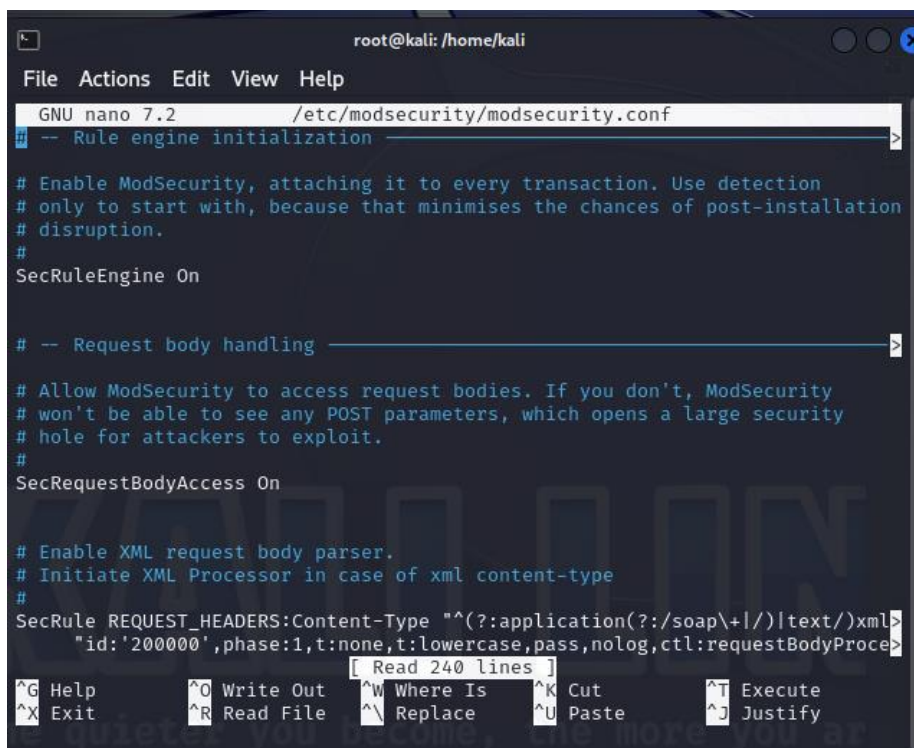
Τώρα είμαστε έτοιμοι να το επεξεργαστούμε και να προσθέσουμε τις κατάλληλες εντολές ώστε να αποτρέψουμε επιθέσεις στον apache server

```
(root@kali)-[/home/kali]
# sudo nano /etc/modsecurity/modsecurity.conf
```

Εικόνα 43: Επεξεργαζόμαστε το configuration file

Το modsecurity από προεπιλογή κάνει detect τις επιθέσεις, χωρίς να τις σταματάει.

Επομένως, βρίσκουμε την εντολή **SecRuleEngine DetectionOnly** και την αλλάζουμε σε **SecRuleEngine On**.



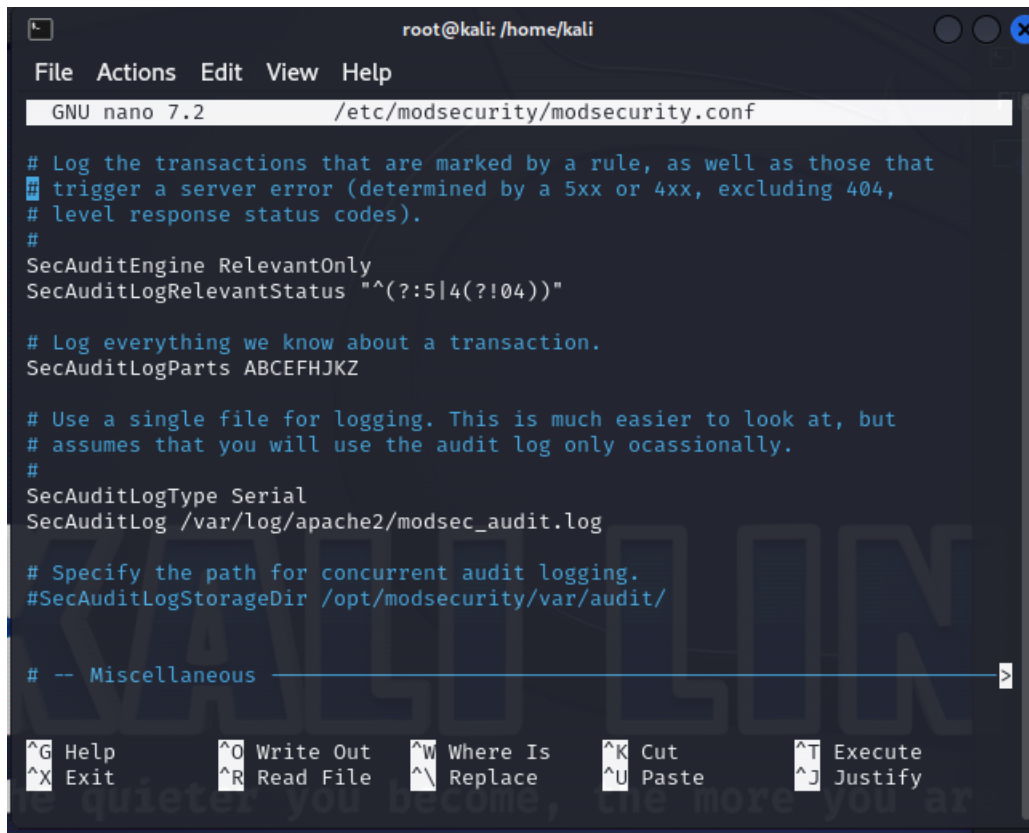
```
root@kali: /home/kali
File Actions Edit View Help
GNU nano 7.2 /etc/modsecurity/modsecurity.conf
-- Rule engine initialization --
# Enable ModSecurity, attaching it to every transaction. Use detection
# only to start with, because that minimises the chances of post-installation
# disruption.
#
SecRuleEngine On

# -- Request body handling --
# Allow ModSecurity to access request bodies. If you don't, ModSecurity
# won't be able to see any POST parameters, which opens a large security
# hole for attackers to exploit.
#
SecRequestBodyAccess On

# Enable XML request body parser.
# Initiate XML Processor in case of xml content-type
#
SecRule REQUEST_HEADERS:Content-Type "(?:application(?:/soap\+|/)|text/)xml"
"id:'200000',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProce>
[ Read 240 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify
```

Εικόνα 44: Αλλάζουμε το SecRuleEngine σε On

Στη συνέχεια αλλάζουμε την εντολή **SecAuditLogParts ABDEFHIJZ** σε **SecAuditLogParts ABCEFHIJKZ**



```
root@kali: /home/kali
File Actions Edit View Help
GNU nano 7.2 /etc/modsecurity/modsecurity.conf

# Log the transactions that are marked by a rule, as well as those that
# trigger a server error (determined by a 5xx or 4xx, excluding 404,
# level response status codes).
#
SecAuditEngine RelevantOnly
SecAuditLogRelevantStatus "^(?:5|4(?:?!04))"

# Log everything we know about a transaction.
SecAuditLogParts ABCEFHIJKZ

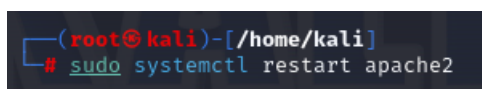
# Use a single file for logging. This is much easier to look at, but
# assumes that you will use the audit log only occasionally.
#
SecAuditLogType Serial
SecAuditLog /var/log/apache2/modsec_audit.log

# Specify the path for concurrent audit logging.
#SecAuditLogStorageDir /opt/modsecurity/var/audit/

# -- Miscellaneous -----
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify
```

Εικόνα 45: Αλλάζουμε SecAuditLogParts σε ABCEFHIJKZ

Τέλος, αποθηκεύουμε το αρχείο και στη συνέχεια κάνουμε επανεκκίνηση τον Apache



```
(root@kali)-[/home/kali]
# sudo systemctl restart apache2
```

Εικόνα 46: Κάνουμε επανεκκίνηση τον Apache server

Για την προστασία του apache server με το ModSecurity, θα εγκαταστήσουμε υπάρχοντα σύνολα κανόνων όπως το OWASP CR. Είναι δωρεάν, ευρέως χρησιμοποιούμενο και προστατεύει ενάντια σε κοινές επιθέσεις όπως SQL injection και cross-site scripting XSS. Ενσωματώνει το Project Honeypot, εντοπίζει bots και έχει ελάχιστα false positives.

Κατεβάζουμε τη τελευταία έκδοση OWASP CRS

```
(root@kali)-[/home/kali]
# wget https://github.com/coreruleset/coreruleset/archive/refs/tags/v4.0.0.tar.gz
--2024-02-28 05:52:22-- https://github.com/coreruleset/coreruleset/archive/refs/tags/v4.0.0.tar.gz
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443 ... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://codeload.github.com/coreruleset/coreruleset/tar.gz/refs/tag/v4.0.0 [following]
--2024-02-28 05:52:22-- https://codeload.github.com/coreruleset/coreruleset/tar.gz/refs/tags/v4.0.0
Resolving codeload.github.com (codeload.github.com)... 140.82.121.10
Connecting to codeload.github.com (codeload.github.com)|140.82.121.10|:443 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-gzip]
Saving to: 'v4.0.0.tar.gz'

v4.0.0.tar.gz      [  =>  ] 492.89K  566KB/s   in 0.9s

2024-02-28 05:52:24 (566 KB/s) - 'v4.0.0.tar.gz' saved [504720]
```

Εικόνα 47: Κατεβάζουμε το OWASP CRS

Στη συνέχεια κάνουμε αποσυμπίεση τα αρχεία

```
(root@kali)-[/home/kali]
# tar xvf v4.4.0.tar.gz
```

Εικόνα 48: Αποσυμπιέζουμε το αρχείο

```
(root@kali)-[/home/kali]
# sudo mkdir /etc/apache2/modsecurity-crs/
```

Εικόνα 49: Δημιουργούμε τον φάκελο modsecurity-crs

```
(root@kali)-[/home/kali]
# sudo mv coreruleset-4.0.0/ /etc/apache2/modsecurity-crs/
```

Εικόνα 50: Μεταφέρουμε τα αρχεία στον φάκελο που φτιάξαμε

```
(root@kali)-[/etc/apache2/modsecurity-crs/coreruleset-3.3.0]
# cd /etc/apache2/modsecurity-crs/coreruleset-4.0.0/
```

Εικόνα 51: Μεταβαίνουμε στον φάκελο coreruleset-4.0.0

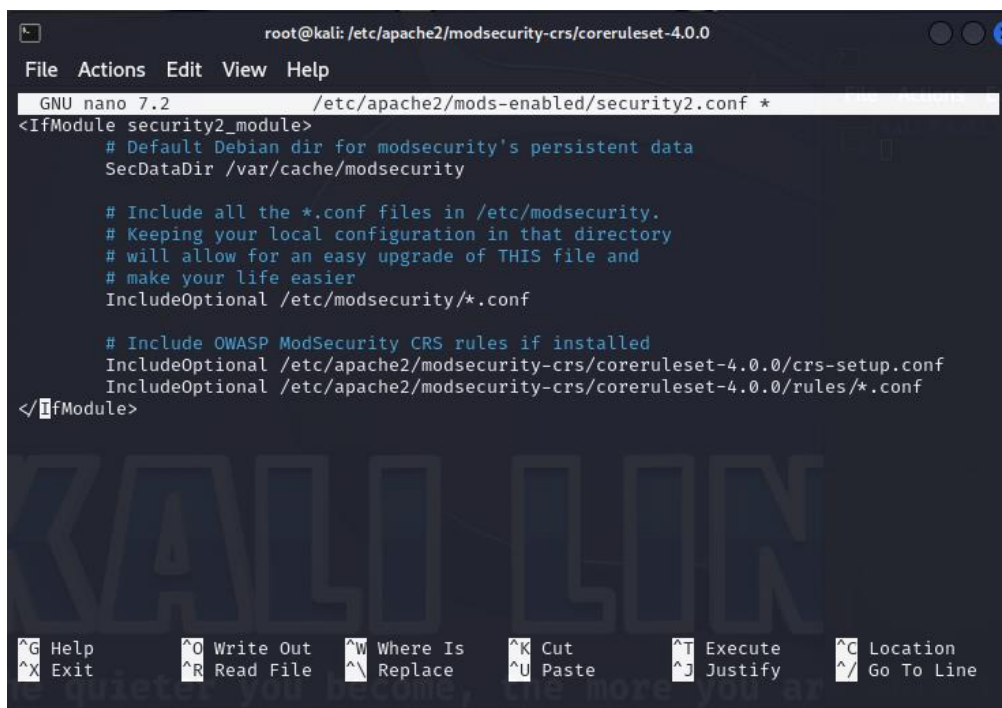


```
(root@kali)-[/etc/apache2/modsecurity-crs/coreruleset-4.0.0]
# sudo mv crs-setup.conf.example crs-setup.conf
```

Εικόνα 52: Μετανομάζουμε το αρχείο crs-setup.conf.example

```
(root@kali)-[/etc/apache2/modsecurity-crs/coreruleset-4.0.0]
# sudo nano /etc/apache2/mods-enabled/security2.conf
```

Εικόνα 53: Επεξεργαζόμαστε το security2.conf



```
root@kali: /etc/apache2/modsecurity-crs/coreruleset-4.0.0
File Actions Edit View Help
GNU nano 7.2 /etc/apache2/mods-enabled/security2.conf *
<IfModule security2_module>
# Default Debian dir for modsecurity's persistent data
SecDataDir /var/cache/modsecurity

# Include all the *.conf files in /etc/modsecurity.
# Keeping your local configuration in that directory
# will allow for an easy upgrade of THIS file and
# make your life easier
IncludeOptional /etc/modsecurity/*.conf

# Include OWASP ModSecurity CRS rules if installed
IncludeOptional /etc/apache2/modsecurity-crs/coreruleset-4.0.0/crs-setup.conf
IncludeOptional /etc/apache2/modsecurity-crs/coreruleset-4.0.0/rules/*.conf
</IfModule>

^K Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute  ^C Location
^X Exit      ^R Read File ^_ Replace   ^U Paste     ^J Justify  ^_ Go To Line
```

Εικόνα 54: Προσθέτουμε τα OWASP Rules

Στη συνέχεια αποθηκεύουμε το αρχείο και επανεκκινούμε τον apache server

```
(root@kali)-[/etc/apache2/modsecurity-crs/coreruleset-4.0.0]
# sudo systemctl restart apache2
```

Εικόνα 55: Κάνουμε επανεκκίνηση τον apache

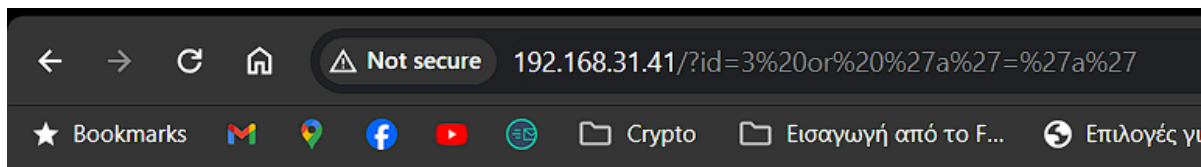
Πριν ελέγξουμε τη λειτουργία του modsecurity με τα παραπάνω rules, μπορούμε να διαβάσουμε το documentation του OWASP CRS τρέχοντας την εντολή

***sudo nano /etc/apache2/modsecurity-crs/coreruleset-3.3.0/crs-setup.conf***



Δοκιμάζουμε από το Windows Μηχάνημά μας, να μεταβούμε στον apache server εκτελώντας ένα απλό SQL Injection attack

`http://192.168.31.41/id=3 or 'a'='a'`

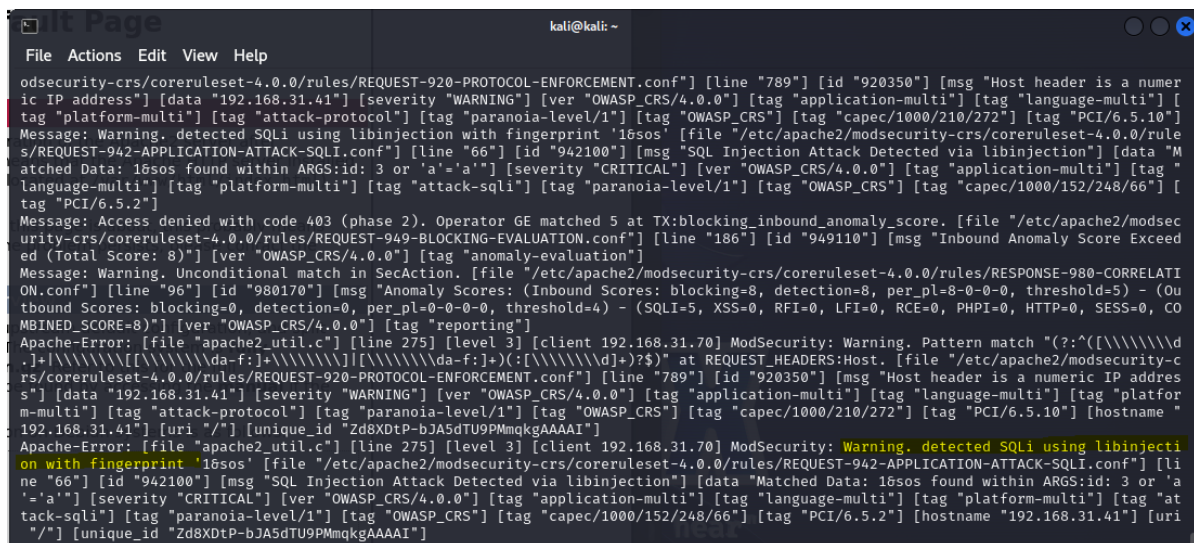


## Forbidden

You don't have permission to access this resource.

*Apache/2.4.58 (Debian) Server at 192.168.31.41 Port 80*

Εικόνα 56: Επιβεβαιώνουμε πως έγινε αποτροπή της επίθεσης



Εικόνα 57: Επιβεβαιώνουμε από τα logs την αποτροπή της επίθεσης

### 3.1 Έλεγχος Κανόνων

#### 1. Ping

Αρχικά, επιβεβαιώνουμε πως μόνο τα τερματικά στο 10.0/16 subnet μπορούν να κάνουν ping το VM1

Μεταβαίνουμε στο kali 2 και επιχειρούμε να κάνουμε ping το kali 1 (firewall)

```
(kali㉿kali)-[~]  
$ ping 10.0.0.3  
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.  
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=0.279 ms  
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.644 ms  
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.306 ms  
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=0.338 ms  
64 bytes from 10.0.0.3: icmp_seq=5 ttl=64 time=0.358 ms  
^C  
— 10.0.0.3 ping statistics —  
5 packets transmitted, 5 received, 0% packet loss, time 4075ms  
rtt min/avg/max/mdev = 0.279/0.385/0.644/0.132 ms
```

Εικόνα 58: Επιβεβαιώνουμε πως το vm2 μπορεί να κάνει ping το vm1

Στη συνέχεια, αρκεί να επιβεβαιώσουμε ότι κανένα άλλο icmp εισερχόμενο πακέτο δε θα γίνεται δεκτό.

Δοκιμάζουμε από το kali 3 να κάνουμε ping το kali 1 (firewall)

```
(kali㉿kali)-[~] the quieter you become, the more you are able to hear  
$ ping 10.1.0.2  
PING 10.1.0.2 (10.1.0.2) 56(84) bytes of data.  
^C  
— 10.1.0.2 ping statistics —  
6 packets transmitted, 0 received, 100% packet loss, time 5111ms
```

Εικόνα 59: Επιβεβαιώνουμε πως το vm3 δε μπορεί να κάνει ping το vm1

Επίσης, επιβεβαιώνουμε και στο δικό μας σύστημα Windows σύστημα πως δε μπορούμε να κάνουμε ping το kali 1 (firewall)



```
C:\Users\Bill>ping 192.168.31.41

Pinging 192.168.31.41 with 32 bytes of data:
Request timed out.

Ping statistics for 192.168.31.41:
    Packets: Sent = 1, Received = 0, Lost = 1 (100% loss),
Control-C
^C
```

Εικόνα 60: Επιβεβαιώνουμε πως το δικό μας σύστημα δε μπορεί να κάνει ping το vm1

Τώρα, βάσει τους εξερχόμενους κανόνες θα πρέπει να γίνεται accept τα outbound icmp πακέτα που έχουν destination το 10.0/16 δικτυο.

Αρα επιχειρούμε στο kali 1 (firewall) να κάνουμε ping το kali 2

```
(kali㉿kali)-[~]
$ ping 10.0.0.6
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
64 bytes from 10.0.0.6: icmp_seq=1 ttl=64 time=0.266 ms
64 bytes from 10.0.0.6: icmp_seq=2 ttl=64 time=0.249 ms
64 bytes from 10.0.0.6: icmp_seq=3 ttl=64 time=0.408 ms
64 bytes from 10.0.0.6: icmp_seq=4 ttl=64 time=0.245 ms
64 bytes from 10.0.0.6: icmp_seq=5 ttl=64 time=0.484 ms
^C
— 10.0.0.6 ping statistics —
5 packets transmitted, 5 received, 0% packet loss, time 4087ms
rtt min/avg/max/mdev = 0.245/0.330/0.484/0.097 ms
```

Εικόνα 61: Επιβεβαιώνουμε στο vm1 πως μπορεί να γίνει ping το vm2



## 2. Loopback

Επιβεβαιώνουμε ότι επιτρέπεται το loopback στο vm2

```
(kali㉿kali)-[~]
$ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.026 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.058 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.061 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.054 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.053 ms
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.062 ms
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.053 ms
64 bytes from 127.0.0.1: icmp_seq=8 ttl=64 time=0.054 ms
64 bytes from 127.0.0.1: icmp_seq=9 ttl=64 time=0.056 ms
64 bytes from 127.0.0.1: icmp_seq=10 ttl=64 time=0.069 ms
64 bytes from 127.0.0.1: icmp_seq=11 ttl=64 time=0.058 ms
64 bytes from 127.0.0.1: icmp_seq=12 ttl=64 time=0.045 ms
64 bytes from 127.0.0.1: icmp_seq=13 ttl=64 time=0.061 ms
^C
— 127.0.0.1 ping statistics —
13 packets transmitted, 13 received, 0% packet loss, time 12271ms
rtt min/avg/max/mdev = 0.026/0.054/0.069/0.009 ms
```

Εικόνα 62: Επιβεβαιώνουμε στο vm2 πως επιτρέπεται το loopback

## 3. Κίνηση DNS

Επιβεβαιώνουμε ότι επιτρέπεται η κίνηση DNS στο vm2

```
(kali㉿kali)-[~]
$ nslookup unipi.gr
Server:      10.0.0.3
Address:     10.0.0.3#53
Non-authoritative answer:
Name:   unipi.gr
Address: 195.251.229.4
```

Εικόνα 63: Επιβεβαιώνουμε στο vm2 πως επιτρέπεται η κίνηση DNS

## 4. HTTP/HTTPS

Επιβεβαιώνουμε στο vm2 πως επιτρέπεται η εξερχόμενη κίνηση http/https.

```
(kali㉿kali)-[~]  
$ curl -I http://unipi.gr  
HTTP/1.1 200 OK  
Date: Sat, 03 Feb 2024 12:36:33 GMT  
Server: Apache  
Accept-Ranges: bytes  
Content-Length: 126  
Connection: close  
Content-Type: text/html; charset=UTF-8
```

Εικόνα 64: Επιβεβαιώνουμε στο vm2 πως επιτρέπεται η εξερχόμενη http κίνηση

```
(kali㉿kali)-[~]  
$ curl -I https://www.google.gr  
HTTP/2 200  
content-type: text/html; charset=ISO-8859-7  
content-security-policy-report-only: object-src 'none';base-uri 'self';script  
-src 'nonce-MP582fKQUcSmKgsPZ9LLRw' 'strict-dynamic' 'report-sample' 'unsafe-  
eval' 'unsafe-inline' https: http:;report-uri https://csp.withgoogle.com/csp/  
gws/other-hp  
p3p: CP="This is not a P3P policy! See g.co/p3phelp for more info."  
date: Sat, 03 Feb 2024 13:44:44 GMT  
server: gws  
x-xss-protection: 0  
x-frame-options: SAMEORIGIN  
expires: Sat, 03 Feb 2024 13:44:44 GMT  
cache-control: private  
set-cookie: AEC=Ae3NU9NiwR8Sw0jYlB320xM7jnEQKBG7Gqi05QFEItRK0u70t8pB00tb9g;  
expires=Thu, 01-Aug-2024 13:44:44 GMT; path=/; domain=.google.gr; Secure; Htt  
pOnly; SameSite=lax  
set-cookie: __Secure-ENID=17.SE=Lja-Xv2oi_aVTMhKblzAkq6ygjKa2PJPUHXR38XTwB4Vg  
00a68ZwGo53C0LLHZYbeonW8aaCKz04Do9AJNginRFNChbcVpqUUUEMVx4VilJeJmsuWYqBE0rd5G  
9000SKhgfgzQuP8JQmugR92CbIUL5S4ewlgiptATJv4sQ2R6mVIEs; expires=Wed, 05-Mar-20  
25 06:03:02 GMT; path=/; domain=.google.gr; Secure; HttpOnly; SameSite=lax  
set-cookie: CONSENT=PENDING+636; expires=Mon, 02-Feb-2026 13:44:44 GMT; path=
```

Εικόνα 65: Επιβεβαιώνουμε στο vm2 πως επιτρέπεται η εξερχόμενη https κίνηση



## 4 Δοκιμή επιθέσεων και υλοποίηση κανόνων αποτροπής

### 4.1 Υλοποίηση επίθεσης

Υλοποιούμε scanning επίθεση στο vm3 (public host) από τερματικό που βρίσκεται στο εξωτερικό δίκτυο.

Χρησιμοποιώντας το hping3 με SYN (-S) flag στη θύρα 80, βλέπουμε ότι είναι ορατή μιας και δέχεται τα 2 πακέτα που του στείλαμε.

```
(root@kali)-[/home/kali]
# sudo hping3 -S 192.168.31.41 -p 80 -c 2
HPING 192.168.31.41 (eth0 192.168.31.41): S set, 40 headers + 0 data bytes
len=46 ip=192.168.31.41 ttl=63 DF id=0 sport=80 flags=SA seq=0 win=64240 rtt=
3.8 ms
len=46 ip=192.168.31.41 ttl=63 DF id=0 sport=80 flags=SA seq=1 win=64240 rtt=
3.2 ms
— 192.168.31.41 hping statistic —
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 3.2/3.5/3.8 ms
```

### 4.2 Αποτροπή επίθεσης

Για να αποτρέψουμε την παραπάνω επίθεση (συμπεριλαμβανομένου και την DDos) προχωράμε με την εξής στρατηγική.

1. Αποκλεισμός Μη Έγκυρων Πακέτων: Θα πρέπει να υπάρχει κανόνας που θα φιλτράρει όλα τα πακέτα που δεν είναι τύπου SYN και δεν ανήκουν σε μια ήδη established σύνδεση TCP
2. Αποκλεισμός Νέων Πακέτων που δεν είναι SYN: Αυτός ο κανόνας φιλτράρει όλα τα πακέτα που είναι νέα (δεν ανήκουν σε μια established σύνδεση) και δεν χρησιμοποιούν το flag SYN. Είναι παρόμοιος με τον κανόνα "Αποκλεισμός Μη Έγκυρων Πακέτων", αλλά απορρίπτει κάποια πακέτα που ο άλλος δεν ανιχνεύει.
3. Αποκλεισμός Πακέτων με uncommon MSS: Αυτός ο κανόνας φιλτράρει νέα πακέτα (μόνο τύπου SYN μπορεί να είναι νέα σύμφωνα με τους προηγούμενους κανόνες) που χρησιμοποιούν μια τιμή MSS TCP που δεν είναι συνηθισμένη. Αυτό βοηθά στον αποκλεισμό απλών επιθέσεων SYN.



4. Αποκλεισμός Πακέτων με Λανθασμένα flag TCP: Αυτή η σειρά κανόνων θα αποκλείει πακέτα που χρησιμοποιούν λανθασμένα flags TCP, δηλαδή flags που ένα φυσιολογικό πακέτο δε θα χρησιμοποιούσε. Αυτό μπορεί να περιλαμβάνει πακέτα με συνδυασμούς flags που δεν είναι επιτρεπτοί σε ένα κανονικό TCP πακέτο.

```
(root@kali)-[/home/kali/Desktop/snortlogs]
# iptables -t mangle -A PREROUTING -m conntrack --ctstate INVALID -j DROP

(root@kali)-[/home/kali/Desktop/snortlogs]
# iptables -t mangle -A PREROUTING -p tcp ! --syn -m conntrack --ctstate NEW -j DROP

(root@kali)-[/home/kali/Desktop/snortlogs]
# iptables -t mangle -A PREROUTING -p tcp -m conntrack --ctstate NEW -m tcpmss ! --mss 536:65535 -j DROP

(root@kali)-[/home/kali/Desktop/snortlogs]
# iptables -t mangle -A PREROUTING -p tcp --tcp-flags FIN,SYN FIN,SYN -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags FIN,RST FIN,RST -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags FIN,ACK FIN -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ACK,URG URG -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ACK,PSH PSH -j DROP
iptables -t mangle -A PREROUTING -p tcp --tcp-flags ALL NONE -j DROP
```

Τέλος επιβεβαιώνουμε το επιτυχές mitigation των παραπάνω επιθέσεων

```
(root@kali)-[/home/kali]
# sudo hping3 -S 192.168.31.41 -p 80 -c 2
HPING 192.168.31.41 (eth0 192.168.31.41): S set, 40 headers + 0 data bytes

— 192.168.31.41 hping statistic —
2 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Πρώτα υλοποιούμε εκ νέου ένα scanning attack που μόνο τώρα έγιναν όλα τα πακέτα loss

```
(root@kali)-[/home/kali]
# sudo hping3 -c 20000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.168.31.41
HPING 192.168.31.41 (eth0 192.168.31.41): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
^C
— 192.168.31.41 hping statistic —
345755 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Στη συνέχεια πραγματοποιούμε ένα DDos attack όπου το -c flag χρησιμοποιείται για την καταμέτρηση των πακέτων, -d flag για το μέγεθος των δεδομένων, -w για το word size, -p τη θύρα και --rand-source flag ώστε να είναι εντελώς τυχαία η ip του source.

Επομένως, από το παραπάνω screenshot επιβεβαιώνουμε και την επιτυχή αποτροπή του ddos attack.

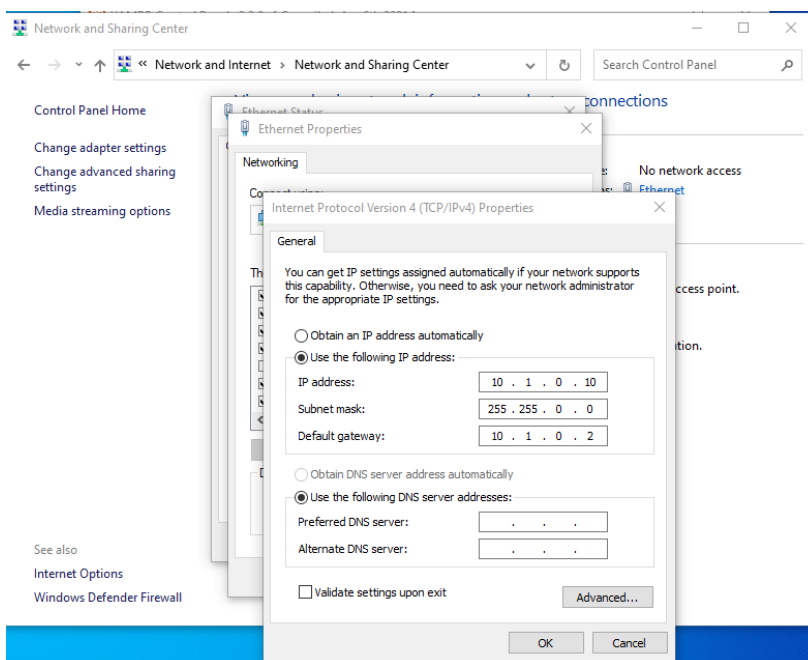
Τέλος, βλέπουμε ότι μια τυπική σύνδεση http φορτώνει κανονικά

```
(root@kali)-[/home/kali]
# curl -I http://192.168.31.41/
HTTP/1.1 200 OK
Date: Thu, 29 Feb 2024 21:54:08 GMT
Server: Apache/2.4.58 (Debian)
Last-Modified: Thu, 30 Nov 2023 16:54:46 GMT
ETag: "29cd-60b61824dd4c0"
Accept-Ranges: bytes
Content-Length: 10701
Vary: Accept-Encoding
Content-Type: text/html
```

## 5 Ανίχνευση επιθέσεων με τη χρήση IDS

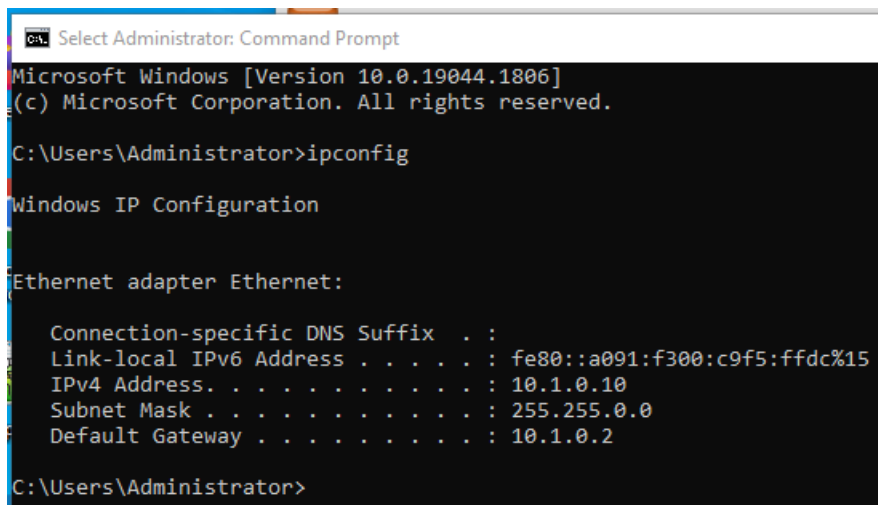
Αρχικά, για να διατηρήσουμε την ίδια τοπολογία χωρίς να αυξήσουμε ιδιαίτερα τα resources, προσθέτουμε ένα windows virtual machine στο public host.

Ξεκινάμε με τη βασική παραμετροποίηση του windows συστήματος, δίνοντάς του μία στατική ip και έχοντας ως gateway το vm1.



Εικόνα 66: Ορίζουμε ως gateway το vm1

Αποθηκεύουμε και επιβεβαιώνουμε από το terminal



```
Microsoft Windows [Version 10.0.19044.1806]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Administrator>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

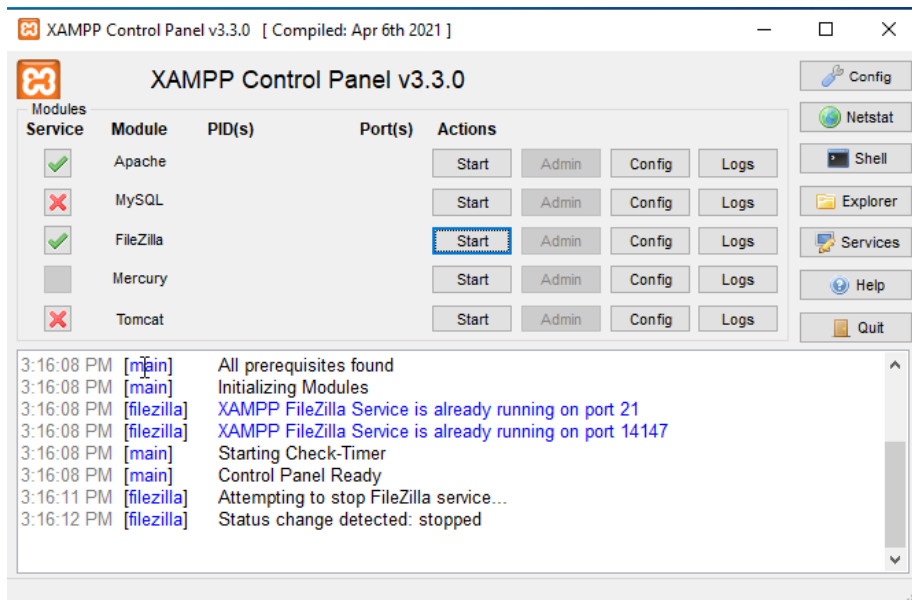
    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::a091:f300:c9f5:ffdc%15
    IPv4 Address. . . . . : 10.1.0.10
    Subnet Mask . . . . . : 255.255.0.0
    Default Gateway . . . . . : 10.1.0.2

C:\Users\Administrator>
```

**Εικόνα 67:** Επιβεβαιώνουμε τις παραπάνω ρυθμίσεις

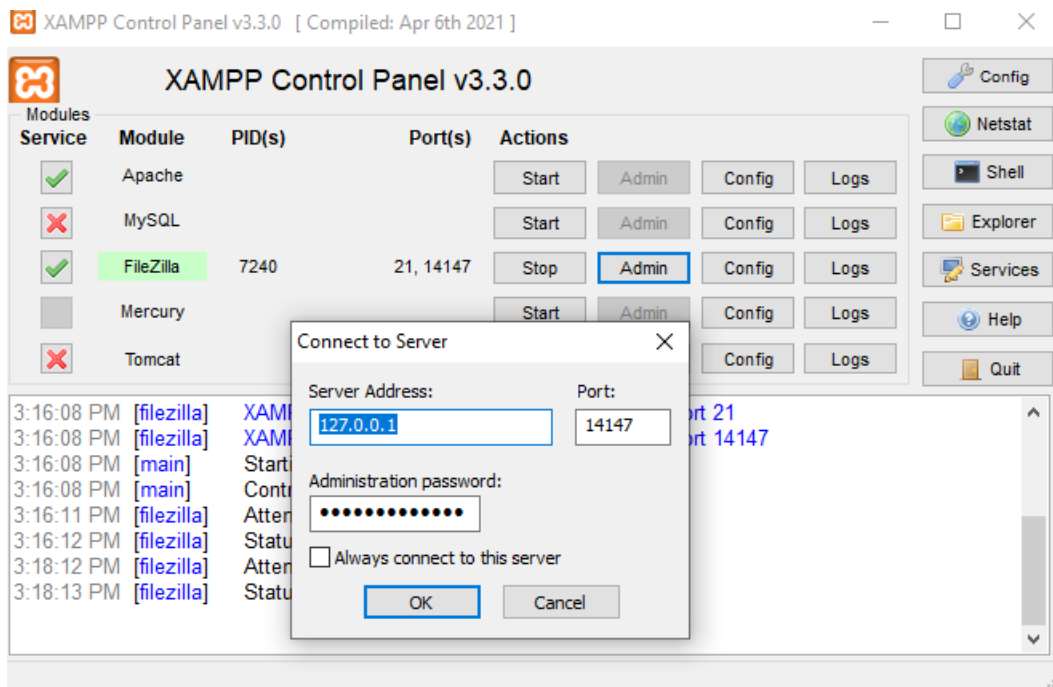
Στη συνέχεια εγκαθιστούμε στο Windows VM το xampp και προχωράμε με την παραμετροποίηση του ftp server.

Πρώτα εκκινούμε τον ftp server πατώντας Start στο FileZilla και στη συνέχεια επιλέγουμε την επιλογή Admin.



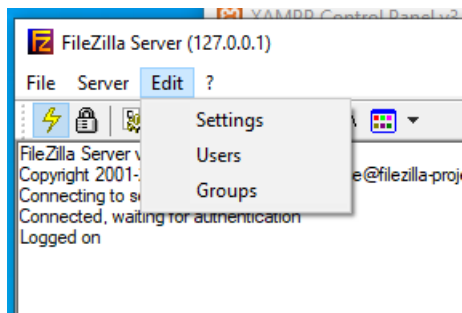
Εικόνα 68: Εκκινούμε τον ftp server

Στη συνέχεια επιλέγουμε ένα password για τον admin user και στη συνέχεια ακολουθούμε τη διαδικασία για τη δημιουργία νέου χρήστη.



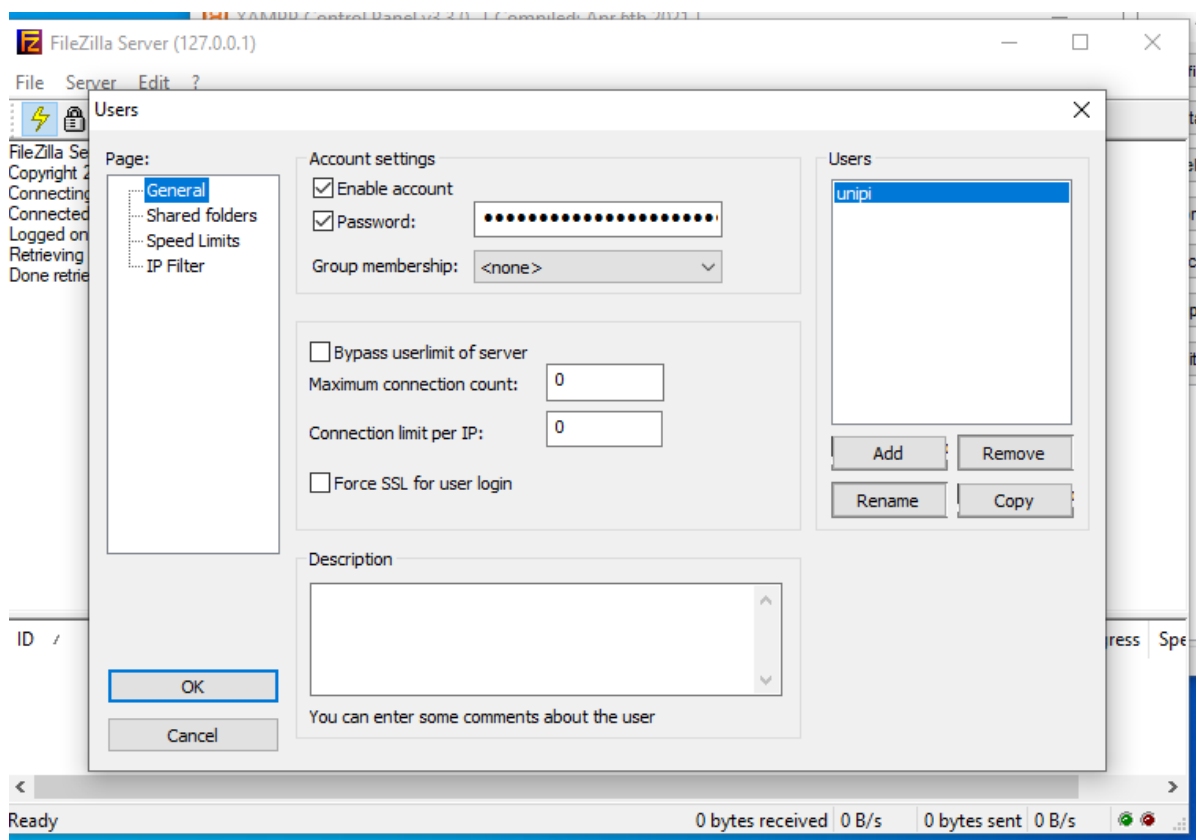
Εικόνα 68: Επιλέγουμε admin password και συνδεόμαστε στον local ftp server

Αφού συνδεθούμε, πηγαίνουμε την επιλογή users για να δημιουργήσουμε νέο χρήστη



**Εικόνα 69:** Μεταβαίνουμε στην επιλογή Users για τη δημιουργία νέου χρήστη

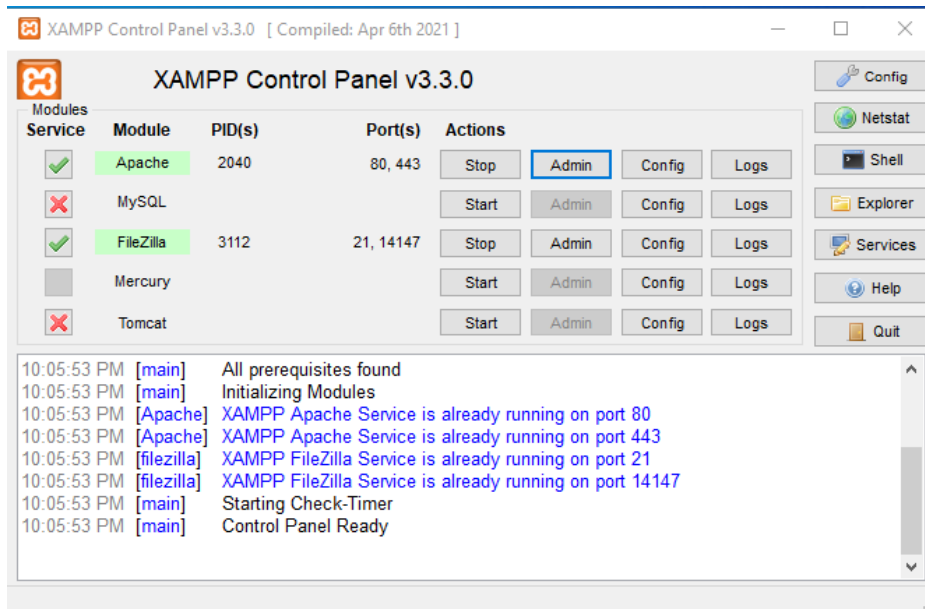
Στη συνέχεια, δημιουργούμε τον χρήστη 'unipi' ορίζοντας κωδικό και shared folder.



**Εικόνα 70:** Δημιουργούμε τον χρήστη unipi



Στη συνέχεια, επιβεβαιώνουμε πως ο Apache και ο ftp server τρέχουν κανονικά.



**Εικόνα 71:** Βεβαιωνόμαστε ότι ο Apache Server και ο Filezilla τρέχουν

Πριν υλοποιήσουμε τις επιθέσεις, πρέπει να κάνουμε προσβάσιμο τον windows server από το εξωτερικό δίκτυο, μέσω του iptables.

Αρκεί να προσθέσουμε τα παρακάτω rules στο nat table.

```
iptables -t nat -A PREROUTING -d 192.168.31.41/32 -i eth0 -p tcp -m tcp --dport 443 -j DNAT --to-destination 10.1.0.10
```

```
iptables -t nat -A PREROUTING -d 192.168.31.41/32 -i eth0 -p tcp -m tcp --dport 21 -j DNAT --to-destination 10.1.0.10
```

```
(root@kali)-[/home/kali/Desktop/snortlogs]
# iptables -t nat -A PREROUTING -d 192.168.31.41/32 -i eth0 -p tcp -m tcp --dport 21 -j DNAT --to-destination 10.1.0.10
```

**Εικόνα 72:** Επιτρέπουμε τη πρόσβαση στον ftp server

```
(root@kali)-[/home/kali/Desktop/snortlogs]
# iptables -t nat -A PREROUTING -d 192.168.31.41/32 -i eth0 -p tcp -m tcp --dport 443 -j DNAT --to-destination 10.1.0.10
```

**Εικόνα 73:** Επιτρέπουμε τη πρόσβαση στον apache server

Επίσης, δημιουργούμε ένα local.rules αρχείο, που θα περιέχει τα rules του snort.

Πιο συγκεκριμένα,



```
block tcp any any -> any 21 (msg:"FTP Brute Force Attack"; flow:to_server,established;  
content:"PASS "; nocase; threshold: type threshold, track by_src, count 2, seconds 60;  
sid:1000001; rev:1;)
```

```
reject tcp any any -> any 443 (flags: S; msg:"Possible DoS Attack Blocked : SYNflood";  
flow:stateless; sid:1000006; detection_filter:track by_dst, count 20, seconds 10;)
```

Για το ftp brute force και το http flood αντίστοιχα.

Προχωράμε με την αποθήκευση αυτού του local.rules αρχείου

### 5.1 FTP BruteForce attack

Εξηγώντας τον παραπάνω κανόνα,

Ενέργεια: **block** - Αυτό υποδηλώνει ότι αν οι συνθήκες που καθορίζονται στον κανόνα πληρούνται, το IDS θα αποκλείσει την κίνηση.

Πρωτόκολλο: **tcp** - Αυτός ο κανόνας θα ελέγχει την TCP κίνηση.

Πηγή και Προορισμός: **any any -> any 21** - Καθορίζει ότι ο κανόνας ισχύει για κίνηση από οποιαδήποτε πηγή προς οποιονδήποτε προορισμό στη θύρα 21, η οποία είναι η προεπιλεγμένη θύρα για το FTP (File Transfer Protocol).

Μήνυμα: **msg:"FTP Brute Force Attack"** -Αυτή η φράση θα εμφανίζεται στο console του snort όταν ικανοποιείται ο κανόνας.

**flow:to\_server,established** - Αυτό τμήμα του κανόνα εξασφαλίζει ότι η ροή κατευθύνεται προς τον ftp server και είναι ήδη established . Αυτό είναι σημαντικό για να αποφευχθούν τα false positives που πραγματοποιούν μόνο νέα σύνδεση.

**content:"PASS "; nocase** - Αυτό ελέγχει την παρουσία της φράσης "PASS ". Αυτό είναι σημαντικό επειδή το "PASS" είναι η εντολή που χρησιμοποιείται στο FTP για την αυθεντικοποίηση με κωδικό πρόσβασης.

**threshold: type threshold, track by\_src, count 2, seconds 60** - Αυτό το τμήμα καθορίζει ένα κατώτατο όριο για την ενεργοποίηση του κανόνα. Καθορίζει ότι εάν δύο πακέτα που



ταιριάζουν με αυτόν τον κανόνα εντοπίζονται από την ίδια πηγή εντός ενός χρονικού πλαισίου 60 δευτερολέπτων, ο κανόνας θα κάνει την ενέργεια που του έχουμε ορίσει.

### Σημείωση

Το 2 μπήκε δοκιμαστικά για την εξοικονόμηση χρόνου και την αποφυγή wordlist

SID: **sid:1000001** – Είναι το του κανόνα (πρέπει να είναι μοναδικό)

**rev:1** – Είναι τα revisions που έγιναν στον κανόνα (βοηθάει στη περίπτωση που χρειάζεται μικρά fine tunings)

Συνεχίζοντας με την επίθεση,

Ενεργοποιούμε το snort σε κατάσταση IDS mode

```
(root@kali) - [ /home/kali/Desktop/snortlogs ]  
# sudo snort -i eth0:eth2 -q -A console -c /etc/snort/rules/local.rules -Q --daq afpacket -l /home/kali/Desktop/snortlogs
```

### Εικόνα 74: Ενεργοποίηση του snort σε IDS Mode

Στη συνέχεια πηγαίνουμε σε ένα kali που ανήκει στο εξωτερικό δίκτυο (bridged) και πραγματοποιούμε την brute force επίθεση.

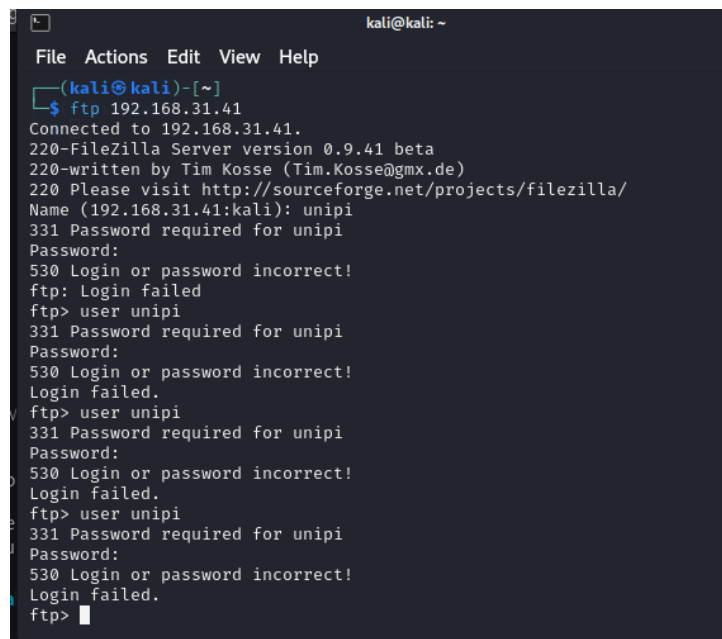
Εδώ επειδή το count στον κανόνα είναι μικρό, θα πραγματοποιήσουμε την επίθεση χωρίς wordlist.

Δοκιμάζοντας σε γρήγορο χρονικό διάστημα τυχαίους κωδικούς το snort βλέπουμε πως έχει αναγνωρίσει την επίθεση και έχει κόψει την κίνηση.

```
(root@kali) - [ /home/kali ]  
# sudo snort -i eth0:eth2 -q -A console -c /etc/snort/rules/local.rules -Q --daq afpacket -l /home/kali/Desktop/snortlogs  
02/29-15:18:44.970784 [**] [1:1000001:0] Incoming FTP connection [**] [Priority: 0] {TCP} 192.168.31.87:37294 → 10.1.0.10:21  
02/29-15:18:44.970743 [**] [1:1000001:0] Incoming FTP connection [**] [Priority: 0] {TCP} 192.168.31.87:37294 → 192.168.31.41:21  
02/29-15:18:49.361922 [Drop] [**] [1:1000003:1] FTP Brute Force Blocked [**] [Priority: 0] {TCP} 192.168.31.87:37294 → 192.168.31.41:21  
02/29-15:18:55.688072 [Drop] [**] [1:1000003:1] FTP Brute Force Blocked [**] [Priority: 0] {TCP} 192.168.31.87:37294 → 192.168.31.41:21  
02/29-15:19:01.755056 [Drop] [**] [1:1000003:1] FTP Brute Force Blocked [**] [Priority: 0] {TCP} 192.168.31.87:37294 → 192.168.31.41:21  
02/29-15:19:03.896505 [**] [1:10000004:0] ICMP Packet found [**] [Priority: 0] {IPV6-ICMP} fe80::a00:27ff:fe59:4304 → ff02::2  
02/29-15:19:09.397801 [Drop] [**] [1:1000003:1] FTP Brute Force Blocked [**] [Priority: 0] {TCP} 192.168.31.87:37294 → 192.168.31.41:21  
^C** Caught Int-Signal
```

### Εικόνα 75: Snort output for ftp brute force

Ενώ επίσης, στο kali που τρέξουμε την επίθεση,



```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ ftp 192.168.31.41  
Connected to 192.168.31.41.  
220-FileZilla Server version 0.9.41 beta  
220-written by Tim Kosse (Tim.Kosse@gmx.de)  
220 Please visit http://sourceforge.net/projects/filezilla/  
Name (192.168.31.41:kali): unipi  
331 Password required for unipi  
Password:  
530 Login or password incorrect!  
ftp: Login failed  
ftp> user unipi  
331 Password required for unipi  
Password:  
530 Login or password incorrect!  
Login failed.  
ftp> user unipi  
331 Password required for unipi  
Password:  
530 Login or password incorrect!  
Login failed.  
ftp> user unipi  
331 Password required for unipi  
Password:  
530 Login or password incorrect!  
Login failed.  
ftp>
```

Εικόνα 76: Kali ftp brute force screen

## 5.2 HTTP FLOOD ATTACK

reject tcp any any -> any 443 (flags: S; msg:"Possible DoS Attack Blocked : SYNflood";  
flow:stateless; sid:1000006; detection\_filter:track by\_dst, count 20, seconds 10;)

Αυτός ο κανόνας απορρίπτει οποιαδήποτε προσπάθεια σύνδεσης TCP από οποιαδήποτε πηγή προς οποιοδήποτε προορισμό στη θύρα 443. Συγκεκριμένα, απορρίπτονται μόνο τα πακέτα που έχουν το SYN flag (Αίτηση Έναρξης Σύνδεσης) στη θύρα 443. Αυτό θα μπορούσε να υποδείξει μια επίθεση DoS. Επιπλέον, χρησιμοποιείται ένα φίλτρο ανίχνευσης (detection filter) που παρακολουθεί τον αριθμό των αιτήσεων που φτάνουν στον ίδιο προορισμό εντός 10 δευτερολέπτων. Αν καταγραφούν περισσότερες από 20 αιτήσεις, τότε η επίθεση αποκόπτεται.

Για αυτό το λόγο, δημιουργήσαμε ένα python script που θα στέλνει μαζικά αιτήματα έναρξης σύνδεσης στον apache server.

```
(kali@kali)-[~]
$ nano httpflood.py

File Actions Edit View Help
GNU nano 7.2 httpflood.py
import requests
import time

cnt=0
for i in range(0,100):
    r=requests.get("https://192.168.31.41/",verify=False)
    time.sleep(0.25)
    print(r.status_code)

action has timed out
192.168.31.41 is taking too long to respond.
be temporarily unavailable or too busy. Try again in a few moments.
able to load any pages, check your computer's network connection.
ter or network is protected by a firewall. Be that Firefox is permitted to access the

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify
```

Εικόνα 77: Python DoS script

Με τον ίδιο τρόπο ενεργοποιούμε το snort και τρέχουμε το python script.

```
(root@kali)-[/home/kali/Desktop/snortlogs]
# sudo snort -i eth0:eth2 -q -A console -c /etc/snort/rules/local.rules -Q --daq afpacket -l /home/kali/Desktop/snortlogs
```

Εικόνα 78: Ενεργοποίηση του snort σε IDS Mode

Παρακάτω επιβεβαιώνουμε και την παρέμβαση του snort

```
(root@kali)-[/home/kali/Desktop/snortlogs]
# sudo snort -i eth0:eth2 -q -A console -c /etc/snort/rules/local.rules -Q --daq afpacket -l /home/kali/Desktop/snortlogs
02/29-15:44:57.690898 [Drop] [**] [1:1000006:0] Possible DoS Attack Blocked : SYNflood [**] [Priority: 0] {TCP} 192.168.31.87:59628 → 10.1.0.10:443
02/29-15:44:57.690850 [Drop] [**] [1:1000006:0] Possible DoS Attack Blocked : SYNflood [**] [Priority: 0] {TCP} 192.168.31.87:59628 → 192.168.31.41:443
02/29-15:44:58.010826 [Drop] [**] [1:1000006:0] Possible DoS Attack Blocked : SYNflood [**] [Priority: 0] {TCP} 192.168.31.87:38002 → 10.1.0.10:443
02/29-15:44:58.010777 [Drop] [**] [1:1000006:0] Possible DoS Attack Blocked : SYNflood [**] [Priority: 0] {TCP} 192.168.31.87:38002 → 192.168.31.41:443
02/29-15:44:58.327269 [Drop] [**] [1:1000006:0] Possible DoS Attack Blocked : SYNflood [**] [Priority: 0] {TCP} 192.168.31.87:38008 → 10.1.0.10:443
02/29-15:44:58.327203 [Drop] [**] [1:1000006:0] Possible DoS Attack Blocked : SYNflood [**] [Priority: 0] {TCP} 192.168.31.87:38008 → 192.168.31.41:443
02/29-15:44:58.646793 [Drop] [**] [1:1000006:0] Possible DoS Attack Blocked : SYNflood [**] [Priority: 0] {TCP} 192.168.31.87:38020 → 10.1.0.10:443
02/29-15:44:58.646739 [Drop] [**] [1:1000006:0] Possible DoS Attack Blocked : SYNflood [**] [Priority: 0] {TCP} 192.168.31.87:38020 → 192.168.31.41:443
02/29-15:44:58.964857 [Drop] [**] [1:1000006:0] Possible DoS Attack Blocked : SYNflood [**] [Priority: 0] {TCP} 192.168.31.87:38028 → 10.1.0.10:443
02/29-15:44:58.964810 [Drop] [**] [1:1000006:0] Possible DoS Attack Blocked : SYNflood [**] [Priority: 0] {TCP} 192.168.31.87:38028 → 192.168.31.41:443
```

Εικόνα 79: Μπλοκάρισμα http flood επίθεσης



Όπως παρατηρούμε και στο επόμενο screenshot από το kali που τρέξαμε την επίθεση, το python script έτρεξε 20 φορές και μετά τερματίστηκε λόγω refused connection

```
kali@kali: ~  
File Actions Edit View Help  
  
File "/home/kali/httpflood.py", line 7, in <module>  
    r=requests.get("https://192.168.31.41/",verify=False)  
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^  
  
File "/usr/lib/python3/dist-packages/requests/api.py", line 73, in get  
    return request("get", url, params=params, **kwargs)  
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^  
  
File "/usr/lib/python3/dist-packages/requests/api.py", line 59, in request  
    return session.request(method=method, url=url, **kwargs)  
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^  
  
File "/usr/lib/python3/dist-packages/requests/sessions.py", line 589, in re  
quest  
    resp = self.send(prepare_request(urllib3.util.url_to_http(url), headers=headers, auth=  
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^  
  
File "/usr/lib/python3/dist-packages/requests/sessions.py", line 703, in se  
nd  
    r = adapter.send(request, **kwargs)  
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^  
  
File "/usr/lib/python3/dist-packages/requests/adapters.py", line 519, in se  
nd  
    raise ConnectionError(e, request=request)  
requests.exceptions.ConnectionError: HTTPSConnectionPool(host='192.168.31.41'  
, port=443): Max retries exceeded with url: / (Caused by NewConnectionError('urllib3.connection.HTTPSConnection object at 0x7f31f5bf6550': Failed to esta  
blish a new connection: [Errno 111] Connection refused'))  
  
[kali@kali]-[~]  
$
```

### Εικόνα 80: Connection Refused



```
# Generated by iptables-save v1.8.10 (nf_tables) on Thu Feb 29 17:02:10 2024
*mangle
:PREROUTING ACCEPT [47:4953]
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A PREROUTING -m conntrack --ctstate INVALID -j DROP
-A PREROUTING -p tcp -m tcp ! --tcp-flags FIN,SYN,RST,ACK SYN -m conntrack -
ctstate NEW -j DROP
-A PREROUTING -p tcp -m conntrack --ctstate NEW -m tcpmss ! --mss
536:65535 -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN FIN,SYN -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags SYN,RST SYN,RST -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,RST FIN,RST -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,ACK FIN -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags ACK,URG URG -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags PSH,ACK PSH -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG NONE -j
DROP
COMMIT
# Completed on Thu Feb 29 17:02:10 2024
# Generated by iptables-save v1.8.10 (nf_tables) on Thu Feb 29 17:02:10 2024
*filter
:INPUT DROP [117:28651]
:FORWARD DROP [12353:556562]
:OUTPUT DROP [2200:145572]
:DmzOnline - [0:0]
:Dmznet - [0:0]
:ltznet - [0:0]
-A INPUT -d 10.0.0.0/16 -p icmp -m icmp --icmp-type 8 -j ACCEPT
-A INPUT -s 10.0.0.0/16 -p icmp -m icmp --icmp-type 0 -j ACCEPT
-A FORWARD -d 10.0.0.0/16 -i eth0 -j ltznet
-A FORWARD -s 10.0.0.0/16 -i eth1 -j ltznet
```



```
-A FORWARD -j Dmznet
-A OUTPUT -d 10.0.0.0/16 -p icmp -m icmp --icmp-type 0 -j ACCEPT
-A OUTPUT -d 10.0.0.0/16 -p icmp -m icmp --icmp-type 8 -j ACCEPT
-A DmzOnline -s 10.1.0.0/16 -i eth2 -j ACCEPT
-A DmzOnline -d 10.1.0.0/16 -i eth0 -j ACCEPT
-A Dmznet -d 10.1.0.0/16 -i eth0 -m state --state NEW,ESTABLISHED -j ACCEPT
-A Dmznet -s 10.1.0.0/16 -i eth2 -m state --state RELATED,ESTABLISHED -j
ACCEPT
-A Itznet -s 10.0.0.0/16 -i eth1 -p udp -m udp --dport 53 -j ACCEPT
-A Itznet -s 10.0.0.0/16 -i eth1 -p tcp -m tcp --dport 443 -m state --state NEW -j
ACCEPT
-A Itznet -s 10.0.0.0/16 -i eth1 -p tcp -m tcp --dport 80 -m state --state NEW -j
ACCEPT
-A Itznet -d 10.0.0.0/16 -i eth0 -m state --state RELATED,ESTABLISHED -j
ACCEPT
-A Itznet -s 10.0.0.0/16 -i eth1 -m state --state RELATED,ESTABLISHED -j
ACCEPT
-A Itznet -d 10.0.0.0/16 -i eth0 -j LOG --log-prefix "Suspicious: "
-A Itznet -j DROP
COMMIT
# Completed on Thu Feb 29 17:02:10 2024
# Generated by iptables-save v1.8.10 (nf_tables) on Thu Feb 29 17:02:10 2024
*nat
:PREROUTING ACCEPT [8352:1040933]
:INPUT ACCEPT [164:15484]
:OUTPUT ACCEPT [2295:171599]
:POSTROUTING ACCEPT [2398568:383724956]
-A PREROUTING -i eth1 -p udp -m udp --dport 53 -j DNAT --to-destination
192.168.31.123
-A PREROUTING -d 192.168.31.41/32 -i eth0 -p tcp -m tcp --dport 80 -j DNAT --
to-destination 10.1.0.3
-A POSTROUTING -s 10.0.0.0/16 -o eth0 -j MASQUERADE
-A POSTROUTING -s 10.1.0.0/16 -o eth0 -j MASQUERADE
COMMIT
# Completed on Thu Feb 29 17:02:10 2024
```



