

ÖNSÖZ

Bu projede yol gösterici olan Sayın Yrd. Doç. Dr. Mehmet Demirci Hocamıza, Bilgisayar Mühendisi Jankowski Damian'a ve Ankara Kent Park NOKIA'da SDN Data Center alanı üzerinde çalışan Kürşat Göl'e yardımları için içten teşekkürlerimizi sunarız.

Haziran, 2017

Ankara

İÇİNDEKİLER

	Sayfa
1.KISALTMA LİSTESİ	3
2.ÖZET.....	4
3.GİRİŞ	4
4.MALZEME VE YÖNTEM	5
5.TARTIŞMA VE SONUÇ.....	19
6.KAYNAKÇA.....	20
EKLER.....	21

1.KISALTMA LİSTESİ

SDN : Yazılım Tanımlı Ağlar (Software Defined Network)

IDS : Saldırı Tespit Sistemi (Intrusion Detection System)

IPS : Saldırı Engelleme Sistemi (Intrusion Prevention System)

2.ÖZET

Bu çalışmanın amacı, günümüzde büyük bir sorun haline gelen ağ saldırılarını tespit ederek kullanıcıyı bu saldırı girişiminden haberdar etmek ve bu saldırıları engellemeye çalışmaktır. Bu ağ saldırı tespit ve önleme sistemi SDN tabanlı geliştirilmiştir.

Projenin yapım aşamasında yapılan saldırıları tespit edip engellemek amacıyla Snort IDS ve Snort IPS kullanılmıştır [4]. Sanal makine üzerinde Victim, Attacker sunucuları ve Snort-Router yönlendiricisi oluşturulmuştur. Snort-Router, Attacker sunucusundan Victim sunucusuna yapılan erişimi oluşturulan kurallar ile drop etmeye çalışmıştır. Mininet komutlarıyla oluşturulan SDN, host ve switchler arasındaki bağlantıları görüntülemek için Ryu Controller kullanılmıştır.

3.GİRİŞ

Yazılım Tanımlı Ağ (SDN), şu anda mevcut olan ağ mimarisine farklı bir yaklaşım ile daha esnek daha yönetilebilir daha güzel monitör edilebilir yeni bir yapı ortaya koyan bir teknolojidir. Günümüzde gelişmekte olan bir ağ yönetim teknolojisi olup, ana fikri ağın kontrol ve veri yönlendirme düzlemlerini birbirinden ayırmaktır. SDN'nin bu ayrımı yapmaktaki amacı ağların bakım yükünü büyük ölçüde azaltmak, kaynakların daha verimli kullanılmasını sağlamak ve kontrol mekanizmasını merkezi hale getirerek ağın yönetilmesini kolaylaştırmaktır [1-5].

Günümüzde saldırı teknikleri o kadar karmaşık bir hal almıştır ki bazen sadece bir güvenlik duvarı ile bu saldırıları engellemek pek mümkün olmamaktadır. Bu nedenle internetten gelen veri paketlerini inceleyen (Saldırı Tespit Sistemi), bu paketlerdeki veriyi önceki paketlerle karşılaştıran ve saldırı olma ihtimali olan paketlerin geçmesini engelleyecek sistemlere ihtiyaç olmuştur. Bu tip sistemlere **Saldırı Önleme Sistemleri (Intrusion Prevention Systems)** denilmektedir [2].

Ağ saldırı tespit sistemleri, tüm tedbirlere karşın bilgisayar sistemlerine yapılan saldırıları gerçekleştirirken ya da gerçekleştikten sonra tespit etmek, internet veya yerel ağdan gelebilecek, ağdaki sistemlere zarar verebilecek, çeşitli paket ve verilerden oluşan bu saldırıları fark etmek üzere tasarlanmış sistemlerdir ve bu saldırılara yanıt vermeyi amaçlayan bir güvenlik teknolojisidir. Ağ saldırı önleme sistemleri ise gelen paketleri

inceleyerek zararlı olup olmayacağına karar verir ve buna göre aksiyon alır. Hatta günümüzde IPS cihazları güvenlik duvarından önce önde konumlandırılmakta olup paketleri inceler. Eğer paketler zararlı kodlar (kurtçuklar, Truva atları, uygulama seviyesi saldırılar, önbellek taşmaları gibi) içeriyorsa IPS, tehditlere karşı kapsamlı koruma sağlamaktadır [2].

Ağ saldırı tespit ve engelleme sistemlerinde SDN kullanılarak ağların kontrolü tek bir kontrol noktasıyla sağlanır. Bu yapıyı kullanan saldırı engelleme sistemine örnek olarak Sphinx verilebilir. Sphinx uygulamasının ana çalışma mantığı gerçek zamanlı olarak artan ağ akış grafikleri oluşturmak ve tüm SDN durumunu doğrulamak için OpenFlow kontrol mesajlarından topolojik ve yönlendirme meta verilerini toplamaktır [7]. Bu veriler aracılığıyla da topoloji üzerine yapılan güvenlik saldırılarını tespit ederek engeller.

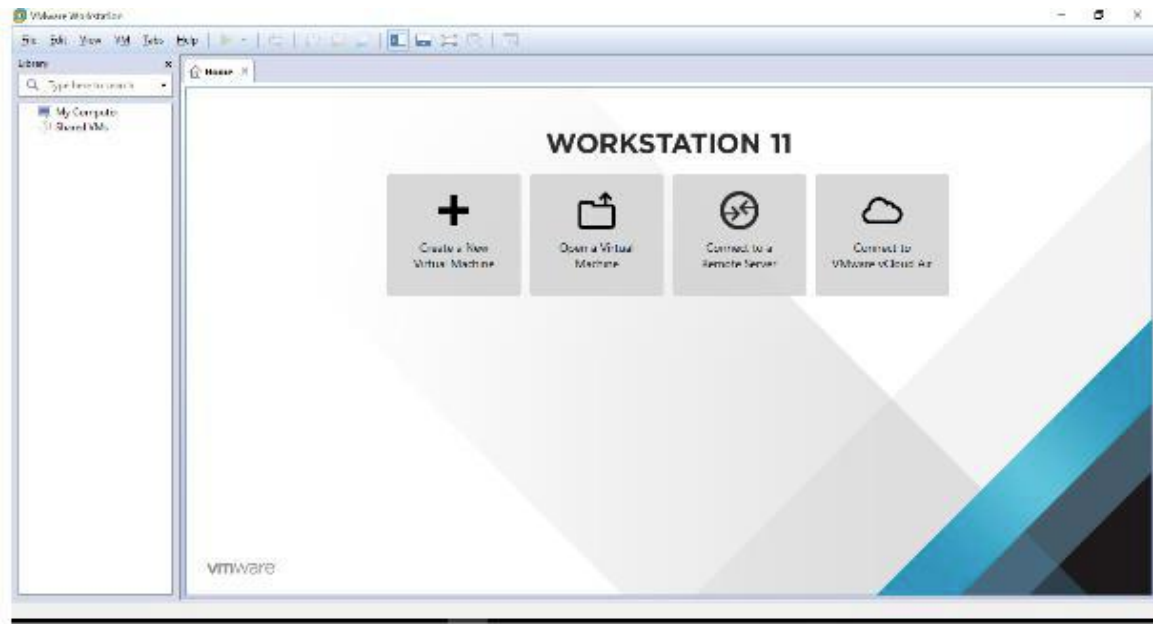
Var olan çalışmalar göz önüne alınarak bu projede SDN tanımlı ağ saldırı tespit ve önleme sistemi geliştirilmiştir.

Saldırı tespit ve engelleme sistemi olarak Snort'un tercih edilmesinin sebebi Snort'ta var olan kuralların ihtiyaçlar doğrultusunda kolay bir şekilde değiştirilebilmesidir. Snort'u inline modda yapılandırarak IPS olarak kullanabiliyoruz. Yani Snort, sisteme sızma girişimleri tespit etmekle beraber bu girişimlere ait paketlerin de silinmesini sağlıyor. Bu inline modu ise paketler iptables üzerinden yönlendirme yapılarak Snort'a iletiliyor ve yönlendirilmiş olan bu paketler yine daha önceden tanımlanmış kurallar uyarınca geçiriliyor ya da drop ediliyor. Yani kendi başına IDS olarak çalışan Snort, inline modunda çalıştırılarak IPS özelliği kazanmış oluyor [3].

4.MALZEME VE YÖNTEM

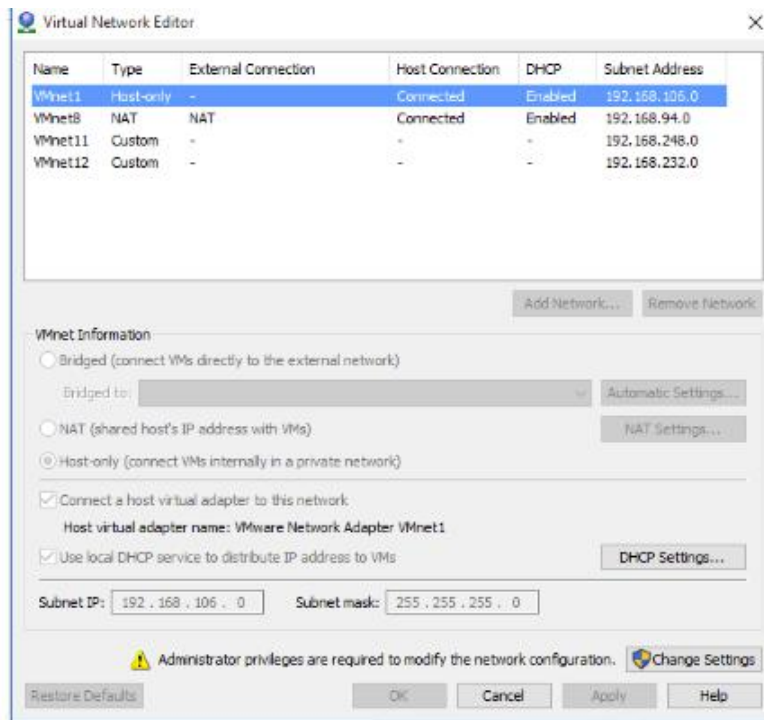
Çalışmada VMware Workstation sanal makinesinde 3 adet sanal makine oluşturuldu. İlkinde **Snort Router** için bir Ubuntu işletim sistemi oluşturuldu. Bu makinede Snort kurulumu yapıldı, kurallar oluşturuldu. 2. sanal makinede (**Victim**), ağ saldırısına uğrayacak sunucu için bir Ubuntu işletim sistemi oluşturuldu. 3.sanal makinede (**Attacker** sunucusu), **Victim** sunucusuna karşı ağ saldırısını gerçekleştirecek bir Kali Linux işletim sistemi oluşturuldu. Bu sanal makinelerin oluşturulma aşamaları şöyledir:

VMware Workstation programının kurulumunu gerçekleştirdikten sonra açıyoruz:



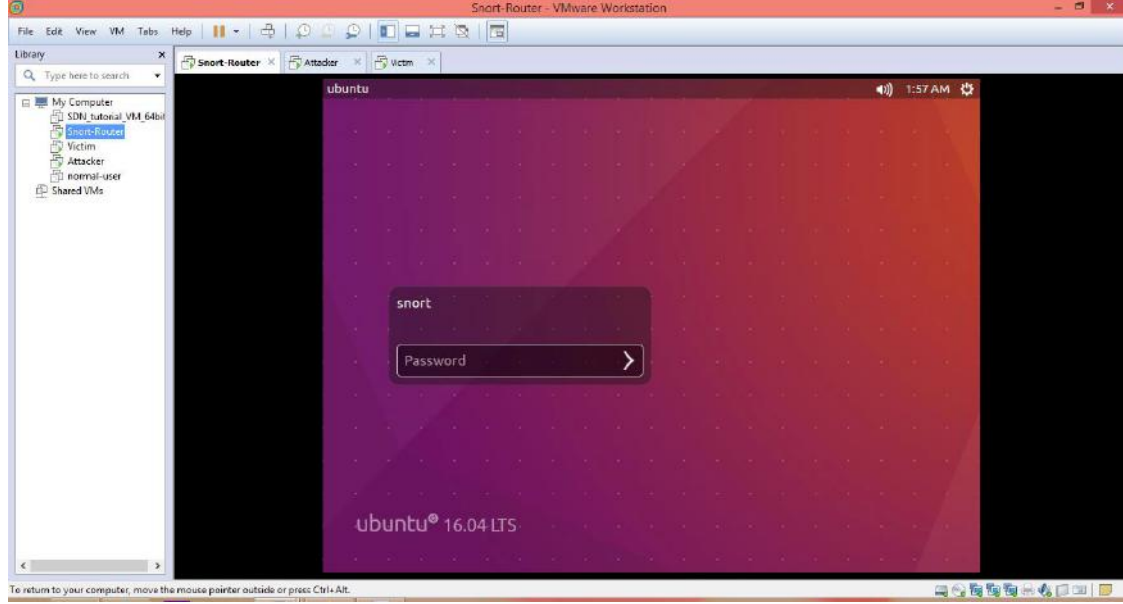
Şekil 4.1 VMware Workstation 11

Victim ve **Attacker** sunucuları için 2 adet network adapter ekliyoruz. **Victim** için Vmnet11 virtual switch'ini seçiyoruz, Subnet IP değeri olarak 192.168.248.0/24, Subnet mask değeri olarak 255.255.255.0 değerlerini veriyoruz. **Attacker** için Vmnet12 virtual switch'ini seçiyoruz. Subnet IP değeri olarak 192.168.232.0/24, Subnet mask değeri olarak 255.255.255.0 değerlerini veriyoruz:



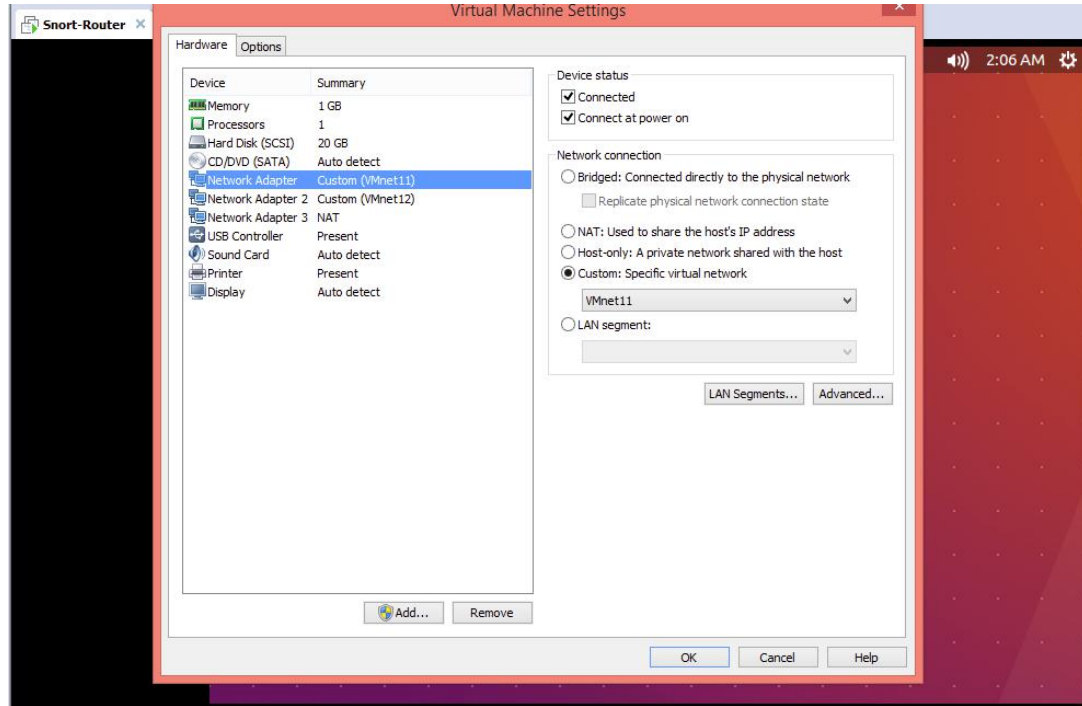
Şekil 4.2 Virtual Network Editor

VMware üzerinde **Snort-Router** sanal makinesini oluşturuyoruz. **Snort-Router**, **Victim** ve **Attacker** sunucuları arasında yönlendirilen paketleri incelemek için oluşturuldu:



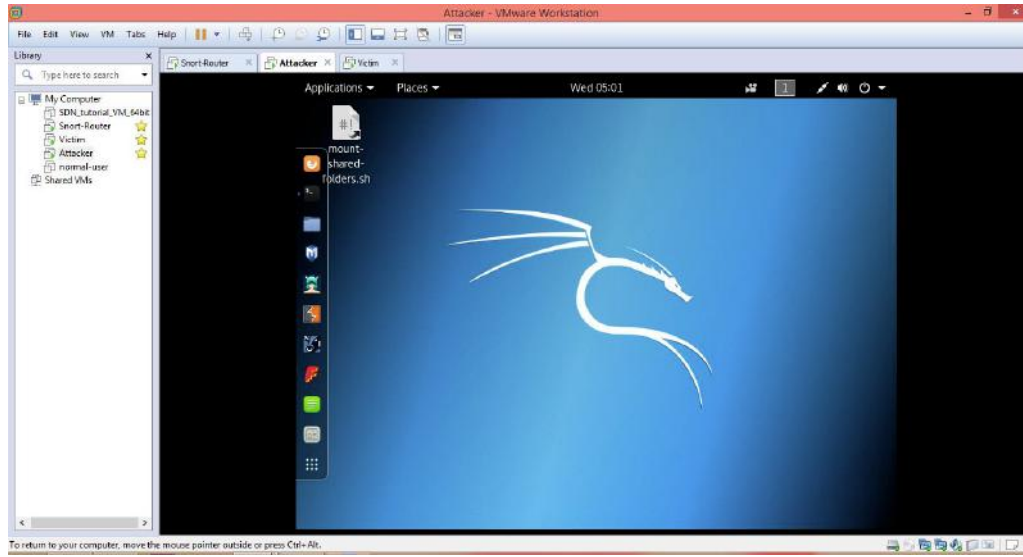
Şekil 4.3 Snort-Router

Snort-Router'ın **Victim** ve **Attacker** sunucuları arasındaki yönlendirmeyi sağlayabilmesi bu sunucuların network adaptörlerini **Snort-Router**'a ekledik. **Snort-Router**'in internet bağlantısını sağlayabilmek için ayrı bir NAT adaptörü de eklendi:



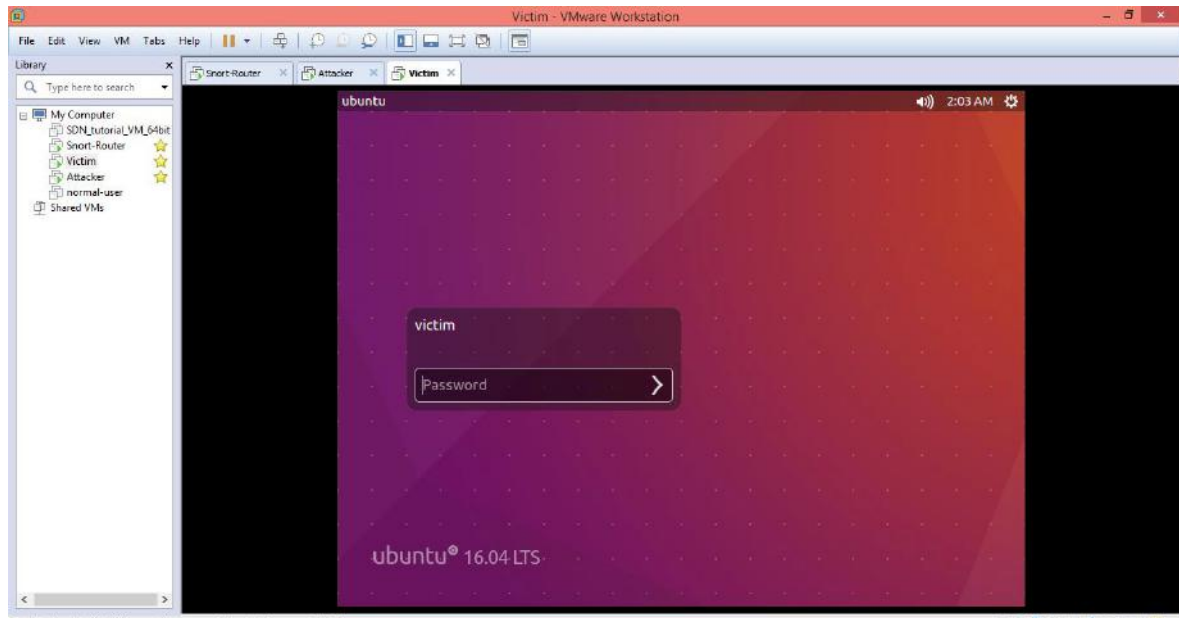
Şekil 4.4 Snort-Router settings

VMware üzerinde **Attacker** sanal makinesini oluşturuyoruz. **Attacker**, Kali Linux işletim sisteminde kuruldu ve **Victim** sunucusuna ağ saldırısı yapmak amacıyla oluşturuldu:



Şekil 4.5 Attacker

VMware üzerinde **Victim** sanal makinesini oluşturuyoruz. **Victim**, Ubuntu işletim sisteminde kuruldu ve ağ saldırısına uğramak amacıyla oluşturuldu:



Şekil 4.6 Victim

sudo apt-get install snort komutuyla **Snort-Router**'da Snort programının kurulumunu gerçekleştirdik:


```

snort@ubuntu: ~
snort@ubuntu:~$ sudo apt-get install snort
Reading package lists... Done
Building dependency tree
Reading state information... Done
snort is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 249 not upgraded.

```

Şekil 4.7 Snort kurulumu

sudo nano /etc/network/interfaces komutuyla *eth0* (**Victim** için), *eth1* (**Attacker** için) IP adresleri belirlendi. Bu IP adresleri Victim ve Attacker için **gateway** olarak kullanıldı:

```

GNU nano 2.5.3      File: /etc/network/interfaces
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.248.1
netmask 255.255.255.0

auto eth1
iface eth1 inet static
address 192.168.232.1
netmask 255.255.255.0

```

Şekil 4.8 IP adresleri belirleme

ifconfig komutuyla ethernetlere atanan IP adreslerini gösterdik:

```

snort@ubuntu: ~
snort@ubuntu:~$ sudo nano /etc/network/interfaces
[sudo] password for snort:
snort@ubuntu:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:a7:95:55
          inet addr:192.168.248.1  Bcast:192.168.248.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fea7:9555/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:250043 errors:0 dropped:0 overruns:0 frame:0
          TX packets:312860 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:51660032 (51.6 MB)  TX bytes:35277180 (35.2 MB)

eth1      Link encap:Ethernet  HWaddr 00:0c:29:a7:95:5f
          inet addr:192.168.232.1  Bcast:192.168.232.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fea7:955f/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:250230 errors:0 dropped:0 overruns:0 frame:0
          TX packets:312958 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:26620919 (26.6 MB)  TX bytes:60322396 (60.3 MB)

eth2      Link encap:Ethernet  HWaddr 00:0c:29:a7:95:69
          inet addr:192.168.48.129  Bcast:192.168.48.255  Mask:255.255.255.0
          inet6 addr: fe80::c81b:97bd:9ced:f5fa/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10470 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6719 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6971771 (6.9 MB)  TX bytes:1334411 (1.3 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128  Scope:Host

```

Şekil 4.9 Ethernetleri görüntüleme

`sudo apt-get install nmap` komutuyla **Attacker**'da IP taraması yapabilmek için **nmap** kurulumunu yaptık:

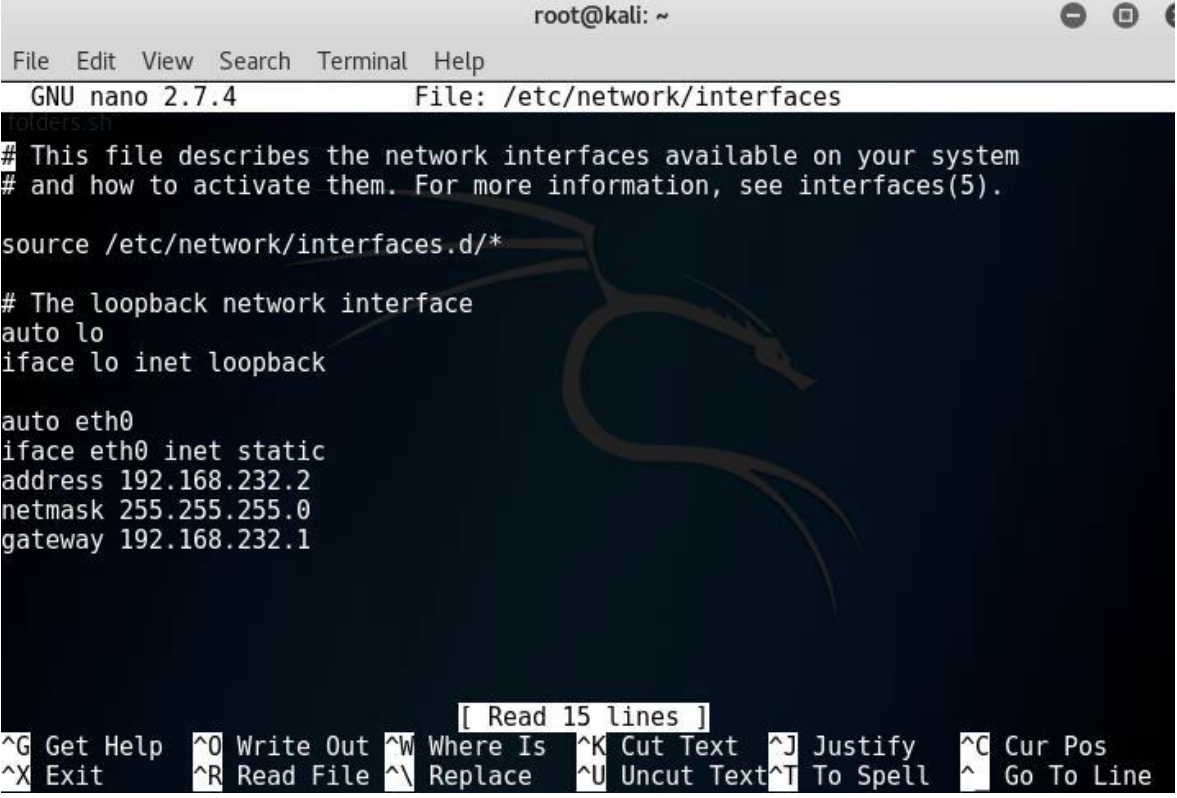
```

root@kali:~# sudo apt-get install nmap
Reading package lists... Done
Building dependency tree
Reading state information... Done
nmap is already the newest version (7.40-1kali1).
0 upgraded, 0 newly installed, 0 to remove and 394 not upgraded.
root@kali:~#

```

Şekil 4.10 nmap kurulumu

`sudo nano /etc/network/interfaces` komutuyla **eth0** (**Attacker** için) IP adresi ve gateway belirlendi:



```

root@kali: ~
File Edit View Search Terminal Help
GNU nano 2.7.4 File: /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

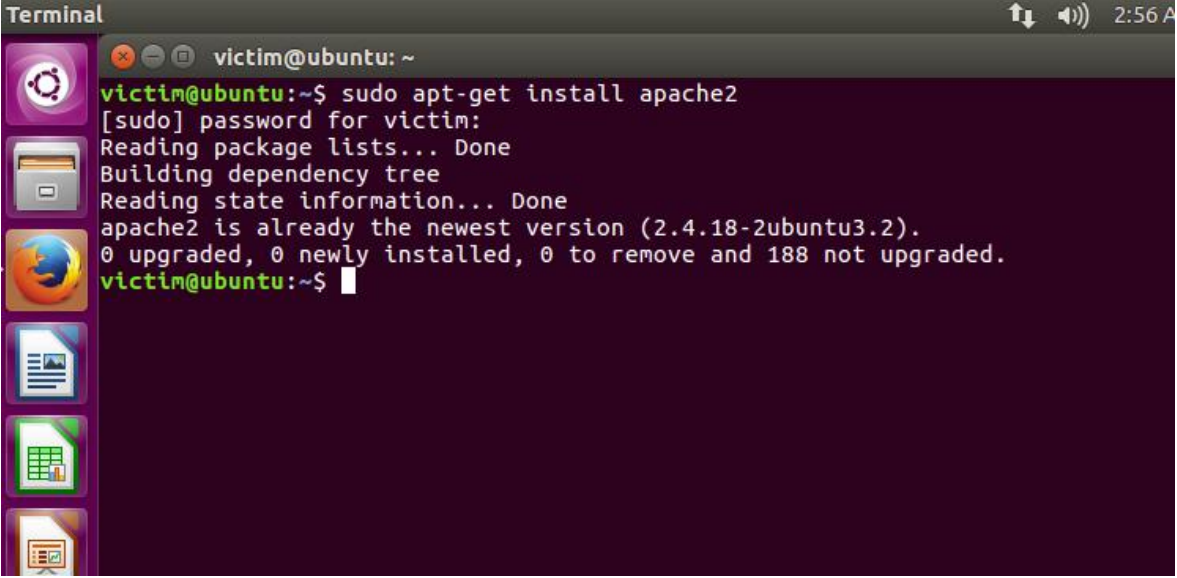
auto eth0
iface eth0 inet static
address 192.168.232.2
netmask 255.255.255.0
gateway 192.168.232.1

[ Read 15 lines ]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line

```

Şekil 4.11 IP adresi ve gateway belirleme

Apache web sunucusunun kurulumu yapıldı:



```

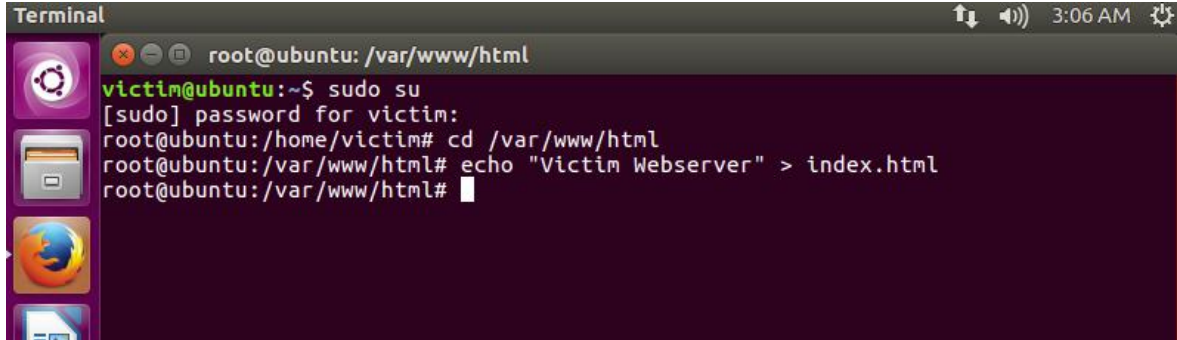
Terminal
victim@ubuntu: ~
victim@ubuntu:~$ sudo apt-get install apache2
[sudo] password for victim:
Reading package lists... Done
Building dependency tree
Reading state information... Done
apache2 is already the newest version (2.4.18-2ubuntu3.2).
0 upgraded, 0 newly installed, 0 to remove and 188 not upgraded.
victim@ubuntu:~$

```

Şekil 4.12 Apache server kurulumu

Victim sunucusunun web server'ına bağlanabilmek için gerekli olan kimlik doğrulama ve bağlandıktan sonra web'de görünecek mesajı düzenlemek için birkaç adım uygulandı. Bu adımlar sırasıyla şöyledir:

İlk olarak servera bağlandığımızda ekranda görünecek mesaj belirlendi:



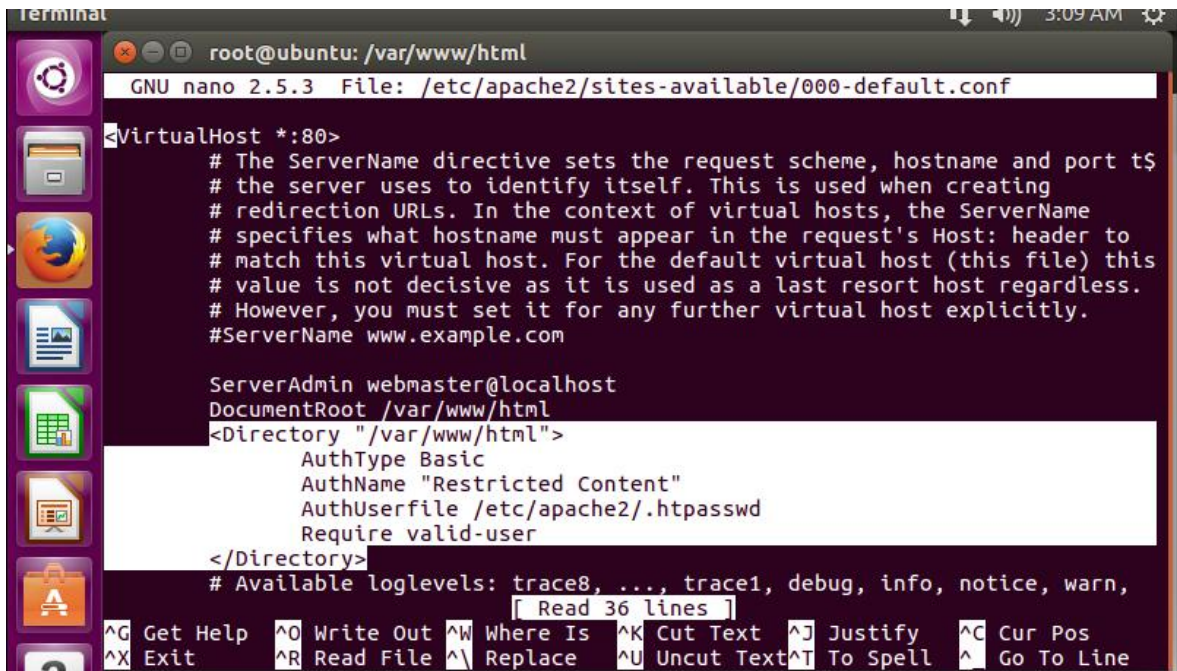
```

Terminal
root@ubuntu: /var/www/html
victim@ubuntu:~$ sudo su
[sudo] password for victim:
root@ubuntu:/home/victim# cd /var/www/html
root@ubuntu:/var/www/html# echo "Victim Webserver" > index.html
root@ubuntu:/var/www/html#

```

Şekil 4.13 Html mesajının belirlenmesi

`nano /etc/apache2/sites-available/000-default.conf` komutuyla içine girilen dosyada şekildeki düzenleme yapıldı:



```

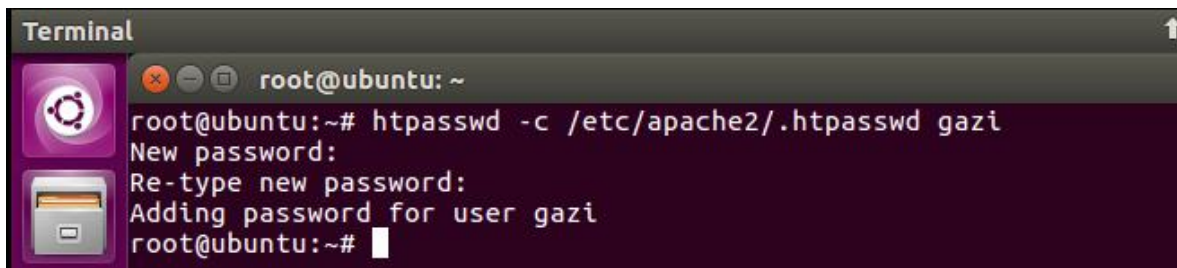
Terminal
root@ubuntu: /var/www/html
GNU nano 2.5.3 File: /etc/apache2/sites-available/000-default.conf
<VirtualHost *:80>
# The ServerName directive sets the request scheme, hostname and port to
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
#ServerName www.example.com

ServerAdmin webmaster@localhost
DocumentRoot /var/www/html
<Directory "/var/www/html">
    AuthType Basic
    AuthName "Restricted Content"
    AuthUserFile /etc/apache2/.htpasswd
    Require valid-user
</Directory>
# Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
[ Read 36 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text ^T To Spell  ^_ Go To Line

```

Şekil 4.14 000-default.conf dosyasının düzenlenmesi

Dosya düzenlendikten sonra servera giriş yapacağımız kullanıcı adı ve şifresi belirlendi:



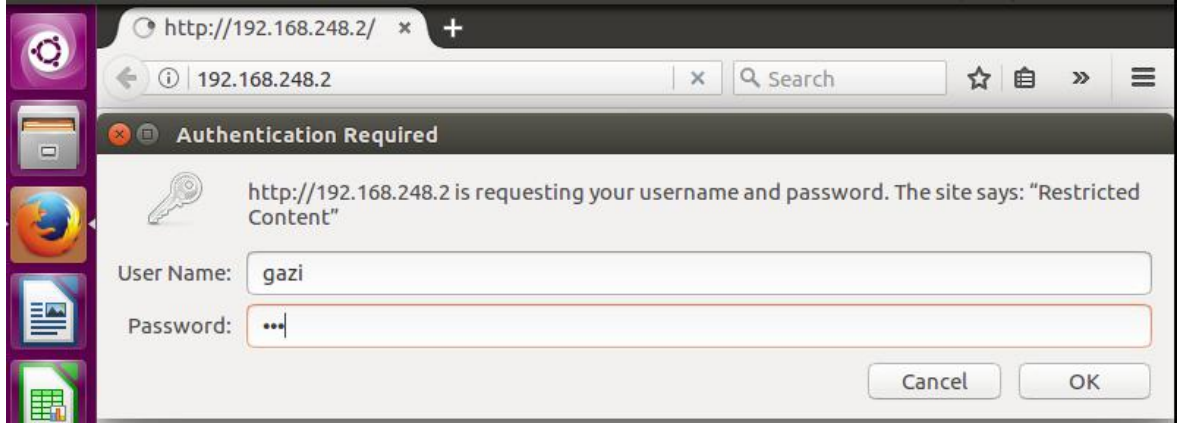
```

Terminal
root@ubuntu: ~
root@ubuntu:~# htpasswd -c /etc/apache2/.htpasswd gazi
New password:
Re-type new password:
Adding password for user gazi
root@ubuntu:~#

```

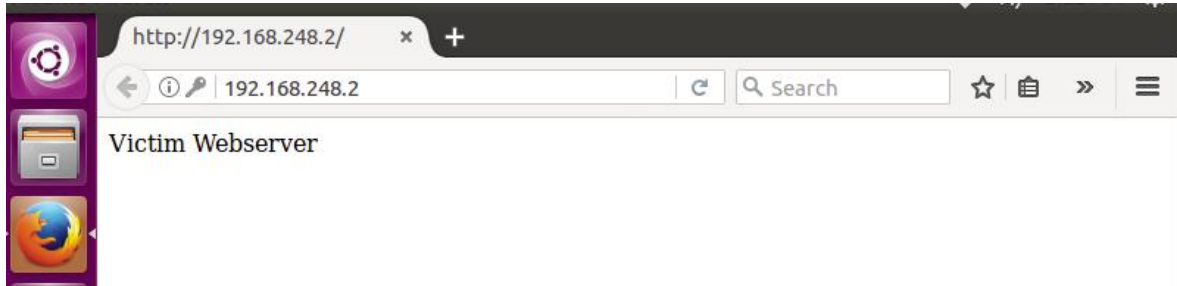
Şekil 4.15 Kullanıcı adı ve şifresinin belirlenmesi

Yapmış olduğumuz düzenlemelerin derlendiğini görmek için bir web tarayıcı açıp **Victim** sunucusunun server'ına bağlanmaya çalışıyoruz:



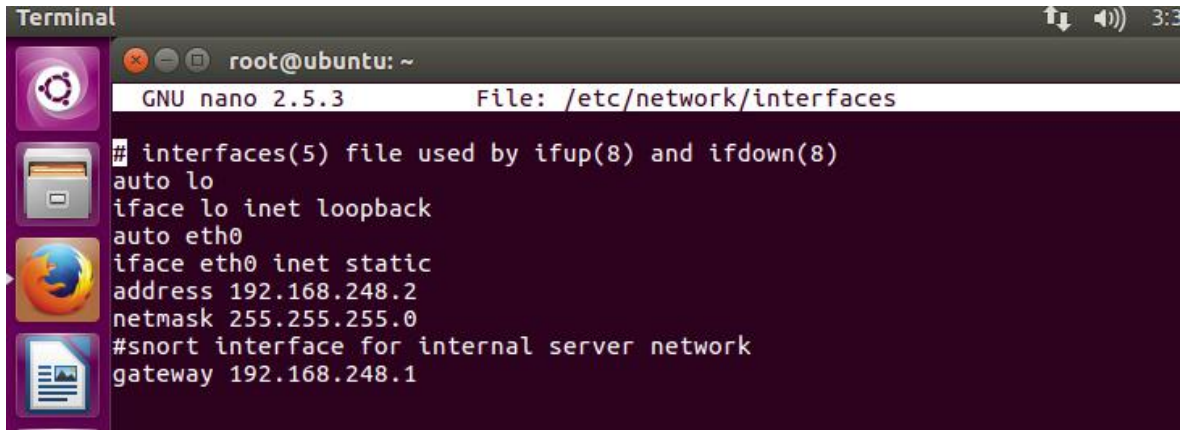
Şekil 4.15 Kimlik doğrulamasının yapılması

Kimlik doğrulamasının başarı ile yapılmasının ardından önceden belirlemiş olduğumuz mesaj tarayıcıda görüntülendi:



Şekil 4.16 Server mesajı

nano /etc/network/interfaces komutuyla içine girmiş olduğumuz dosyada *eth0* (**Attacker** için) düzenlemesi yaparak *eth0* için IP adresi ve gateway belirlendi:



```

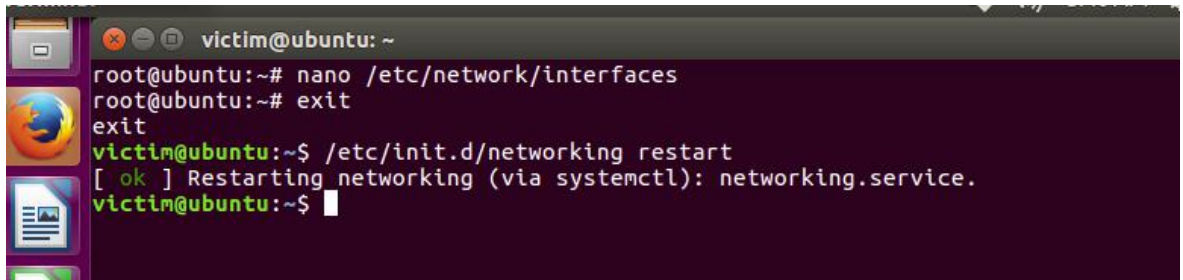
Terminal
root@ubuntu: ~
GNU nano 2.5.3 File: /etc/network/interfaces

# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback
auto eth0
iface eth0 inet static
address 192.168.248.2
netmask 255.255.255.0
#snort interface for internal server network
gateway 192.168.248.1

```

Şekil 4.17 IP adresi ve gateway belirleme

Sonrasında ise SSH, FTP, DNS ve TELNET için kurulumlar gerçekleştirildi. İşlemler bittikten sonra ağı yeniden başlattık:



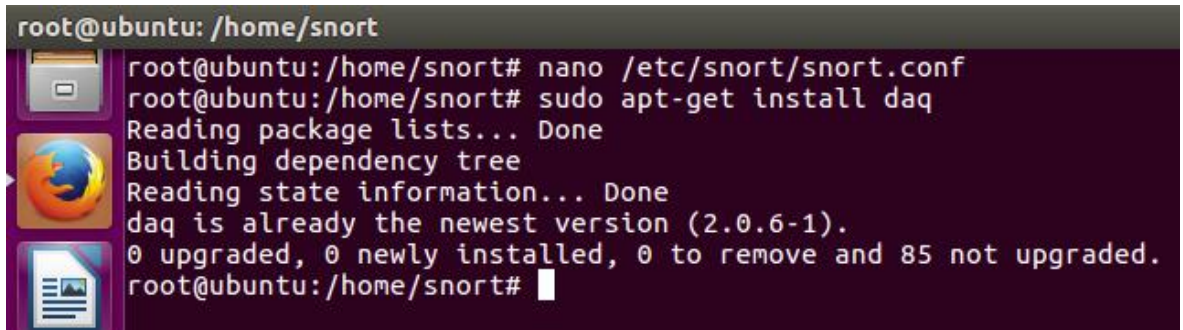
```

victim@ubuntu: ~
root@ubuntu:~# nano /etc/network/interfaces
root@ubuntu:~# exit
exit
victim@ubuntu:~$ /etc/init.d/networking restart
[ ok ] Restarting networking (via systemctl): networking.service.
victim@ubuntu:~$

```

Şekil 4.18 Ağı yeniden başlatılması

Snort-Router'da, paket yakalama için doğrudan PCAP'in kullanılması yerine PCAP'in üzerinde soyut bir ara katman görevi yapmakta olan DAQ (*Data Acquisition Library*) kütüphanesi kuruldu:



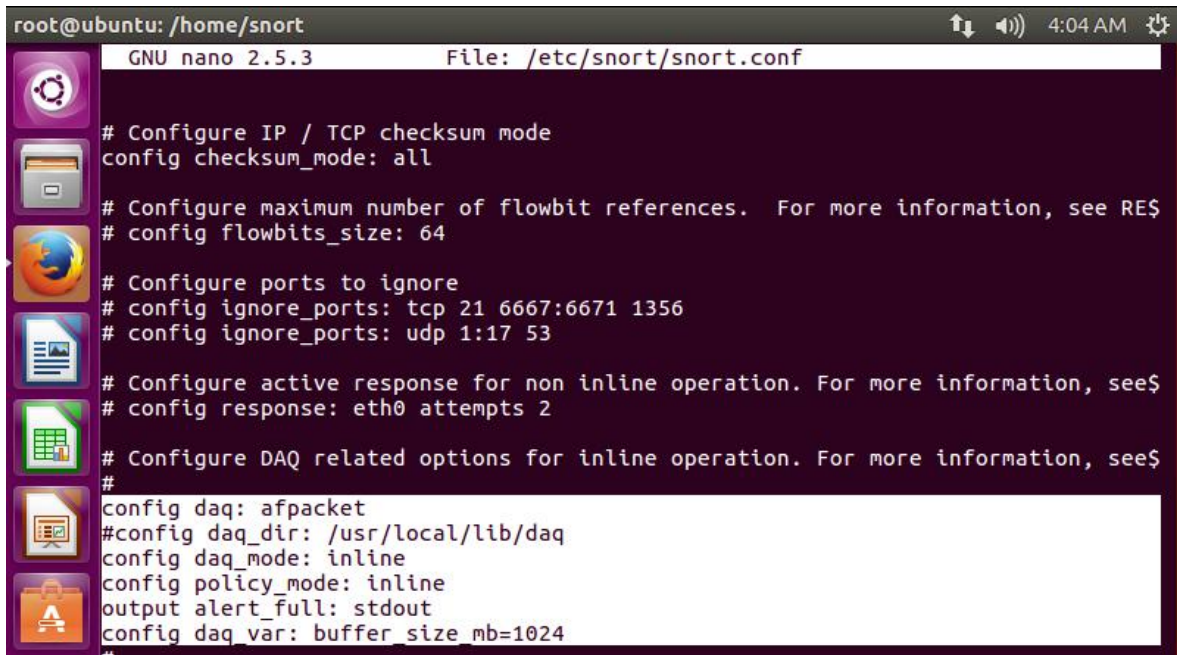
```

root@ubuntu: /home/snort
root@ubuntu:/home/snort# nano /etc/snort/snort.conf
root@ubuntu:/home/snort# sudo apt-get install daq
Reading package lists... Done
Building dependency tree
Reading state information... Done
daq is already the newest version (2.0.6-1).
0 upgraded, 0 newly installed, 0 to remove and 85 not upgraded.
root@ubuntu:/home/snort#

```

Şekil 4.19 DAQ kütüphanesinin kurulumu

DAQ kütüphanesi birkaç moda çalışabilmektedir, biz bunlardan *afpacket*'i etkinleştirmek için `nano /etc/snort/snort.conf` komutu ile girdiğimiz dosyada düzenlemeler yaptık:



```

root@ubuntu: /home/snort
GNU nano 2.5.3 File: /etc/snort/snort.conf

# Configure IP / TCP checksum mode
config checksum_mode: all

# Configure maximum number of flowbit references. For more information, see RE$
# config flowbits_size: 64

# Configure ports to ignore
# config ignore_ports: tcp 21 6667:6671 1356
# config ignore_ports: udp 1:17 53

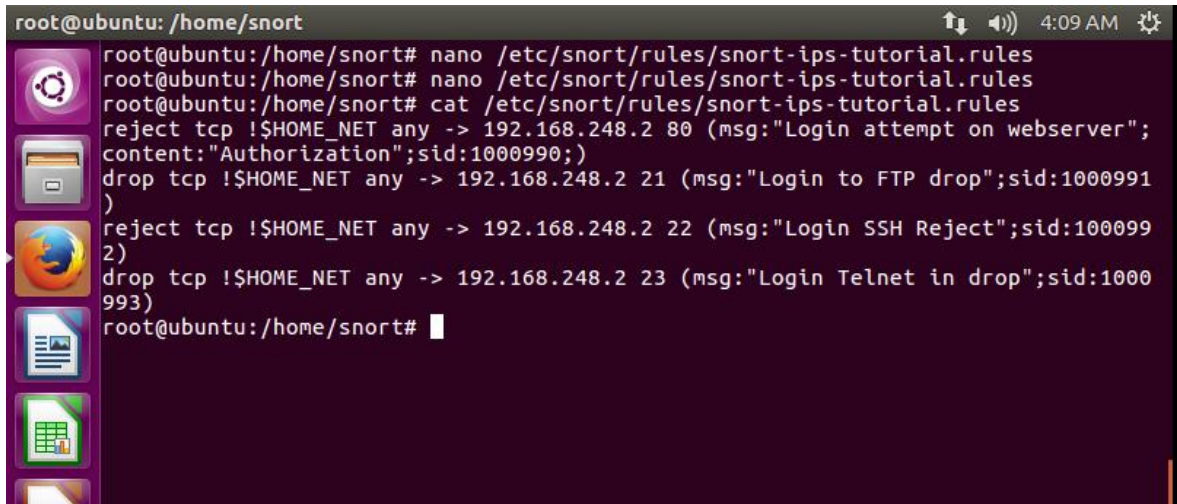
# Configure active response for non inline operation. For more information, see$
# config response: eth0 attempts 2

# Configure DAQ related options for inline operation. For more information, see$
#
config daq: afpacket
#config daq_dir: /usr/local/lib/daq
config daq_mode: inline
config policy_mode: inline
output alert_full: stdout
config daq_var: buffer_size mb=1024
#

```

Şekil 4.20 /etc/snort/snort.conf dosyasının düzenlenmesi

nano /etc/snort/rules/snort-ips-tutorial.rules komutuyla kurallar ekledik. *Cat /etc/snort/rules/snort-ips-tutorial* komutuyla yazılan kuralları gösterdik:



```

root@ubuntu: /home/snort
root@ubuntu:/home/snort# nano /etc/snort/rules/snort-ips-tutorial.rules
root@ubuntu:/home/snort# nano /etc/snort/rules/snort-ips-tutorial.rules
root@ubuntu:/home/snort# cat /etc/snort/rules/snort-ips-tutorial.rules
reject tcp !$HOME_NET any -> 192.168.248.2 80 (msg:"Login attempt on webserver";
content:"Authorization";sid:1000990;)
drop tcp !$HOME_NET any -> 192.168.248.2 21 (msg:"Login to FTP drop";sid:1000991
)
reject tcp !$HOME_NET any -> 192.168.248.2 22 (msg:"Login SSH Reject";sid:100099
2)
drop tcp !$HOME_NET any -> 192.168.248.2 23 (msg:"Login Telnet in drop";sid:1000
993)
root@ubuntu:/home/snort#

```

Şekil 4.21 Kurallar

Kuralın derlenebilmesi için *etc/snort/snort.conf* dosyası içine ekledik:

```

root@ubuntu: /home/snort
GNU nano 2.5.3      File: /etc/snort/snort.conf

include $RULE_PATH/telnet.rules \
include $RULE_PATH/tftp.rules \
include $RULE_PATH/virus.rules \
include $RULE_PATH/voip.rules \
include $RULE_PATH/web-activex.rules \
include $RULE_PATH/web-attacks.rules \
include $RULE_PATH/web-cgi.rules \
include $RULE_PATH/web-client.rules \
include $RULE_PATH/web-coldfusion.rules \
include $RULE_PATH/web-frontpage.rules \
include $RULE_PATH/web-iis.rules \
include $RULE_PATH/web-misc.rules \
include $RULE_PATH/web-php.rules \
include $RULE_PATH/x11.rules \
include $RULE_PATH/snort-ips-tutorial.rules \

```

Şekil 4.22 Kuralın eklenmesi

snort -c /etc/snort/snort.conf -i eth0:eth1 -Q > /var/log/snort/snort.log komutuyla Snort'u inline moda afpacket yardımıyla çalıştırıyoruz:

```

root@ubuntu: /home/snort
Reload thread starting...
Reload thread started, thread 0x7f94c9641700 (3579)

---= Initialization Complete =---

-*> Snort! <*-
Version 2.9.9.0 GRE (Build 56)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014-2016 Cisco and/or its affiliates. All rights reserved.

Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.7.4
Using PCRE version: 8.38 2015-11-23
Using ZLIB version: 1.2.8

Rules Engine: SF_SNORT_DETECTION_ENGINE Version 3.0 <Build 1>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_POP Version 1.0 <Build 1>
Preprocessor Object: SF_DNS Version 1.1 <Build 4>
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>

Commencing packet processing (pid=3578)
Decoding Ethernet

```

Şekil 4.23 Snort'un eth0 ve eth1 'i dinlemeye geçmesi

Sonrasında Attacker sunucusundan Victim'e bağlanmaya çalışıyoruz, bu bağlantı isteklerinin reddedildiği mesajı Attacker'da görünmektedir:


```

root@kali:~# nmap -script http-brute -p 80 192.168.248.2
Starting Nmap 7.40 ( https://nmap.org ) at 2017-06-07 07:19 EDT
Nmap scan report for 192.168.248.2
Host is up (0.00071s latency).
PORT      STATE SERVICE
80/tcp    open  http
| http-brute:
|   Accounts: No valid accounts found
|_  Statistics: Performed 50009 guesses in 54 seconds, average tps: 935.1

Nmap done: 1 IP address (1 host up) scanned in 56.07 seconds
root@kali:~# ssh 192.168.248.2
ssh: connect to host 192.168.248.2 port 22: Connection refused
root@kali:~# telnet 192.168.248.2
Trying 192.168.248.2...
telnet: Unable to connect to remote host: Connection refused

```

Şekil 4.24 Attacker'dan Victim'e bağlantı kurma istekleri

Eth0 ve eth1 arasındaki paket gönderiminin dinlenmesi sonlandırıldı ve alınan ve alındıktan sonra drop edilen paketlerin durumu incelendi:

```

Terminal
root@ubuntu: /home/snort

Memory usage summary:
Total non-mmapped bytes (arena):      14356480
Bytes in mapped regions (hblkhd):     21590016
Total allocated space (uordblks):     3394768
Total free space (fordblks):          10961712
Topmost releasable block (keepcost):  9124752

=====
Packet I/O Totals:
Received:      126676
Analyzed:      126676 (100.000%)
Dropped:       873842 ( 87.339%)
Filtered:       0 ( 0.000%)
Outstanding:   0 ( 0.000%)
Injected:       0

=====
Breakdown by protocol (includes rebuilt packets):
Eth:           128816 (100.000%)
VLAN:           0 ( 0.000%)
IP4:           128805 ( 99.991%)
Frag:           0 ( 0.000%)
ICMP:           8 ( 0.006%)
UDP:            0 ( 0.000%)
TCP:           128797 ( 99.985%)
IP6:            0 ( 0.000%)

```

Şekil 4.25.a Paket incelemesi

```

Terminal
root@ubuntu: /home/snort

Tracked: 0

=====
HTTP Inspect - encodings (Note: stream-reassembled packets included):
POST methods: 0
GET methods: 2673
HTTP Request Headers extracted: 2673
HTTP Request Cookies extracted: 0
Post parameters extracted: 0
HTTP response Headers extracted: 0
HTTP Response Cookies extracted: 0
Unicode: 0
Double unicode: 0
Non-ASCII representable: 0
Directory traversals: 0
Extra slashes ("//"): 0
Self-referencing paths ("."): 0
HTTP Response Gzip packets extracted: 0
Gzip Compressed Data Processed: n/a
Gzip Decompressed Data Processed: n/a
Http/2 Rebuilt Packets: 0
Total packets processed: 11472
=====
Snort exiting

```

Şekil 4.25.b Paket incelemesi

```

Terminal
root@ubuntu: /home/snort

TCP Timeouts: 0
TCP Overlaps: 0
TCP Segments Queued: 9516
TCP Segments Released: 9516
TCP Rebuilt Packets: 282
TCP Segments Used: 7286
TCP Discards: 10380
TCP Gaps: 0
UDP Sessions Created: 0
UDP Sessions Deleted: 0
UDP Timeouts: 0
UDP Discards: 0
Events: 1315
Internal Events: 0
TCP Port Filter
Filtered: 0
Inspected: 0
Tracked: 126657
UDP Port Filter
Filtered: 0
Inspected: 0
Tracked: 0
=====

```

Şekil 4.25.c Paket incelemesi

5.TARTIŞMA VE SONUÇ

Bu projede donanımdan bağımsız sanal bir ağ oluşturulmuş ve böylece bu ağın yönetimi daha kolay bir hale gelmiştir. Bir ağın tek bir noktadan yönetilmesi demek o ağa yapılabilecek saldırıların tek bir noktadaki paketlerin incelenmesi sayesinde daha az zahmetle tespit edilip engellenmesi demektir. Bu saldırılara imza vermek bizim işimizi bir nebze de olsa kolaylaştırmaktadır. Çünkü her yeni saldırı türüne karşı baştan yeni bir ürün hazırlamak hiç kolay bir iş değildir. Bunun yerine daha az emek ile işimizi karşılayabiliyor olmamız bir fırsattır.

Ağ tabanlı saldırıların maalesef çok olduğu günümüzde bu saldırılardan korunabilmek için bir takım araçlara ihtiyacımız var. Bunun ülkeler arası boyutunu da göz önüne alırsak şahsi veya kurumsal olan bilgiler gibi devlete ait bilgilerin de bu saldırılardan korunması gerektiğini anlamış oluruz. Siber terörizm kavramının da daha çok kullanılır olduğunu dikkate aldığımızda bir saldırı engelleme sisteminin (IPS) ne kadar önemli olduğunun bir kez daha idrakine varmış bulunmaktayız. Bu konu üzerine daha çok araştırma yapıp yeni projeler üretmemiz gerekmektedir.

7.KAYNAKÇA

1. (İnternet) <https://www.grotto-networking.com/SDNfun.html> (2016)
2. Davis, Z. (2005), Network Intrusion Prevention Systems Product Comparison Guide, Tips-It Publishing, New York.
3. (İnternet) <http://bkarakan96.blogspot.com.tr/> (2016)
4. Dayioğlu, B. (2003). “Snort ile Saldırı Tespit ve Engelleme Sistemi”.
5. Jammal, M., Singh, T., Shami, A., Asal, R., and Li, Y. (2014). *Software defined networking: State of the art and research challenges*. *Computer Networks*, 72, 74-98.
6. Chowdhury, S., R. (2015). “Software Defined Networking Tutorial”. *University of Waterloo*.
7. Dhawan, M., Poddar, R., Mahajan, K., Mann, K. “SPHINX: Detecting Security Attacks in Software-Defined Networks”.

EKLER

Bu kısımda gösterilecek bir ek bulunmamaktadır.