



Challenge voorstel

Adam Nunes

Mijn challenge voorstel gaat over het hacken van een webserver. Hierbij wil ik de volgende onderwerpen behandelen.

SQL injection

Python hack tooltje maken (brute force attack)

Kubernetes hacken

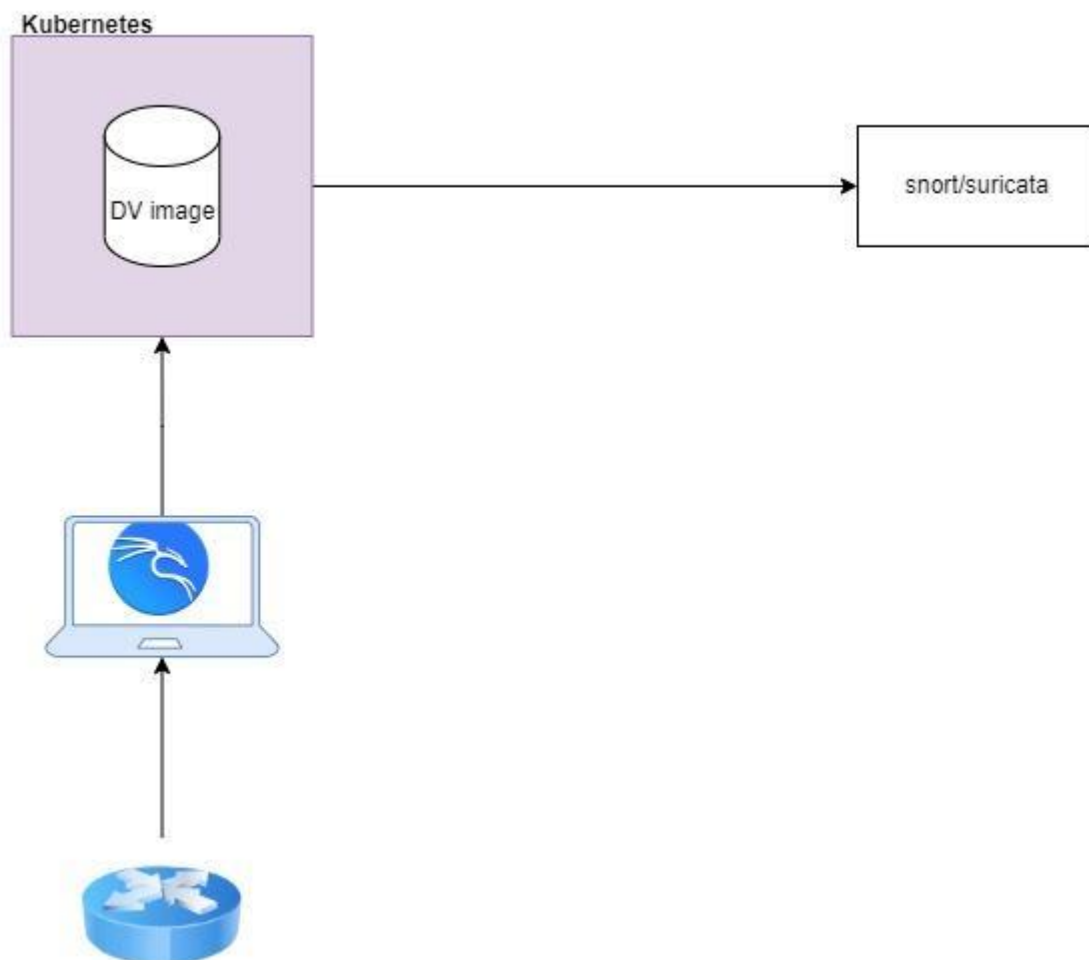
Website hacken

https://demo.testfire.net/index.jsp?content=inside_contact.htm

Het idee van het project is als volgt.

Er word door middel van Kubernetes een webserver gedeployed. Met het werkstation wordt de webserver bereikt. De website wordt aangevallen door middel van een sql injection en een brute force attack.

De logs van de webserver worden verbonden aan snort of suricata. De monitoring services blokkeren de mogelijke aanvaller. Verder wordt een TCO gemaakt van het project. Tot slot zal er worden gekeken naar het automatiseren van het netwerk.



De opdrachten die ik in dit project ga maken zijn gerangschikt in een Must Should Could have methode. Let op het is de bedoeling dat ik al deze opgave ga maken. Echter heb ik deze schema opgesteld om aan te geven hoe hoog ik de prioriteiten inschat per opdracht.

M	S	C
SQL injecties	snort/suricata banned aanvaller	Automation
Hack Tool	TCO hacktool	
snort logs	Ip adres migration	
scaling		

K3s installatie

Stap 1 Download k3s

```
curl -sfL https://get.k3s.io | sh -
```

Stap 2 check of de installatie is gewerkt

```
sudo k3s kubectl get node
```

Installatie voor worker node

Stap 1 Genereer een token vanuit de master node

```
sudo cat /var/lib/rancher/k3s/server/node-token
```

Stap 2 verbind de worker node met de master node

```
curl -sfL https://get.k3s.io | K3S_TOKEN="JOUWTOKEN" K3S_URL="https://[your IPaddress]:6443" K3S_NODE_NAME="Naammagjezelfkiezen" sh -
```

Stap 3 Check de resultaten door terug te gaan naar de master node en type het volgende:

```
sudo k3s kubectl get node
```

Damn Vulnerable Web Application

Om de webapplicatie te deployen moet het volgende yaml script worden gebruikt.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: dvwa-deployment
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: dvwa
  template:
    metadata:
      labels:
        app: dvwa
    spec:
      containers:
        - name: dvwa
          image: vulnerables/web-dvwa
          imagePullPolicy: Always
          ports:
            - containerPort: 80
              name: web
              protocol: TCP
          readinessProbe:
            httpGet:
              port: web
              path: /
```

Deze website draait op de cluster echter kan een gebruiker de website niet benaderen omdat de website een intern ip adres heeft gekregen van kubernetes.

Om toch de website te bereiken moet er een service worden gedeployed die een soort portforwarding in werking zet dit wordt ook wel een node port genoemd.

Omdat de website poort 80 zal gebruiken zal de node poort 31111 worden geopend.

```

apiVersion: v1
kind: Service
metadata:
  name: dvwa-nodeport
  namespace: default
spec:
  type: NodePort
  selector:
    app: dvwa
  ports:
    - name: web
      port: 80
      targetPort: web // refereert naar de poort die de container gebruikt in de
deployment file
      nodePort: 31111 // luistert op poort 80

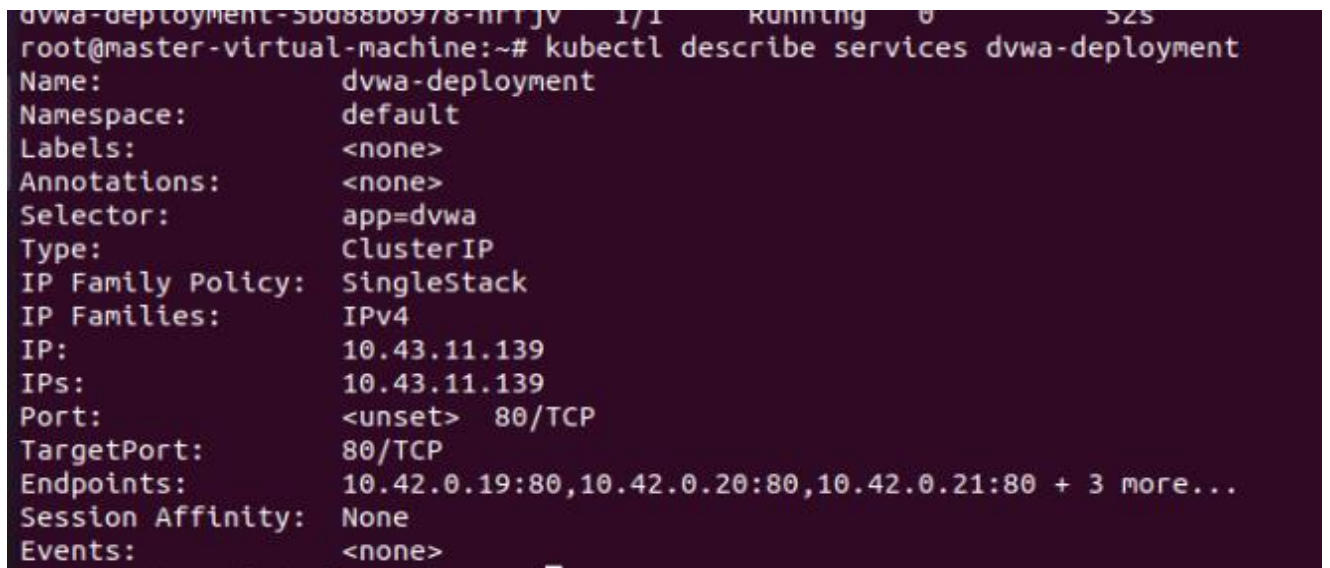
```

Wanneer de websites overladen wordt door bijvoorbeeld hacks dan wordt een loadbalancer toegepast. Deze loadbalancer maakt een nieuwe website aan en geeft het een aparte interne ip kubernetes ip adres. Echter ziet de gebruiker hier niets van terug.

```

kubectrl expose deployment dvwa --port=80 --target-port=80 \
--name=dvwa-service -- type=LoadBalancer

```



```

dvwa-deployment-5b088b0978-nr1jv 1/1 Running 0 52s
root@master-virtual-machine:~# kubectl describe services dvwa-deployment
Name:                dvwa-deployment
Namespace:           default
Labels:              <none>
Annotations:         <none>
Selector:            app=dvwa
Type:               ClusterIP
IP Family Policy:   SingleStack
IP Families:        IPv4
IP:                 10.43.11.139
IPs:                10.43.11.139
Port:               <unset> 80/TCP
TargetPort:         80/TCP
Endpoints:          10.42.0.19:80,10.42.0.20:80,10.42.0.21:80 + 3 more...
Session Affinity:   None
Events:             <none>

```

Wanneer gekeken wordt naar de regel **Endpoints** dan zie je dat er 6 interne ip adressen zijn aangemaakt voor dezelfde website.

Hack simulatie (IDS/IPS)

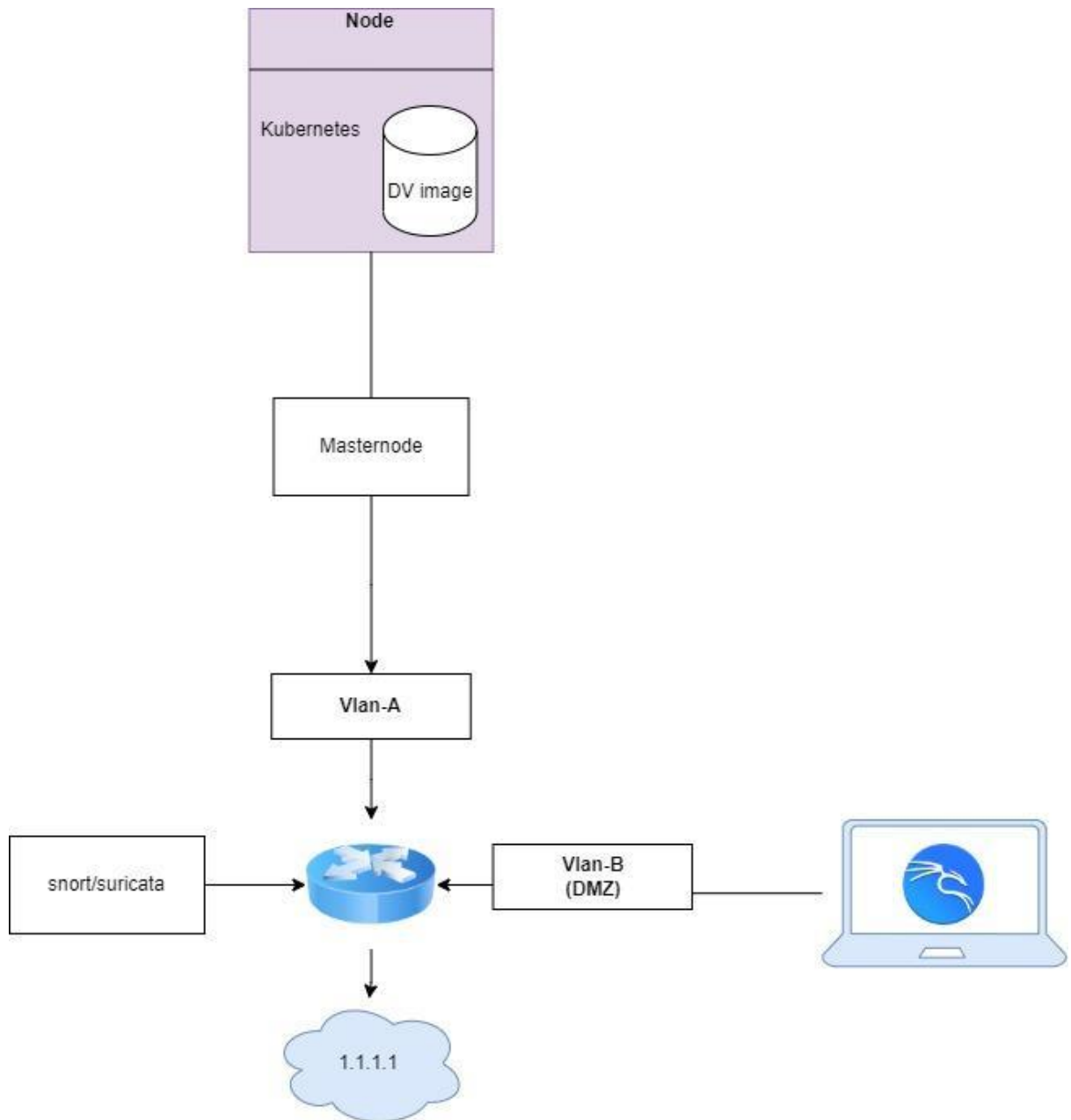
Om ervoor te zorgen dat een sql injectie niet mogelijk gemaakt wordt. Zal ik suricata installeren. Om de regels van suricata te testen zal ik d.m.v een dmz kali linux voor de router plaatsen in de hoop dat dit een reactie uitlokt van suricata.

DMZ

In tegenstelling tot de gebruikelijke configuratie zal deze documentatie een dmz zone bevatten waarin slechts een router gebruikt wordt. Dit bespaard hardware kosten en maakt onderhoud overzichtelijker.

1. Open seclap → kies pfsense → voeg dhcp, vlan A en vlan B toe
2. Open de Pfsense Gui → click op interfaces daarna op Add → Noem de interface Dmz en click op save
3. Open de nieuw gemaakte Dmz interface → vink Enable interface aan → Click op Static IPv4 → Bij static IPv4 address type de default gateway in → Click op save
4. Click op Firewall → Rules → DMZ → Add rule → Bij Protocol kies any → bij source kies DMZ net → Bij Destination kies any → save

Nieuwe netwerktekening













Vanwege de hoge effectiviteit en gratis kosten zal ik snort kiezen als ips/ids systeem.

Snort installatie

1. Open pfsense → System → PacketManager → Available package type in “snort” install → Click op services en vervolgens op snort → Click op snort interfaces → Add → click op het potloodje.
2. Click op enable interfaces → Kies Wan → save
3. Click op Global settings → Enable Snort VP → Vul snort oink code in (Deze code krijg je pas nadat je een account hebt aangemaakt) → Klik op enable Snort GPLv2 (Omdat ik geen abonnement heb bij snort kies ik voor de gratis community versie)

- Enable ET open → Enable FEODE tracker botnet rules (Voegt een reeks regels toe aangaande botnets)
- Update interval (Dit zijn de momenten waarop updates uitgevoerd kunnen worden **heel belangrijk** omdat nieuwe kwetsbaarheden op elk moment gevonden kunnen worden) Voor deze documentatie word er gekozen voor 6 uur.
 - Bij General settings → Remove blocker host wordt in deze situatie op 15 minuten gezet. (Dit is de tijd die een ip adres is geblokkeerd van het netwerk dit gebeurd wanneer de host van het ip een security regel breekt)
 - Save
 - Click op updates → Update rules
 - Click op snort interfaces → Click op het pijltje Interface is geïnstalleerd
 - Click op alerts

2022-12-05 09:41:05		3	TCP	Not Suspicious Traffic	192.168.246.18	11801	185.125.190.39	80	1:2013504	ET POLICY GNU/Linux APT User-Agent Outbound likely related to package management
2022-12-05 09:41:05		3	TCP	Not Suspicious Traffic	192.168.246.18	6149	213.136.12.213	80	1:2013504	ET POLICY GNU/Linux APT User-Agent Outbound likely related to package management
2022-12-05 09:41:05		3	TCP	Not Suspicious Traffic	192.168.246.18	45058	213.136.12.213	80	1:2013504	ET POLICY GNU/Linux APT User-Agent Outbound likely related to package management
2022-12-05 09:40:12		3	TCP	Not Suspicious Traffic	192.168.246.18	45994	91.189.91.39	80	1:2013504	ET POLICY GNU/Linux APT User-Agent Outbound likely related to package management
2022-12-05 09:35:34		3	TCP	Not Suspicious Traffic	192.168.246.18	60308	91.189.91.39	80	1:2013504	ET POLICY GNU/Linux APT User-Agent Outbound likely related to package management
2022-12-05 09:35:34		3	UDP	Misc activity	192.168.246.18	52039	1.1.1.1	53	1:2035465	ET INFO Observed Discord Domain in DNS Lookup (discord.com)
2022-12-05 09:35:34		3	UDP	Misc activity	192.168.246.18	27358	1.1.1.1	53	1:2035465	ET INFO Observed Discord Domain in DNS Lookup (discord.com)
2022-12-05 09:35:10		3	UDP	Misc activity	192.168.246.18	41875	1.1.1.1	53	1:2035465	ET INFO Observed Discord Domain in DNS Lookup (discord.com)
2022-12-05 09:35:10		3	UDP	Misc activity	192.168.246.18	45068	1.1.1.1	53	1:2035465	ET INFO Observed Discord Domain in DNS Lookup (discord.com)
2022-12-05 09:29:01		3	TCP	Not Suspicious Traffic	192.168.246.18	22647	185.125.190.36	80	1:2013504	ET POLICY GNU/Linux APT User-Agent Outbound likely related to package management

Dit zijn de alerts die bij mij zijn binnengekomen. 192.168.246.18 is mijn wan ip. Er staan hier twee soorten meldingen "ET POLICY GNU/LINUX User-Agent" en ET info ...Discord.com.

De reden dat deze meldingen binnen zijn gekomen is omdat de wan word benaderd door discord door de app te openen. En de tweede melding werd geregistreerd door een sudo apt update commando.

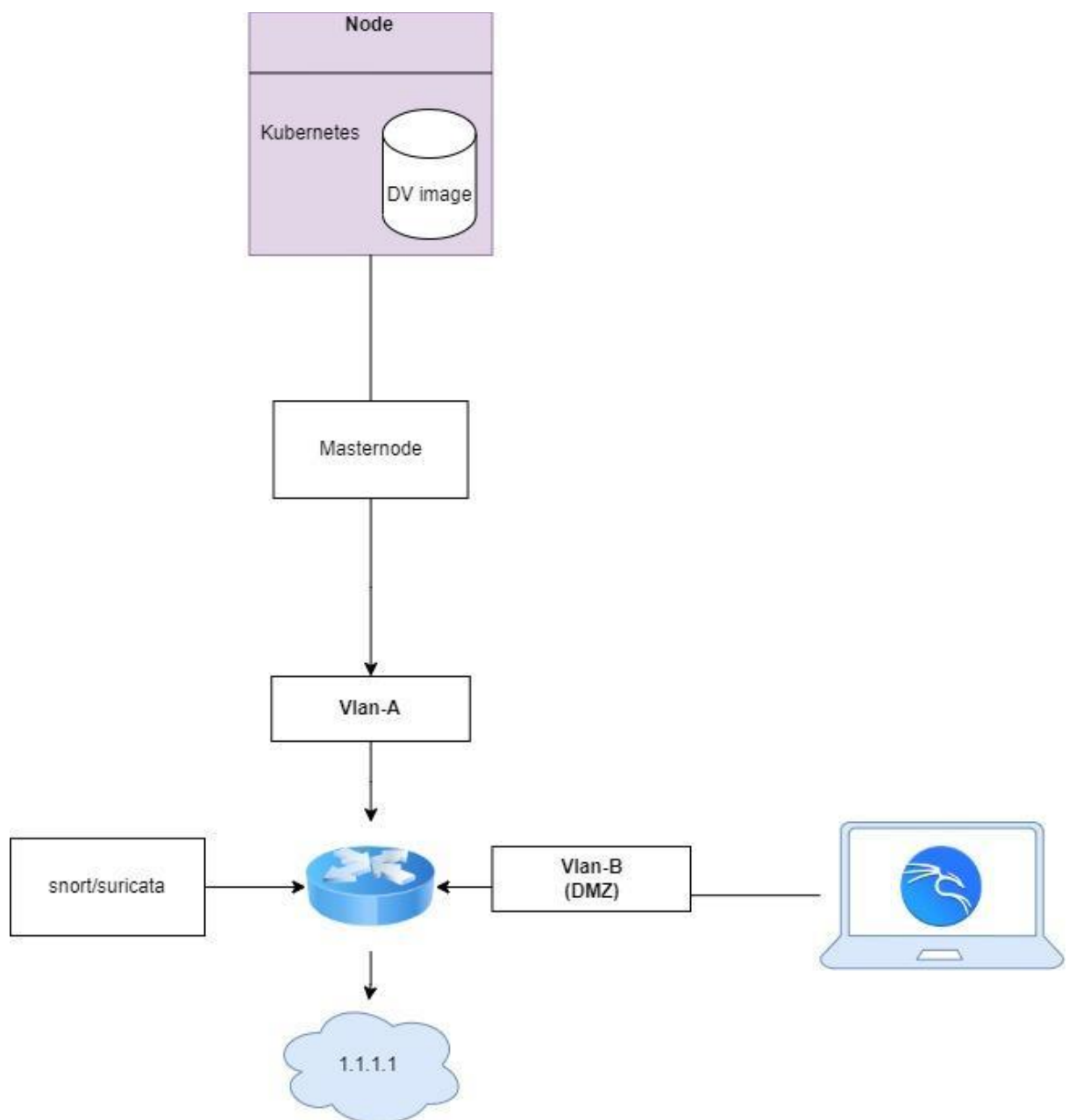
Nu is het doel om meer dan alleen een melding te genereren. De volgende stap is het laten blokkeren van de kali linux machine.

Om dit te verwezenlijken zal een nmap scan worden uitgevoerd op de router.

Open kali → In terminal type `nmap -sT -v -P0 192.168.246.18` (De wan Ip)

Dit commando checked of de poorten open staan.

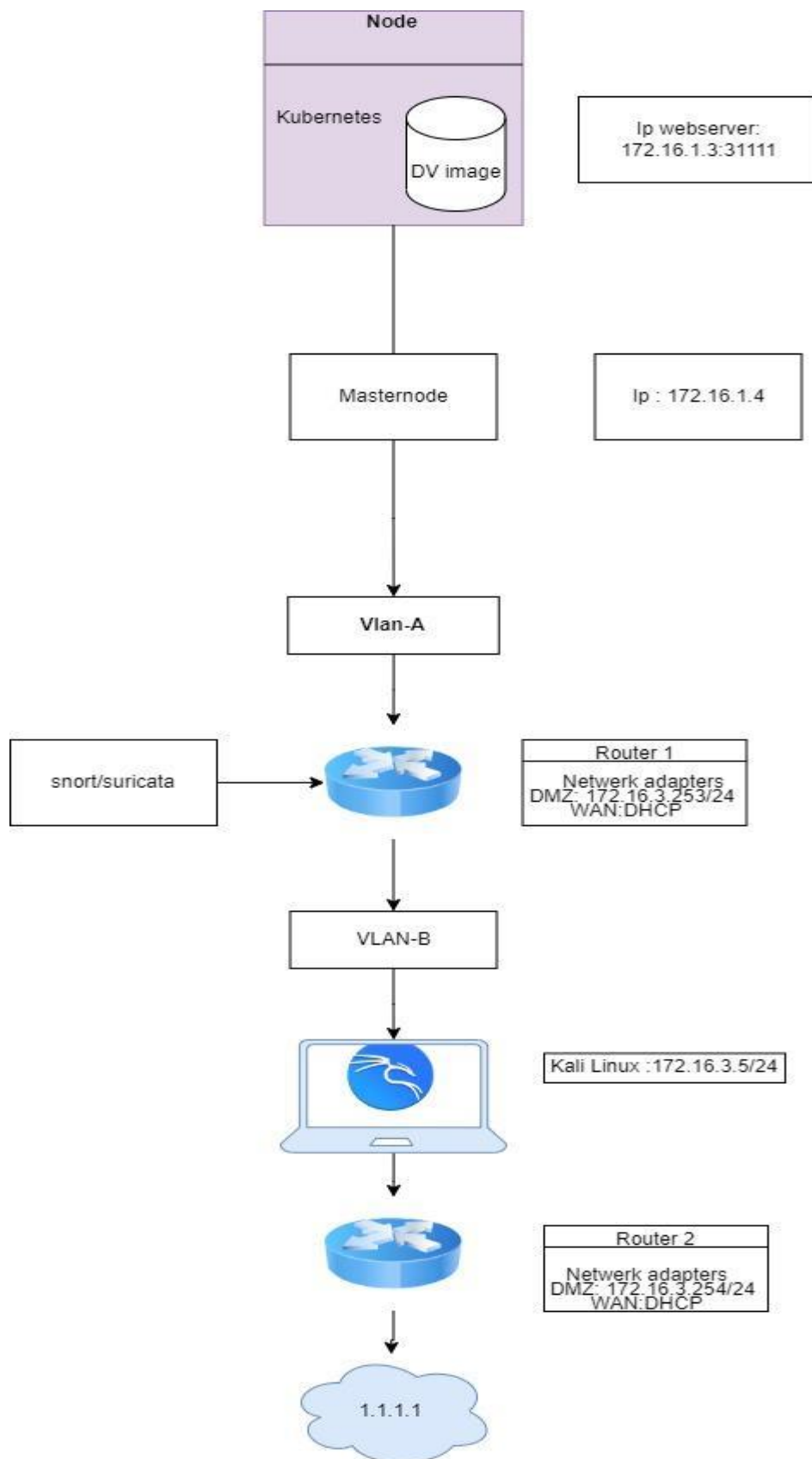
In principe zou ik geblokkeerd moeten worden echter is dit niet gebeurd. Om dit probleem op te lossen nam ik een ander kijkje naar de netwerktekening.



Bij het bekijken van mijn tekening viel het mij op hoe de dmz zone in principe dezelfde wan kant gebruikt. Wellicht dat de nmap scan niet word gezien als een bedreiging omdat de machine in het lan van het netwerk staat. Daarom zal ik nu een traditionele DMZ gebruiken waarbij de kali letterlijk voor de router staat.

Er kan wellicht worden afgevraagd waarom ik niet gewoon snort op de lan interface zet. Dit heeft twee redenen De eerste is dat het niet heel realistisch meeste is aanvallen worden vanuit de wan gedaan. Ten tweede heb ik dit al geprobeerd maar bracht het mij geen succes.

Volledige netwerktekening



Wanneer ik nu dezelfde nmap scan doen krijg ik de volgende meldingen.

30 Entries in Active Log										
Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2022-12-09 09:45:46		2	TCP	Attempted Information Leak	172.16.3.5 	34586	172.16.3.253 	5802	1:2002910 	ET SCAN Potential VNC Scan 5800-5820
2022-12-09 09:45:36		2	TCP	Potentially Bad Traffic	172.16.3.5 	49472	172.16.3.253 	5432	1:2010939 	ET SCAN Suspicious inbound to PostgreSQL port 5432
2022-12-09 09:45:36		2	TCP	Potentially Bad Traffic	172.16.3.5 	49470	172.16.3.253 	5432	1:2010939 	ET SCAN Suspicious inbound to PostgreSQL port 5432
2022-12-09 09:45:32		2	TCP	Attempted Information Leak	172.16.3.5 	38260	172.16.3.253 	5901	1:2002911 	ET SCAN Potential VNC Scan 5900-5920
2022-12-09 09:45:32		2	TCP	Potentially Bad Traffic	172.16.3.5 	46668	172.16.3.253 	1521	1:2010936 	ET SCAN Suspicious inbound to Oracle SQL port 1521
2022-12-09 09:45:32		2	TCP	Potentially Bad Traffic	172.16.3.5 	46666	172.16.3.253 	1521	1:2010936 	ET SCAN Suspicious inbound to Oracle SQL port 1521

Nu staat er onder class “Potential Badd traffic” met de Ip van mijn kali en de specifieke poort die gescanned is.

Blocked Hosts

Download

Clear

All blocked hosts will be saved

All blocked hosts will be removed

Refresh and Log View

Save

Refresh

500

Save auto-refresh and view settings

Default is ON

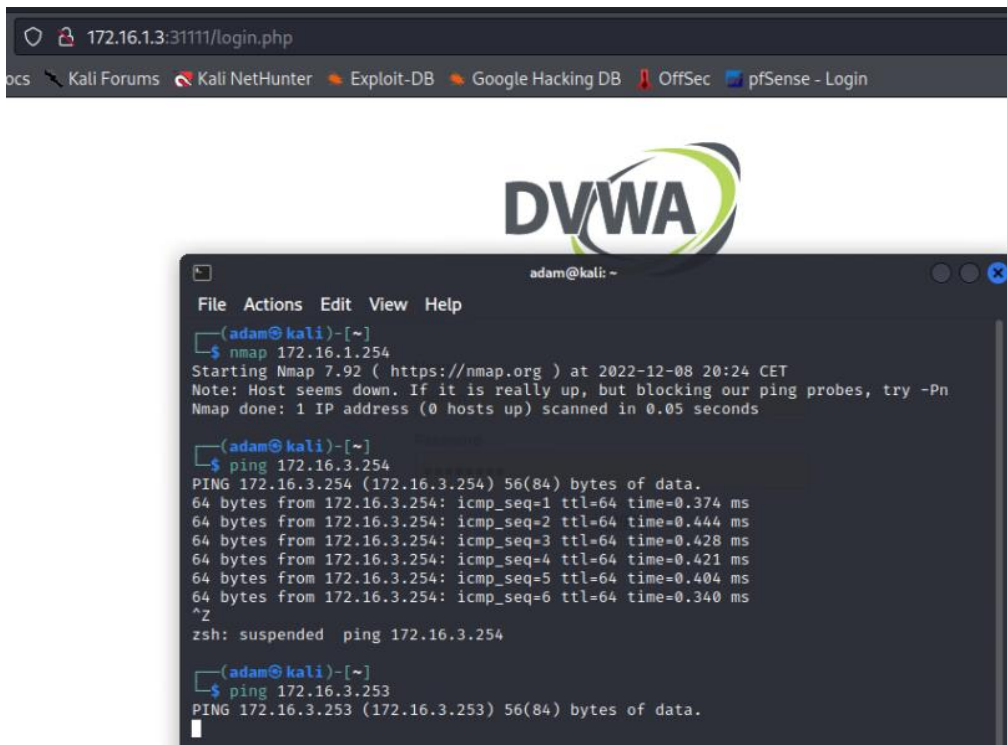
Number of blocked entries to view. Default is 500

Last 500 Hosts Blocked by Snort (only applicable to Legacy Blocking Mode Interfaces)

#	IP	Alert Descriptions and Event Times	Remove
1	172.16.3.5 	ET SCAN Suspicious inbound to MySQL port 3306 – 2022-12-09 09:45:31 ET SCAN Suspicious inbound to MSSQL port 1433 – 2022-12-09 09:46:19 ET SCAN Potential VNC Scan 5900-5920 – 2022-12-09 09:47:10 ET SCAN Suspicious inbound to PostgreSQL port 5432 – 2022-12-09 09:45:36 ET SCAN Suspicious inbound to Oracle SQL port 1521 – 2022-12-09 09:45:32 ET SCAN Potential VNC Scan 5800-5820 – 2022-12-09 09:45:46	

1 host IP address is currently being blocked Snort on Legacy Blocking Mode interfaces.

Als we kijken bij blocked hosts dan zien we de Ip er bij staan met allen activiteiten. Verder staat er de optie remove waarmee de Ip weer op het netwerk mag.



Hier is te zien hoe kali niet meer naar de router kan pingen. Waardoor kali ook niet de webserver kan bereiken.

SQL injectie

Een sql query is een stukje code die met een database interacteert. Zo kan een programmeur met sql query's data oproepen. Echter kunnen deze query's ook door hackers gebruikt worden om ongeoorloofde data te stelen.

zie de volgende query

admin'OR'=1'

//Uitleg query

User ID:

ID: admin'OR'1='1
First name: admin
Surname: admin

ID: admin'OR'1='1
First name: Gordon
Surname: Brown

ID: admin'OR'1='1
First name: Hack
Surname: Me

ID: admin'OR'1='1
First name: Pablo
Surname: Picasso

ID: admin'OR'1='1
First name: Bob
Surname: Smith

IDS

Om de sql query's te detecteren moet de volgende configuratie worden gemaakt.

1. Ga naar Snort Interfaces → click op potlood → DMZ rules → voeg de volgende regels toe

**alert tcp any any -> any 80 (msg: "Error Based SQL Injection Detected";
content: "%27" ; sid:100000011;)**

**alert tcp any any -> any 80 (msg: "Error Based SQL Injection Detected";
content: "22" ; sid:100000012;)**

**alert tcp any any -> any 80 (msg: "AND SQL Injection Detected";
content: "and" ; nocase; sid:100000060;)**

**alert tcp any any -> any 80 (msg: "OR SQL Injection Detected"; content:
"or" ; nocase; sid:100000061;)**

2022-12-13 09:54:04	⚠	0	TCP	172.16.1.4 🔍+	58826	213.136.12.213 🔍+	80	1:100000012 ⊕✖	Error Based SQL Injection Detected
2022-12-13 09:54:04	⚠	0	TCP	172.16.1.4 🔍+	58826	213.136.12.213 🔍+	80	1:100000061 ⊕✖	OR SQL Injection Detected

Hier is te zien hoe de snort regels de sql aanvallen detecteert.

[Snort Interfaces](#)
[Global Settings](#)
[Updates](#)
[Alerts](#)
[Blocked](#)
[Pass Lists](#)
[Suppress](#)
[IP Lists](#)
[SID Mgmt](#)
[Log Mgmt](#)
[Sync](#)

Blocked Hosts and Log View Settings

Blocked Hosts

Download

Clear

All blocked hosts will be saved

All blocked hosts will be removed

Refresh and Log View

Save

Refresh

500

Save auto-refresh and view settings

Default is ON

Number of blocked entries to view. Default is 500

Last 500 Hosts Blocked by Snort (only applicable to Legacy Blocking Mode interfaces)

#	IP	Alert Descriptions and Event Times	Remove
1	172.16.3.5 🔍	ET SCAN Suspicious inbound to MySQL port 3306 -- 2022-12-12 17:55:09 ET SCAN Suspicious inbound to MSSQL port 1433 -- 2022-12-12 17:55:38 ET SCAN Potential VNC Scan 5900-5920 -- 2022-12-12 17:56:24 ET SCAN Suspicious inbound to PostgreSQL port 5432 -- 2022-12-12 18:02:16 ET SCAN Suspicious inbound to Oracle SQL port 1521 -- 2022-12-12 17:56:19 ET SCAN Potential VNC Scan 5800-5820 -- 2022-12-12 17:56:04 OR SQL Injection Detected -- 2023-01-07 15:06:53	✖

Hier is te zien hoe de host geblokkeerd is door het uitvoeren van een sql injectie.

