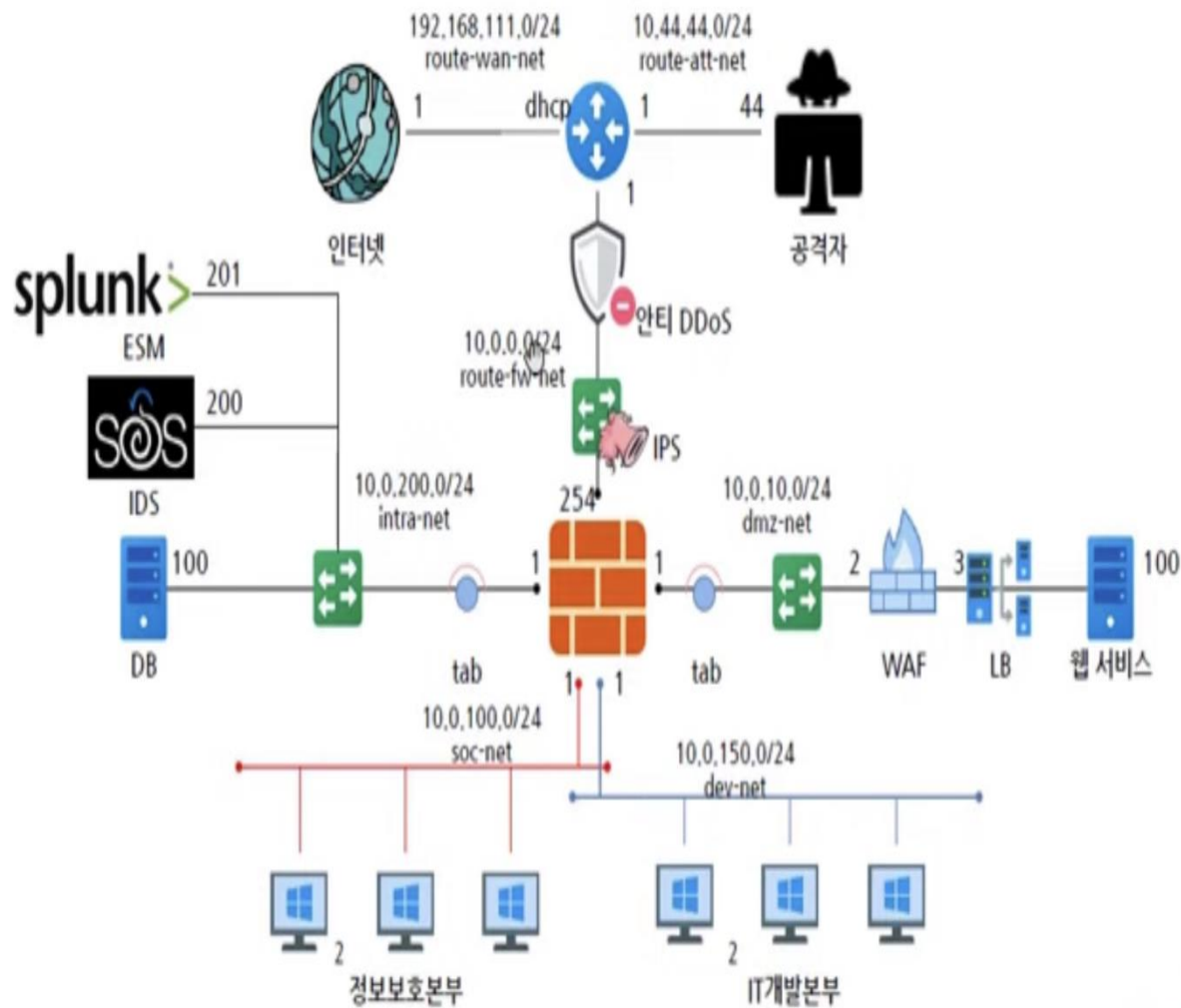


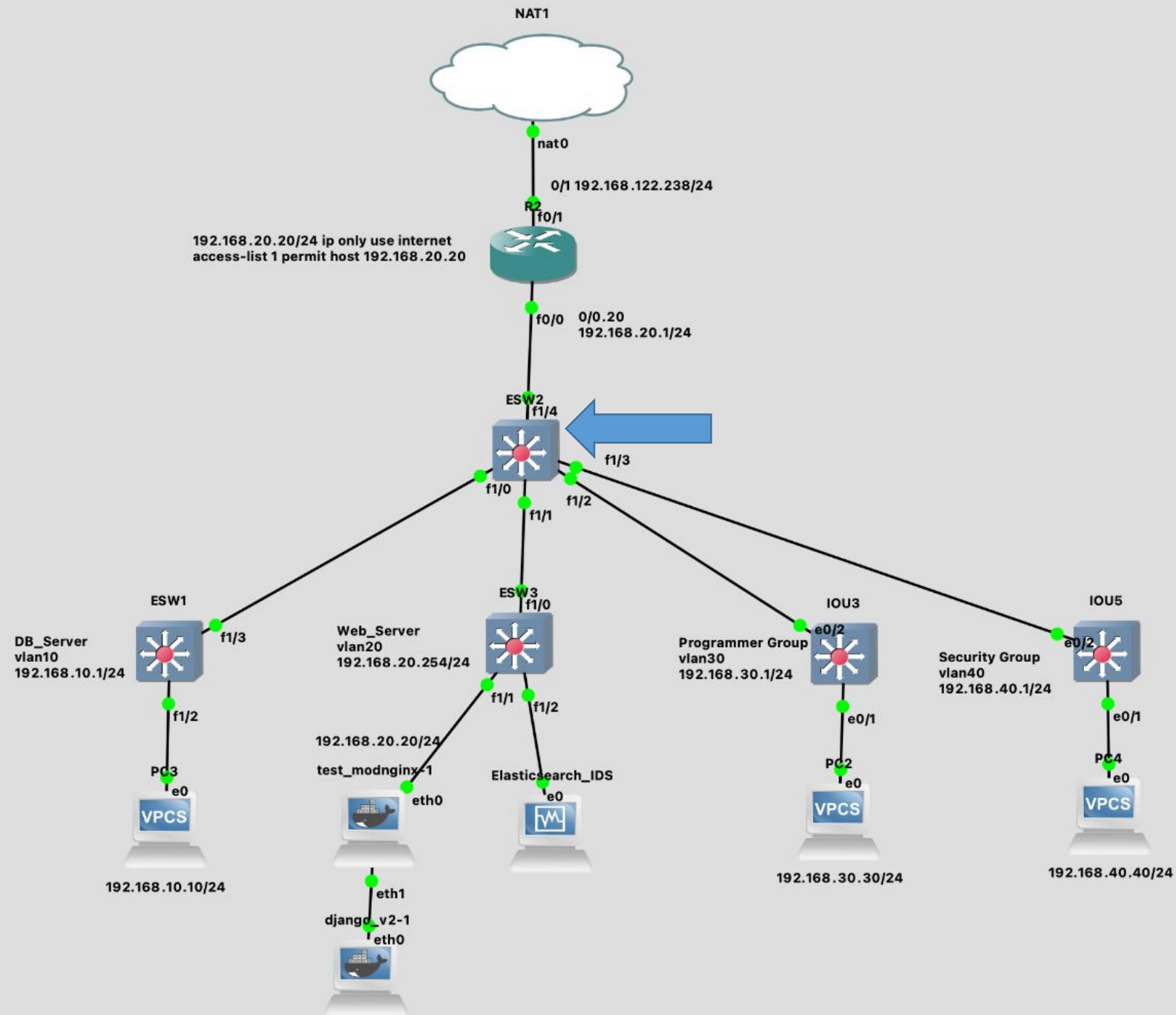
GNS3 Network Topology



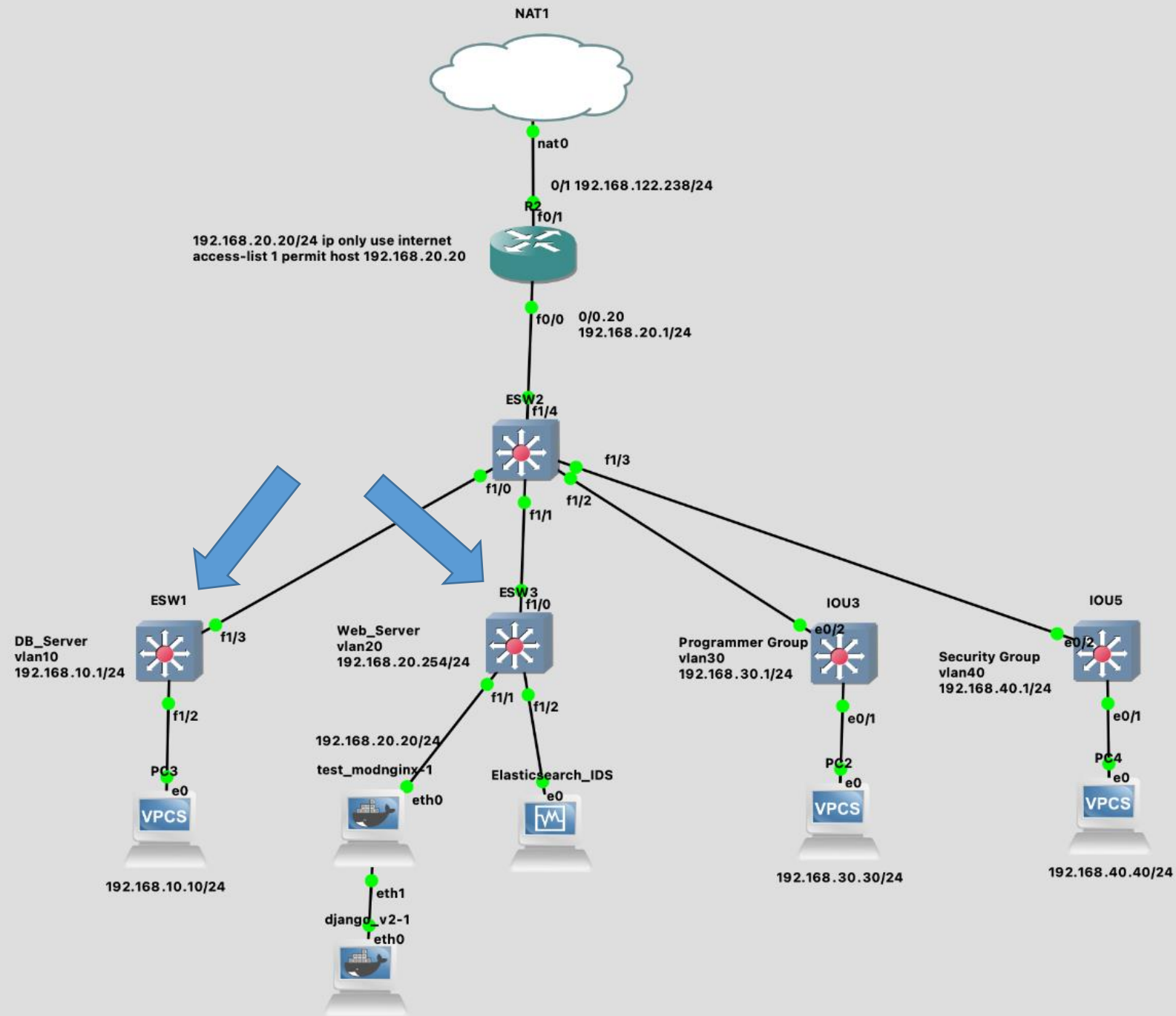
제작 동기

이 프로젝트를 시작한 계기는 유튜브를 보다가 이러한 토폴로지를 설명해주는 영상이었습니다. 이 영상의 내용은 이러한 토폴로지는 실무에서 사용되는 토폴로지라 설명을 하였습니다. 이에 실무의 조금이나마 도움이 되고자 제작했습니다.

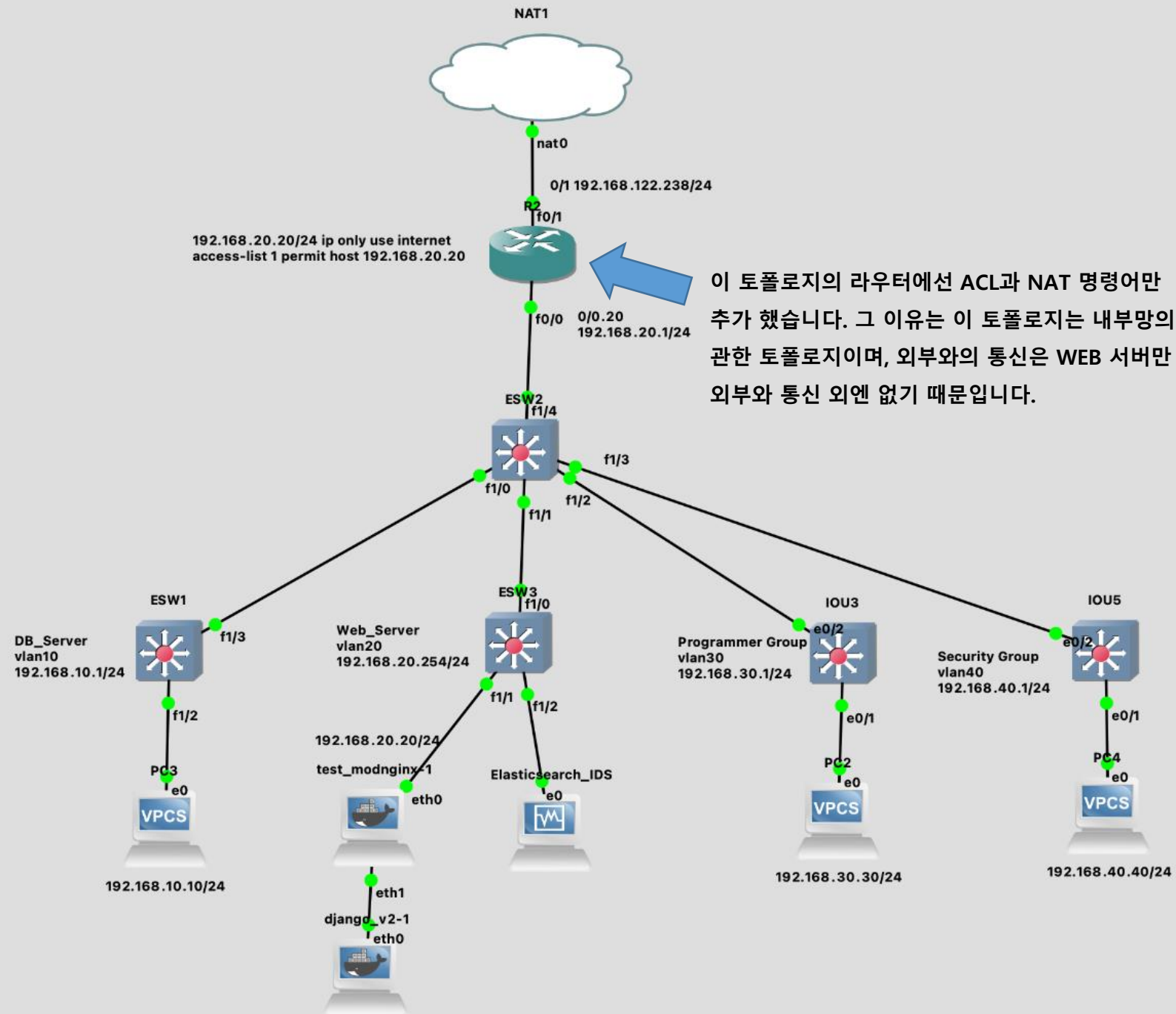
All Icons by Ico



백본에 ESW를 쓰는 이유는 백본스위치는 네트워크의 중심에 있으며, 모든 데이터가 지나가는 곳이기 때문에 기가비트 이더넷 포트를 주로 사용하며, GNS3에서 ESW가 기가비트 이더넷 포트를 지원하기 때문입니다.



DB서버와 WEB서버에 ESW를 사용한 이유는 데이터의 트래픽과 WEB서버에 접속하는 트래픽에 대하여 원활하게 하기 ESW를 사용했으며, 결정적으로 IDS를 사용하기 위해 미러링 포트를 스위치에 설정해주어야 하는데 IOU는 미러링 포트를 지원하지 않고 ESW는 미러링 포트를 지원하기에 ESW를 사용하게 되었습니다.



```

root@NetworkAutomation-1:~# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
 64 bytes from 8.8.8.8: icmp_seq=1 ttl=61 time=35.3 ms
 64 bytes from 8.8.8.8: icmp_seq=2 ttl=61 time=37.4 ms
 64 bytes from 8.8.8.8: icmp_seq=3 ttl=61 time=35.0 ms
^Z
[1]+  Stopped                  ping 8.8.8.8
root@NetworkAutomation-1:~# ping google.com
PING google.com (142.251.42.174) 56(84) bytes of data.
 64 bytes from nrt12s46-in-f14.1e100.net (142.251.42.174): icmp_seq=1 t
=34.9 ms
 64 bytes from nrt12s46-in-f14.1e100.net (142.251.42.174): icmp_seq=2 t
=36.6 ms
 64 bytes from nrt12s46-in-f14.1e100.net (142.251.42.174): icmp_seq=3 t
=36.9 ms
^Z
[2]+  Stopped                  ping google.com

```

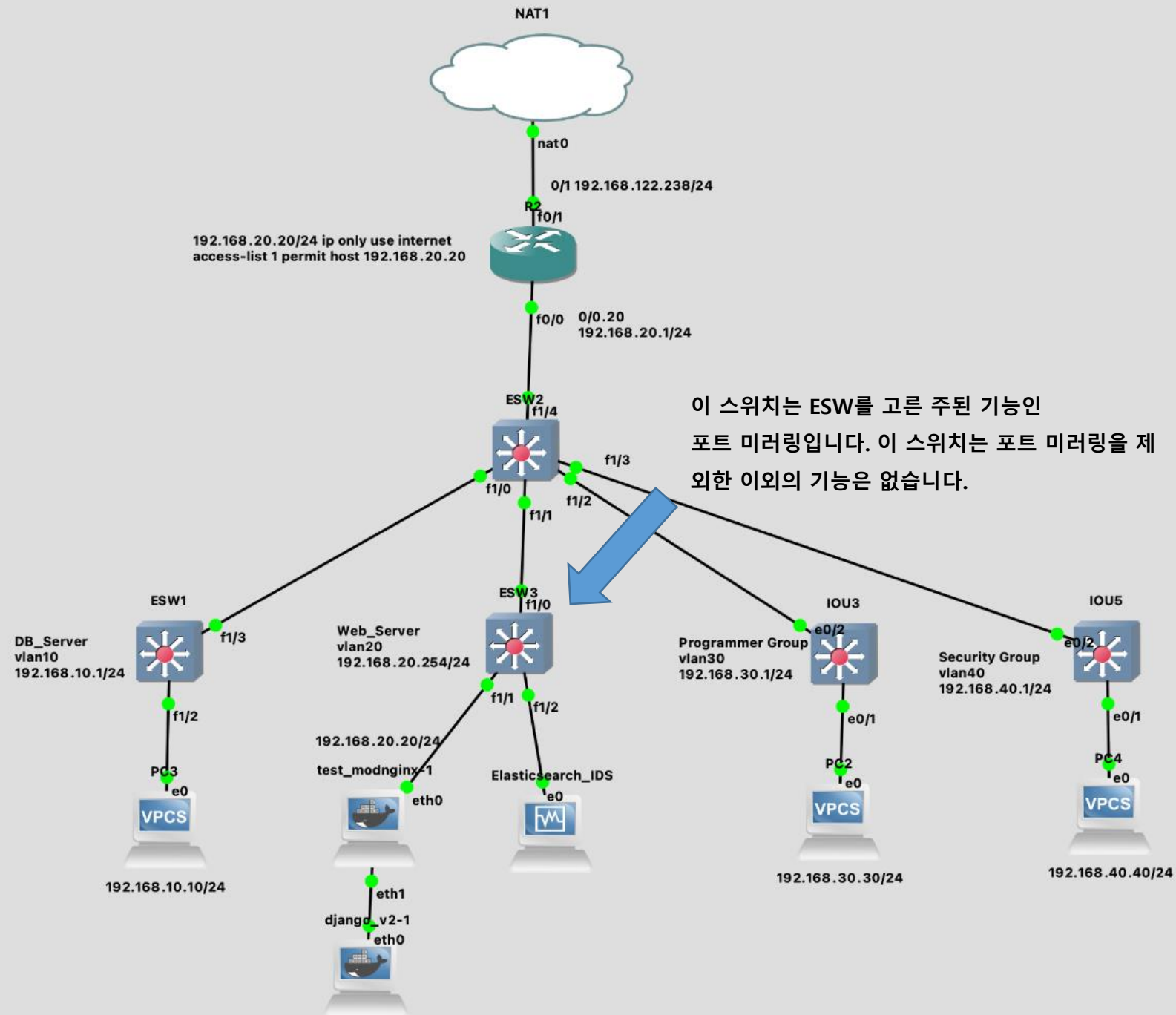
nat 설정 후 외부와 통신 성공한 사진

```

R2#
R2#
R2# csh
R2# sh
R2# show ac
R2# show acce
R2# show access-li
R2# show access-lists
Standard IP access list 1
 10 permit 192.168.20.20
R2#

```

라우터의 ACL 1번의 WEB 서버 IP만 허용시킨 사진



```

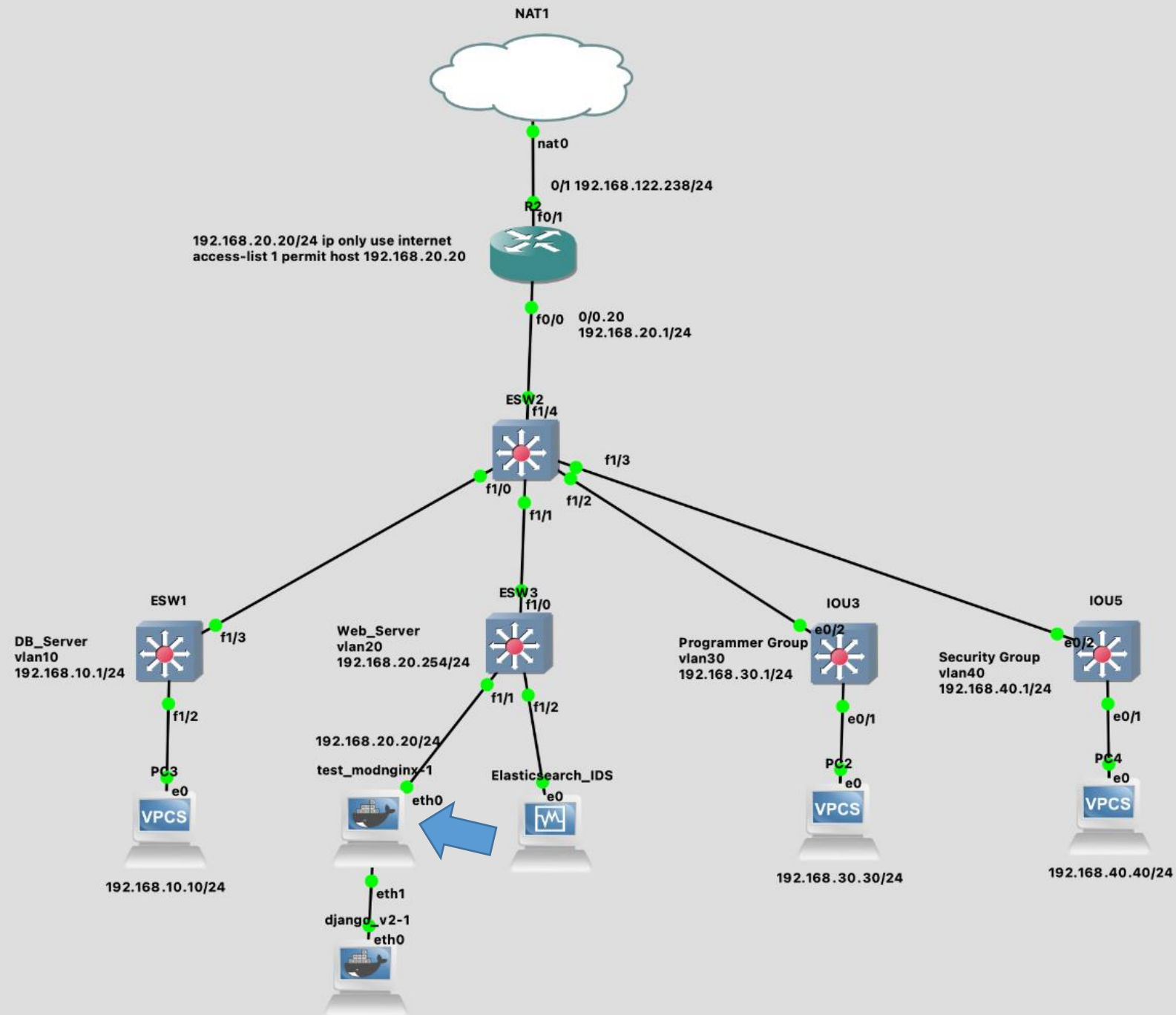
ESW3(config)#monitor se
ESW3(config)#monitor session 1 des
ESW3(config)#monitor session 1 destination int
ESW3(config)#monitor session 1 destination interface fa
ESW3(config)#monitor session 1 destination interface fastEthernet 1/2
ESW3(config)#
ESW3(config)#do show mo
ESW3(config)#do show moni
ESW3(config)#do show monitor session 1
Session 1
-----
Source Ports:
  RX Only:      None
  TX Only:      None
  Both:         Fa1/1
Source VLANs:
  RX Only:      None
  TX Only:      None
  Both:         None
Destination Ports: Fa1/2
Filter VLANs:    None

```

ESW3(config)#

Fa1/1: 출발지 포트

Fa1/2: 도착지 포트



WAF로 Modsecurity nginx를 채용한이유는 오픈소스 이기 때문입니다. 그리고 오픈소스의 장점으로 많은 자료가 있기에 정보도 얻기 편했으며, Dockerfile도 굉장히 잘 나와있기 때문입니다.

Modsecurity Nginx의 파일구조

```
jeff125 — test_modnginx-1 — telnet 192.168.56.108 5023 — 80x24
/etc/nginx # ls
① conf.d      ② mime.types  ③ modules    ④ scgi_params
④ fastcgi_params ⑤ modsecurity ⑥ nginx.conf ④ uwsgi_params
/etc/nginx # nano nginx.conf
```

- ① Modsecurity 구동의 필요한 setup on 명령어 및 백엔드 서버와 proxy 연결하는 파일이 있습니다.
- ② 확장자 명과 MIME 타입 목록입니다.
- ③ Nginx와 연동이 필요한 모듈이 있습니다
- ④ 웹 서버와 애플리케이션 사이를 연결해주는 파일입니다.
- ⑤ Rule 설정 할 수 있는 파일이 있습니다. *rule 파일은 OWASP Modsecurity 홈페이지에서 Core-Rule을 배포하고 있으며, github등 에서 rule을 추가 할 수 있습니다
- ⑥ Nginx 설정의 필요한 파일입니다.

Modsecurity 기본 설정

```
jeff125 — test_modnginx-1 — telnet 192.168.56.108 5023 — 80x24
GNU nano 5.4 conf.d/default.conf
server {
    listen      80;
    listen  [::]:80;
    server_name 192.168.10.2;
    modsecurity on;
    modsecurity_rules_file /etc/nginx/modsecurity/setup.conf;
    #access_log /var/log/nginx/host.access.log main;

    location / {
        proxy_pass http://192.168.10.2:8000;
        #root    /usr/share/nginx/html;
        #index   index.html index.htm;
    }

    #error_page 404              /404.html;

    # redirect server error pages to the static page /50x.html
    #
    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
```

- ① 백엔드 서버 IP를 입력해줍니다.
- ② Modsecurity on 해줍니다.
- ③ Rule 파일 경로를 입력해줍니다.
- ④ 기본 static 파일 경로 입력 및 백엔드 서버 IP를 입력해줍니다.

^G 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

```

GNU nano 5.4 nginx.conf
load_module modules/nginx_http_modsecurity_module.so; ①
user nginx;
worker_processes auto;

error_log /var/log/nginx/error.log notice;
pid /var/run/nginx.pid;

events {
    worker_connections 1024;
}

http {
    include /etc/nginx/mime.types;
    default_type application/octet-stream;

    log_format main '$remote_addr - $remote_user [$time_local] "$request" '
        '$status $body_bytes_sent "$http_referer" '
        '"$http_user_agent" "$http_x_forwarded_for"';
    [ Read 32 lines ]
    #G 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

jeff125 — test_modnginx-1 — telnet 192.168.56.108 5023 — 80x24
GNU nano 5.4 nginx.conf
http {
    include /etc/nginx/mime.types;
    default_type application/octet-stream;

    log_format main '$remote_addr - $remote_user [$time_local] "$request" '
        '$status $body_bytes_sent "$http_referer" '
        '"$http_user_agent" "$http_x_forwarded_for"';

    access_log /var/log/nginx/access.log main;

    sendfile on;
    #tcp_nopush on;

    keepalive_timeout 65;

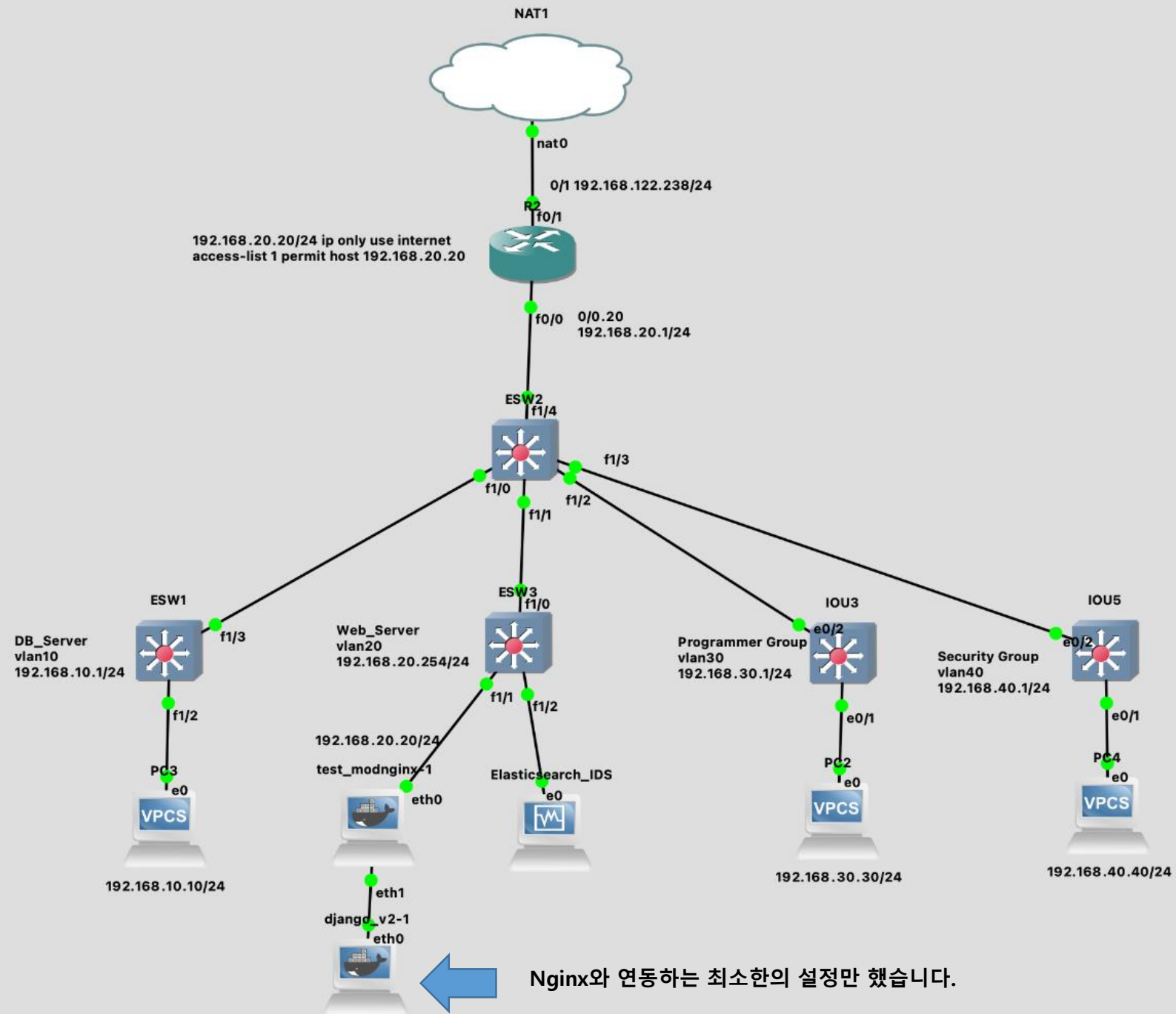
    #gzip on;

    include /etc/nginx/conf.d/*.conf; ②
}

#G 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

```

- ① Modsecurity와 nginx를 연동하기 위해 모듈 불러오는 구문입니다.
- ② Nginx의 설정 파일 위치입니다.



백엔드 서버 실행 명령어

jeff125 — django_v2-1 — telnet 192.168.56.108 5012 — 80x24

```
/home # uwsgi --http :8000 --chdir /home/ -w mysite.wsgi
```

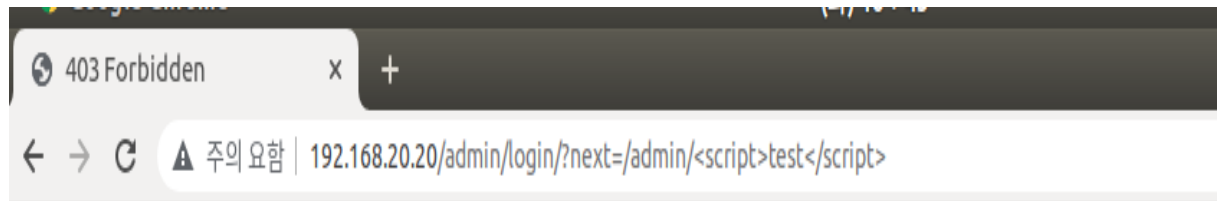
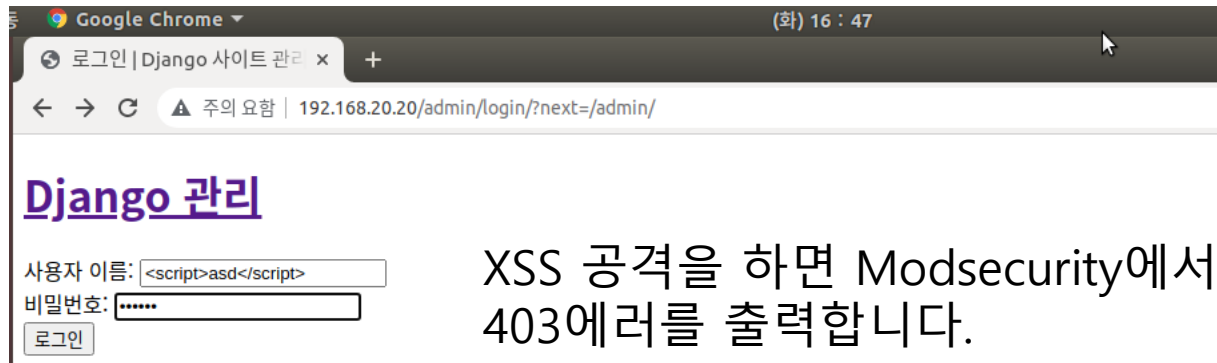
8000포트 프로젝트 경로 프로젝트 이름.wsgi
허용

Nginx와 Django 연동 성공

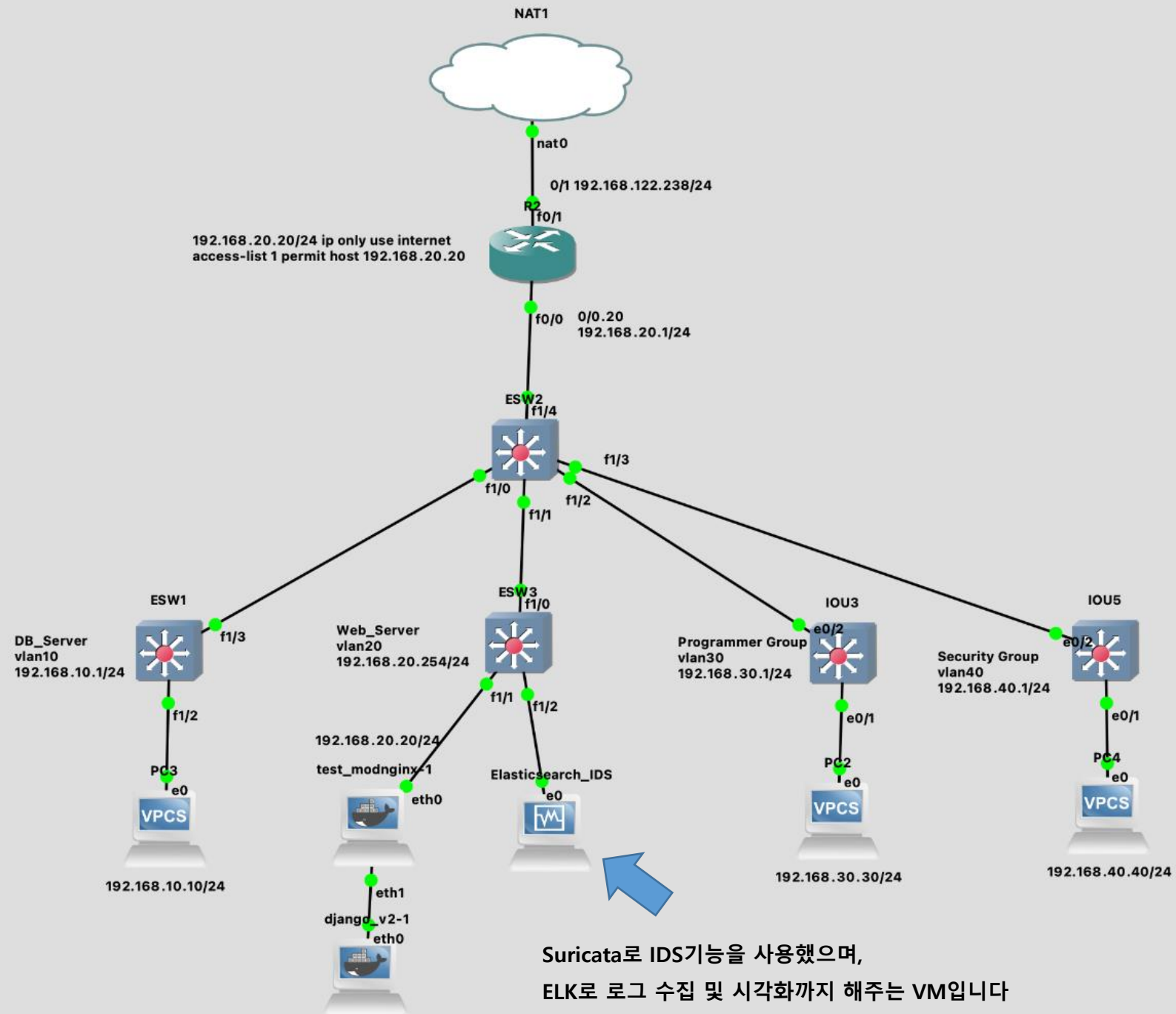
jeff125 — test_modnginx-1 — telnet 192.168.56.108 5023 — 80x24

```
/etc/nginx # curl -XGET 192.168.10.2:8000
<!DOCTYPE html>
<html lang="en">
<head>
  <meta http-equiv="content-type" content="text/html; charset=utf-8">
  <title>Page not found at /</title>
  <meta name="robots" content="NONE,NOARCHIVE">
  <style type="text/css">
    html * { padding:0; margin:0; }
    body * { padding:10px 20px; }
    body * * { padding:0; }
    body { font:small sans-serif; background:#eee; color:#000; }
    body>div { border-bottom:1px solid #ddd; }
    h1 { font-weight:normal; margin-bottom:.4em; }
    h1 span { font-size:60%; color:#666; font-weight:normal; }
    table { border:none; border-collapse: collapse; width:100%; }
    td, th { vertical-align:top; padding:2px 3px; }
    th { width:12em; text-align:right; color:#666; padding-right:.5em; }
    #info { background:#f6f6f6; }
    #info ol { margin: 0.5em 4em; }
    #info ol li { font-family: monospace; }
    #summary { background: #ffc; }
    #explanation { background:#eee; border-bottom: 0px none; }
    pre.exception_value { font-family: sans-serif; color: #575757; font-size: 1.2em; }
```

WAF 작동 확인



403 Forbidden



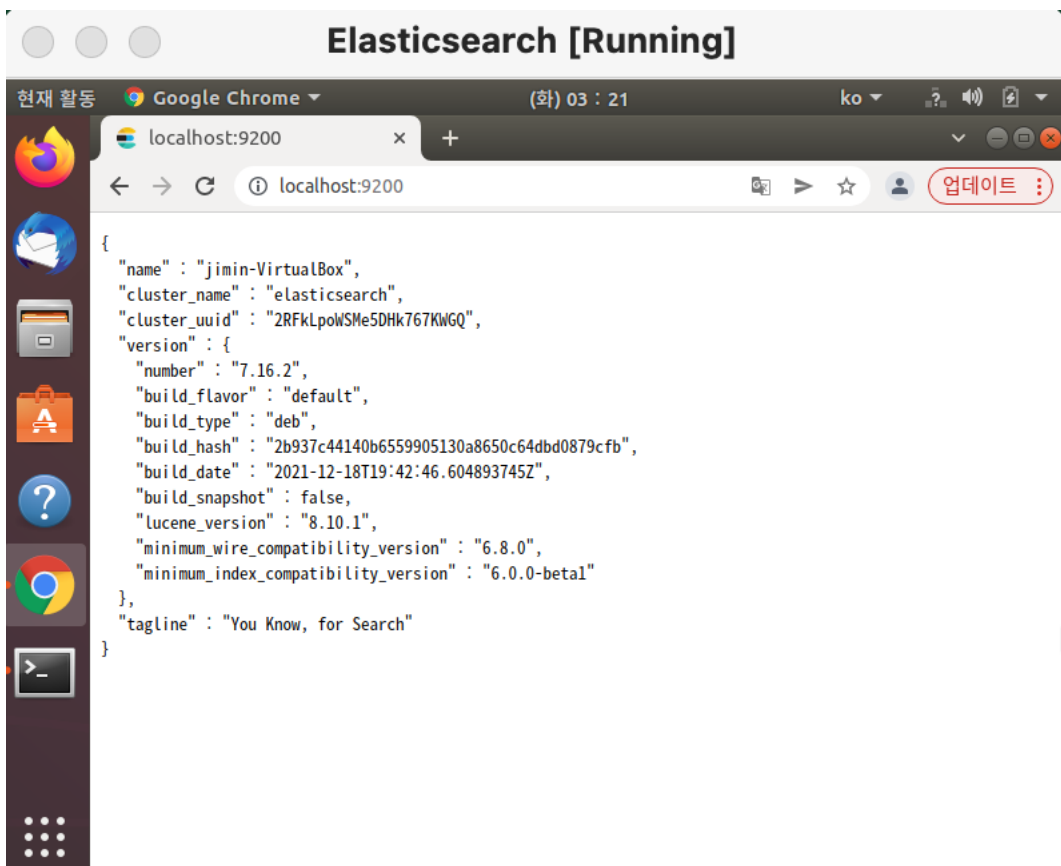
Suricata로 IDS기능을 사용했으며,
ELK로 로그 수집 및 시각화까지 해주는 VM입니다

Suricata 기본 설정

```
jimin@jimin-VirtualBox: /etc/logstash/conf.d
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
jimin@jimin-VirtualBox:/etc/logstash/conf.d$ nano logstash.conf
jimin@jimin-VirtualBox:/etc/logstash/conf.d$ nano logstash-sample.conf
jimin@jimin-VirtualBox:/etc/logstash/conf.d$ sudo chmod 775 /var/log/suricata/eve.json
```

IDS 기능을 수행하기 위해 우선 Suricata를 설치하고
/var/log/suricata/eve.json 파일의 권한을 설정합니다.
왜냐하면 eve.json에 로그가 기록 되며 Logstash가 eve.json
파일을 읽어서 Elasticsearch와 연동을 하는데 권한이 없으면
연동 할 수 없기 때문입니다.

Elasticsearch 설치 확인



Logstash나 Kibana 설정 전에 9200번 포트 접속에 접속하여 다음과 같은 화면이 나오는지 확인합니다.

Logstash 기본 설정

```
jimin@jimin-VirtualBox: /etc/logstash/conf.d
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
GNU nano 2.9.3 logstash-sample.conf

input {
  file {
    path => ["/var/log/suricata/eve.json"]
    codec => "json"
    type => "SuricataIDPS"
  }
}

filter {
  if [type] == "SuricataIDPS" {
    date {
      match => [ "timestamp", "ISO8601" ]
    }
    ruby {
      code => "
        if event.get('[event_type]') == 'fileinfo'
          event.set('[fileinfo][type]', event.get('[fileinfo][magic]').to_s.split(',')[0])
        end

        #database => "/opt/logstash/vendor/geoip/GeoLiteCity.dat"
        add_field => [ "[geoip][coordinates]", "%{[geoip][longitude]}" ]
        add_field => [ "[geoip][coordinates]", "%{[geoip][latitude]}" ]
      ]
      mutate {
        convert => [ "[geoip][coordinates]", "float" ]
      }
    }
  }
}

output {
  elasticsearch {
    hosts => "localhost"
  }
}
```

/etc/logstash/conf.d

- ① Eve.json의 파일 위치 및 세부 설정을 합니다.
- ② 읽어온 json파일을 elasticsearch에 전달해 주기 위해 필요한 IP주소를 입력해 줍니다.

Kibana 기본 설정

```
root@jimin-VirtualBox: /etc/kibana
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
GNU nano 2.9.3 kibana.yml 변경됨.
#Kibana is served by a back end server. This setting specifies the port to use.
server.port: 5601 ①

# Specifies the address to which the Kibana server will bind. IP addresses and $
# The default is 'localhost', which usually means remote machines will not be a$
# To allow connections from remote users, set this parameter to a non-loopback $
#server.host: "localhost"

# Enables you to specify a path to mount Kibana at if you are running behind a $
# Use the `server.rewriteBasePath` setting to tell Kibana if it should remove t$
# from requests it receives, and to prevent a deprecation warning at startup.
# This setting cannot end in a slash.
#server.basePath: ""

# The Kibana server's name. This is used for display purposes.
#server.name: "your-hostname"


# The URLs of the Elasticsearch instances to use for all your queries.
elasticsearch.hosts: ["http://localhost:9200"]
#elasticsearch.hosts: ["http://localhost:9200"]
```

/etc/kibana/kibana.yml

각 번호의 주석을 지워주세요

- ① Kibana의 기본 포트는 5601입니다.
- ② Elasticsearch의 기본 포트는 9200입니다.







Kibana log 템플릿 추가하기

 StamusNetworks / **KTS7** Public

[Code](#) [Issues 4](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#)

master 1 branch 0 tags

[Go to file](#) [Add file](#) [Code](#)

 pevma	kts7: Major update of dashboards and vizs.	8d63f15 on 15 Mar 17 commits
 API-KIBANA7	kts7: Major update of dashboards and vizs.	5 months ago
 dashboards	kts7: Major update of dashboards and vizs.	5 months ago
 es-template	dashboards: Initial dashboard upload	2 years ago
 LICENSE	Initial commit	2 years ago
 README.rst	doc: Update import instructions	2 years ago

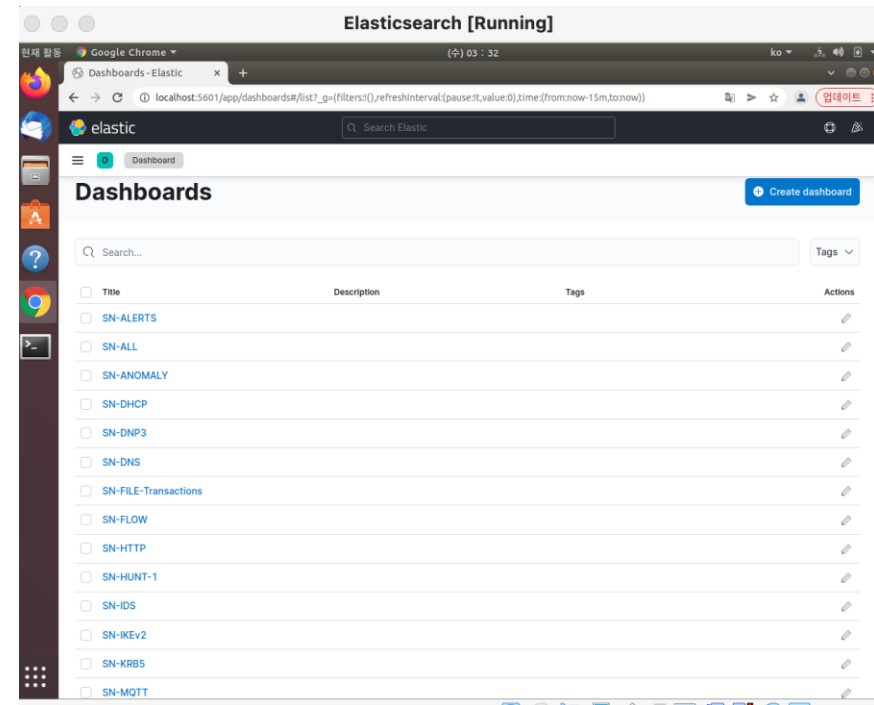
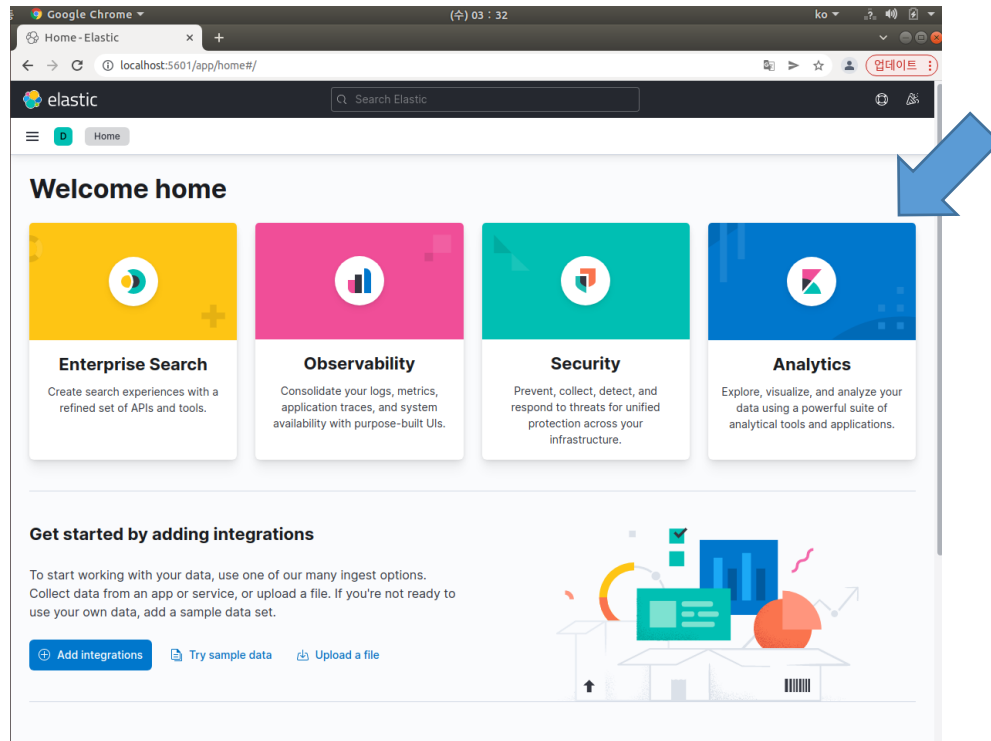
How to use

To import all the vizualizaitons and dahsboards to Kibana 7 using the native API - on the host runing Kibana 7 or ELK7:

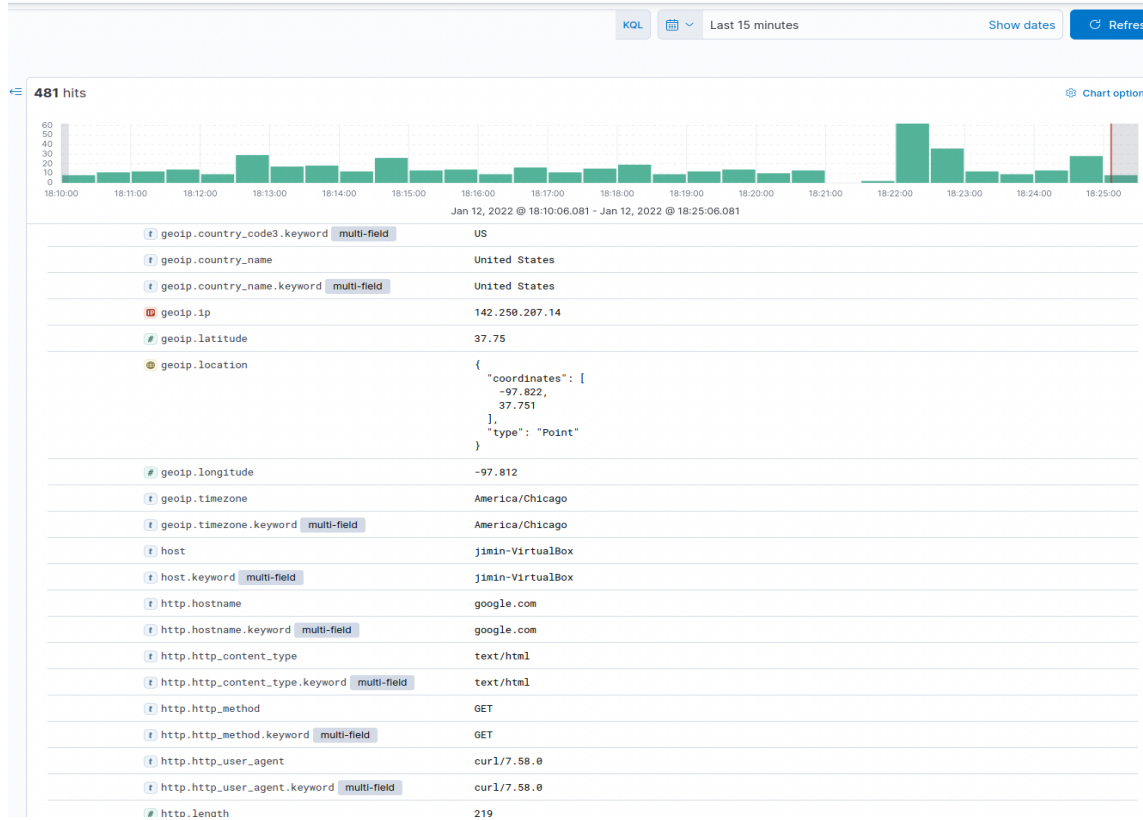
```
cd API-KIBANA7
curl -X POST "localhost:5601/api/saved_objects/_import" -H 'kbn-xsrf: true' --form file=@index-pattern.ndjson
curl -X POST "localhost:5601/api/saved_objects/_import" -H 'kbn-xsrf: true' --form file=@search.ndjson
curl -X POST "localhost:5601/api/saved_objects/_import" -H 'kbn-xsrf: true' --form file=@visualization.ndjson
curl -X POST "localhost:5601/api/saved_objects/_import" -H 'kbn-xsrf: true' --form file=@dashboard.ndjson
curl -X POST "localhost:5601/api/saved_objects/_import" -H 'kbn-xsrf: true' --form file=@query.ndjson
service kibana restart
```

Kibana에 로그 데이터 시각화 템플릿입니다.

Kibana 템플릿 연동 확인



```
[53]+ Stopped ping 8.8.8.8
[root@pc1:~# ping google.com
PING google.com (142.251.42.174) 56(84) bytes of data.
64 bytes from nrt12s46-in-f14.1e100.net (142.251.42.174): i
9 time=49.8 ms
64 bytes from nrt12s46-in-f14.1e100.net (142.251.42.174): i
9 time=56.5 ms
64 bytes from nrt12s46-in-f14.1e100.net (142.251.42.174): i
9 time=59.4 ms
^Z
[54]+ Stopped ping google.com
root@pc1:~#
```



이제 설정이 끝나 WAF PC의 로그를 확인하기 위해 WAF PC에서 google.com으로 핑을 보낸 후 IDS PC에서 로그 확인 한 사진입니다.

프로젝트 후기

노트북에서 진행하는 프로젝트이기에 최대한 PC자원을 아끼면서 프로젝트를 진행하고 싶었습니다. 그렇기에 Docker를 활용하여 자원을 효율적으로 이용했습니다. 하지만 **단점**도 있었습니다.

GNS3에서 Docker 실행 구조가 호스트 PC에서 컨테이너 실행이 아닌, GNS3 VM에서 컨테이너가 올라가는 구조이기 때문에 컨테이너의 메모리, 프로세스를 원하는 만큼 추가해주기 어려운 점이 있었습니다. 이에 ELK가 최소 8GB의 메모리를 할당하지 못해 호스트 PC에서 VM연동하여 사용했었습니다.

이 프로젝트를 진행하기 전에는 Docker와 ELK가 편하고 되게 좋다고만 들었었지 직접 사용해 본적은 없었습니다. 하지만 프로젝트를 제작하면서 Docker의 컨테이너의 강한 장점을 직접 실감하게 되었으며, ELK의 데이터 시각화의 강점을 알게 되었지만, 결정적으로 Elasticsearch의 데이터 검색의 **역색인 (inverted index)** 제대로 활용하지 못한 점이 아쉽다. 수천, 수만의 로그가 그 만큼 없었으며, PC의 사양으로 인해 ELK 실행 조차 벅차 제대로 된 검색 속도를 체감하기가 어려웠습니다.