

# Verslag labo 1

Nslookup naar het domein van UGent

```
PS C:\Users\gille> nslookup www.ugent.be
Server: ns1.hogent.be
Address: 193.190.173.1

Non-authoritative answer:
Name: www.UGent.be
Address: 157.193.43.50
```

Met een opgegeven server

```
PS C:\Users\gille> nslookup gent.be 8.8.8.8
Server: dns.google
Address: 8.8.8.8

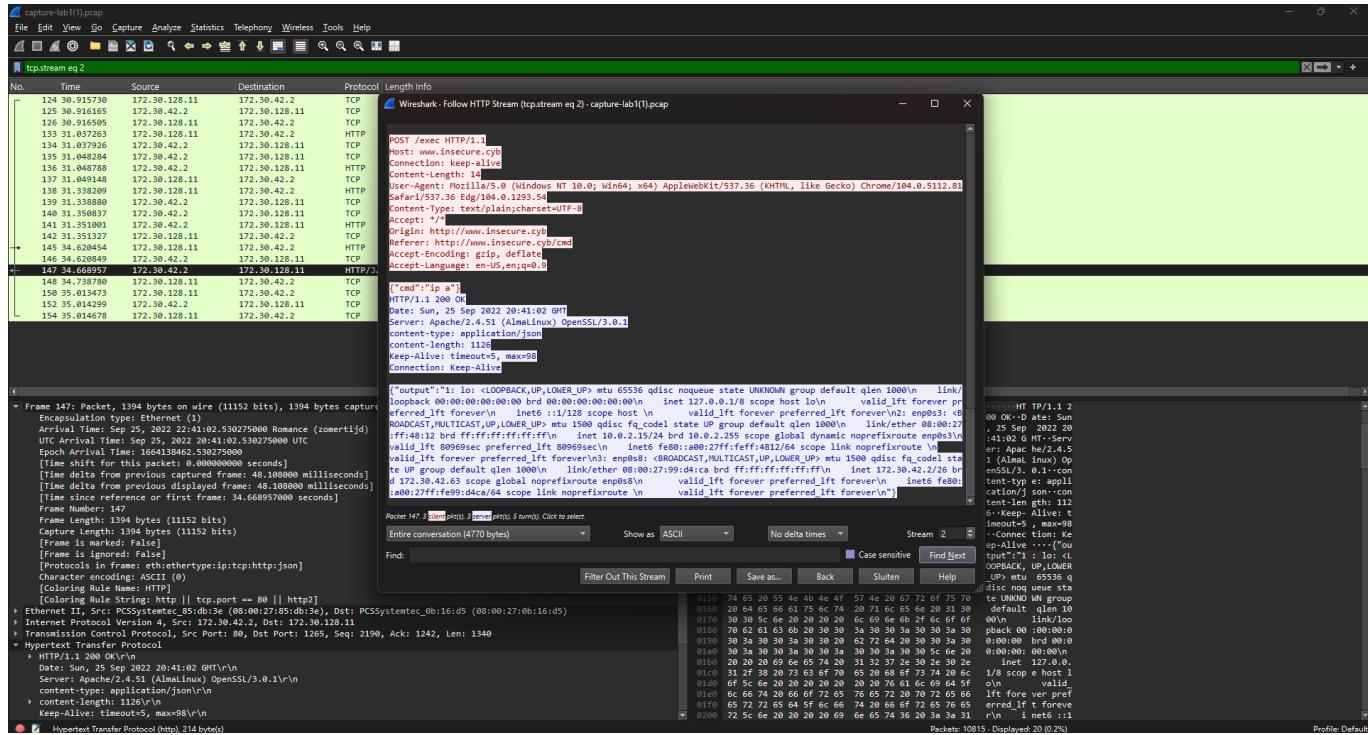
Non-authoritative answer:
Name: gent.be
Addresses: 2001:67c:1902:1305::138
           212.123.24.4
```

## Recap Wireshark

In cybersecurity and virtualization we got to know Wireshark. Most captures were relatively small in size. In a more realistic setting, a capture file will have a lot of packets, often unrelated to what you are searching for. Open the capture-lab1.pcap file and try to answer the following questions:

1. What layers of the OSI model are captured in this capturefile?  
Laag 2-4 en een beetje laag 7.
2. Take a look at the conversations. What do you notice? Er is heel veel communicatie tussen 172.30.128.10 en 172.30.42.2, voor de rest is er een beetje communicatie tussen andere adressen.
3. Take a look at the protocol hierarchy. What are the most "interesting" protocols listed here? Heel veel SSH verkeer, een paar windows gerelateerde protocollen zoals LDAP, kerberos. De rest is allemaal ARP, TCP en UDP.
4. Can you spot an SSH session that got established between 2 machines? List the 2 machines. Who was the SSH server and who was the client? What ports were used? Are these ports TCP or UDP? Client: 172.30.128.10 Server: 172.30.42.2 Port: 22, 37700 Protocol: TCP
5. Some cleartext data was transferred between two machines. Can you spot the data? Can you deduce what happened here? Ik zocht dit op door te filteren op http en zag dan een paar requests en

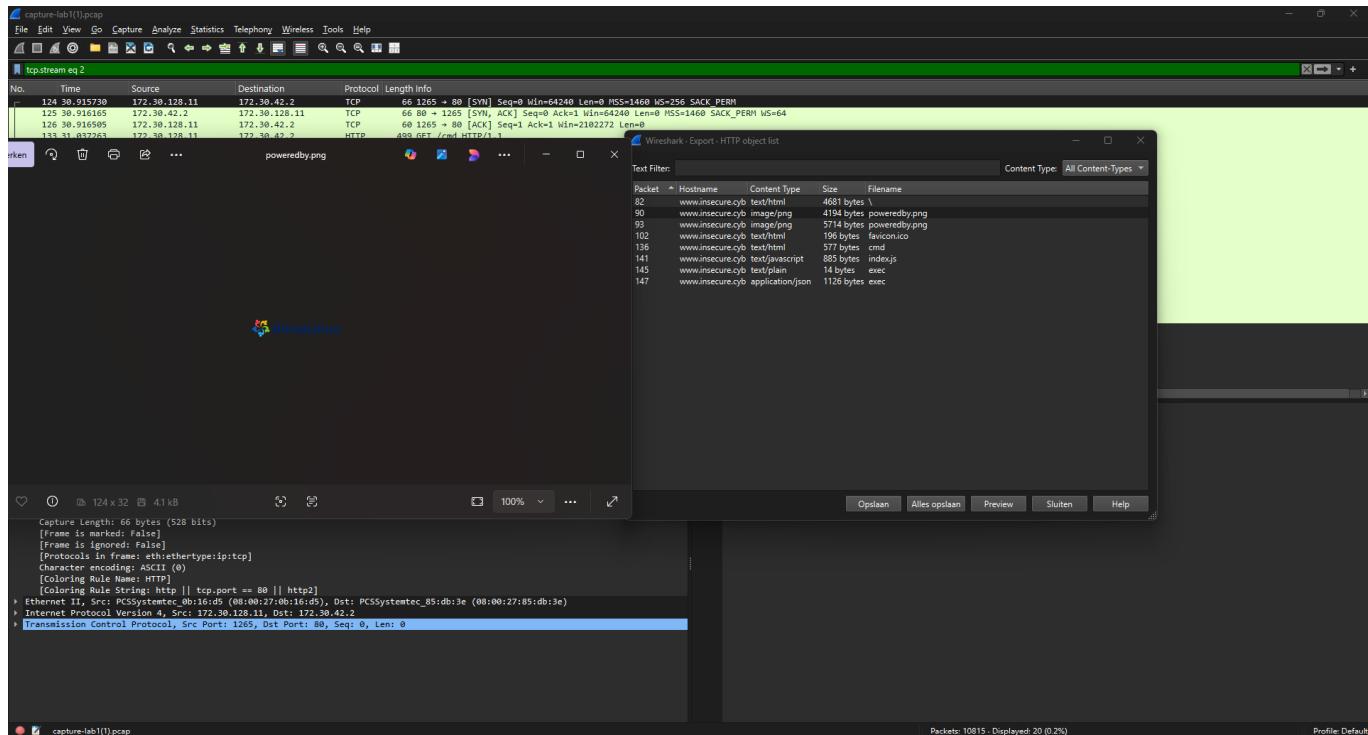
responses. Het ging hier om een webpagina die werd opgevraagd en een POST. Het was op de cmd pagina een ip a commando die werd uitgevoerd.



6. Someone used a specific way to transfer a png on the wire. Is it possible to export this png easily? Dit is zeker mogelijk. Er is gebruik gemaakt van HTTP om de png te transfereren dus deze zit in de capture.

De stappen hiervoor zijn: Navigeer naar Export: Ga in de menubalk bovenin naar:

- File (Bestand)
- Export Objects
- HTTP...



```
[vagrant@companyrouter ~]$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN
group default qlen 1000
    link/ether 08:00:27:e8:a5:e1 brd ff:ff:ff:ff:ff:ff
    altname enp0s3
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
default qlen 1000
    link/ether 08:00:27:26:49:5e brd ff:ff:ff:ff:ff:ff
    altname enp0s8
    inet 192.168.62.253/24 brd 192.168.62.255 scope global noprefixroute eth1
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe26:495e/64 scope link
        valid_lft forever preferred_lft forever
4: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
default qlen 1000
    link/ether 08:00:27:92:4c:3c brd ff:ff:ff:ff:ff:ff
    altname enp0s9
    inet 172.30.255.254/16 brd 172.30.255.255 scope global noprefixroute eth2
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe92:4c3c/64 scope link
        valid_lft forever preferred_lft forever
[vagrant@companyrouter ~]$ ip route show
default via 192.168.62.254 dev eth1 proto static metric 100
172.30.0.0/16 dev eth2 proto kernel scope link src 172.30.255.254 metric 101
192.168.62.0/24 dev eth1 proto kernel scope link src 192.168.62.253 metric 100
```

## Capture traffic using the CLI

### **Interfaces kiezen op de CompanyRouter**

Om dit te bepalen, kijken we met ip a op de companyrouter welke interfaces er zijn en welke IP-adressen ze hebben.

Voor het verkeer van de DNS naar naar het internet: we kiezen de interface die verbonden is met het WAN/ISP netwerk, bij mij is dit eth1 (192.168.62.253/24).

Van DNS naar Employee, we kiezen de interface die verbonden is met het interne netwerk, bij mij is dit eth3 (172.30.20.254/24).

### **Pings testen met tcpdump**

Test this out by pinging from employee to the companyrouter and from employee to the dns. Are you able to see all pings in tcpdump on the companyrouter?

We maken verbinding met employee via ssh en pingen de companyrouter en dns aan:

```
[vagrant@companyrouter ~]$ sudo tcpdump -i eth3 icmp
dropped privs to tcpdump
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
14:59:04.029148 IP 172.30.20.123 > companyrouter: ICMP echo request, id 2, seq 0,
length 64
14:59:04.029187 IP companyrouter > 172.30.20.123: ICMP echo reply, id 2, seq 0,
length 64
14:59:05.029222 IP 172.30.20.123 > companyrouter: ICMP echo request, id 2, seq 1,
length 64
14:59:05.029270 IP companyrouter > 172.30.20.123: ICMP echo reply, id 2, seq 1,
length 64
14:59:06.029253 IP 172.30.20.123 > companyrouter: ICMP echo request, id 2, seq 2,
length 64
14:59:06.029295 IP companyrouter > 172.30.20.123: ICMP echo reply, id 2, seq 2,
length 64
14:59:07.029340 IP 172.30.20.123 > companyrouter: ICMP echo request, id 2, seq 3,
length 64
14:59:07.029382 IP companyrouter > 172.30.20.123: ICMP echo reply, id 2, seq 3,
length 64
14:59:08.029399 IP 172.30.20.123 > companyrouter: ICMP echo request, id 2, seq 4,
length 64
14:59:08.029474 IP companyrouter > 172.30.20.123: ICMP echo reply, id 2, seq 4,
length 64
14:59:09.029476 IP 172.30.20.123 > companyrouter: ICMP echo request, id 2, seq 5,
length 64
14:59:09.029523 IP companyrouter > 172.30.20.123: ICMP echo reply, id 2, seq 5,
length 64
```

Van de DNS naar de employee zag ik niets in tcpdump. Dit is logisch want de pings gaan niet via de companyrouter maar rechtstreeks via het interne netwerk.

### Opslaan naar een bestand en analyseren

We kunnen een capture opslaan naar een bestand met de -w flag:

```
sudo tcpdump -i any -w capture_test.pcap
```

Om het bestand te analyseren, kopieerde ik het met scp naar de kali machine:

```
scp vagrant@192.168.62.253:~/home/vagrant/capture_test.pcap .
```

### SSH-verkeer wegfilteren

Wanneer je zelf via SSH op de router zit, vervult je eigen verbinding de output (elke letter die je typt is een pakketje). Je filtert dit uit met de not port optie:

```
sudo tcpdump -i any not port 22
```

## Alleen HTTP-verkeer van/naar de Webserver

Om alleen HTTP-verkeer van/naar de webserver te capturen, gebruik je het volgende commando:

```
sudo tcpdump -i any host 172.30.10.10 and port 80
[vagrant@companyrouter ~]$ sudo tcpdump -i any host 172.30.10.10 and port 80
tcpdump: data link type LINUX_SLL2
dropped privs to tcpdump
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144
bytes
15:24:20.273951 eth3  In  IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [S], seq 2222187697, win 64240, options [mss 1460,sackOK,TS val 3421474802 ecr 0,nop,wscale 5], length 0
15:24:20.273980 eth2  Out IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [S], seq 2222187697, win 64240, options [mss 1460,sackOK,TS val 3421474802 ecr 0,nop,wscale 5], length 0
15:24:20.274217 eth2  In  IP 172.30.10.10.http > 172.30.20.123.54104: Flags [S.], seq 4223126026, ack 2222187698, win 65160, options [mss 1460,sackOK,TS val 1348158462 ecr 3421474802,nop,wscale 7], length 0
15:24:20.274225 eth3  Out IP 172.30.10.10.http > 172.30.20.123.54104: Flags [S.], seq 4223126026, ack 2222187698, win 65160, options [mss 1460,sackOK,TS val 1348158462 ecr 3421474802,nop,wscale 7], length 0
15:24:20.274340 eth3  In  IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [.], ack 1, win 2008, options [nop,nop,TS val 3421474803 ecr 1348158462], length 0
15:24:20.274346 eth2  Out IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [.], ack 1, win 2008, options [nop,nop,TS val 3421474803 ecr 1348158462], length 0
15:24:20.274406 eth3  In  IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [P.], seq 1:85, ack 1, win 2008, options [nop,nop,TS val 3421474803 ecr 1348158462], length 84: HTTP: GET / HTTP/1.1
15:24:20.274409 eth2  Out IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [P.], seq 1:85, ack 1, win 2008, options [nop,nop,TS val 3421474803 ecr 1348158462], length 84: HTTP: GET / HTTP/1.1
15:24:20.274580 eth2  In  IP 172.30.10.10.http > 172.30.20.123.54104: Flags [.], ack 85, win 509, options [nop,nop,TS val 1348158463 ecr 3421474803], length 0
15:24:20.274592 eth3  Out IP 172.30.10.10.http > 172.30.20.123.54104: Flags [.], ack 85, win 509, options [nop,nop,TS val 1348158463 ecr 3421474803], length 0
15:24:20.274917 eth2  In  IP 172.30.10.10.http > 172.30.20.123.54104: Flags [P.], seq 1:467, ack 85, win 509, options [nop,nop,TS val 1348158463 ecr 3421474803], length 466: HTTP: HTTP/1.1 301 Moved Permanently
15:24:20.274927 eth3  Out IP 172.30.10.10.http > 172.30.20.123.54104: Flags [P.], seq 1:467, ack 85, win 509, options [nop,nop,TS val 1348158463 ecr 3421474803], length 466: HTTP: HTTP/1.1 301 Moved Permanently
15:24:20.275031 eth3  In  IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [.], ack 467, win 2003, options [nop,nop,TS val 3421474803 ecr 1348158463], length 0
```

```

15:24:20.275035 eth2 Out IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [ . ],
ack 467, win 2003, options [nop,nop,TS val 3421474803 ecr 1348158463], length 0
15:24:20.275256 eth3 In IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [F.],
seq 85, ack 467, win 2003, options [nop,nop,TS val 3421474804 ecr 1348158463],
length 0
15:24:20.275261 eth2 Out IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [F.],
seq 85, ack 467, win 2003, options [nop,nop,TS val 3421474804 ecr 1348158463],
length 0
15:24:20.275398 eth2 In IP 172.30.10.10.http > 172.30.20.123.54104: Flags [F.],
seq 467, ack 86, win 509, options [nop,nop,TS val 1348158464 ecr 3421474804],
length 0
15:24:20.275401 eth3 Out IP 172.30.10.10.http > 172.30.20.123.54104: Flags [F.],
seq 467, ack 86, win 509, options [nop,nop,TS val 1348158464 ecr 3421474804],
length 0
15:24:20.275487 eth3 In IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [ . ],
ack 468, win 2003, options [nop,nop,TS val 3421474804 ecr 1348158464], length 0
15:24:20.275495 eth2 Out IP 172.30.20.123.54104 > 172.30.10.10.http: Flags [ . ],
ack 468, win 2003, options [nop,nop,TS val 3421474804 ecr 1348158464], length 0

```

Interfaces companyrouter:

eth0: down → geen gebruik.

eth1: 192.168.62.253/24 → default gateway gaat via 192.168.62.254. Dit is dus de WAN/ISP-kant.

eth2: 172.30.255.254/16 → intern netwerk. Hier hangen employee, dns, web op.

What did you have to configure on your red machine to have internet and to properly ping the web machine (is the ping working on IP only or also on hostname)? De default gateway moest ingesteld worden op 192.168.62.254. Daarnaast had ik ook nog een probleem met de DNS. Dit viel op te lossen door:

```
echo "nameserver 1.1.1.1" | sudo tee /etc/resolv.conf
```

Hierna kon ik alles bereiken vanop de kali.

Bij het uitvoeren van een nmap zien we alle routers staan.

```

—(vagrant㉿red)-[~]
└$ nmap -sn 192.168.62.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-06 08:42 EDT
Nmap scan report for 192.168.62.0
Host is up (0.00022s latency).
MAC Address: 0A:00:27:00:00:0A (Unknown)
Nmap scan report for 192.168.62.42
Host is up (0.0012s latency).
MAC Address: 08:00:27:80:13:EC (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.62.253
Host is up (0.00029s latency).
MAC Address: 08:00:27:26:49:5E (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.62.254

```

```
Host is up (0.00039s latency).
MAC Address: 08:00:27:46:85:65 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.62.110
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 2.32 seconds
```

## DNS zone transfer

```
└─(vagrant㉿red)-[~]
└─$ dig axfr @172.30.0.4 cybersec.internal

; <>> DiG 9.20.11-4+b1-Debian <>> axfr @172.30.0.4 cybersec.internal
; (1 server found)
;; global options: +cmd
cybersec.internal.    86400   IN      SOA     dns.cybersec.internal.
admin.cybersec.internal. 2023092301 3600 1800 1209600 86400
cybersec.internal.    86400   IN      NS      dns.cybersec.internal.
txt.at.cybersec.internal. 86400  IN      TXT     "Greetings from CSA team!"
dns.cybersec.internal. 86400   IN      A       172.30.0.4
www.cybersec.internal. 86400   IN      A       172.30.0.10
cybersec.internal.    86400   IN      SOA     dns.cybersec.internal.
admin.cybersec.internal. 2023092301 3600 1800 1209600 86400
;; Query time: 0 msec
;; SERVER: 172.30.0.4#53(172.30.0.4) (TCP)
;; WHEN: Mon Oct 06 09:08:23 EDT 2025
;; XFR size: 6 records (messages 1, bytes 267)
```

We maken verbinding met de DNS server via ssh en we zoeken uit wat voor DNS server er op deze machine draait:

```
dns:~$ sudo netstat -tulpn | grep :53
tcp        0      0 172.30.0.4:53          0.0.0.0:*          LISTEN
1779/named
tcp        0      0 127.0.0.1:53          0.0.0.0:*          LISTEN
1779/named
tcp        0      0 ::1:53                ::::*               LISTEN
1779/named
tcp        0      0 fe80::a00:27ff:fed2:42d:53 ::::*             LISTEN
1779/named
udp        0      0 172.30.0.4:53          0.0.0.0:*          LISTEN
1779/named
udp        0      0 127.0.0.1:53          0.0.0.0:*          LISTEN
1779/named
udp        0      0 ::1:53                ::::*               LISTEN
1779/named
udp        0      0 fe80::a00:27ff:fed2:42d:53 ::::*             LISTEN
1779/named
```

Dit is BIND. Standaard staat zone transfers open voor iedereen.

We controleren de configuratie:

```
sudo cat /etc/bind/named.conf
options {
    directory "/var/bind";
    allow-transfer { any; };      # Allow zone transfers from any machine
    listen-on { any; };           # Listen on all interfaces
    listen-on-v6 { any; };        # Listen on all IPv6 interfaces
    recursion yes;               # Enable recursion for forwarding other queries
    forwarders {
        192.168.62.254;          # Forward all other queries to this DNS server
    };
    allow-query { any; };         # Allow queries from any IP
};

zone "cybersec.internal" {
    type master;
    file "/var/bind/cybersec.internal";
    allow-transfer { any; };    # Allow zone transfers from any machine
```

We veranderen de configuratie van dit:

```
options {
    directory "/var/bind";
    allow-transfer { any; };    # ← Dit moeten we aanpassen
    listen-on { any; };
    listen-on-v6 { any; };
    recursion yes;
    forwarders {
        192.168.62.254;
    };
    allow-query { any; };
};

zone "cybersec.internal" {
    type master;
    file "/var/bind/cybersec.internal";
    allow-transfer { any; };   # ← EN DIT OOK
};
```

naar dit:

```
options {
    directory "/var/bind";
    allow-transfer { none; };   # ← BLOKKEER zone transfers
    listen-on { any; };
```

```

listen-on-v6 { any; };
recursion yes;
forwarders {
    192.168.62.254;
};
allow-query { any; };
};

zone "cybersec.internal" {
    type master;
    file "/var/bind/cybersec.internal";
    allow-transfer { none; }; # ← BLOKKER zone transfers
};

```

Herstart de DNS server:

```

sudo named-checkconf
sudo /etc/init.d/named restart

```

## Verslag labo 2 firewalls

---

Op het domein: <http://www.cybersec.internal/cmd> draait een terminal sessie.

Commandos die ik heb uitgevoerd met hun resultaten:

### 1. id

- Resultaat: `uid=0(root) gid=0(root) groups=0(root)`  
`context=system_u:system_r:unconfined_service_t:s0`

### 2. uname -a

- Resultaat: `Linux web 5.14.0-570.12.1.el9_6.x86_64 #1 SMP PREEMPT_DYNAMIC Tue May 13 06:11:55 EDT 2025 x86_64 x86_64 x86_64 GNU/Linux`

### 3. ls -la

- Resultaat: Lijst van bestanden en mappen in de huidige directory met gedetailleerde informatie.

```

total 28
dr-xr-xr-x. 19 root      root      250 Sep 26 13:45 .
dr-xr-xr-x. 19 root      root      250 Sep 26 13:45 ..
dr-xr-xr-x.  2 root      root       6 Oct  2 2024 afs
lrwxrwxrwx.  1 root      root       7 Oct  2 2024 bin -> usr/bin
dr-xr-xr-x.  5 root      root     4096 May 22 13:03 boot
drwxr-xr-x. 18 root      root    3120 Oct 14 10:38 dev
drwxr-xr-x. 102 root     root   8192 Oct 14 10:38 etc
drwxr-xr-x.  3 root      root     21 May 22 13:04 home
lrwxrwxrwx.  1 root      root       7 Oct  2 2024 lib -> usr/lib
lrwxrwxrwx.  1 root      root       9 Oct  2 2024 lib64 -> usr/lib64
drwxr-xr-x.  2 root      root       6 Oct  2 2024 media
drwxr-xr-x.  2 root      root       6 Oct  2 2024 mnt

```

```
drwxr-xr-x.  5 root      root      73 Sep 26 14:18 opt
dr-xr-xr-x. 196 root      root      0 Oct 14 10:38 proc
dr-xr-x---.  4 root      root     117 Sep 26 14:17 root
drwxr-xr-x.  28 root      root     900 Oct 14 10:38 run
lrwxrwxrwx.   1 root      root      8 Oct  2 2024 sbin -> usr/sbin
drwxr-xr-x.   2 root      root      6 Oct  2 2024 srv
dr-xr-xr-x.  13 root      root      0 Oct 14 10:38 sys
drwxrwxrwt.  12 root      root    4096 Oct 14 10:38 tmp
drwxr-xr-x.  12 root      root    144 May 22 13:01 usr
drwxrwxrwx.   1 vagrant  vagrant 4096 Oct  6 12:37 vagrant
drwxr-xr-x.  20 root      root    4096 Sep 26 14:17 var
```

#### 4. cat /etc/passwd

- Resultaat: Lijst van gebruikersaccounts op het systeem.

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:65534:65534:Kernel Overflow User:/:/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
tss:x:59:59:Account used for TPM access:/:/usr/sbin/nologin
sssd:x:998:998:User for sssd:/:/sbin/nologin
chrony:x:997:997:chrony system user:/var/lib/chrony:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/usr/share/empty.sshd:/usr/sbin/nologin
vagrant:x:1000:1000::/home/vagrant:/bin/bash
rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:/sbin/nologin
polkitd:x:996:995:User for polkitd:/:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
tcpdump:x:72:72:::/sbin/nologin
vboxadd:x:995:1:::/var/run/vboxadd:/bin/false
rtkit:x:172:172:RealtimeKit:/:/sbin/nologin
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
pipewire:x:994:992:PipeWire System Daemon:/run/pipewire:/usr/sbin/nologin
geoclue:x:993:991:User for geoclue:/var/lib/geoclue:/sbin/nologin
flatpak:x:992:990:Flatpak system helper:/:/usr/sbin/nologin
```

#### Nmap default scan

Ik heb een nmap default scan uitgevoerd op alle machines in het netwerk met het commando:

```
└─(vagrant㉿red)-[~]
└─$ nmap -v 172.30.20.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-04 04:44 EST
Initiating Ping Scan at 04:44
Scanning 256 hosts [4 ports/host]
Completed Ping Scan at 04:45, 3.55s elapsed (256 total hosts)
Initiating Parallel DNS resolution of 4 hosts. at 04:45
Completed Parallel DNS resolution of 4 hosts. at 04:45, 0.00s elapsed
Nmap scan report for 172.30.20.0 [host down]
Nmap scan report for 172.30.20.1 [host down]
Nmap scan report for 172.30.20.2 [host down]
Nmap scan report for 172.30.20.3 [host down]
Nmap scan report for 172.30.20.255 [host down]
Initiating SYN Stealth Scan at 04:45
Scanning 4 hosts [1000 ports/host]
Discovered open port 3306/tcp on 172.30.20.15
Discovered open port 53/tcp on 172.30.20.4
Discovered open port 22/tcp on 172.30.20.15
Discovered open port 22/tcp on 172.30.20.123
Discovered open port 111/tcp on 172.30.20.254
Discovered open port 22/tcp on 172.30.20.4
Discovered open port 22/tcp on 172.30.20.254
Discovered open port 2222/tcp on 172.30.20.254
Completed SYN Stealth Scan against 172.30.20.15 in 0.16s (3 hosts left)
Completed SYN Stealth Scan against 172.30.20.123 in 0.16s (2 hosts left)
Completed SYN Stealth Scan against 172.30.20.4 in 0.17s (1 host left)
Completed SYN Stealth Scan at 04:45, 0.17s elapsed (4000 total ports)
Nmap scan report for 172.30.20.4
Host is up (0.00033s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
53/tcp    open  domain

Nmap scan report for 172.30.20.15
Host is up (0.00064s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
3306/tcp  open  mysql

Nmap scan report for 172.30.20.123
Host is up (0.00065s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap scan report for 172.30.20.254
Host is up (0.00020s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
```

```
2222/tcp open EtherNetIP-1

Read data files from: /usr/share/nmap
Nmap done: 256 IP addresses (4 hosts up) scanned in 3.82 seconds
          Raw packets sent: 6021 (252.748KB) | Rcvd: 4012 (160.620KB)
```

Banner grab scan:

```
└─(vagrant㉿red)-[~]
└─$ nmap -sV -p 22,53,80,3306,8000 172.30.20.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-04 04:48 EST
Nmap scan report for 172.30.20.4
Host is up (0.00062s latency).
```

PORT	STATE	SERVICE	VERSION
22/tcp	open	ssh	OpenSSH 9.3 (protocol 2.0)
53/tcp	open	domain	ISC BIND 9.18.37
80/tcp	closed	http	
3306/tcp	closed	mysql	
8000/tcp	closed	http-alt	

```
Nmap scan report for 172.30.20.15
Host is up (0.00046s latency).
```

PORT	STATE	SERVICE	VERSION
22/tcp	open	ssh	OpenSSH 9.3 (protocol 2.0)
53/tcp	closed	domain	
80/tcp	closed	http	
3306/tcp	open	mysql	MariaDB 5.5.5-10.11.11
8000/tcp	closed	http-alt	

```
Nmap scan report for 172.30.20.123
Host is up (0.00068s latency).
```

PORT	STATE	SERVICE	VERSION
22/tcp	open	ssh	OpenSSH 9.3 (protocol 2.0)
53/tcp	closed	domain	
80/tcp	closed	http	
3306/tcp	closed	mysql	
8000/tcp	closed	http-alt	

```
Nmap scan report for 172.30.20.254
Host is up (0.00040s latency).
```

PORT	STATE	SERVICE	VERSION
22/tcp	open	ssh	OpenSSH 9.2p1 Debian 2+deb12u3 (protocol 2.0)
53/tcp	closed	domain	
80/tcp	closed	http	
3306/tcp	closed	mysql	
8000/tcp	closed	http-alt	

Service Info: OS: Linux; CPE: cpe:/o:linux:linux\_kernel

```
Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 256 IP addresses (4 hosts up) scanned in 9.87 seconds
```

Scan op de database server:

```
└──(vagrant㉿red)-[~]
└─$ nmap -sV -sC 172.30.20.15
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-04 04:47 EST
Nmap scan report for 172.30.20.15
Host is up (0.00068s latency).

Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 9.3 (protocol 2.0)
| ssh-hostkey:
|   256 a6:f5:68:ed:ff:72:b1:c8:50:a0:62:ad:57:fa:08:2d (ECDSA)
|_  256 75:99:6b:07:14:ee:04:6b:20:a8:05:60:32:14:03:d8 (ED25519)
3306/tcp  open  mysql   MariaDB 5.5.5-10.11.11
| mysql-info:
|   Protocol: 10
|   Version: 5.5.5-10.11.11-MariaDB
|   Thread ID: 4
|   Capabilities flags: 63486
|   Some Capabilities: FoundRows, LongColumnFlag, IgnoreSpaceBeforeParenthesis,
ODBCClient, Speaks41ProtocolNew, Support41Auth, Speaks41ProtocolOld,
InteractiveClient, DontAllowDatabaseTableColumn, ConnectWithDatabase,
SupportsTransactions, IgnoreSigpipes, SupportsLoadDataLocal, SupportsCompression,
SupportsMultipleResults, SupportsMultipleStatements, SupportsAuthPlugins
|   Status: Autocommit
|   Salt: 48^San)|Vy!2w$K+6l@9
|_  Auth Plugin Name: mysql_native_password

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 5.79 seconds
```

Er draait een MySQL 5.5.5-10.11.11-MariaDB server op de database server.

## Brute force van de database server

Try to search for a nmap script to brute-force the database. Another (even easier tool) is called hydra (<https://github.com/vanhauser-thc/thc-hydra>). Search online for a good wordlist. For example "rockyou" or <https://github.com/danielmiessler/SecLists>. We suggest to try the default username of the database software and attack the database machine. Another interesting username worth a try is "toor".

Ik gebruikte het volgende commando:

```
└──(vagrant㉿red)-[~]
└─$ nmap -p 3306 --script mysql-brute --script-args
```

```
userdb=/usr/share/wordlists/nmap.lst,passdb=/usr/share/wordlists/rockyou.txt
172.30.20.15
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-04 04:55 EST
Nmap scan report for 172.30.20.15
Host is up (0.00077s latency).

PORT      STATE SERVICE
3306/tcp   open  mysql
| mysql-brute:
|   Accounts: No valid accounts found
|   Statistics: Performed 201204 guesses in 247 seconds, average tps: 840.2
|_  ERROR: The service seems to have failed or is heavily firewalled...

Nmap done: 1 IP address (1 host up) scanned in 263.89 seconds
```

Het lukt niet om in te breken op de database server met de gebruikte lijsten.

Ik probeerde ook met hydra:

```
hydra -l root -P /usr/share/wordlists/rockyou.txt 172.30.20.15 mysql
```

Proberen om te ssh verbinden van de red machine naar een andere machine met vagrant/vagrant:

```
—(vagrant@red)-[~]
└$ ssh vagrant@192.168.62.42
The authenticity of host '192.168.62.42 (192.168.62.42)' can't be established.
ED25519 key fingerprint is SHA256:tVgSkWqegBlTs+mcUNdtVa1PitC6LZF18Qu921xy9cw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.62.42' (ED25519) to the list of known hosts.
vagrant@192.168.62.42's password:
Permission denied, please try again.
vagrant@192.168.62.42's password:
Last failed login: Mon Oct 20 12:20:43 UTC 2025 from 192.168.62.110 on ssh:notty
There was 1 failed login attempt since the last successful login.
Last login: Mon Oct  6 07:26:42 2025 from 192.168.62.254
[vagrant@db ~]$
```

Dit is mogelijk.

Kijken welke versie er op de webserver draait:

```
—(vagrant@red)-[~]
└$ nmap -sV -p80,443 172.30.10.10
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-04 05:02 EST
Nmap scan report for 172.30.10.10
Host is up (0.00084s latency).
```

```
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.62 ((AlmaLinux) OpenSSL/3.5.1)
443/tcp   open  ssl/http Apache httpd 2.4.62 ((AlmaLinux) OpenSSL/3.5.1)

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.47 seconds
```

De webserver draait Apache httpd 2.4.62 op (AlmaLinux).

Gebruik -sC optie met nmap, wat is het?

-sC voert de default NSE scripts uit (Nmap Scripting Engine) een set van scripts die vaak basale informatie en checks uitvoeren (vulnerability checks, banners, http-enum, etc.). Het is handig voor een snelle extra laag informatieverzameling.

```
└─(vagrant㉿red)-[~]
└─$ nmap -sV -sC -p80,443 172.30.10.10
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-04 05:09 EST
Nmap scan report for 172.30.10.10
Host is up (0.00079s latency).

PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.62 ((AlmaLinux) OpenSSL/3.5.1)
|_http-server-header: Apache/2.4.62 (AlmaLinux) OpenSSL/3.5.1
|_http-title: Did not follow redirect to https://www.cybersec.internal/
443/tcp   open  ssl/http Apache httpd 2.4.62 ((AlmaLinux) OpenSSL/3.5.1)
|_http-title: Welcome to Example Test Environment
| http-methods:
|_ Potentially risky methods: TRACE
| ssl-cert: Subject: commonName=www.cybersec.internal
| Subject Alternative Name: DNS:www.cybersec.internal,
DNS:services.cybersec.internal
| Not valid before: 2025-12-28T11:00:50
|_Not valid after: 2028-04-01T11:00:50
|_http-server-header: Apache/2.4.62 (AlmaLinux) OpenSSL/3.5.1

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.71 seconds
```

De output toont dat de webserver mogelijk risicovolle HTTP-methoden zoals TRACE toestaat, wat een potentieel beveiligingsrisico kan zijn.

## Netwerksegmentatie

Wat wordt bedoeld met de term attack vector?

Een attack vector is het pad of de methode waarmee een aanvaller toegang kan krijgen tot een systeem of netwerk. Door het netwerk op te splitsen in segmenten en verkeer tussen die zones te beperken, verklein je het aantal mogelijke aanvalspaden.

Is er al network segmentation gedaan op het huidige (interne) bedrijfennetwerk?

Nee, nog niet. Momenteel bevinden alle interne hosts (web, database, dns, employee) zich in hetzelfde subnet 172.30.0.0/16 zonder filtering of zones. Ook is het “fake internet” rechtstreeks verbonden met het bedrijf via de companyrouter, zonder firewallregels die inkomend verkeer beperken.

Wat is een DMZ en welke machines horen daarin?

Een DMZ (Demilitarized Zone) is een netwerkzone tussen het interne LAN en het internet. Ze bevat systemen die zowel door interne gebruikers als externe bezoekers moeten kunnen worden bereikt.

Client ↔ server communicatie kan geblokkeerd worden door de firewall. Bijvoorbeeld:

De webserver kan geen verbinding meer maken met de database.

Interne werknemers kunnen geen DNS-resolutie meer doen als de firewall te streng is.

Configuratie van de companyrouter met nieuwe subnetten voor het intern LAN en de firewallregels:

```
sudo tee /etc/nftables.conf > /dev/null << 'EOF'
#!/usr/sbin/nft -f

flush ruleset

#
# Zone definitions
#
define WAN = "eth0"
define EXTERNAL = "eth1"
define DMZ_IF = "eth2"
define INT_IF = "eth3"

define fake_internet = 192.168.62.0/24
define dmz_net = 172.30.10.0/24
define intranet_net = 172.30.20.0/24

# Specifieke IP's voor strakkere beveiliging
define webserver = 172.30.10.10
define dns_server = 172.30.20.4

table inet filter {
    chain input {
        type filter hook input priority 0; policy drop;

        iif lo accept
        ct state established,related accept

        # ICMP toestaan (handig voor troubleshooting)
        ip protocol icmp accept
    }
}
```

```

# SSH beheer (beperkt tot Intranet en Kali voor testen)
ip saddr { $intranet_net, 192.168.62.110 } tcp dport 22 accept
}

chain forward {
    type filter hook forward priority 0; policy drop;

    # 1. Toestaan van bestaande verbindingen (Cruciaal!)
    ct state established,related accept

    # 2. Intranet -> WAN & External (Internet toegang)
    iifname $INT_IF oifname { $WAN, $EXTERNAL } accept

    # 3. Intranet -> DMZ (Volledig beheer)
    iifname $INT_IF oifname $DMZ_IF accept

    # 4. DMZ -> Intranet (Beperk dit! Meestal alleen naar DB of SIEM)
    # Voor nu conform jouw script (open), maar overweeg dit te beperken:
    iifname $DMZ_IF oifname $INT_IF accept

    # 5. Fake internet -> DMZ Webserver
    ip saddr $fake_internet ip daddr $webserver tcp dport { 80, 443, 8000,
9200 } accept
        ip saddr $fake_internet ip daddr $webserver icmp type echo-request accept

    # 6. Fake internet -> Specifieke DNS server (Poort 53)
    ip saddr $fake_internet ip daddr $dns_server udp dport 53 accept
    ip saddr $fake_internet ip daddr $dns_server tcp dport 53 accept

    # 7. DMZ naar buiten (voor updates etc.)
    iifname $DMZ_IF oifname { $WAN, $EXTERNAL } accept
}

chain output {
    type filter hook output priority 0; policy accept;
}
}

table ip nat {
    chain postrouting {
        type nat hook postrouting priority 100;

        # Masquerade alles wat naar het internet (WAN/External) gaat
        oifname { $WAN, $EXTERNAL } masquerade
    }
}
EOF

```

ik herstart de nftables service:

```
sudo systemctl enable nftables
sudo systemctl restart nftables
sudo nft list ruleset
```

Ook moest ik de router nog de nieuwe subnetten meegeven:

```
[vagrant@companyrouter ~]$
sudo ip addr add 172.30.10.254/24 dev eth2
sudo ip addr add 172.30.20.254/24 dev eth2
```

Ook ipv4 forwarding moest nog ingeschakeld worden.

```
sudo sysctl -w net.ipv4.ip_forward=1
echo "net.ipv4.ip_forward=1" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
```

Nu teste ik de connectiviteit van de verschillende hosts met een scriptje:

Vanaf de router:

Toestel	IP-adres	Status
employee	172.30.20.123	[ BEREIKBAAR ]
companyrouter DMZ	172.30.10.254	[ BEREIKBAAR ]
db	172.30.20.15	[ BEREIKBAAR ]
webserver	172.30.10.10	[ BEREIKBAAR ]
remote-employee	172.10.10.123	[ BEREIKBAAR ]
dns	172.30.20.4	[ BEREIKBAAR ]
companyrouter INT	172.30.20.254	[ BEREIKBAAR ]
SIEM	172.30.20.50	[ MISLUKT ]
isp router	192.168.62.254	[ BEREIKBAAR ]
kali	192.168.62.110	[ BEREIKBAAR ]
homeroouter	192.168.62.42	[ BEREIKBAAR ]
companyrouter	192.168.62.253	[ BEREIKBAAR ]
Windows client	172.30.20.51	[ MISLUKT ]

Vanaf de webserver (DMS):

Toestel	IP-adres	Status
employee	172.30.20.123	[ BEREIKBAAR ]
companyrouter DMZ	172.30.10.254	[ BEREIKBAAR ]
db	172.30.20.15	[ BEREIKBAAR ]
webserver	172.30.10.10	[ BEREIKBAAR ]
remote-employee	172.10.10.123	[ BEREIKBAAR ]
dns	172.30.20.4	[ BEREIKBAAR ]
companyrouter INT	172.30.20.254	[ BEREIKBAAR ]
SIEM	172.30.20.50	[ MISLUKT ]
isp router	192.168.62.254	[ BEREIKBAAR ]
kali	192.168.62.110	[ BEREIKBAAR ]
homeroouter	192.168.62.42	[ BEREIKBAAR ]
companyrouter	192.168.62.253	[ BEREIKBAAR ]
Windows client	172.30.20.51	[ MISLUKT ]

Vanaf de database server (Internal LAN):

```
database:~$ ./test.sh
=====
      Netwerk Connectiviteit Rapport - database
=====

Toestel           | IP-adres   | Status
-----|-----|-----
employee          | 172.30.20.123 | [ BEREIKBAAR ]
companyrouter DMZ | 172.30.10.254 | [ BEREIKBAAR ]
db                | 172.30.20.15  | [ BEREIKBAAR ]
webserver         | 172.30.10.10  | [ BEREIKBAAR ]
remote-employee   | 172.10.10.123 | [ BEREIKBAAR ]
dns               | 172.30.20.4   | [ BEREIKBAAR ]
companyrouter INT | 172.30.20.254 | [ BEREIKBAAR ]
SIEM              | 172.30.20.50  | [ MISLUKT ]
isp router        | 192.168.62.254 | [ BEREIKBAAR ]
kali              | 192.168.62.110 | [ BEREIKBAAR ]
homerouter        | 192.168.62.42  | [ BEREIKBAAR ]
companyrouter     | 192.168.62.253 | [ BEREIKBAAR ]
Windows client    | 172.30.20.51  | [ MISLUKT ]
-----|-----|-----

database:~$
```

De SIEM en windows stonden uit tijdens de tests ivm resources.

Poortstatussen in nmap

nmap -p 22,80,666 172.30.10 -sV -Pn gaf:

22/tcp — open (OpenSSH 8.7)

80/tcp — open (Apache httpd 2.4.62)

666/tcp — closed (geen service luistert; host stuurt RST)

1. Wat betekent dit

open: er luistert een service die reageert op connecties (SYN → SYN/ACK). Nmap kan conversatie afhandelen en identificeert de service (SSH, HTTP).

closed: het host-systeem is bereikbaar en reageert, maar er luistert geen service op die poort; de host stuurt een RST. Nmap weet dus zeker dat poort gesloten is.

filtered (wat je eerder zag): er komt geen antwoord terug — meestal omdat een firewall het pakket dropt; nmap kan niet vaststellen of er een service luistert.

# Verslag labo 3: Secure Shell (SSH)

---

## SSH Client config

Ik begon met het kopiëren van de publieke sleutel van mijn laptop naar alle verschillende VMs met het commando: type \$env:USERPROFILE.ssh\id\_rsa.pub | ssh vagrant@172.30.10.10 "cat >> .ssh/authorized\_keys"

### What files are transferred to what machines?

De publieke sleutel van mijn laptop (~/.ssh/id\_rsa.pub) is gekopieerd naar de authorized\_keys bestanden van alle machines in het netwerk (webserver, database, dns, employee workstation, companyrouter).

De private key (~/.ssh/id\_rsa) blijft op mijn laptop staan en wordt niet gekopieerd. Als deze sleutel zou lekken, zou iedereen die deze heeft toegang kunnen krijgen tot de machines waar de publieke sleutel is toegevoegd.

```
ssh-copy-id -i ~/.ssh/id_rsa.pub vagrant@172.30.20.50
cat ~/.ssh/id_rsa.pub | ssh vagrant@172.30.10.10 "mkdir -p ~/.ssh && chmod 700
~/ssh && cat >> ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys"
cat ~/.ssh/id_rsa.pub | ssh vagrant@172.30.20.4 "mkdir -p ~/.ssh && chmod 700
~/ssh && cat >> ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys"
cat ~/.ssh/id_rsa.pub | ssh vagrant@172.30.20.15 "mkdir -p ~/.ssh && chmod 700
~/ssh && cat >> ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys"
cat ~/.ssh/id_rsa.pub | ssh vagrant@172.30.20.123 "mkdir -p ~/.ssh && chmod 700
~/ssh && cat >> ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys"
```

Opmerking: later heb ik ook de public key van de companyrouter zelf gekopieerd naar dezelfde hosts zodat de router zonder wachtwoord naar de interne machines kan inloggen.

In de ssh/config van mijn windows laptop heb ik het volgende gezet zodat ik niet van alle machines het ip elke keer moest opzoeken:

```
# === Bastion / companyrouter ===
Host companyrouter
  HostName 192.168.62.253
  User vagrant
  IdentityFile ~/.ssh/id_rsa
  Port 22
  ServerAliveInterval 60
  ForwardAgent no

# === DMZ / Webserver (via bastion) ===
Host web
  HostName 172.30.10.10
  User vagrant
  IdentityFile ~/.ssh/id_rsa
  ProxyJump companyrouter

# === Database (via bastion) ===
```

```
Host db
  HostName 172.30.20.15
  User vagrant
  IdentityFile ~/.ssh/id_rsa
  ProxyJump companyrouter

# === DNS (via bastion) ===
Host dns
  HostName 172.30.20.4
  User vagrant
  IdentityFile ~/.ssh/id_rsa
  ProxyJump companyrouter

# === Employee workstation (via bastion) ===
Host employee1
  HostName 172.30.20.123
  User vagrant
  IdentityFile ~/.ssh/id_rsa
  ProxyJump companyrouter

# === Kali (direct op 192.168.62.x netwerk) ===
Host kali
  HostName 192.168.62.110
  User vagrant
  IdentityFile ~/.ssh/id_rsa

# === ISP router (direct) ===
Host isprouter
  HostName 192.168.62.254
  User admin      # pas aan als gebruiker anders is (root, admin, etc.)
  IdentityFile ~/.ssh/id_rsa

# === Shortcut: directe sessie naar companyrouter ===
Host companyrouter-ssh
  HostName 192.168.62.253
  User vagrant
  IdentityFile ~/.ssh/id_rsa
  ServerAliveInterval 60
```

Hierna heb ik de verschillende ssh verbindingen uitgetest door vanaf mijn host

```
ssh db
ssh web
ssh companyrouter
...
```

Indien we dit nog extra veilig willen maken kunnen we een nieuwe gebruiker installen naast de vagrant gebruiker. Ik schakelde ook de wachtwoord authenticatie uit in /etc/ssh/sshd\_config door PasswordAuthentication op no te zetten.

## SSH port forwarding:

Webserver bekijken op je laptop Je stuurt poort 8080 op je laptop door naar poort 80 van de webserver via de router.

```
# Formule: ssh -L [lokale_poort]:[doel_ip]:[doel_poort] [bastion_host]
ssh -L 8080:172.30.10.10:80 companyrouter
```

Database bereikbaar maken op de router Iemand op het 'fake internet' kan dan verbinden met de router op poort 3306 en wordt doorgestuurd naar de DB.

```
# Formule: ssh -R [poort_op_router]:[doel_ip]:[doel_poort] [bastion_host]
ssh -R 3306:172.30.20.15:3306 companyrouter
```

Je wilt je laptop zo instellen dat al je browserverkeer via de router loopt, zodat je alle interne IP-adressen direct kunt intypen.

```
ssh -D 9090 companyrouter
# Dan in je browser de proxy instellen op localhost:9090
```

## SSH Port Forwarding vragen

### Waarom is dit interessant vanuit security oogpunt?

Het stelt je in staat om firewall-restricties te omzeilen door verkeer te "tunnellen" door een reeds toegestane SSH-verbinding (meestal poort 22). Hierdoor kun je diensten bereiken die niet direct aan het internet zijn blootgesteld, wat zowel handig is voor beheer als voor aanvallers.

### Wanneer gebruik je Local Port Forwarding (-L)?

Wanneer je vanaf je lokale machine toegang wilt tot een dienst in het remote netwerk.

Voorbeeld: Je wilt de database op een interne server bekijken in een tool op je eigen laptop.

### Wanneer gebruik je Remote Port Forwarding (-R)?

Wanneer je een dienst op je lokale machine (of jouw netwerk) beschikbaar wilt maken voor de remote server.

Voorbeeld: Je wilt dat een server in de cloud tijdelijk toegang krijgt tot een webserver die op jouw laptop draait (vaak gebruikt bij malware C2-callbacks of het omzeilen van NAT).

### Welke is gebruikelijker in security?

Local Port Forwarding (-L) is de standaard voor systeembeheer en tunneling. Echter, in red teaming/hacking is Remote Port Forwarding (-R) cruciaal voor het opzetten van reverse shells en het exfiltreren van data vanuit afgeschermde netwerken.

## Waarom de "Poor man's VPN"?

Omdat SSH-tunneling een versleutelde verbinding (tunnel) creëert tussen twee netwerken zonder dat je complexe VPN-software (zoals OpenVPN of IPsec) hoeft te configureren. Het biedt encryptie en toegang tot interne resources, maar mist de geavanceerde routing- en managementfuncties van een echte VPN.

# Verslag labo 4: Honeypots

---

## Wat is een honeypot?

Een honeypot is een beveiligingsmechanisme dat is ontworpen om aanvallers aan te trekken, te detecteren en te analyseren door een ogen schijnlijk kwetsbaar of waardevol systeem, dienst of resource aan te bieden. Het doel van een honeypot is om kwaadwillende activiteiten te lokken en te observeren zonder dat deze daadwerkelijk schade kunnen aanrichten aan het echte netwerk of systeem.

## Uitwerking LABO

Why is companyrouter, in this environment, an interesting device to configure with a SSH honeypot? What could be a good argument to NOT configure the router with a honeypot service?

Het is een centraal netwerkapparaat, dus aanvallers zullen dit vaak scannen of proberen binnen te geraken.

Dat maakt het een realistisch doelwit → de honeypot gaat effectief aanvallen aantrekken.

Het router IP is meestal default gateway, dus makkelijk vindbaar.

Argument tegen:

Een router moet stabiel en veilig blijven. Een honeypot draaien op dezelfde machine verhoogt de risico's → extra software, extra attack surface.

Als de honeypot crasht of misbruikt wordt, kan dit impact hebben op de routerfunctie.

SSH verschuiven van poort 22 -> 2222

Op de companyrouter openen we de SSH config. Dit doen we met het commando:

```
sudo nano /etc/ssh/sshd_config
```

Nu zoeken we de lijn waarop de poort gedeclareerd wordt. Deze staat in commentaar dus we halen de hash weg en maken er 2222 van. Hierna paste ik deze poort ook nog aan in de ssh config van mijn gebruiker op mijn host zodat het commando ssh companyrouter meteen op de juiste poort probeert te verbinden.

Installeren van cowrie

Hierna installeerde ik cowrie op de VM, voor het gemak deed ik dit via docker.

Ik moest hiervoor uiteraard eerst docker installeren op de VM, dit deed ik met de volgende commandos:

```
sudo dnf install -y dnf-plugins-core  
sudo dnf config-manager --add-repo  
https://download.docker.com/linux/centos/docker-ce.repo  
sudo dnf install -y docker-ce docker-ce-cli containerd.io
```

```
sudo systemctl enable --now docker
```

Gebruiker toevoegen aan de docker groep zodat er geen sudo meer nodig is:

```
sudo usermod -aG docker vagrant
```

Nu kunnen we beginnen aan de cowrie installatie.

```
sudo mkdir /opt/cowrie  
sudo mkdir /opt/cowrie/log  
sudo mkdir /opt/cowrie/data  
sudo mkdir /opt/cowrie/etc
```

```
sudo curl -o /opt/cowrie/etc/cowrie.cfg \https://raw.githubusercontent.com/cowrie/cowrie/master/cowrie.cfg.dist
```

```
sudo docker run -d \  
--name cowrie \  
-p 22:2222 \  
-v /opt/cowrie/log:/cowrie/var/log/cowrie \  
-v /opt/cowrie/data:/cowrie/var/lib/cowrie \  
-v /opt/cowrie/etc/cowrie.cfg:/cowrie/etc/cowrie.cfg \  
cowrie/cowrie:latest
```

Om de container automatisch te laten starten bij het opstarten van de VM:

```
sudo docker update --restart unless-stopped cowrie
```

cowrie draait nu op de ssh poort. We kunnen verbinding maken met de router.

via het ssh commando: [ssh root@192.168.62.253](#)

What credentials work? Do you find credentials that don't work?

Je kunt met eender welk wachtwoord inloggen. Op een paar wachtwoorden na: root - root werkt bijvoorbeeld niet. Dit is blijkbaar gedaan om het realistisch te laten zijn. De aanvaller krijgt de indruk dat het een correct beveiligd systeem is, maar kan hij alsnog verder proberen met andere combinaties.

Do you get a shell?

We krijgen een shell, maar dit is geen echte shell, we zien ook een verschil aan de structuur tov de normale alma shell. Het is een emulated environment die Cowrie voorziet.

Are your commands logged? Is the IP address of the SSH client logged? If this is the case, where?

Ja, alle ingegeven commando's worden gelogd, inclusief:

- Het IP-adres van de aanvaller
- De ingegeven username en password
- Alle commando's die worden uitgevoerd
- Eventuele bestanden die geprobeerd worden te downloaden of uploaden

Can an attacker perform malicious things? Are the actions, in other words, the commands, logged to a file? Which file?

Deze komen in het logbestand cowrie.json te staan, dit bestand is terug te vinden in de docker volume op de host:

```
sudo cat  
/var/lib/docker/volumes/0ba09fa5d96ac103633bbe1a1f25b002b3218dd6a4e305cac89c363efd  
60bd3b/_data/log/cowrie/cowrie.json | jq .
```

Deel van de output:

```
"message": "SSH client hassh fingerprint: 701158e75b508e76f0410d5d22ef9df0",  
"sensor": "1b264fc3979f",  
"uuid": "5c16120c-e619-11f0-a404-7eb184fd5be4",  
"timestamp": "2026-01-02T11:13:08.907586Z",  
"src_ip": "192.168.62.0",  
"session": "e9c7ef908578",  
"protocol": "ssh"  
}  
{  
"eventid": "cowrie.login.success",  
"username": "root",  
"password": "hhdd",  
"message": "login attempt [root/hhdd] succeeded",  
"sensor": "1b264fc3979f",  
"uuid": "5c16120c-e619-11f0-a404-7eb184fd5be4",  
"timestamp": "2026-01-02T11:13:10.972102Z",  
"src_ip": "192.168.62.0",  
"session": "e9c7ef908578",  
"protocol": "ssh"  
}
```

```
{  
    "eventid": "cowrie.client.size",  
    "width": 102,  
    "height": 14,  
    "message": "Terminal Size: 102 14",  
    "sensor": "1b264fc3979f",  
    "uuid": "5c16120c-e619-11f0-a404-7eb184fd5be4",  
    "timestamp": "2026-01-02T11:13:11.052099Z",  
    "src_ip": "192.168.62.0",  
    "session": "e9c7ef908578",  
    "protocol": "ssh"  
}  
}  
{  
    "eventid": "cowrie.session.params",  
    "arch": "linux-x64-lsb",  
    "message": [],  
    "sensor": "1b264fc3979f",  
    "uuid": "5c16120c-e619-11f0-a404-7eb184fd5be4",  
    "timestamp": "2026-01-02T11:13:11.053932Z",  
    "src_ip": "192.168.62.0",  
    "session": "e9c7ef908578",  
    "protocol": "ssh"  
}  
}  
{  
    "eventid": "cowrie.command.input",  
    "input": "curl localhost",  
    "message": "CMD: curl localhost",  
    "sensor": "1b264fc3979f",  
    "uuid": "5c16120c-e619-11f0-a404-7eb184fd5be4",  
    "timestamp": "2026-01-02T11:13:17.495586Z",  
    "src_ip": "192.168.62.0",  
    "session": "e9c7ef908578",  
    "protocol": "ssh"  
}  
}  
{  
    "eventid": "cowrie.command.input",  
    "input": "ls",  
    "message": "CMD: ls",  
    "sensor": "1b264fc3979f",  
    "uuid": "5c16120c-e619-11f0-a404-7eb184fd5be4",  
    "timestamp": "2026-01-02T11:13:29.512887Z",  
    "src_ip": "192.168.62.0",  
    "session": "e9c7ef908578",  
    "protocol": "ssh"  
}  
}  
{  
    "eventid": "cowrie.session.closed",  
    "duration": "200.5",  
    "message": "Connection lost after 200.5 seconds",  
    "sensor": "1b264fc3979f",  
    "uuid": "5c16120c-e619-11f0-a404-7eb184fd5be4",  
    "timestamp": "2026-01-02T11:16:29.355355Z",  
    "src_ip": "192.168.62.0",  
    "session": "e9c7ef908578",  
}
```

```
    "protocol": "ssh"  
}
```

We kunnen ook de logs in real-time bekijken met:

```
sudo docker logs -f cowrie
```

Nee, een aanvaller kan geen echte schade aanrichten. Alles gebeurt binnen de geëmuleerde omgeving:

- Ze kunnen geen echte systeemwijzigingen doorvoeren
- Ze kunnen geen netwerkscans van het echte systeem uitvoeren
- Ze kunnen geen bestanden aanpassen buiten de Cowrie-container
- Zelfs commando's zoals rm -rf / of wget zijn fake en worden enkel gelogd

Cowrie vangt alle acties op en logt ze, maar voert niets echt uit op het hostsysteem. Dit is exact het doel van een honeypot: aanvallers bezig houden en observeren zonder risico.

If you are an experienced hacker, how would/can you realize this is not a normal environment?

Het valt af te leiden uit een aantal zaken, bijvoorbeeld:

- Veel standaardtools (vim, nano, top, iptables, systemctl, ...) ontbreken of geven rare output.
- Directoryinhoud klopt niet met een echte Linux-installatie, of bepaalde paden zijn leeg.
- Commando's zoals ls, uname, df of cat /etc/passwd geven hardcoded output.
- Zelfs bij zware commando's blijft het systeem reageren zonder CPU-load.
- Commando's zoals ping, wget, of curl lijken te werken, maar er gebeurt niets effectiefs.
- ps toont een heel beperkt procesoverzicht.

## Critical thinking (security) when using "Docker as a service"

Wanneer je beslist om Cowrie (of gelijk welke service) te draaien in Docker, kies je voor een bepaalde deployment-architectuur. Dit heeft voor- en nadelen, zeker in een beveiligingscontext.

### Voordelen van services draaien in Docker

Een container draait geïsoleerd van het host-systeem.

Minder risico dat een kwetsbaarheid in het honeypot-proces de hele host infecteert. Je kan eenvoudig testen, experimenteren en dingen stukmaken zonder je basis OS te breken.

### Een nadeel van services draaien in Docker

De Docker daemon (dockerd) draait standaard als root.

Als een aanvaller kan uitbreken uit de container (container escape) → dan heeft hij toegang tot de hele host met root-privileges.

Containers worden vaak gezien als mini-VM's, maar in realiteit delen ze dezelfde kernel.

Voor een honeypot verhoogt dit risico: de honeypot moet kwetsbaar lijken, maar je wil niet dat een kwetsbaarheid in Cowrie de hele host besmet.

Wat betekent "Docker uses a client-server architecture"?

Docker bestaat uit:

- docker daemon (dockerd) → de server
- draait achtergrondprocessen
- maakt containers aan
- beheert volumes, netwerken, images
- draait meestal als root

docker CLI → de client

- het programma docker dat jij uitvoert in de terminal
- stuurt opdrachten naar de Docker daemon
- communiceert meestal via /var/run/docker.sock

Dus: Je CLI voert zelf niets uit. De Docker daemon doet al het werk. Jij stuurt enkel API-calls.

Onder welke user draait de Docker daemon?

Volgens de officiële documentatie draait de Docker daemon standaard als root.

What could be an advantage of running a honeypot inside a virtual machine compared to running it inside a container?

Sterkere isolatie

Een VM heeft een volledig aparte kernel, hardware-abstractie en geheugenruimte.

Als een aanvaller: Cowrie probeert te misbruiken, een exploit uitvoert, privilege escalation doet of een container escape vindt dan geraakt hij in het ergste geval enkel in de honeypot-VM, niet op de fysieke host.

Voor een honeypot is dit veel veiliger, want je laat bewust "kwetsbare of misleidende" software draaien.

Conclusie: Een VM is veiliger dan een container

## Docker deep dive

Waarom wordt /var/run/docker.sock gemount?

Omdat dit het communicatiekanaal is tussen de Docker CLI en de Docker daemon. We hebben dit nodig zodat de container opdrachten kan sturen naar de host Docker daemon.

- containers te starten/stoppen
- images te pullen

- volumes/netwerken te beheren
- logs op te vragen
- statistieken op te vragen

Wat is docker.sock precies?

een UNIX domain socket bestand, de lokale API endpoint van de Docker daemon (dockerd) te vinden op /var/run/docker.sock

Waarom brengt dit ernstige security risico's?

Mounten van /var/run/docker.sock in een container ≈ container volledige root-toegang geven tot je host.

Met toegang tot de Docker API kan een aanvaller:

- een privileged container starten
- een container starten met -v /:/host → volledige host filesystem
- bestanden aanpassen op de host
- persistente malware installeren
- netwerken aanpassen
- logs uitlezen met gevoelige informatie
- environment variables en secrets van andere containers zien

"Access to docker.sock = root access to the host."

Daarom is dit een high-impact security risk.

## Other honeypot types

Wat voor type honeypot is honeyup?

Honeyup is een:

- Honeypot network analyzer / honeynet visualizer
- Ontworpen om netwerkscans te detecteren, vooral ARP-scan, Nmap, DHCP-scans, etc.

Het is dus een network-level honeypot die footprinting/scanning zichtbaar maakt.

Wat is het idee achter OpenCanary?

OpenCanary is een: lightweight host-based honeypot dat nep-services aanbiedt zoals:

- FTP
- SSH
- HTTP
- Redis
- SMB
- RDP
- etc.

Het doel is intruders detecteren, niet te "foppen".

Idee:

Je draait nep-services die nooit in normale werking gebruikt worden. Als iemand ermee verbindt → onmiddellijke alarmmelding.

Is a HTTP(S) honeypot a good idea? Why or why not?

Ja én nee, afhankelijk van de context.

Voordelen:

- Simuleert een realistische aanvalsvector
- Je kan exploit attempts, payloads, bots en scans zien
- Geen echte schade want de backend is nep

Nadelen:

- HTTP(S) verkeer is enorm → veel ruis
- Veel false positives (bots, scrapers) Aanvallers zien soms te snel dat het een honeypot is
- HTTPS vergt certificaatmanagement

Conclusie:

Goed idee voor research of threat intel, maar niet ideaal als primaire security measure.

Ons /cmd endpoint als honeypot: hoe commands loggen?

Je kan alle input die gebruikers posten loggen. Voorbeeld:

```
@app.route('/cmd', methods=['POST'])
def cmd():
    command = request.form.get("command")
    with open("/var/log/cmd_honeypot.log", "a") as logfile:
        logfile.write(f"{datetime.now()} - {command}\n")
    return "OK"
```

Is dit haalbaar?

Ja, zelfs heel eenvoudig. Alles wat binnenkomt via een form POST kan je gewoon dumpen naar:

- een logfile
- een database
- syslog
- Elasticsearch/Grafana Loki

Zo heb je een volledig overzicht van alle gebruikte commando's door aanvallers.

## Labo 5: backups

---

Om dit labo te voltooien volgde ik de stappen uit de opdracht:

```
[vagrant@web ~]$ mkdir important-files
[vagrant@web ~]$ cd !$
[vagrant@web important-files]$ curl --remote-name-all
https://video.blender.org/download/videos/bf1f3fb5-b119-4f9f-9930-8e20e892b898-
720.mp4 https://www.gutenberg.org/ebooks/100.txt.utf-8
https://www.gutenberg.org/ebooks/996.txt.utf-8
https://upload.wikimedia.org/wikipedia/commons/4/40/Toreador_song_cleaned.ogg
[vagrant@web important-files]$ mv 100.txt.utf-8 100.txt # Optional
[vagrant@web important-files]$ mv 996.txt.utf-8 996.txt # Optional
[vagrant@web important-files]$ ll
total 109992
-rw-r--r--. 1 vagrant vagrant      300 Nov  4 12:37 100.txt
-rw-r--r--. 1 vagrant vagrant      300 Nov  4 12:37 996.txt
-rw-r--r--. 1 vagrant vagrant  1702187 Nov  4 12:37 Toreador_song_cleaned.ogg
-rw-r--r--. 1 vagrant vagrant 110916740 Nov  4 12:37 bf1f3fb5-b119-4f9f-9930-
8e20e892b898-720.mp4
```

We controleerden of we de bestanden konden openen:

```
[vagrant@web important-files]$ ls
100.txt  996.txt  bf1f3fb5-b119-4f9f-9930-8e20e892b898-720.mp4
Toreador_song_cleaned.ogg
```

Hierna maakte ik op de DB een map backups aan om de backups in op te slaan:

```
mkdir backups
```

We controleerden of borgbackup in de dnf package manager stond en installeerden het indien nodig:

```
No matches found.
```

Dit gaf helaas geen matches. We moesten dus eerst de EPEL repository toevoegen:

```
sudo dnf install epel-release
sudo dnf update
sudo dnf search borgbackup
```

Op de alpine:

```
apk update
apk add borgbackup
```

Nu moesten we de repository initialiseren:

```
borg init --encryption=repokey vagrant@172.30.20.15:~/backups
```

Passphrase: test

Om met keys te werken, moesten we eerst een SSH key genereren op de webserver:

```
ssh-keygen -t rsa -b 4096  
ssh-copy-id vagrant@172.30.20.15
```

Nu exporteren we de borg-keyfile:

```
borg key export ~/borg-keyfile vagrant@172.30.20.15:~/backups
```

Met borg --version controleren we of alles correct is geïnstalleerd:

```
borg --version  
borg 1.2.7
```

Nu stellen we de borg key veilig:

```
borg key export vagrant@172.30.20.15:~/backups ~/borg-key.txt
```

Eerste backup maken:

```
borg create \  
vagrant@172.30.20.15:~/backups::first \  
/home/vagrant/important-files
```

De repository info bekijken:

```
[vagrant@web important-files]$ borg info vagrant@172.30.20.15:~/backups  
Enter passphrase for key ssh://vagrant@172.30.20.15/~/backups:  
Repository ID: ddfe327a44b883e5ba50899fa89a27b15f9f7ac8af80081aa41c11e3a7b03e65  
Location: ssh://vagrant@172.30.20.15/~/backups  
Encrypted: Yes (repokey)  
Cache:  
/home/vagrant/.cache/borg/ddfe327a44b883e5ba50899fa89a27b15f9f7ac8af80081aa41c11e3  
a7b03e65
```

```
Security dir:  
/home/vagrant/.config/borg/security/ddfe327a44b883e5ba50899fa89a27b15f9f7ac8af8008  
1aa41c11e3a7b03e65  
-----  
Original size Compressed size Deduplicated size  
All archives: 0 B 0 B 0 B  
  
Unique chunks Total chunks  
Chunk index: 0 0
```

## Tweede backup maken na een wijziging:

We maken een extra bestand aan in de map important-files:

```
echo "Hello world" > ~/important-files/test.txt
```

Nu maken we een tweede backup:

```
borg create \  
vagrant@172.30.20.15:~/backups::second \  
/home/vagrant/important-files
```

Backups en hun inhoud bekijken:

```
borg list vagrant@172.30.20.15:~/backups
```

Per backup de inhoud bekijken:

```
borg list vagrant@172.30.20.15:~/backups::first  
borg list vagrant@172.30.20.15:~/backups::second
```

## Grootte van de backups vergelijken:

```
du -h ~/important-files  
du -h --si ~/important-files  
1.7M /home/vagrant/important-files  
1.8M /home/vagrant/important-files
```

grootte van de backup op de server bekijken:

```
[vagrant@web ~]$ du -h ~/important-files
du -h --si ~/important-files
1.7M    /home/vagrant/important-files
1.8M    /home/vagrant/important-files
```

We moeten ook regelmatig de integriteit van de repository controleren:

```
borg check vagrant@172.30.20.15:~/backups
```

Optioneel kan je er ook voor kiezen om een volledige check te doen:

```
borg check --verify-data --verbose vagrant@172.30.20.15:~/backups
```

## Data verwijderen en terugzetten

Nu gaan we de data in de map important-files verwijderen:

```
rm -rf ~/important-files
```

Deze is nu leeg:

```
[vagrant@web important-files]$ ls
[vagrant@web important-files]$
```

We gaan nu de data terugzetten vanuit de eerste backup:

```
[vagrant@web ~]$ borg extract vagrant@172.30.20.15:~/backups::first --strip-components 3
Enter passphrase for key ssh://vagrant@172.30.20.15/~/backups:
[vagrant@web ~]$ ls
100.txt  bf1f3fb5-b119-4f9f-9930-8e20e892b898-720.mp4  test.txt
996.txt  borg-key.txt                                Toreador_song_cleaned.ogg
```

De bestanden zijn succesvol teruggezet!

## Backup script

We maken een script aan met touch borg-backup.sh en voegen de volgende inhoud toe:

```
#!/bin/bash
REPO="vagrant@172.30.20.15:~/backups"
SOURCE="/home/vagrant/important-files"

borg create \
--stats \
--compression lz4 \
$REPO::'{now:%Y-%m-%d_%H:%M}' \
$SOURCE

borg prune \
--keep-minute=12 \
--keep-hourly=24 \
--keep-daily=7 \
--keep-monthly=6 \
$REPO

borg compact $REPO
```

Nu maken we het script uitvoerbaar:

```
chmod +x borg-backup.sh
```

We maken er een systemd service van en zetten er een timer op:

```
sudo vi /etc/systemd/system/borg-backup.service
```

Met de volgende inhoud:

```
[Unit]
Description=Borg Backup

[Service]
Type=oneshot
ExecStart=/home/vagrant/borg-backup.sh
```

De timer zetten we als volgt op:

```
sudo vi /etc/systemd/system/borg-backup.timer
```

Met de volgende inhoud:

**[Unit]**

Description=Run Borg backup every **5** minutes

**[Timer]**

OnBootSec=**5min**

OnUnitActiveSec=**5min**

**[Install]**

WantedBy=timers.target

We activeren en starten alles met de volgende commando's:

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable --now borg-backup.timer
```

We zien nu dat hij draait:

```
[vagrant@web ~]$ systemctl list-timers | grep borg
Tue 2025-12-23 09:51:12 UTC 4min 54s left Tue 2025-12-23 09:46:12 UTC 5s ago
borg-backup.timer
    borg-backup.service
```

Borg compact verwijdert onnodige data uit de repository om ruimte te besparen.

A brain teaser: can I use tools like borg to backup an active database? Why (not)? Read <https://borgbackup.readthedocs.io/en/stable/quickstart.html#important-note-about-files-changing-during-the-backup-process> for more information. Should I take any extra measures to do this safely?

There is a tool that has been built on top of borg called borgmatic. What does it do? Could it be useful to you? Why (not)?

Borg is geen ideale tool om een actieve database te backen omdat de data tijdens het backup-proces kan veranderen, wat kan leiden tot inconsistentie backups. Extra maatregelen zoals het gebruik van database-specificke backup tools of het tijdelijk stilleggen van de database kunnen nodig zijn om een betrouwbare backup te garanderen.

Borgmatic is een wrapper rond borg die het backup-proces vereenvoudigt door configuratiebestanden en automatisering te bieden. Het kan nuttig zijn voor gebruikers die regelmatig backups willen maken zonder de complexiteit van borg zelf te hoeven beheren, vooral in omgevingen waar consistentie en eenvoud belangrijk zijn.

## Labo 6: CA HTTPS

Eerst moeten we de webserversoftware analyseren, we doen dit met de volgende commando's:

```
ps aux | egrep 'nginx|apache|httpd'
systemctl status httpd
```

Wat webserver software is being used on the webserver? (apache, nginx, iis, ...) We zien dat httpd (Apache) draait.

How is the webserver configured as a reverse proxy? Where is this defined? What config file? De reverse proxy is geconfigureerd in de httpd.conf file, deze bevindt zich in /etc/httpd/conf/httpd.conf.

Wat we hieruit kunnen afleiden:

```
ProxyPass "/services" "http://localhost:9200"
ProxyPassReverse "/services" "http://localhost:9200"
ProxyPass "/cmd" "http://localhost:8000/"
ProxyPassReverse "/cmd" "http://localhost:8000/"
ProxyPass "/assets" "http://localhost:8000/assets"
ProxyPassReverse "/assets" "http://localhost:8000/assets"
ProxyPass "/exec" "http://localhost:8000/exec"
ProxyPassReverse "/exec" "http://localhost:8000/exec"
```

/cmd and /services are both (systemd) services running on the webserver. In other words it is not the webserver software that is hosting this. What are the names of these (systemd) services, what programming languages are used and on what port is each service listening for incoming requests? Can you find the files that are needed by each of the services?

We kunnen de services vinden met het volgende commando:

```
systemctl list-units --type=service
```

De services zijn als volgt:

Service Name	Status	Description	Port	Language/Framework
httpd.service	active	Apache HTTP Server	80/443 (HTTP/HTTPS)	Apache
insecurewebapp.service	active	Java app (start script)	9200	Java (app.jar)
flaskapp.service	failed	Flask app (start script)	8000	Python/Flask

De cmd service draait op poort 8000 en is geschreven in Python met het Flask framework. De /services service draait op poort 9200 en is geschreven in Java.

De bestanden voor de cmd service bevinden zich in /opt/flaskapp/ en voor de services service in /opt/insecurewebapp/.

## HTTPS

We gaan in dit labo onze isprouter configureren als een Certificate Authority (CA) en zorgen ervoor dat onze webserver HTTPS gebruikt.

Eerst installeren we de benodigde pakketten op de alpine linux machine (isprouter):

```
apk add openssl ca-certificates
```

Vervolgens genereren we de private key voor onze CA:

```
openssl genrsa -out ca.key 4096
```

Hierna maken we het self-signed root certificate aan:

```
openssl req -x509 -new -nodes \
-key ca.key \
-sha256 -days 3650 \
-out ca.crt
```

We kunnen nu het certificaat bekijken:

```
openssl x509 -in ca.crt -noout -text
```

Nu genereren we de private key voor de webserver:

```
openssl genrsa -out webserver.key 2048
```

### CRS: Certificate Signing Request

We stellen eerst de SAN (Subject Alternative Name) configuratie in:

```
sudo vi san.cnf
```

Met de volgende inhoud:

```
[req]
default_bits = 2048
prompt = no
default_md = sha256
distinguished_name = dn
req_extensions = req_ext

[dn]
CN = www.cybersec.internal

[req_ext]
subjectAltName = @alt_names

[alt_names]
DNS.1 = www.cybersec.internal
DNS.2 = services.cybersec.internal
```

Vervolgens maken we de CSR aan:

```
openssl req -new \
-key webserver.key \
-out webserver.csr \
-config san.cnf
```

Nu ondertekenen we de CSR met onze CA om een certificaat voor de webserver te maken:

```
openssl x509 -req \
-in webserver.csr \
-CA ca.crt \
-CAkey ca.key \
-CACreateserial \
-out webserver.crt \
-days 825 \
-sha256 \
-extfile san.cnf \
-extensions req_ext
```

Nu is ons webserver certificaat ondertekend door onze eigen CA.

Hierna paste ik de apache webserver aan om HTTPS te gebruiken.

```
sudo vi /etc/httpd/conf.d/ssl-tls12.conf
```

Met de volgende aanpassingen:

```
<VirtualHost *:443>
    ServerName www.cybersec.internal

    SSLEngine on
    SSLCertificateFile /etc/pki/tls/certs/webserver.crt
    SSLCertificateKeyFile /etc/pki/tls/private/webserver.key

    SSLProtocol TLSv1.2
    SSLCipherSuite RSA+AES256-SHA
    SSLHonorCipherOrder on
    SSLSessionTickets off

    ProxyPass /cmd http://localhost:8000/
    ProxyPassReverse /cmd http://localhost:8000/

    ProxyPass /services http://localhost:9200/
    ProxyPassReverse /services http://localhost:9200/
</VirtualHost>

<VirtualHost *:80>
    ServerName www.cybersec.internal
    Redirect permanent / https://www.cybersec.internal/
</VirtualHost>
```

We testen de configuratie en herstarten apache:

```
sudo apachectl configtest
sudo systemctl restart httpd
```

Na een dig test zien we dat alles correct is ingesteld:

```
—(vagrant@red)-[~/Desktop]
└$ dig www.cybersec.internal

; <>> DiG 9.20.11-4+b1-Debian <>> www.cybersec.internal
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17112
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: ac079fe1330aa15601000000695131f40832601b90735ef3 (good)
;; QUESTION SECTION:
;www.cybersec.internal.      IN      A

;; ANSWER SECTION:
www.cybersec.internal.  86400   IN      A      172.30.10.10
```

```
;; Query time: 4 msec
;; SERVER: 172.30.20.4#53(172.30.20.4) (UDP)
;; WHEN: Sun Dec 28 08:34:46 EST 2025
;; MSG SIZE  rcvd: 94
```

Hierna kopieerde ik de certificaten en sleutels naar de juiste locaties op de isprouter:

```
sudo mkdir -p /etc/ssl/certs
sudo mkdir -p /etc/ssl/private
```

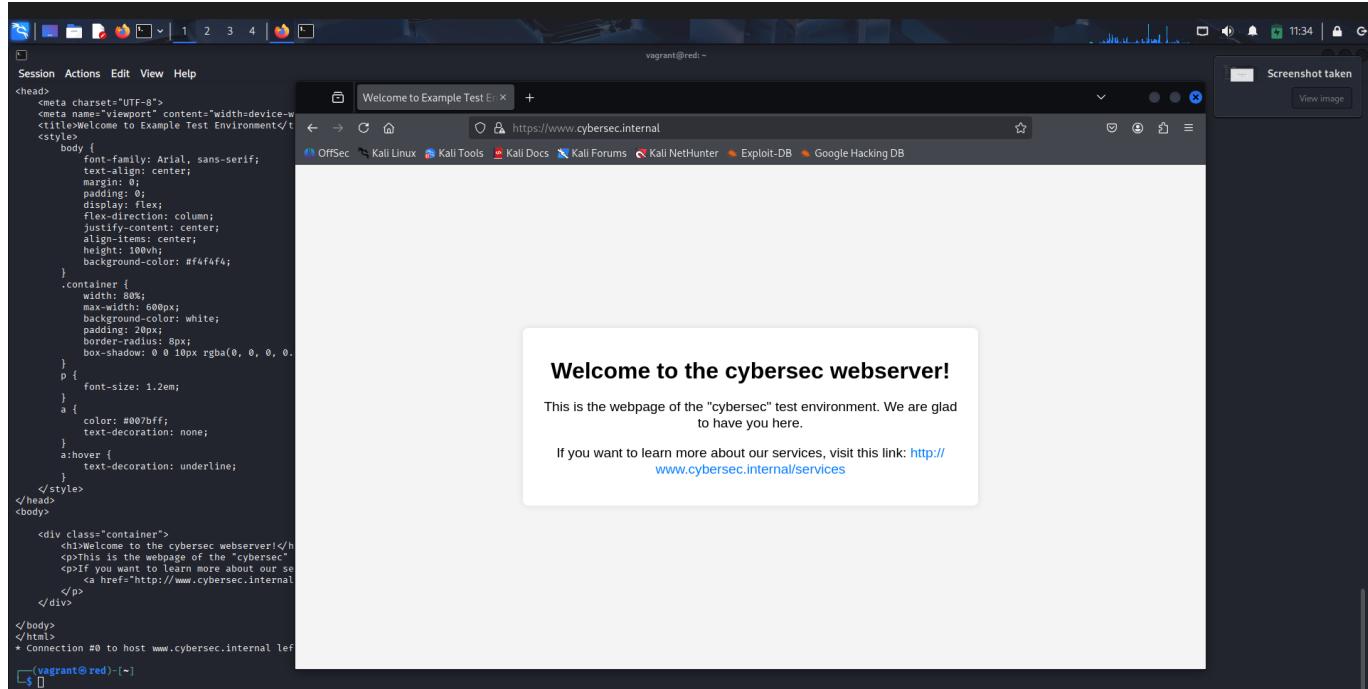
```
isprouter:~$ sudo cp ~/webserver.crt /etc/ssl/certs/
sudo cp ~/webserver.key /etc/ssl/private/
```

```
isprouter:~$ sudo chown root:root /etc/ssl/certs/webserver.crt
/etc/ssl/private/webserver.key
sudo chmod 644 /etc/ssl/certs/webserver.crt
sudo chmod 600 /etc/ssl/private/webserver.key
```

Op de isp router moest ik ook nog het verkeer toestaan in de firewall:

```
sudo nft add rule inet filter input tcp dport 443 accept
sudo nft add rule inet filter input tcp dport 80 accept
```

Hierna is de website bereikbaar via HTTPS:



```

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Welcome to Example Test Environment</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      text-align: center;
      margin: 0;
      padding: 0;
      display: flex;
      flex-direction: column;
      justify-content: center;
      align-items: center;
      height: 100vh;
      background-color: #f4f4f4;
    }
    .container {
      width: 80px;
      max-width: 600px;
      background-color: white;
      padding: 20px;
      border-radius: 8px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    }
    p {
      font-size: 1.2em;
    }
    a {
      color: #007bff;
      text-decoration: none;
    }
    a:hover {
      text-decoration: underline;
    }
  </style>
</head>
<body>
  <div class="container">
    <h1>Welcome to the cybersec webserver!</h1>
    <p>This is the webpage of the "cybersec" test environment.<br/>
      If you want to learn more about our services,<br/>
      <a href="http://www.cybersec.internal/services">click here</a></p>
  </div>
</body>
</html>
* Connection #0 to host www.cybersec.internal left open

```

CA trusten op de kali machine:

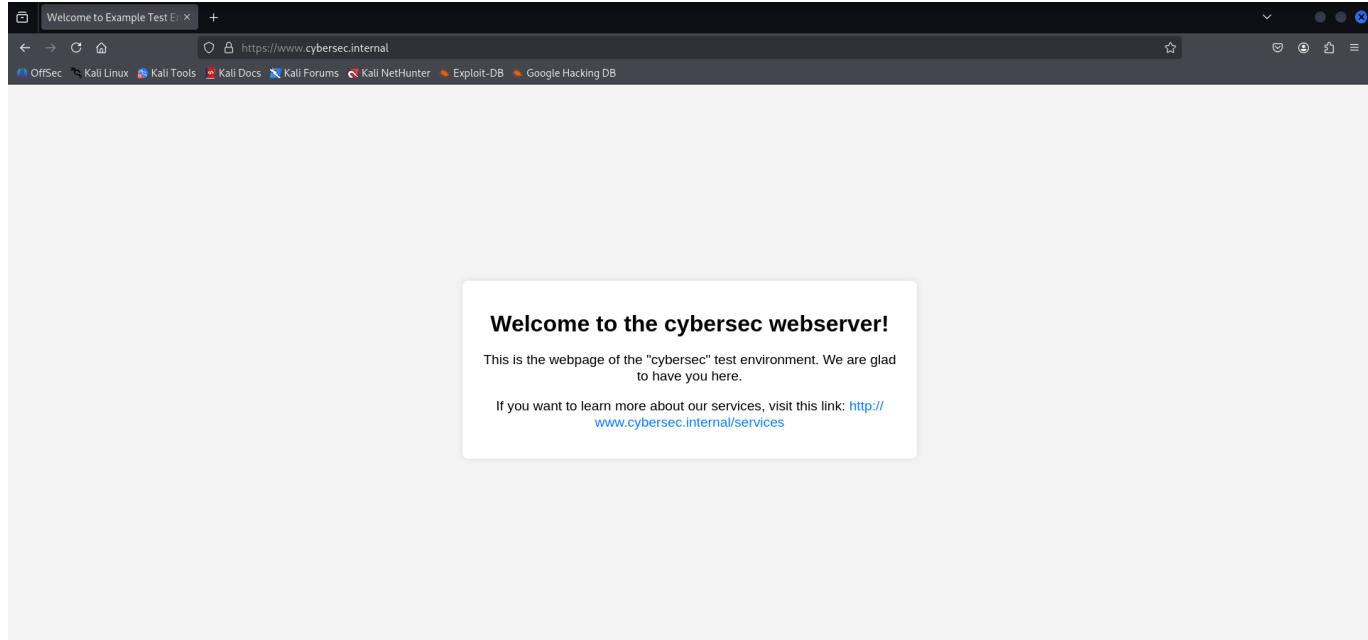
Om de CA te trusten op de kali machine, kopieerde ik het ca.crt bestand naar de kali machine en voegde ik het toe aan de vertrouwde certificaten:

```

sudo cp ca.crt /usr/local/share/ca-certificates/
sudo update-ca-certificates

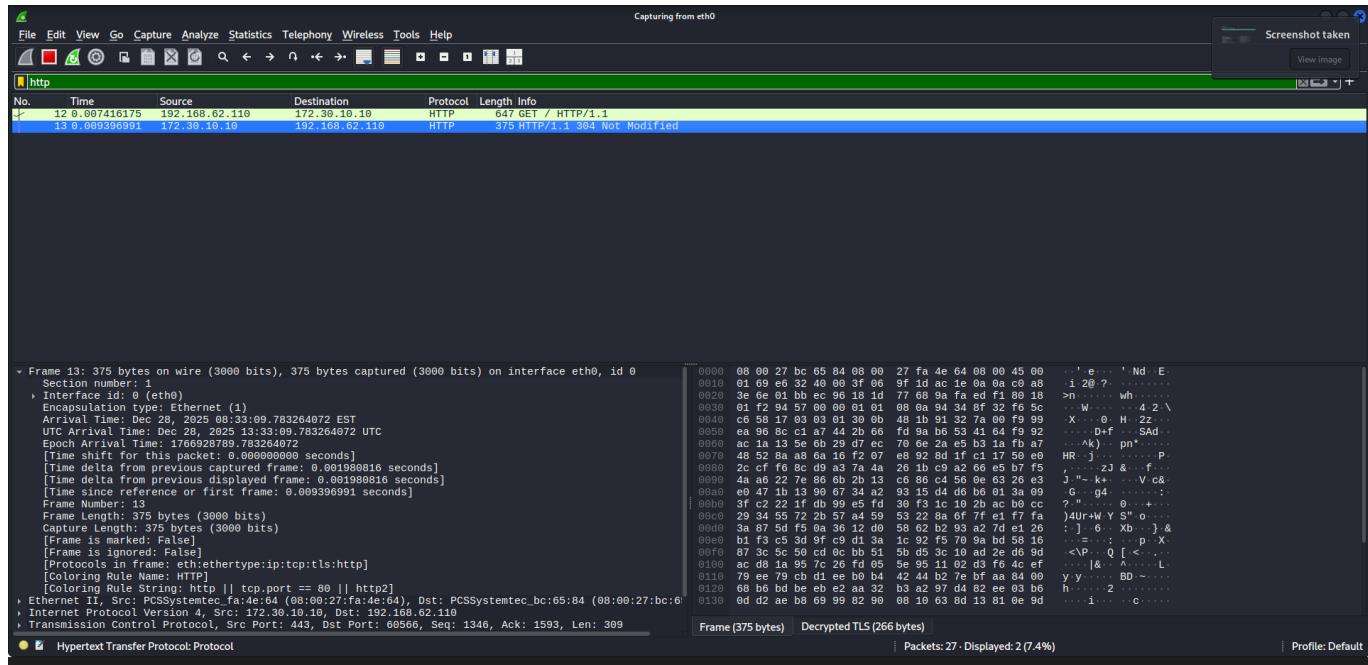
```

We zien nu een mooi slotje in de browser:



## Wireshark capture

Om het verkeer met wireshark te bekijken hebben we nood aan de private key van de webserver. We importeren deze in wireshark.



## HTTPS TLS 1.3

Om TLS 1.3 te configureren, maakte ik een nieuwe private key aan op de isprouter:

```
openssl genrsa -out webserver_tls13.key 2048
```

Vervolgens maakte ik een nieuwe CSR aan:

```
openssl req -new \
-key webserver_tls13.key \
-out webserver_tls13.csr \
-config san.cnf
```

Hierna ondertekende ik de CSR met onze CA om een certificaat voor TLS 1.3 te maken:

```
openssl x509 -req \
-in webserver_tls13.csr \
-CA ca.crt \
-CAkey ca.key \
-CAcreateserial \
-out webserver_tls13.crt \
-days 825 \
-sha256 \
-extfile san.cnf \
-extensions req_ext
```

Ik plaatste alles in de juiste mappen:

```
# Certificaten verplaatsen naar de juiste map
sudo cp webserver_tls13.crt /etc/ssl/certs/
sudo cp webserver_tls13.key /etc/ssl/private/

# Permissies correct instellen
sudo chown root:root /etc/ssl/certs/webserver_tls13.crt
/etc/ssl/private/webserver_tls13.key
sudo chmod 644 /etc/ssl/certs/webserver_tls13.crt
sudo chmod 600 /etc/ssl/private/webserver_tls13.key
```

Op de webserver paste ik de apache configuratie aan om TLS 1.3 te gebruiken:

```
sudo vi /etc/httpd/conf.d/ssl-tls13.conf
```

Met de volgende inhoud:

```
<VirtualHost *:443>
    ServerName www.cybersec.internal

    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/webserver_tls13.crt
    SSLCertificateKeyFile /etc/ssl/private/webserver_tls13.key

    SSLProtocol TLSv1.3
    # Cipher suites voor TLS 1.3 worden automatisch gekozen
    # Je hoeft SSLCipherSuite niet aan te passen tenzij je wil beperken

    ProxyPass /cmd http://localhost:8000/
    ProxyPassReverse /cmd http://localhost:8000/

    ProxyPass /services http://localhost:9200/
    ProxyPassReverse /services http://localhost:9200/
</VirtualHost>

<VirtualHost *:80>
    ServerName www.cybersec.internal
    Redirect permanent / https://www.cybersec.internal/
</VirtualHost>
```

Hierna testte ik de configuratie en herstartte ik apache:

```
sudo apachectl configtest
sudo systemctl restart httpd
```

We kunnen nu controleren of TLS 1.3 actief is met het volgende commando:

```
[vagrant@web ~]$ openssl s_client -connect www.cybersec.internal:443 -tls1_3
Connecting to 127.0.2.1
CONNECTED(00000003)
40C7904F467F0000:error:0A00042E:SSL routines:ssl3_read_bytes:tlsv1 alert protocol
version:ssl/record/rec_layer_s3.c:916:SSL alert number 70
---
no peer certificate available
---
No client certificate CA names sent
Negotiated TLS1.3 group: <NULL>
---
SSL handshake has read 7 bytes and written 262 bytes
Verification: OK
---
New, (NONE), Cipher is (NONE)
Protocol: TLSv1.3
This TLS version forbids renegotiation.
Compression: NONE
Expansion: NONE
No ALPN negotiated
Early data was not sent
Verify return code: 0 (ok)
---
```

We zien dat TLSv1.3 actief is.

## SSLKEYLOGFILE

SSLKEYLOGFILE laat een client (browser) de gebruikte sessiesleutels wegschrijven naar een bestand zodat tools zoals Wireshark HTTPS-verkeer kunnen ontsleutelen, zelfs bij TLS 1.3.

We kunnen dit op onze kali machine doen door de volgende stappen te volgen:

Eerst killen we alle open browsers:

```
pkill firefox
```

Vervolgens stellen we de omgevingsvariabele in:

```
export SSLKEYLOGFILE=/home/kali/sslkeys.log
```

Nu starten we de browser vanuit dezelfde terminal:

```
firefox &
```

We bezoeken de website:

Afbeelding van de website met HTTPS

We sluiten de browser en stellen het bestand met de sessiesleutels veilig:

```
cp ~/sslkeys.log ~/sslkeys_copy.log
```

Helaas heb ik dit niet kunnen exporteren naar het bestand; ik kreeg steeds een leeg bestand.

```
└─(vagrant㉿red)-[~/Desktop]
└─$ Exiting due to channel error.

[1] + terminated  firefox https://www.cybersec.internal
└─(vagrant㉿red)-[~/Desktop]
└─$ ls -l /home/vagrant/Desktop/sslkeys.log

ls: cannot access '/home/vagrant/Desktop/sslkeys.log': No such file or directory

└─(vagrant㉿red)-[~/Desktop]
└─$ echo $HOME

/home/vagrant

└─(vagrant㉿red)-[~/Desktop]
└─$ pkill -9 firefox

└─(vagrant㉿red)-[~/Desktop]
└─$ touch /home/vagrant/Desktop/sslkeys.log
chmod 600 /home/vagrant/Desktop/sslkeys.log
ls -l /home/vagrant/Desktop/sslkeys.log

-rw----- 1 vagrant vagrant 0 Dec 28 09:22 /home/vagrant/Desktop/sslkeys.log

└─(vagrant㉿red)-[~/Desktop]
└─$ SSLKEYLOGFILE=/home/vagrant/Desktop/sslkeys.log firefox
https://www.cybersec.internal &

[1] 34246

└─(vagrant㉿red)-[~/Desktop]
└─$ pkill firefox
```

```
└─(vagrant㉿red)-[~/Desktop]
  └─$ Exiting due to channel error.

[1] + terminated  SSLKEYLOGFILE=/home/vagrant/Desktop/sslkeys.log firefox
└─(vagrant㉿red)-[~/Desktop]
  └─$ ls -l /home/vagrant/Desktop/sslkeys.log
head /home/vagrant/Desktop/sslkeys.log

-rw----- 1 vagrant vagrant 0 Dec 28 09:22 /home/vagrant/Desktop/sslkeys.log
```

## Vragen bij het labo:

Does the CA uses a private key?

De CA gebruikt een private key, deze is gegenereerd met het commando `openssl genrsa -out ca.key 4096`.

Does the CA uses a certificate?

De CA gebruikt een self-signed certificate, deze is gegenereerd met het commando `openssl req -x509 -new -nodes -key ca.key -sha256 -days 3650 -out ca.crt`.

Does the web server uses a private key?

De webserver gebruikt een private key, deze is gegenereerd met het commando `openssl genrsa -out webserver.key 2048`.

Does the web server uses a certificate?

De webserver gebruikt een certificate, dit is gegenereerd met het commando `openssl x509 -req -in webserver.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out webserver.crt -days 825 -sha256 -extfile san.cnf -extensions req_ext`.

When using openssl commands to generate files, are you able to easily spot the function/goal of each file?

Ja, de functie/goal van elk bestand kan worden afgeleid uit de naam en het type van het bestand.

Bijvoorbeeld:

- ca.key: private key van de Certificate Authority
- ca.crt: self-signed certificate van de Certificate Authority
- webserver.key: private key van de webserver
- webserver.csr: Certificate Signing Request van de webserver

How can you view a certificate using openssl?

Je kunt een certificate bekijken met het commando `openssl x509 -in <certificate_file> -noout -text`.

Does the webserver need a specific configuration change to allow HTTPS traffic?

Ja, de webserver moet worden geconfigureerd om HTTPS verkeer toe te staan. Dit omvat het instellen van de juiste poort (meestal 443) en het specificeren van de locaties van het certificaat en de private key in de webserver configuratiebestanden.

What is meant by a CSR? Tip: Do not forget the SAN (Subject Alternative Name) attribute!

Een certificate signing request (CSR) is een gecodeerd bericht dat wordt verzonden naar een Certificate Authority om een digitaal certificaat aan te vragen. Het bevat informatie zoals de openbare sleutel, organisatiegegevens en de domeinnaam waarvoor het certificaat wordt aangevraagd.

De SAN (Subject Alternative Name) is een extensie in het certificaat die extra domeinnamen of IP-adressen specificeert waarvoor het certificaat geldig is. Dit is belangrijk voor het ondersteunen van meerdere domeinen of subdomeinen met één certificaat.

What is a wildcard certificate?

Een wildcard certificate is een type SSL/TLS-certificaat dat kan worden gebruikt om meerdere subdomeinen van een hoofddomein te beveiligen met één enkel certificaat. Het wordt aangeduid met een asterisk (\*) in de domeinnaam, bijvoorbeeld \*.example.com, wat betekent dat het certificaat geldig is voor alle subdomeinen van example.com, zoals www.example.com, mail.example.com, enzovoort.

What file(s) did you add to the browser (or computer) and how?

Om de CA-certificaat toe te voegen aan de browser of computer, heb ik het bestand **ca.crt** geïmporteerd in de vertrouwde rootcertificaten van het besturingssysteem of de browserinstellingen.

Can you easily retrieve your certificates after adding them?

Ja, nadat de certificaten zijn toegevoegd, kunnen ze gemakkelijk worden opgehaald via de certificaatbeheerder van het os of de browser.

Can you review and explain all the files from the OpenVPN lab (see later on in this course) and what they represent? CA? Keys? Certificates?

Ja, in het OpenVPN-labo zijn er verschillende bestanden die elk een specifieke rol spelen:

- ca.crt: Dit is het certificaat van de Certificate Authority (CA) die wordt gebruikt om de identiteit van de VPN-server en -client te verifiëren.
- server.crt: Dit is het certificaat van de VPN-server, ondertekend door de CA.
- server.key: Dit is de private key van de VPN-server, die geheim moet worden gehouden.
- client.crt: Dit is het certificaat van de VPN-client, ondertekend door de CA.

## Labo 7: SIEM

---

Ik begon met een nieuwe map te maken voor de vagrantfile van dit labo in te zetten. In deze map heb ik dan een vagrantfile aangemaakt waarin ik de machine definieer. Ik heb in het provisioningstuk de nodige commando's toegevoegd om Wazuh te installeren. Daarnaast heb ik ook de interne adapter een vast IP-adres gegeven in het subnet van de companyrouter.

Ik voegde de machine toe aan de ssh config file op mijn laptop zodat ik makkelijk kan inloggen op de machine via ssh zonder telkens het hele commando te moeten intypen.

Ik testte nog even of ik alle services zag draaien:

```
[vagrant@wazuh-server ~]$ sudo systemctl status wazuh-manager
sudo systemctl status wazuh-indexer
sudo systemctl status wazuh-dashboard
● wazuh-manager.service - Wazuh manager
    Loaded: loaded (/usr/lib/systemd/system/wazuh-manager.service; enabled;
preset: disabled)
      Active: active (running) since Mon 2025-12-29 09:50:09 UTC; 18min ago
        Process: 836 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start
(code=exited, status=0/SUC>
          Tasks: 153 (limit: 22767)
        Memory: 1.2G (peak: 1.3G)
          CPU: 1min 13.054s
        CGroup: /system.slice/wazuh-manager.service
                  ├─6295 /var/ossec/framework/python/bin/python3
/var/ossec/api/scripts/wazuh_apid.py
                  ├─6309 /var/ossec/framework/python/bin/python3
/var/ossec/api/scripts/wazuh_apid.py
                  ├─6319 /var/ossec/framework/python/bin/python3
/var/ossec/api/scripts/wazuh_apid.py

● wazuh-indexer.service - wazuh-indexer
    Loaded: loaded (/usr/lib/systemd/system/wazuh-indexer.service; enabled;
preset: disabled)
      Active: active (running) since Mon 2025-12-29 09:50:33 UTC; 17min ago
        Docs: https://documentation.wazuh.com
      Main PID: 829 (java)
        Tasks: 73 (limit: 22767)
      Memory: 1.5G (peak: 1.5G)
        CPU: 1min 3.412s
        CGroup: /system.slice/wazuh-indexer.service
                  └─829 /usr/share/wazuh-indexer/jdk/bin/java -Xshare:auto -
Dopensearch.networkaddress.cac>

● wazuh-dashboard.service - wazuh-dashboard
● wazuh-dashboard.service - wazuh-dashboard
    Loaded: loaded (/etc/systemd/system/wazuh-dashboard.service; enabled; preset:
disabled)
      Active: active (running) since Mon 2025-12-29 09:49:38 UTC; 18min ago
● wazuh-dashboard.service - wazuh-dashboard
    Loaded: loaded (/etc/systemd/system/wazuh-dashboard.service; enabled; preset:
disabled)
      Active: active (running) since Mon 2025-12-29 09:49:38 UTC; 18min ago
      Main PID: 730 (node)
        Tasks: 11 (limit: 22767)
● wazuh-dashboard.service - wazuh-dashboard
    Loaded: loaded (/etc/systemd/system/wazuh-dashboard.service; enabled; preset:
```

```
disabled)
  Active: active (running) since Mon 2025-12-29 09:49:38 UTC; 18min ago
    Main PID: 730 (node)
      Tasks: 11 (limit: 22767)
      Memory: 274.9M (peak: 337.2M)
        CPU: 12.346s
      CGroup: /system.slice/wazuh-dashboard.service
              └─730 /usr/share/wazuh-dashboard/node/bin/node /usr/share/wazuh-
                dashboard/src/cli/dist

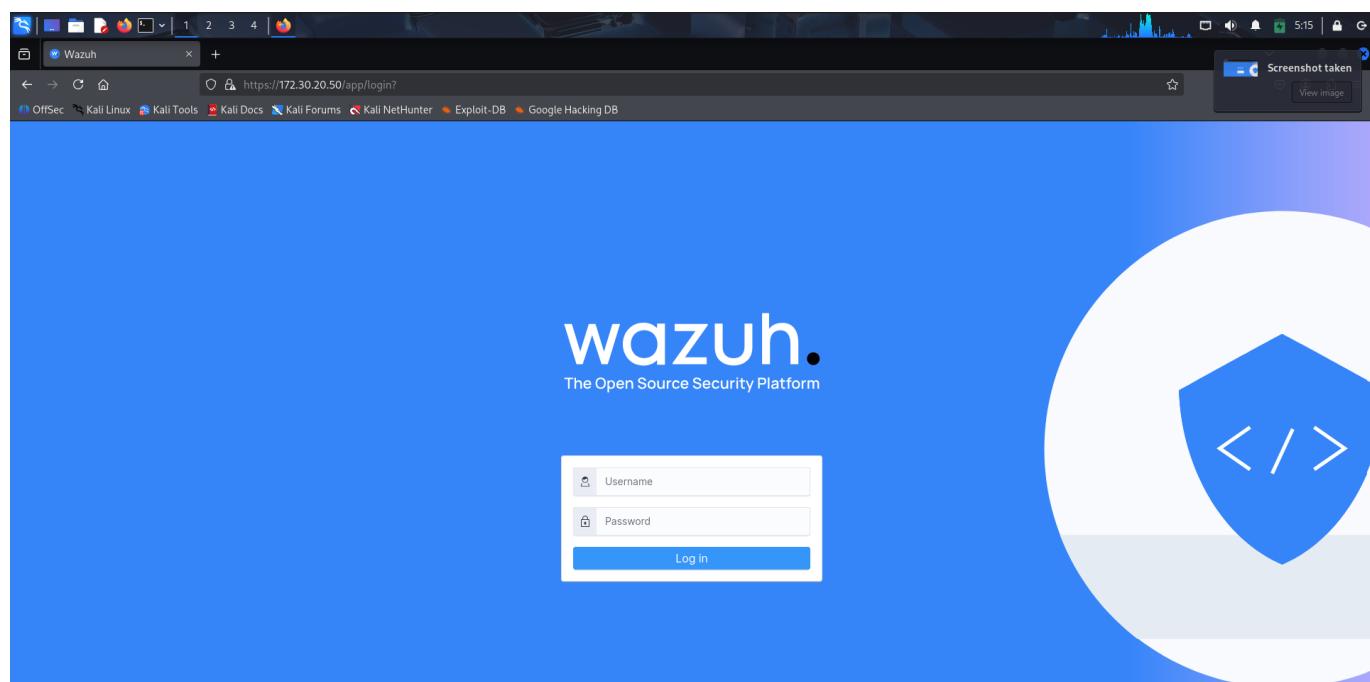
Dec 29 09:50:49 wazuh-server opensearch-dashboards[730]:
{"type":"log","@timestamp":"2025-12-29T09:50>
Dec 29 09:50:49 wazuh-server opensearch-dashboards[730]:
{"type":"log","@timestamp":"2025-12-29T09:50>
Dec 29 09:50:50 wazuh-server opensearch-dashboards[730]:
{"type":"log","@timestamp":"2025-12-29T09:50>
lines 1-17...skipping...
● wazuh-dashboard.service - wazuh-dashboard
  Loaded: loaded (/etc/systemd/system/wazuh-dashboard.service; enabled; preset:
  disabled)
    Active: active (running) since Mon 2025-12-29 09:49:38 UTC; 18min ago
      Main PID: 730 (node)
        Tasks: 11 (limit: 22767)
        Memory: 274.9M (peak: 337.2M)
          CPU: 12.346s
        CGroup: /system.slice/wazuh-dashboard.service
                └─730 /usr/share/wazuh-dashboard/node/bin/node /usr/share/wazuh-
                  dashboard/src/cli/dist

Dec 29 09:50:49 wazuh-server opensearch-dashboards[730]:
{"type":"log","@timestamp":"2025-12-29T09:50>
Dec 29 09:50:49 wazuh-server opensearch-dashboards[730]:
{"type":"log","@timestamp":"2025-12-29T09:50>
Dec 29 09:50:50 wazuh-server opensearch-dashboards[730]:
{"type":"log","@timestamp":"2025-12-29T09:50>
Dec 29 09:55:01 wazuh-server opensearch-dashboards[730]:
{"type":"log","@timestamp":"2025-12-29T09:55>
lines 1-18...skipping...
```

```
● wazuh-dashboard.service - wazuh-dashboard
   Loaded: loaded (/etc/systemd/system/wazuh-dashboard.service; enabled; preset: disabled)
     Active: active (running) since Mon 2025-12-29 09:49:38 UTC; 18min ago
       Main PID: 730 (node)
          Tasks: 11 (limit: 22767)
        Memory: 274.9M (peak: 337.2M)
         CPU: 12.346s
        CGroup: /system.slice/wazuh-dashboard.service
                  └─730 /usr/share/wazuh-dashboard/node/bin/node /usr/share/wazuh-dashboard/src/cli/dist

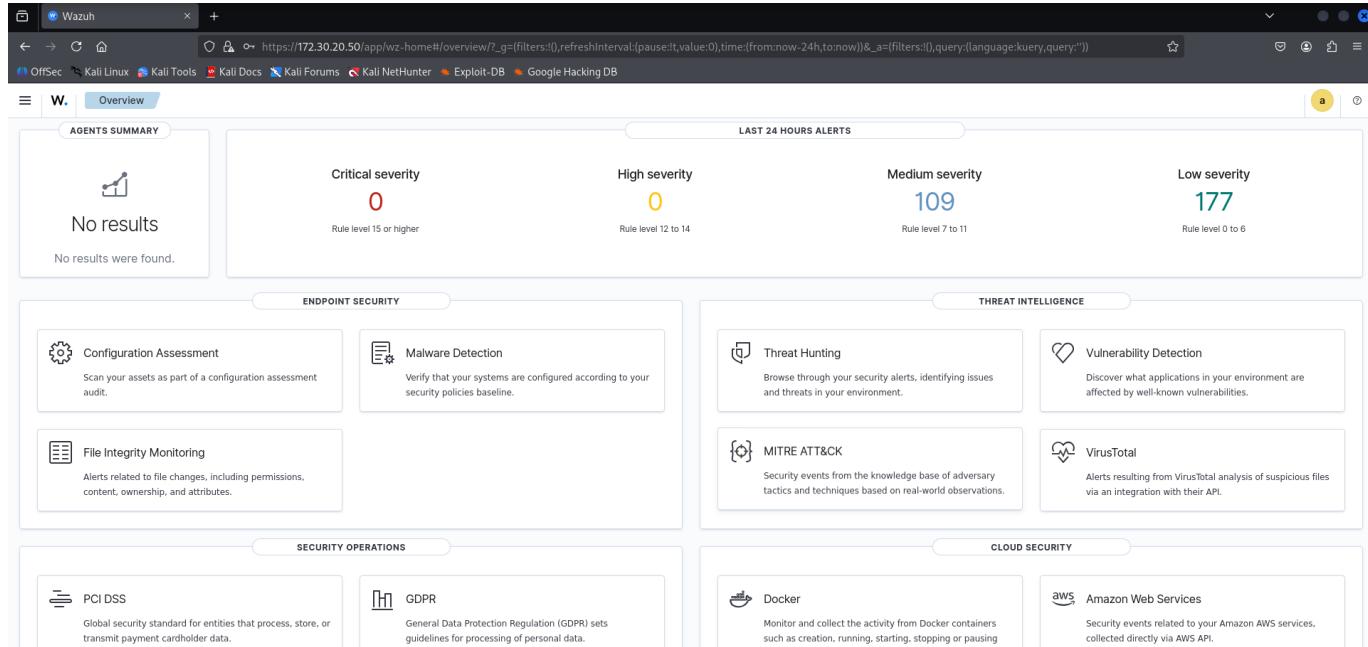
Dec 29 09:50:49 wazuh-server opensearch-dashboards[730]: {"type": "log", "@timestamp": "2025-12-29T09:50>
Dec 29 09:50:49 wazuh-server opensearch-dashboards[730]: {"type": "log", "@timestamp": "2025-12-29T09:50>
Dec 29 09:50:50 wazuh-server opensearch-dashboards[730]: {"type": "log", "@timestamp": "2025-12-29T09:50>
Dec 29 09:50:50 wazuh-server opensearch-dashboards[730]: {"type": "log", "@timestamp": "2025-12-29T09:50>
Dec 29 09:50:50 wazuh-server opensearch-dashboards[730]: {"type": "log", "@timestamp": "2025-12-29T09:50>
Dec 29 09:50:50 wazuh-server opensearch-dashboards[730]: {"type": "log", "@timestamp": "2025-12-29T09:50>
Dec 29 09:50:50 wazuh-server opensearch-dashboards[730]: {"type": "log", "@timestamp": "2025-12-29T09:50>
Dec 29 09:55:01 wazuh-server opensearch-dashboards[730]: {"type": "log", "@timestamp": "2025-12-29T09:55>
Dec 29 09:55:01 wazuh-server opensearch-dashboards[730]: {"type": "log", "@timestamp": "2025-12-29T09:55>
```

We kunnen het wazuh dashboard bereiken door in een browser naar het IP-adres van de SIEM server:



Ik melde mij aan met het admin account en het wachtwoord dat werd aangemaakt tijdens de installatie.

Ik kreeg het dashboard te zien:



TzGHeQOZyZ0rmcBL?43oykJyEFuzfxmP

Hierna begon ik met het toevoegen van de agents aan alle almalinux machines. Dit deed ik door in het dashboard naar server management te gaan en daar op add agent te klikken. Ik volgde de stappen in de wizard om de agent te installeren op de verschillende machines. Na het toevoegen van de agents kon ik in het dashboard zien dat ze verbonden waren.

Voorbeeld op de companyrouter:

The screenshot shows the 'Deploy new agent' wizard step 1: Server address. The steps are as follows:

- Server address:** This is the address the agent uses to communicate with the server. Enter an IP address or a fully qualified domain name (FQDN).  
Assign a server address: 172.30.255.254  
Remember server address:
- Optional settings:** By default, the deployment uses the hostname as the agent name. Optionally, you can use a different agent name in the field below.  
Assign an agent name: Companyrouter  
The agent name must be unique. It can't be changed once the agent has been enrolled.
- Select one or more existing groups:** Default
- Run the following commands to download and install the agent:**

Op de homerouter voerde ik dan nog de volgende commando's uit om de agent te installeren en te verbinden met de SIEM server:

```
curl -o wazuh-agent-4.9.2-1.x86_64.rpm https://packages.wazuh.com/4.x/yum/wazuh-agent-4.9.2-1.x86_64.rpm && sudo WAZUH_MANAGER='172.30.20.50'
```

```
WAZUH_AGENT_NAME='Homerouter' rpm -ihv wazuh-agent-4.9.2-1.x86_64.rpm
```

```
sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
```

Ik zag deze meteen verschijnen in het dashboard:

ID	Name	IP address	Group(s)	Operating system	Cluster node	Version	Status	Actions
001	Homerouter	192.168.62.42	default	AlmaLinux 9.6	node01	v4.9.2	active	...

Ik deed hetzelfde voor alle andere almalinux machines.

ID	Name	IP address	Group(s)	Operating system	Cluster node	Version	Status	Actions
001	Homerouter	192.168.62.42	default	AlmaLinux 9.6	node01	v4.9.2	active	...
002	Companyrouter	172.30.255.254	default	AlmaLinux 9.6	node01	v4.9.2	active	...
003	Web	172.30.10.10	default	AlmaLinux 9.7	node01	v4.9.2	disconnected	...
004	Remote-employee	172.10.10.123	default	AlmaLinux 9.6	node01	v4.9.2	active	...

(web is niet beschikbaar omdat ik deze had uitgeschakeld om wat ram te besparen)

Ik recupererde een windows client van het vak windows server. Ik stopte ook deze in het netwerk. Ik voegde de agent toe op dezelfde manier als bij de almalinux machines.

## File Integrity Monitoring

Ik voegde in de /var/ossec/etc/ossec.conf van de SIEM server de volgende regels toe om file integrity monitoring in te schakelen:

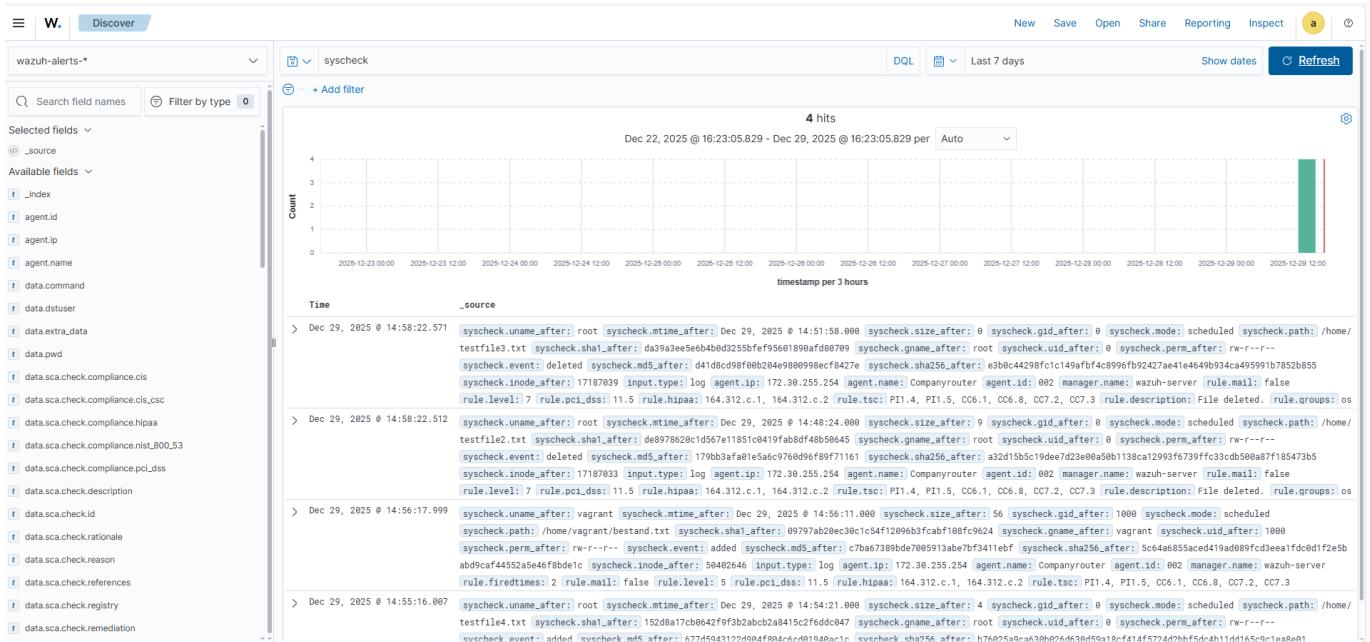
```
<directories check_all="yes">/home</directories>
```

Hierdoor worden alle bestanden in de home directory van alle gebruikers gemonitord op wijzigingen.

We doen een klein testje door een bestand aan te maken in de home directory van de vagrant gebruiker op de companyrouter:

```
touch /home/testfile.txt
echo "test" >> /home/testfile.txt
rm /home/testfile.txt
```

Ik had veel problemen om de logs zichtbaar te krijgen in het dashboard, dit kwam doordat mijn VM een schijf had van slechts 20GB en de indexer hierdoor vastliep. Na het vergroten van de schijf naar 50GB werkte alles terug naar behoren.



## Windows endpoint detection

Hierna ging ik verder met het installeren van sysmon op de windows client om zo windows endpoint detection in te schakelen. Ik downloadde sysmon van de microsoft website en installeerde het met de volgende commando's.

Ik downloade eerst de sysmon zip file en pakte deze uit:

<https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon>

Deze pakte ik uit in de map C:\Tools\Sysmon

Ik downloade ook een config file voor sysmon op github: <https://github.com/SwiftOnSecurity/sysmon-config>

Nu voerde ik het volgende commando uit in een powershell terminal met administrator rechten om sysmon te installeren met de config file:

```
PS C:\Tools\Sysmon>
>> .\Sysmon64.exe -accepteula -i sysmonconfig.xml

System Monitor v15.15 - System activity monitor
By Mark Russinovich and Thomas Garnier
Copyright (C) 2014-2024 Microsoft Corporation
Using libxml2. libxml2 is Copyright (C) 1998-2012 Daniel Veillard. All Rights Reserved.
Sysinternals - www.sysinternals.com

Loading configuration file with schema version 4.50
Sysmon schema version: 4.90
Configuration file validated.
Sysmon64 installed.
SysmonDrv installed.
Starting SysmonDrv.
SysmonDrv started.
Starting Sysmon64..
Sysmon64 started.
```

In de event viewer kon ik nu de sysmon logs zien onder Applications and Services Logs -> Microsoft -> Windows -> Sysmon -> Operational

Je ziet dat ik een notepad opgestart heb.

Event ID	Task Category
1	Process Create (rule:ProcessCreate)
22	Dns query (rule:DnsQuery)
1	Process Create (rule:ProcessCreate)
1	Process Create (rule:ProcessCreate)
13	Registry value set (rule:RegistryEvent)
4	Sysmon service state changed
16	Sysmon config state changed

## Powershell logging

Om powershell logging in te schakelen ging ik naar de local group policy editor door gpedit.msc uit te voeren in het startmenu.

onder Computer Configuration -> Administrative Templates -> Windows Components -> Windows PowerShell zette ik de volgende policies op enabled:

- Module Logging
- Script Block Logging

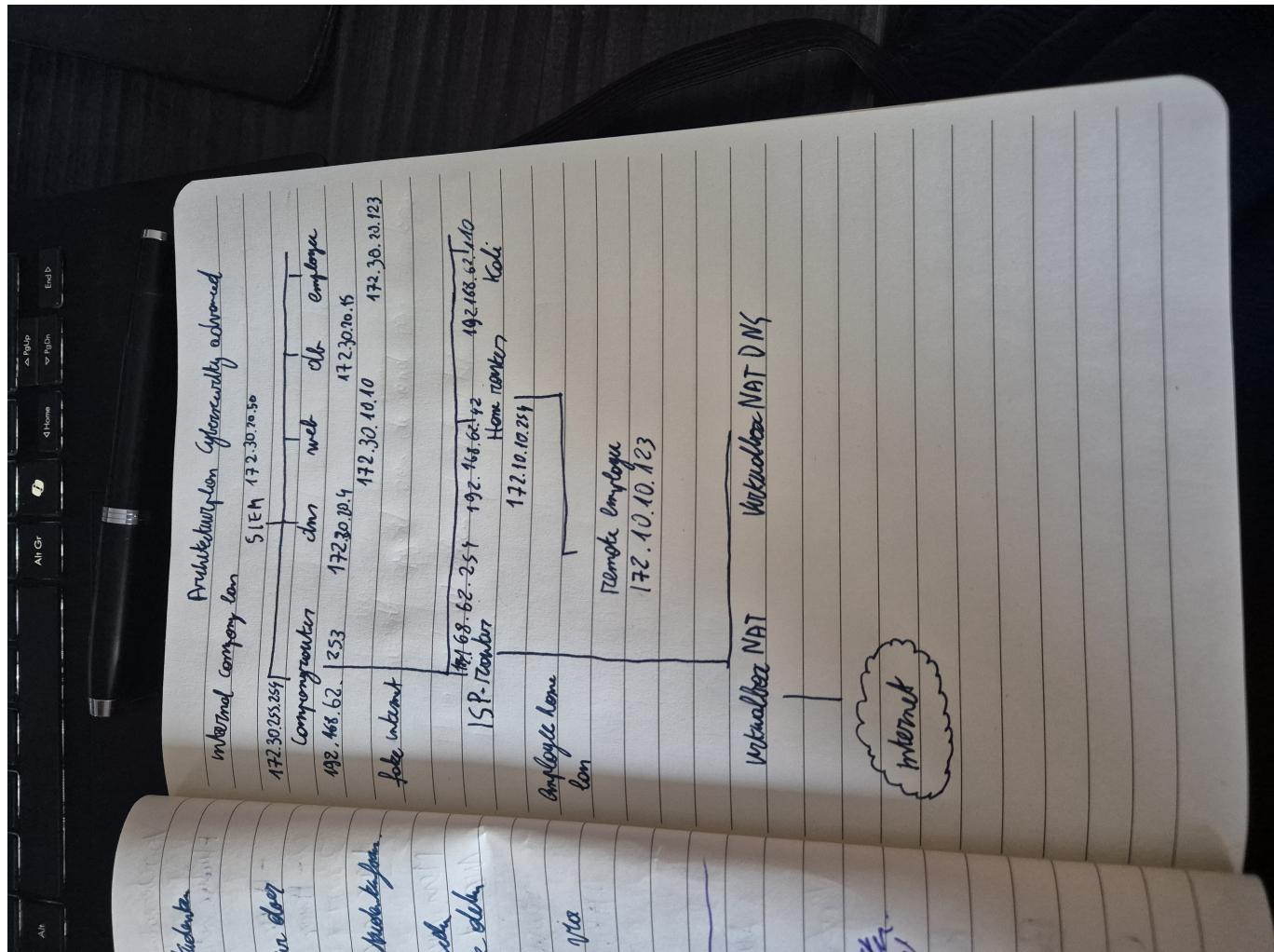
Setting	State	Comment
Turn on Module Logging	Enabled	No
Turn on PowerShell Script Block Logging	Enabled	No
Turn on Script Execution	Not configured	No
Turn on PowerShell Transcription	Not configured	No
Set the default source path for Update-Help	Not configured	No

We zien deze logs nu verschijnen onder threat hunting in het wazuh dashboard:

924 hits Dec 23, 2025 @ 09:30:53.957 - Dec 30, 2025 @ 09:30:53.957				
timestamp	agent.name	rule.description	rule.level	rule.id
Dec 30, 2025 @ 09:30:18.590	Windows-client	Executable file dropped in folder commonly used by user	15	92213
Dec 30, 2025 @ 09:29:05.845	Windows-client	PowerShell process created an executable file in Win32	9	92205
Dec 30, 2025 @ 09:29:04.715	Windows-client	C:\Windows\SysWOW64\SecEdit.exe binary in a suspicious location	4	92066
Dec 30, 2025 @ 09:29:04.673	Windows-client	PowerShell process created an executable file in Win32	9	92205
Dec 30, 2025 @ 09:29:03.749	Windows-client	Discovery activity executed	3	92031
Dec 30, 2025 @ 09:29:03.733	Windows-client	Discovery activity executed	3	92031
Dec 30, 2025 @ 09:29:03.717	Windows-client	A net.exe account discovery command was initiated	3	92039
Dec 30, 2025 @ 09:29:03.717	Windows-client	A net.exe account discovery command was initiated	3	92039
Dec 30, 2025 @ 09:29:03.685	Windows-client	A net.exe account discovery command was initiated	3	92039
Dec 30, 2025 @ 09:29:03.666	Windows-client	A net.exe account discovery command was initiated	3	92039
Dec 30, 2025 @ 09:29:03.650	Windows-client	A net.exe account discovery command was initiated	3	92039
Dec 30, 2025 @ 09:29:03.650	Windows-client	A net.exe account discovery command was initiated	3	92039
Dec 30, 2025 @ 09:29:03.636	Windows-client	A net.exe account discovery command was initiated	3	92039
Dec 30, 2025 @ 09:29:03.621	Windows-client	A net.exe account discovery command was initiated	3	92039
Dec 30, 2025 @ 09:29:03.580	Windows-client	Discovery activity executed	3	92031

## Lab 8: IPsec

Ik begon met het tekenen van een netwerkdiagram om de verschillende componenten en hun verbindingen in het IPsec-lab te visualiseren. Dit hielp me om een duidelijk overzicht te krijgen van hoe de verschillende apparaten met elkaar communiceren.



Hierna sshde ik naar de homerouter om de routing aan te zetten.

Ik maakte ook een route aan naar het fake internet:

```
sudo ip route add 172.30.0.0/16 via 192.168.62.253
```

Ik controleerde ook of het verkeer niet via de ISP-router ging:

```
[vagrant@homeroouter ~]$ traceroute 172.30.20.4
traceroute to 172.30.20.4 (172.30.20.4), 30 hops max, 60 byte packets
1 * *
2 192.168.62.253 (192.168.62.253) 0.363 ms 0.402 ms 0.381 ms
3 172.30.20.4 (172.30.20.4) 0.685 ms 0.637 ms 0.619 ms
```

## MitM attack

Hierna startte ik de kali VM op om de Man in the Middle attack uit te voeren. Ik gebruikte hiervoor het volgende commando:

```
sudo ettercap -Tq -i eth0 -M arp:remote /192.168.62.42// /192.168.62.253//
```

We starten ook wireshark om het verkeer te monitoren met het commando:

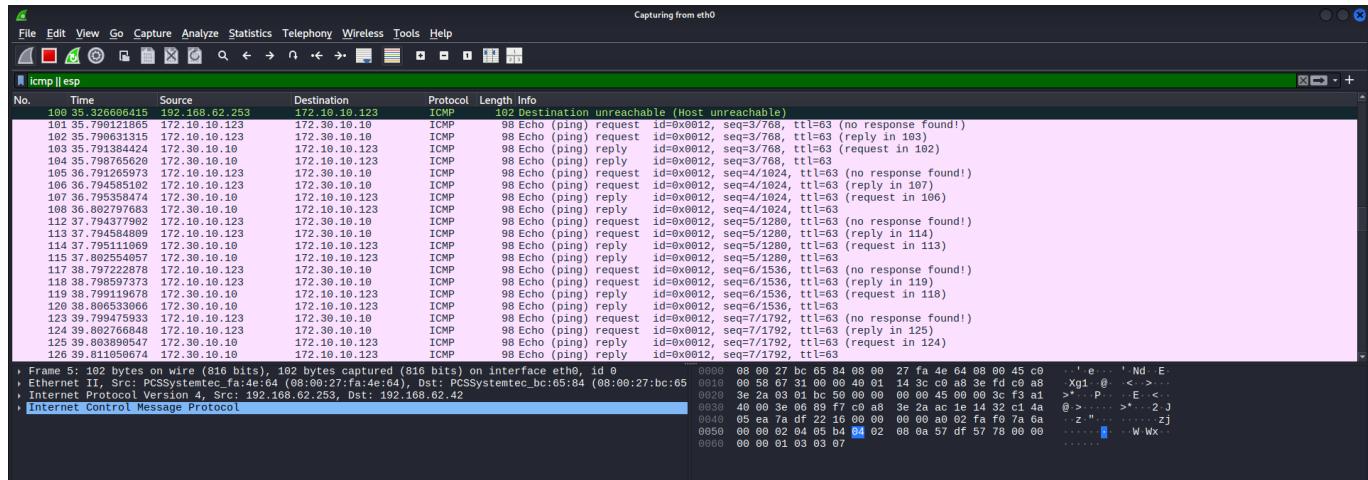
```
sudo wireshark &
```

We zetten een filter op:

icmp || esp

en nu pingen we van de remote-employee naar de webserver:

```
ping 172.30.10.10
```



## IPsec setup

Hierna maakte ik een bestand aan op de homerouter om het script uit de opgave in te plakken en te kunnen uitvoeren op de vm

```
#!/usr/bin/env sh

# 1. Opschonen
sudo ip xfrm state flush
sudo ip xfrm policy flush

# 2. Variabelen (Zorg dat deze exact hetzelfde zijn op beide routers)
SPI_HOME_TO_COMP=0x007
KEY_HOME_TO_COMP=0xFEDCBA9876543210FEDCBA9876543210

SPI_COMP_TO_HOME=0x008
KEY_COMP_TO_HOME=0x8c51dbe8f193bdddddf5d91efd3f7b9fb

# 3. States (SA) - Hoe we versleutelen en ontsleutelen
# Uitgaand naar Company
sudo ip xfrm state add src 192.168.62.42 dst 192.168.62.253 proto esp spi
```

```

$SPI_HOME_TO_COMP mode tunnel enc aes $KEY_HOME_TO_COMP
# Inkomend van Company
sudo ip xfrm state add src 192.168.62.253 dst 192.168.62.42 proto esp spi
$SPI_COMP_TO_HOME mode tunnel enc aes $KEY_COMP_TO_HOME

# 4. Policies (SP) - Welk verkeer moet de tunnel in/uit
# OUT: Van Home-LAN naar Company-LAN
sudo ip xfrm policy add src 172.10.10.0/24 dst 172.30.0.0/16 dir out tmpl src
192.168.62.42 dst 192.168.62.253 proto esp spi $SPI_HOME_TO_COMP mode tunnel
# FWD/IN: Van Company-LAN naar Home-LAN (doorsturen naar de employee)
sudo ip xfrm policy add src 172.30.0.0/16 dst 172.10.10.0/24 dir fwd tmpl src
192.168.62.253 dst 192.168.62.42 proto esp spi $SPI_COMP_TO_HOME mode tunnel
sudo ip xfrm policy add src 172.30.0.0/16 dst 172.10.10.0/24 dir in tmpl src
192.168.62.253 dst 192.168.62.42 proto esp spi $SPI_COMP_TO_HOME mode tunnel

echo "Homerouter IPsec configuratie voltooid."

```

Ik paste dit script aan om het ook op de companyrouter te kunnen uitvoeren:

```

#!/usr/bin/env sh

# 1. Opschonen
sudo ip xfrm state flush
sudo ip xfrm policy flush

# 2. Variabelen (Identiek aan Homerouter)
SPI_HOME_TO_COMP=0x007
KEY_HOME_TO_COMP=0xFEDCBA9876543210FEDCBA9876543210

SPI_COMP_TO_HOME=0x008
KEY_COMP_TO_HOME=0x8c51dbe8f193bdddf5d91efd3f7b9fb

# 3. States (SA)
# Inkomend van Home
sudo ip xfrm state add src 192.168.62.42 dst 192.168.62.253 proto esp spi
$SPI_HOME_TO_COMP mode tunnel enc aes $KEY_HOME_TO_COMP
# Uitgaand naar Home
sudo ip xfrm state add src 192.168.62.253 dst 192.168.62.42 proto esp spi
$SPI_COMP_TO_HOME mode tunnel enc aes $KEY_COMP_TO_HOME

# 4. Policies (SP)
# OUT: Van Company-LAN naar Home-LAN
sudo ip xfrm policy add src 172.30.0.0/16 dst 172.10.10.0/24 dir out tmpl src
192.168.62.253 dst 192.168.62.42 proto esp spi $SPI_COMP_TO_HOME mode tunnel
# FWD/IN: Van Home-LAN naar Company-LAN (doorsturen naar de webserver)
sudo ip xfrm policy add src 172.10.10.0/24 dst 172.30.0.0/16 dir fwd tmpl src
192.168.62.42 dst 192.168.62.253 proto esp spi $SPI_HOME_TO_COMP mode tunnel
sudo ip xfrm policy add src 172.10.10.0/24 dst 172.30.0.0/16 dir in tmpl src
192.168.62.42 dst 192.168.62.253 proto esp spi $SPI_HOME_TO_COMP mode tunnel

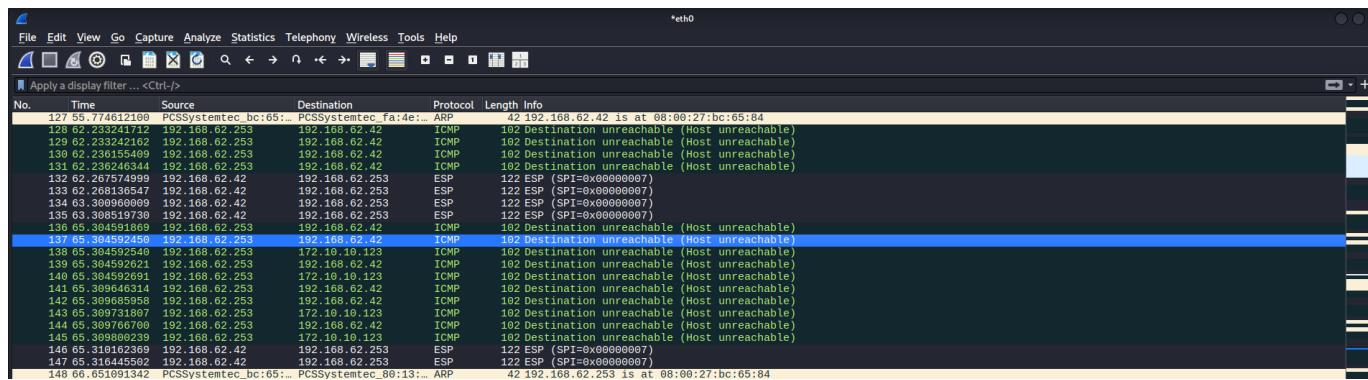
echo "Companyrouter IPsec configuratie voltooid."

```

Nu kunnen we onopgemerkt pingen van de remote-employee naar de webserver:

```
[vagrant@remote-employee ~]$ ping 192.168.62.42
PING 192.168.62.42 (192.168.62.42) 56(84) bytes of data.
64 bytes from 192.168.62.42: icmp_seq=1 ttl=64 time=0.166 ms
64 bytes from 192.168.62.42: icmp_seq=2 ttl=64 time=0.389 ms
64 bytes from 192.168.62.42: icmp_seq=3 ttl=64 time=0.324 ms
64 bytes from 192.168.62.42: icmp_seq=4 ttl=64 time=0.371 ms
64 bytes from 192.168.62.42: icmp_seq=5 ttl=64 time=0.397 ms
^C
--- 192.168.62.42 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4077ms
rtt min/avg/max/mdev = 0.166/0.329/0.397/0.085 ms
```

We zien in wireshark enkel ESP verkeer langskomen:



Ik voegde in de firewall op de companyrouter nog enkele regels toe om het verkeer toe te laten:

Deze om het verkeer van de remote-employee naar de intranet en dmz toe te laten: ip saddr 192.168.62.42 ip protocol esp accept

En deze om het antwoordverkeer van de intranet en dmz naar de remote-employee toe te laten:

ip saddr 172.10.10.0/24 ip daddr { \$dmz\_net, \$intranet\_net } accept

## Decryption

Decryption!!

You are using static keys, opposed to what IKE would organize for you. But knowing the key, and the SPI: can you decrypt the traffic you have captured in Wireshark?

Ja, wireshark heeft een ingebouwde functionaliteit om IPsec ESP-verkeer te decoderen als je de juiste sleutels en SPI's hebt.

We gaan naar de ESP instellingen in Wireshark:

1. Open Wireshark
2. Ga naar Edit -> Preferences.

### 3. Open ESP

Nu moeten we de sleutels toevoegen:

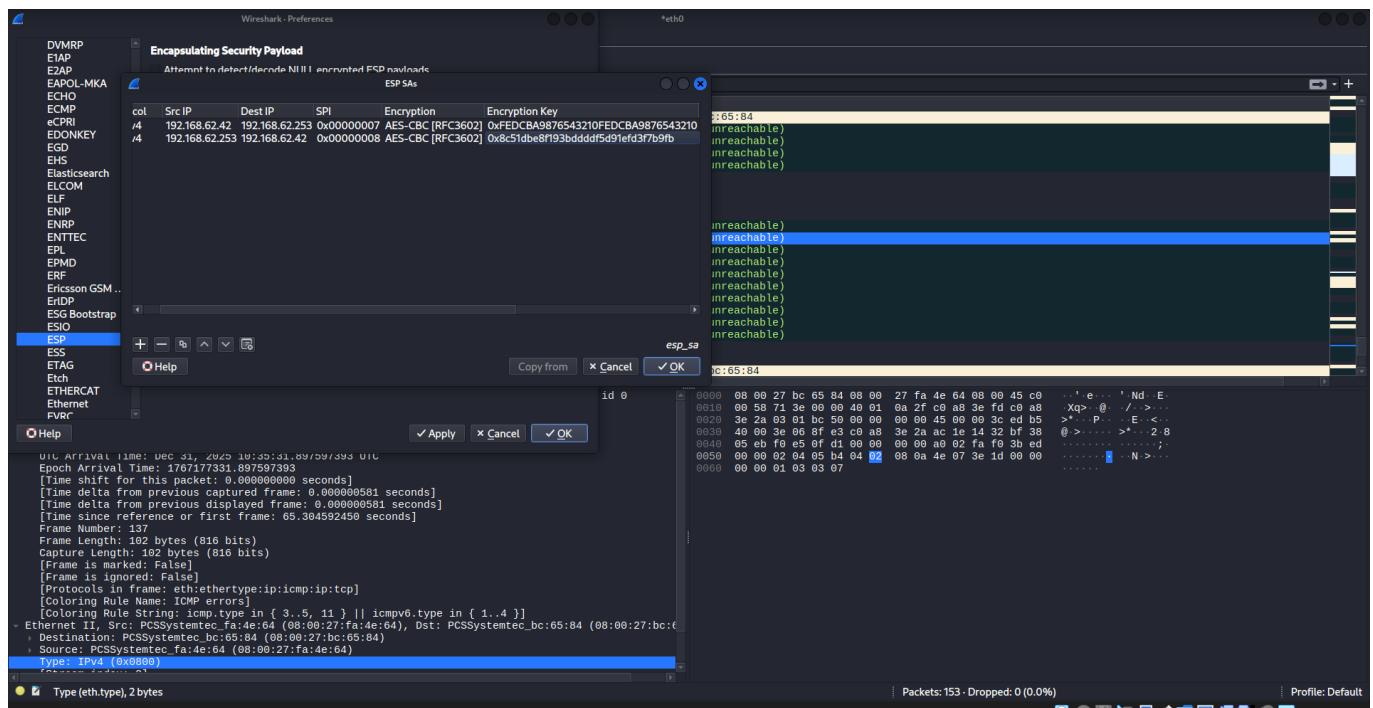
4. Vink het vakje Attempt to detect/decode encrypted ESP payloads aan.
5. Klik op de knop Edit... naast ESP SAs.
6. Klik op het + icoontje om een nieuwe SA toe te voegen.

Vul de volgende gegevens in voor de eerste richting (Home -> Company):

- Src IP: 192.168.62.42
- Dest IP: 192.168.62.253
- SPI: 0x00000007 (of 7)
- Encryption Algorithm: AES-CBC [RFC3602]
- Encryption Key: 0xFEDCBA9876543210FEDCBA9876543210 (zonder de 0x als Wireshark klaagt)
- Authentication Algorithm: NULL (we hebben in het script geen auth gedefinieerd).

Herhaal dit voor de tweede richting (Company -> Home):

- Src IP: 192.168.62.253
- Dest IP: 192.168.62.42
- SPI: 0x00000008
- Encryption Algorithm: AES-CBC [RFC3602]
- Encryption Key: 0x8c51dbe8f193bdddd5d91efd3f7b9fb



Nu is het verkeer gedecodeerd en kunnen we de ICMP-pakketten zien:

No.	Time	Source	Destination	Protocol	Length	Info	
26	3.292066780	192.168.62.253	172.10.10.123	ICMP	102	Destination unreachable (Host unreachable)	
27	3.299886608	192.168.62.253	192.168.62.42	ICMP	102	Destination unreachable (Host unreachable)	
28	3.299955658	192.168.62.253	172.10.10.123	ICMP	102	Destination unreachable (Host unreachable)	
29	3.300019678	192.168.62.253	192.168.62.42	ICMP	102	Destination unreachable (Host unreachable)	
30	3.300041933	192.168.62.253	172.10.10.123	ICMP	102	Destination unreachable (Host unreachable)	
31	3.309451628	172.10.10.123	172.30.20.58	TCP	122	[TCP Retransmission] 50992 -> 1514 [SYN] Seq=0 Win=64240 MSS=1460 SACK_PERM TStamp=2457456840 TSecr=0 WS=128	
32	3.309722385	172.10.10.123	172.30.20.58	TCP	122	[TCP Retransmission] 50992 -> 1514 [SYN] Seq=0 Win=64240 MSS=1460 SACK_PERM TStamp=2457456840 TSecr=0 WS=128	
33	4.004516991	172.10.10.123	172.30.10.10	ICMP	154	Echo (ping) request id=0x000e, seq=12/3072, ttl=63 (no response found!)	
34	4.007938312	172.10.10.123	172.30.10.10	ICMP	154	Echo (ping) request id=0x000e, seq=12/3072, ttl=63 (reply in 35)	
35	4.008471997	172.30.10.10	172.10.10.123	ICMP	154	Echo (ping) reply id=0x000e, seq=12/3072, ttl=63 (request in 34)	
36	4.015925498	172.39.10.10	172.10.10.123	ICMP	154	Echo (ping) reply id=0x000e, seq=12/3072, ttl=63	
37	4.016000000	00:0c:29:9d:bc:65:..	172.10.10.123	ARP	40	Request sender hardware address is 00:0c:29:9d:bc:65, target hardware address is 00:00:27:faf:4e:6a	
38	4.016000000	00:0c:29:9d:bc:65:..	PCSSystemtec_fa:4e:..	PCSSystemtec_bc:65:..	ICMP	60	192.168.62.253 is 00:00:27:faf:4e:6a
39	5.006362801	172.10.10.123	172.30.10.10	ICMP	154	Echo (ping) request id=0x000e, seq=13/3328, ttl=63 (no response found!)	
40	5.011966646	172.10.10.123	172.30.10.10	ICMP	154	Echo (ping) request id=0x000e, seq=13/3328, ttl=63 (reply in 41)	
41	5.012649468	172.30.10.10	172.10.10.123	ICMP	154	Echo (ping) reply id=0x000e, seq=13/3328, ttl=63 (request in 40)	
42	5.020015764	172.30.10.10	172.10.10.123	ICMP	154	Echo (ping) reply id=0x000e, seq=13/3328, ttl=63	
43	6.008500668	172.10.10.123	172.30.10.10	ICMP	154	Echo (ping) request id=0x000e, seq=14/3584, ttl=63 (no response found!)	
44	6.0120988519	172.10.10.123	172.30.10.10	ICMP	154	Echo (ping) request id=0x000e, seq=14/3584, ttl=63 (reply in 45)	
45	6.01261019	172.30.10.10	172.10.10.123	ICMP	154	Echo (ping) reply id=0x000e, seq=14/3584, ttl=63 (request in 44)	
46	6.019982499	172.30.10.10	172.10.10.123	ICMP	154	Echo (ping) reply id=0x000e, seq=14/3584, ttl=63	

## Lab 9: VPN

Ik begon met openvpn te installeren op de companyrouter.

```
sudo dnf install epel-release -y
sudo dnf install --assumeyes openvpn easy-rsa
```

Vervolgens controleerde ik of de installatie gelukt was:

```
[vagrant@companyrouter ~]$ openvpn --version
OpenVPN 2.5.11 x86_64-redhat-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL]
[PKCS11] [MH/PKTINFO] [AEAD] built on Jul 18 2024
library versions: OpenSSL 3.2.2 4 Jun 2024, LZO 2.10
Originally developed by James Yonan
Copyright (C) 2002-2022 OpenVPN Inc <sales@openvpn.net>
```

```
[vagrant@companyrouter ~]$ sudo /usr/share/easy-rsa/3/easyrsa --version
EasyRSA Version Information
Version:      3.2.1
Generated:    Fri Sep 13 13:04:18 CDT 2024
SSL Lib:      OpenSSL 3.2.2 4 Jun 2024 (Library: OpenSSL 3.2.2 4 Jun 2024)
Git Commit:   3f60a68702713161ab44f9dd80ce01f588ca49ac
Source Repo:  https://github.com/OpenVPN/easy-EasyRSA
```

## PKI opzetten

Hierna initialieerde ik de PKI omgeving:

```
/usr/share/easy-rsa/3/easyrsa init-pki
```

Ik maakte een nieuwe CA aan:

```
/usr/share/easy-rsa/3/easyrsa build-ca nopass
```

Als hij om een naam vroeg klikte ik gewoon op enter zodat hij de standaard naam gebruikte.

Vervolgens maakte ik een server certificaat aan:

```
/usr/share/easy-rsa/3/easyrsa gen-req server nopass  
/usr/share/easy-rsa/3/easyrsa sign-req server server
```

Daarna genereerde ik het client certificaat:

```
/usr/share/easy-rsa/3/easyrsa gen-req client nopass  
/usr/share/easy-rsa/3/easyrsa sign-req client client
```

Hierna maakte ik de Diffie-Hellman parameters aan zodat de sleutels veilig uitgewisseld konden worden:

```
/usr/share/easy-rsa/3/easyrsa gen-dh
```

## Server configureren

Ik kopieerde het voorbeeldconfiguratiebestand naar de /etc/openvpn map:

```
sudo cp /usr/share/doc/openvpn/sample/sample-config-files/server.conf  
/etc/openvpn/server.conf
```

We openen het bestand met nano:

```
sudo nano /etc/openvpn/server.conf
```

We zoeken met CTRL+W naar de volgende regels en passen ze aan zodat ze er als volgt uitzien:

```
;local a.b.c.d
```

Dit vervangen we door:

```
local 192.168.62.253
```

Ik zette ook de paden naar de certificaten en sleutels goed:

```
ca /home/vagrant/pki/ca.crt
cert /home/vagrant/pki/issued/server.crt
key /home/vagrant/pki/private/server.key
dh /home/vagrant/pki/dh.pem
```

Daarna stelde ik nog de routes in op basis van mijn netwerkopstelling.

Ik veranderde de volgende regel:

```
;push "route 192.168.10.0 255.255.255.0"
```

Naar:

```
push "route 172.30.0.0 255.255.0.0"
```

Voor topology haalde ik ook nog de punt komma weg, dit zorgt dat clients een IP adres krijgen uit hetzelfde subnet als de server:

```
topology subnet
```

## Client configureren

Ik begon met openvpn te installeren op de client machine:

```
sudo dnf install epel-release -y
sudo dnf install openvpn
```

Op de companyrouter gaf ik de juiste permissies aan de client key zodat de client deze ook kon lezen

```
sudo chmod 644 /home/vagrant/pki/private/client.key
```

Op de remote employee machine maakte ik een map aan voor de vpn bestanden:

```
mkdir ~/openvpn
cd ~/openvpn
```

Daarna kopierte ik de volgende bestanden van de companyrouter naar de remote employee machine:

```
scp vagrant@192.168.62.253:/home/vagrant/pki/ca.crt .
scp vagrant@192.168.62.253:/home/vagrant/pki/issued/client.crt .
scp vagrant@192.168.62.253:/home/vagrant/pki/private/client.key .
```

Hierna kopierte ik de voorbeeldconfiguratie naar de map:

```
cp /usr/share/doc/openvpn/sample/sample-config-files/client.conf
~/openvpn/client.conf
```

Ik paste de volgende regels aan in het configuratiebestand:

```
remote my-server-1 1194
```

Deze veranderde ik naar het IP adres van de companyrouter:

```
remote 192.168.62.253 1194
```

Ik zette ook de paden naar de certificaten en sleutels goed:

```
ca /home/vagrant/openvpn/ca.crt
cert /home/vagrant/openvpn/client.crt
key /home/vagrant/openvpn/client.key
```

Ik voegde ook nog de regel:

```
auth-user-pass
```

en zette de volgende regel uit:

```
;tls-auth ta.key 1
```

## VPN starten

Op de companyrouter startte ik de openvpn server:

```
cd /etc/openvpn
sudo openvpn server.conf
```

De VPN server is nu gestart.

```
[vagrant@companyrouter openvpn]$ sudo openvpn server.conf
2026-01-01 09:46:39 DEPRECATED OPTION: --cipher set to 'AES-256-CBC' but missing
in --data-ciphers (AES-256-GCM:AES-128-GCM). Future OpenVPN version will ignore --
cipher for cipher negotiations. Add 'AES-256-CBC' to --data-ciphers or change --
cipher 'AES-256-CBC' to --data-ciphers-fallback 'AES-256-CBC' to silence this
warning.
2026-01-01 09:46:39 OpenVPN 2.5.11 x86_64-redhat-linux-gnu [SSL (OpenSSL)] [LZO]
[LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] built on Jul 18 2024
2026-01-01 09:46:39 library versions: OpenSSL 3.2.2 4 Jun 2024, LZO 2.10
2026-01-01 09:46:39 net_route_v4_best_gw query: dst 0.0.0.0
2026-01-01 09:46:39 net_route_v4_best_gw result: via 10.0.2.2 dev eth0
2026-01-01 09:46:39 Diffie-Hellman initialized with 2048 bit key
2026-01-01 09:46:39 TUN/TAP device tun0 opened
2026-01-01 09:46:39 net_iface_mtu_set: mtu 1500 for tun0
2026-01-01 09:46:39 net_iface_up: set tun0 up
2026-01-01 09:46:39 net_addr_v4_add: 10.8.0.1/24 dev tun0
2026-01-01 09:46:39 Could not determine IPv4/IPv6 protocol. Using AF_INET
2026-01-01 09:46:39 Socket Buffers: R=[212992->212992] S=[212992->212992]
2026-01-01 09:46:39 UDPv4 link local (bound): [AF_INET]192.168.62.253:1194
2026-01-01 09:46:39 UDPv4 link remote: [AF_UNSPEC]
2026-01-01 09:46:39 MULTI: multi_init called, r=256 v=256
2026-01-01 09:46:39 IFCONFIG POOL IPv4: base=10.8.0.2 size=253
2026-01-01 09:46:39 IFCONFIG POOL LIST
2026-01-01 09:46:39 Initialization Sequence Completed
```

Op de remote employee machine startte ik de openvpn client:

```
cd ~/openvpn
sudo openvpn client.conf
```

Voor de authenticatie vulde ik de volgende gegevens in:

Username: vagrant Password: vagrant

De VPN client is nu gestart. Ik opende een nieuwe terminal op de remote employee. Als ik nu mijn IP adres opvroeg zag ik dat ik een IP adres uit het VPN subnet had gekregen:

```
4: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state
UNKNOWN group default qlen 500
    link/none
    inet 10.8.0.2/24 scope global tun0
```

```

    valid_lft forever preferred_lft forever
inet6 fe80::e8d4:2af5:8cbf:1ab5/64 scope link stable-privacy
    valid_lft forever preferred_lft forever

```

Als we nu een ping doen naar de dns server in het interne netwerk zien we dat de VPN verbinding werkt:

```

[vagrant@remote-employee ~]$ ping 172.30.20.15
PING 172.30.20.15 (172.30.20.15) 56(84) bytes of data.
64 bytes from 172.30.20.15: icmp_seq=1 ttl=63 time=0.921 ms
64 bytes from 172.30.20.15: icmp_seq=2 ttl=63 time=1.22 ms
64 bytes from 172.30.20.15: icmp_seq=3 ttl=63 time=1.07 ms
^C
--- 172.30.20.15 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.921/1.070/1.220/0.122 ms

```

In wireshark zien we dit dan als UDP verkeer.

