
Api alternative graphql

https://pastebin.com/raw/Afhs4GnH

- Intro to Information Security:

We must understand terms:

1. Information Security[InfoSec]:
``````````````````````

-> What is Information: [Differ b/w Info & Data]
> Meaningful form of data
> Can be called as processed data
> Data has no context, info does
eg:
> Computer is data
> This is my computer is information

NOTE: Information in digital systems makes cyber security [Information involving cyber space].
+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+

2. Information Security Threats:
--------------------------------
> Threat is a constant danger to an asset
> It can be a person, object or an event
> Threat can be categorized and ranked
``````````````````````

Type of Threats :
````````````````

1. Inadvertent Threat [ Human Failure ]
2. Physical Threat [ Natural disasters ]
3. Technical Failure [ Hardware / Software ]
4. Deliberate acts [ Hacking / Espionage ]

+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+

3. Introduction to Cyber Security:
---------------------------------

> Information Security does not deal with?

- Cyber Warfare [Govt Web Hacks, DDOS, etc.]
- Information Warfare [Intel]
- Negative Impact of people on Internet ( sexual abuse, cyber stalking, etc. )
- IoT Security

Then who deal with them?

> Cyber Security:

- Protection of Cyber Space against Cyber Threats and Cyber Space vulnerabilities.
- Any threats to Information via Cyber Space
- Deal with Deliberate acts
- Doesn't deal with physical and personal security
- Threats via Cyber Space, not threats for Cyber Space.

> Objective of Cyber Security:

- Confidentiality : No telling to unauthorized parties
- Integrity : Completeness and accuracy of data
- Availability : When needed, data is available
- Non-repudiation : I should accept i sent you the message and you should accept you received it.
- Authenticity : You should actually be who you tell you are

+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+_+

The Basic Fundamental cin

Home work
Required for Cyber Sec :
``````````````````````

Firewalld & Firewall-cmd:
~~~

-> Configuration
- RunTime
- Permanent

-> MUST BE ACTIVATED

```
# systemctl status firewalld
# systemctl start firewalld

# firewall-cmd --get-zones
```

[THEY ALL HAVE PREDEFINED SET OF RULES]

1. block

[All incoming conn are rejected & only outgoing conn are allowed]

2. dmz

[Used for systems located in Demilitarized zone, only selected incoming conn are allowed in this zone]

3. drop

[All incoming conn are dropped without notify, only outgoing conn are allowed]

4. external

[Used for external networks with NAT masquerading enabled when our syatem acts as a gateway or router, here only selected incoming conn are allowed]

5. home

[Other systems in n/w are generally trusted and only selected incoming conn are allowed]

6. internal

[Used in internal network when your system acts as gateway or router, select incoming conn are alllowed]

7. public
~~~

[Used for the untrusted public areas, do not trust other systems on the network but you can allow selected incoming conn]

8. trusted

[All n/w conn are accepted]

9. work

[All systems in n/w generally trusted and only selected incoming conn are allowed]

Which is the default zone:

```
# firewall-cmd --get-default-zone
```

What are the rules for the default zone:

```
# firewall-cmd --list-all
```

If you want to see the rule for other zones:

```
# firewall-cmd --zone=dmz --list-all
```

If you want to connect the inetrface to 'dmz' from 'public', then the interface will move from 'public' to 'dmz' & and 'dmz' will active:

```
# firewall-cmd --zone=dmz --change-interface=ens33
# firewall-cmd --zone=dmz --add-interface=ens33
```

[will only change for runtime]

```
# firewall-cmd --zone=public --list-all [To check]
# firewall-cmd --zone=dmz --list-all [To check]
```

If you reload the firewall and check the 'dmz', interface will set back to 'public' zone, because of the 'realtime' changes:

```
#firewall-cmd --reload
#firewall-cmd --zone=dmz --list-all [To check]
#firewall-cmd --list-all [To check]
```

To make it permanent

```
#firewall-cmd --zone=dmz --change-interface=ens160 --permanent
#firewall-cmd --reload
```

To check all the services to add in 'public' zone:

```
# firewall-cmd --get-services
```

NOTE: All the services have their individual XML files, are located /usr/lib/firewalld/services

If you want to deny access of SSH on the machine, so you just need to remove SSH service from the public(or current) zone:

```
#firewall-cmd --zone=public --remove-service=ssh --permanent
#firewall-cmd --reload
#firewall-cmd --zone=public --list-all
```

To add any service:

```
#firewall-cmd --zone=public --add-service=ssh --permanent
#firewall-cmd --reload
#firewall-cmd --zone=public --list-all
```

To remove port number:

```
#firewall-cmd --zone=public --remove-port=8080/tcp
```

```
#firewall-cmd --reload
#firewall-cmd --zone=public --list-all
```

To add port number

```
#firewall-cmd --zone=public --add-port=2020/tcp --permanent
#firewall-cmd --reload
#firewall-cmd --zone=public --list-all
```

https://tryhackme.com/room/dnsindetail

https://tryhackme.com/room/httpindetail

https://tryhackme.com/room/walkinganapplication

https://tryhackme.com/room/introwebapplicationsecurity

https://tryhackme.com/room/introtooffensivesecurity

https://tryhackme.com/room/owasptop102021

https://cmdchallenge.com/

https://hackxor.net/

Hack the box.

https://www.hackthebox.com/

Network Address Translation (NAT) : Translate the public to private ip

For adding the rule

Wireshark capturing modes:
~~~

Promiscuous mode:
`````````````````

Sets interface to capture all packets on a network

Monitor mode:
`````````````

Sets wireless interface to capture all the traffic it can receive [unix/linux only]

Logical operator:
``````````````````

1. and or && : Logical AND [all the condition should match]
2. or or || : Logical OR [Either all or one should match]
3. xor or ^^ : Logical XOR [Exclusive alteration - only one of the two condition match not both]
4. not or ! : Not [Not equal to]

Filtering packets [display filter]:
~~~

eq or == : equal [ip.dst == x.x.x.x]

ne or != : Not equal [ip.dst != x.x.x.x]
gt or > : Greater than [frame.len > 10]
it or < : less than [frame.len < 10]
ge or >= : Greater than or equal [frame.len >= 10]
le or <= : Less than or equal [frame.len <= 10]

Common filtering commands:
~~~

Filter by IP:
`````````````

ip.addr == x.x.x.x

filter by dst IP:
`````````````````

ip.dst == x.x.x.x

Filter by source IP:
````````````````````

ip.src == x.x.x.x

Filter by IP range:
````````````````````

ip.addr >= x.x.x.1 and ip.addr <= x.x.x.100

Filter by multiple IPs:
```````````````````````

ip.addr == x.x.x.1 && ip.addr == x.x.x.100
~~~

Filter by subnet:

```
ip.addr == 10.0.0.0/24
```

Filter by port:

```
tcp.port == 25
```

Filter by dst port:

```
tcp.dstport == 23
```

Filter by IP addr and port:

```
ip.addr == x.x.x.x and tcp.port == 25
```

Filter by URL:

```
http.host == "hostname"
```

Filter by SYN flag:

```
tcp.flags.syn == 1
tcp.flags.syn == 1 and tcp.flags.ack == 0
```

Analyze Sample PCAP
Report back the following samples (HTTP, DNS, FTP, SMTP)
SMTP:
~~~
Who is the client?
10.10.1.4
Who is the server?
74.53.140.153
What is the subject of the mail?
SMTP
Who are the recipients?
raj_deo12002in@yahoo.co.in
What is the body of the email?
We do not authorize the use of this system to transport unsolicited

FTP:
````
Who is the client?
→ 10.10.1.4
Who is the server?
74.53.140.153
What is the ftp directory?
\r\n
Is any data getting exchanged?
~~~

DNS:
````
Who is the client?
192.168.170.8
Who is the server?
192.168.170.20
What is the dns request?
google.com TXT

HTTP:
`````
Who is the client?
145.254.160.237
Who is the server?
65.208.228.223
What is the web server version/type?
HTTP/1.1
What page is getting requested?
download.html

SMTP:
client:10.10.1.4
server:74.53.140.153
subject:SMTP
recipent:raj_deol2002in@yahoo.co.in
Body: Hello

I send u smtp pcap file
Find the attachment
GPS

DNS:
client:192.168.170.8
server:192.168.170.20
query: google.com, type:TXT, class IN

AND MANY MORE RECORDS FOR DNS QUERIES

HTTP:
client:145.245.160.237
server:65.208.228.223
web server version/type: Apache
page requested: www.ethereal.com/download.html

FTP:
client:81.131.67.131
server:192.88.99.1
ftp directory: /
data exchanged: no

TASK:
``````

1. Exploring Network Traces:
````````````````````````````

NOTE: Security analysts and attackers both frequently study network traffic to search for vulnerabilities and to characterize network behavior.

In this task, you will search for specific vulnerable behaviors and extract relevant details using the Wireshark network analyzer.

Analyse "data-task2.pcap" and Provide concise answers to the following questions. Each response should require at most 2â€“3 sentences.

Q1. Multiple hosts sent packets on the local network. What are their MAC and IP addresses? [easy]

Q2. What type of network does this appear to be (e.g., a large corporation, an ISP backbone, etc.)? Point to evidence from the trace that supports this. [intermediate]

Q3. One of the clients connects to an FTP server during the trace: [intermediate]

a. What is the DNS hostname of the server it connects to?
b. Is the connection using Active or Passive FTP?
c. Based on the packet capture, whatâ€™s one major vulnerability of the FTP protocol?
d. Name at least two network protocols that can be used in place of FTP to provide secure file transfer.

Q4. The trace shows that at least one of the clients makes HTTPS connections to sites other than Facebook. Pick one of these connections and answer the following: [intermediate]

a. What is the domain name of the site the client is connecting to?

b. Is there any way the HTTPS server can protect against the leak of information in (a)?

Q5. One of the clients makes a number of requests to Facebook: [easy]

a. Even though logins are processed over HTTPS, what is insecure about the way the browser is authenticated to Facebook?

b. How would this let an attacker impersonate the user on Facebook?

c. How can users protect themselves against this type of attack?

d. What did the user do while on the Facebook site?

tcpdump is a most powerful and widely used command-line packets sniffer or package analyzer tool which is used to capture or filter TCP/IP packets that are received or transferred over a network on a specific interface.

How to Install tcpdump in Linux
~~~

```
# apt install tcpdump  [On Debian, Ubuntu]
# yum install tcpdump  [On RHEL/CentOS/Fedora]
```

1. Capture Packets from Specific Interface
~~~

```
# tcpdump -i ens33
```

2. Capture Only N Number of Packets
~~~

```
# tcpdump -c 5 -i ens33
```

3. Print Captured Packets in ASCII
`````````````````````````````````````

```
# tcpdump -A -i ens33
```
~~~

4. Display Available Interfaces

```
# tcpdump -D
```

5. Display Captured Packets in HEX and ASCII

```
# tcpdump -XX -i ens33
```

6. Capture and Save Packets in a File

```
# tcpdump -w 0001.pcap -i ens33
```

7. Read Captured Packets File

```
# tcpdump -r 0001.pcap
```

8. Capture IP Address Packets

~~~
# tcpdump -n -i ens33
~~~

9. Capture only TCP Packets
```````````````````````````

~~~
# tcpdump -i ens33 tcp
~~~

10. Capture Packet from Specific Port
````````````````````````````````````

~~~
# tcpdump -i ens33 port 22
~~~

11. Capture Packets from source IP

```
# tcpdump -i ens33 src x.x.x.x
```

12. Capture Packets from destination IP

```
# tcpdump -i eth0 dst x.x.x.x
```

13. Capture all packets in any interface

```
# tcpdump -i any
```

14. Capture specific host

```
# tcpdump -i ens33 host x.x.x.x
```

15. Complex expressions
~~~

```
# tcpdump -i lo src 127.0.0.1 and port 80


# tcpdump -i lo "port 80 and (src x.x.x.x or src x.x.x.x)"
```

https://overthewire.org/wargames/bandit/bandit1.html
~~~

Nginx Simple Web Server:[Dedicated Hosting]

RedHat/CentOS:

```
# yum install nginx -y
```

Debian:

```
# apt update
# apt install nginx -y

# systemctl start nginx
# systemctl enable nginx
```

Default DocumentRoot for RedHat/CentOS:
/usr/share/nginx/html

Default DocumentRoot for Debian:
/var/www/html

On centOS:

~~~
``````````
# cd /usr/share/nginx/html
# cat > index.html
<html>
        <body>
                <h1> Basic Nginx Web Server </h1>
        </body>
</html>
^D
~~~

On Debian:

~~~
``````````
# cd /var/www/html
# cat > index.html
<html>
~~~

~~~
        <body>
                <h1> Basic Nginx Web Server </h1>
        </body>
</html>
^D


# systemctl restart nginx
~~~

To check:

~~~
``````````
# curl http://x.x.x.x:80
<html>
        <body>
                <h1> Basic Nginx Web Server </h1>
        </body>
</html>
~~~

Virtual DNS:

~~~
````````````
# vim /etc/named.conf

# abc.com Forward zone
zone "abc.com" IN {
        type master;
~~~

```
        file "for.abc.com";
};

:wq




# cd /var/named
# cp -av for.hpcsa.com for.abc.com
# vim for.abc.com
$TTL 1D
@       IN SOA master.hpcsa.com. dheeraj@gmail.com. (
                                        0       ; serial
                                        1D      ; refresh
                                        1H      ; retry
                                        1W      ; expire
                                        3H )    ; minimum
        IN      NS      master.hpcsa.com.
abc.com.        IN      A       192.168.82.22
www.abc.com.    IN      A       192.168.82.22

# systemctl restart named
# host abc.com
# host www.abc.com
```

Shared/Virtual Hosting using Nginx:
``````````````````````````````````````
``````````````````````````````````````

0) Enable virtual hosting in nginx.conf:

~~~
````````````````````````````````````````

# mkdir /etc/nginx/{sites-available, sites-enabled}
# vim /etc/nginx/nginx.conf

http {
        ...already_provided_conf....
        include /etc/nginx/conf.d/*.conf
        include /etc/nginx/sites-enabled/*.conf  <<<<<<<<<<<
Add this
        server {
                ...already_provided_conf....
                }
}
:wq
~~~

1) Setting Up New Document Root Directories & sample HTML pages

~~~
``````````````````````````````````````````````````````````````

# mkdir /usr/share/nginx/html <<<< Main DNS
(hpcsa.com)
# cat > /usr/share/nginx/html/index.html
<h1> Hpcsa main server </h1>
# mkdir /usr/share/nginx/html/abc.com <<<< Virtual DNS
(abc.com)
# cat > /usr/share/nginx/html/abc.com/index.html
<h1> ABC vhost main server </h1>
~~~

2) Creating the Server config for vhosts:
`````````````````````````````````````````

a) hpcsa.conf [Creating the MAIN Server config]
```````````````````````````````````````````````

```
# cat /etc/nginx/sites-available/hpcsa.conf
server {
        listen 80 default_server;
        root /usr/share/nginx/html;

        index index.html;
        server_name hpcsa.com www.hpcsa.com;
        access_log /var/log/nginx/hpcsa.com_access_log;
        error_log /var/log/nginx/hpcsa.com_error_log;
}
```

NOTE: Only one of our server blocks on the server can have the default_server option enabled.

a) abc.conf [Creating the vhost Server config]
~~~

```
# cat /etc/nginx/sites-available/abc.conf
server {
        listen 80;
        root /usr/share/nginx/html/abc.com;

        index index.html;
        server_name abc.com www.abc.com;
```
~~~

```
        access_log /var/log/nginx/abc.com_access_log;
        error_log /var/log/nginx/abc.com_error_log;
}
```

3) Enabling your Server configs and Restart Nginx:
~~~

```
# ln -s /etc/nginx/sites-available/hpcsa.conf
/etc/nginx/sites-enabled/hpcsa.conf
# ln -s /etc/nginx/sites-available/abc.conf
/etc/nginx/sites-enabled/abc.conf
```

4) restart nginx service:
````````````````````````

```
# systemctl restart nginx
```

5) Visit:
``````````

http://hpcsa.com
http://abc.com
~~~

Nginx configuration with https

With SSL/TLS:
`````````````

~~~
# mkdir /etc/nginx/certs
# cd /etc/nginx/certs/
~~~

Generate ROOT CA Certificate X.509 with OpenSSL:
```````````````````````````````````````````````

1) Generate private & public certificate
````````````````````````````````````````

~~~
# openssl req -newkey rsa:2048 -nodes -keyout key.pem
-x509 -days 365 -out certificate.pem
~~~

NOTE: writing new private key to 'key.pem'

2) To review the certificate
```````````````````````````

~~~
# openssl x509 -text -noout -in certificate.pem

 Issuer: C = AU, ST = Some-State, O = Internet Widgits
Pty Ltd
        Validity
            Not Before: Mar 14 16:28:14 2022 GMT
            Not After : Mar 14 16:28:14 2023 GMT
~~~

~~~
        Subject: C = AU, ST = Some-State, O = Internet
Widgits Pty Ltd
~~~

+++++++++++++++++++

4) Configure nginx for SSL:
``````````````````````````

~~~
# vim /etc/nginx/sites-available/test.conf

server {
        listen 443;
       ssl on;
       ssl_certificate /etc/nginx/certs/certificate.pem
       ssl_certificate_key /etc/nginx/certs/key.pem

        root /var/www/test.com/html;
        index index.html index.htm index.nginx-debian.html;

        server_name test.com www.test.com;
       access_log /var/log/nginx/test.com.access
       error_log /var/log/nginx/test.com.errors
        location / {
                try_files $uri $uri/ =404;
        }
}
~~~
``````````````````````````

```
:wq

# nginx -t
# systemctl restart nginx

# curl -ks https://test.com
```

squid
-------\
Squid is caching and forwarding HTTP web proxy. Which is used to speed up a web server by caching repeated requests, caching web, DNS and filtering traffic.

```
# yum install squid -y
# apt install squid -y


# systemctl restart squid
# systemctl enable squid
```

To check:
~~~

```
# squid -k check | echo $?
```
~~~

Port : 3128

```
# firewall-cmd --zone=public --add-service=squid
--permanent
# firewall-cmd --reload
```

RULE:
`````
http_access: Allows HTTP clients (browsers) to access the HTTP port. This is the primary access control list.

http_reply_access: Allows HTTP clients (browsers) to receive the reply to their request. This further restricts permissions given by http_access.
`````

```
# squid -v
```

Squid web cach dir: /var/spool/squid

```
# vim /etc/squid/squid.conf
```

Test the syntax of squid.conf:

```
# squid -k parse
```

```
# ls /var/log/squid
# ls /var/spool/squid
```

```
# systemctl restart squid
```

OR

```
# systemctl reload squid
```

OR

We can reload squid configuration file without restarting service:

```
# squid -k reconfigure
```

PORT : 3128
------\

+++Allow local network

```
# vim /etc/squid/squid.conf

acl internet_allow src 192.168.206.121/32
http_access allow internet_allow
```

OR

```
acl internet_allow src 192.168.206.0/24
http_access allow internet_allow
```

++++ Block IP

~~~
acl banned1 src 172.18.90.100-109
http_access deny banned1
http_reply_access allow all
~~~

We will block domains such as facebook.com and youtube.com

~~~
# vim /etc/squid/squid.conf
~~~
add lines:
~~~
``````````
acl blocksite1 dstdomain .facebook.com .youtube.com
http_access deny blocksite1
~~~

+++++ To restrict by a part of the URI, do:

~~~
acl banned_reddit url_regex ˆhttp://.*reddit.com/.*$
http_access deny banned_reddit
~~~

+++++ If you want to allow/block specific website then make the entry of blockedsites

```
# cat > /etc/squid/blockedsites.squid
.tesla.com
.gmail.com
.cdac.in
```

```
#vim /etc/squid/squid.conf
```

```
acl blocksites dstdomain "/etc/squid/blockedsites.squid"
http_access deny blocksites
```

OR

```
acl blocksites url_regex "/etc/squid/blockedsites.squid"
http_access deny blocksites
```

```
curl -x squid-proxy-server-IP:3128 http://google.com -I
HTTP/1.1 200 OK
Cache-Control: max-age=604800
Content-Type: text/html
Date: Fri, 22 Jun 2016 12:00:00 GMT
Expires: Fri, 29 Jun 2016 12:00:00 GMT
Last-Modified: Fri, 09 Aug 2013 23:54:35 GMT
Server: ECS (iad/182A)
Vary: Accept-Encoding
X-Cache: HIT
x-ec-custom-error: 1
Content-Length: 1270
X-Cache: MISS from moon*****************************
<<<<<<<< [1(a)]
X-Cache-Lookup: MISS from moon:3128*****************
<<<<<<<< [1(b)]
Via: 1.1 moon (squid/3.5.16)************************
<<<<<<<< [2]
Connection: close

<!doctype html>
<html>
<head>
   <title>Example domain</title>
```

```
[...]
</body>
</html>
```

[1(a)] : The value of the header X-Cache shows that the requested document was not in the Squid cache (MISS) of the computer moon.

[1(b)] : The example above contains two X-Cache lines. The second one X-Cache-Lookup is produced by the internal caching software of the originating Web server.

[2] : The value of the header Via shows the HTTP version, the name of the computer, and the version of Squid in use.

X-Forwarded-For Header:

````````````````````````
````````````````````````

=> If set to "on", squid will append your client's IP address in HTTP req it forwards like:

```
X-Forwarded-For: 192.1.1.1
```

=> If set to "off", it will appear as

```
X-Forwarded-For: unknown
```

=> If set to "transparent", Squid will not alter the "X-Forwarded-For" header in any way.

=> If set to "delete", Squid will delete the extire "X-Forwarded-For" header.

So modify the entry:

forwarded_for delete

---

configure:
~~~

```
# vim /etc/squid/squid.conf
# go to end of the file and add those entries
forwarded_for delete
via off
```
~~~

```
# squid -k reconfigure
```

squid
-------\
Squid is caching and forwarding HTTP web proxy. Which is used to speed up a web server by caching repeated requests, caching web, DNS and filtering traffic.

```
# yum install squid -y
# apt install squid -y
```

```
# systemctl restart squid
# systemctl enable squid
```

To check:
~~~

```
# squid -k check | echo $?
```

Port : 3128

```
# firewall-cmd --zone=public --add-service=squid
--permanent
# firewall-cmd --reload
```

RULE:
~~~

~~~

http_access: Allows HTTP clients (browsers) to access the HTTP port. This is the primary access control list.

http_reply_access: Allows HTTP clients (browsers) to receive the reply to their request. This further restricts permissions given by http_access.

```
# squid -v
```

Squid web cach dir: /var/spool/squid

```
# vim /etc/squid/squid.conf
```

Test the syntax of squid.conf:

```
# squid -k parse
```

```
# ls /var/log/squid
```
~~~

```
# ls /var/spool/squid
```

```
# systemctl restart squid
```
OR
```
# systemctl reload squid
```
OR

We can reload squid configuration file without restarting service:

```
# squid -k reconfigure
```

PORT : 3128

------\

+++Allow local network

```
# vim /etc/squid/squid.conf

acl internet_allow src 192.168.206.121/32
http_access allow internet_allow
```

OR

```
acl internet_allow src 192.168.206.0/24
http_access allow internet_allow
```

++++ Block IP

```
acl banned1 src 172.18.90.100-109
http_access deny banned1
http_reply_access allow all
```

-------\

Go to browser and set proxy manually

OR

```
curl -x squid-proxy-server-IP:3128 http://google.com -I
curl -O -L "https://www.redhat.com/index.html" -x
"proxy.example.com:3128"
```

OR

NOTE: squidclient, a command-line tool that outputs the response to a Web request, similar to wget or curl

```
# squidclient http://www.example.org

HTTP/1.1 200 OK
Cache-Control: max-age=604800
Content-Type: text/html
Date: Fri, 22 Jun 2016 12:00:00 GMT
Expires: Fri, 29 Jun 2016 12:00:00 GMT
Last-Modified: Fri, 09 Aug 2013 23:54:35 GMT
Server: ECS (iad/182A)
Vary: Accept-Encoding
X-Cache: HIT
x-ec-custom-error: 1
Content-Length: 1270
X-Cache: MISS from moon*****************************
<<<<<<<<< [1(a)]
```

```
X-Cache-Lookup: MISS from moon:3128*****************
<<<<<<<<< [1(b)]
Via: 1.1 moon (squid/3.5.16)************************
<<<<<<<<< [2]
Connection: close

<!doctype html>
<html>
<head>
    <title>Example domain</title>
[...]
</body>
</html>
```

[1(a)] : The value of the header X-Cache shows that the requested document was not in the Squid cache (MISS) of the computer moon.

[1(b)] : The example above contains two X-Cache lines. The second one X-Cache-Lookup is produced by the internal caching software of the originating Web server.

[2] : The value of the header Via shows the HTTP version, the name of the computer, and the version of Squid in use.

X-Forwarded-For Header:
~~~

=> If set to "on", squid will append your client's IP address in HTTP req it forwards like:

```
X-Forwarded-For: 192.1.1.1
```

=> If set to "off", it will appear as

```
X-Forwarded-For: unknown
```

=> If set to "transparent", Squid will not alter the "X-Forwarded-For" header in any way.

=> If set to "delete", Squid will delete the extire "X-Forwarded-For" header.

So modify the entry:
~~~

```
forwarded_for delete
```

And add these after "forwarded_for delete" in the squid.conf file:

```
request_header_access Allow allow all
request_header_access Authorization allow all
request_header_access WWW-Authenticate allow all
request_header_access Proxy-Authorization allow all
request_header_access Proxy-Authenticate allow all
request_header_access Cache-Control allow all
request_header_access Content-Encoding allow all
request_header_access Content-Length allow all
request_header_access Content-Type allow all
request_header_access Date allow all
request_header_access Expires allow all
request_header_access Host allow all
request_header_access If-Modified-Since allow all
request_header_access Last-Modified allow all
request_header_access Location allow all
request_header_access Pragma allow all
request_header_access Accept allow all
request_header_access Accept-Charset allow all
request_header_access Accept-Encoding allow all
request_header_access Accept-Language allow all
request_header_access Content-Language allow all
request_header_access Mime-Version allow all
request_header_access Retry-After allow all
request_header_access Title allow all
request_header_access Connection allow all
request_header_access Proxy-Connection allow all
request_header_access User-Agent allow all
request_header_access Cookie allow all
request_header_access All deny all
```

____________________

OTHER IMP HEADERS: You can block/disallow them if you want

```
follow_x_forwarded_for deny all
forwarded_for delete [covered previously]
via off
```

_____________________

To change squid proxy hostname:
~~~

```
# vim /etc/squid/squid.conf

visible_hostname anything

:wq
```
~~~

-------\
+++++ check logs

```
# tail -f /var/log/squid/access.log
```
-------\

---

---

Enable cache dir :
\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`

# how much disk cache do you want. It is 6400 MB in the following example, change it as per
# your needs. Make sure you have that much disk space free.

```
cache_dir ufs /var/spool/squid 6400 16 256
```

- Squid creates 16 level-1 sub-directories in the /var/spool/squid/ directory.
- Squid creates 256 sub-directories in each level-1 directory.

# how much memory cache do you want? depends on how much memory you have on the machine
cache_mem 200 MB

```
# MAX OBJ SIZE
maximum_object_size 500 MB
maximum_object_size_in_memory 4 MB

# Say "off" if you want the query string to appear in the squid logs.
strip_query_terms off
```

You can put any number of refresh_pattern lines in the configuration file:

SYNATX: refresh_pattern [-i] regexp min percent max [options]

-i : case-insensitive
min : the time (in minutes) an object without an explicit expiry time should be considered fresh.
max : upper limit on how long objects without an explicit expiry time will be considered fresh.

eg:

```
refresh_pattern -i \.jpg 30 20% 4320

refresh_pattern -i \.(css|js|png) 30 20% 4320;
```

#Facebook Pages
refresh_pattern -i \.facebook.com.*\.(jpg|png|gif) 80 20% 4320;

how do I configure Squid not to cache a specific server?
~~~

acl someserver dstdomain .someserver.com
cache deny someserver

_____________________________________________________________________

Anonymous browsing
``````````````````

We will block domains such as facebook.com and youtube.com

# vim /etc/squid/squid.conf
add lines:
``````````
~~~

```
acl blocksite1 dstdomain .facebook.com .youtube.com
http_access deny blocksite1
```

+++++ To restrict by a part of the URI, do:

```
acl banned_reddit url_regex ˆhttp://.*reddit.com/.*$
http_access deny banned_reddit
```

+++++ If you want to allow/block specific website then make the entry of blockedsites

```
# cat > /etc/squid/blockedsites.squid
.tesla.com
.gmail.com
.cdac.in
```

```
#vim /etc/squid/squid.conf
```

```
acl blocksites dstdomain "/etc/squid/blockedsites.squid"
```

```
http_access deny blocksites
```

OR

```
acl blocksites url_regex "/etc/squid/blockedsites.squid"
http_access deny blocksites
```

```
#systemctl restart squid.service
```

-------\

+++++ Block ports
~~~

```
# vim /etc/squid/squid.conf

acl block_port port 80
http_access deny block_port

acl Safe_ports port 80 21 443 563 70 210 1025-65535
http_access deny !Safe_ports
```
~~~

+++++ Block specific words

```
# vim /etc/squid/ban_keywords.txt
gambling
spyware
bad
:wq

# vim /etc/squid/squid.conf
acl bad_keywords url_regex "/etc/squid/ban_keywords.txt"
http_access deny bad_keywords
```

------/

++++++ You can also block files

```
# vim /etc/squid/blockfiles.squid
```

```
\.torrent.*$
\.mp3.*$
\.m3u8.*$
\.mp4.*$

# vim /etc/squid/squid.conf
acl blockfiles urlpath_regex "/etc/squid/blockfiles.squid"
http_access deny blockfiles
```

-------\

++++++ Set working hours

```
#vim /etc/squid/squid.conf


acl working_hours time 10:00-17:00
http_access deny working_hours
```

------------

*scapy

Network Operations with Scapy
------------------------------

Scapy is a powerful interactive packet manipulation/crafting program/framework.

+ Install it

```
# apt install scapy -y
OR
# yum install epel-release -y
# yum install scapy -y
```

+ Interective shell:

```
# scapy
```

ICMP example:
~~~
ip.src eq 1.1.1.1 && ip.dst eq 8.8.8.8

```
>>> x = IP()
>>> y = ICMP()
>>> x.show()
>>> y.show()

>>> x.src="x.x.x.x"
```
~~~

```
>>> x.dst="x.x.x.x"

>>> send(x/y/"LOLOLOLOL")
or
>>> send(x/y, count=10)
or
>>> send(x/y, loop=1)
```

+ CHECK IN WIRESHARK

---

Send TCP Packets:

----------

```
>>> a = IP()
>>> b = TCP()
>>> a.src="x.x.x.x"
>>> a.dst="196.1.113.45"
>>> b.sport=53
>>> b.dport=80

>>> send((a/b), count=10)
```

IDS/IPS (SPI/DPI)

``````````````````

What is a Intrusion Detection System?
``````````````````

~~~

- An intrusion detection system (IDS) is a device or software application that monitors a network for malicious activity or policy violations.

- Some IDSâ€™s are capable of responding to detected intrusion upon discovery.

IDS Detection Types
````````````````````

+ Network intrusion detection systems (NIDS)
~~~

      - A system/hardware that analyzes incoming network traffic.

+ Host-based intrusion detection systems (HIDS)
```````````````````````````````````````````````

      - A system that monitors important operating system files.

+ Protocol-based Intrusion Detection System (PIDS):
```

````````````````````````````````````````````````

- It is trying to secure the web server by regularly monitoring the HTTPS protocol stream and accept the related HTTP protocol.

+ Application Protocol-based Intrusion Detection System (APIDS):
`````````````````````````````````````````````````````````````

- It identifies the intrusions by monitoring and interpreting the communication on application-specific protocols.

+ Hybrid Intrusion Detection System :
~~~

- In the hybrid intrusion detection system, host agent or system data is combined with network information to develop a complete view of the network system.

Detection Method of IDS:
````````````````````````

+ Signature-based:
`````````````````

- looking for specific patterns, such as byte sequences in network traffic
~~~

- Although signature-based IDS can easily detect known attacks
- it is impossible to detect new attacks, for which no pattern is available.

+ Anomaly-based:
````````````````

- a newer technology designed to detect and adapt to unknown attacks
- detection method uses machine learning to create a defined model of trustworthy activity, and then compare new behavior against this trust model.

-=-=-=-=-=-=

What is an Intrusion Prevention System?
````````````````````````````````````````

- An intrusion prevention system (IPS) is an network security device used to monitor and respond to potential threats.

- The main functions of an IPS are to identify suspicious activity, log relevant information, attempt to block the activity, and finally to report it.

Detection mechanisms:
~~~

- Address matching
- HTTP string and substring matching
- Generic pattern matching
- TCP connection analysis
- Packet anomaly detection
- Traffic anomaly detection
- TCP/UDP port matching

IPS Classifications
``````````````````

Network-based intrusion prevention system (NIPS):
~~~

- Analyzes protocol activity across the entire network, looking for any untrustworthy traffic.

Wireless intrusion prevention system (WIPS):
``````````````````````````````````````````
``````````````````````````````````````````

- Analyzes network protocol activity across the entire wireless network, looking for any untrustworthy traffic.

Host-based intrusion prevention system (HIPS):
~~~

- that follows a single host for malicious activity, and analyzes events occurring within said host.

IPS Detection Methods
`````````````````````

Signature-based detection:
``````````````````````````

- Signature-based IDS monitors packets in the network and compares with predetermined attack patterns, known as â€œsignaturesâ€ .

Statistical anomaly-based detection:
````````````````````````````````````

- An anomaly-based IPS will monitor network traffic and compare it to expected traffic patterns.

- The baseline will identify what is "normal" for that network â€“ what sort of packets generally through the network and what protocols are used.

Debian:
~~~

```````

# apt install snort -y

Update shared libs: .so
```````````````````

# ldconfig

2. Info:
````````

# cat /etc/passwd | grep snort

snort:x:130:138:Snort IDS:/var/log/snort:/usr/sbin/nologin

3. Config:
```````````

# ls /etc/snort

Log:
````

# ls /var/log/snort

main configuration file: /etc/snort/snort.conf

NOTE: take backup of original data.

# cp -av /etc/snort/snort.conf
/etc/snort/snort.conf_orig_$(date -I)
````

4. Analyse config file:

~~~
``````````````````````

# vim /etc/snort/snort.conf
~~~

Comment this line:

```
#ipvar HOME_NET any
```

And add new line

```
ipvar HOME_NET 192.168.206.0/24
```

And comment all the rules accept local_rules.

```
:wq
```

Test the snort rule/config:

~~~
``````````````````````````

# snort -T -i enp0s3 -c /etc/snort/snort.conf
~~~

-T : Test
-i : interface
-c : config

```
Initializing rule chains...
0 Snort rules read
        0 detection rules
        0 decoder rules
```

```
        0 preprocessor rules
0 Option Chains linked into 0 Chain Headers
```

Snort as a daemon
~~~

```
# snort -D
```

Time to configure own rules:
~~~

** Snort rules are devided into two logical sections, the rule header and the rule options. **

1. Rule Header: The rule header contains the rule's action, protocol, source and destination IP address and netmask, the source and destination port information.

2. Rule Options: The rule option section contains alert message and information on which part of packet should be inspected to determine if rule action should be taken.

Rule Syntax:
````````````
````````````

<action> <proto> <src_ip> <src_port> -> <dst_ip>
<dst_port> (msg:"<msg>"; sid:"<signature>"; rev:1;)

For custom rule, signature ID starts from 100001, before 100000 reserved for snort.

Rule Actions:
\`\`\`\`\`\`\`\`\`\`\`\`\`

| | |
|---|---|
| alert | : Alert and log the packet |
| log | : log the packet |
| pass | : ignore the packet |
| drop | : block and log the packet |
| reject | : block the packet, log it and send TCP reset |
| sdrop | : block the packet and do not log it |

++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++

___________________________
a) ANY packet detection:
\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`

# vim /etc/snort/rules/local.rules

alert ip any any -> any any (msg: "IP Packet detected"; sid: 10000; rev:1;)

```
# snort -A console -q -i eth0 -c /etc/snort/snort.conf

02/13-12:39:48.921203  [**] [1:10000:1] IP Packet
detected [**] [Priority: 0] {UDP} 143.244.134.227:123 ->
192.168.86.128:48971
```

---

b) PING Detection rule:
~~~

```
# vim /etc/snort/rules/local.rules

alert icmp any any -> $HOME_NET any (msg: "LOL
ICMP"; sid: 1000001; rev: 1;)

# snort -T -i eth0 -c /etc/snort/snort.conf  [ TEST ]
# snort -A console -q -i eth0 -c /etc/snort/snort.conf [
Debug ]

02/13-12:27:06.070450  [**] [1:1000001:1] LOL ICMP [**]
[Priority: 0] {ICMP} 192.168.86.1 -> 192.168.86.128
02/13-12:27:06.070470  [**] [1:1000001:1] LOL ICMP [**]
[Priority: 0] {ICMP} 192.168.86.128 -> 192.168.86.1
```

---

c) FTP Connection Detection rule:
~~~

```
# vim /etc/snort/rules/local.rules

alert tcp any any -> $HOME_NET 21 (msg: "FTP
Connection"; sid: 1000002; rev: 1;)

# snort -A console -q -i eth0 -c /etc/snort/snort.conf

02/13-12:32:43.635836  [**] [1:1000002:1] FTP
Connection [**] [Priority: 0] {TCP} 192.168.86.1:37519 ->
192.168.86.128:21
02/13-12:32:43.636304  [**] [1:1000002:1] FTP
Connection [**] [Priority: 0] {TCP} 192.168.86.1:37519 ->
192.168.86.128:21
02/13-12:32:43.692072  [**] [1:1000002:1] FTP
Connection [**] [Priority: 0] {TCP} 192.168.86.1:37519 ->
192.168.86.128:21
02/13-12:32:45.712857  [**] [1:1000002:1] FTP
Connection [**] [Priority: 0] {TCP} 192.168.86.1:37519 ->
192.168.86.128:21
02/13-12:32:45.714615  [**] [1:1000002:1] FTP
Connection [**] [Priority: 0] {TCP} 192.168.86.1:37519 ->
192.168.86.128:21
```

---

d) FLAG detection rule:

``````````````````````
``````````````````````

F - FIN
S - SYN
R - RST
P - PSH
A - ACK
U - URG
0 (zero) - NO FLAG

There are also logical operators:
~~~

+ - ALL flag, match on all specified flags plus any others
* - ANY flag, match on any of the specified flags
! - NOT flag, match if the specified flags aren't set in the packet

```
# vim /etc/snort/rules/local.rules
```

```
alert tcp any any -> $HOME_NET any (flags: S; msg:
"SYN Packet"; sid: 1000003; rev: 1;)
```

```
# snort -A console -q -i eth0 -c /etc/snort/snort.conf
```

SNORT Exercises
---------------
~~~

Exercise 1 : Write a rule to check Nmap SYN scan [Half Open Scan] on your server from external network

Exercise 2 : Write a rule to check any external network access to the webserver /admin pages

Exercise 3 : Write a rule to check FTP failed login attempt

Exercise 4 : Write a rule to detect HTTP packet.

h) Classtype:
`````````````

The classtype keyword is used to categorize a rule as detecting an attack. Snort provides a default set of attack classes that are used by the default set of rules it provides.

Syntax: classtype:<class name>;

Attack classifications defined by Snort reside in the `classification.config` file.
`````````````

______________
i) priority:
~~~
The priority tag assigns a severity level to rules. A classtype rule assigns a default priority in `classification.config` file. A priority of 1 (high) is the most severe and 4 (very low) is the least severe.

Syntax: priority:<priority integer>;

```
alert tcp any any -> any 80 (msg:"EXPLOIT";
content:"/internal-admin"; classtype:attempted-admin;
priority:10 );
```

=============================================
==========
SNORT - Intrusion Prevention System (INLINE MODE)

Snort Modes

1) Packet Capture Mode
   ```
   # snort -i eth0
   ```
2) IDS Mode
3) Inline Mode
~~~

1) Install SNORT:
``````````````````

CentOS -7 :
````````
Manual:

~~~
# yum install -y zlib-devel libpcap-devel pcre-devel
libdnet-devel openssl-devel libnghttp2-devel luajit-devel
gcc flex flex-devel bison

# wget
https://www.snort.org/downloads/snort/daq-2.0.7.tar.gz

# tar -xzvf daq-2.0.7.tar.gz
# cd daq-2.0.7
# ./configure && make && make install


# wget
https://www.snort.org/downloads/snort/snort-2.9.20.tar.gz

# tar -xzvf snort-2.9.20.tar.gz
~~~

~~~
# cd snort-2.9.20/

# ./configure --enable-sourcefire --disable-open-appid

# make

# make install
~~~

Update lib:

~~~
````````````
# ldconfig
# ln -s /usr/local/bin/snort /usr/sbin/snort
~~~

Other stuff:

~~~
````````````
sudo groupadd snort
sudo useradd snort -r -s /sbin/nologin -c SNORT_IDS -g
snort

sudo mkdir -p /etc/snort/rules
sudo mkdir /var/log/snort
sudo mkdir /usr/local/lib/snort_dynamicrules

sudo chmod -R 5775 /etc/snort
sudo chmod -R 5775 /var/log/snort
sudo chmod -R 5775 /usr/local/lib/snort_dynamicrules
~~~

```
sudo chown -R snort:snort /etc/snort
sudo chown -R snort:snort /var/log/snort
sudo chown -R snort:snort
/usr/local/lib/snort_dynamicrules

sudo touch /etc/snort/rules/white_list.rules
sudo touch /etc/snort/rules/black_list.rules
sudo touch /etc/snort/rules/local.rules


sudo cp -av ~/snort-2.9.20/etc/*.conf* /etc/snort
sudo cp -av ~/snort-2.9.20/etc/*.map /etc/snort


wget https://www.snort.org/rules/community -O
~/community.tar.gz
tar -xvf ~/community.tar.gz -C ~/
cp ~/community-rules/* /etc/snort/rules


sed -i 's/include $RULE_PATH/#include $RULE_PATH/'
/etc/snort/snort.conf
-------
# vim /etc/snort/snort.conf
```

change the default [~/root] path to [/etc/snort]

from:
`````
var RULE_PATH ../rules
var SO_RULE_PATH ../so_rules
var PREPROC_RULE_PATH ../preproc_rules
var WHITE_LIST_PATH ../rules
var BLACK_LIST_PATH ../rules
`````

to:
```
var RULE_PATH /etc/snort/rules
var SO_RULE_PATH /etc/snort/so_rules
var PREPROC_RULE_PATH /etc/snort/preproc_rules
var WHITE_LIST_PATH /etc/snort/rules
var BLACK_LIST_PATH /etc/snort/rules
```

Debian:
```````
# apt install snort -y
```````

Update shared libs: .so
```````````````````

~~~
# ldconfig
~~~

2. Info:
````````

~~~
# cat /etc/passwd | grep snort

snort:x:130:138:Snort IDS:/var/log/snort:/usr/sbin/nologin
~~~

3. Config:
```````````

~~~
# ls /etc/snort
~~~

Log:
````

~~~
# ls /var/log/snort
~~~

main configuration file: /etc/snort/snort.conf

NOTE: take backup of original data.

~~~
# cp -av /etc/snort/snort.conf
/etc/snort/snort.conf_orig_$(date -I)
~~~

4. Analyse config file:
``````````````````````
``````````````````````

```
# vim /etc/snort/snort.conf
```

Comment this line:

```
#ipvar HOME_NET any
```

And add new line

```
ipvar HOME_NET 192.168.206.0/24
```

And comment all the rules accept local_rules.

```
:wq
```

Test the snort rule/config:

~~~
``````````````````````````
# snort -T -i enp0s3 -c /etc/snort/snort.conf
~~~

-T : Test
-i : interface
-c : config

```
Initializing rule chains...
0 Snort rules read
	0 detection rules
	0 decoder rules
	0 preprocessor rules
0 Option Chains linked into 0 Chain Headers
```

Snort as a daemon
``````````````````

# snort -D

Time to configure own rules:
`````````````````````````````

** Snort rules are devided into two logical sections, the rule header and the rule options. **

1. Rule Header: The rule header contains the rule's action, protocol, source and destination IP address and netmask, the source and destination port information.

2. Rule Options: The rule option section contains alert message and information on which part of packet should be inspected to determine if rule action should be taken.

Rule Syntax:
````````````

<action> <proto> <src_ip> <src_port> -> <dst_ip> <dst_port> (msg:"<msg>"; sid:"<signature>"; rev:1;)
````````````

For custom rule, signature ID starts from 100001, before 100000 reserved for snort.

Rule Actions:
\`\`\`\`\`\`\`\`\`\`\`\`\`

| | |
|---|---|
| alert | : Alert and log the packet |
| log | : log the packet |
| pass | : ignore the packet |
| drop | : block and log the packet |
| reject | : block the packet, log it and send TCP reset |
| sdrop | : block the packet and do not log it |

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

_______________________________

a) ANY packet detection:
\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`\`

```
# vim /etc/snort/rules/local.rules

alert ip any any -> any any (msg: "IP Packet detected";
sid: 10000; rev:1;)

# snort -A console -q -i eth0 -c /etc/snort/snort.conf
```

```
02/13-12:39:48.921203  [**] [1:10000:1] IP Packet
detected [**] [Priority: 0] {UDP} 143.244.134.227:123 ->
192.168.86.128:48971
```

---

b) PING Detection rule:
~~~

```
# vim /etc/snort/rules/local.rules

alert icmp any any -> $HOME_NET any (msg: "LOL
ICMP"; sid: 1000001; rev: 1;)

# snort -T -i eth0 -c /etc/snort/snort.conf  [ TEST ]
# snort -A console -q -i eth0 -c /etc/snort/snort.conf [
Debug ]

02/13-12:27:06.070450  [**] [1:1000001:1] LOL ICMP [**]
[Priority: 0] {ICMP} 192.168.86.1 -> 192.168.86.128
02/13-12:27:06.070470  [**] [1:1000001:1] LOL ICMP [**]
[Priority: 0] {ICMP} 192.168.86.128 -> 192.168.86.1
```

---

c) FTP Connection Detection rule:
~~~

```
# vim /etc/snort/rules/local.rules
```

```
alert tcp any any -> $HOME_NET 21 (msg: "FTP
Connection"; sid: 1000002; rev: 1;)

# snort -A console -q -i eth0 -c /etc/snort/snort.conf

02/13-12:32:43.635836  [**] [1:1000002:1] FTP
Connection [**] [Priority: 0] {TCP} 192.168.86.1:37519 ->
192.168.86.128:21
02/13-12:32:43.636304  [**] [1:1000002:1] FTP
Connection [**] [Priority: 0] {TCP} 192.168.86.1:37519 ->
192.168.86.128:21
02/13-12:32:43.692072  [**] [1:1000002:1] FTP
Connection [**] [Priority: 0] {TCP} 192.168.86.1:37519 ->
192.168.86.128:21
02/13-12:32:45.712857  [**] [1:1000002:1] FTP
Connection [**] [Priority: 0] {TCP} 192.168.86.1:37519 ->
192.168.86.128:21
02/13-12:32:45.714615  [**] [1:1000002:1] FTP
Connection [**] [Priority: 0] {TCP} 192.168.86.1:37519 ->
192.168.86.128:21
```

---

d) FLAG detection rule:

````````````````````````

F - FIN
S - SYN
````````````````````````

R - RST
P - PSH
A - ACK
U - URG
0 (zero) - NO FLAG

There are also logical operators:
~~~

+ - ALL flag, match on all specified flags plus any others
* - ANY flag, match on any of the specified flags
! - NOT flag, match if the specified flags aren't set in the packet

```
# vim /etc/snort/rules/local.rules
```

```
alert tcp any any -> $HOME_NET any (flags: S; msg: "SYN Packet"; sid: 1000003; rev: 1;)
```

```
# snort -A console -q -i eth0 -c /etc/snort/snort.conf
```

NOTE: To check use 'scapy' IP()/TCP()

```
>>> a = IP()
>>> b = TCP()
>>> a.src = "x.x.x.x"
```
~~~

```
>>> a.dst = "x.x.x.x"
>>> b.flags = "S"
>>> send(a/b)
```

```
alert tcp any any -> 192.168.1.105 22 (msg: "NMAP TCP
Scan";sid:10000005; rev:2; )
alert tcp any any -> 192.168.1.105 22 (msg:"Nmap FIN
Scan"; flags:F; sid:1000006; rev:1; )
alert tcp any any -> 192.168.1.105 22 (msg:"Nmap XMAS
Scan"; flags:FPU; sid:1000006; rev:1; )
alert tcp any any -> 192.168.1.105 22 (msg:"Nmap NULL
Scan"; flags:0; sid:1000009; rev:1; )
```

---

e) Content Matching rule:

~~~
````````````````````````
~~~

```
# vim /etc/snort/rules/local.rules

alert tcp any any -> $HOME_NET 22 (msg: "SSH
Attempt"; content: "SSH"; sid: 1000003; rev: 1;)

# snort -A console -q -i eth0 -c /etc/snort/snort.conf

02/13-12:56:02.128231  [**] [1:1000003:1] SSH Attempt
[**] [Priority: 0] {TCP} 192.168.86.1:38256 ->
192.168.86.128:22
```

NOTE: To check use 'scapy' IP()/TCP()

```
>>> a = IP()
>>> b = TCP()
>>> a.src = "x.x.x.x"
>>> a.dst = "x.x.x.x"
>>> b.flags = "S"
>>> send(a/b/"SSH")
```

f) logging and storing ASCII:

~~~
````````````````````````````
~~~

```
# vim /etc/snort/rules/local.rules
```

```
alert tcp any any -> $HOME_NET 22 (content: "SSH";
msg: "SSH Attempt"; sid: 1000003; rev: 1;)
```

```
# snort -A console -q -i eth0 -c /etc/snort/snort.conf -K
ascii
```

```
02/13-12:56:02.128231  [**] [1:1000003:1] SSH Attempt
[**] [Priority: 0] {TCP} 192.168.86.1:38256 ->
192.168.86.128:22
```

```
# cd /var/log/snort/
```

=> One log file which is common:

```
# ls -l snort.log*
```

You can also open this file in wireshark.

=> If '-K ascii' used, then it will start logging for individual host:

```
# cd x.x.x.x;ls
TCP:38256-22
```

---

g) Variables:
`````````````You can also create new `var` for rules:

Syntax: var: <name> <value>

```
# vim /etc/snort/rules/local.rules

var MY_NET [192.168.1.0/24,10.1.1.0/24]
alert tcp any any -> $MY_NET any (flags: S; msg: "SYN
packet";)
```

---

h) Classtype:
`````````````

The classtype keyword is used to categorize a rule as detecting an attack. Snort provides a default set of attack classes that are used by the default set of rules it provides.

Syntax: classtype:<class name>;

Attack classifications defined by Snort reside in the `classification.config` file.

______________

i) priority:
````````````

The priority tag assigns a severity level to rules. A classtype rule assigns a default priority in `classification.config` file. A priority of 1 (high) is the most severe and 4 (very low) is the least severe.

Syntax: priority:<priority integer>;

alert tcp any any -> any 80 (msg:"EXPLOIT"; content:"/internal-admin"; classtype:attempted-admin; priority:10 );

______________________________________________________
____________________________________________

Bypass IDS:
````````````

````````````

[+] Detection Rule:
``````````````````

~~~
# vim /etc/snort/rules/local.rules

var HTTP_WEB_SERVERS [x.x.x.x, x.x.x.x]
var HTTP_PORTS [80, 443]
alert $EXTERNAL_NET any -> $HTTP_WEB_SERVERS
$HTTP_PORTS (msg:"LFI Attack detected"; content:"../";
sid:1000001; rev:1)

# snort -A console -q -i ens33 -c /etc/snort/snort.conf

http://domain.tld/index.php?file=../../../etc/passwd
~~~

[+] Possible bypass:
````````````````````

URL Encode:

~~~
. => %2E
/ => %2F

http://domain.tld/index.php?file=%2E%2E%2F%2E%2E%
2F%2E%2E%2Fetc%2Fpasswd ====> 200 ok
~~~
````````````````````

*** A useful tool to write a basic snort rule.
https://github.com/chrisjd20/Snorpy

```
# git clone https://github.com/chrisjd20/Snorpy.git
# cd Snorpy/
# docker build -t snorpy_app .
# docker run -p 8080:8080 -it --rm --name
snorpy_container snorpy_app
```

OR

```
# cd Snorpy
# apt install npm node -y
# npm install express
# node app.js
worker 2
worker 1
```

http://localhost:8080

Inline Mode [IPS]
~~~

IDS + IPS: IDPS

Download and Install DAQ (Data Acquisition):

```
# cd /etc/snort
# wget
https://www.snort.org/downloads/snort/daq-2.0.7.tar.gz
# tar -xvzf daq-2.0.7.tar.gz
```

DAQ Dep:
`````````

```
# apt install -y gcc libpcre3-dev zlib1g-dev libluajit-5.1-dev
libpcap-dev openssl libssl-dev libnghttp2-dev
libdumbnet-dev bison flex libdnet autoconf libtool make
build-essential
```

```
# cd daq-2.0.7
# ./configure && make && make install
```
~~~

DAQ [Data Acquisition library] - Inline Mode:[IPS]
~~~

Step 1
``````

```
# snort --daq-list
	Available DAQ modules:
	pcap(v3): the default mode, used for sniffer and IDS modes
	nfq(v7): inline on Linux netfilter
	ipfw(v3): used divert sockets with the pf & ipfw firewalls
	dump(v3): allow testing of online
	afpacket(v5): On linux using two bridged interface
```

```
# cp -av ~/daq-2.0.7 /etc/snort/
```

Step 2
``````

```
# vim /etc/snort/snort.conf
```

Uncomment DAQ config

```
# config daq: <type>
# config daq_dir: <dir>
```
~~~

# config daq_mode: <mode>
# config daq_var: <var>

to --->

 config daq: afpacket
 config daq_dir: /etc/snort/daq-2.0.7
 config daq_mode: inline
 config daq_var: buffer_size_mb=512

:wq

Step 3
``````
# vim /etc/snort/rules/local.rules

drop icmp $HOME_NET any -> any any (msg:"ICMP detect and drop"; sid:1000001; rev:1; classtype:icmp-event;)

:wq

# snort -T -i eth0:eth1 -Q -q -c /etc/snort/snort.conf      [ TEST ]
``````

```
# snort -A console -q -Q -i eth0:eth1 -c
/etc/snort/snort.conf
```

VPN

CentOS:
```````Add two interfaces in VM one NAT and another custom Host-Only
ens33 : 192.168.206.130/24 [ VPN Server Public IP ]
ens37 : 10.0.0.101/24

a) Create a internal web server:

```
# yum install httpd -y
```

b) Open config:

```
# vim /etc/httpd/conf/httpd.conf

Listen 10.0.0.101:80

:wq

# systemctl restart httpd
```

NOTE: Now web server in running on 10.0.0.100:80 only.
```````

---

---

https://github.com/angristan/openvpn-install
``````````````````

# wget
https://raw.githubusercontent.com/angristan/openvpn-insta
ll/master/openvpn-install.sh
# bash openvpn-install.sh

Welcome to the OpenVPN installer!
The git repository is available at:
https://github.com/angristan/openvpn-install

I need to ask you a few questions before starting the setup.
You can leave the default options and just press enter if you are ok with them.

I need to know the IPv4 address of the network interface you want OpenVPN listening to.
Unless your server is behind NAT, it should be your public IPv4 address.
IP address: 10.0.0.101

It seems this server is behind NAT. What is its public IPv4 address or hostname?
We need it for the clients to connect to the server.
``````````````````

```
Public IPv4 address or hostname: 192.168.206.133

Checking for IPv6 connectivity...

Your host does not appear to have IPv6 connectivity.

Do you want to enable IPv6 support (NAT)? [y/n]: n

What port do you want OpenVPN to listen to?
   1) Default: 1194
   2) Custom
   3) Random [49152-65535]
Port choice [1-3]: 1

What protocol do you want OpenVPN to use?
UDP is faster. Unless it is not available, you shouldn't use
TCP.
   1) UDP
   2) TCP
Protocol [1-2]: 1

What DNS resolvers do you want to use with the VPN?
   1) Current system resolvers (from /etc/resolv.conf)
   2) Self-hosted DNS Resolver (Unbound)
   3) Cloudflare (Anycast: worldwide)
   4) Quad9 (Anycast: worldwide)
   5) Quad9 uncensored (Anycast: worldwide)
   6) FDN (France)
   7) DNS.WATCH (Germany)
   8) OpenDNS (Anycast: worldwide)
   9) Google (Anycast: worldwide)
   10) Yandex Basic (Russia)
   11) AdGuard DNS (Anycast: worldwide)
   12) NextDNS (Anycast: worldwide)
   13) Custom
DNS [1-12]: 11


Do you want to use compression? It is not recommended
since the VORACLE attack makes use of it.
Enable compression? [y/n]: n


Do you want to customize encryption settings?
Unless you know what you're doing, you should stick with
the default parameters provided by the script.
Note that whatever you choose, all the choices presented
in the script are safe. (Unlike OpenVPN's defaults)
See
https://github.com/angristan/openvpn-install#security-and-
encryption to learn more.

Customize encryption settings? [y/n]: n
```

```
Okay, that was all I needed. We are ready to setup your
OpenVPN server now.
You will be able to generate a client at the end of the
installation.
Press any key to continue...


Tell me a name for the client.
The name must consist of alphanumeric character. It may
also include an underscore or a dash.
Client name: kazama

Do you want to protect the configuration file with a
password?
(e.g. encrypt the private key with a password)
   1) Add a passwordless client
   2) Use a password for the client
Select an option [1-2]: 1

Client kazama added.

The configuration file has been written to
/root/kazama.ovpn.
Download the .ovpn file and import it in your OpenVPN
client.
```

---

---

On Server:

~~~

```
# scp -r /root/kazama.ovpn 192.168.206.134:/root/
root@192.168.206.134's password:
kazama.ovpn
```

On Client:

~~~

```
# apt install openvpn -y

# openvpn --config ~/kazama.ovpn
```

On Windows

```````````

---

Install OPenVPN-AS on CentOS:

````````````````````````````

```
# yum -y install centos-release-scl-rh

# yum -y install
https://as-repository.openvpn.net/as-repo-centos7.rpm
```

```
# yum -y install openvpn-as
```

## Install OpenVPN-AS on Ubuntu:

As always, first make sure that your system has up-to-date packages.

```
# apt update
# apt upgrade
```

Next, install required dependencies.

```
# apt update && apt -y install ca-certificates wget net-tools
gnupg
```

Add the OpenVPN server to your repository list.

```
# wget
https://as-repository.openvpn.net/as-repo-public.asc -qO
/etc/apt/trusted.gpg.d/as-repository.asc
# echo "deb [arch=amd64
signed-by=/etc/apt/trusted.gpg.d/as-repository.asc]
http://as-repository.openvpn.net/as/debian bookworm
main">/etc/apt/sources.list.d/openvpn-as-repo.list
# apt update
```

Finally, install the OpenVPN access server.

```
# apt install openvpn-as -y
```

Access the Admin Dashboard
You can access the OpenVPN administrator dashboard at https://<your-ip>:943/admin or https://<your-domain>:943/admin. In either case, the default username is openvpn.

You will need to set the password for the openvpn user. You can do this with the passwd command.

```
passwd openvpn
```

OpenVPN Client Setup
In the admin dashboard, you can add users under User Management. Users can access the OpenVPN server at https://<your-ip>:943 or https://<your-domain>:943 where

they can login and download the client software for their device. Supported operating systems include Mac, Windows, iOS, Android, and Linux.

https://tldp.org/