

Lab report, Gruppe 4

Projekt Netzwerk-Infrastruktur WS 2017/18

Dozent: Robert Olotu

vorgelegt von

Dewin Bagci: 5bagci@informatik.uni-hamburg.de (6815336)

Karan Popat: karan.popat@outlook.de (6600283)

Hanife Demircioglu: h.demircioglu@hotmail.de (6816065)

MIN-Fakultät

Fachbereich Informatik

Abgabedatum: 01.03.2018

Inhaltsverzeichnis

Part 3: Network Troubleshooting Utilities	3
Exercise 6: Managing Services (Please use pnidX-svr-mu	3
Exercise 7: Configure the following network (figure 1) using ifconfig and route add	12
Exercise 8: Configure the following network (figure 1) using ip and nmcli . . .	22
Exercise 9: Configure the following network (figure 1) using GUI	30
Part 4: Network Scanning	44
Exercise 1: Configure the networks of figure 1	45
Exercise 2: NMAP	48
Exercise 3: Nessus network device identification	54
Exercise 4: OpenVAS Network device identification	69
Part 5: Sniffing, Virtual Private Network (VPN)	81
Exercise 1: Configure and set the networks shown below (figure1 and 2) . . .	82
Exercise 2: Getting started with network monitoring tools	87
Exercise 3: TCPDUMP	93
Exercise 4: Wireshark	103
Exercise 5: Experimenting with network monitoring tools	110
Exercise 6: Set up a host-to-host VPN using preshared key	111
Exercise 7: Set up a host-to-host VPN using RSA keys	126
Exercise 8: Set up a network-to-network VPN using preshared key	138
Exercise 9: Set up a network-to-network VPN using RSA secrets keys	151
Abbildungsverzeichnis	164

Part 3: Network Troubleshooting Utilities

Exercise 6: Managing Services (Please use pnidX-svr-mu)

Please type and explain the meaning of the following commands:

- 1) # chkconfig
- 2) # chkconfig --list iptables
- 3) # chkconfig --level 2 iptables off
- 4) # chkconfig --level 2345 iptables off
- 5) # chkconfig iptables on | off
- 6) # chkconfig tftp on
- 7) # chkconfig --level 2 vsftpd off
- 8) # chkconfig --level 2345 vsftpd off
- 9) # Explain the function of xinetd

The super server xinetd controlled services are automatically enabled or disabled by chkconfig.

Please type and explain the meaning of the following commands:

- 10) # service network stop
- 11) # service network start

Please type and explain the meaning of the following commands:

- 1) # chkconfig

Die folgenden Kommandos wurden auf Rechner pnid4-svr-mu mit dem Betriebssystem Centos-6.5-x86_64 ausgeführt.

Zeigt an welche Dienste in ihren jeweiligen runlevels aktiviert bzw. deaktiviert sind.

```
[root@localhost ~]# chkconfig
NetworkManager 0:off 1:off 2:on 3:on 4:on 5:on 6:off
abrt-ccpp 0:off 1:off 2:off 3:on 4:off 5:on 6:off
abrtd 0:off 1:off 2:off 3:on 4:off 5:on 6:off
acpid 0:off 1:off 2:on 3:on 4:on 5:on 6:off
atd 0:off 1:off 2:off 3:on 4:on 5:on 6:off
auditd 0:off 1:off 2:on 3:on 4:on 5:on 6:off
autofs 0:off 1:off 2:off 3:on 4:on 5:on 6:off
blk-availability 0:off 1:on 2:on 3:on 4:on 5:on 6:off
certmonger 0:off 1:off 2:off 3:on 4:on 5:on 6:off
cpuspeed 0:off 1:on 2:on 3:on 4:on 5:on 6:off
crond 0:off 1:off 2:on 3:on 4:on 5:on 6:off
cups 0:off 1:off 2:on 3:on 4:on 5:on 6:off
dnsmasq 0:off 1:off 2:off 3:off 4:off 5:off 6:off
firstboot 0:off 1:off 2:off 3:off 4:off 5:off 6:off
haldaemon 0:off 1:off 2:off 3:on 4:on 5:on 6:off
ip6tables 0:off 1:off 2:on 3:on 4:on 5:on 6:off
iptables 0:off 1:off 2:on 3:on 4:on 5:on 6:off
irqbalance 0:off 1:off 2:off 3:on 4:on 5:on 6:off
kdump 0:off 1:off 2:on 3:on 4:on 5:on 6:off
lvm2-monitor 0:off 1:on 2:on 3:on 4:on 5:on 6:off
mdmonitor 0:off 1:off 2:on 3:on 4:on 5:on 6:off
messagebus 0:off 1:off 2:on 3:on 4:on 5:on 6:off
netconsole 0:off 1:off 2:off 3:off 4:off 5:off 6:off
netfs 0:off 1:off 2:off 3:on 4:on 5:on 6:off
network 0:off 1:off 2:on 3:on 4:on 5:on 6:off
nfs 0:off 1:off 2:off 3:off 4:off 5:off 6:off
nfsslock 0:off 1:off 2:off 3:on 4:on 5:on 6:off
ntpd 0:off 1:off 2:off 3:off 4:off 5:off 6:off
ntpdate 0:off 1:off 2:off 3:off 4:off 5:off 6:off
odjobd 0:off 1:off 2:off 3:off 4:off 5:off 6:off
portreserve 0:off 1:off 2:on 3:on 4:on 5:on 6:off
postfix 0:off 1:off 2:on 3:on 4:on 5:on 6:off
psacct 0:off 1:off 2:off 3:off 4:off 5:off 6:off
quota_nld 0:off 1:off 2:off 3:off 4:off 5:off 6:off
rdisc 0:off 1:off 2:off 3:off 4:off 5:off 6:off
```

Abbildung 1: aktivierte bzw. deaktivierte Dienste eines runlevels

2) # chkconfig -- list iptables

Zeigt an in welchen runlevel iptables eingeschaltet bzw. ausgeschaltet ist.

```
[root@localhost ~]# chkconfig --list iptables
iptables 0:off 1:off 2:on 3:on 4:on 5:on 6:off
[root@localhost ~]# █
```

Abbildung 2: Runlevels, in denen iptables eingeschaltet bzw. ausgeschaltet sind

3) # chkconfig --level 2 iptables off

Deaktiviert iptables im runlevel 2 Wir sehen, dass zuvor iptables im runlevel 2 aktiviert war.

```
[root@localhost ~]# chkconfig --list iptables
iptables 0:off 1:off 2:on 3:on 4:on 5:on 6:off
[root@localhost ~]# chkconfig --level 2 iptables off
[root@localhost ~]# chkconfig --list iptables
iptables 0:off 1:off 2:off 3:on 4:on 5:on 6:off
```

Abbildung 3: runlevel 2 wird ausgeschaltet

```
4) # chkconfig --level 2345 iptables off
```

Deaktiviert iptables im runlevel 2, 3, 4 und 5

```
[root@localhost ~]# chkconfig --level 2 iptables off
[root@localhost ~]# chkconfig --list iptables
iptables      0:off  1:off  2:off  3:on   4:on   5:on   6:off
[root@localhost ~]# chkconfig --level 2345 iptables off
[root@localhost ~]# chkconfig --list iptables
iptables      0:off  1:off  2:off  3:off  4:off  5:off  6:off
[root@root@localhost:~]
```

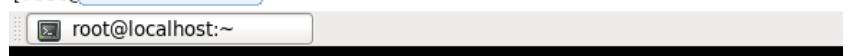


Abbildung 4: Runlevel 2,3,4,5 werden deaktiviert

```
5) # chkconfig iptables on | off
```

Mit iptables off wird iptables auf jedem runlevel deaktiviert. Mit iptables on wird iptables auf die default Konfiguration zurückgesetzt. Das bedeutet die runlevels 2,3,4 und 5 sind wieder aktiviert.

```
[root@localhost ~]# chkconfig --list iptables
iptables      0:off  1:off  2:off  3:off  4:off  5:off  6:off
[root@localhost ~]# chkconfig iptables on
[root@localhost ~]# chkconfig --list iptables
iptables      0:off  1:on   2:on   3:on   4:on   5:on   6:off
[root@localhost ~]# chkconfig iptables off
[root@localhost ~]# chkconfig --list iptables
iptables      0:off  1:off  2:off  3:off  4:off  5:off  6:off
[root@localhost ~]#
```



Abbildung 5: chkconfig iptables on | off

```
6) # chkconfig tftp on
```

Tftp ist ein Vorgänger des FTP-Protokolls. Dieser service ist nicht automatisch auf Centos-6.5-x86_64 vorinstalliert und wird durch den Superserver xinetd, welcher ebenfalls nicht automatisch vorinstalliert ist, verwaltet. Damit wir die gewissen Pakete mit allen Abhängigkeiten für tftp und xinetd über das Terminal mit yum (Yellow dog Updater, Modified) installieren können, müssen wir ein Quellpaket Repository einrichten. Zuerst erstellen wir einen Ordner mit # mkdir /dvdrom im Verzeichnis /etc/yum.repos.d Danach fügen wir das Verzeichnis als neues Repository hinzu, indem wir die Konfigurationsdatei mit dem vi Editor öffnen # vi /etc/yum.repos.d/local.repo und das Repository hinzufügen.

```
[LocalRepo]
name=Local Repository
baseurl=file:///dvdrom
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
```

Abbildung 6: vi /etc/yum.repos.d/local.repo

Zuletzt mounten wir das Verzeichnis mit dem Befehl # mount -t iso9660 /dev/sr0/dvdrom

```
[root@localhost yum.repos.d]# mount -t iso9660 /dev/sr0 /dvdrom
mount: block device /dev/sr0 is write-protected, mounting read-only
```

Abbildung 7: mounten

Nach dem wir den Befehl #yum clean all im Terminal ausgeführt haben, kann die Installation beginnen. Dies geschieht wie folgt:

Wir führen im Terminal den Befehl #yum install tftp aus, sodass die Installation starten kann.

```
[root@localhost yum.repos.d]# yum install tftp
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
LocalRepo
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package tftp.x86_64 0:0.49-7.el6 will be installed
--> Finished Dependency Resolution
Dependencies Resolved

=====
| Package          | Arch | Version | Repository | Size |
=====
| Installing:    |      |          |            |       |
| tftp             | x86_64 | 0.49-7.el6 | LocalRepo | 32 k |
| Transaction Summary |           |           |
| Install   1 Package(s) |           |           |
```

Abbildung 8: yum install tftp

Nachdem tftp installiert wurde, muss außerdem xinetd installiert werden. Ansonsten kann tftp nicht verwendet werden. Mit dem Befehl #yum install xinetd wird xinetd installiert.

```
[root@localhost ~]# yum install xinetd
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Package xinetd.x86_64 2:2.3.14-39.el6_4 will be installed
--> Finished Dependency Resolution
Dependencies Resolved

=====
| Package           | Arch      | Version        | Repository | Size   |
|=====             | =====    | ======         | =====      | ===== |
| Installing:     |           |               |            |        |
| xinetd           | x86_64   | 2:2.3.14-39.el6_4 | LocalRepo | 121 k |
| Transaction Summary |          |               |            |        |
| Install 1 Package(s) |          |               |            |        |
| Total download size: 121 k |          |               |            |        |
| Installed size: 259 k |          |               |            |        |
| Is this ok [y/N]: y |          |               |            |        |
| Downloading Packages: |          |               |            |        |
| Running Transaction Test |          |               |            |        |
| Running Transaction Test |          |               |            |        |
| Transaction Test Succeeded |          |               |            |        |
| Running Transaction |          |               |            |        |
| Installing : 2:xinetd-2.3.14-39.el6_4.x86_64 |          |               |        |
| Verifying : 2:xinetd-2.3.14-39.el6_4.x86_64 |          |               |        |
| Installed: |          |               |            |        |
| xinetd.x86_64 2:2.3.14-39.el6_4 |          |               |        |
| Complete! |          |               |            |        |
=====
```

Abbildung 9: yum install xinetd

Zunächst muss xinetd gestartet werden, damit wir Zugriff auf tftp haben. Dies geschieht mit dem Befehl # service xinetd start.

```
[root@localhost ~]# service xinetd start
Starting xinetd: [ OK ]
```

Abbildung 10: service xinetd start

Die Dateien im Verzeichnis /etc/xinetd.d/ enthalten die Konfigurationsdateien für jeden von xinetd verwalteten Dienst. Die Konfigurationsdatei tftp muss wie in Abbildung 15 angepasst werden. Damit tftp funktioniert, muss disable=no sein. Disable legt fest, ob der Dienst aktiv ist oder nicht. Im Regelfall ist "disable = yes" zu Beginn. Dieser muss dann geändert werden zu "diable = no". Nach der Konfiguration kann tftp genutzt werden, wie in Abbildung 12 zu sehen ist.

```
[root@localhost ~]# vi /etc/xinetd.d/tftp
[root@localhost ~]# service xinetd start
Starting xinetd:
[root@localhost ~]# chkconfig tftp on
[root@localhost ~]# chkconfig
```

Abbildung 11: xinetd tftp config

```
service tftp
{
    disable = no
    socket_type = dgram
    protocol = udp
    wait = yes
    user = root
    server = /usr/sbin/in.tftpd
    server_args = -s /var/lib/tftboot
    per_source = 11
    cps = 100 2
    flags = IPv4
}
```

Abbildung 12: vi tftp service

Nachdem der Befehl # chkconfig tftp on ausgeführt wurde, kann man sich mit dem Befehl #chkconfig anzeigenlassen, ob der Dienst wirklich aktiviert wurde, da dieser angibt welche Dienste in ihren jeweiligen runlevels aktiviert bzw. deaktiviert sind. In der Abbildung 13 sieht man, dass tftp aktiviert ist. Tftp findet man unten im Bild bei den "xinetd based services".

```

smartd      0:off  1:off  2:off  3:off  4:off  5:off  6:off
snmpd      0:off  1:off  2:off  3:off  4:off  5:off  6:off
snmptrapd  0:off  1:off  2:off  3:off  4:off  5:off  6:off
spice-vdagentd 0:off  1:off  2:off  3:off  4:off  5:on   6:off
sshd        0:off  1:off  2:on   3:on   4:on   5:on   6:off
sssd         0:off  1:off  2:off  3:off  4:off  5:off  6:off
sysstat     0:off  1:on   2:on   3:on   4:on   5:on   6:off
udev-post   0:off  1:on   2:on   3:on   4:on   5:on   6:off
wdaemon     0:off  1:off  2:off  3:off  4:off  5:off  6:off
winbind     0:off  1:off  2:off  3:off  4:off  5:off  6:off
wpa_supplicant 0:off  1:off  2:off  3:off  4:off  5:off  6:off
kinetd      0:off  1:off  2:off  3:on   4:on   5:on   6:off
ypbind      0:off  1:off  2:off  3:off  4:off  5:off  6:off

kinetd based services:
    chargen-dgram: off
    chargen-stream: off
    daytime-dgram: off
    daytime-stream: off
    discard-dgram: off
    discard-stream: off
    echo-dgram: off
    echo-stream: off
    rsync: off
    tcpmux-server: off
    tftp: on
    time-dgram: off
    time-stream: off

```

Abbildung 13: chkconfig

7) # chkconfig --level 2 vsftpd off

Um diesen Befehl ausführen zu können, muss zunächst vsftpd installiert werden. Dies geschieht mit dem Befehl # yum install vsftpd. Nach der erfolgreichen Installation kann vsftpd verwendet werden.

```

[root@localhost ~]# yum install vsftpd
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
Setting up Install Process
Resolving Dependencies
--> Running Transaction check
--> Package vsftpd.x86_64 0:2.2.2-11.el6_4.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
| Package           | Arch | Version | Repository | Size |
=====
| Installing:      |       |          |            |       |
| vsftpd           | x86_64 | 2.2.2-11.el6_4.1 | LocalRepo | 151 k |
=====

Transaction Summary
=====
| Install 1 Package(s)
Total download size: 151 k
Installed size: 151 k
Is this ok [y/N]: y
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : vsftpd-2.2.2-11.el6_4.1.x86_64
  Verifying  : vsftpd-2.2.2-11.el6_4.1.x86_64
                                                               1/1
Installed:
  vsftpd.x86_64 0:2.2.2-11.el6_4.1
Complete!

```

Abbildung 14: yum install vsftpd

Mit dem Befehl `#chkconfig --level 2 vsftpd off` wird der Runlevel 2 von vsftpd deaktiviert.

```
[root@localhost ~]# chkconfig --level 2 vsftpd off  
[root@localhost ~]# chkconfig
```

Abbildung 15: Runlevel 2 von vsftpd wird deaktiviert

8)`# chkconfig --level 2345 vsftpd off`

Mit dem Befehl `# chkconfig --level 2345 vsftpd off` werden die Runlevels 2, 3, 4 und 5 deaktiviert.

Zunächst haben wir mit dem Befehl "`# chkconfig --level 2345 vsftpd`" die Runlevels 2, 3, 4 und 5 aktiviert, wie in Abbildung 16 zu sehen ist.

```
[root@localhost ~]# chkconfig --level 2345 vsftpd on
```

Abbildung 16: chkconfig –level

Hier sieht man, dass nach der Aktivierung die entsprechenden Runlevels aktiviert wurden.

```
|vsftpd          0:off  1:off  2:on   3:on   4:on   5:on   6:off
```

Abbildung 17: vsftpd runlevels

Anschließend werden mit dem Befehl "`# chkconfig --level 2345 vsftpd off`" die Runlevels 2, 3, 4 und 5 deaktiviert.

```
[root@localhost ip nmcli]# chkconfig --level 2345 vsftpd off
```

Abbildung 18: chkconfig –level

Man erkennt, dass die aktivierte Runlevels nach dem Ausführen des Befehls ausgeschaltet wurden.

```
|vsftpd          0:off  1:off  2:off  3:off  4:off  5:off  6:off
```

Abbildung 19: vsftpd runlevels

9) # Explain the function of xinetd

Bei xinetd handelt es sich um einen open source Superserver für Unix-Systeme. Dieser verwaltet verschiedene Dienste u.a. den FTP / HTTP Server.

Xinetd bietet gegenüber dem Vorgänger inetd noch weitere zusätzliche Dienste an um eine verbesserte Sicherheit zu ermöglichen. Dazu zählen Zugangskontrollen, zeitliche Beschränkung von Diensten (nach Datum und Uhrzeit), sowie einen Verteidigungsmechanismus gegen Portscanner. Sobald der xinetd Superserver eingeschaltet ist, lässt sich im Terminal nachvollziehen, welche Dienste über xinetd verwaltet werden.

The super server xinetd controlled services are automatically enabled or disabled by chkconfig.

Please type and explain the meaning of the following commands:

10) # service network stop

Der command stoppt alle konfigurierten Netzwerk interfaces. 11) # service network start

Der command aktiviert alle konfigurierten Netzwerk interfaces.

```
[root@localhost ~]# service network stop
Shutting down interface eth0:                                [  OK  ]
Shutting down loopback interface:                            [  OK  ]
[root@localhost ~]# service network start
Bringing up loopback interface:                             [  OK  ]
[root@root@localhost:~]
[  OK  ]
```

Abbildung 20: Netzwerk Interface aktivieren

Exercise 7: Configure the following network (figure 1) using ifconfig and route add

You need to set the network depicted on figure 1 by doing the following:

Use the "ifconfig" and the "route add" commands to configure all the subnets 10.88.X.32/27, 10.88.X.64/27, 10.88.X.96/27 and 10.88.X.128/27. For this exercise you will use the hosts pnidX-svr-mu, pnidX-WEB-hn, pnidX-svr-bln and pnidX-svr-hh. Furthermore you have to configure the routers pnidX-rou-1, pnidX-rou-2 and pnidX-rou-3

Hint 1: Remember after rebooting the system, the ifconfig and route add configuration will disappear

Hint 2: Do not forget to flush the firewall by issuing the command "iptables -F"

Please use Kali Linux as root:

```
# zenmap
```

Scan the networks:

10.88.X.32/27

10.88.X.64/27

10.88.X.96/27

and 10.88.X.128/27

Ziel der Aufgaben 7, 8 und 9 ist, das folgende Netzwerk mithilfe verschiedener Kommandos aufzubauen um alle zugehörigen Subnetze zu konfigurieren.

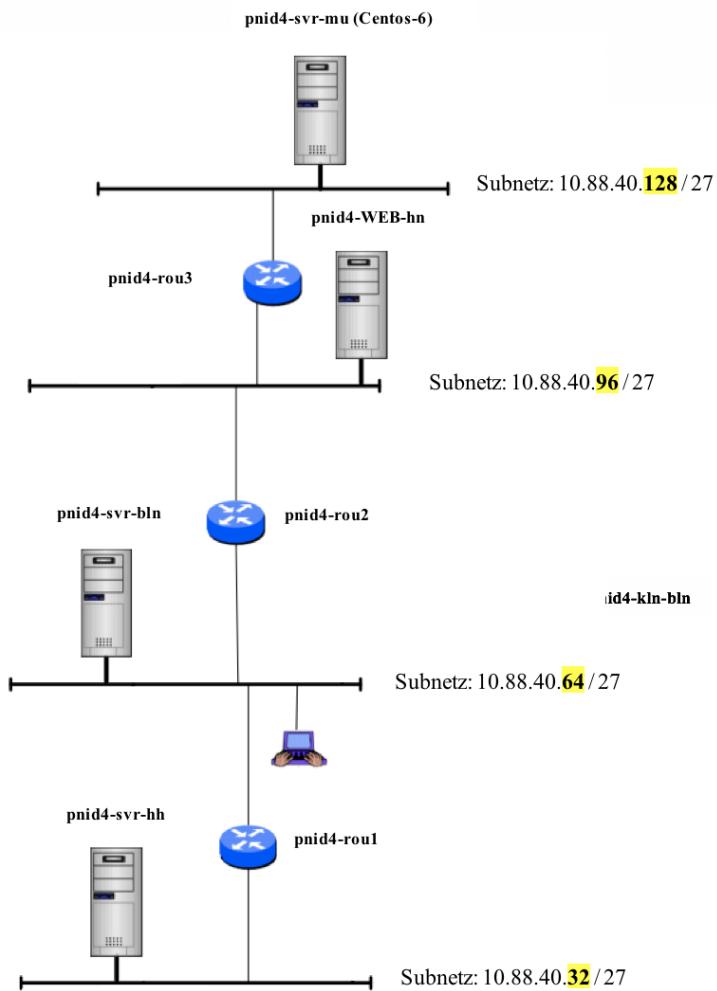


Abbildung 21: Netzwerk figure 1

Zunächst haben wir das Netzwerk für den pnid4-svr-mu (Sever München) konfiguriert, indem wir die Ethernetkarte eth0 zustehende Adresse 10.88.40.129 zugewiesen haben und die Netmask-Adresse mit einbinden. Dieses haben wir ermöglicht, indem wir /27 im Anschluss hinzugefügt haben, jedoch ist es auch über dem Schlüsselwort netmask und die komplette Adresse realisierbar.

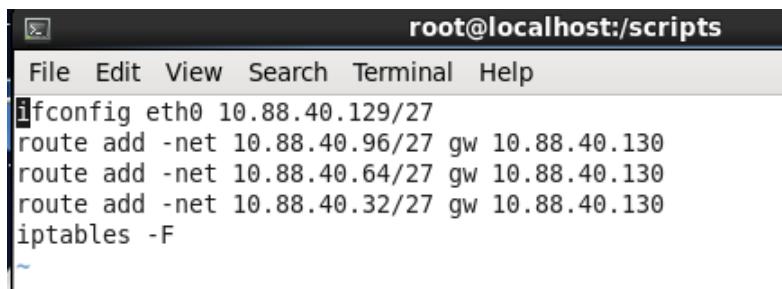
Anschließend haben wir die Routing tables über dem Kommando route add erstellt. Hier haben wir die Netze 10.88.40.96, 10.88.40.64 und 10.88.40.32 mit dem Kommando versehen, da wir über alle drei Netze eine Verbindung herstellen möchten.

Nachstehend haben wir die Firewall dieses Servers ausgeschaltet, da wir später eine Verbindung mit anderen Servern und Routern aufbauen möchten. Wir haben die ganze Konfiguration in einer txt-Datei geschrieben und im Anschluss einmal ausgeführt, sodass wir die Konfiguration gespeichert haben und nicht bei jedem Shut-Down erneut einstellen müssen.

Konfigurationsdatei von Server-München:(Routing-Tabelle)

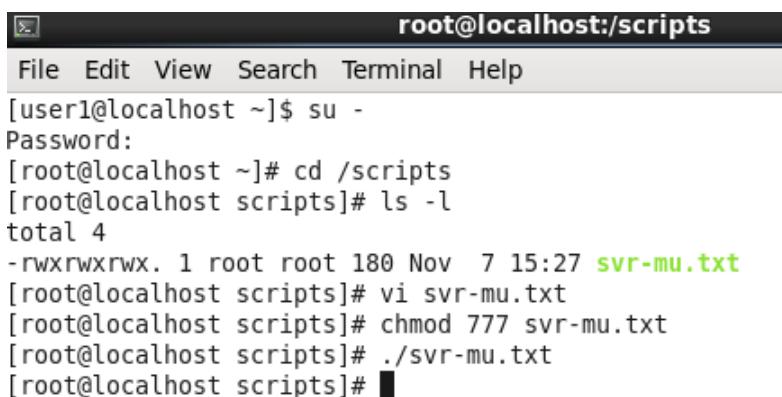
ifconfig eth0 weißt der Netzwerkkarte die zugehörige IP-Adresse zu.

Mit dem Befehl route add werden statische Routen zu Rechnern und Netzwerken festgelegt.



```
root@localhost:/scripts
File Edit View Search Terminal Help
ifconfig eth0 10.88.40.129/27
route add -net 10.88.40.96/27 gw 10.88.40.130
route add -net 10.88.40.64/27 gw 10.88.40.130
route add -net 10.88.40.32/27 gw 10.88.40.130
iptables -F
~
```

Abbildung 22: ifconfig Server München



```
root@localhost:/scripts
File Edit View Search Terminal Help
[user1@localhost ~]$ su -
Password:
[root@localhost ~]# cd /scripts
[root@localhost scripts]# ls -l
total 4
-rwxrwxrwx. 1 root root 180 Nov  7 15:27 svr-mu.txt
[root@localhost scripts]# vi svr-mu.txt
[root@localhost scripts]# chmod 777 svr-mu.txt
[root@localhost scripts]# ./svr-mu.txt
[root@localhost scripts]#
```

Abbildung 23: Server München Routing-Tabelle anwenden

Konfigurationsdatei von WEB-Hannover:(Routing-Tabelle)

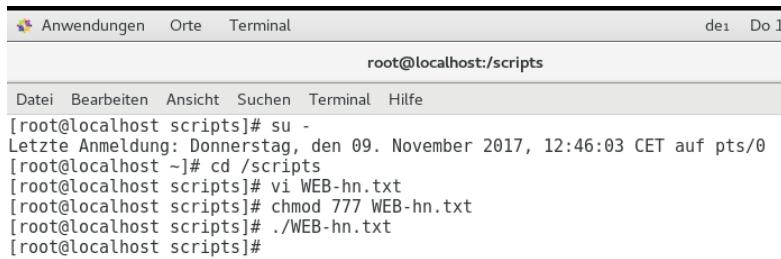
ifconfig ens33 weißt der Netzwerkkarte die zugehörige IP-Adresse zu.

Mit dem Befehl route add werden statische Routen zu Rechnern und Netzwerken festgelegt.



```
root@localhost:/scripts
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
ifconfig ens33 10.88.40.98/27
route add -net 10.88.40.128/27 gw 10.88.40.97
route add -net 10.88.40.64/27 gw 10.88.40.99
route add -net 10.88.40.32/27 gw 10.88.40.99
iptables -F
```

Abbildung 24: ifconfig Webserver Hannover



```
root@localhost:/scripts
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
[root@localhost scripts]# su -
Letzte Anmeldung: Donnerstag, den 09. November 2017, 12:46:03 CET auf pts/0
[root@localhost ~]# cd /scripts
[root@localhost scripts]# vi WEB-hn.txt
[root@localhost scripts]# chmod 777 WEB-hn.txt
[root@localhost scripts]# ./WEB-hn.txt
[root@localhost scripts]#
```

Abbildung 25: Webserver Hannover Routing-Tabelle anwenden

Konfigurationsdatei von Server-Berlin:(Routing-Tabelle)

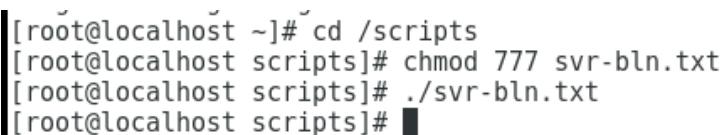
ifconfig ens33 weißt der Netzwerkkarte die zugehörige IP-Adresse zu.

Mit dem Befehl route add werden statische Routen zu Rechnern und Netzwerken festgelegt.



```
Anwendungen Orte Terminal  
hanifka@localhost:scripts  
Datei Bearbeiten Ansicht Suchen Terminal Hilfe  
ifconfig ens33 10.88.40.66/27  
route add -net 10.88.40.128/27 gw 10.88.40.65  
route add -net 10.88.40.96/27 gw 10.88.40.65  
route add -net 10.88.40.32/27 gw 10.88.40.67  
iptables -F
```

Abbildung 26: ifconfig Server Berlin



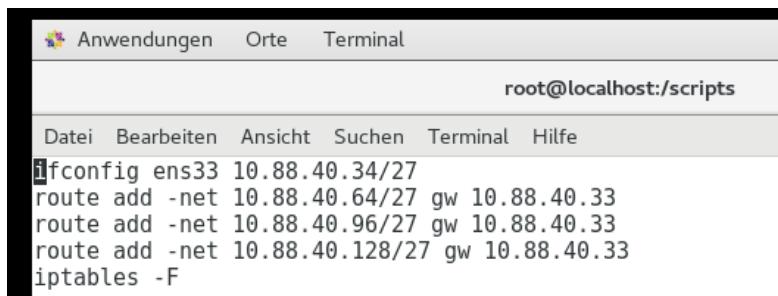
```
[root@localhost ~]# cd /scripts  
[root@localhost scripts]# chmod 777 svr-bln.txt  
[root@localhost scripts]# ./svr-bln.txt  
[root@localhost scripts]#
```

Abbildung 27: Server Berlin Routing-Tabelle anwenden

Konfigurationsdatei von Server-Hamburg:(Routing-Tabelle)

ifconfig ens33 weißt der Netzwerkkarte die zugehörige IP-Adresse zu.

Mit dem Befehl route add werden statische Routen zu Rechnern und Netzwerken festgelegt.



```
Anwendungen Orte Terminal  
root@localhost:scripts  
Datei Bearbeiten Ansicht Suchen Terminal Hilfe  
ifconfig ens33 10.88.40.34/27  
route add -net 10.88.40.64/27 gw 10.88.40.33  
route add -net 10.88.40.96/27 gw 10.88.40.33  
route add -net 10.88.40.128/27 gw 10.88.40.33  
iptables -F
```

Abbildung 28: ifconfig Server Hamburg

```
[root@localhost ~]# cd /scripts
[root@localhost scripts]# vi svr-hh.txt
[root@localhost scripts]# chmod 777 svr-hh.txt
[root@localhost scripts]# ./svr-hh.txt
[root@localhost scripts]# █
```

Abbildung 29: Server Hamburg Routing-Tabelle anwenden

Konfigurationsdatei von Router 1:(Routing-Tabelle)

Wie in folgenden Ausschnitten zusehen ist haben wir die Router ebenfalls, so wie oben beschrieben, konfiguriert. Allerdings haben wir hier zwei Ethernet-Anbindungen. Denn ein Router hat immer eine Verbindung zwischen mindestens zwei Netzwerken und leitet Datenpakete anhand von Information der IP-Adressen zwischen den Netzwerken weiter. Mit ifconfig ens33 und ens37 weißt man den Netzwerkarten die zugehörige IP-Adresse zu.

Mit dem Befehl route add werden statische Routen zu Rechnern und Netzwerken festgelegt.



```
root@localhost:/scripts
ifconfig ens33 10.88.40.67/27
ifconfig ens37 10.88.40.33/27
route add -net 10.88.40.96/27 gw 10.88.40.65
route add -net 10.88.40.128/27 gw 10.88.40.65
iptables -F
```

Abbildung 30: ifconfig Router 1



```

Anwendungen Orte Terminal de1 Do 13:00
root@localhost:/scripts -
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
[hanifka@localhost ~]$ su -
Passwort:
Letzte Anmeldung: Dienstag, den 07. November 2017, 14:32:53 CET auf pts/0
[root@localhost ~]# cd /scripts
[root@localhost scripts]# vi rou1.txt
[root@localhost scripts]# chmod 777 rou1.txt
[root@localhost scripts]# ./rou1.txt
[root@localhost scripts]#

```

Abbildung 31: Router 1 Routing-Tabelle anwenden

Konfigurationsdatei von Router 2:(Routing-Tabelle)

Mit ifconfig ens33 und ens37 weist man den Netzwerkkarten die zugehörige IP-Adresse zu.

Mit dem Befehl route add werden statische Routen zu Rechnern und Netzwerken festgelegt.



```

Anwendungen Orte Terminal root@localhost:/scripts
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
ifconfig ens33 10.88.40.99/27
ifconfig ens37 10.88.40.65/27
route add -net 10.88.40.128/27 gw 10.88.40.97
route add -net 10.88.40.32/27 gw 10.88.40.67
iptables -F

```

Abbildung 32: ifconfig Router 2



```

Anwendungen Orte Terminal de1 Do
root@localhost:/scripts
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
[hanifka@localhost ~]$ su -
Passwort:
Letzte Anmeldung: Dienstag, den 07. November 2017, 14:30:18 CET auf pts/0
[root@localhost ~]# cd /scripts
-bash: cd: /scirpts: Datei oder Verzeichnis nicht gefunden
[root@localhost ~]# cd /scripts
[root@localhost scripts]# vi rou2.txt
[root@localhost scripts]# chmod 777 rou2.txt
[root@localhost scripts]# ./rou2.txt
[root@localhost scripts]#

```

Abbildung 33: Router 2 Routing-Tabelle anwenden

Konfigurationsdatei von Router 3:(Routing-Tabelle)

Mit ifconfig ens33 und ens37 weißt man den Netzwerkkarten die zugehörige IP-Adresse zu.

Mit dem Befehl route add werden statische Routen zu Rechnern und Netzwerken festgeleget.

```
root@localhost:/scripts
ifconfig ens33 10.88.40.130/27
ifconfig ens37 10.88.40.97/27
route add -net 10.88.40.64/27 gw 10.88.40.99
route add -net 10.88.40.32/27 gw 10.88.40.99
iptables -F
```

Abbildung 34: ifconfig Router 3

```
root@localhost:/scripts
[hanifka@localhost ~]$ su -
Passwort:
Letzte Anmeldung: Dienstag, den 07. November 2017, 14:46:57 CET auf pts/0
[root@localhost ~]# cd /scripts
[root@localhost scripts]# vi rou3.txt
[root@localhost scripts]# chmod 777 rou3.txt
[root@localhost scripts]# ./rou3.txt
[root@localhost scripts]#
```

Abbildung 35: Router 3 Routing-Tabelle anwenden

Please use Kali Linux as root: # zenmap

Scan the networks:

10.88.X.32/27

10.88.X.64/27

10.88.X.96/27

and 10.88.X.128/27

Zenmap

Zenmap ist eine grafische Ansicht für Nmap, der Ports scannen kann. Wenn man einen Rechner auf offene Ports checken möchte, dann kommt Nmap zum Einsatz. Der Network Mapper ist dafür da, um alle aktiven Hosts in der Netzwerkumgebung (über Ping) sowie deren Betriebssystem und Versionsnummern installierter Dienste herauszufinden. Infolgedessen konnten wir mit dem Kommando zenmap die Verbindungen von Netzwerk 10.88.40.32, 10.88.40.64, 10.88.40.96 und 10.88.40.128 grafisch darstellen und demonstrieren, dass wir die oben aufgeführte Abbildung und somit unser Ziel erreicht haben.

Das Netz 10.88.40.32/27 wird gescannt:

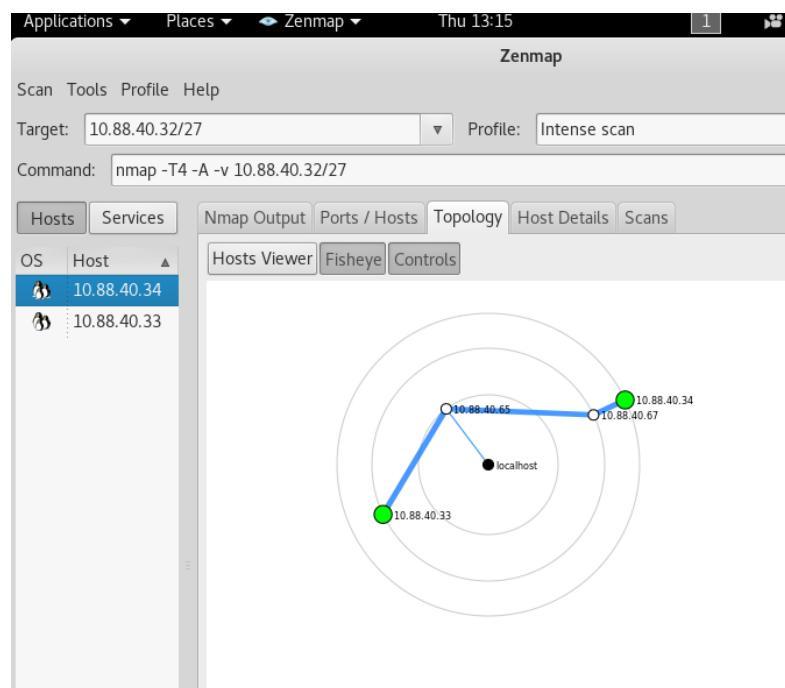


Abbildung 36: Netz: 10.88.40.32/27

Das Netz 10.88.40.64/27 wird gescannt:

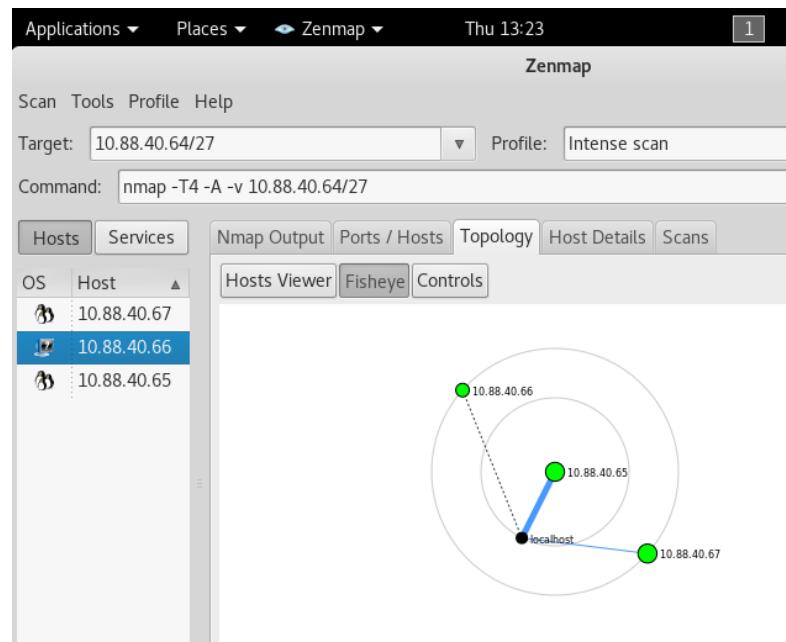


Abbildung 37: Netz: 10.88.40.64/27

Das Netz 10.88.40.96/27 wird gescannt:

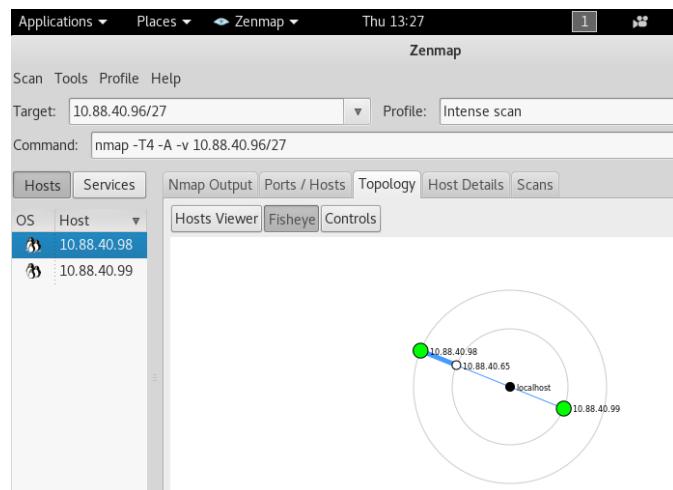


Abbildung 38: Netz: 10.88.40.96/27

Das Netz 10.88.40.128/27 wird gescannt:

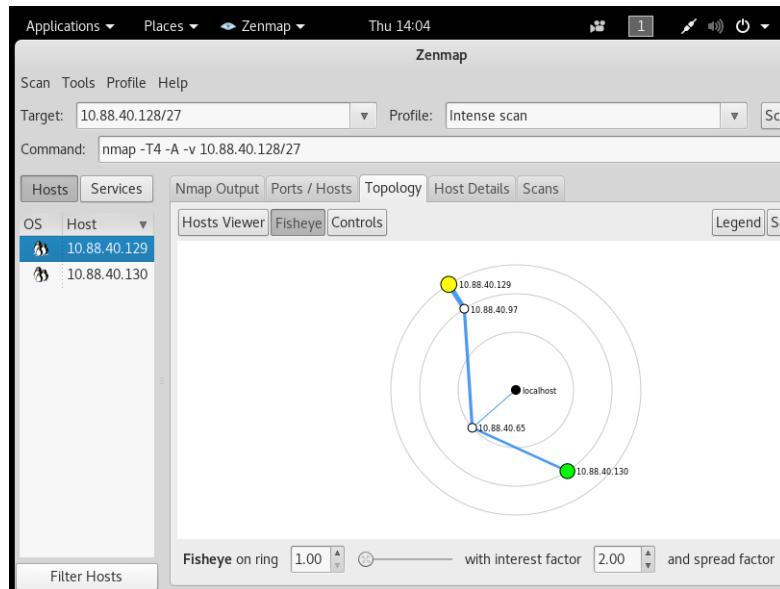


Abbildung 39: Netz: 10.88.40.128/27

Exercise 8: Configure the following network (figure 1) using ip and nmcli

You need to set the network depicted on figure 1 by doing the following:

Use the ?ifconfig? and the ?route add? commands to configure all the subnets 10.88.X.32/27, 10.88.X.64/27, 10.88.X.96/27 and 10.88.X.128/27. For this exercise you will use the hosts pnidX-svr-mu, pnidX-WEB-hn, pnidX-svr-blh and pnidX-svr-hh. Furthermore you have to configure the routers pnidX-rou-1, pnidX-rou-2 and pnidX- rou-3

Hint 1: Remember after rebooting the system, the ifconfig and route add configuration will disappear

Hint 2: Do not forget to flush the firewall by issuing the command iptables -F

Please use Kali Linux as root:

```
# zenmap
```

Scan the networks:

```
10.88.X.32/27  
10.88.X.64/27  
10.88.X.96/27  
10.88.X.128/27
```

Einleitung:

In dieser Aufgabe geht es darum das Netzwerk mittels ip und nmcli zu konfigurieren. Ip ist die neuere Version des Kommandozeilenprogramms von ifconfig. Demzufolge ist Sie leistungsfähiger und wird irgendwann ifconfig ersetzen. Man muss sich nur mit der Syntax vertraut machen. In den nächsten Schritten wird gezeigt, wie man mittels ip die Netze konfiguriert. In dieser Aufgabe gehen wir vom Aufbau her wie in Exercise 7 vor. Deswegen betrachten wir der übersichtshalber nur die vi-Datei, die entsprechend angepasst werden muss.

Vergleich von Nmcli und IP:

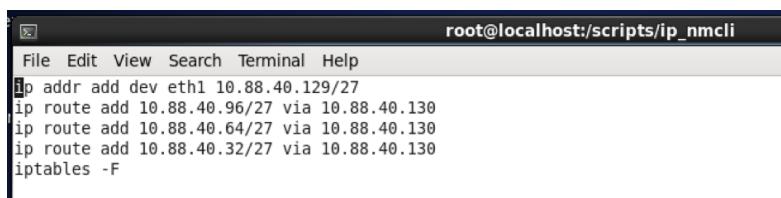
Nmcli auch bekannt als Network Manager hat viele Vorteile gegenüber anderen Verfahren, insbesondere bei einer WLAN-Verbindung. Denn Nmcli regelt die Verbindung zum Netzwerk nicht ausschließlich beim Hochfahren, sondern überwacht auch im Laufe eines Betriebes die Verbindung und wenn nötig stellt nmcli die Verbindung nach Unterbrechungen wieder her. Deswegen ist der Network Manager auch für Server-Installationen interessant, quasi als Ersatz für die Netzwerkkonfiguration. Auch bei der Fehlersuche spielt nmcli eine bedeutende Rolle. Die Möglichkeiten von nmcli und die dabei zu verwendende Kommandosprache sind versionsabhängig. Zunächst betrachten wir die für CentOS 7 verwendete Kommandosprache.

Ähnlich wie bei ip wird zunächst die IP-Adresse festgelegt. Dies geschieht mit dem Befehl nmcli con add con-name mit dem jeweiligen Namen der Netzwerkkarte. In diesem Fall ist es ens33 mit dem Typen type ethernet ifname, wie man der Abbildung 48

entnehmen kann. Anschließend folgt die IP-Adresse in Version 4. Deswegen schreibt man ip4 und direkt danach die dazugehörige IP-Adresse mit der entsprechenden Subnetzmaske. In diesem Fall wieder 255.255.255.224 und vereinfacht dargestellt als /27 direkt hinter der IP-Adresse.

Die nächsten Zeilen legen fest, durch welches Gateway ein Datenpaket weitergeleitet werden soll, wenn es nicht im entsprechenden Subnetz ist, sprich wenn die Netzadresse nicht dieselbe ist. Der Befehl nmcli connection modify dient dazu die entsprechenden Routen zu bestimmen. Im Anschluss steht der Name der Netzwerkkarte. Der Befehl +ipv4.routes sorgt dafür, dass das entsprechende Subnetz angegeben wird, in das geroutet werden soll. Dafür muss man noch das jeweilige Gateway angeben, durch welches man in das entsprechende Subnetz gelangt. Dies geschieht mittels ipv4.gateway. Am Ende wird noch mit dem Befehl iptables -F die Firewall ausgeschaltet.

Konfigurationsdatei von Server München mittels ip addr (Routing-Tabelle)



```
root@localhost:/scripts/ip_nmcli
File Edit View Search Terminal Help
ip addr add dev eth1 10.88.40.129/27
ip route add 10.88.40.96/27 via 10.88.40.130
ip route add 10.88.40.64/27 via 10.88.40.130
ip route add 10.88.40.32/27 via 10.88.40.130
iptables -F
```

Abbildung 40: Konfig. Server München ip addr

Konfigurationsdatei von Router 3 mittels ip addr (Routing-Tabelle)

```
ip addr add dev ens33 10.88.40.130/27
ip addr add dev ens37 10.88.40.97/27
ip route add 10.88.40.64/27 via 10.88.40.99
ip route add 10.88.40.32/27 via 10.88.40.99
iptables -F
```

Abbildung 41: Konfig. Router 3 ip addr

Gleiche Konfigurationsdatei von Router 3 mittels nmcli (Routing-Tabelle)

```
Anwendungen Orte Terminal de1 Sa 0
root@localhost:/scripts/nmcli
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
nmcli con add con-name ens33 type ethernet ifname ens33 ip4 10.88.40.130/27
nmcli con add con-name ens37 type ethernet ifname ens37 ip4 10.88.40.97/27
nmcli con mod ens33 +ipv4.routes 10.88.40.64/27 ipv4.gateway 10.88.40.99
nmcli con mod ens33 +ipv4.routes 10.88.40.32/27 ipv4.gateway 10.88.40.99
iptables -F
```

Abbildung 42: Konfig. Router 3 nmcli

Nach dem Ausführen des Skriptes wurden die Verbindungen ens33 und ens37 erfolgreich hinzugefügt. Mit nmcli con up wird die Verbindung aktiviert und anschließend wird über netstat -nr die Routing-Tabelle angezeigt.

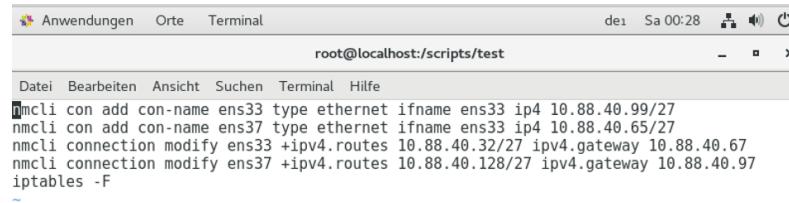
```
[root@localhost nmcli]# chmod 777 rou-3.txt
[root@localhost nmcli]# ./rou-3.txt
Verbindung »ens33« (318e4197-c623-45c3-baac-d1224d67f7c9) erfolgreich hinzugefügt.
Verbindung »ens37« (cb236514-d214-49ab-95fc-03fc165ca9f0) erfolgreich hinzugefügt.
[root@localhost nmcli]# nmcli con up ens33
Verbindung wurde erfolgreich aktiviert (aktiver D-Bus-Pfad: /org/freedesktop/NetworkManager/ActiveConnection/20)
[root@localhost nmcli]# nmcli con up ens37
Verbindung wurde erfolgreich aktiviert (aktiver D-Bus-Pfad: /org/freedesktop/NetworkManager/ActiveConnection/21)
[root@localhost nmcli]# netstat -nr
Kernel IP Routentabelle
Ziel      Router      Genmask      Flags   MSS Fenster irtt Iface
0.0.0.0    10.88.40.99  0.0.0.0      UG        0 0       0 ens33
10.88.40.32 0.0.0.0    255.255.255.224 U          0 0       0 ens33
10.88.40.64 0.0.0.0    255.255.255.224 U          0 0       0 ens33
10.88.40.96 0.0.0.0    255.255.255.224 U          0 0       0 ens37
10.88.40.99 0.0.0.0    255.255.255.255 UH        0 0       0 ens33
10.88.40.128 0.0.0.0   255.255.255.224 U          0 0       0 ens33
192.168.122.0 0.0.0.0  255.255.255.0     U        0 0       0 virbr0
[root@localhost nmcli]#
```

Konfigurationsdatei von Router 2 mittels ip addr (Routing-Tabelle)

```
Anwendungen Orte Terminal
root@localhost:/scripts/ip_nmcli
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
ip addr add dev ens33 10.88.40.99/27
ip addr add dev ens37 10.88.40.65/27
ip route add 10.88.40.128/27 via 10.88.40.97
ip route add 10.88.40.32/27 via 10.88.40.67
iptables -F
```

Abbildung 43: Konfig. Router 2 ip addr

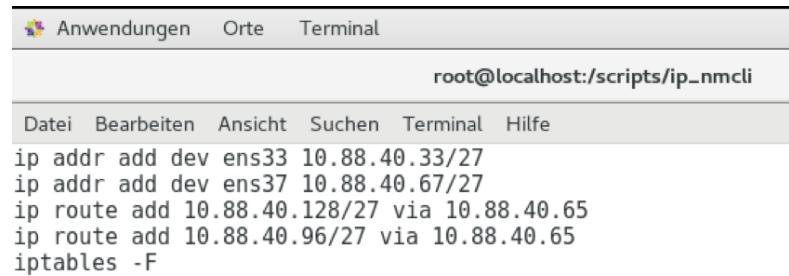
Gleiche Konfigurationsdatei von Router 2 mittels nmcli (Routing-Tabelle)



```
Anwendungen Orte Terminal de1 Sa 00:28
root@localhost:/scripts/test
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
nmcli con add con-name ens33 type ethernet ifname ens33 ip4 10.88.40.99/27
nmcli con add con-name ens37 type ethernet ifname ens33 ip4 10.88.40.65/27
nmcli connection modify ens33 +ipv4.routes 10.88.40.32/27 ipv4.gateway 10.88.40.67
nmcli connection modify ens37 +ipv4.routes 10.88.40.128/27 ipv4.gateway 10.88.40.97
iptables -F
```

Abbildung 44: Konfig. Router 2 nmcli

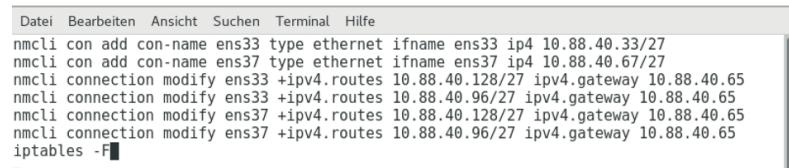
Konfigurationsdatei von Router 1 mittels ip addr (Routing-Tabelle)



```
Anwendungen Orte Terminal
root@localhost:/scripts/ip_nmcli
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
ip addr add dev ens33 10.88.40.33/27
ip addr add dev ens37 10.88.40.67/27
ip route add 10.88.40.128/27 via 10.88.40.65
ip route add 10.88.40.96/27 via 10.88.40.65
iptables -F
```

Abbildung 45: Konfig. Router 1 ip addr

Gleiche Konfigurationsdatei von Router 1 mittels nmcli (Routing-Tabelle)



```
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
nmcli con add con-name ens33 type ethernet ifname ens33 ip4 10.88.40.33/27
nmcli con add con-name ens37 type ethernet ifname ens37 ip4 10.88.40.67/27
nmcli connection modify ens33 +ipv4.routes 10.88.40.128/27 ipv4.gateway 10.88.40.65
nmcli connection modify ens33 +ipv4.routes 10.88.40.96/27 ipv4.gateway 10.88.40.65
nmcli connection modify ens37 +ipv4.routes 10.88.40.128/27 ipv4.gateway 10.88.40.65
nmcli connection modify ens37 +ipv4.routes 10.88.40.96/27 ipv4.gateway 10.88.40.65
iptables -F
```

Abbildung 46: Konfig. Router 1 nmcli

Konfigurationsdatei von Webserver Hannover mittels ip addr (Routing-Tabelle)

```
Anwendungen Orte Terminal
root@localhost:/scripts/ip_nmcli
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
ip addr add dev ens33 10.88.40.98/27
ip route add 10.88.40.128/27 via 10.88.40.97
ip route add 10.88.40.64/27 via 10.88.40.99
ip route add 10.88.40.32/27 via 10.88.40.99
iptables -F
```

Abbildung 47: Konfig. Webserver Hannover ip addr

Konfigurationsdatei von Server Berlin mittels nmcli (Routing-Tabelle)

```
Anwendungen Orte Terminal
root@localhost:/scripts/nmcli
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
nmcli con add con-name ens33 type ethernet ifname ens33 ip4 10.88.40.66/27
nmcli connection modify ens33 +ipv4.routes 10.88.40.96/27 ipv4.gateway 10.88.40.65
nmcli connection modify ens33 +ipv4.routes 10.88.40.128/27 ipv4.gateway 10.88.40.65
nmcli connection modify ens33 +ipv4.routes 10.88.40.32/27 ipv4.gateway 10.88.40.67
iptables -F
```

Abbildung 48: Konfig. Server Berlin nmcli

```
[root@localhost nmcli]# ./svr-bln.txt
Verbindung >ens33< (fee9cc86-f0aa-48ab-b377-c76b71ceb9bb) erfolgreich hinzugefügt.
[root@localhost nmcli]#
[root@localhost nmcli]# nmcli con up ens33
Verbindung wurde erfolgreich aktiviert (aktiver D-Bus-Pfad: /org/freedesktop/NetworkManager/ActiveConnection/9)
[root@localhost nmcli]# netstat -nr
Kernel IP Routentabelle
Ziel      Router      Genmask      Flags   MSS Fenster irtt Iface
0.0.0.0    10.88.40.67  0.0.0.0      UG        0 0          0 ens33
10.88.40.32 0.0.0.0    255.255.255.224 U          0 0          0 ens33
10.88.40.64 0.0.0.0    255.255.255.224 U          0 0          0 ens33
10.88.40.96 0.0.0.0    255.255.255.224 U          0 0          0 ens33
10.88.40.128 0.0.0.0   255.255.255.224 U          0 0          0 ens33
192.168.122.0 0.0.0.0   255.255.255.0   U          0 0          0 virbr0
[root@localhost nmcli]#
```

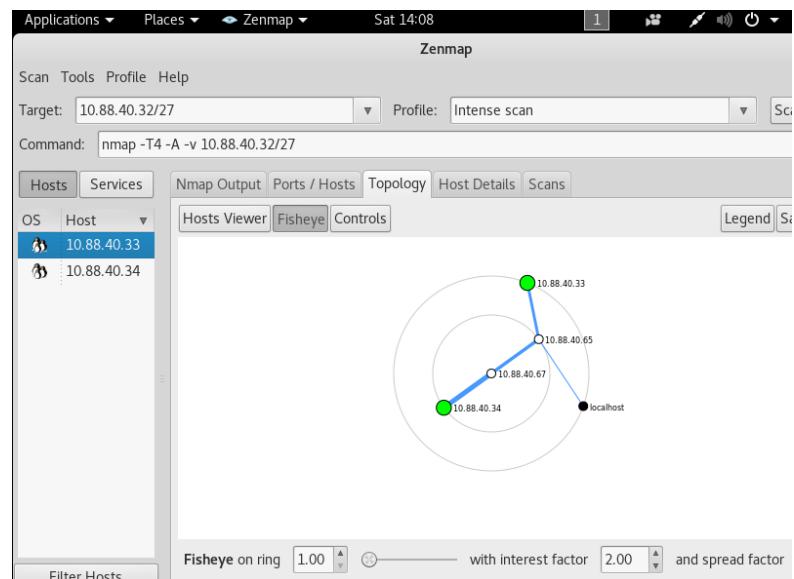
Konfigurationsdatei von Server Hamburg mittels ip addr (Routing-Tabelle)

```
root@localhost:/script# ip addr
Datei  Bearbeiten  Ansicht  Suchen  Terminal  Hilfe
ip addr add dev ens33 10.88.40.34/27
ip route add 10.88.40.64/27 via 10.88.40.33
ip route add 10.88.40.96/27 via 10.88.40.33
ip route add 10.88.40.128/27 via 10.88.40.33
iptables -F
```

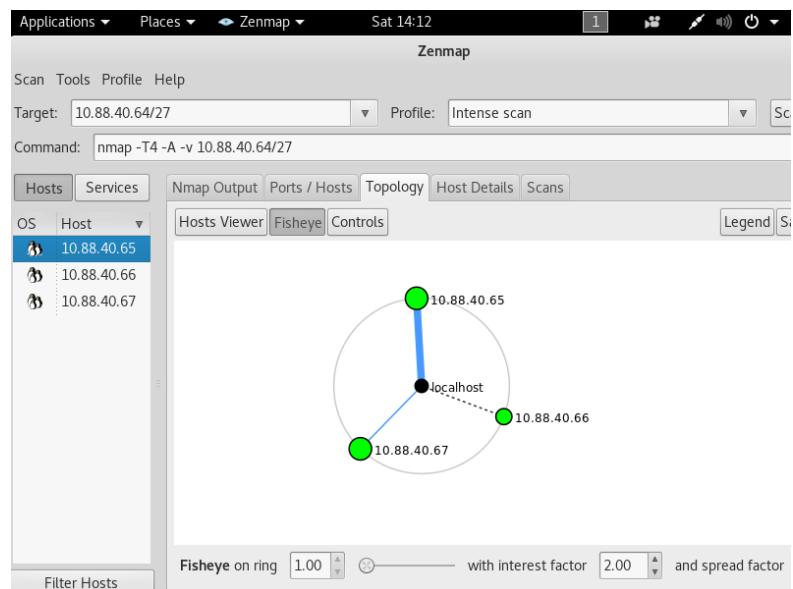
Abbildung 49: Konfig. Server Hamburg ip addr

Zenmap Ergebnisse der gescannten Subnetze:

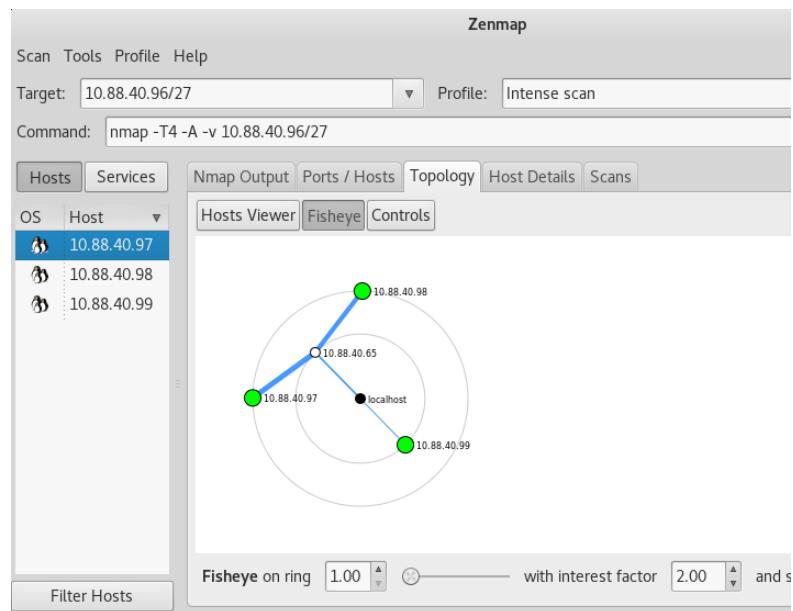
Scan des Subnetzes 10.88.40.32/27



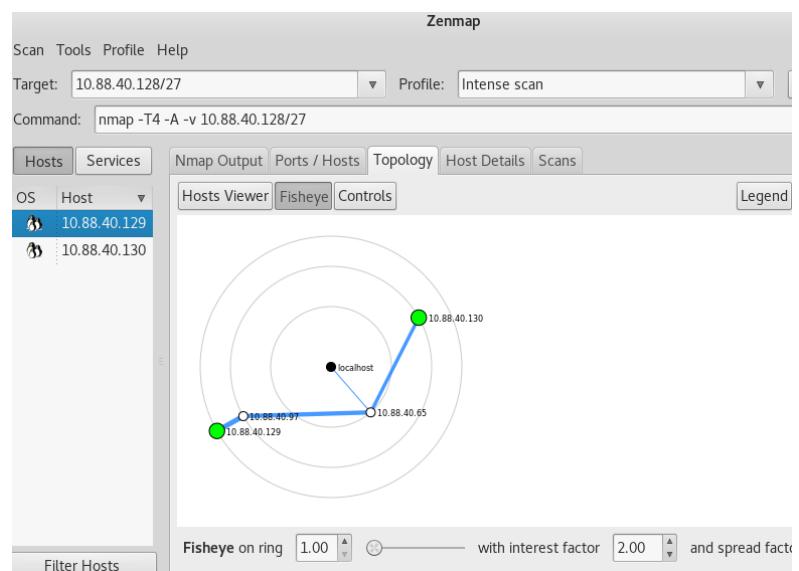
Scan des Subnetzes 10.88.40.64/27



Scan des Subnetzes 10.88.40.96/27



Scan des Subnetzes 10.88.40.128/27

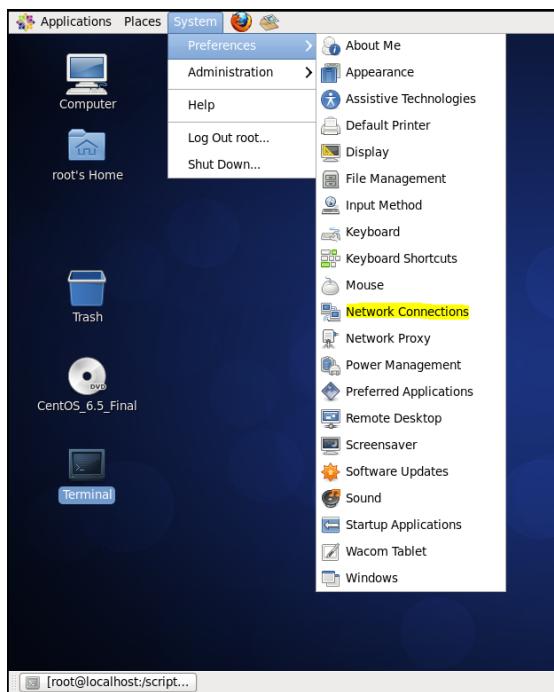


Exercise 9: Configure the following network (figure 1) using GUI

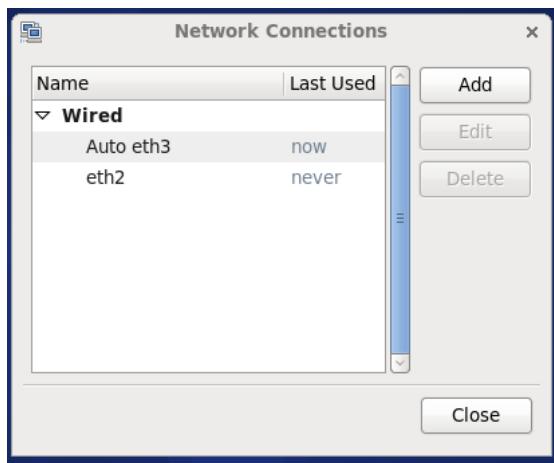
Damit die Konfiguration problemlos gelingen kann, sollte auf jedem Host, im Terminal, mit dem Kommando `iptables -F` die Firewall deaktiviert werden.

Konfiguration von Server München über die GUI:

Wir starten die Netzwerkkonfiguration auf dem Host pnid4-svr-mu mit dem Betriebssystem Centos 6. Zunächst gehen wir im Auswahlmenü auf System -> Preferences und wählen Network Connections aus.



Es öffnet sich ein Fenster in der wir die Bezeichner alle verfügbaren Netzwerk Interfaces sehen. Wir Wählen Auto eth3 aus und bestätigen mit Edit. Mit Add lässt sich ein neues Interface einrichten.

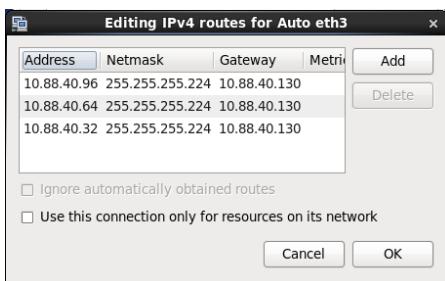


Wir können nun unser bereits vorhandenes Netzwerk Interface Auto eth3 bearbeiten. Dazu Wählen wir in der Navigation IPv4 Settings aus und erhalten folgende Ansicht. Falls Auto ausgewählt ist, ändern wie dieses auf Manual, damit wir eine statische IP Adresse eintragen können. Wir können im Bereich Addresses mit Add eine statische

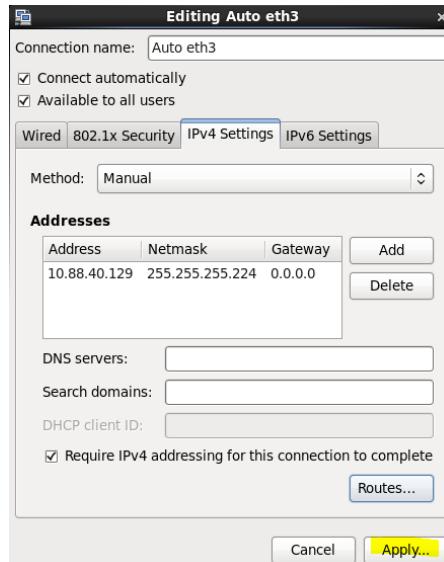
IPv4 Adresse für den Host eintragen. Da wir auf pnid4-svr-mu, vergeben wir die Adresse 10.88.40.129 mit der Subnetzmaske 255.255.255.224. Zuletzt müssen wir noch die jeweiligen Subnetze Eintragen und das Gateway über das wir die anderen Hosts erreichen. Dazu bestätigen wir den Button Routes.



Der Button Routes, öffnet ein neues Fenster. Mit Add können wir nun die Subnetze, dessen Subnetzmaske und das Gateway unseres Edge Routers (Router 3) eintragen. Da wir von pnid4-svr-mu über Router 3 alle anderen Subnetze erreichen können ist das Gateway für alle Subnetze gleich und lautet 10.88.40.130. Abschließend bestätigen wir mit OK.



Damit alle Änderungen für diesen Host übernommen werden bestätigen wir noch mal mit Apply.

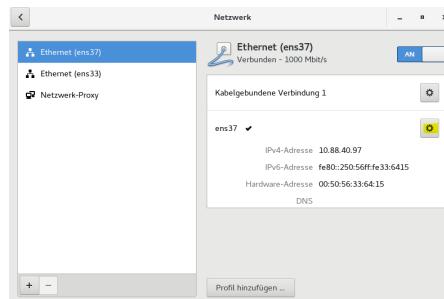


Das Netzwerk ist für den Host pnid4-svr-mu nun erfolgreich eingerichtet.

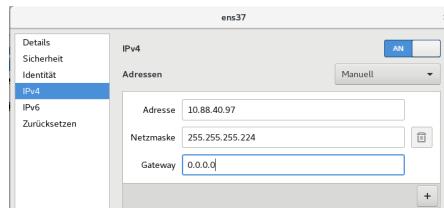
Konfiguration von Router 3 über die GUI:

Router 1, 2 und 3 laufen mit dem Betriebssystem Linux Centos 7. Dementsprechend ist die Netzwerkkonfiguration auf allen Routern die gleiche. Wir erläutern die Konfiguration von Router 3 ausführlich und zeigen dann, welche IP Adressen Router 2 und Router 1 besitzen.

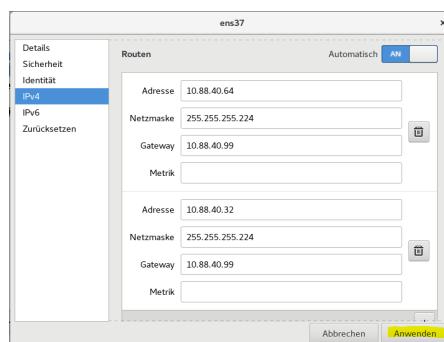
Wir Wählen zuerst unter dem Menüpunkt Anwendungen -> Systemsteuerung die Option Einstellungen aus. Im Auswahlmenü der Systemsteuerung wählen wir Netzwerk aus. Wir erhalten nun eine Netzwerkansicht bei der wir alle Netzwerk Interfaces ansehen können. Zurzeit existieren die Interfaces ens37 und ens33, da wir sie bereits beim Kopieren des Betriebssystems in der VMware Workstation festgelegt haben.



Mit dem Einstellungssymbol können wir die Schnittstellen konfigurieren. Wir wählen also ens37 aus und bestätigen mit Einstellungssymbol. Danach Navigieren wir an der linken Seite zu dem Namen IPv4. Falls noch nicht ausgewählt, stellen wir Adressen auf Manuell, damit wir eine statische IP Adresse vergeben können. Nun tragen wir die IP Adresse für die erste Schnittstelle des Routers ein. Als Gateway wählen wir 0.0.0.0 aus.



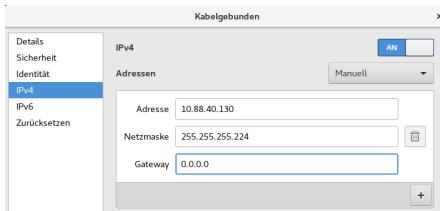
In der gleichen Ansicht wie zuvor scrollen wir etwas runter bis zum Abschnitt Routen und tragen nun die Adressen der Subnetze, die Subnetzmaske und das Gateway, über welches wir die Subnetze erreichen können, ein. Zuletzt bestätigen wir die Konfiguration mit Anwenden.



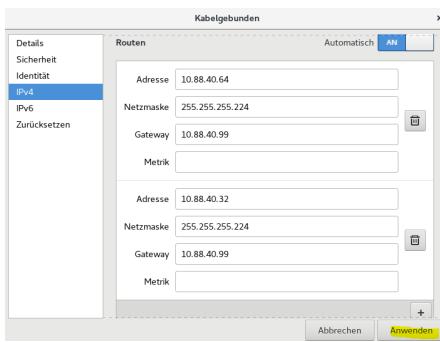
Damit haben wir das Interface mit dem Namen ens37 konfiguriert und müssen uns der zweiten Schnittstelle ens33 zuwenden. Wie eben können wir das Interface über den Button mit dem Einstellungssymbol konfigurieren.



Dieses Mal tragen wir die zweite statische IP Adresse für die zweite Schnittstelle ens33 ein.



Analog zum vorherigen Schritt scrollen wir runter und tragen die IP Adressen für die Subnetze, Subnetzmasken und das Gateway ein. Die Subnetze, Subnetzmasken und das Gateway sind bei ens33 und ens37 absolut identisch.

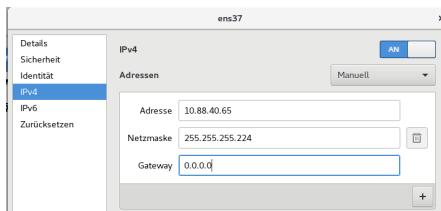


Zuletzt bestätigen wir die Konfiguration mit Anwenden. Der Router mit dem Namen pnid4-rou3 ist jetzt vollständig konfiguriert. Es folgen noch pnid4-rou2 (Router 2) und pnid4-rou1 (Router 1).

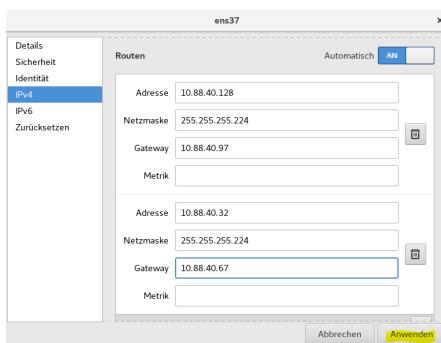
Konfiguration von Router 2 über die GUI:

Die Konfiguration auf allen Routern ist identisch. Lediglich die IP Adressen und das Routen unterscheidet sich, deshalb zeigen wir von nun an jeweils die Vergabe der statischen IP Adressen.

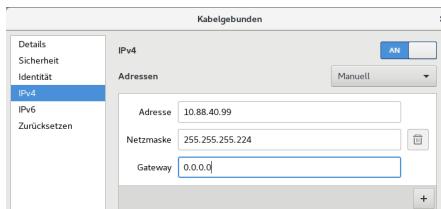
IP Adresse für das Interface ens37 festlegen:



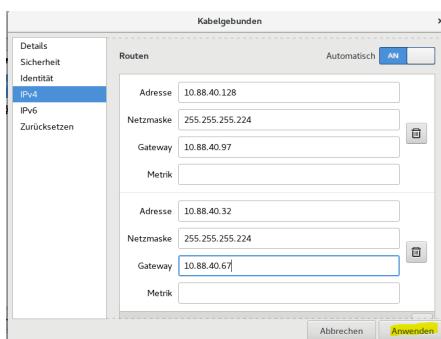
Adressen für die Subnetze festlegen und das jeweilige Gateway einstellen. Zuletzt mit Anwenden bestätigen.



IP Adresse für das Interface ens33 festlegen:



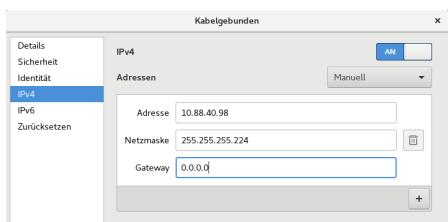
Adressen für die Subnetze festlegen und das jeweilige Gateway einstellen. Zuletzt mit Anwenden bestätigen. Damit ist die Konfiguration von Router 1 abgeschlossen.



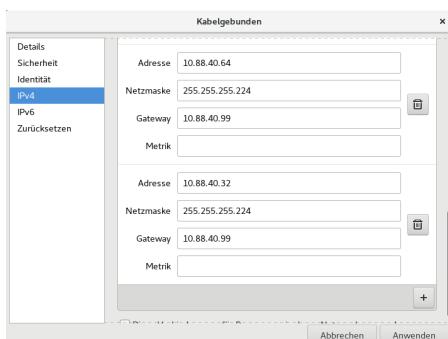
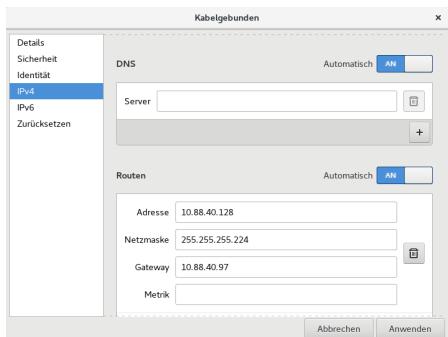
Konfiguration von Webserver Hannover über die GUI:

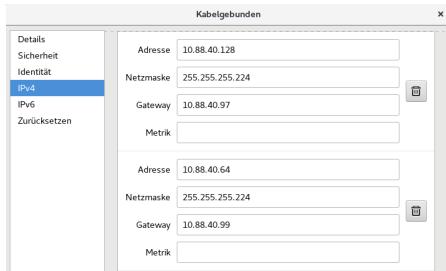
Dieser Host hat ebenfalls Linux Centos 7 als Betriebssystem, deshalb sind die Konfigurationsschritte dieselben wie bei den Routern.

IP Adresse für das Interface ens33 festlegen:



Adressen für die Subnetze festlegen, das jeweilige Gateway einstellen und die Routen eintragen.





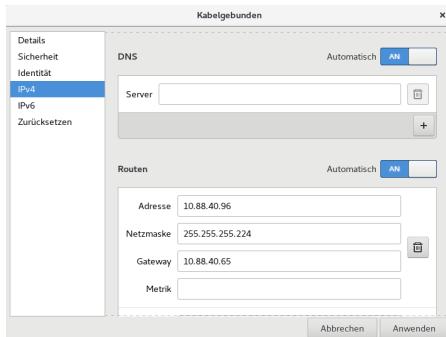
Zuletzt mit Anwenden bestätigen. Damit ist auch der Host pnid4-WEB-hn vollständig für die Subnetze konfiguriert.

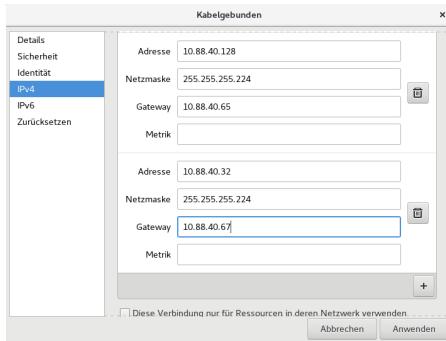
Konfiguration von Server Berlin über die GUI:

IP Adresse für das Interface ens33 festlegen:



Adressen für die Subnetze festlegen, das jeweilige Gateway einstellen und die Routen eintragen.

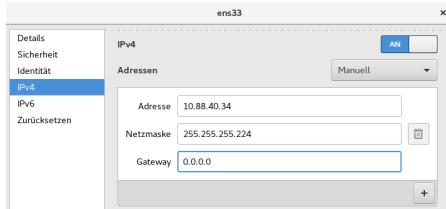




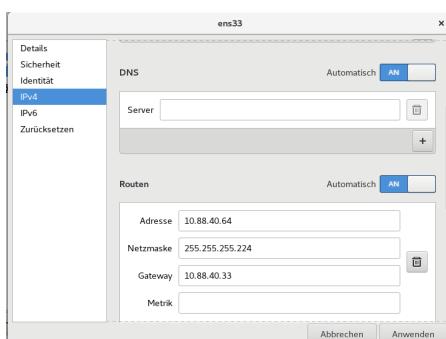
Zuletzt mit Anwenden bestätigen. Damit ist auch der Host pnid4-svr-bln vollständig für die Subnetze konfiguriert.

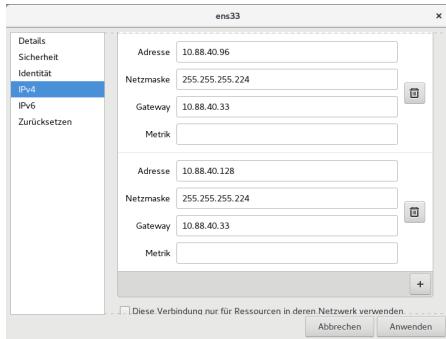
Konfiguration von Server Hamburg über die GUI:

IP Adresse für das Interface ens33 festlegen:



Adressen für die Subnetze festlegen, das jeweilige Gateway einstellen und die Routen eintragen.





Zuletzt mit Anwenden bestätigen. Damit ist auch der Host pnid4-svr-hh vollständig für die Subnetze konfiguriert.

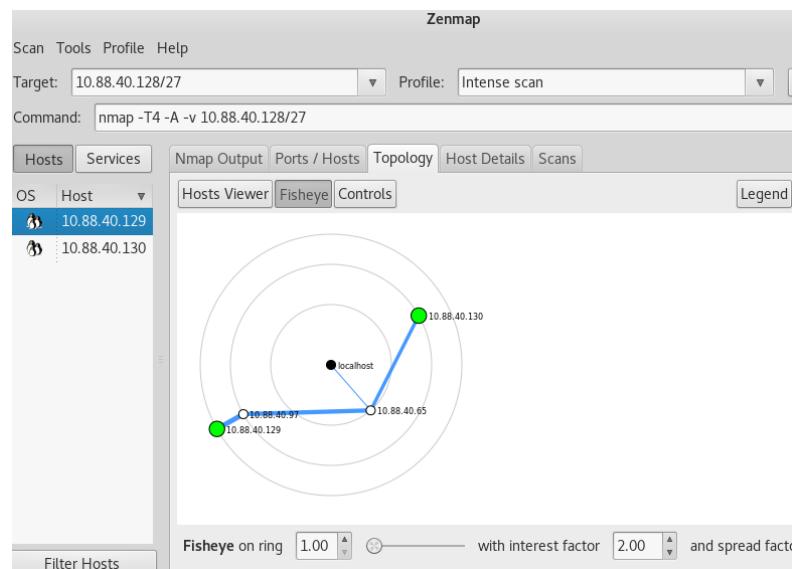
Zenmap Ergebnisse der gescannten Subnetze über die GUI:

Alle Hosts sind jetzt in ihren jeweiligen Subnetzen konfiguriert. Um einen Visuellen Überblick über die Netzwerktopologie der einzelnen Subnetze zu erhalten, können wir mithilfe der Zenmap die einzelnen Netzwerke scannen. Dazu melden wir uns auf dem Host mit dem Betriebssystem Kali Linux an, welcher die IP Adresse 10.88.40.69/27 hält und geben in das Terminal den Befehl # zenmap ein.

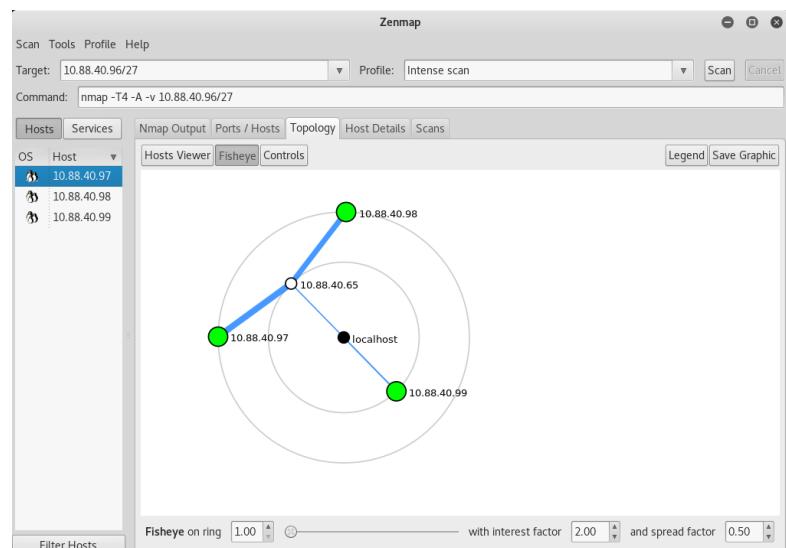
Scan des Subnetzes 10.88.40.64/27

In der Suchleiste Target können wir nun die IP Adresse des Subnetzes eingeben, welchen wir scannen möchten. Wir starten mit dem Subnetz 10.88.40.128 und bestätigen den Vorgang mit Scan.

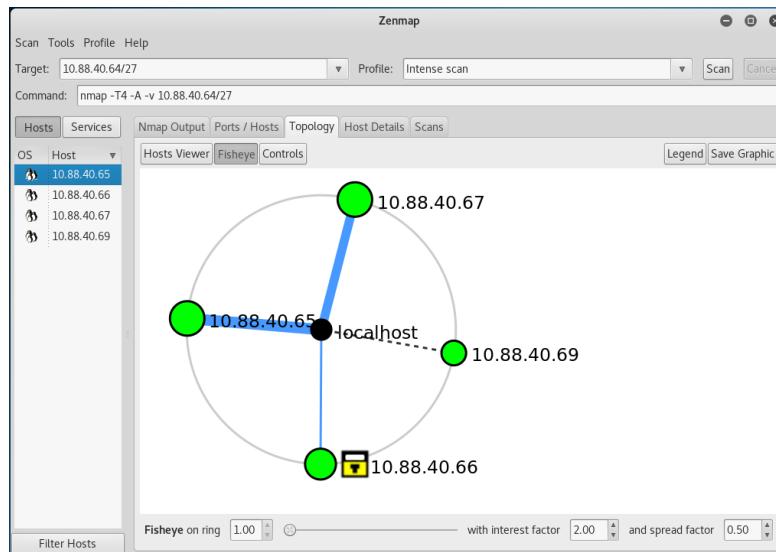
Scan des Subnetzes 10.88.40.128/27



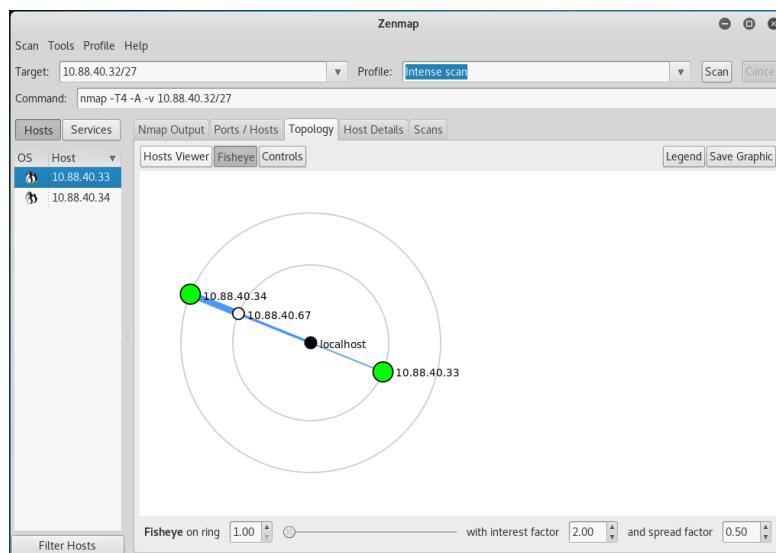
Scan des Subnetzes 10.88.40.96/27



Scan des Subnetzes 10.88.40.64/27



Scan des Subnetzes 10.88.40.32/27



Part 4: Network Scanning

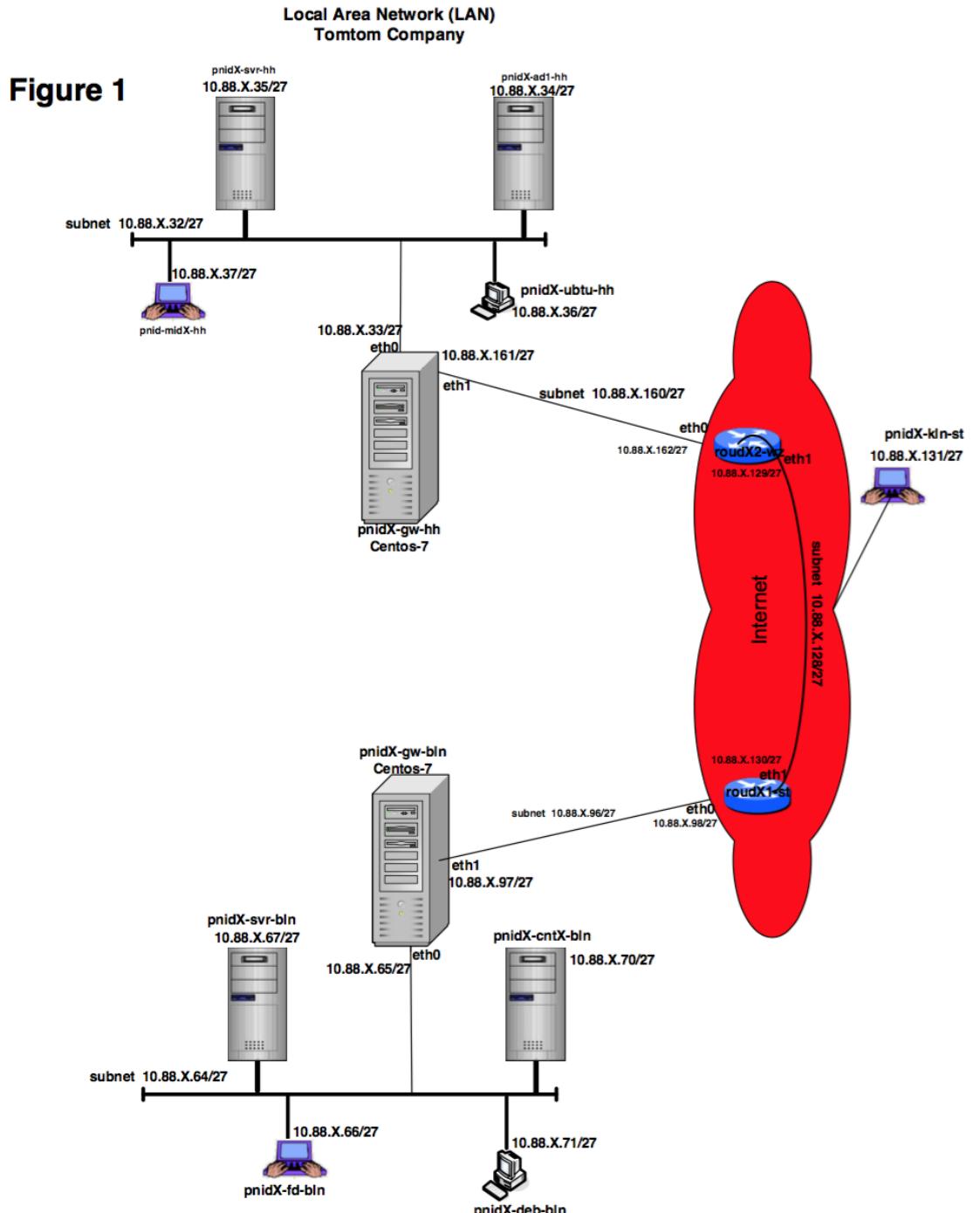


Abbildung 50: Netzwerk mit allen Hosts und Subnetzen

Exercise 1: Configure the networks of figure 1

a) Please copy, configure and set the networks for the following virtual machines provided by your instructor:

vm-Debian-8.5 copy from USB provided ,

vm-Ubuntu-16-10 copy from USB provided.

The password for the virtual machines is hamburg99tkrn for Ubuntu and Debian.

b) Please scan the following networks: 10.88.X.64/27, 10.88.X.96/27, 10.88.X.128/27, 10.88.X.160/27 and 10.88.X.32/27

c) Use Zenmap to scan all the above networks

Zenmap liefert uns alle statisch vergebenen IP Adressen der Hosts. Als Zusatz erhalten wir die Netzwerktopologie, ausgehend von dem aktuellen Host.

Solution of b)

Scan des Subnetzes 10.88.40.64/27



Abbildung 51: Subnetz 10.88.40.64/27

Scan des Subnetzes 10.88.40.96/27

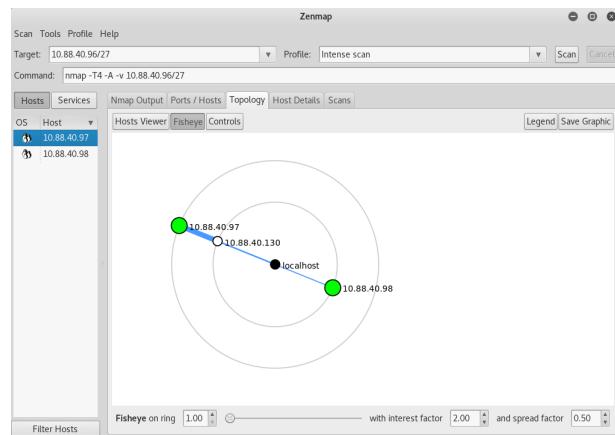


Abbildung 52: Subnetz 10.88.40.96/27

Scan des Subnetzes 10.88.40.128/27

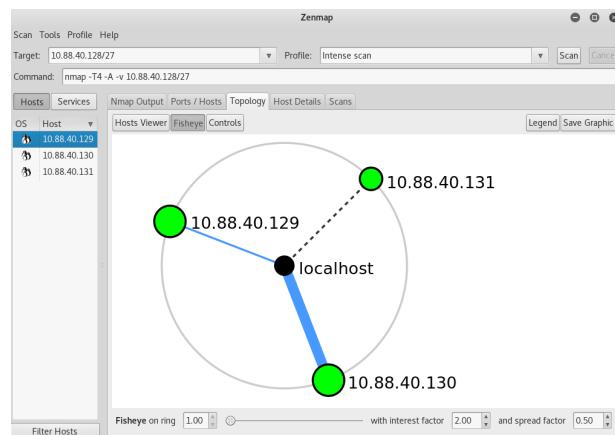


Abbildung 53: Subnetz 10.88.40.128/27

Scan des Subnetzes 10.88.40.160/27

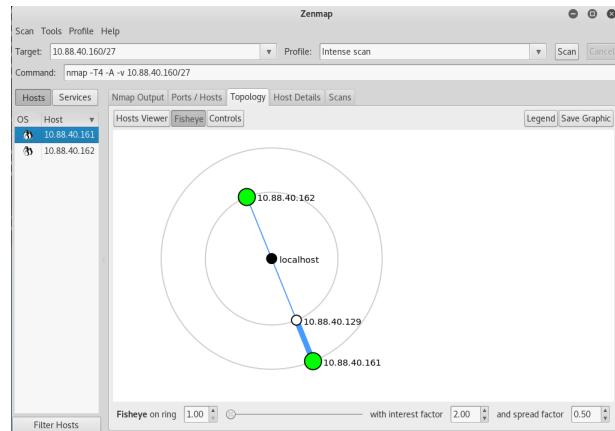


Abbildung 54: Subnetz 10.88.40.160/27

Scan des Subnetzes 10.88.40.32/27

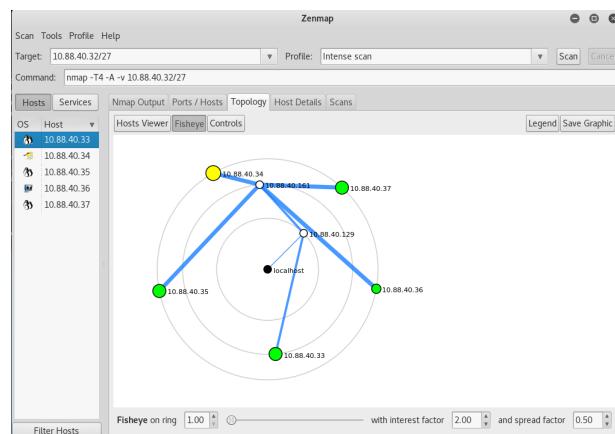


Abbildung 55: Subnetz 10.88.40.32/27

Solution of c) Use Zenmap to scan all the above networks

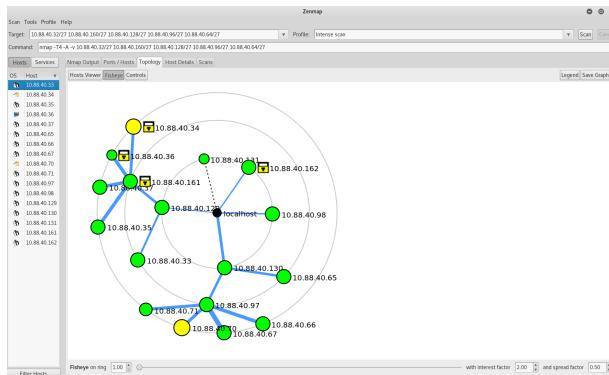


Abbildung 56: Alle Subnetze

Alternativ hätte man als Target auch folgendes in die Eingabemaske einfügen können:
10.88.40.32-160/27

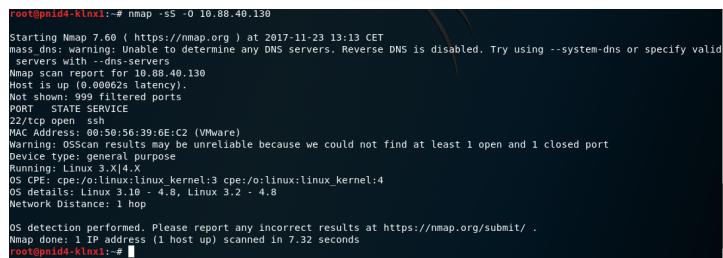
Exercise 2: NMAP

Analyze your host system and your virtual machines with Nmap. Please test the following commands before explaining the meaning.

- Q1: Please type and explain the nmap command: nmap -sS -O 10.88.X.?
- Q2: Please type and explain the nmap command: nmap -sF 10.88.X.?-oN outfile
- Q3: Please type and explain the nmap command: nmap -sS 10.88.X.?-D 10.100.X.P
- Q4: Please type and explain the nmap command: nmap -sS -O 10.88.X.Y/Z
- Q5: Please type and explain the nmap command: nmap -sP -PS 10.88.X.?
- Q6: Please type and explain the nmap command: nmap -sP -PS25 10.88.X.?
- Q7: Please type and explain the nmap command: nmap -sP -PS80 10.88.X.?/Z
- Q8: Please type and explain the nmap command: nmap -sP -PS53 10.88.X.?/27
- Q9: Please type and explain the nmap command: nmap -sS -v 10.88.X.?
- Q10: Please type and explain the nmap command: nmap -sP -v 10.88.X.?

Question 1: Please type and explain the nmap command: nmap -sS -O 10.88.X.?

Answer 1: Scannt das Subnetz 10.88.40.128 nach dem Host mit der IP Adresse 10.88.40.130 -sS bedeutet in diesem Zusammenhang SYN-Stealth-Scan. Dabei wird keine vollständige TCP/IP Verbindung aufgebaut und ist deshalb unauffälliger als das der Parameter -sT, welcher als einziger ohne root rechte Funktioniert. -O steht für OS-Detection. Es wird versucht, an besonderen Eigenarten der Netzwerkimplementierungen des Betriebssystems des Ziels zu identifizieren. Im Gegensatz zu Zenmap, wird nmap im Terminal durchgeführt und besitzt keine grafische Benutzeroberfläche um die Netzwerktopologie zu visualisieren.



```
root@pnid4-klnx1:~# nmap -sS -O 10.88.40.130
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-23 13:13 CET
Nmap: warning: unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid
      servers with dns-servers
Nmap scan report for 10.88.40.130
Host is up (0.00062s latency).
Not shown: 999 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
Nmap done: 1 IP address (1 host up) scanned in 7.32 seconds
root@pnid4-klnx1:~#
```

Abbildung 57: nmap -sS -O 10.88.40.130

Question 2: Please type and explain the nmap command: nmap -sF 10.88.X.? -oN outfile

Answer 2: Bei diesem Scan wird das Subnetz 10.88.40.128/27 mit dem Argument -sF gescannt. -sF bezeichnet die Art des Scans bei der nur Pakete mit FIN-Flags zum Ziel Host gesendet werden. Durch das Argument -oN outfile erstellen wir ein externes Logfile, mit dem Namen outfile.

```

root@pnid4-klnx1:~#
File Edit View Search Terminal Help
root@pnid4-klnx1:~# nmap -sF 10.88.40.128/27 -oN outfile
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-23 12:19 CET
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 10.88.40.129
Host is up (0.00046s latency).
All 1000 scanned ports on 10.88.40.129 are open|filtered
MAC Address: 00:50:56:29:79:5E (VMware)

Nmap scan report for 10.88.40.130
Host is up (0.0016s latency).
All 1000 scanned ports on 10.88.40.130 are open|filtered
MAC Address: 00:50:56:39:6E:C2 (VMware)

Nmap scan report for 10.88.40.131
Host is up (0.000010s latency).
All 1000 scanned ports on 10.88.40.131 are closed

Nmap done: 32 IP addresses (3 hosts up) scanned in 43.88 seconds
root@pnid4-klnx1:~#

```

Abbildung 58: nmap -sF 10.88.40.128 -oN outfile

```

Open ▾ A outfile Save = ⌂ ⌂ ⌂
# Nmap 7.60 scan initiated Thu Nov 23 12:19:19 2017 as: nmap -sF -oN outfile 10.88.40.128/27
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --
system-dns or specify valid servers with --dns-servers
Nmap scan report for 10.88.40.129
Host is up (0.00046s latency).
All 1000 scanned ports on 10.88.40.129 are open|filtered
MAC Address: 00:50:56:29:79:5E (VMware)

Nmap scan report for 10.88.40.130
Host is up (0.0016s latency).
All 1000 scanned ports on 10.88.40.130 are open|filtered
MAC Address: 00:50:56:39:6E:C2 (VMware)

Nmap scan report for 10.88.40.131
Host is up (0.000010s latency).
All 1000 scanned ports on 10.88.40.131 are closed

# Nmap done at Thu Nov 23 12:20:03 2017 -- 32 IP addresses (3 hosts up) scanned in 43.88 seconds|

```

Abbildung 59: externes Logfile

Mit dem Befehl -sF werden die Ports, die gescannt werden manipuliert, in dem verfälschte TCP-Pakete versendet werden. Dadurch erhält man die Information, ob ein Port offen oder von einer Firewall geschützt ist. Die Ausgabe wird im Terminal angezeigt, sowie durch das Argument -oN outfile, in einer separaten outfile.txt Datei.

Question 3: Please type and explain the nmap command: Please type and explain the nmap command: nmap -sS 10.88.X.? -D 10.100.X.P

Answer 3: Dieser Befehl führt einen Decoy-Scan durch. Mit dem Argument -D 10.88.40.36 legen wir einen Köder aus, mit dem wir den Ziel Host / Subnetz scannen. Diese Methode wird verwendet um die eigene IP Adresse zu verbergen, jedoch sollte der Host, welcher als Köder benutzt wird eingeschaltet sein.

```

root@pnid4-klnx1:~# nmap -sS 10.88.40.130 -D 10.88.40.36
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-23 13:02 CET
mass dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid
servers with --dns-servers
Nmap scan report for 10.88.40.130
Host is up (0.00050s latency).
Not shown: 999 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
MAC Address: 00:50:56:39:6E:C2 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 4.73 seconds
root@pnid4-klnx1:~#

```

Abbildung 60: nmap -sS 10.88.40.130 -D 10.88.40.36

Question 4: Please type and explain the nmap command: nmap -sS -O 10.88.X.Y/Z

Answer 4: Der Befehl versucht alle erreichbaren Hosts im Netzwerk X, mit ihrer IP Adresse anzuzuzeigen. Zusätzlich wird -sS (SYN-Stealth-Scan) und -O (OS-Detection) verwendet.

```

root@pnid4-klnx1:~# nmap -sS -O 10.88.40.160/27
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-23 12:26 CET
mass dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid
servers with --dns-servers
Nmap scan report for 10.88.40.161
Host is up (0.00075s latency).
All 1000 scanned ports on 10.88.40.161 are filtered
Too many fingerprints match this host to give specific OS details

Nmap scan report for 10.88.40.162
Host is up (0.00045s latency).
Not shown: 999 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Linux 3.x|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.10 - 4.8, Linux 3.2 - 4.8
OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .

Nmap done: 32 IP addresses (2 hosts up) scanned in 26.72 seconds

```

Abbildung 61: nmap -sS -O 10.88.40.160/27

Question 5: Please type and explain the nmap command: nmap -sP -PS 10.88.X.?

Answer 5: Das Argument -sP ist der sogenannte Ping-Scan. Es werden alle Hosts ausgegeben, welche auf den Scan geantwortet haben. So kann die Verfügbarkeit eines Rechners im Netzwerk gezählt werden, sowie die Server-Verfügbarkeit überwacht werden. Das Argument -PS sendet ein leeres TCP-Paket mit gesetzten SYN-Flag. Ein SYN-Flag ist ein Synchronisations-Flag, bestehend aus einem Bit. Ist dieser Flag gesetzt, will der Sender eine Verbindung zum Empfänger aufbauen.

```
root@pnid4-klnx1:~# nmap -sP -PS 10.88.40.130
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-23 13:17 CET
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid
servers with --dns-servers
Nmap scan report for 10.88.40.130
Host is up (0.00069s latency).
MAC Address: 00:50:56:39:6E:C2 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.28 seconds
root@pnid4-klnx1:~#
```

Abbildung 62: nmap -sP -PS 10.88.40.130

Question 6: Please type and explain the nmap command: nmap -sP -PS25 10.88.X.?/Z

Answer 6: Mit dem Argument -sP wird die Ping-Scan Methode ausgewählt. -PS wird verwendet um SYN-Pakete, mit gesetztem SYN-flag über den Port 25 (SMTP) zu senden.

```
root@pnid4-klnx1:~# nmap -sP -PS25 10.88.40.130
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-23 13:49 CET
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid
servers with --dns-servers
Nmap scan report for 10.88.40.130
Host is up (0.00076s latency).
MAC Address: 00:50:56:39:6E:C2 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.29 seconds
root@pnid4-klnx1:~#
```

Abbildung 63: nmap -sP -PS25 10.88.40.130

Question 7: Please type and explain the nmap command: nmap -sP -PS80 10.88.X.?/Z

Answer 7: Mit dem Argument -sP wird die Ping-Scan Methode ausgewählt. -PS wird verwendet um SYN-Pakete, mit gesetztem SYN-flag über den Port 80 zu senden. Port 80 ist zuständig für den Hypertext Transfer Protocol (HTTP). HTTP verwendet das TCP-Protokoll und benutzt diesen am Port 80.

```
root@pnid4-klnx1:~# nmap -sP -PS80 10.88.40.130/27
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-23 13:50 CET
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid
servers with --dns-servers
Nmap scan report for 10.88.40.129
Host is up (0.0017s latency).
MAC Address: 00:50:56:29:79:5E (VMware)
Nmap scan report for 10.88.40.130
Host is up (0.0011s latency).
MAC Address: 00:50:56:39:6E:C2 (VMware)
Nmap scan report for 10.88.40.131
Host is up (0.0011s latency).
MAC Address: 00:50:56:39:6E:C2 (VMware)
Nmap done: 32 IP addresses (3 hosts up) scanned in 0.79 seconds
root@pnid4-klnx1:~#
```

Abbildung 64: nmap -sP -PS80 10.88.40.130/27

Question 8: Please type and explain the nmap command: nmap -sP -PS53 10.88.X.?/27

Answer 8: Hier wird ebenso ein TCP SYN-Scan auf dem Port 53 durchgeführt. Port 53 wird verwendet für das Domain Name System (DNS) und wird meist über UDP verwendet.

```
root@pnid4-klnx1:~# nmap -sP -PS53 10.88.40.130/27
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-23 13:51 CET
massdns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid
servers with --dns-servers
Nmap scan report for 10.88.40.129
Host is up (0.0017s latency).
MAC Address: 00:50:56:29:79:5E (VMware)
Nmap scan report for 10.88.40.130
Host is up (0.0013s latency).
MAC Address: 00:50:56:39:6E:C2 (VMware)
Nmap scan report for 10.88.40.131
Host is up.
Nmap done: 32 IP addresses (3 hosts up) scanned in 0.79 seconds
root@pnid4-klnx1:~#
```

Abbildung 65: nmap -sP -PS53 10.88.40.130/27

Question 9: Please type and explain the nmap command: nmap -sS -v 10.88.X.?/27

Answer 9: -sS-Der SYN-Scan ist eine Methode fürs schnelle Scannen und scannt dabei Tausende von Ports pro Sekunde, wenn es nicht von einer Firewall gestört wird. Der Syn-Scan schließt die TCP-Verbindungen nicht ab. Außerdem kann zwischen den Zuständen offen, geschlossen und gefiltert unterschieden werden. Da keine vollständigen TCP-Verbindungen hergestellt werden, wird dies auch als halboffenes Scannen bezeichnet. Ein SYN-Paket wird gesendet. Dann wird auf eine Antwort gewartet. Ein SYN/ACK gibt an, dass jemand auf dem Port lauscht. Dies ist an einem offenen Port erkennbar. RST jedoch bedeutet, dass der Port geschlossen ist. -v bedeutet, dass die Ausführlichkeitsstufe erhöht wird(verbosity-Level), um mehr Wirkung zu erzielen.

```
root@pnid4-klnx1:~# nmap -sS -v 10.88.40.130
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-23 13:52 CET
Initiating ARP Ping Scan at 13:52
Scanning 10.88.40.130 [1 port]
Completed ARP Ping Scan at 13:52, 0.22s elapsed (1 total hosts)
massdns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid
servers with --dns-servers
Initiating SYN Stealth Scan at 13:52
Scanning 10.88.40.130 [1000 ports]
Completed SYN Stealth Scan at 13:52, 14.92s elapsed (1000 total ports)
Nmap scan report for 10.88.40.130
Host is up (0.00044s latency).
Not shown: 999 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
MAC Address: 00:50:56:39:6E:C2 (VMware)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 15.28 seconds
Raw packets sent: 2981 (131.148KB) | Rcvd: 23 (1.556KB)
root@pnid4-klnx1:~#
```

Abbildung 66: nmap -sS -v 10.88.40.130

Question 10: Please type and explain the nmap command: nmap -sP -v 10.88.X.?

Answer 10: Ein Ping-Scan wird ausgeführt mit erhöhtem Verbosity Level.

```
root@pnid4-klnxi:~# nmap -sP -v 10.88.40.130
Warning: The -sP option is deprecated. Please use -sn
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-23 13:53 CET
Initiating ARP Ping Scan at 13:53
Scanning 10.88.40.130 [1 port]
Completed ARP Ping Scan at 13:53. 0.23s elapsed (1 total hosts)
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid
servers with --dns-servers
Nmap scan report for 10.88.40.130
Host is up (0.0000s latency).
MAC Address: 50:56:c6:6f:C2 (VMware)
Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.29 seconds
Raw packets sent: 1 (28B) | Rcvd: 1 (28B)
root@pnid4-klnxi:~#
```

Abbildung 67: nmap -sP -v 10.88.40.130

Exercise 3: Nessus network device identification

Nessus is a scanner program.

It includes following features:

- a) It is a vulnerability scanner
- b) It is a port scanner
- c) It is a host/device detection program
- d) It can be used to scan Netbios Servers e.g. Windows Servers and Samba Servers
- e) Nessus can be used as a penetrating testing tool
- f) It is a client-server-system. The server performs the actual scan but it is controlled through the client. Both client and server can be run on the same system

In this exercise you will download, install and configure Nessus-6.11.2-es7.x86_64.rpm (use the RPM from your instructor) Do the following steps to install and run nessus:

- g. # Go to the nessus website and register your product for home feed means free cost
<https://www.tenable.com/products/nessus/select-your-operating-system>#tos h. # rpm -Uvh Nessus-6.11.2-es7.x86_64.rpm on Centos 7
- i. # After registration open your email and copy the activation code and type your activation key
- j. Now open a web browser and type the following: <https://10.88.X.P:8834/>

Enter your login name and password that you enter while installation. Note: download the user manual to obtain more information or refer to the internet. Then you will perform a scan on all subnet as below and analyze the result. Exercise:

- 1.) Download Nessus from www.nessus.org (Linux Version Nessus-6.11.2 - es7.x86_64.rpm)
- 2.) Install the Nessus-6.11.2-es7.x86_64.rpm Binary on your Centos7 Virtual Machine
- 3.) Register Nessus to obtain the plugins (note: choose the offline method)
- 4.) Install the plugins
- 5.) Perform a host identification of the localhost
- 6.) Please scan the following networks: 10.88.X.64/27, 10.88.X.96/27, 10.88.X.128/27, 10.88.X.160/27 and 10.88.X.32/27 using nessus
- 7.) Perform a network device identification on your subnet 10.88.X.P/27, see figure 1

Einleitung: Nessus ist ein Netzwerk- und Vulnerability Scanner. Unter einem Vulnerability Scanner versteht man Computerprogramme. Diese sind dafür zuständig Zielsysteme auf Sicherheitslücken bzw. Schwachstellen zu untersuchen. Der Scanner hat somit Zugriff auf entsprechende Datenbanken, um Informationen zu Sicherheitsproblemen zu bekommen. Dazu gehören der Einsatz bzw. das Vorhandensein von unsicheren oder nicht benötigten Diensten, Fehler in der Konfiguration bzw. Anwendung von Passwort- und Benutzerrichtlinien sowie offene Ports.

Nessus beruht auf einem Client-Server-Prinzip. Hierbei wird auf einem Rechner der Nessusserver gestartet und im Anschluss wird eine Verbindung zu anderen Rechnern hergestellt. Sobald der Server gestartet wird, werden die Plug-ins geladen. Diese sind notwendig, da man Sicherheitslücken des Betriebssystems finden kann während des Scans eines Hostes. Nessus kann als Penetrationstest verwendet werden. Darunter versteht man Sicherheitstests eines Rechners oder von Netzwerken. Dabei wird die Sicherheit der Systembestandteile und Anwendungen eines Netzwerks überprüft. Die Mittel, die dafür verwendet werden sind Methoden, die ein Angreifer verwenden würden, um unautorisiert in das System einzudringen, weshalb dies Penetration bezeichnet wird. Man möchte somit vor Angriffen schützen.

Solution of 1) Download Nessus from www.nessus.org (Linux Version Nessus-6.11.2-es7.x86_64.rpm)

Zunächst geht man zur folgenden Webseite: <https://www.tenable.com/products/nessus/select-your-operating-system>.

Anschließend wird die entsprechende Datei für CentOS 7 runtergeladen. Diese ist: Nessus-6.11.3-es7.x86_64.rpm. Nach dem Runterladen muss man sich noch registrieren, damit man einen Aktivierungs Code bekommt. Diesen erhält man per Email. Nach dem Herunterladen wird die Binary, wie man unten im Bild sieht, in die virtuelle Maschine reinkopiert.

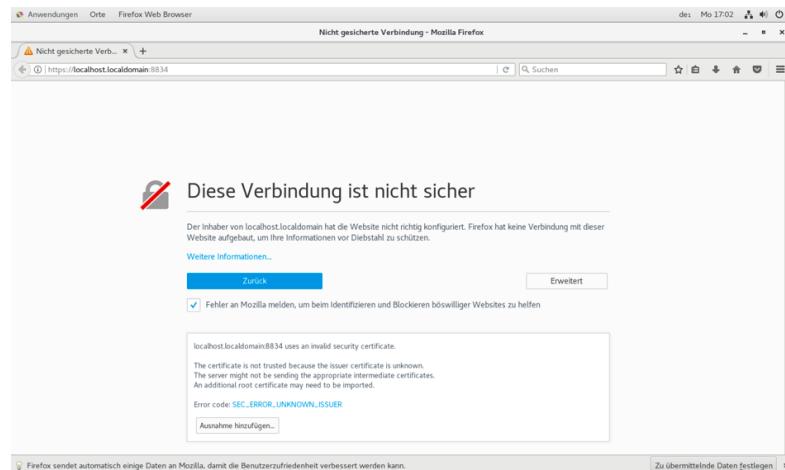
Solution of 2) Install the Nessus-6.11.2-es7.x86_64.rpm Binary on your Centos7 Virtual Machine

Um die Binary zu installieren wird der Befehl: rpm -Uvh /home/hanifka/Schreibtisch/Nessus-6.11.2-es7.x86_64.rpm im Terminal ausgeführt.

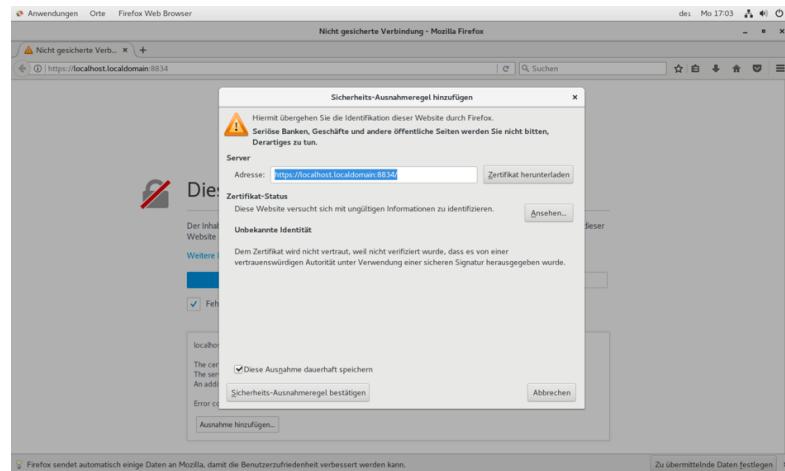
```
[root@localhost ~]# rpm -Uvh /home/hanifka/Schreibtisch/Nessus-6.11.2-es7.x86_64.rpm
Warning: /home/hanifka/Schreibtisch/Nessus-6.11.2-es7.x86_64.rpm: Header V4 RSA/SHA1 Signature, Schlüssel-ID 1c0c4a5d: NOKEY
Vorbereiten...
Aktualisierung/ Installation...
1:Nessus-6.11.2-es7
Unpacking Nessus Core Components...
nessusd (Nessus) 6.11.2 [build M20102] for Linux
Copyright (C) 1998 - 2017 Tenable Network Security, Inc
Processing the Nessus plugins...
[########################################] [100%]
All plugins loaded (1sec)
- You can start Nessus by typing /bin/systemctl start nessusd.service
- Then go to https://localhost.localdomain:8834/ to configure your scanner
[root@localhost ~]#
```

Abbildung 68: Installation von Nessus im Terminal

Nach erfolgreicher Installation kann Nessus mit folgendem Befehl gestartet werden: /bin/systemctl start nessusd.service. Anschließend wird der Browser geöffnet und man gibt folgenden Link ein: <https://localhost.localdomain:8834/>, um den Scanner konfigurieren zu können.



Damit dies funktioniert muss man auf den Button, Ausnahme hinzufügen, klicken. Danach muss man die Sicherheits-Ausnahmeregel bestätigen. Auch auf diesen Button wird geklickt.



Jetzt kann mit der Konfiguration gestartet werden.

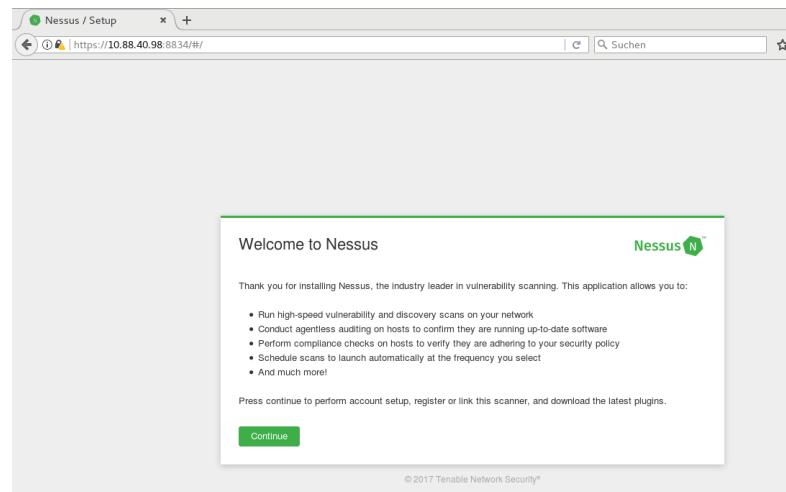


Abbildung 69: Nessus Konfiguration Startseite

Solution of 3) Register Nessus to obtain the plugins (note: choose the offline method)

Damit man sich erfolgreich registrieren kann, wählt man die offline Methode. Dann erhält man einen challenge code, den man fürs weitere Vorgehen benötigen wird.

The screenshot shows the 'Registration' page. At the top, it says 'As new vulnerabilities are discovered and released into the public domain, Tenable's research staff creates plugins that allow Nessus to detect their presence. These plugins contain vulnerability information, algorithms to test for the presence of the issue, and a set of remediation actions. Registering this scanner will grant you access to download these plugins.' A dropdown menu shows 'Offline' selected. Below it, a text box contains a challenge code: '7c237713c30156ac01921154d086a73491313538'. There is a 'Nessus License' field which is currently empty. At the bottom, there are 'Continue', 'Back', and 'Advanced Settings' buttons.

Abbildung 70: Nessus Registrierung

Im Browser wird nun folgender Tab geöffnet. Hier wird zuerst unser zu eben erzeugter challenge code eingegeben und unten wird der der Aktivierungs Code eingegeben, den man per Email nach der Registrierung erhalten hat.

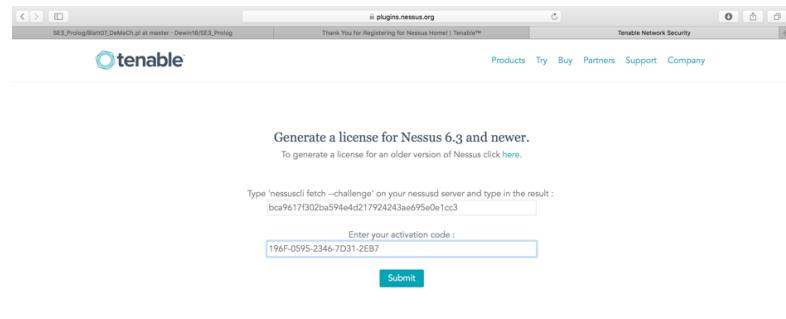


Abbildung 71: Nessus Aktivierungskey

Nach dem Klicken auf Submit erhalten wir unsere Nessus License.

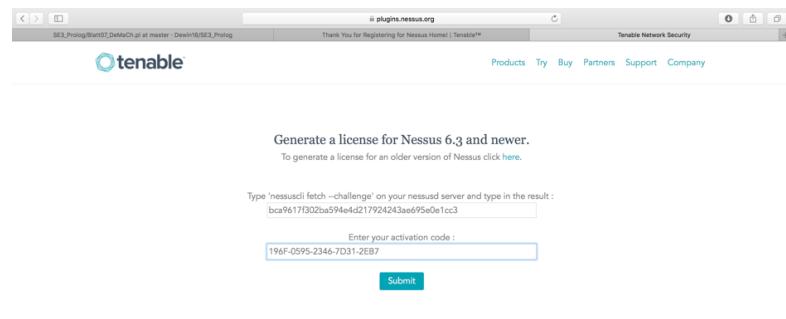
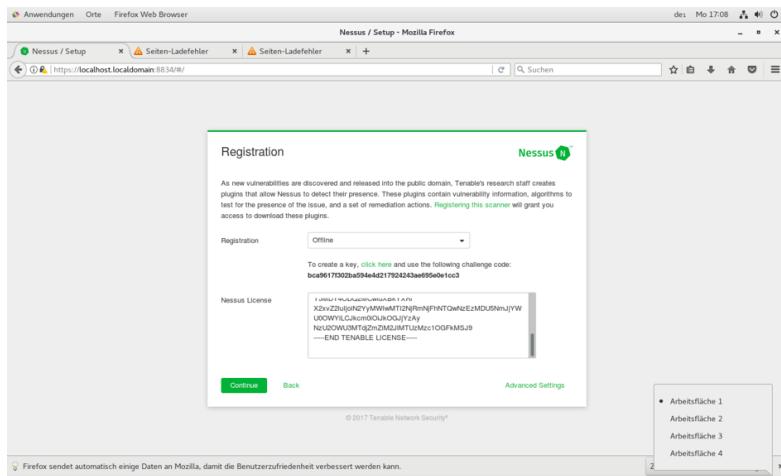
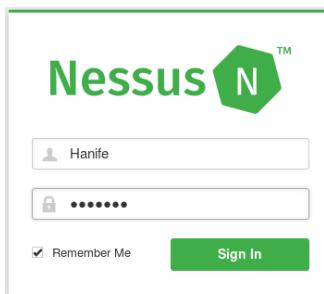


Abbildung 72: Nessus License

Die Nessus License kopieren wir und fügen diese ein, da sie für die erfolgreiche Registrierung notwendig ist.



Nach dem alles erfolgreich konfiguriert wurde und man sich erfolgreich registriert hat, kann man sich anmelden.

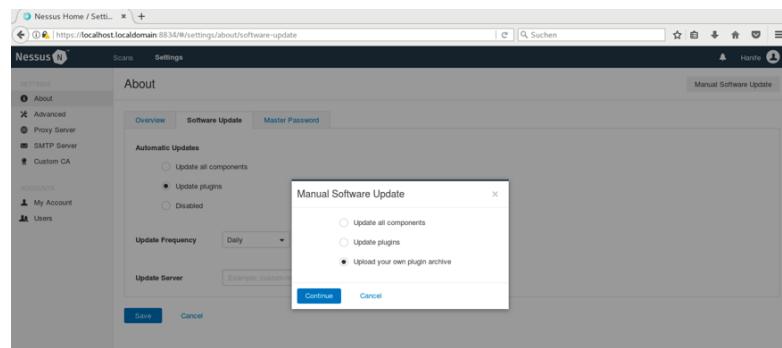


Solution of 4) Install the plugins

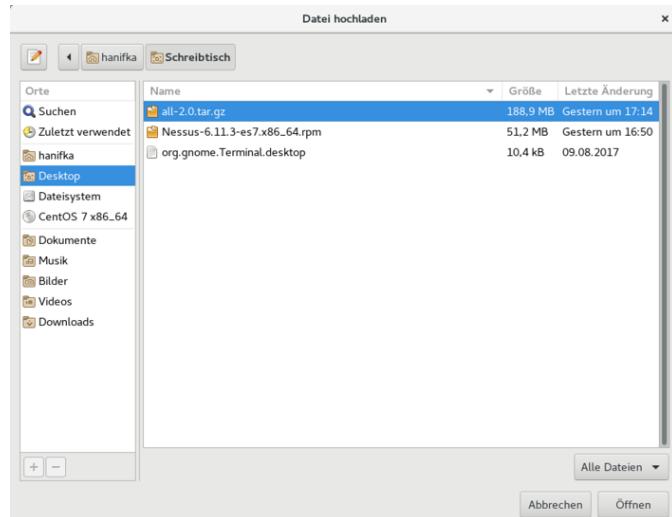
Um die Plugins zu installieren, laden wir die notwendige Datei all-2.0.tar.gz runter und kopieren diese anschließend in die virtuelle Machine. Dort kann sie wie folgt installiert werden übers Terminal oder manuell über die GUI.

```
[root@localhost sbin]# /opt/nessus/sbin/nessuscli update /home/hanifka/Schreibtisch/all-2.0.tar.gz
* Update successful. The changes will be automatically processed by Nessus.
[root@localhost sbin]#
```

Zunächst melden wir uns bei Nessus an. Unter dem Reiter Software Update können wir über den Button Manual Software Update unsere Datei all-2.0.tar.gz mit den ganzen Plugins reinladen.



Hier sieht man, dass die Datei all-2.0.tar.gz ausgewählt wird und anschließend geladen werden kann.



Solution of 5) Perform a host identification of the localhost and **Solution of 6)** Please scan the following networks: 10.88.40.64/27, 10.88.40.96/27, 10.88.40.128/27, 10.88.40.160/27 and 10.88.40.32/27 using nessus

Das Netz 10.88.40.32/27 wird gescannt. Wir führen eine Host Discovery durch. Die Scans verlaufen alle sehr schnell. Unter einer Host-Discovery versteht man Methoden oder Maßnahmen, um die erreichbaren Rechner in einem Netzwerk zu identifizieren. Im Allgemeinen versucht man die Rechner zu ermitteln, die über eine IP-Adresse erreichbar sind. Zuerst versucht man alle Hosts in einem Netzwerk zu identifizieren und danach erfolgt eine differenzierte Untersuchung des entsprechenden Hosts. Dadurch erhält man Informationen über die Hardware, das Betriebssystem, laufende Dienste

und offene Ports, welches man als Host-Scan bezeichnet.

Scan des Subnetzes 10.88.40.32/27:

Wir geben bei Targets das zugehörige Netz an, welches gescannt werden soll. Diesem können wir einen Namen und eine Beschreibung hinzufügen

The screenshot shows the Nessus web interface for creating a new scan. The left sidebar has sections for FOLDERS (My Scans, All Scans, Trash), RESOURCES (Policies, Plugin Rules, Scanners), and a navigation bar with Scans and Settings. The main area is titled 'New Scan / Host Discovery' with a 'Back to Scan Templates' link. It has tabs for Settings, BASIC (with General, Schedule, Notifications), DISCOVERY, REPORT, and a large 'Targets' section. In the Targets section, the 'Targets' input field contains '10.88.40.32/27'. Below the input field are 'Upload Targets' and 'Add File' buttons. At the bottom are 'Save' and 'Cancel' buttons.

Abbildung 73: Scan des Subnetzes 10.88.40.32/27

Hier sieht man alle zugehörigen Netze, die zum Subnetz 10.88.40.32/27 gehören, die durch Host-Discovery ermittelt wurden.

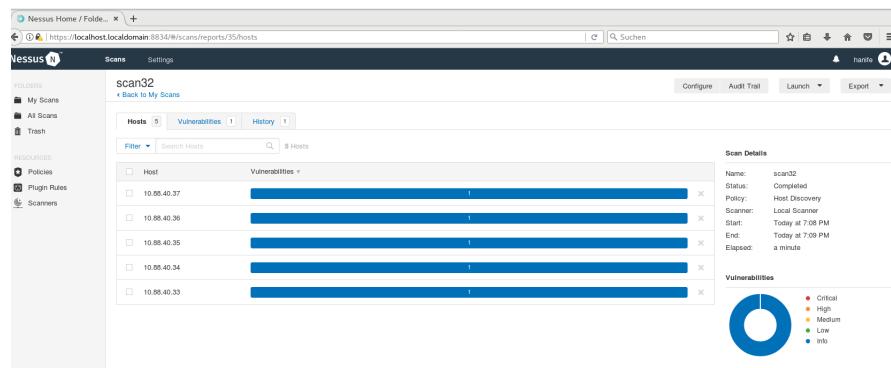


Abbildung 74: Alle Hosts im Subnetz 10.88.40.32/27

Scan des Subnetzes 10.88.40.64/27:

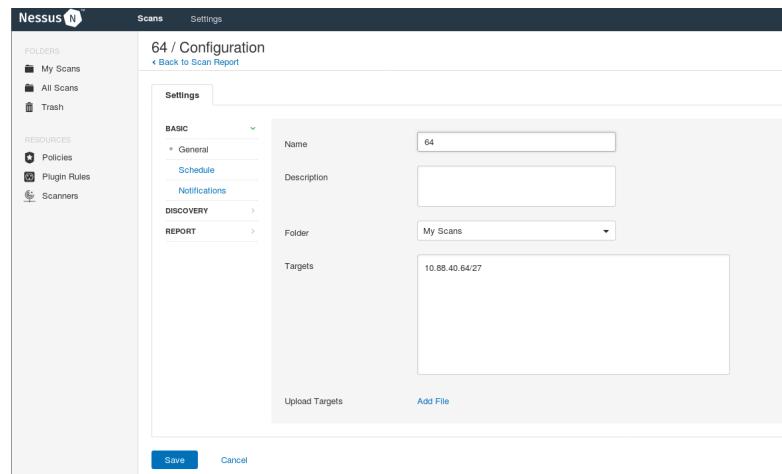


Abbildung 75: Scan des Subnetzes 10.88.40.64/27

Hier sieht man alle zugehörigen Netze, die zum Subnetz 10.88.40.64/27 gehören, die durch Host-Discovery ermittelt wurden.

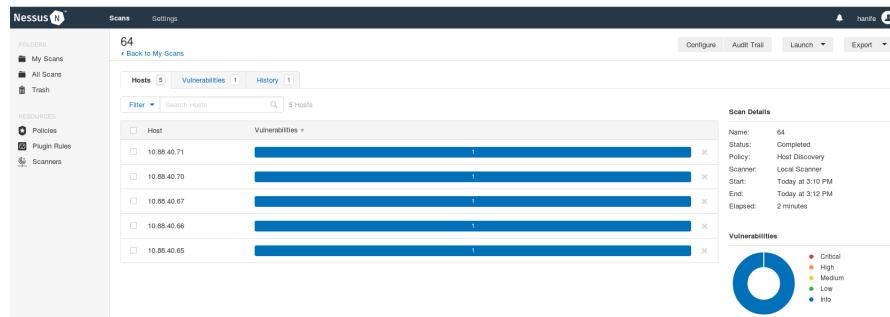


Abbildung 76: Alle Hosts im Subnetz 10.88.40.64/27

Scan des Subnetzes 10.88.40.96/27:

New Scan / Host Discovery

< Back to Scan Templates

Settings

BASIC

- + General
- Schedule
- Notifications

DISCOVERY

- REPORT

Name: 96

Description:

Folder: My Scans

Targets: 10.88.40.96/27

Upload Targets Add File

Save Cancel

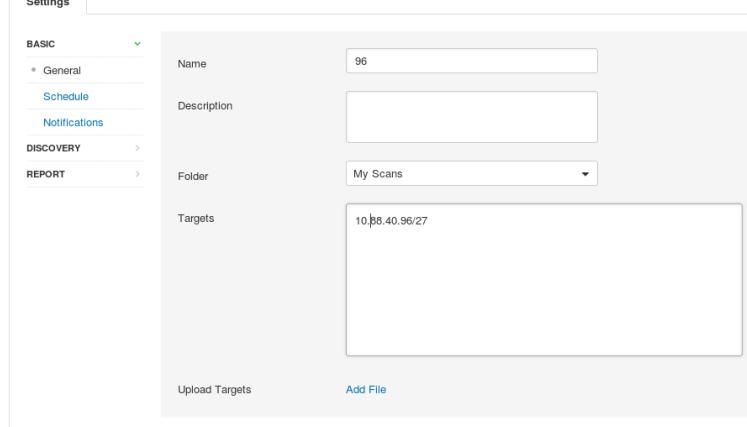


Abbildung 77: Scan des Subnetzes 10.88.40.96/27

Hier sieht man alle zugehörigen Netze, die zum Subnetz 10.88.40.96/27 gehören, die durch Host-Discovery ermittelt wurden.

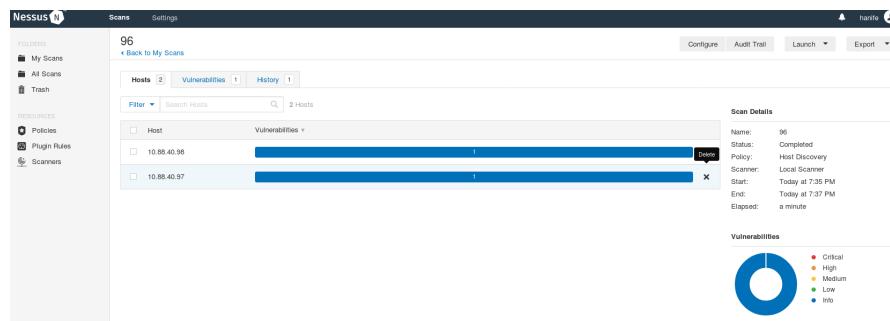


Abbildung 78: Alle Hosts im Subnetz 10.88.40.96/27

Scan des Subnetzes 10.88.40.128/27:

128 / Configuration

< Back to Scan Report

Settings

BASIC

- « General
- Schedule
- Notifications

DISCOVERY

REPORT

Name: 128

Description:

Folder: My Scans

Targets: 10.88.40.128/27

Upload Targets Add File

Save Cancel

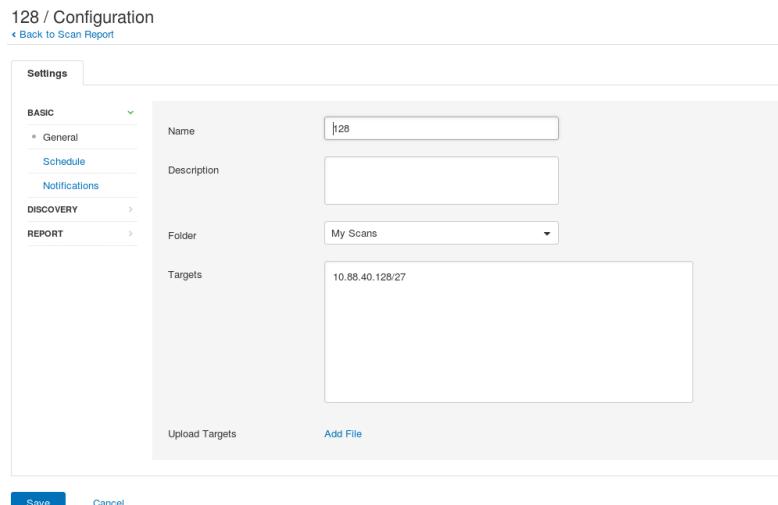


Abbildung 79: Scan des Subnetzes 10.88.40.128/27

Hier sieht man alle zugehörigen Netze, die zum Subnetz 10.88.40.128/27 gehören, die durch Host-Discovery ermittelt wurden.

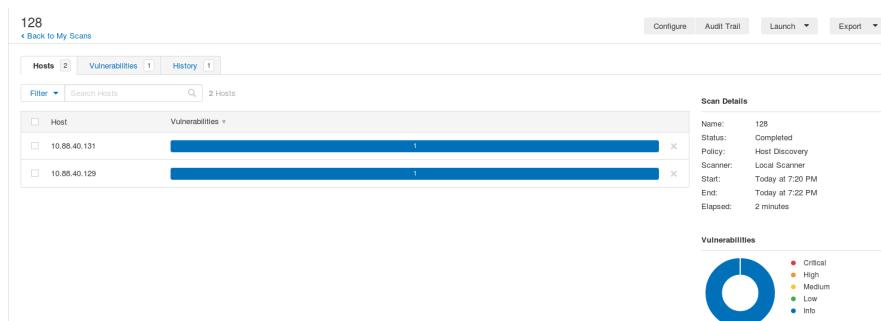


Abbildung 80: Alle Hosts im Subnetz 10.88.40.128/27

Scan des Subnetzes 10.88.40.160/27:

New Scan / Host Discovery

[Back to Scan Templates](#)

Settings

BASIC

- General
- Schedule
- Notifications

DISCOVERY

REPORT

Name: 160

Description:

Folder: My Scans

Targets: 10.88.40.160/27

Upload Targets Add File

Abbildung 81: Scan des Subnetzes 10.88.40.160/27

Hier sieht man alle zugehörigen Netze, die zum Subnetz 10.88.40.160/27 gehören, die durch Host-Discovery ermittelt wurden.

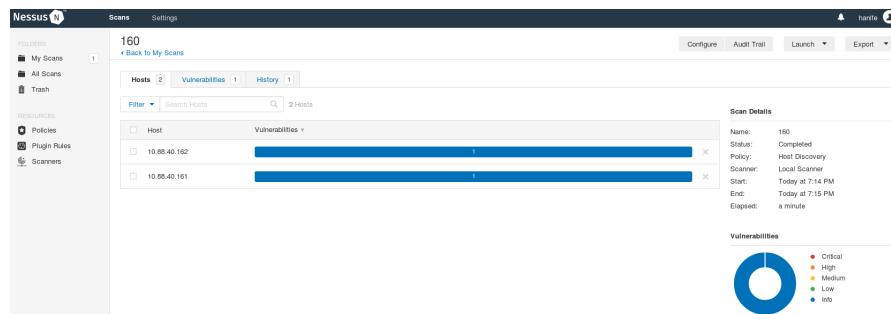


Abbildung 82: Alle Hosts im Subnetz 10.88.40.160/27

Alle Subnetze:

Im folgenden Bild wurden alle Subnetze gescannt, nämlich 10.88.40.64/27, 10.88.40.96/27, 10.88.40.128/27, 10.88.40.160/27 und 10.88.40.32/27.

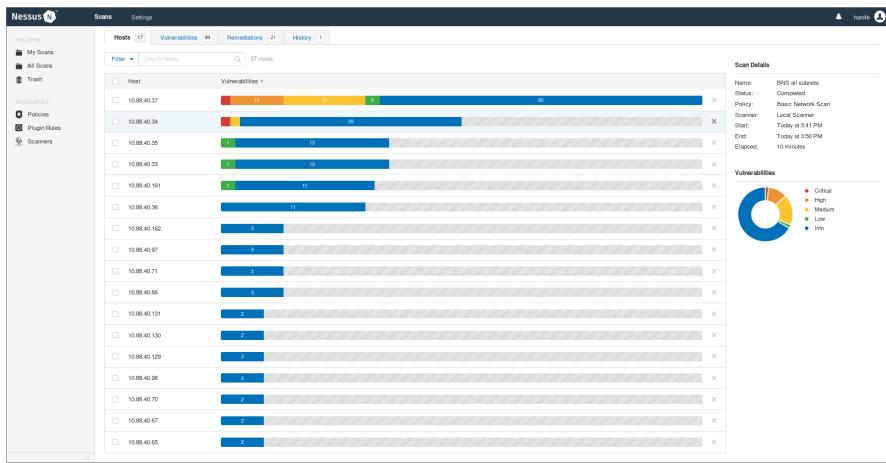


Abbildung 83: Alle Subnetze

Ergebnis aller Scans: Die Scans wurden als PDF-Datei extrahiert. Darin stehen alle relevanten Informationen, die über das erfolgreiche Scannen der Host Discovery ermittelt wurden. Die gesamte Datei befindet sich im Anhang der Email.

Solution of 7) Perform a network device identification on your subnet 10.88.X.P/27, see figure 1

set the ip for this machine to 172.16.15.P, where P stands for the ip address for your reserved ip, see below and change the dns to 134.100.9.61

group 1 will be 172.16.15.13
 group 2 will be 172.16.15.23
 group 3 will be 172.16.15.33
 group 4 will be 172.16.15.43
 group 5 will be 172.16.15.53
 group 6 will be 172.16.15.63
 group 7 will be 172.16.15.73
 group 8 will be 172.16.15.83
 group 9 will be 172.16.15.93
 group 10 will be 172.16.15.103

```
group 11 will be 172.16.15.113  
group 12 will be 172.16.15.123
```

Please download the installation guide and the user guide and read the Linux section carefully to solve the exercise.

Solution: Damit wir eine Internetverbindung von einem Hostrechner herstellen können wählen wir zuerst einen Host aus. Wir entschieden uns für pnid4-svr-hh mit dem Betriebssystem Centos 7. Als nächstes öffnen wir auf diesem Host die Netzwerkeinstellungen und fügen ein neue Schnittstelle (hier Profil 1) hinzu.

Da wir die Gruppe 4 sind, vergeben wir die IP Adresse 172.16.15.43 und tragen als Vorgaberoute die IP Adresse 172.16.15.249 ein. Der öffentliche DNS server der Uni-Hamburg läuft unter der IP Adresse 134.100.9.61, welche wir ebenfalls eintragen. Die Konfiguration über die GUI sieht dann folgendermaßen aus:

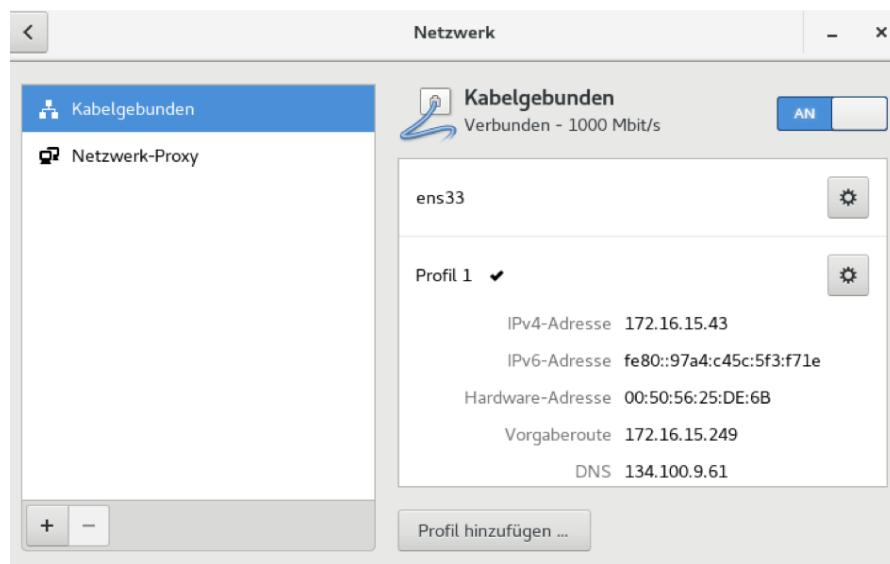


Abbildung 84: Netzwerkkonfiguration über die GUI

Die Internetverbindung ist nun erfolgreich hergestellt, sodass wir beliebige Webseiten, wie zum Beispiel www.google.de, auf unserer virtuellen Maschine (pnid4-svr-hh -> Server Hamburg) aufrufen können.

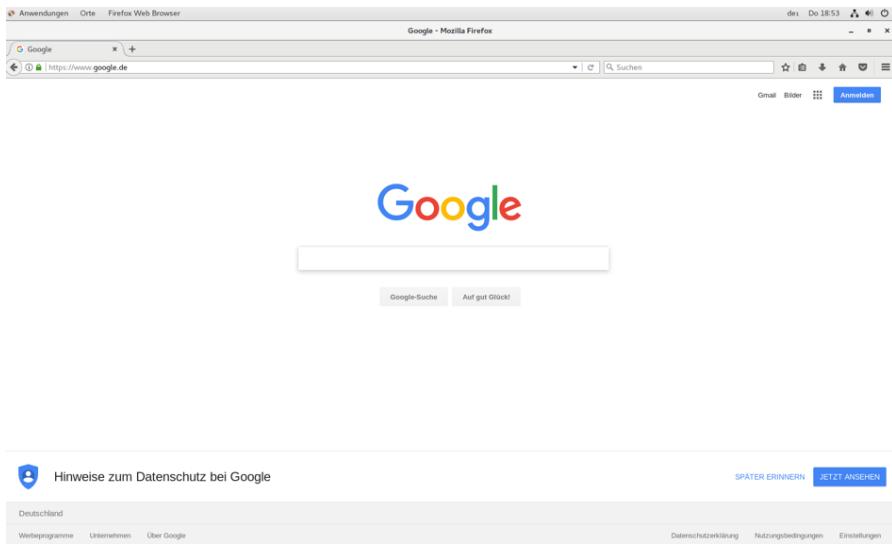


Abbildung 85: erfolgreiche Internetverbindung

Exercise 4: OpenVAS Network device identification

OpenVAS is a free scanner program. Look at the website https://www.bsi.bund.de/DE/Themen/ProdukteTools/OpenVAS/OpenVAS_node.html#doc2450430bodyText2 for more information or Google

OpenVAS includes following features:

- a) It is a vulnerability scanner
- b) It is a port scanner
- c) It is a host/device detection program
- d) It can be used to scan Netbios Servers e.g. Windows Servers and Samba Servers
- e) OpenVAS can be used as a penetrating testing tool
- f) It is a client-server-system. The server performs the actual scan but it is controlled through the client. Both client and server can be run on the same system

Do the following steps to install and run OpenVAS:

1. Create a new Nessus virtual machine as shown in figure 1.
2. Login with your user name if not existent, please create it.
3. #su- <cr> 4. # vi /etc/selinux/config Set SELINUX=permissive 5. # set the ip for this machine to 172.16.15.P, where P stands for the ip address for your reserved ip, see below and change the dns to 134.100.9.61

group 1 will be 172.16.15.13
group 2 will be 172.16.15.23
group 3 will be 172.16.15.33
group 4 will be 172.16.15.43
group 5 will be 172.16.15.53
group 6 will be 172.16.15.63
group 7 will be 172.16.15.73
group 8 will be 172.16.15.83
group 9 will be 172.16.15.93
group 10 will be 172.16.15.102
group 11 will be 172.16.15.113
group 12 will be 172.16.15.123

```
# apt-get update && apt-get install -y openvas  
# openvas-setup  
# openvasmd --user=admin --new-password=admin  
# openvas-start
```

6. Use your webbrowser <https://10.88.X.P:9392/>
7. Now you must use openVAS to identify all the devices running on the following network as depicted on figure 1:

10.88.X.32/27
10.88.X.64/27
10.88.X.96/27
10.88.X.128/27
10.88.X.160/27

Einleitung OpenVas

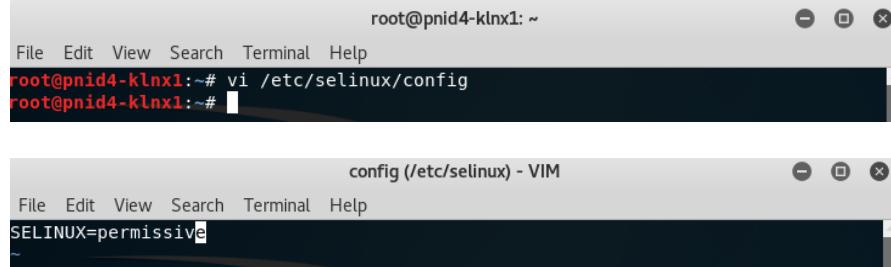
OpenVAS (Open Vulnerability Assessment System[1]) ist eine Open Source Software (OSS), die Sicherheitslücken im Netzwerk sammelt und grafisch darstellt. Dabei kann der Benutzer dieses Dienstes die Prüftiefen der Sicherheitsuntersuchungen individuell konfigurieren. Sobald ein Scan durchgeführt ist, stellt OpenVAS eine Vielzahl von informationsreichen Abbildungen dar. Ebenso hat der Benutzer die Möglichkeit in den Prüfberichten zu sehen, um noch genauere und detailliertere Auskünfte einzusehen. Die, durch den Scan, gefundenen Probleme werden in Sicherheitsklassen wie Low und Medium eingestuft, sodass eine Priorität der zu behebenden Schwachstellen sich automatisch erschließen lässt. Es gibt eine Menge Funktionalitäten die ein OpenVAS ermöglicht, einige dieser Funktionalitäten werden in folgendem Absatz stichpunktartig ausgelegt:

- Security-Scanning von ganzen Netzwerken
- Einbindung verschiedener Sicherheits-Tools dank entsprechender Schnittstellen und Steuerprotokolle
- Zentrale Sammlung und Auswertung der Prüfergebnisse
- Prüfung von Web Anwendungen möglich
- Management von Scan-Aufgaben
- Vergleich von Berichten
- Benutzeroberfläche -, GUI- und Web-basiert
- Alarmierung bei Richtlinien-Verstoß oder Gefahr

Installation: Um OpenVAS zu installieren, sollten wir nach Angaben im Skript folgende Aufgaben lösen:

1. Create a new Nessus virtual machine as shown in figure 1.
2. Login with your user name if not existent, please create it.
3. # su - <cr>
4. # vi /etc/selinux/config
Set SELINUX=permissive

So wie die Anleitung beschreibt haben wir die Schritte durchgeführt und SELINUX = permissive gesetzt.



```
root@pnid4-klnx1: ~
File Edit View Search Terminal Help
root@pnid4-klnx1:~# vi /etc/selinux/config
root@pnid4-klnx1:~#
```



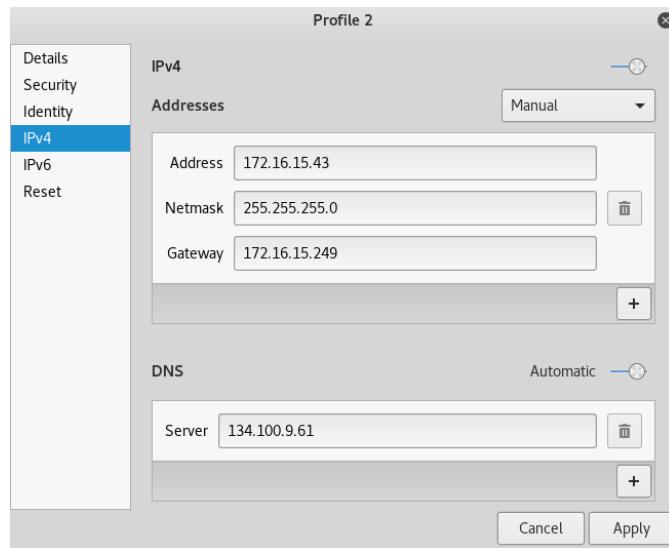
```
config (/etc/selinux) - VIM
File Edit View Search Terminal Help
SELINUX=permissive
~
```

SELinux (Security-Enhanced Linux) ist in allgemeinen eine Erweiterung des Linux Kernels. Es verwaltet die Berechtigungen von unterschiedlichen Unix-Systemen. Die Berechtigung kann von dem Benutzer selbst gesetzt werden.

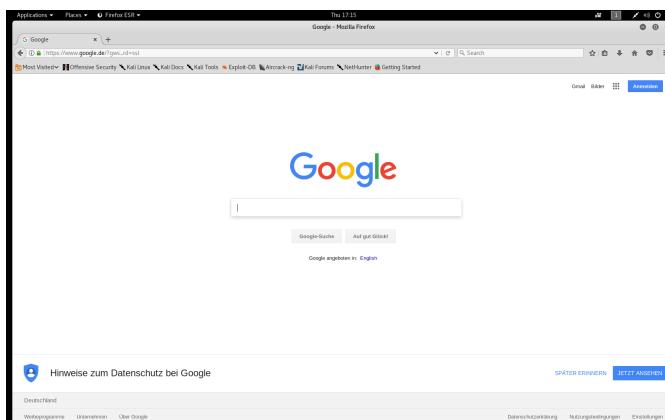
Es ist möglich SELinux im Enforcing oder Permissive Modus zu setzen. Enforcing, wie der Name selbst sagt, ist der Modus wo die SELinux Policy erzwungen wird. Im Gegenteil dazu werden im Permissive Modus alle Verletzungen protokolliert, aber nicht verhindert.

5. # set the ip for this machine to 172.16.15.P, where P stands for the ip address for your reserved ip, see below and change the dns to 134.100.9.61, group 4 will be 172.16.15.43

Wie die Aufgabe appelliert, haben wir die IP-Adresse 172.16.15.43 über den DNS 134.100.9.61 für pnid4-klnx1 konfiguriert.



Infolgedessen konnten wir über diese VM eine Verbindung zum Internet aufbauen, als Beispiel hier zu Google.



Nachdem wir IPv4 für diese VM eingerichtet haben, lassen wir die Paketlisten (`sources.list`) durch den Kommando `apt-get update` neu einlesen. Es wird auf die Signature dieser Paketlisten gepürft.

```
# deb cdrom:[Debian GNU/Linux 2017.2 Kali-rolling] - Official Snapshot amd64 LIVE/INSTALL Binary 20170917-01:51/ kali-rolling contrib main non-free
# deb cdrom:[Debian GNU/Linux 2017.2 Kali-rolling] - Official Snapshot amd64 LIVE/INSTALL Binary 20170917-01:51/ kali-rolling contrib main non-free
# deb http://http.kali.org/kali kali-rolling main
```

```

root@pnid4-klnx1:~# vi /etc/apt/sources.list
root@pnid4-klnx1:~# apt-get update
Get:1 http://ftp.halifax.rwth-aachen.de/kali kali-rolling InRelease [30.5 kB]
Get:2 http://ftp.halifax.rwth-aachen.de/kali kali-rolling/main amd64 Packages [15.6 MB]
Fetched 15.7 MB in 5s (2620 kB/s)
Reading package lists... Done
root@pnid4-klnx1:~#

```

Abbildung 86: apt-get update

Nachdem wir das update durchgeführt haben, installieren wir Pakete wenn es welche gibt auf eine aktuelle Version durch den Kommando apt-get upgrade.

```

root@pnid4-klnx1:~# apt-get upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
Retired Kali sana (2.0) Repositories
The following packages were automatically installed and are no longer required:
  docutils-docx-to-tools glibmm-1.0 libblas-common libgomp-1.0-common libqcustomplot1.3 python3.5 python3.5-minimal tcpd
Use 'apt autoremove' to remove them.
The following packages have been kept back:
  hdfnroxy hind9-host config dhconf-i18n dnsutils e2fslibs e2fsprogs_ndb_nrb_nir1_2-ndknbxbuf2_0_nir1_2-javascriptcoreatk-4_0_nir1_2-totem-1_0_nir1_2-webkit2-4_0_nis

```

Abbildung 87: apt-get upgrade

Und um openVas vollständig zu installieren, haben wir den Kommando apt -get install -y openvas genutzt. Das Installieren hat sehr viel Zeit in Anspruch genommen, denn über diesen Kommando werden nicht nur die Pakete installiert, sondern darüber hinaus noch nicht installierte Abhängigkeiten werden ebenfalls installiert. Man sieht dann auch im Terminal welche Pakete installiert wurden.

```

root@pnid4-klnx1:~# apt-get install -y openvas
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docutils-docx-to-tools glibmm-1.0 libblas-common libgomp-1.0-common libqcustomplot1.3 python3.5 python3.5-minimal tcpd
Use 'apt autoremove' to remove them.
The following additional packages will be installed:
  doc-base fonts-texgyre gnutls-bin greenbone-security-assistant greenbone-security-assistant-common libgnutls-dane0 libhiredis0.13 libmicrohttpd12 libopenvas9 libunbound2
  libuid-perl libyaml-tiny-perl openvas-cli openvas-manager openvas-scanner preview-latex-style redis-server redis-tools texlive-fonts-recommended
  texlive-fonts-recommended-dot texlive-latex-extra texlive-latex-recommended texlive-latex-recommended-doc texlive-pictures texlive-pictures-doc
  texlive-latex-plain-generic tipa
Suggested packages:
  rarian-compat openvas-client pscan strobe ruby-redist libgcc-profiles libfile-which-perl libgridsheet-parseexcel-perl texlive-pstricks dot2tex prereq ruby-tcltk | libtcltk-ruby
The following NEW packages will be installed:
  doc-base fonts-texgyre gnutls-bin greenbone-security-assistant greenbone-security-assistant-common libgnutls-dane0 libhiredis0.13 libmicrohttpd12 libopenvas9 libunbound2
  libuid-perl libyaml-tiny-perl openvas openvas-cli openvas-manager openvas-manager-common openvas-scanner preview-latex-style redis-server redis-tools tex-gyre
  texlive-fonts-recommended texlive-latex-extra texlive-latex-recommended texlive-latex-recommended-doc texlive-pictures texlive-pictures-doc
  texlive-latex-plain-generic tipa
0 upgraded, 31 newly installed, 0 to remove and 127 not upgraded.
Need to get 625 MB of archives.
After this operation, 945 MB of additional disk space will be used.
  Get:1 http://ftp.halifax.rwth-aachen.de/kali kali-rolling/main amd64 libuid-perl amd64 0.27-1+b2 [18.4 kB]
  Get:2 http://ftp.halifax.rwth-aachen.de/kali kali-rolling/main amd64 libyaml-tiny-perl all 1.70-1 [32.0 kB]
  Get:3 http://ftp.halifax.rwth-aachen.de/kali kali-rolling/main amd64 doc-base all 0.10.7 [100 kB]
  Get:4 http://ftp.halifax.rwth-aachen.de/kali kali-rolling/main amd64 fonts-texgyre all 20160520-1 [8761 kB]

```

Abbildung 88: OpenVas Installation

Anschließend haben wir OpenVas eingerichtet, indem wir folgenden Kommando in den Terminal eingegeben haben: `openvas-setup`. Wie der Name bereits verrät ist dieser Kommando für die Einrichtung für OpenVAS da.

```

root@pnid4-klnx1:~# systemctl list-unit-files |grep rsync
rsync.service                                         enabled
root@pnid4-klnx1:~# systemctl disable rsync.service
Synchronizing state of rsync.service with Sysv service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable rsync
root@pnid4-klnx1:~# systemctl list-unit-files |grep rsync
rsync.service                                         disabled
root@pnid4-klnx1:~# systemctl enable rsync.service
Synchronizing state of rsync.service with Sysv service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable rsync
root@pnid4-klnx1:~# systemctl list-unit-files |grep rsync
rsync.service                                         enabled
root@pnid4-klnx1:~# systemctl start rsync.service
root@pnid4-klnx1:~# systemctl |grep rsync
root@pnid4-klnx1:~# openvas-setup
OK: Directory for keys (/var/lib/openvas/private/CA) exists.
OK: Directory for certificates (/var/lib/openvas/CA) exists.
OK: CA key found in /var/lib/openvas/private/CA/cakey.pem
OK: CA certificate found in /var/lib/openvas/CA/cacert.pem
OK: CA certificate verified.
OK: Certificate /var/lib/openvas/CA/servecert.pem verified.
OK: Certificate /var/lib/openvas/CA/clientcert.pem verified.

OK: Your OpenVAS certificate infrastructure passed validation.
OpenVAS Community feed server - http://www.openvas.org/
This service is hosted by Greenbone Networks - http://www.greenbone.net/

All transactions are logged.

If you have any questions, please use the OpenVAS mailing lists
or the OpenVAS IRC chat. See http://www.openvas.org/ for details.

By using this service you agree to our terms and conditions.

Only one sync per time, otherwise the source ip will be blocked.

receiving incremental file list
plugin_feed_info.inc
    1,131 100%   1.08MB/s   0:00:00 (xfr#1, to-chk=0/1)

sent 43 bytes  received 1,235 bytes  852.00 bytes/sec
total size is 1,131  speedup is 0.88
OpenVAS Community feed server - http://www.openvas.org/
This service is hosted by Greenbone Networks - http://www.greenbone.net/

All transactions are logged.

If you have any questions, please use the OpenVAS mailing lists
or the OpenVAS IRC chat. See http://www.openvas.org/ for details.

By using this service you agree to our terms and conditions.

```

Abbildung 89: OpenVas Konfiguration

Im nächsten Schritt haben wir uns einen Benutzer und das dazugehörige Passwort für den Manager daemon über den Kommandoro `openvasmd --user=admin --new-password=admin` angelegt. Im Anschluss dessen haben wir die Services anhand Eingabe des `openvas-start` Kommando gestartet.

```

root@pnid4-klnx1:~# openvasmd --user=admin --new-password=admin
root@pnid4-klnx1:~# openvas-start
Starting OpenVas Services
root@pnid4-klnx1:~#

```

Abbildung 90: user anlegen

6. Use your webbrowser https:// 127.0.0.1:9392/

Im nächsten Schritt haben wir https://127.0.0.1:9392/ in die Adresszeile unseres Browsers eingegeben und haben unsere Login-Daten eingetippt.

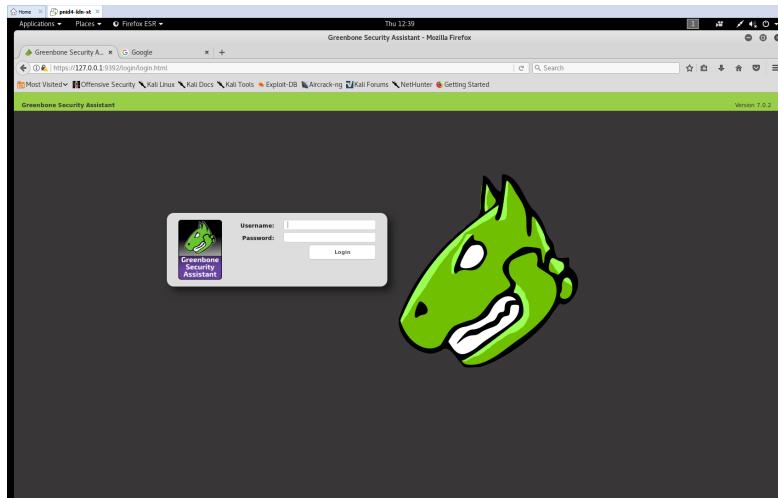


Abbildung 91: Login OpenVas

7. Now you must use openVAS to identify all the devices running on the following network as depicted on figure 1:

Daraufhin haben wir OpenVAS genutzt um alle VM in unserem Netz zu scannen. In Folgenden Bildausschnitten ist zu erkennen, welche VM (mit Angabe der IP-Adresse) gescannt wurde.

Scan des Subnetzes 10.88.40.32/27:

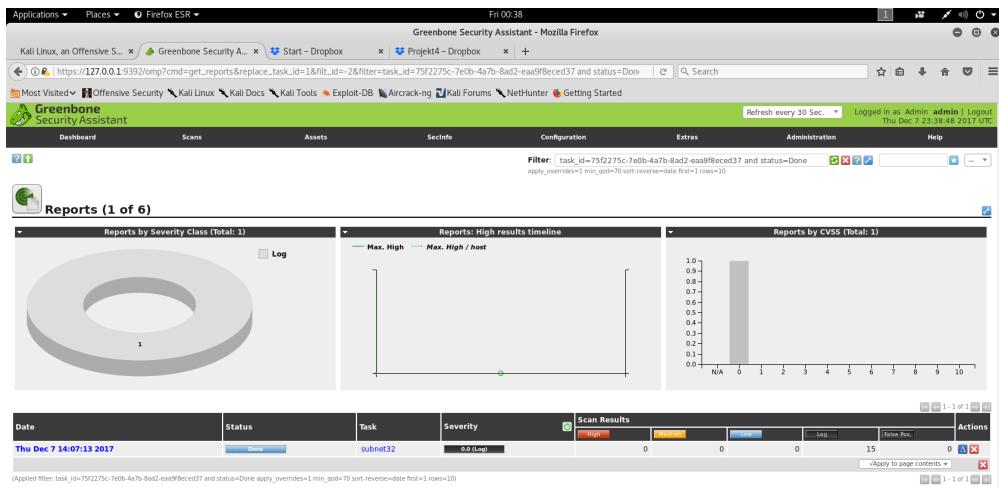


Abbildung 92: Scan des Subnetzes 10.88.40.32/27

Scan des Subnetzes 10.88.40.64/27:

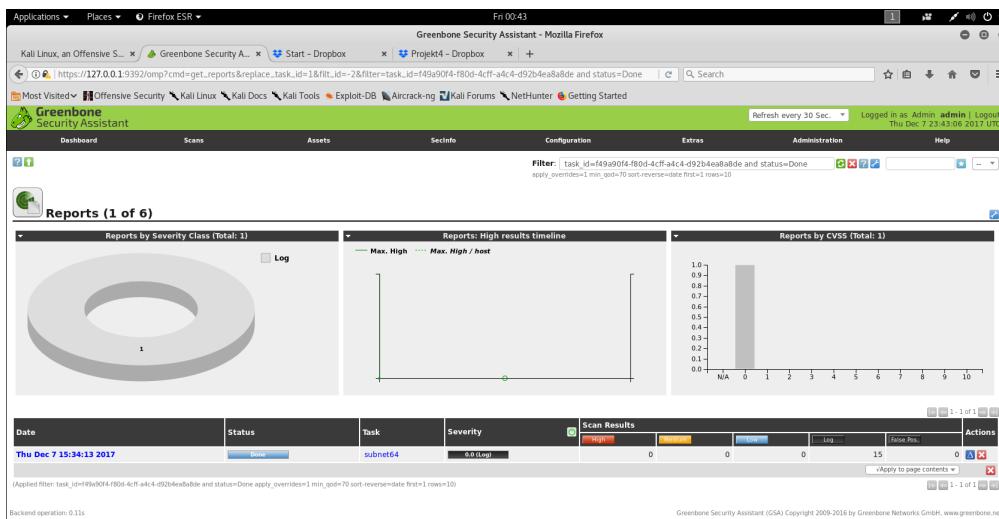


Abbildung 93: Scan des Subnetzes 10.88.40.64/27

Scan des Subnetzes 10.88.40.96/27:

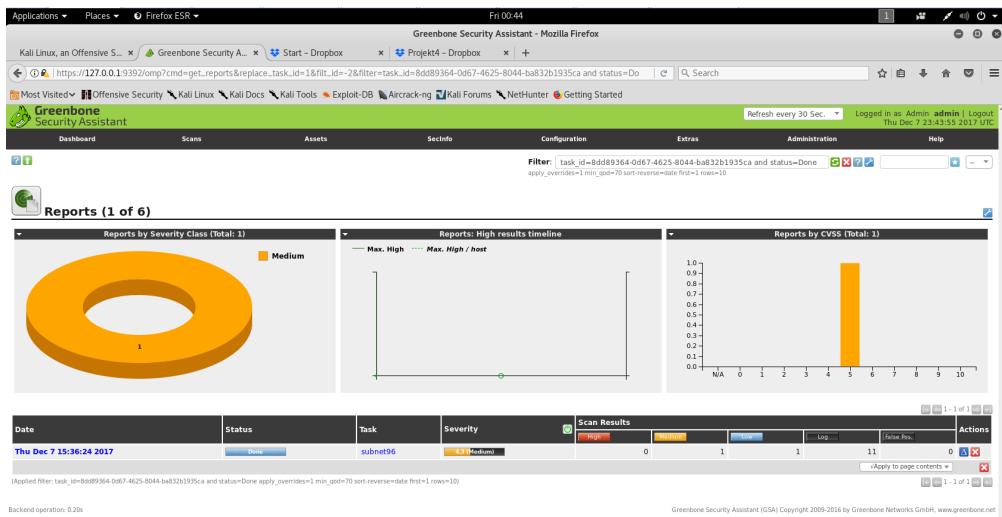


Abbildung 94: Scan des Subnetzes 10.88.40.96/27



Scan des Subnetzes 10.88.40.128/27:

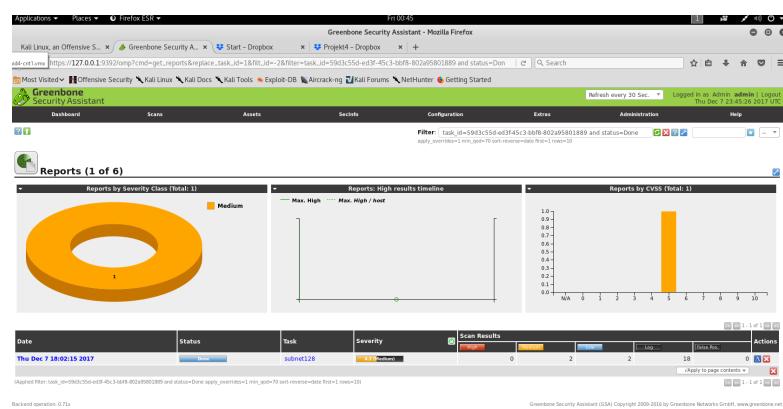


Abbildung 95: Scan des Subnetzes 10.88.40.128/27

Vulnerability	Severity	QoD	Host	Location	Actions
SSH Weak Encryption Algorithms Supported	47 (Medium)	95%	10.88.40.129	22/tcp	
SSH Weak Encryption Algorithms Supported	47 (Medium)	95%	10.88.40.130	22/tcp	
TCP Timestamps	24 (Low)	80%	10.88.40.129	general/tcp	
TCP Timestamps	24 (Low)	80%	10.88.40.130	general/tcp	

Scan des Subnetzes 10.88.40.160/27:

Date	Status	Task	Severity	Scan Results
Thu Dec 7 13:35:00 2017	Done	subnet160	47 Medium	0 1 1 11 0 0

Abbildung 96: Scan des Subnetzes 10.88.40.160/27

Vulnerability	Severity	QoD	Host	Location	Actions
SSH Weak Encryption Algorithms Supported	47 (Medium)	95%	10.88.40.162	22/tcp	
TCP Timestamps	24 (Low)	80%	10.88.40.162	general/tcp	

Ebenfalls ist hier eine Übersicht aller (10.88.40.32/27 ,10.88.40.64/27 ,10.88.40.96/27 10.88.40.128/27 und 10.88.40.160/27) gescannten Netze. Wir haben 5 Berichte durch den Scan erhalten, welche wir als PDF im Anhang beifügen werden.

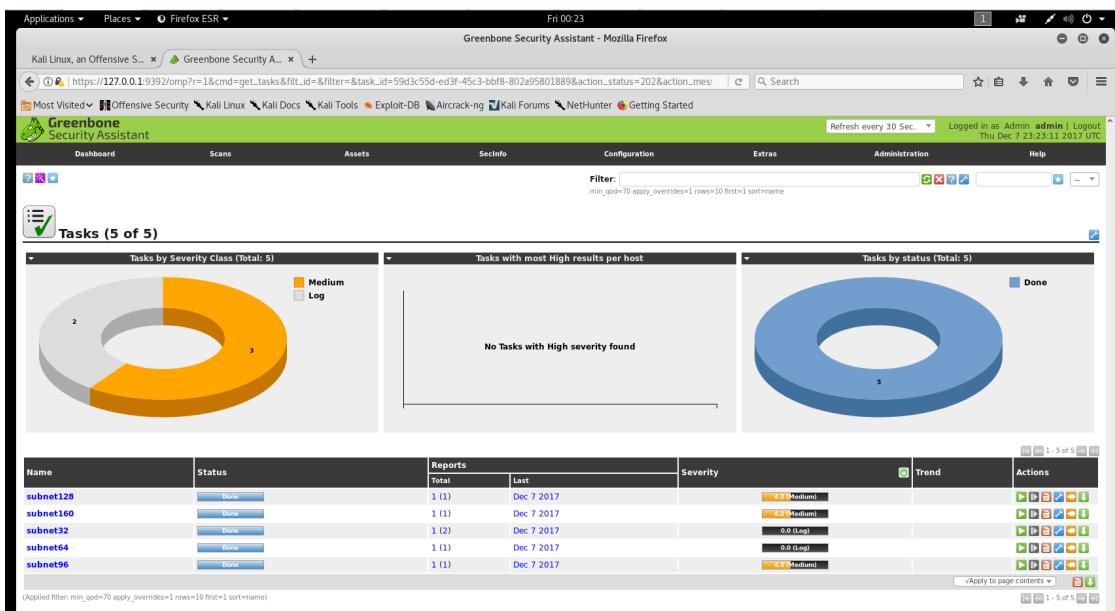


Abbildung 97: Scan aller Subnetze

Part 5: Sniffing, Virtual Private Network (VPN)

Theoretische Grundlagen

Sniffing

Der Begriff Sniffing stammt aus dem englischen und bedeutet übersetzt schnüffeln. Mit Hilfe eines Tools der Netzwerkanalyse, welches als ein Sniffer bezeichnet wird, ist das Sniffen möglich. Dies bedeutet unter anderem, dass der Datenverkehr eines Netzwerks auf Abnormitäten und potenziellen Gefahren überprüft werden kann. In Detail kann jedes übertragene Paket gelesen werden, Information der Quelle von Paketen und Zielort der Pakete festgestellt werden. Beispielweise kann man mit einem Sniffer Passwörter oder andere Daten auskundschaften.

Virtual Private Network

Virtual Private Network (kurz: VPN) ist eine Verschlüsselung, die es ermöglicht eine Verbindung über einen VPN-Server zu einem privaten Netzwerk in Echtzeit herzustellen. Dabei ist der Ort von dem man sich Verbinden möchte völlig irrelevant. Demzufolge ist beispielweise die Verbindung vom einem Rechner, welches sich in einem anderen Ort befindet als das Unternehmensnetzwerk, zum Unternehmensnetzwerk realisierbar. Da dies verschlüsselt funktioniert, ist es einem Dritten nicht möglich die übertragenen Informationen abzufangen.

Exercise 1: Configure and set the networks shown below (figure1 and 2)

a) Please configure the networks depicted on figure 1 and 2. Please scan the Network below(Figure 2) from pnidX-mid-hh to pnidX-cnt-blIn and from pnidX-cnt-blIn to pnidX-mid-hh.

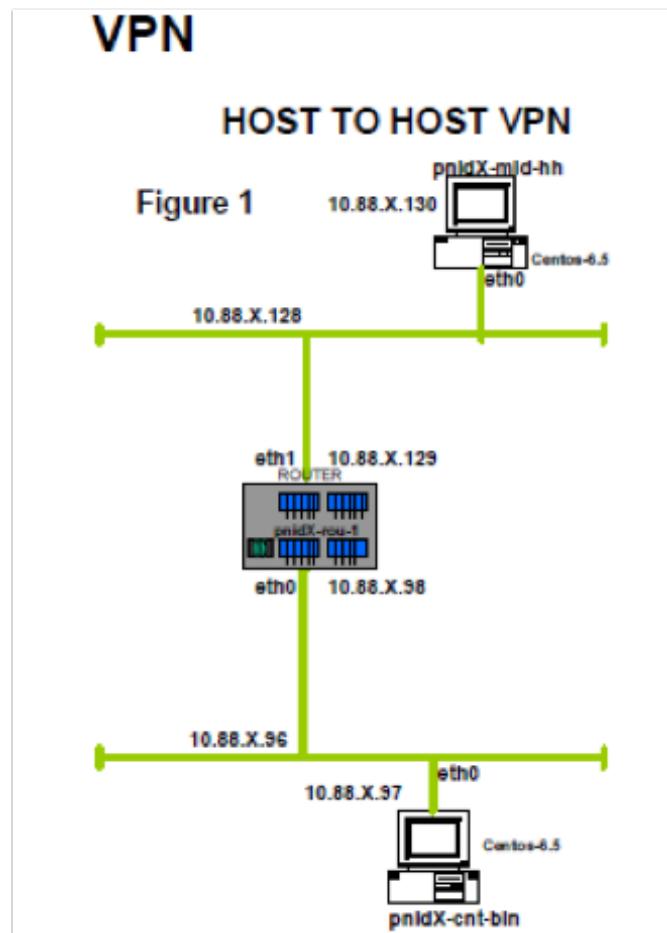


Abbildung 98: Abbildung Figur 1

Figure 2

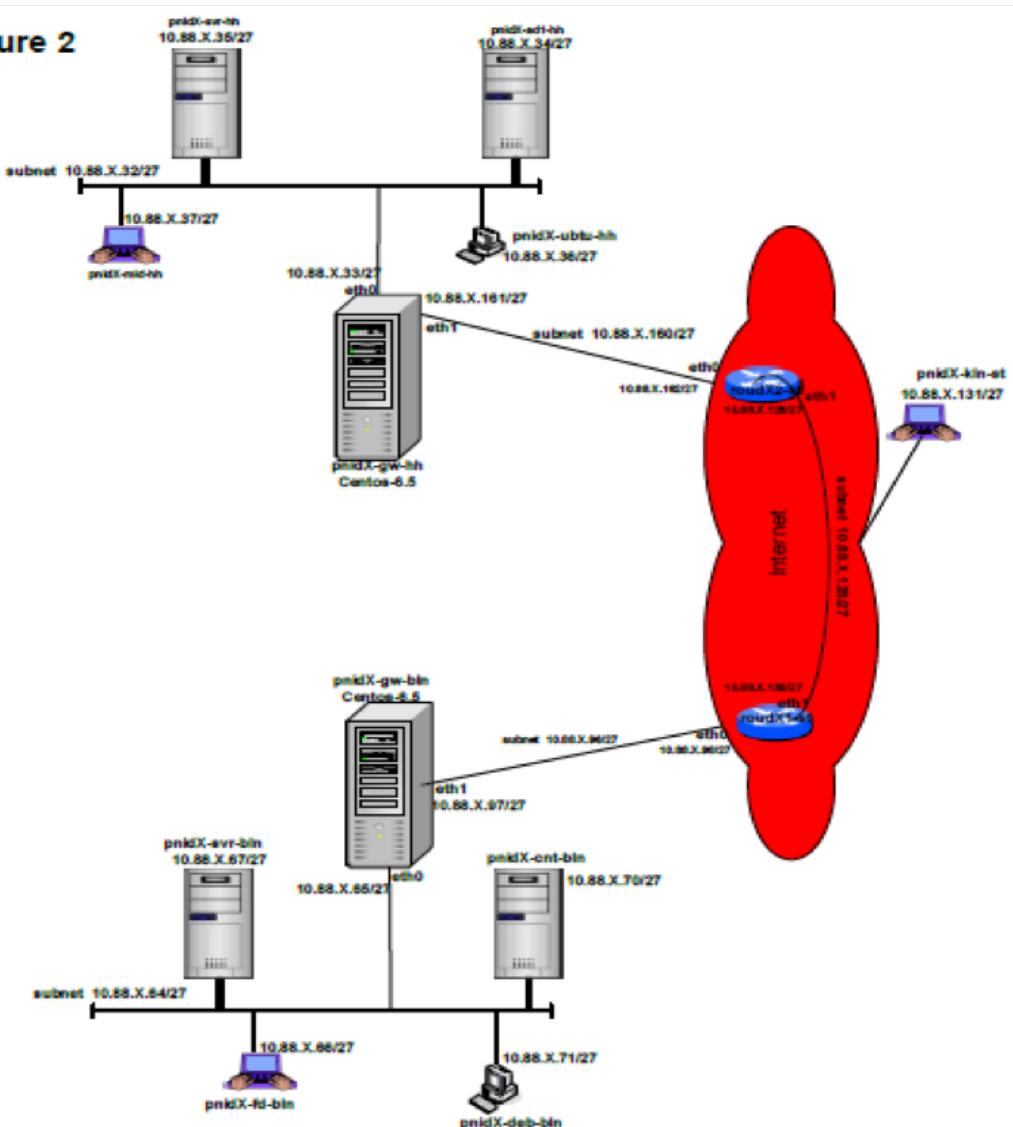


Abbildung 99: Abbildung Figur 2

Wie in der Aufgabe gefordert, haben wir die Netze genauso konfiguriert wie in es im Figur 1 und Figur 2 vorgegeben ist. Anschließend war es notwendig Zenmap auf beiden virtuellen Maschinen (`pnid4-mid-hh` und `pnid4-cnlt-bln`) zu installieren.

Zenmap ist ein Nmap Sicherheitsscanner - weitere Infos sind bereits auf S. 19 hinterlegt.

Für die Installation von Zenmap haben wir folgendes Kommando genutzt:

`yum install nmap-frontend`

```
[root@localhost yum.repos.d]# yum install nmap-frontend
Loaded plugins: fastestmirror, refresh-packagekit, security
Determining fastest mirrors
Loading mirror speeds from cached hostfile
LocalRepo/primary_db                                | 4.0 kB     00:00 ...
LocalRepo/primary_db                                | 4.4 MB     00:00 ...
Setting up Install Process
Resolving Dependencies
--> Running transaction check
--> Package nmap-frontend.noarch 2:5.51.3.el6 will be installed
--> Processing Dependency: nmap = 2:5.51 for package: 2:nmap-frontend-5.51-3.el6
.noarch
--> Running transaction check
--> Package nmap.x86_64 2:5.51-3.el6 will be installed
--> Finished Dependency Resolution
Dependencies Resolved

=====
Package           Arch   Version        Repository      Size
=====
Installing:
nmap-frontend    noarch  2:5.51-3.el6  LocalRepo       600 k
Installing For dependencies:
nmap             x86_64  2:5.51-3.el6  LocalRepo      2.7 M

Transaction Summary
=====
Install 2 Package(s)

Total download size: 3.3 M
Installed size: 12 M
Is this ok [y/N]: y
Downloading Packages:

Error Downloading Packages:
 2:nmap-frontend-5.51-3.el6.noarch: failure: Packages/nmap-frontend-5.51-3.el6.
noarch.rpm from LocalRepo: [Errno 256] No more mirrors to try.
```

Abbildung 100: Abbildung install nmap-frontend

Doch wie auf der letzten Abbildung zu sehen ist, konnte ein Paket nicht geladen werden. Dieses Paket haben wir dann auf der nächsten Abbildung explizit installiert, sodass wir Zenmap vollständig installiert haben.

```
[root@localhost ~]# yum install /root/Desktop/nmap-frontend-5.51-6.el6.noarch.rpm
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
Setting up Install Process
Examining /root/Desktop/nmap-frontend-5.51-6.el6.noarch.rpm: 2:nmap-frontend-5.5
1-6.el6.noarch
Marking /root/Desktop/nmap-frontend-5.51-6.el6.noarch.rpm to be installed
Resolving Dependencies
--> Running transaction check
--> Package nmap-frontend.noarch 2:5.51-6.el6 will be installed
--> Finished Dependency Resolution
Dependencies Resolved

=====
Package           Arch   Version        Repository      Size
=====
Installing:
nmap-frontend    noarch  2:5.51-6.el6  /nmap-frontend-5.51-6.el6.noarch  2.5 M

Transaction Summary
=====
Install 1 Package(s)

Total size: 2.5 M
Installed size: 2.5 M
Is this ok [y/N]: y
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : 2:nmap-frontend-5.51-6.el6.noarch
  Verifying   : 2:nmap-frontend-5.51-6.el6.noarch
  Installed:  nmap-frontend.noarch 2:5.51-6.el6
                                         1/1
                                         1/1
Complete!
```

Abbildung 101: Abbildung install nmap-frontend 2

Folglich haben wir wie in der Aufgabe das Netz von **pnid4-mid-hh** nach **pnid4-cnt-bl1n** gescannt. Somit sah es wie folgt aus:

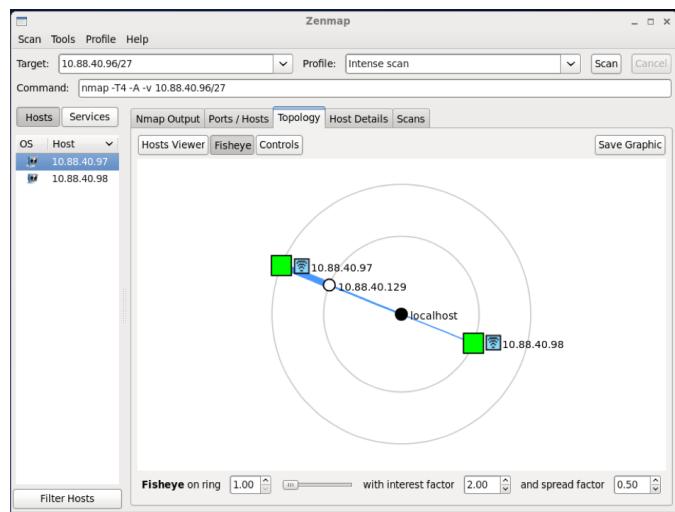


Abbildung 102: pnid4-mid-hh nach pnid4-cnt-bln

Im folgendem Screenshot ist das Netz **pnid4-mid-hh** nach **pnid4-cnt-bln** gescannt:

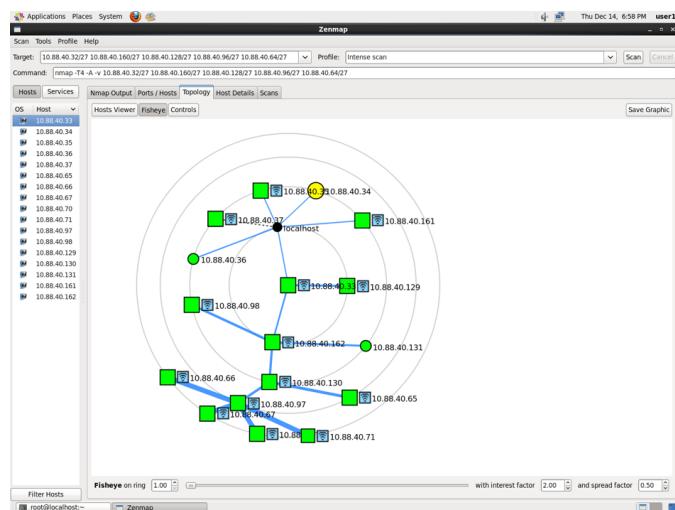


Abbildung 103: Abbildung pnid4-mid-hh nach pnid4-cnt-bln 2

Dasselbe taten wir dann auch für das Netz von **pnid4-cnt-bln** nach **pnid4-mid-hh**

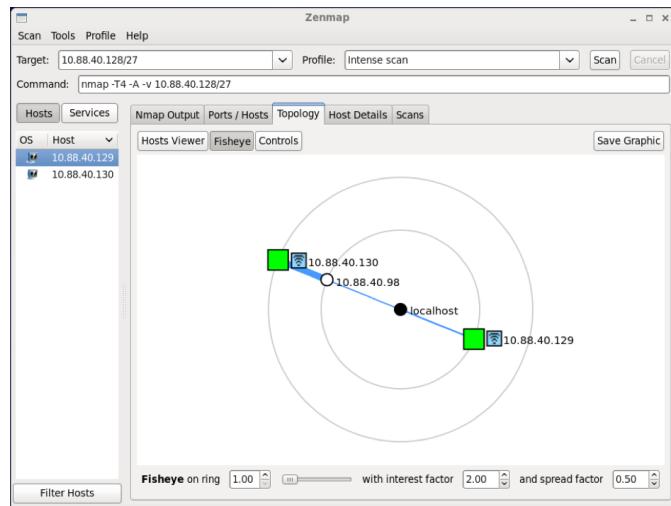


Abbildung 104: Abbildung pnid4-cnt-bln nach pnid4-mid-hh

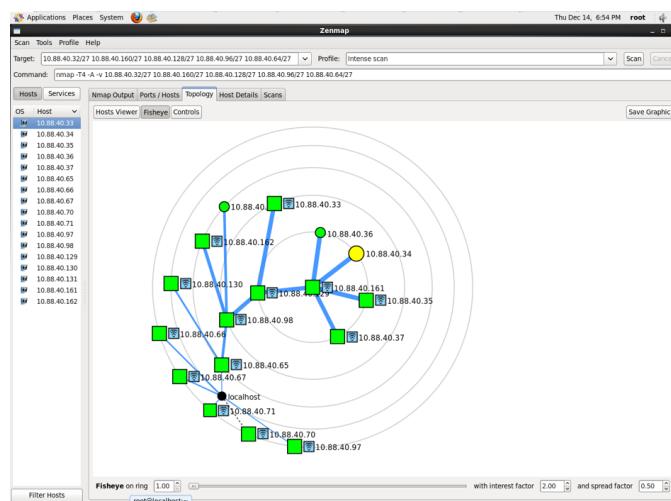


Abbildung 105: Abbildung pnid4-cnt-blن nach pnid4-mid-hh 2

Exercise 2: Getting started with network monitoring tools

Installation of network monitoring tools

Install Wireshark, Wireshark-gnome, vsftpd and telnetd from the Centos6.5 ISO on pnid4-mid-hh and pnid4-cnt-bln. Get familiar (e.g. by using the manuals) with the

following tools: tcpdump Wireshark

The tools and manuals are available on the Centos6.5 ISO. Please refer to the internet for more information. You need tcpdump and Wireshark on at least one workstation and on your server.

For each exercise, analyze the traffic with different tools.

- a) Start the virtual machines as depicted in figure 2 (pnidX-mid-hh server and pnidX-cnt-bln server) connected. Connect from the pnidX-mid-hh to the webserver running on the server pnidX-cnt-bln, if the httpd server on pnidX-cnt-bln is not available, you have to start the httpd server (search google to find out how to do this).

Follow several links on the server while watching packets flow (with Wireshark and tcpdump). Try other services as well (ftp, ssh, ...).

Note: Do not forget to install Wireshark and Telnetd Service on server pnidX-mid-hh and pnidX-cnt-bln.

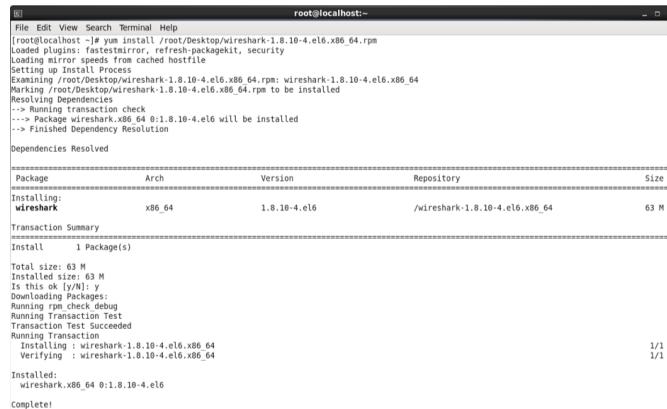
Connect to the webserver running on 10.88.X.?, (?) is an integer >0<254) from a virtual machine. Set filters so you will see only the traffic from your virtual machine to the webserver. Set another filter so you can see only http traffic. Save the TCP-handshake with 10.88.X.? in a file, where X is your group number and ? is your subnet ip address.

In den theoretischen Grundlagen wurde bereits der Ausdruck Sniffing erklärt. Dies wird in diesem Aufgabenteil nochmal aufgegriffen, da wir hier die Sniffing-Tools Wireshark und Tcpdump nutzen.

Wireshark ist eine freizugängliche Software, welches für die Netzwerküberwachung und grafische Aufbereitung von Sniffern dient. Dabei kann der Anwender sich die Inhalte und die Protokollheader der transportierten Datenpakete anzeigen lassen.

Auch das Tcpdump ist ein weltweit bekanntes Sniffing-Programm, welches Datenpakete mitschneiden kann. Tcpdump gibt es für die meisten Unix-Betriebssysteme. Doch Tcpdump stellt im Gegensatz zu Wireshark keine grafische Oberfläche für die Analyse bereit. Stattdessen besteht die Möglichkeit, den Netzwerkverkehr über Kommandozeilen mitzuschneiden und diese dann in Wireshark grafisch zu analysieren.

Um die Analyse der Netzwerkpakete zwischen den Kommunikationsmedien ausarbeiten zu können, haben wir zunächst Wireshark installiert. Zusätzlich haben wir Libsmi, eine C Bibliothek welche den Zugang zu Informationen von MIB Modulen bietet, installiert.



The screenshot shows a terminal window titled 'root@localhost:~'. The command entered is 'yum install /root/Desktop/wireshark-1.8.10-4.el6.x86_64.rpm'. The output of the command is displayed below:

```
[root@localhost ~]# yum install /root/Desktop/wireshark-1.8.10-4.el6.x86_64.rpm
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
Setting up Install Process
Examining /root/Desktop/wireshark-1.8.10-4.el6.x86_64.rpm: wireshark-1.8.10-4.el6.x86_64
Marking /root/Desktop/wireshark-1.8.10-4.el6.x86_64.rpm to be installed
Dependencies Resolved
=====
 Package           Arch      Version        Repository      Size
=====
 Installing:      wireshark   x86_64    1.8.10-4.el6   /wireshark-1.8.10-4.el6.x86_64  63 M
 Transaction Summary
=====
 Install 1 Package(s)
 Total size: 63 M
 Installed size: 63 M
 Is this ok [y/N]: y
 Downloading Packages:
 Running rpm check debug
 Running Transaction Test
 Transaction Test succeeded
 Running Transaction
   Installing : wireshark-1.8.10-4.el6.x86_64
   Verifying  : wireshark-1.8.10-4.el6.x86_64
 1/1
 Installed:
   wireshark.x86_64 0:1.8.10-4.el6
 Complete!
```

Abbildung 106: Abbildung installation wireshark

```
root@localhost:~  
[root@localhost ~]# rpm -Uvh libsmi-0.4.8-4.el6.x86_64.rpm  
Loading plugins: fastenmirror, refresh-packagekit, security  
Loading mirror speeds from cached hostfile  
Setting up Install Process  
Resolving Dependencies  
--> Package libsmi.x86_64 0:0.4.8-4.el6 will be installed  
Marking /root/Desktop/libsmi-0.4.8-4.el6.x86_64.rpm to be installed  
Resolving Dependencies  
--> Running transaction check  
--> Package libsmi.x86_64 0:0.4.8-4.el6 will be installed  
--> Finished Dependency Resolution  
  
Dependencies Resolved  
  
=====  
          Package           Arch      Version       Repository  
=====  
Installing:  libsmi           x86_64    0.4.8-4.el6   /libsmi-0.4.8-4.el6.x86_64  
=====  
  
Transaction Summary  
=====  
Install      1 Package(s)  
  
Total size: 16 M  
Installed size: 16 M  
Is this ok [y/N]: y  
Downloading Packages:  
  Running rpm_check_debug  
  Running Transaction Test  
  Transaction Test Succeeded  
  Running Transaction  
    Installing : libsmi-0.4.8-4.el6.x86_64  
    Verifying  : libsmi-0.4.8-4.el6.x86_64  
  
Installed:  
  libsmi.x86_64 0:0.4.8-4.el6  
  
[Complete]
```

Abbildung 107: Abbildung installation libsmi

Anschließend haben wir in exakt dieser Reihenfolge folgendes installiert:

Httpd - ein freinutzbarer Webserver im Internet von Apache Software Foundation

```

Transaction Summary
=====
Install 6 Packages(s)
Total download size: 1.1 M
Installed size: 3.6 M
(0 packages)
Downloaded Packages:
=====
Total 1.2 MB/s | 1.1 MB 00:00

Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
=====
Installing : apr-util-1.3.9-3.el6_2.x86_64
Installing : apr-util-1.3.9-3.el6_2.x86_64
Installing : apr-util-1.3.9-3.el6_2.x86_64
Installing : httpd-tools-2.2.15-29.el6.centos.x86_64
Installing : mod-ldap-1.31-2.el6.nourch
Installing : mod-ldap-1.31-2.el6.nourch.x86_64
Verifying : httpd-tools-2.2.15-29.el6.centos.x86_64
Verifying : mod-ldap-1.31-2.el6.nourch.x86_64
Verifying : mod-ldap-1.31-2.el6.nourch.x86_64
Verifying : apr-1.3.9-3.el6_2.x86_64
Verifying : apr-util-1.3.9-3.el6_2.x86_64
Verifying : mod-ldap-1.31-2.el6.nourch
Verifying : apr-util-1.3.9-3.el6_0.1.x86_64

Installed:
httpd.x86_64 0:2.2.15-29.el6.centos

Dependency Installed:
httpd-tools.x86_64 0:1.3.9-3.el6_0.1   apr-util-ldap.x86_64 0:1.3.9-3.el6_0.1   httpd-tools.x86_64 0:2.2.15-29.el6.centos

Complete!

```

Abbildung 108: Abbildung installation httpd

```
[root@localhost ~]# rpm -qf /usr/bin/ping install httpd
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
 * base           - http://mirror.centos.org/centos/6/os/x86_64/
 * epel            - http://dl.fedoraproject.org/pub/epel/6/x86_64/
Resolving Dependencies
  --> Processing Dependency: libcurl.so.4()(64bit) for package: httpd-2.2.15-29.el6.centos.x86_64
--> Processing Dependency: curl-libs-7.15.5-1.el6.x86_64 for package: httpd-2.2.15-29.el6.centos.x86_64
--> Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd-2.2.15-29.el6.centos.x86_64
--> Processing Dependency: libapr-1.so.0()(64bit) for package: httpd-2.2.15-29.el6.centos.x86_64
--> Processing Dependency: libxml2-2.7.8-2.el6.x86_64 for package: httpd-2.2.15-29.el6.centos.x86_64
--> Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd-2.2.15-29.el6.centos.x86_64
--> Processing Dependency: libapr-1.so.0()(64bit) for package: httpd-2.2.15-29.el6.centos.x86_64
--> Processing Dependency: libcurl.so.4()(64bit) will be installed
--> Package httpd-2.2.15-29.el6.centos.x86_64 will be installed
--> Package apr-0.8.10-10.el6.x86_64 will be installed
--> Package apr-util-1.5.2-1.el6.x86_64 will be installed
--> Package curl-7.15.5-1.el6.x86_64 will be installed
--> Package libxml2-2.7.8-2.el6.x86_64 will be installed
--> Package mailcap.nroff-2.0.1-3.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

Transaction Summary
  0 package updates
  0 packages total

Nothing to do or update.
[root@localhost ~]#
```

Abbildung 109: Abbildung installation httpd 2

Tcpdump - schneidet transportierte Datenpakete mit

```
[root@localhost ~]# yum install tcpdump
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
Setting up Install Process
Package 14:tcpdump-4.0.0-3.20090921gitf3cb4.2.el6.x86_64 already installed and latest version
Nothing to do
```

Abbildung 110: Abbildung installation tcpdump

Telnet - ein Netzwerkprotokoll für die Verbindung zu einem anderen Rechner über den Telnet-Server

```
[root@localhost ~]# yum install /root/Desktop/telnet-0.17-47.el6.3.i.x86_64.rpm
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
Setting up Install Process
Examining /root/Desktop/telnet-0.17-47.el6.3.i.x86_64.rpm: telnet-0.17-47.el6.3.i.x86_64
Marking /root/Desktop/telnet-0.17-47.el6.3.i.x86_64.rpm to be installed
Resolving Dependencies
--> Finished Dependency Resolution
Dependencies Resolved

Transaction Summary
=====================================================================
Install       1 Package(s)

Total size: 109 k
Installed size: 109 k
Is this ok [y/N]: y
Downloading Packages:
telnet-0.17-47.el6.3.i.x86_64.rpm: [download] http://192.168.1.100/centos/6.3/x86_64/optional/telnet-0.17-47.el6.3.i.x86_64.rpm
Running rpm check
Running transaction test
Transaction Test Succeeded
Running Transaction
  Installing : telnet-0.17-47.el6.3.i.x86_64
  Verifying   : telnet-0.17-47.el6.3.i.x86_64

Installed:
  telnet.x86_64 0:0.17-47.el6.3.i

Complete!

```

Abbildung 111: Abbildung installation telnet

```
[root@localhost yum.repos.d] yum install /home/user1/Desktop/wireshark-gnome-1.8.10-4.el6.x86_64.rpm
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
Setting up Install Process
Examining /home/user1/Desktop/wireshark-gnome-1.8.10-4.el6.x86_64.rpm: wireshark-gnome-1.8.10-4.el6.x86_64
Marking /home/user1/Desktop/wireshark-gnome-1.8.10-4.el6.x86_64.rpm to be installed
Resolving Dependencies
--> Running transaction check
---> Package wireshark-gnome.x86_64 0:1.8.10-4.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
| Package           | Arch   | Version        | Repository |
| ======            | ====== | ======         | ======      |
| Installing:      |        |                |             |
| wireshark-gnome | x86_64 | 1.8.10-4.el6  | /wireshark-gnome-1.8.10-4.el6.x86_64 | 2.5 M
|                   |        |                |             |
Transaction Summary

Install       1 Package(s)

Total size: 2.5 M
Installed size: 2.5 M
Is this ok [y/N]: y
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : wireshark-gnome-1.8.10-4.el6.x86_64
  Verifying   : wireshark-gnome-1.8.10-4.el6.x86_64

Installed:
  wireshark-gnome.x86_64 0:1.8.10-4.el6

[Complete!]
```

Abbildung 112: Abbildung installation wireshark

Nach den sämtlichen Installationen haben wir für den nächsten Schritt den Kommando `service httpd start` genutzt, somit sind die Dienste von Apache aktiviert worden.



```
root@localhost:~# service httpd start
Starting httpd: httpd: Could not reliably determine the server's fully qualified domain name, using localhost.localdomain for ServerName
[  OK  ]
[root@localhost ~]#
```

Abbildung 113: Abbildung Apache aktivieren

Infolgedessen ist uns gelungen eine Verbindung zum Apache http Server herzustellen, wie man auf dem folgenden Screenshot sehen kann. Um dieses aufzurufen, haben wir in der Adresszeile die IP-Adresse unseres lokalen Host eingegeben.

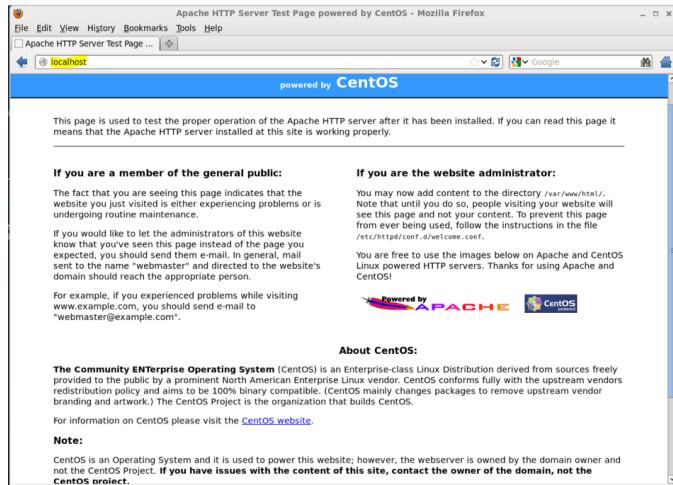


Abbildung 114: Verbindung zum http Server

Schließlich haben wir Wireshark gestartet und nach httpd gefiltert, sodass dann die Netzwerkpakete zum Server demonstriert wurden.

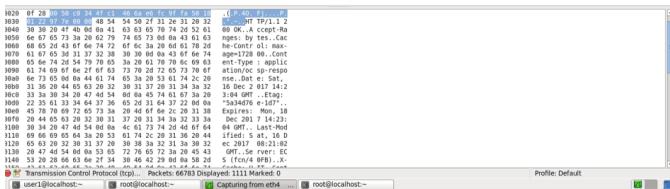
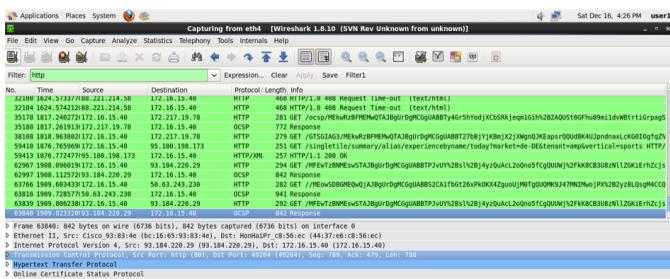


Abbildung 115: Wireshark Analyse

Wie in der Abbildung 116 zu erkennen, haben wir ebenfalls über Tcpdump den Netzverkehr im Terminal lassen.

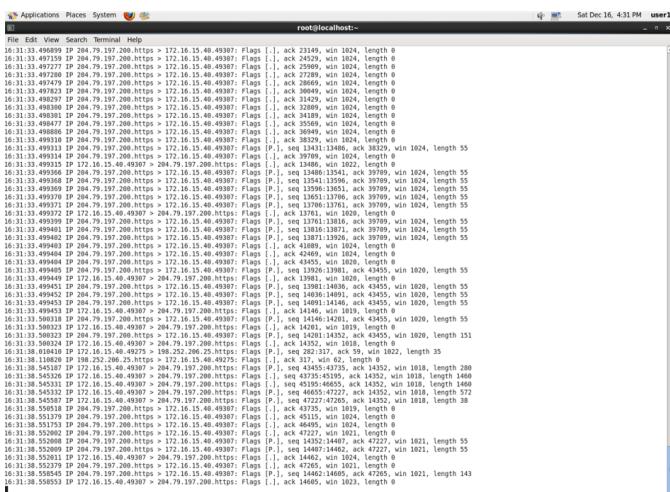


Abbildung 116: Tcpdump Analyse

Exercise 3: TCPDUMP

In this part of the exercise, you are going to explore tcpdump in more detail. In particular, you will answer more questions involving the use of tcpdump.

Q1: Please type and explain the syntax of a tcpdump command that captures packets containing IP datagram with source or destination IP address equal to 10.88.X. Δ where Δ is the IP address of pnidX-svr-hh(Centos7).

Der Kommando `tcpdump -i ens33` gibt an, dass die Netzwerkschnittstelle `ens33` analysiert werden soll. Dieses wird solange durchgeführt bis der Service nicht explizit gestoppt wird.

```
[root@localhost ~]# tcpdump -i ens33
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
12:29:19.442879 IP 10.88.100.130.mdns > 224.0.0.251.mdns: 0 PTR (QM)? _nmea-0183
    .tcp.local. (39)
12:29:19.614725 ARP, Request who-has 8.8.8.8 tell 10.88.60.161, length 46
12:29:19.822492 ARP, Request who-has 8.8.4.4 tell 10.88.100.161, length 46
12:29:20.619507 ARP, Request who-has 8.8.8.8 tell 10.88.60.161, length 46
12:29:20.825979 ARP, Request who-has 8.8.4.4 tell 10.88.100.161, length 46
```

Abbildung 117: tcpdump -i ens33

Q2: Please type and explain the syntax of a tcpdump command that captures 12 packets from the eth1 interface of your roudX1-st.

Der Kommando `tcpdump -c 12 -i ens33` analysiert ebenfalls die Netzwerkschnittstelle `ens33`, jedoch nicht wie beim Kommando von Q1 unendlich lang, sondern die Untersuchung wird nach dem 12 Packet automatisch abgebrochen. Dieses ist wegen dem `-c` praktikabel.

Abbildung 118: tcpdump -c 12 -i ens33

Q3: Please type and explain the syntax of a tcpdump command that captures packets containing ICMP messages with source or destination IP address equal to e.g. 10.88.X.?

Durch das eingeben von `tcpdump -i ens33 ip host 10.88.40.130 and icmp`"werden die Pakete, die ICMP Nachrichten enthalten, beim Netzwerkverkehr mit IP-Host 10.88.40.130 analysiert.

```
[root@localhost ~]# tcpdump -c 12 -i ens33 ip host 10.88.40.130 and icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
12:58:34.698663 IP 10.88.40.129 > localhost.localdomain: ICMP echo request, id 2967, seq 1, length 64
12:58:34.698737 IP localhost.localdomain > 10.88.40.129: ICMP echo reply, id 2967, seq 1, length 64
12:58:35.698243 IP 10.88.40.129 > localhost.localdomain: ICMP echo request, id 2967, seq 2, length 64
12:58:35.698295 IP localhost.localdomain > 10.88.40.129: ICMP echo reply, id 2967, seq 2, length 64
```

Abbildung 119: `tcpdump -c 12 -i ens33`

Q4: Please type and explain the syntax of a `tcpdump` command that captures packets containing IP datagram between `pnidX-mid-hh` and `pnidX-cnt-blن` with IP addresses `10.88.X.?` and `10.88.X.P.`

Hier wird die Netzwerkschnittstelle `eth4`, die Netzwerkpakete zwischen `pnid4-mid-hh` und `pnid4-cnt-blن` transportiert analysiert.

```
[root@localhost ~]# tcpdump -i eth4 host 10.88.40.70 and host 10.88.40.37
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth4, link-type EN10MB (Ethernet), capture size 65535 bytes
14:40:38.506192 IP 10.88.40.37 > 10.88.40.70: ICMP echo request, id 522, seq 1, length 64
14:40:38.507149 IP 10.88.40.37 > 10.88.40.70: ICMP echo request, id 522, seq 1, length 64
14:40:38.507154 IP 10.88.40.37 > 10.88.40.70: ICMP echo request, id 522, seq 1, length 64
14:40:38.507155 IP 10.88.40.37 > 10.88.40.70: ICMP echo request, id 522, seq 1, length 64
```

Q5: Please type and explain a `tcpdump` filter expression that captures packets containing TCP segments with source or destination IP address equal to `10.88.X.Δ`, where Δ is the ip address of `pnidX-svr-blن`.

Auch bei diesem Kommando ist es notwendig erst `tcpdump` zu tippen, sodass verstanden wird worum es sich handeln soll. Anschließend muss durch `-i` Name der Netzwerkschnitte wird verdeutlich um welche Netzwerkschnittstelle es sich handeln soll. Des Weiteren ist leicht zu erkennen, dass zwischen TCP Abschnitte und Ausgangs- bzw. die Zieladresse analysiert werden soll.

```
[root@localhost ~]# tcpdump -i ens33 tcp and src 10.88.40.67 or dst 10.88.40.67
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
16:02:40.980042 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 1, length 64
16:02:40.980226 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 1, length 64
16:02:40.980310 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 1, length 64
16:02:40.980566 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 1, length 64
16:02:40.980781 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 1, length 64
16:02:41.983588 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 2, length 64
16:02:41.983793 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 2, length 64
16:02:41.984099 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 2, length 64
16:02:41.985779 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 2, length 64
16:02:41.985952 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 2, length 64
16:02:42.988375 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 3, length 64
16:02:42.988559 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 3, length 64
16:02:42.988730 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 3, length 64
16:02:42.988944 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 3, length 64
16:02:42.989112 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 3, length 64
16:02:43.991586 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 4, length 64
16:02:43.991820 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 4, length 64
16:02:43.992169 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 4, length 64
16:02:43.992304 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 4, length 64
16:02:43.992547 IP 10.88.40.37 > localhost.localdomain: ICMP echo request, id 18442, seq 4, length 64
16:02:43.993596 ARP, Request who-has localhost.localdomain tell 10.88.40.65, length 46
16:02:45.998644 ARP, Reply 10.88.40.65 is-at 00:50:36:22:06:df (oui Unknown), length 46
```
22 packets captured
22 packets received by filter
0 packets dropped by kernel
```

Q6: Please type and explain a tcpdump filter expression that, in addition to the constraints in Q5, only captures packets using port number 23. What is port number 23, which file in Linux/Windows contains port 23 and which path?

Mit Hilfe des Kommando `tcpdump -i ens33 tcp and scr 10.88.50.67 or dst 10.88.40.67 and tcp and port 23` und `tcp and port 23` erreichen wir im Grunde dasselbe wie bei Q5, nur mit der Ausnahme das hier nur der Port 23 berücksichtigt wird. Port 23 steht für Telnet und in Linux findet man diesen unter diesem Pfad: `/etc/services/telnet 23/tcp`

```
[root@localhost ~]# tcpdump -i ens33 -l -w /tmp/test.pcap -n -s 65536 -e ether -A -s 1480 -B 1000 -f "tcp and dst net 10.88.40.67 and icmp"
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
[...]
tcpdump: packet 1 captured
1 packet captured
1 packet received by filter
0 packets dropped by kernel
```

Q7: Please type and explain a tcpdump command that collects all ICMP packets with destination IP address 10.88.X.Δ.

Der Kommando `tcpdump -l ens33 -v dst net 10.88.40.67 and icmp` fängt alle ICMP-Pakete die als Ziel die IP-Adresse 10.88.40.67 haben.

```
[root@localhost ~]# tcpdump -i ens33 -v dst net 10.88.40.67 and icmp
tcpdump: listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
16:20:44.598364 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 1, length 64
16:20:45.599269 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 2, length 64
16:20:46.603096 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 3, length 64
16:20:47.605680 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 4, length 64
16:20:48.606672 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 5, length 64
16:20:49.609703 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 6, length 64
16:20:50.610170 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 7, length 64
16:20:51.610613 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 8, length 64
16:20:52.612780 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 9, length 64
16:20:53.613114 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 10, length 64
16:20:54.615026 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.88.40.70 > localhost.localdomain: ICMP echo request, id 2828, seq 11, length 64
^C
11 packets captured
11 packets received by filter
0 packets dropped by kernel
```

Q8: Assume we are about to capture all telnet packets destined to IP address 10.88.X.H on port 23 (telnet port). Type and examine the following command and explain if it is correct or not: `tcpdump host 10.88.X.△ and dst port 23` Accordingly, if not correct, please provide the syntax.

Dieses Kommando ist richtig, wenn man alle Telnet-Pakete die 10.88.40.35 auf dem Port 23 als Ziel erreichen möchte.

```
[root@localhost ~]# tcpdump host 10.88.40.35 and dst port 23
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on virbr0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Q9: Please type and explain the syntax of a tcpdump command that collects all IP packets between host 10.88.X.△ and host 10.88.X.P on each interface.

Tcpdump -i any host 10.88.40.34 and host 10.88.40.38 and icmp bewirkt, dass jedes einzeln übertragenes Netzwerkpacet zwischen 10.88.40.34 und 10.88.40.38 auf beiden Interfaces aufgezeichnet wird.

```
[root@localhost ~]# tcpdump -i any host 10.88.40.34 and host 10.88.40.38 and icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
```

1. Start your virtual machine pnidX-mid-hh with IP-address 10.88.X.Δ
2. Login as root.
3. Create a non-privileged account with the name e.g. "trumpX".
4. Create a password for the above user. Make the password "clintonX".
5. Try to ping another machine on your network from pnidX-mid-hh to pnidX-cnt-bln.

Um die Frage 10 (Q10) beantworten zu können haben wir zunächst die virtuelle Maschine pnid4-mid-hh gestartet, mit dem Kommando su - und dann das zugehörige Passwort eingegeben. Anschließend haben wir ein Konto mit dem Namen "trump4" und Passwort als "clinton4" eingerichtet. Mit einem ping von pnid4-mid-hh nach pnid4-cnt-bln haben wir bestätigt, dass beide virtuellen Maschinen erreichbar sind und wir mit den folgenden Schritten fortfahren können.

6. Try to telnet from pnidX-mid-hh to the virtual machine pnidX-cnt-bln, you should be able to login as trumpX. If successful, just log out of pnidX-mid-hh.

Hier haben wir von der virtuellen Maschine pnid4-mid-hh die virtuelle Maschine pnid4-cnt-cln "getelnet". Dies war durchführbar mit dem Kommando telnet 10.88.40.37.

```
[root@localhost ~]# telnet 10.88.40.37
Trying 10.88.40.37...
Connected to 10.88.40.37.
Escape character is '^>'.
CentOS release 6.5 (Final)
Kernel 2.6.32-431.el6.x86_64 on an x86_64
login: trump4
Password:
Last login: Tue Dec 19 17:18:52 from 10.88.40.35
[trump4@localhost ~]$ █
```

Abbildung 120: Telnet mid-hh to cnt-bln

7. Now, as root on virtual machine pnidX-mid-hh, start tcpdump with the command tcpdump -vvv > pnidX-mid-hh-dump.1

Dieser Kommando gibt sehr viele Informationen. Je mehr V's desto mehr Information, wobei das Erste V für very steht, das Zweite V auch für very steht und das Dritte V für verbose.

```
[root@localhost ~]# tcpdump -i eth4 -vvv > pnid4-cnt-bln-dump.1
tcpdump: listening on eth4, link-type EN10MB (Ethernet), capture size 65535 bytes
5
^C601 packets captured
601 packets received by filter
0 packets dropped by kernel
```

Abbildung 121: tcpdump -vvv

8. On pnidX-mid-hh, open another shell and telnet from pnidX-mid-hh to pnidX-cnt-bln and login as user "trumpX". Logout of pnidX-cnt-bln.

Auch in diesem Schritt haben wir uns (pnid4-mid-hh) Mithilfe von telnet unter trump4 angemeldet. Hinterher haben wir uns vom pnid4-cnt-bln abgemeldet.

Abbildung 122: telnet trump4

9. Stop tcpdump, examine the file pnidX-mid-hh-dump.1 with vi.

Durch den Kommand vi pnid4-mid-hh-dump.1 haben wir den folgenden Inhalt anzeigen können:

Abbildung 123: pnidX-mid-hh-dump.1

10. Now, as root on pnidX-mid-hh, again start tcpdump with the command `tcpdump -a -i ethX` (X for your interface) > pnidX-mid-hh-dump.2

Hier haben wir den Kommando `tcpdump -a -l eth4 > pnid4-cnt-bln-dump.2` ausgeführt.

```
[root@localhost ~]# tcpdump -a -i eth4 > pnid4-cnt-bln-dump.2
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth4, link-type EN10MB (Ethernet), capture size 65535 bytes
^C3158 packets captured
3163 packets received by filter
5 packets dropped by kernel
```

11. Again, open another shell on pnidX-mid-hh and ssh from pnidX-mid-hh to pnidX-cnt-bln. After logging in to pnidX-cnt-bln just logout again.

Hier haben wir den Kommand ssh an 10.88.40.37 ausgeführt.

```
[root@localhost ~]# ssh 10.88.40.37
root@10.88.40.37's password:
Last login: Thu Dec 21 15:32:29 2017 from 10.88.40.37
[root@localhost ~]# logout
Connection to 10.88.40.37 closed.
```

Abbildung 124: ssh 10.88.40.37

## 12. Examine the pnidX-mid-hh-dump.2

Wie in diesem Schritt gefordert, haben wir die durch vi pnidX-mid-hh-dump.2 den Inhalt angezeigt

Abbildung 125: pnidX-mid-hh-dump.2

13. Finally, again as root on pnidX-mid-hh, again start the tcpdump with the command  
tcpdump -x > pnidX-mid-hh-dump.3

Hier haben wir durch den Kommando `tcpdump -l eth4 -x > pnid4-cnt-bln-dump.3` die Datenpakete der Netzwerkschnittstelle `eth4` mitgeschnitten.

```
[root@localhost ~]# tcpdump -i eth4 -x > pnid4-cnt-bln-dump.3
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth4, link-type EN10MB (Ethernet), capture size 65535 bytes
^C149 packets captured
149 packets received by filter
0 packets dropped by kernel
```

Abbildung 126: pnidX-mid-hh-dump.2

14. Again, open another shell on pnidX-mid-hh and from pnidX-mid-hh ftp to pnidX-cnt-bln. When you login to pnidX-cnt-bln, just logout.

In diesem Fall haben wir ftp 10.88.40.37 ausgeführt.

```
[root@localhost ~]# ftp 10.88.40.37
Connected to 10.88.40.37 (10.88.40.37).
220 (vsFTPd 2.2.2)
Name (10.88.40.37:root): trump4
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> exit
221 Goodbye.
```

Abbildung 127: pnidX-mid-hh-dump.2

### 15. Examine the pnidX-mid-hh-dump.3

```
File Edit View Search Terminal Help
18:38:05.327534 ARP, Request who-has 8.8.4.4 tell 10.88.60.161, length 46
 0x0000: 0001 0800 0604 0001 0050 5633 d46a 0a58
 0x0010: 3ca1 0000 0000 0000 0808 0404 0000 0000
 0x0020: 0000 0000 0000 0000 0000 0000 0000 0000
18:38:07.339129 ARP, Request who-has 172.16.15.249 (bc:16:65:93:83:4e (oui Unknown)) tell 172.16.15.40, length 46
 0x0000: 0001 0800 0604 0001 4437 e6c8 56ec ac10
 0x0010: 0ff9 bc16 6593 834e ac10 0ff9 0000 0000
 0x0020: 0000 0000 0000 0000 0000 0000 0000 0000
18:38:07.339651 ARP, Reply 172.16.15.249 is-at bc:16:65:93:83:4e (oui Unknown), length 46
 0x0000: 0001 0800 0604 0002 bc16 6593 834e ac10
 0x0010: 0ff9 4437 e6c8 56ec ac10 0f28 0000 0000
 0x0020: 0000 0000 0000 0000 0000 0000 0000 0000
18:38:08.566041 ARP, Request who-has 8.8.8.8 tell 10.88.60.161, length 46
 0x0000: 0001 0800 0604 0001 0050 5633 d46a 0a58
 0x0010: 3ca1 0000 0000 0000 0808 0808 0000 0000
 0x0020: 0000 0000 0000 0000 0000 0000 0000 0000
18:38:09.569376 ARP, Request who-has 8.8.8.8 tell 10.88.60.161, length 46
 0x0000: 0001 0800 0604 0001 0050 5633 d46a 0a58
 0x0010: 3ca1 0000 0000 0000 0808 0808 0000 0000
 0x0020: 0000 0000 0000 0000 0000 0000 0000 0000
@ "pnid4-cnt-bln-dump.3" 796L, 49212C
```

Abbildung 128: pnidX-mid-hh-dump.3

Q10: Now answer the following questions:

1. What is the difference in the three tcpdump commands? In their outputs?

Der Kommand `tcpdump -i eth4 -vvv > pnid4-cnt-bln-dump.1` gibt sehr viele Informationen aus. Je mehr `-v` Parameter im Kommando verwendet werden desto, mehr Details werden bereitgestellt.

Für `tcpdump -a -i eth4 > pnid4-cnt-bln.dump.2` hier werden alle Pakete von der Netzwerkschnittstelle angesammelt und als ASCII ausgegeben.

Bei Verwendung des Befehls `tcpdump -x > pnid4-cnt-bln.dump.3` werden die Pakete als Hexadezimalcode ausgegeben.

2. Is it possible to discover "trumplX"password with tcpdump? How would you do this?

Theoretisch kann man mit tcpdump ein Passwort "sniffen". Mit egrep kann man dies beispielweise durch folgenden Kommando Passwörter aufspüren:

```
tcpdump port http or port ftp or port smtp or port imap or port pop3
-l -A | egrep -i 'pass=/pwd=/log=/login=/user=/username=/pw=/passw=/passwd=/
password=/pass:/user:/username:/password:/login:/pass /user' --color=auto -line-buffered
-B20
```

3. How can you tell if a network interface is running in "promiscuous mode"?

Anhand dem Kommando `netstat -i` kann man sehen ob ein Interface auf "promiscuous mode" läuft. Falls ein P unter der Bezeichnung "Flg" bedeutet dies, dass für dieses Interface "promiscuous mode" geschaltet ist.

4. Can a normal, unprivileged user, run tcpdump?

Nur root kann tcpdump ausführen, da andere User keine Rechte dafür haben.

## Exercise 4: Wireshark

**Question 1:** Please type and examine the syntax for a Wireshark command which capture filter so that all IP datagrams with source or destination IP address equal to 10.88.X.? are recorded.

**Answer 1:** Mit dem Filter: `ip.addr == 10.88.40.70` können wir alle Netzwerkpakete, welche über die Schnittstelle 10.88.40.70 gesendet oder empfangen werden, abfangen und anzeigen lassen.

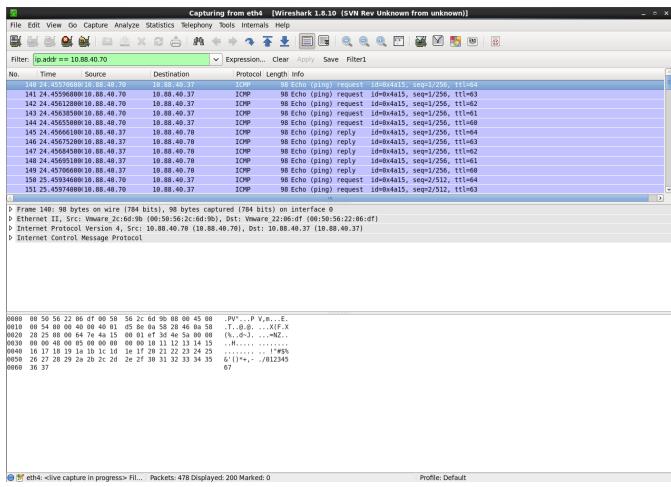


Abbildung 129: Wireshark Filter `ip.addr == 10.88.40.70`

**Question 2:** Please type and examine the syntax for a Wireshark display filter that shows IP datagrams with destination IP address equal to 10.88.X.? and frame size greater than 400 bytes.

**Answer 2:** Um alle Datenpakete abzufangen, die mindestens 400 Byte groß sind, bedarf eine kleine Erweiterung des vorherigen Befehls. Der Filter lautet nun: `ip.addr == 10.88.40.70 && frame.len > 400`. Mit dem Teil `frame.len > X` können wir die Datenpakete nach Bytegröße X Filtern. Für X gilt,  $X < 2^{32} \wedge X \in \mathbb{N}$ .

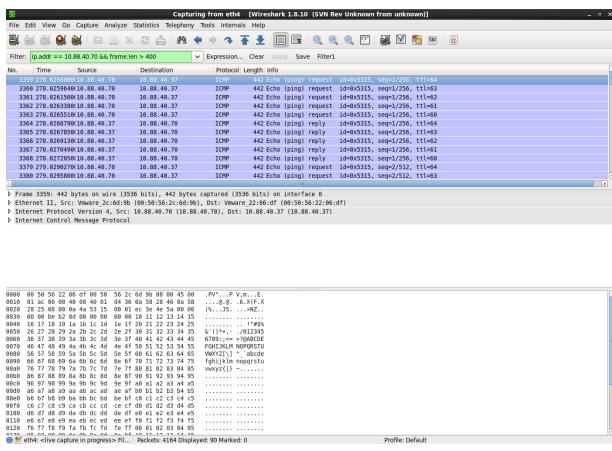


Abbildung 130: Wireshark Filter  $ip.addr == 10.88.40.70 \&\& frame.len > 400$

**Question 3:** Please type and examine the syntax for a Wireshark display filter that shows packets containing ICMP messages with source or destination IP address equal to 10.88.X.? and frame numbers between 15 and 30

**Answer 3:** Der Filter lautet:  $ip.addr == 10.88.40.70 \&\& (frame.number > 15 \&\& frame.number < 30)$ . ICMP steht für Internet Control Message Protocol und übermittelt hauptsächlich Diagnose-informationen zwischen dem Router und dem Host.



Abbildung 131: Wireshark Filter  $ip.addr == 10.88.40.70 \&\& (frame.number > 15 \&\& frame.number < 30)$

**Question 4:** Please type and examine the syntax for a Wireshark display filter that shows packets containing TCP segments with source or destination IP address equal to 10.88.X.? and using port number 23.

**Answer 4:** Damit wir alle TCP Pakete eines Hosts über die Port 23 abfangen können wird der folgende Filter eingesetzt:  $ip.dst == 10.88.40.70 \text{ and } tcp.port == 23$ . Bei

TCP handelt es sich um ein Übertragungsprotokoll (Transmission Control Protocol) aus der Familie der Internetprotokolle. Port 23 ist standardisiert für den Service Telnet.

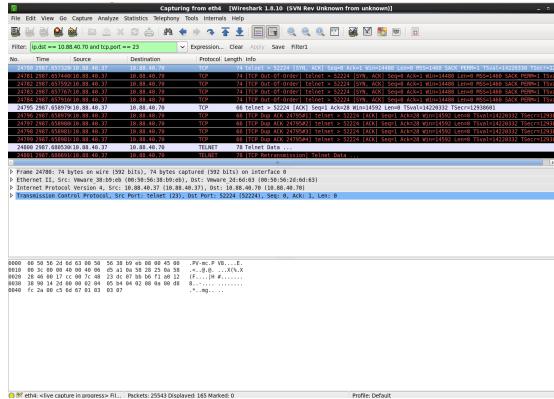


Abbildung 132: Wireshark Filter ip.dst == 10.88.40.70 and tcp.port == 23

**Question 5:** Please type and examine a Wireshark capture filter expression for Q4.

**Answer 5:** Der Filter ist ähnlich wie in Q4, lediglich die Konfiguration findet an einer anderen Stelle statt.

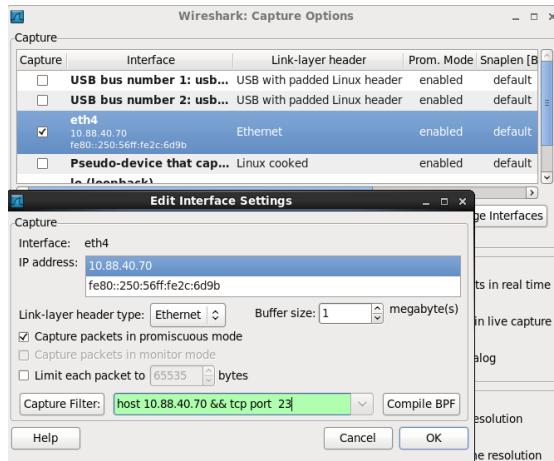


Abbildung 133: Wireshark Filter host 10.88.40.70 && tcp port 23

**Question 6:** Please type and examine the syntax for a Wireshark command which, by default, collects packets with source or destination IP address 10.88.X.? on interface eth4.

**Answer 6:** Innerhalb des Terminals lässt sich der Filter: `wireshark -i eth4 -k -f "host 10.88.40.70"`, anwenden. Die Argumente bedeuten dabei folgendes: -i eth4 steht für Interface, -k startet das Abfangen von Paketen und -f "host 10.88.40.70", ist der Paketfilter.

**Question 7:** Please type and examine the syntax of a display filter which selects the TCP packets with destination IP address 10.88.X.?, and TCP port number 23.

**Answer 7:** Der Filter lautet: `ip.addr == 10.88.40.70 && tcp.port == 23` und fängt alle ein-/ausgehenden Pakete der Ip Adresse 10.88.40.70 über den Port 23 (Telnet) ab.

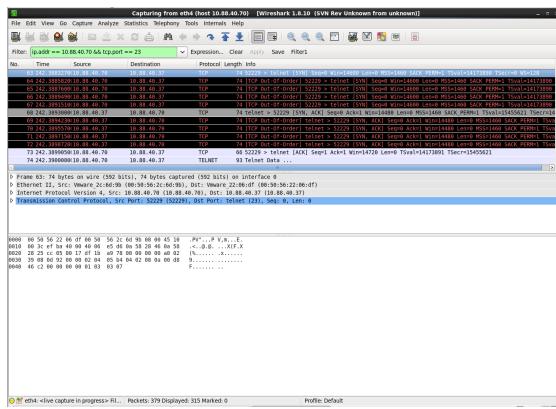


Abbildung 134: Wireshark Filter `ip.addr == 10.88.40.70 && tcp.port == 23`

**Question 8:** Please login to the server pnidX-mid-hh and start an ftp client to the server pnidXcnt-bln(vsftpd daemon should be running on pnidX-cnt-bln). Please use wireshark on pnidX-mid-bln to sniff or capture the username and password of the ftp service between pnidX-mid-hh and pnidX-cnt-bln. Is this possible, show your result of the capture

**Answer 8:** Mithilfe von Wireshark können wir leicht das ftp login Passwort herausfinden, da bei der Übertragung via ftp die Pakete unverschlüsselt übertragen werden. Dazu starten wir zunächst Wirehsark auf dem Host pnid4-mid-hh und führen ein ftp login, von cnt-bln nach mid-hh, durch. Zuerst muss der Service ftp auf beiden Host aktiv sein, deshalb überprüfen wir den Status.

```
[root@localhost ~]# service vsftpd status
vsftpd (pid 1768) is running...
[root@localhost ~]#
```

Danach starten wir wireshark auf dem Host mid-hh und melden uns über den Host cnt-bln bei dem Host mid-hh über den ftp servie an.

```
[root@localhost ~]# ftp 10.88.40.37
Connected to 10.88.40.37 (10.88.40.37).
220 (vsFTPD 2.2.2)
Name (10.88.40.37:root): trump4
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> █
```

Über die Ausgabe Login successfull sehen wir, dass das anmelden erfolgreich war. Wir öffnen nun Wireshark auf dem Host mid-hh und filtern nach ftp Paketen. Dazu reicht es aus ftp in die Filtermaske einzugeben.

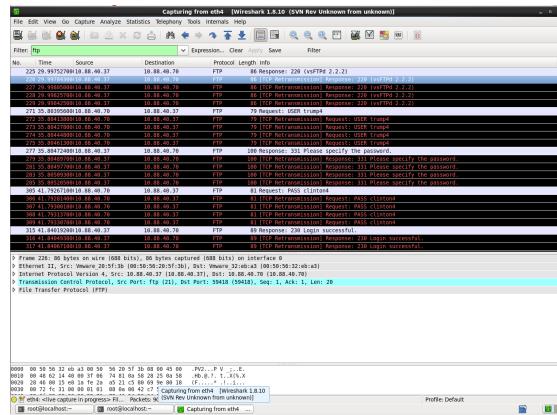


Abbildung 135: Wireshark Filter ftp

Wir schauen uns nun die Pakete genauer an und können die Logininformationen in einem der Pakete anzeigen lassen.

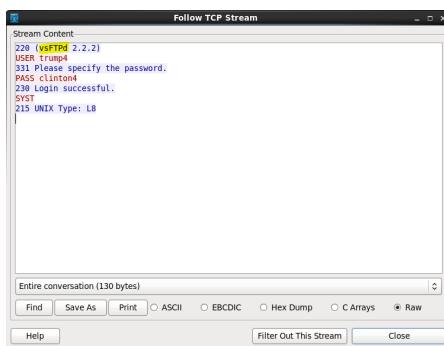


Abbildung 136: Wireshark ftp Login Passwort

Sofort sehen wir den Benutzernamen trump4 und das Passwort clinton4. Dieses Szenario zeigt wie einfach es ist die Logininformationen herauszulesen, wenn die Datenpakete unverschlüsselt übertragen werden.

**Question 9:** Please login to the server pnidX-mid-hh and start an ssh client to the server pnidX-cnt-bln(sshd daemon should be running on pnidX-cnt-bln). Please use wireshark on pnidX-mid-bln to sniff or capture the username and password of the ssh service between pnidX-mid-hh and pnidX-cnt-bln. Is this possible, show the result of the capture.

**Answer 9:** Anders als ftp werden bei ssh (Secure Shell) die Pakete verschlüsselt übertragen, sodass es nicht möglich ist das Passwort mitzulesen. Zuerst prüfen wir, ob der ssh service auf beiden Hosts aktiv ist.

```
[root@localhost ~]# service sshd status
openSSH-daemon (pid 1749) is running...
[root@localhost ~]#
```

Danach starten wir wireshark auf dem Host mid-hh und melden uns über den Host cnt-bln bei dem Host mid-hh über den ssh servie an.

```
[root@localhost ~]# ssh 10.88.40.37
root@10.88.40.37's password:
Last login: Thu Jan 4 13:55:43 2018 from 10.88.40.70
[root@localhost ~]#
```

Über die Ausgabe Last login..., sehen wir, dass das anmelden erfolgreich war. Wir öffnen nun Wireshark auf dem Host mid-hh und filtern nach ssh Paketen. Dazu reicht es aus ssh in die Filtermaske einzugeben.

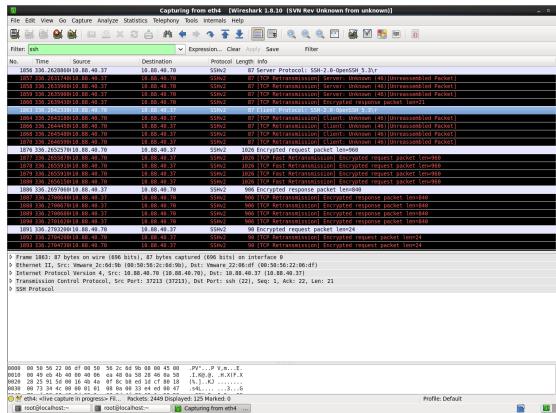


Abbildung 137: Wireshark Filter ssh

Wir schauen uns nun die Pakete genauer an und können keine Informationen über das Login erhalten, da alle Datenfragmente verschlüsselt wurden.

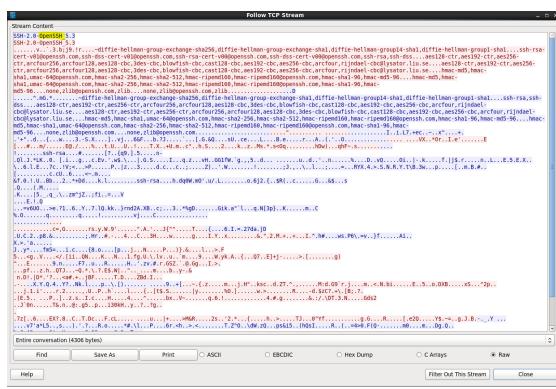


Abbildung 138: Wireshark verschlüsselte Datenfragmente

## Exercise 5: Experimenting with network monitoring tools

**Exercise:** In this exercise you will connect to the webserver pnidX-cnt-bln from pnidX-mid-hh. Let pnidX-cnt-bln determine your IP-address and the OS you are running. Then, connect to a service of your choice (e.g. ftp, http, ssh etc.) on pnidX cnt-bln. Let pnidX-mid-hh determine which services are running on pnidX-cnt-bln.

**Solution:** Wir führen zunächst nmap auf dem Host pnid4-cnt-bln aus und übergeben dabei die Zieladresse des Hosts pnid4-mid-hh, damit wir sehen können welche Ports geöffnet sind bzw. welcher Service auf dem Zielhost gerade aktiv ist.

```
[root@localhost ~]# nmap -sF -O 10.88.40.37
Starting Nmap 3.51 (http://nmap.org) at 2018-01-04 14:33 CET
nmap: dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify v
alid servers with --dns-servers
Nmap scan report for 10.88.40.37
Host is up (0.0025s latency).

Not shown: 995 closed ports
PORT STATE SERVICE
21/tcp open|filtered ftp
22/tcp open|filtered ssh
23/tcp open|filtered telnet
80/tcp open|filtered http
111/tcp open|filtered rpcbind
113/tcp open|filtered ncftpd
TIP: Fingerprint this host to give specific OS details
Network Distance: 4 hops
OS detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 3.02 seconds
[root@localhost ~]$
```

Abbildung 139: Zeigt uns die offenen Ports an

Wir sehen nun, dass die Ports 21 ftp, 22 ssh, 23 telnet, 80 http und 111 rpcbind offen sind und entscheiden uns via Telnet vom Quellhost pnid4-cnt-bln bei dem Zielhost pnid-mid-hh anzumelden.

```
[root@localhost ~]# telnet 10.88.40.37
Trying 10.88.40.37...
Connected to 10.88.40.37.
Escape character is '^>'.
CentOS release 6.5 (Final)
Kernel 2.6.32-431.el6.x86_64 on an x86_64
login: trump4
Password:
Last login: Thu Jan 4 14:04:27 from 10.88.40.70
[trump4@localhost ~]$
```

Abbildung 140: Anmeldung via Telnet

Die Ausgabe des letzten Logins zeigt uns, dass die Anmeldung erfolgreich war.

## Exercise 6: Set up a host-to-host VPN using preshared key

### VPN: Host-to-host

Bei einem Host-to-Host-VPN greift ein Client auf einen anderen Clienten in einem entfernten Netzwerk zu. Dabei wird ein VPN-Tunnel aufgebaut, der die zwei Hosts miteinander verbindet. Auf beiden Seiten muss eine entsprechende VPN-Software installiert und konfiguriert sein. Der Verbindungsaufbau geschieht im Allgemeinen nur durch die Unterstützung einer zwischengeschalteten Station. Das bedeutet, eine direkter Verbindungsaufbau von Host zu Host ist nicht möglich. Daher bauen beide Seiten eine Verbindung zu einem Router auf, dass die beiden Verbindungen dann zusammenbringt.

Da man die Daten sicher über das Internet übertragen möchte, versucht man mittels eines Tunneling-Protokolls eine verschlüsselte Verbindung, den VPN-Tunnel aufzubauen. Wenn der Tunnel aufgebaut ist, ist der Inhalt der Daten für andere nicht sichtbar. Einzelne Clients bindet man in der Regel per Tunnelmodus an.

### Preshared-Key:

Ein **symmetrisches** Verschlüsselungsverfahren, bei dem die Schlüssel schon zu Beginn der Kommunikation beiden Partnern bekannt ist. Vorher müssen diese Schlüssel jedoch im Geheimen ausgetauscht werden. Für viele Anwendungen im Internet ist dieses Verfahren ungeeignet, da hierfür ein großer Aufwand notwendig wäre.

To create a host-to-host VPN as shown in Figure 1 using preshared keys both systems must have OPENS/WAN properly installed and tested. Next, you must ensure that IP networking is functioning. For this exercise you will capture http packets between the hosts. This capture will allow you to compare and prove that IPSec is functioning after the tunnel is created between the hosts. Make sure Apache and Wireshark are installed. See Figure 1.

Hints: CREATING PRESHARED KEY use ipsec ranbits 256 > filename

## Installation von Openswan auf beiden Rechnern Hamburg (pnid4-mid-hh) und Berlin (pnid4-cnt-bln)

```
[root@localhost ~]# mount -t iso9660 /dev/sr0 /dvdrom
mount: block device /dev/sr0 is write-protected, mounting read-only
[root@localhost ~]# yum -y install openswan
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
LocalRepo | 4.0 kB 00:00 ...
Setting up Install Process
Resolving Dependencies
--> Running transaction check
-->> Package openswan.x86_64 0:2.6.32-27.el6 will be installed
-->> Finished Dependency Resolution

Dependencies Resolved

=====
Package Arch Version Repository Size
=====
Installing:
 openswan x86_64 2.6.32-27.el6 LocalRepo 895 k

Transaction Summary
=====
Install 1 Package(s)

Total download size: 895 k
Installed size: 2.6 M
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
 Installing : openswan-2.6.32-27.el6.x86_64 1/1
 Verifying : openswan-2.6.32-27.el6.x86_64 1/1

Installed:
 openswan.x86_64 0:2.6.32-27.el6

Complete!
[root@localhost ~]# █
```

Abbildung 141: Installation Openswan

## Netzwerk für die Konfiguration der VPN-Verbindung

# VPN

## HOST TO HOST VPN

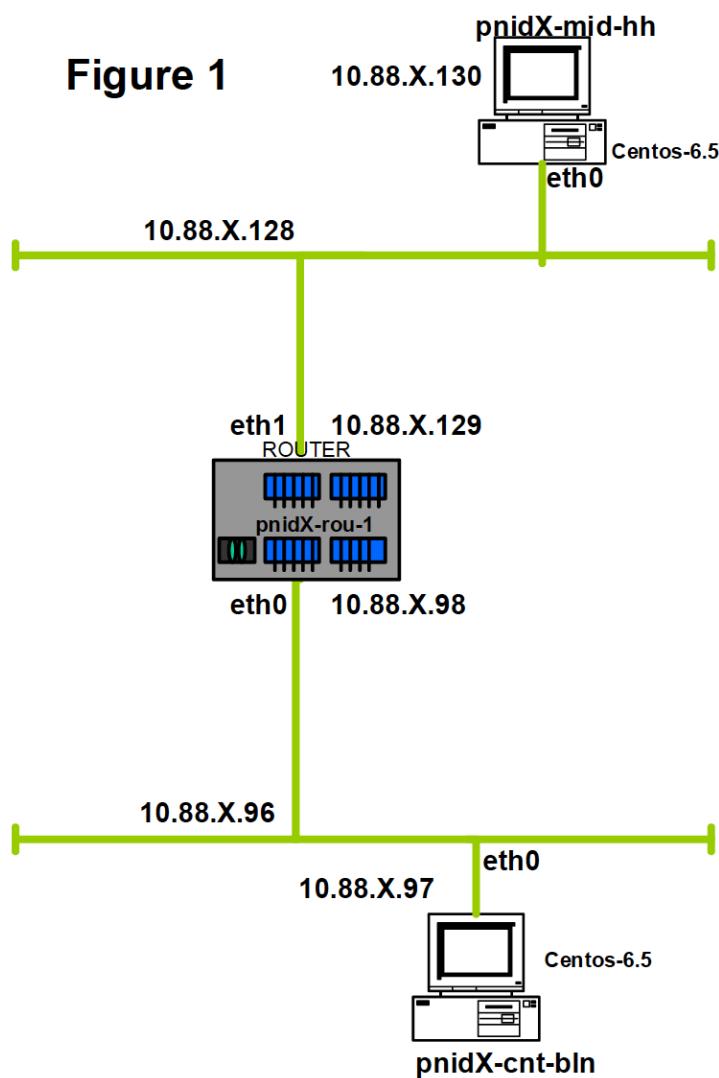


Abbildung 142: Figure 1: HOST-TO-HOST-VPN

i) Capture http packets between pnidX-mid-hh and pnidX-cnt-bln using Wireshark before establishing a tunnel and save the packet captured. Please include this with your lab report.

Vor der Einrichtung von Openswan können HTTP-Pakete von beliebigen Angreifern ohne Probleme abgehört werden. VPN soll dafür sorgen, dass dies verhindert wird. Zunächst starten wir auf pnid4-cnt-bln Wireshark, um HTTP-Pakete abzufangen. Um HTTP-Pakete von pnid4-mid-hh abzufangen, geben wir den Filter "dst 10.88.40.130 && tcp && port 80" ein (siehe Abbildung ..).

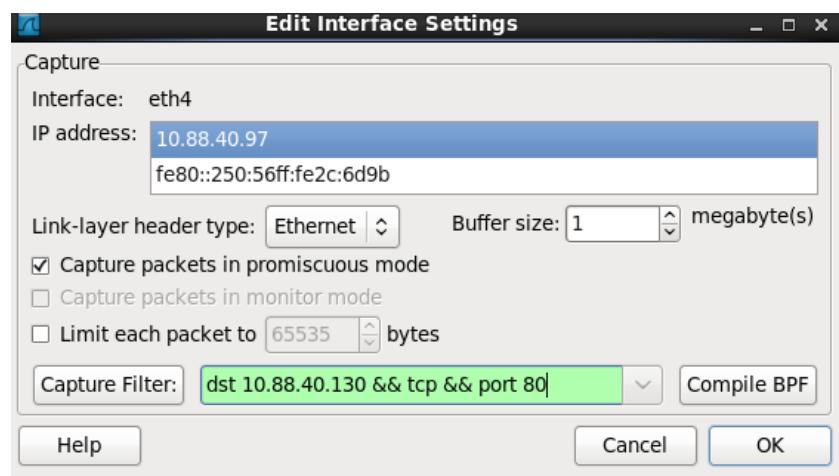


Abbildung 143: dst 10.88.40.130 && tcp && port 80

Auf dem Rechner pnid4-cnt-bln wird im Browser die IP-Adresse von pnid4-mid-hh eingeben, um HTTP-Pakete abzufangen (siehe Abbildung 139)

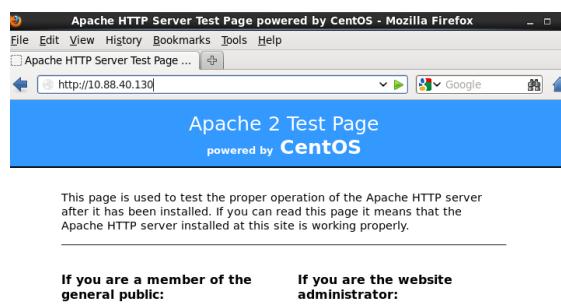


Abbildung 144: http://10.88.40.130

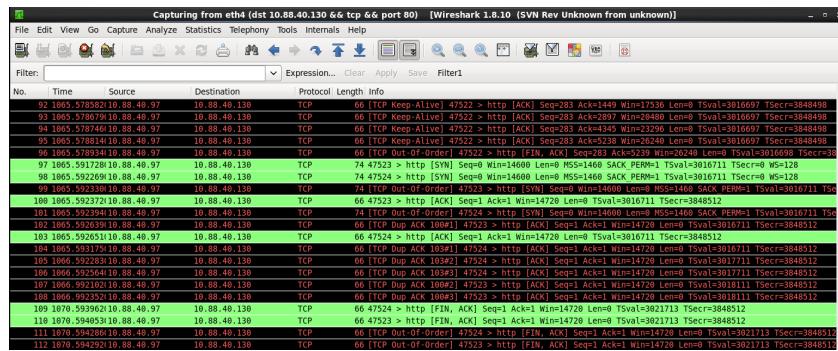


Abbildung 145: Filter: dst 10.88.40.130 && tcp && port 80

- ii) Establish a tunneled IPSec connection, using a 256 preshared secret. Confirm that the connection is established.

Capture again http packets between pnidX-mid-hh and pnidX-cnt-bln using Wireshark after establishing a tunnel and save the packet captured. Please include this with your lab report.

Mit Hilfe des Befehls `# ipsec ranbits 256 > psk.secrets` wird ein zufällig erzeugter Key generiert, der in der Datei `psk.secrets` gespeichert wird.

```
[root@pnid4-mid-hh ipsec.d]# ipsec ranbits 256 > psk.secrets
```

Abbildung 146: ipsec ranbits 256 >

Die Datei wird so angepasst, dass Sie folgende Struktur bekommt:

% IP-Adresse-mid-hh % IP-Adresse-cnt-bln : PSK "der\_Key\_welches\_über\_ipsec\_ranbits\_generiert\_wurde "(siehe Abb. ..)

```
[root@pnid4-mid-hh ipsec.d]# cat psk.secrets
10.88.40.97 10.88.40.130 : PSK _0x08ba6fd0_212c2a15_fa671f53_16fc4b87_dc231639_479d3714_b3f46bb5_58dc7fc2
```

Abbildung 147: psk.secrets

Als nächstes wird die psk.conf Datei erstellt, welches sich ebenfalls im Verzeichnis ipsec.d befindet. Hier erfolgt die Konfiguration des Tunnels. Dabei wird die Authentifizierung über authby angegeben. Mit type wird der Typ der Verbindung (tunnel, transport) bezeichnet. Die betroffenen Kommunikationspartner werden mit left und right dargestellt. Left bezeichnet dabei den Host, auf den wir uns befinden. Right hingegen bezeichnet den dazugehörigen Kommunikationspartner. Left und right entsprechen dabei die IP-Adressen der jeweiligen Kommunikationspartner. Auto bezeichnet die Operation, welche beim starten von IPsec ausgeführt werden soll.

```
[root@pnid4-mid-hh ipsec.d]# vi psk.conf
[root@pnid4-mid-hh ipsec.d]# cat psk.conf
conn psk
 type=tunnel
 auto=add
 authby=secret

 left=10.88.40.97
 #leftsubnet=10.88.40.96/27

 right=10.88.40.130
 #rightsubnet=10.88.40.128/27
[root@pnid4-mid-hh ipsec.d]#
```

Abbildung 148: psk.conf

Damit alle Dateien innerhalb des Verzeichnisses /etc/ipsec.d mit der Endung .conf eingebunden werden können, muss die letzte Zeile (siehe Abb. ) auskommentiert werden.

```
[root@pnid4-mid-hh etc]# cat ipsec.conf
/etc/ipsec.conf - Openswan IPsec configuration file
#
Manual: ipsec.conf.5
#
Please place your own config files in /etc/ipsec.d/ ending in .conf
version 2.0 # conforms to second version of ipsec.conf specification

basic configuration
config setup
 # Debug-logging controls: "none" for (almost) none, "all" for lots.
 # klipsdebug=none
 # plutodebug="control parsing"
 # For Red Hat Enterprise Linux and Fedora, leave protostack=netkey
 protostack=netkey
 nat_traversal=yes
 virtual_private=
 oe=off
 # Enable this if you see "failed to find any available worker"
 # nhelpers=0

#You may put your configuration (.conf) file in the "/etc/ipsec.d/" and uncomment this.
include /etc/ipsec.d/*.conf
```

Damit beide Kommunikationspartner dieselben Dateien haben, wird die Datei psk.secrets und die Datei psk.conf mit dem Befehl scp zum jeweils anderen Kommunikationspartner geschickt. Scp(Secure Copy) ist ein Protokoll und sorgt für eine verschlüsselte Übertragung von Daten zwischen zwei Computern.

```
[root@pnid4-mid-hh ipsec.d]# scp /etc/ipsec.d/psk.secrets root@10.88.40.97:/etc/ipsec.d/psk.secrets
root@10.88.40.97's password: 100% 105 0.1KB/s 00:00
psk.secrets
[root@pnid4-mid-hh ipsec.d]# scp /etc/ipsec.d/psk.conf root@10.88.40.97:/etc/ipsec.d/psk.conf
root@10.88.40.97's password: 100% 145 0.1KB/s 00:00
psk.conf
[root@pnid4-mid-hh ipsec.d]#
```

Auf beiden Rechnern wird ipsec gestartet:

```
[root@pnid4-mid-hh /]# ipsec setup reload
ipsec_setup: Stopping Openswan IPsec...
ipsec_setup: Starting Openswan IPsec U2.6.32/K2.6.32-431.el6.x86_64...
ipsec_setup: /usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fi
ps_enabled
```

Abbildung 149: ipsec setup reload

Mit dem Befehl ipsec auto --add wird eine neue Verbindung aus der Konfigurationsdatei in die Pluto-Datenbank eingelesen.

```
[root@pnid4-mid-hh /]# ipsec auto --add psk
/usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips_enabled
[root@pnid4-mid-hh /]# ipsec setup status
IPsec running - pluto pid: 3092
pluto pid 3092
No tunnels up
```

Abbildung 150: ipsec auto –add

Mit dem Befehl ipsec auto --up psk wird die Verbindung gestartet. Dabei versucht Pluto in der internen Datenbank der geladenen Konfiguration die benötigte Verbindung aufzubauen.

```
[root@pnid4-mid-hh /]# ipsec auto --up psk
104 "psk" #1: STATE_MAIN_I1: initiate
003 "psk" #1: received Vendor ID payload [Openswan (this version) 2.6.32]
003 "psk" #1: received Vendor ID payload [Dead Peer Detection]
003 "psk" #1: received Vendor ID payload [RFC 3947] method set to=109
106 "psk" #1: STATE_MAIN_I2: sent MI2, expecting MR2
003 "psk" #1: NAT-Traversal: Result using RFC 3947 (NAT-Traversal): no NAT detected
108 "psk" #1: STATE_MAIN_I3: sent MI3, expecting MR3
003 "psk" #1: received Vendor ID payload [CAN-IKEv2]
004 "psk" #1: STATE_MAIN_I4: ISAKMP SA established {auth=OAKLEY_PRESHARED_KEY cipher=aes_128 prf=oakley sha group=modp2048}
117 "psk" #2: STATE_QUICK_I1: initiate
004 "psk" #2: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel mode {ESP=>0 x8d14b6d0 <0x724dfaaf xfrm=AES_128-HMAC_SHA1 NATOA=none NATD=none DPD=none}
```

Abbildung 151: ipsec auto –up

Pluto ist der IKE-Daemon, der das IKE-Protokoll implementiert. Pluto erstellt automatische Sicherheitsbeziehungen, die untereinander geteilt werden. Für die Authentifikation verwendet Pluto Shared Secrets oder RSA Signaturen.

```
[root@pnid4-mid-hh /]# ipsec setup status
IPsec running - pluto pid: 3092
pluto pid 3092
1 tunnels up
some eroutes exist
```

- iii) Secure the computer traffic by creating iptables rules to permit only IPSec traffic between the two computers. All other, non-IPSec packets must be denied.

```
[root@pnid4-mid-hh etc]# iptables -F
[root@pnid4-mid-hh etc]# iptables -A FORWARD -p esp -s 10.88.40.130 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh etc]# iptables -A FORWARD -p esp -s 10.88.40.97 -d 10.88.40.130 -j ACCEPT
[root@pnid4-mid-hh etc]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.130 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh etc]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.97 -d 10.88.40.130 -j ACCEPT
[root@pnid4-mid-hh etc]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.130 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh etc]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.97 -d 10.88.40.130 -j ACCEPT
[root@pnid4-mid-hh etc]# iptables -A FORWARD -j REJECT
```

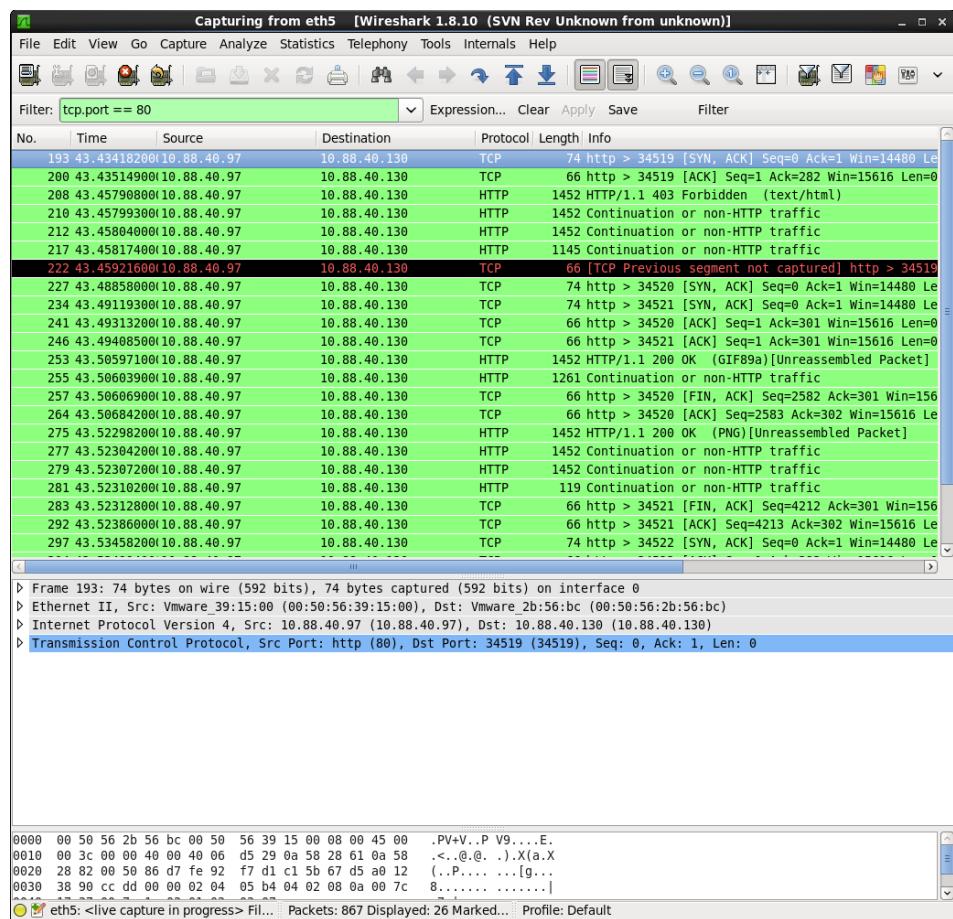
Abbildung 152: iptables rules

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_iptables_config
Generated by iptables-save v1.4.7 on Thu Jan 18 17:32:12 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Thu Jan 18 17:32:12 2018
```

```
[root@pnid4-mid-hh ipsec.d]# iptables -L
Chain INPUT (policy ACCEPT)
target prot opt source destination
REJECT all -- anywhere anywhere reject-with icmp-port-unreachable

Chain FORWARD (policy ACCEPT)
target prot opt source destination
ACCEPT esp -- 10.88.40.130 10.88.40.97
ACCEPT esp -- 10.88.40.97 10.88.40.130
ACCEPT udp -- 10.88.40.130 10.88.40.97 udp dpt:isakmp
ACCEPT udp -- 10.88.40.97 10.88.40.130 udp dpt:isakmp
ACCEPT udp -- 10.88.40.130 10.88.40.97 udp dpt:ipsec-nat-t
ACCEPT udp -- 10.88.40.97 10.88.40.130 udp dpt:ipsec-nat-t
REJECT all -- anywhere anywhere reject-with icmp-port-unreachable

Chain OUTPUT (policy ACCEPT)
target prot opt source destination
[root@pnid4-mid-hh ipsec.d]#
```



iv) After a successful IPSec connection has been established, create the following files using the given commands:

```
File ipsec_ifconfig.dump
ifconfig > ipsec_fconfig.dump
File ipsec_look.dump
ipsec look > ipsec_look.dump
File ipsec_route.dump
Route -n > ipsec_route.dump
```

The following files must be included with the lab report:

- 1) /etc/ipsec.secrets
- 2) /etc/ipsec.conf
- 3) /etc/ipsec\_iptables - a file which contains your iptables rules
- 4) ipsec\_ifconfig.dump
- 5) ipsec\_look.dump
- 6) ipsec\_route.dump

#### Files from Berlin:

```
[root@localhost ipsec.d]# ifconfig > ipsec_fconfig.dump
[root@localhost ipsec.d]# ipsec look > ipsec_look.dump
[root@localhost ipsec.d]# route -n > ipsec_route.dump
[root@localhost ipsec.d]# █
```

#### 1) /etc/ipsec.secrets

```
[root@localhost ipsec.d]# cat /etc/ipsec.d/psk.secrets
10.88.40.97 10.88.40.130 : PSK 0x08ba6fd0_212c2a15_fa671f53_16fc4b87_dc231639_47
9d3714_b3f46bb5_58dc7fc2
```

Abbildung 153: /etc/ipsec.secrets

## 2) /etc/ipsec.conf

```
[root@localhost ipsec.d]# cat /etc/ipsec.d/psk.conf
conn psk
 type=tunnel
 auto=add
 authby=secret

 left=10.88.40.97
 #leftsubnet=10.88.40.96/27

 right=10.88.40.130
 #rightsubnet=10.88.40.128/27
```

Abbildung 154: /etc/ipsec.conf

## 3) /etc/ipsec\_iptables - a file which contains your iptables rules

```
[root@localhost ipsec.d]# cat /etc/ipsec.d/ipsec_iptables_config
Generated by iptables-save v1.4.7 on Thu Jan 18 17:32:12 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Thu Jan 18 17:32:12 2018
```

Abbildung 155: /etc/ipsec\_iptables

#### 4) ipsec\_ifconfig.dump

```
[root@localhost ipsec.d]# cat /etc/ipsec.d/ipsec_fconfig.dump
eth5 Link encap:Ethernet HWaddr 00:50:56:23:6C:C4
 inet addr:10.88.40.97 Bcast:10.88.40.127 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe23:6cc4/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:10969 errors:0 dropped:0 overruns:0 frame:0
 TX packets:4282 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:770189 (752.1 KiB) TX bytes:302051 (294.9 KiB)

lo Link encap:Local Loopback
 inet addr:127.0.0.1 Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING MTU:16436 Metric:1
 RX packets:4226 errors:0 dropped:0 overruns:0 frame:0
 TX packets:4226 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:186016 (181.6 KiB) TX bytes:186016 (181.6 KiB)
```

Abbildung 156: ipsec\_ifconfig.dump

#### 5) ipsec\_look.dump

```
[root@localhost ipsec.d]# cat /etc/ipsec.d/ipsec_look.dump
localhost.localdomain Thu Jan 18 18:26:56 CET 2018
IPSEC TABLE
ROUTING TABLE
10.88.40.96/27 dev eth5 proto kernel scope link src 10.88.40.97 metric 1
10.88.40.128/27 via 10.88.40.98 dev eth5 proto static
default via 10.88.40.98 dev eth5 proto static
```

Abbildung 157: ipsec\_look.dump

#### 6) ipsec\_route.dump

```
[root@localhost ipsec.d]# cat /etc/ipsec.d/ipsec_route.dump
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
10.88.40.96 0.0.0.0 255.255.255.224 U 1 0 0 eth5
10.88.40.128 10.88.40.98 255.255.255.224 UG 0 0 0 eth5
0.0.0.0 10.88.40.98 0.0.0.0 UG 0 0 0 eth5
```

Abbildung 158: ipsec\_route.dump

**Files from Hamburg:**

```
[root@pnid4-mid-hh ipsec.d]# ifconfig > ipsec_fconfig.dump
[root@pnid4-mid-hh ipsec.d]# ipsec look > ipsec_look.dump
[root@pnid4-mid-hh ipsec.d]# route -n > ipsec_route.dump
```

**1) /etc/ipsec.secrets**

```
[root@pnid4-mid-hh ipsec.d]# cat /etc/ipsec.d/psk.secrets
10.88.40.97 10.88.40.130 : PSK 0x08ba6fd0_212c2a15_fa671f53_16fc4b87_dc231639_479d3714_b3f46bb5_58dc7fc2
[root@pnid4-mid-hh ipsec.d]#
```

Abbildung 159: /etc/ipsec.secrets

**2) /etc/ipsec.conf**

```
[root@pnid4-mid-hh ipsec.d]# cat /etc/ipsec.d/psk.conf
conn psk
 type=tunnel
 auto=add
 authby=secret

 left=10.88.40.97
 #leftsubnet=10.88.40.96/27

 right=10.88.40.130
 #rightsubnet=10.88.40.128/27
[root@pnid4-mid-hh ipsec.d]#
```

Abbildung 160: /etc/ipsec.conf

### 3) /etc/ipsec\_iptables - a file which contains your iptables rules

```
[root@pnid4-mid-hh ipsec.d]# cat /etc/ipsec.d/ipsec_iptables_config
Generated by iptables-save v1.4.7 on Thu Jan 18 17:32:12 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Thu Jan 18 17:32:12 2018
```

Abbildung 161: /etc/ipsec\_iptables

### 4) ipsec\_ifconfig.dump

```
[root@pnid4-mid-hh ipsec.d]# cat /etc/ipsec.d/ipsec_fconfig.dump
eth5 Link encap:Ethernet HWaddr 00:50:56:2B:56:BC
 inet addr:10.88.40.130 Bcast:10.88.40.159 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe2b:56bc/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:12882 errors:0 dropped:0 overruns:0 frame:0
 TX packets:294 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:1207838 (1.1 MiB) TX bytes:57575 (56.2 KiB)

lo Link encap:Local Loopback
 inet addr:127.0.0.1 Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING MTU:16436 Metric:1
 RX packets:2533 errors:0 dropped:0 overruns:0 frame:0
 TX packets:2533 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:128404 (125.3 KiB) TX bytes:128404 (125.3 KiB)
```

Abbildung 162: ipsec\_ifconfig.dump

## 5) ipsec\_look.dump

```
[root@pnid4-mid-hh ipsec.d]# cat /etc/ipsec.d/ipsec_look.dump
pnid4-mid-hh.localdomain Thu Jan 18 18:09:01 CET 2018
IPSEC TABLE
ROUTING TABLE
10.88.40.96/27 via 10.88.40.129 dev eth5 proto static
10.88.40.128/27 dev eth5 proto kernel scope link src 10.88.40.130 metric 1
default via 10.88.40.129 dev eth5 proto static
[root@pnid4-mid-hh ipsec.d]# █
```

Abbildung 163: ipsec\_look.dump

## 6) ipsec\_route.dump

```
[root@pnid4-mid-hh ipsec.d]# cat /etc/ipsec.d/ipsec_route.dump
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
10.88.40.96 10.88.40.129 255.255.255.224 UG 0 0 0 eth5
10.88.40.128 0.0.0.0 255.255.255.224 U 1 0 0 eth5
0.0.0.0 10.88.40.129 0.0.0.0 UG 0 0 0 eth5
[root@pnid4-mid-hh ipsec.d]# █
```

Abbildung 164: ipsec\_route.dump

## Exercise 7: Set up a host-to-host VPN using RSA keys

**RSA ist ein asymmetrisches Verschlüsselungsverfahren.** Diese verwendet man zum Verschlüsseln und Signieren. Es besteht aus einem Schlüsselpaar, welches sich zusammensetzt aus einem öffentlichen und privaten Schlüssel. Mit dem **öffentlichen Schlüssel verschlüsselt** man die Nachricht oder prüft Signaturen. Mit dem **privaten Schlüssel** hingegen **entschlüsselt** man die Nachricht.

Hints: CREATING RSA PRIVATE KEYS SECRETS do the following:

- certutil -N -d /etc/ipsec.d
  - ipsec newhostkey --configdir /etc/ipsec.d/ --output /etc/ipsec.d/keys.secrets
- To create a host-to-host VPN as shown in figure 1, using RSA keys both systems must

have OPENS/WAN properly installed and tested. Next, you must ensure that IP networking is functioning. For this exercise you will capture http packets between the hosts. This capture will allow you to compare and prove that IPSec is functioning after the tunnel is created between the hosts. Make sure Apache and Wireshark are installed.

i) Capture http packets between pnidX-mid-hh and pnidX-cnt-bln using Wireshark before establishing a tunnel and save the packet captured. Please include this with your lab report

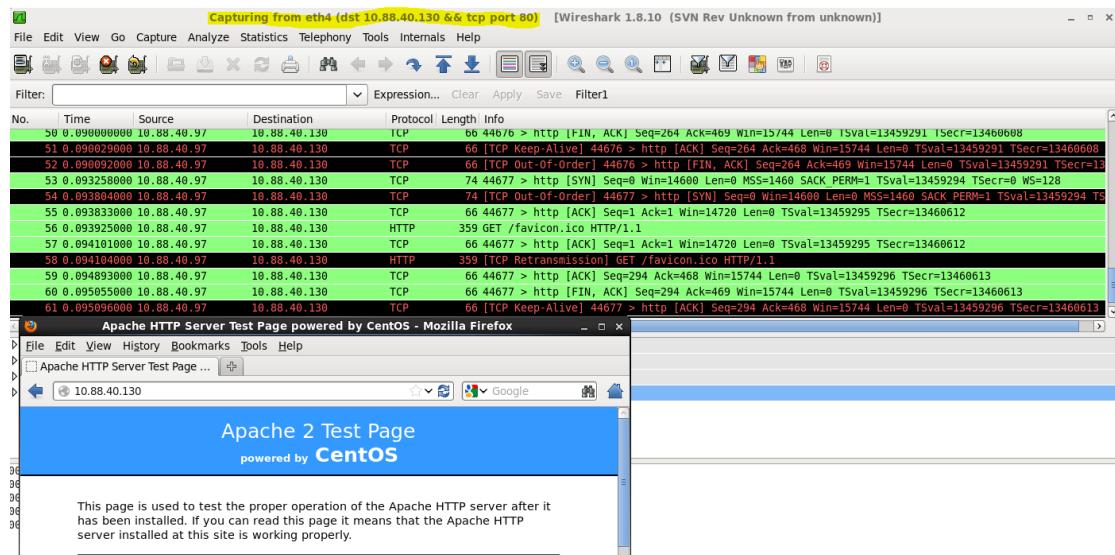


Abbildung 165: dst 10.88.40.130 && tcp port 80

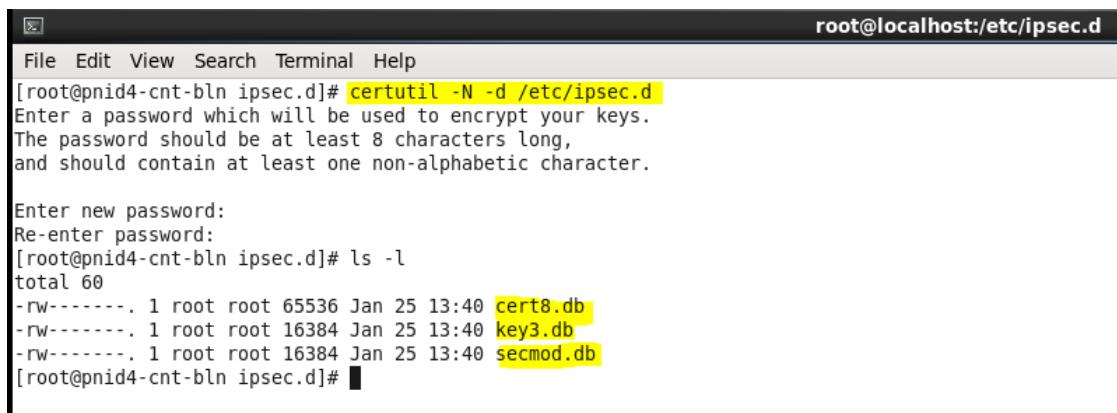
ii) Establish a tunneled IPSec connection, using RSA Method between the two computers.

Confirm that the connection is established.

Capture again http packets between pnidX-mid-hh and pnidX-cnt-bln using Wireshark after establishing a tunnel and save the packet captured. Please include this with your lab report

## Privaten RSA-Schlüssel generieren

a) certutil -N -d /etc/ipsec.d

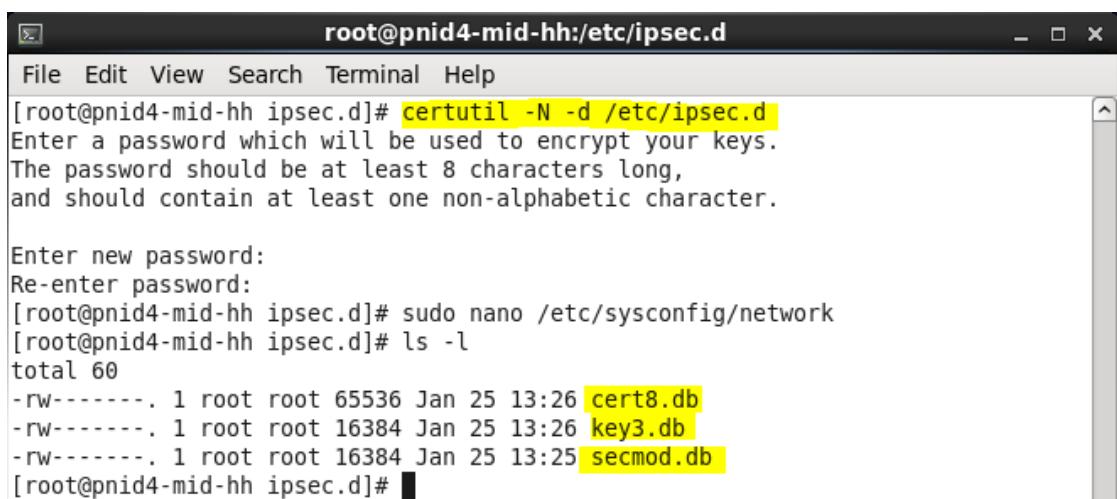


A terminal window titled "root@localhost:/etc/ipsec.d". The command "certutil -N -d /etc/ipsec.d" is entered, followed by prompts for a password. The output shows three database files: cert8.db, key3.db, and secmod.db.

```
File Edit View Search Terminal Help
root@localhost:/etc/ipsec.d#
[root@pnid4-cnt-bln ipsec.d]# certutil -N -d /etc/ipsec.d
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
[root@pnid4-cnt-bln ipsec.d]# ls -l
total 60
-rw-----. 1 root root 65536 Jan 25 13:40 cert8.db
-rw-----. 1 root root 16384 Jan 25 13:40 key3.db
-rw-----. 1 root root 16384 Jan 25 13:40 secmod.db
[root@pnid4-cnt-bln ipsec.d]#
```

Abbildung 166: certutil -N -d /etc/ipsec.d



A terminal window titled "root@pnid4-mid-hh:/etc/ipsec.d". The command "certutil -N -d /etc/ipsec.d" is entered, followed by prompts for a password. The output shows three database files: cert8.db, key3.db, and secmod.db.

```
File Edit View Search Terminal Help
root@pnid4-mid-hh:/etc/ipsec.d#
[root@pnid4-mid-hh ipsec.d]# certutil -N -d /etc/ipsec.d
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
[root@pnid4-mid-hh ipsec.d]# sudo nano /etc/sysconfig/network
[root@pnid4-mid-hh ipsec.d]# ls -l
total 60
-rw-----. 1 root root 65536 Jan 25 13:26 cert8.db
-rw-----. 1 root root 16384 Jan 25 13:26 key3.db
-rw-----. 1 root root 16384 Jan 25 13:25 secmod.db
[root@pnid4-mid-hh ipsec.d]#
```

Abbildung 167: certutil -N -d /etc/ipsec.d

b) ipsec newhostkey --configdir /etc/ipsec.d/ --output /etc/ipsec.d/keys.secrets  
**Der Befehl generiert 2 RSA Authentifikationsschlüssel für Hamburg und Berlin.**

```
[root@pnid4-cnt-bln ipsec.d]# ipsec newhostkey --configdir /etc/ipsec.d/ --output /etc/ipsec.d/rsa.secrets
Generated RSA key pair using the NSS database
[root@pnid4-cnt-bln ipsec.d]# ls -l
total 68
-rw----- 1 root root 65536 Jan 25 13:40 cert8.db
-rw----- 1 root root 16384 Jan 25 13:46 key3.db
-rw----- 1 root root 1578 Jan 25 13:46 rsa.secrets
-rw----- 1 root root 16384 Jan 25 13:40 secmod.db
[root@pnid4-cnt-bln ipsec.d]#
```

Abbildung 168: ipsec newhostkey 1

```
[root@pnid4-mid-hh ipsec.d]# ipsec newhostkey --configdir /etc/ipsec.d/ --output /etc/ipsec.d/rsa.secrets
Generated RSA key pair using the NSS database
[root@pnid4-mid-hh ipsec.d]#
[root@pnid4-mid-hh ipsec.d]# ls -l
total 68
-rw----- 1 root root 65536 Jan 25 13:26 cert8.db
-rw----- 1 root root 16384 Jan 25 13:45 key3.db
-rw----- 1 root root 1589 Jan 25 13:45 rsa.secrets
-rw----- 1 root root 16384 Jan 25 13:25 secmod.db
[root@pnid4-mid-hh ipsec.d]#
```

Abbildung 169: ipsec newhostkey 2

Die **rsa.conf** -Datei wird auf beiden Rechnern wie folgt angepasst.

```
[root@pnid4-cnt-bln ipsec.d]# cat rsa.conf
conn rsa
 type=tunnel
 auto=add
 authby=rsasig

 #SRC HOST pnid4-mid-hh
 left=10.88.40.130
 leftsubnet=10.88.40.128/27
 leftsaorigkey=0$APzUpfje9/kcBsRcv5+1xk021L9dHbp42NxlHoAdj3dkuFs25hD5XmdD8AURZYwHUWiydGltdw0AwkIVgIgaCCYYphTRBB9EAzDMhyVZFrLT+GqdK8yh+6933RlVW7R2jnmJylUI+qJZUCKdv
MrU6T1Fc6C3tWk6SXWpJ2o1cdmZFL1OR3FPrvEMjTC13LrNof81BtqG9GJFrB/z/y6lUu0g/B5GDHO/Tm983xrN8p0teaQY+0mVAvNpkEzXbKtHAWEBqvLkppd4C94r38os8wXZMY3GnSCdI0M1kpI1wi5wvzkv0vfgYsVQ
+/jpy6v7zmhk0i0FVKSLMNCIigVfqwceIrEZlzo56mWFL

 #0$T pnid4-cnt-bln
 right=10.88.40.97
 rightsubnet=10.88.40.96/27
 rightsaorigkey=0$A00q1w/TV/BUYe7TH4ZwMjpLr7e7dY6pdTUvR08w2bg2lZeTzhZ+yCMlbeb9lY6Nm1i0wIdUNQc01Fu0bpLYnsdscl/S9y200T5oPDarpNpID1J07X0ANm+MZJJ0000Siis5F/SxeTANI6V64s2
ZWoBTX4szortmj4FLG5cIAyqL77aeC0+tjKRf0r8yKzy5ZE7tquSueB2+upNKjM3XL/ZycfnU1VnxHsxzj+gY03lbe6ph0tly9qN0ZIAuHxD0Nd18N5oLH2Tjaz2Myxuo0aJiqYy67PNV50+7tebdrERa3kZ/WCGjn2nBXJb0
uVT+eaBL0VnPjRTIpF7nxAAZ26t5y647UQ04USRR1hb07Y/ah
[root@pnid4-cnt-bln ipsec.d]#
```

Abbildung 170: rsa.conf

Auf beiden Rechnern wird ipsec gestartet:

```
[root@pnid4-cnt-bln ipsec.d]# ipsec setup reload
ipsec_setup: Stopping Openswan IPsec...
ipsec_setup: Starting Openswan IPsec U2.6.32/K2.6.32-431.el6.x86_64...
ipsec_setup: /usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips_enabled
```

Abbildung 171: ipsec setup reload

```
[root@pnid4-mid-hh ipsec.d]# ipsec setup reload
ipsec_setup: Stopping Openswan IPsec...
ipsec_setup: Starting Openswan IPsec U2.6.32/K2.6.32-431.el6.x86_64...
ipsec_setup: /usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips_enabled
```

Abbildung 172: ipsec setup reload

Die Verbindung wird in die Datenbank eingelesen:

```
[root@pnid4-mid-hh ipsec.d]# ipsec auto --add rsa
/usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips_enabled
[root@pnid4-mid-hh ipsec.d]# ipsec setup status
IPsec running - pluto pid: 3029
pluto pid 3029
No tunnels up
```

Abbildung 173: ipsec auto -- add rsa

Verbindung wird hergestellt und ein Tunnel aufgebaut.

```
[root@pnid4-mid-hh ipsec.d]# ipsec auto --up rsa
104 "rsa" #1: STATE MAIN I1: initiate
003 "rsa" #1: received Vendor ID payload [Openswan (this version) 2.6.32]
003 "rsa" #1: received Vendor ID payload [Dead Peer Detection]
003 "rsa" #1: received Vendor ID payload [RFC 3947] method set to=109
106 "rsa" #1: STATE MAIN I2: sent MI2, expecting MR2
003 "rsa" #1: NAT-Traversal: Result using RFC 3947 (NAT-Traversal): no NAT detected
108 "rsa" #1: STATE MAIN I3: sent MI3, expecting MR3
003 "rsa" #1: received Vendor ID payload [CAN-IKEv2]
004 "rsa" #1: STATE_MAIN I4: ISAKMP SA established {auth=OAKLEY_RSA_SIG cipher=aes_128 prf=oakley_sha group=modp2048}
117 "rsa" #2: STATE QUICK I1: initiate
004 "rsa" #2: STATE QUICK I2: sent QI2, IPsec SA established tunnel mode {ESP=>0x6aa2531e <0xb1b5ee78 xfrm=AES_128-HMAC_SHA1 NATOA=none NATD=none DPD=none}
[root@pnid4-mid-hh ipsec.d]# ipsec setup status
IPsec running - pluto pid: 3029
1 tunnel's up
some routes exist
```

Abbildung 174: ipsec auto -- up rsa

- iii) Secure the computer traffic by creating iptables rules to permit only IPSec traffic between the two computers. All other, non-IPSec packets must be denied.

### **iptables Rules für Hamburg:**

```
[root@pnid4-mid-hh ipsec.d]# iptables -F
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.130 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.97 -d 10.88.40.130 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.130 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.97 -d 10.88.40.130 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.130 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.97 -d 10.88.40.130 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -j REJECT
```

```
[root@pnid4-mid-hh ipsec.d]# iptables-save > /etc/ipsec.d/ipsec_iptables
```

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_iptables
Generated by iptables-save v1.4.7 on Thu Jan 25 15:26:03 2018
*filter
:INPUT ACCEPT [1:328]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Thu Jan 25 15:26:03 2018
```

### **iptables Rules für Berlin:**

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_iptables
Generated by iptables-save v1.4.7 on Thu Jan 25 15:47:22 2018
*filter
:INPUT ACCEPT [7:2088]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [1:120]
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Thu Jan 25 15:47:22 2018
```

iv) After a successful IPSec connection has been established, create the following files using the given commands:

```
File ipsec_ifconfig.dump
ifconfig > ipsec_fconfig.dump
File ipsec_look.dump
ipsec look > ipsec_look.dump
File ipsec_route.dump
Route -n > ipsec\route.dump
```

The following files must be included with the lab report:

- 1) /etc/ipsec.secrets
- 2) /etc/ipsec.conf
- 4) ipsec\_ifconfig.dump
- 4) ipsec\_look.dump
- 5) ipsec\_route.dump

### Files from Berlin:

#### 1) /etc/ipsec.secrets

```
[dhcp-172-21-25-244:~ hanife]$ cat /Users/hanife/Documents/GitHub/Projekt/bilder/dateien/aufgabe\ 7/bln/rsa.secrets
: RSA {
 # RSA 2192 bits pnid4-cnt-blن Thu Jan 25 13:46:40 2018
 # for signatures only, UNSAFE FOR ENCRYPTION
 #pubkey=0sAQ0o1+TV/BUYe7TH4ZWMjpLR7e7dy6pdTUvRQ8w2bgZlZeTzhZ+yCmLbeb91Y6Nm1i0wIdUNQcQ1Fu0bpLYnsdscU/S9y2QOT
5oPDarpNpIDIJ07XANm+MZZJD008SIis5f/SxeTANI6V64s2ZWoBTx4szortmj4f1G5cIAyql77aECQ+tjKrfqr8yKzy5ZE7tquSueB2+upNKjM3XL
/ZYcfnuVnxHsxzj+gYo31beGph0Ty19qNOZ1AuHxDNd18NS5oLH2Tjaz2MYxucoQaJiqYY67PNVS8+7tobDrERa3kZ/WCGjn2nBXJb0uVT+eaBL6Vh
PjRtIpF7nxAZ20of5sy047UQ04USRRIhb07V/ah
 Modulus: 0xa897efd357f05461eed31f8656323a4b47b7bb758ea975352f450f30d9b819959793ce167ec9c3256de6fd958e8d9b588
ec08754350710d45b46e92d89e0c76c714fd2f72d90393e683c36aba4da480c82+ed7d00366f8c64924338ed12222b3917f4b179300d23a57ae
2cd99556a014d7e2cce8aed9a3e05946e5c20062a2fbeda10243eb632917d0afcc8acf2e5913bb6ab92b9e076faea4d2a33375cbfd961c7e7522
567c47b31ce3fa0628de56de1a98744f297da8d399200b87c4e0cd775f0d4b9a0b1f64e36b3d8c631b9ca106898aa615ebb3cd552d3eeda1b0e
b1116b7919fd60868e7da785725b3ae553f9e6812fa5613e346d22917b9f1019d86a1fe72d38ed440ee1449144885ba3b63f6a1
 PublicExponent: 0x03
 # everything after this point is CKA_ID in hex format when using NSS
 PrivateExponent: 0x458b0c2d53874e9c4e5f7752fea38c14c7bf3406
 Prime1: 0x458b0c2d53874e9c4e5f7752fea38c14c7bf3406
 Prime2: 0x458b0c2d53874e9c4e5f7752fea38c14c7bf3406
 Exponent1: 0x458b0c2d53874e9c4e5f7752fea38c14c7bf3406
 Exponent2: 0x458b0c2d53874e9c4e5f7752fea38c14c7bf3406
 Coefficient: 0x458b0c2d53874e9c4e5f7752fea38c14c7bf3406
 CKAI0NSS: 0x458b0c2d53874e9c4e5f7752fea38c14c7bf3406
}
do not change the indenting of that "}"
```

Abbildung 175: /etc/ipsec.secrets

## 2) /etc/ipsec.conf

```
[root@pnid4-cnt-bln ipsec.d]# cat rsa.conf
conn rsa
 type=tunnel
 auto=add
 authby=rsasig
 #SRC HOST pnid4-mid-hh
 left=10.88.40.130
 leftsubnet=10.88.40.128/27
 leftrsasigkey=0sAQZupFjze9/kcBsRcvS+1xk02i2L9dHbp42NxIHoAdj3dkuFs25hD5Xmd08AURZYwHUWIdyGltdw0AwkIVvIgaCCYYpHTRBB9EAzDMhyVZFrLT+G0dK8yh+6933RlVwR2jnm3yUUI+qJZUCKdv
MrU6T1Fc0C931wK6BXMrPjz0lcdmcZFL10R3FPrveMjTC13LrNof81BtqG9GJFrB/z/y6lUu0g/BSDOHQ/Tm983xrN8p0teaqY+0mVAvNpkEzXbKtHAWEbqvLkppd4C94r38os8wXZMY3GnScdI0MIKpI1wi5wvzkv0vfgYsvQ
+jpy6v7ZmlhkeIoFVK5sLMNCICigVFqWceIrEZlzo56mWWFL
 #DST pnid4-cnt-bln
 right=10.88.40.97
 rightsubnet=10.88.40.96/27
 rightrsasigkey=0sA00oL+/TVUYe7TH4ZwMjplR7e7dY6pdTUvR08w2bg2lZeTzhZ+ycMlbeb9lY6Nm1i0wIdUN0c01Fu0bpLynsdscU/S9y200T5oP0arpNpID1J07X0Anm+MZJD0008Siis5F/SxeTANI6V64s2
ZvWoBTX4szortmj4FLG5cIAYql77aeC0+tjKRf0r8yKzy5ZE7tquSueB2+uphKjM3XL/ZYcfnU1VnxHsxzj+gYo3lbeGph0Ty19qNOZIAuHxDNd18N55oLH2Tjaz2MyxucoQajqYv67PNV50+7tobDrERa3KZ/WC6jn2nBXJb0
uVT+eaBL6VnPjRTIpF7nxA2ZGof5y04UQ04USRRlhbo7Y/ah
[root@pnid4-cnt-bln ipsec.d]#
```

Abbildung 176: /etc/ipsec.conf

## 3) /etc/ipsec\_iptables - a file which contains your iptables rules

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_iptables
Generated by iptables-save v1.4.7 on Thu Jan 25 15:47:22 2018
*filter
:INPUT ACCEPT [7:2088]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [1:120]
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.49/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Thu Jan 25 15:47:22 2018
```

Abbildung 177: /etc/ipsec\_iptables

#### 4) ipsec\_ifconfig.dump

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_ifconfig.dump
eth5 Link encap:Ethernet HWaddr 00:50:56:23:6C:C4
 inet addr:10.88.40.97 Bcast:10.88.40.127 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe23:6cc4/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:14613 errors:0 dropped:0 overruns:0 frame:0
 TX packets:109 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:1111480 (1.0 MiB) TX bytes:18880 (18.4 KiB)

lo Link encap:Local Loopback
 inet addr:127.0.0.1 Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING MTU:16436 Metric:1
 RX packets:138 errors:0 dropped:0 overruns:0 frame:0
 TX packets:138 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:11116 (10.8 KiB) TX bytes:11116 (10.8 KiB)
```

Abbildung 178: ipsec\_ifconfig.dump

#### 5) ipsec\_look.dump

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_look.dump
pnid4-cnt-bln Thu Jan 25 15:53:02 CET 2018
IPSEC TABLE
ROUTING TABLE
10.88.40.96/27 dev eth5 proto kernel scope link src 10.88.40.97
10.88.40.128/27 via 10.88.40.98 dev eth5
169.254.0.0/16 dev eth5 scope link metric 1002
default via 10.88.40.98 dev eth5 proto static
```

Abbildung 179: ipsec\_look.dump

## 6) ipsec\_route.dump

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_route.dump
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
10.88.40.96 0.0.0.0 255.255.255.224 U 0 0 0 eth5
10.88.40.128 10.88.40.98 255.255.255.224 UG 0 0 0 eth5
169.254.0.0 0.0.0.0 255.255.0.0 U 1002 0 0 eth5
0.0.0.0 10.88.40.98 0.0.0.0 UG 0 0 0 eth5
```

Abbildung 180: ipsec\_route.dump

## Files from Hamburg:

### 1) /etc/ipsec.secrets

```
|[dhcp-172-21-25-244:~ hanife$ cat /Users/hanife/Documents/GitHub/Projekt/bilder/dateien/aufgabe\ 7/mid/rsa.secrets
: RSA {
 # RSA 2192 bits pnid4-mid-hh.localdomain Thu Jan 25 13:45:32 2018
 # for signatures only, UNSAFE FOR ENCRYPTION
 #pubkey=0sAQPUzpFjze9/kcBsRcvS+1xkO2iL9dHbp42NX1HoAdj3dkuFs25hD5XmdD8AURZYwHUWiydGltdw0AwkIVvIgaCCYYpHTRBB9
EAzDMhyVZFrLT+GQdK8yh+6933R1VW7R2jmJy1UI+qJZUCKdvMrU6T1Fc6CR3twk6BSXMwpj2o1cdmCZFL10R3FPrveMjTCI3LrNof8I1BtqG9GJFRb
/z/y6lUu0g/BSGDHQ/Tm983xrN8pQteQY+0mVAwNpkEZxbKtHAWEbqvLkppd4C94rz38os8wXZMY3GnSCdI0MIKpIlwi5wvzkv0vfgYsvQ+/jpy6v7Zm
hk0ioFVKSLMNcICigVFqWceIxEZlzoS6mVWL
 Modulus: 0xd9ba9163cdef7f91c06c45c552fb5c643b68b62fd7476e9e36357947a000e3ddd92e16cdb9843e5799d0fc0144596301d
4588c9d1a5b5dc3403090856f220682098291d344107d180cc3321c95645acb4fe19074af3287eepbdff7465556ed1da39e627295423ea896540
a476f32b53a4f515ce82477b7893a0525ccc0f8f6a2571d9826452f5391dc53ebde3234c22372eb3687fc23500da86f4624545bf3fff2ea552e
d20fc14860c743f4e6f7cdf1acd2942d79a418fb499502f369904cd76cab4701611baaf2e4a697780bde2bdfca2cf305d9318dc69d209d23430
82a9225c22e70bf392fd2f7e062cb0fb8e9cbabfb666864d22a0554a4ac2cc35c2028a0545a9671e22b119973a12ea655614b
 PublicExponent: 0x03
 # everything after this point is CKA_ID in hex format when using NSS
 PrivateExponent: 0x0a6e8b382a805aa01fd36ce747c0835ea74d1b5b
 Prime1: 0x0a6e8b382a805aa01fd36ce747c0835ea74d1b5b
 Prime2: 0x0a6e8b382a805aa01fd36ce747c0835ea74d1b5b
 Exponent1: 0x0a6e8b382a805aa01fd36ce747c0835ea74d1b5b
 Exponent2: 0x0a6e8b382a805aa01fd36ce747c0835ea74d1b5b
 Coefficient: 0x0a6e8b382a805aa01fd36ce747c0835ea74d1b5b
 CKAIDNSS: 0xa6e8b382a805aa01fd36ce747c0835ea74d1b5b
}
do not change the indenting_of that "}"|
```

Abbildung 181: /etc/ipsec.secrets

## 2) /etc/ipsec.conf

```
[root@pnid4-mid-hh ipsec.d]# cat rsa.conf
conn rsa
 type=tunnel
 auto=add
 authby=rsasig
 #SRC HOST pnid4-mid-hh
 left=10.88.40.130
 leftsubnet=10.88.40.128/27
 leftsasigkey=@sAQPZupF1ze@/kcBsRcvS+1xk02i2L9dHbp42NXlHoAdj3dkuFs25hD5XmdD8AURZYwhHUWIdyGltdw@AwkIVvIgaCCYYpHTRB9EAzDMhyVZFrLT+G0dK8yh+6933rlVW7R2jnmJylUI+qJZUCkdV
MrUGt1Fc6CR3twk6BXMwPj20lcdmcZFL10R3FPrveMjTC13LrNof81BtqG9GJFrB/z/y6lu0g/BSGDHO/Tm983xrN8pQteaQY+0mVAvNpKEzXbKtHAWebqvLkppd4C94r38os8wXZMY3GnSCdI0MIkPIlwlw5wzvkv0vfgYsvQ
+jpy6v7Zmlhk0idFVK5slMNCICigVfqNceIrEZlzoS6mWFL
 #DST pnid4-cnt-bl
 right=10.88.40.97
 rightsubnet=10.88.40.96/27
 rightssasigkey=@sAQo0l+TVBlYe7TH4ZwMjplR7e7dY6pdTUvR08w2bg2LzeTzhZ+ycMlbeb9lY6Nm1i0wIdUNQcQ1Fu0bpLynsdscU/S9y200T5oP0arpNpID1J07X0Anm+MZJJ00005Iis5F/SxeTANi6V64s2
ZVWoBTX4szortmj4FlG5cIAyql77aeC0+tjKRf0r8yKzysZE7lquSueB2+upNKjM3XL/ZYcfnU1VnxHsxzj+gyo3lbeGph0Tyl9qN0ZIAuHxDNd18N5oLH2Tjaz2Myxuc0qaJ1qY67PNV50+7tobdrERa3kZ/WCGjn2nBXJb0
jVT+eaBL6VnPjRtIpF7nxAZ26ofsy047UQ04U$RRIRhb0Y/ah
```

Abbildung 182: /etc/ipsec.conf

## 3) /etc/ipsec\_iptables - a file which contains your iptables rules

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_iptables
Generated by iptables-save v1.4.7 on Thu Jan 25 15:26:03 2018
*filter
:INPUT ACCEPT [1:328]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.130/32 -d 10.88.40.97/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.130/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Thu Jan 25 15:26:03 2018
```

Abbildung 183: /etc/ipsec\_iptables

#### 4) ipsec\_ifconfig.dump

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_ifconfig.dump
eth5 Link encap:Ethernet HWaddr 00:50:56:2B:56:BC
 inet addr:10.88.40.130 Bcast:10.88.40.159 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe2b:56bc/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:16145 errors:0 dropped:0 overruns:0 frame:0
 TX packets:90 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:1500187 (1.4 MiB) TX bytes:15154 (14.7 KiB)

lo Link encap:Local Loopback
 inet addr:127.0.0.1 Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING MTU:16436 Metric:1
 RX packets:553 errors:0 dropped:0 overruns:0 frame:0
 TX packets:553 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:68277 (66.6 KiB) TX bytes:68277 (66.6 KiB)

[root@pnid4-mid-hh ipsec.d]#]
```

Abbildung 184: ipsec\_ifconfig.dump

#### 5) ipsec\_look.dump

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_look.dump
pnid4-mid-hh.localdomain Thu Jan 25 15:31:50 CET 2018
IPSEC TABLE
ROUTING TABLE
10.88.40.96/27 via 10.88.40.129 dev eth5
10.88.40.128/27 dev eth5 proto kernel scope link src 10.88.40.130
169.254.0.0/16 dev eth5 scope link metric 1002
default via 10.88.40.129 dev eth5 proto static
```

Abbildung 185: ipsec\_look.dump

## 6) ipsec\_route.dump

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_route.dump
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
10.88.40.96 10.88.40.129 255.255.255.224 UG 0 0 0 eth5
10.88.40.128 0.0.0.0 255.255.255.224 U 0 0 0 eth5
169.254.0.0 0.0.0.0 255.255.0.0 U 1002 0 0 eth5
0.0.0.0 10.88.40.129 0.0.0.0 UG 0 0 0 eth5
[root@pnid4-mid-hh ipsec.d]# █
```

Abbildung 186: ipsec\_route.dump

## Exercise 8: Set up a network-to-network VPN using preshared key

To create a network-to-network VPN as shown in Figure 3 using preshared key both systems must have OPENS/WAN properly installed and tested. Next, you must ensure that IP networking is functioning. For this exercise you will capture http packets between the hosts. This capture will allow you to compare and prove that IPSec is functioning after the tunnel is created between the hosts. Make sure Apache and Wireshark are installed.

Network-to-network ist eine VPN-Situation, indem eine Vielzahl lokaler Netzwerke zu einem virtuellen Netzwerk zusammengeschaltet werden. Ebenfalls besteht die Möglichkeit eine Festverbindung zwischen den verschiedenen Netzwerken, die an unterschiedlichen Orten sind, herzustellen. Jedoch ist der Kostenaufwand hierfür deutlich höher als wenn man die Netzwerke mit einer VPN-Verbindung über das Internet aufbaut.

- i) Test connectivity between pnidX-mid-hh and pnidX-cnt-bln using ping, see figure 3 and save the result.

### Ping between pnid4-mid-hh and pnid4-cnt-bln

```
[root@pnid4-mid-hh ~]# ping -c 5 10.88.40.70
PING 10.88.40.70 (10.88.40.70) 56(84) bytes of data.
64 bytes from 10.88.40.70: icmp_seq=1 ttl=60 time=1.28 ms
64 bytes from 10.88.40.70: icmp_seq=2 ttl=60 time=1.87 ms
64 bytes from 10.88.40.70: icmp_seq=3 ttl=60 time=1.92 ms
64 bytes from 10.88.40.70: icmp_seq=4 ttl=60 time=1.60 ms
64 bytes from 10.88.40.70: icmp_seq=5 ttl=60 time=1.69 ms

--- 10.88.40.70 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4017ms
rtt min/avg/max/mdev = 1.287/1.677/1.927/0.227 ms
[root@pnid4-mid-hh ~]#
```

Abbildung 187: ping-Ergebnis

### Zenmap between pnid4-mid-hh and pnid4-cnt-bln

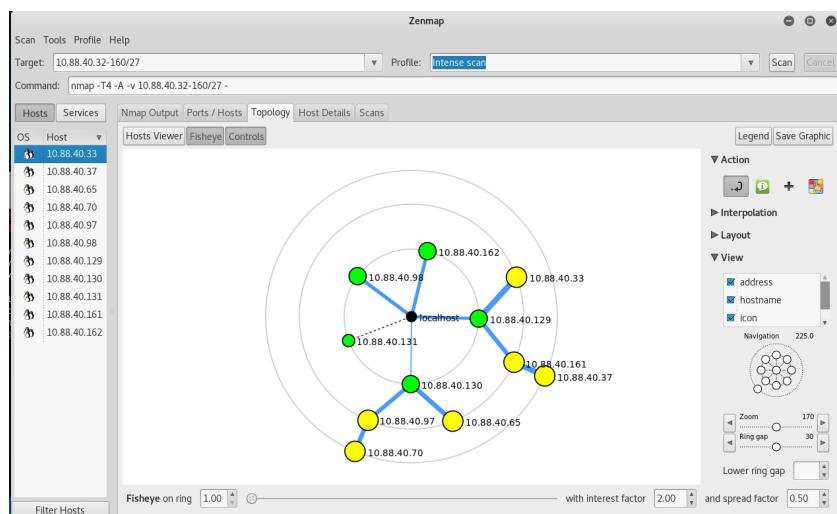
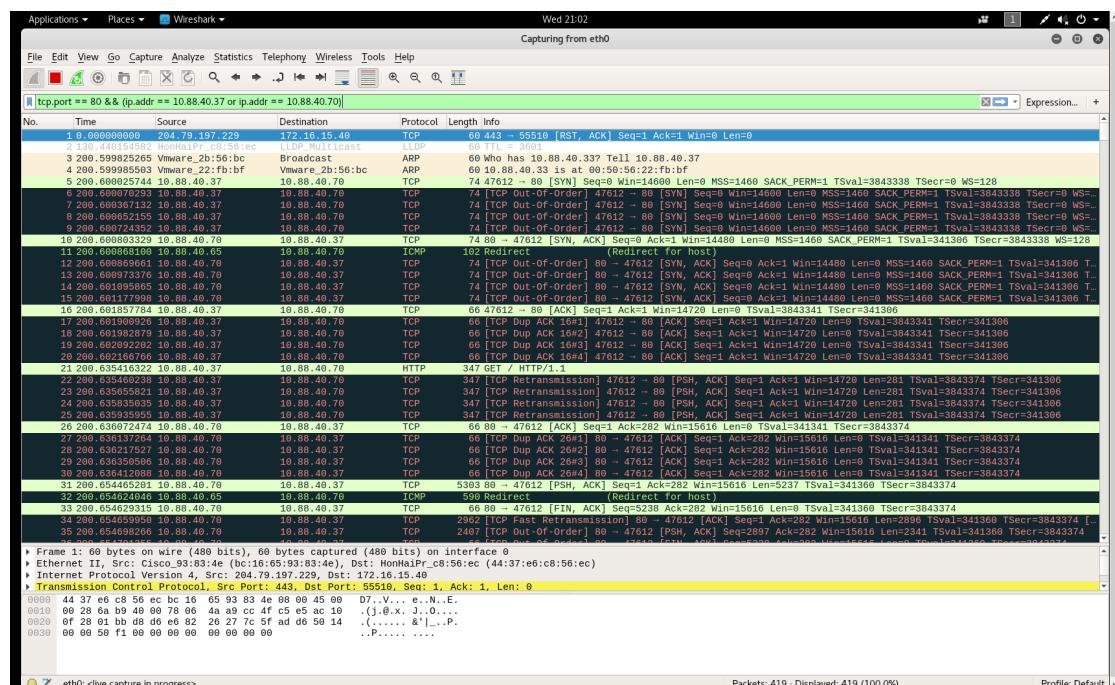
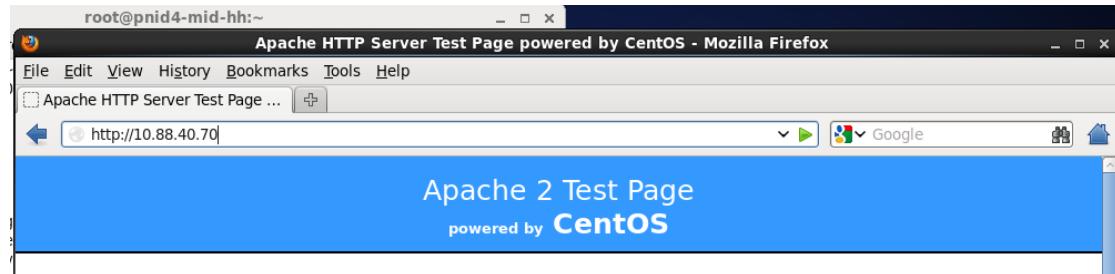


Abbildung 188: zenmap for testing the connectivity

- ii) Capture http packets from pnidX-mid-hh to pnidX-cnt-bln see figure 3, by using Wireshark on pnidX-kln-st before establishing a tunnel and save the packet captured. Please include this with your lab report

## HTTP-Pakete mit Wireshark filtern

dst 10.88.40.130 && tcp && port 80



iii) Establish a tunneled IPSec connection, using the preshared key Method between the two gateways. Confirm that the connection is established. Capture again http packets from pnidX-mid-hh to pnidX-cnt-bln see figure 3, by using Wireshark on pnidX-kln-st after establishing a tunnel and save the packet captured. Please include this with your lab report

Mit Hilfe des Befehls # ipsec ranbits 256 > net-to-net-psk.secrets wird ein zufällig erzeugter Key generiert, der in der Datei net-to-net-psk.secrets gespeichert wird.

```
[root@pnid4-mid-hh ipsec.d]# ipsec ranbits 256 > /etc/ipsec.d/net-to-net-psk.secrets
[root@pnid4-mid-hh ipsec.d]# ls -l
total 4
-rw-r--r--. 1 root root 74 Feb 14 15:17 net-to-net-psk.secrets
[root@pnid4-mid-hh ipsec.d]# vi net-to-net-psk.secrets
[root@pnid4-mid-hh ipsec.d]# cat net-to-net-psk.secrets
10.88.40.161 10.88.40.97 : PSK 0xab0e1199_27c7bf03_92efd7f3_f0903705_713c1411_6daleb40_e086901a_1b40a760
```

Abbildung 189: net-to-net-psk.secrets

#### Konfiguration der net-to-net-psk.conf Datei

```
[root@pnid4-mid-hh ipsec.d]# vi net-to-net-psk.conf
[root@pnid4-mid-hh ipsec.d]# cat net-to-net-psk.conf
conn net-to-net-psk
 type=tunnel
 auto=add
 authby=secret

 #GATEWAY Hamburg
 left=10.88.40.161
 leftsubnet=10.88.40.32/27
 leftnexthop=10.88.40.162

 #GATEWAY Berlin
 right=10.88.40.97
 rightsubnet=10.88.40.64/27
 rightnexthop=10.88.40.98
```

Abbildung 190: net-to-net-psk.conf

Da auf beiden Rechnern, dieselbe secrets und conf Datei ist, senden wir diese mit dem scp Protokoll rüber.

```
[root@pnid4-mid-hh ipsec.d]# scp /etc/ipsec.d/net-to-net-psk.secrets root@10.88.40.97:/etc/ipsec.d/net-to-net-psk.secrets
root@10.88.40.97's password:
net-to-net-psk.secrets
[root@pnid4-mid-hh ipsec.d]# scp /etc/ipsec.d/net-to-net-psk.conf root@10.88.40.97:/etc/ipsec.d/net-to-net-psk.conf
root@10.88.40.97's password:
net-to-net-psk.conf
```

|      |     |         |       |
|------|-----|---------|-------|
| 100% | 105 | 0.1KB/s | 00:00 |
| 100% | 240 | 0.2KB/s | 00:00 |

## Auf beiden Rechnern wird ipsec gestartet

```
[root@pnid4-mid-hh ipsec.d]# ipsec setup reload
ipsec_setup: Stopping Openswan IPsec...
ipsec_setup: stop ordered, but IPsec appears to be already stopped!
ipsec_setup: doing cleanup anyway...
ipsec_setup: Starting Openswan IPsec U2.6.32/K2.6.32-431.el6.x86_64...
ipsec_setup: no default routes detected
ipsec_setup: /usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips_enabled
```

## Der Status wird geprüft

```
[root@pnid4-mid-hh ipsec.d]# ipsec setup status
IPsec running - pluto pid: 3978
pluto pid 3978
No tunnels up
```

## Verbindung wird hergestellt und Tunnel aufgebaut

```
[root@pnid4-mid-hh ipsec.d]# ipsec auto --add net-to-net-psk
/usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips_enabled
[root@pnid4-mid-hh ipsec.d]# ipsec auto --up net-to-net-psk
104 "net-to-net-psk" #1: STATE_MAIN_I1: initiate
003 "net-to-net-psk" #1: received Vendor ID payload [Openswan (this version) 2.6.32]
003 "net-to-net-psk" #1: received Vendor ID payload [Dead Peer Detection]
003 "net-to-net-psk" #1: received Vendor ID payload [RFC 3947] method set to=109
106 "net-to-net-psk" #1: STATE_MAIN_I2: sent MI2, expecting MR2
003 "net-to-net-psk" #1: NAT-Traversal: Result using RFC 3947 (NAT-Traversal): no NAT detected
106 "net-to-net-psk" #1: STATE_MAIN_I3: sent MI3, expecting MR3
003 "net-to-net-psk" #1: received Vendor ID payload [CAN-IKEv2]
004 "net-to-net-psk" #1: STATE_MAIN_I4: ISAKMP SA established {auth=OAKLEY_PRESHARED_KEY cipher=aes_128 prf=oakley_sha group=modp2048}
117 "net-to-net-psk" #2: STATE_QUICK_I1: initiate
004 "net-to-net-psk" #2: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel mode {ESP=>0x7334e6e8 <0x63fe5269 xfrm=AES_128-HMAC_SHA1 NATOA=None NATD=None DPD=None}
```

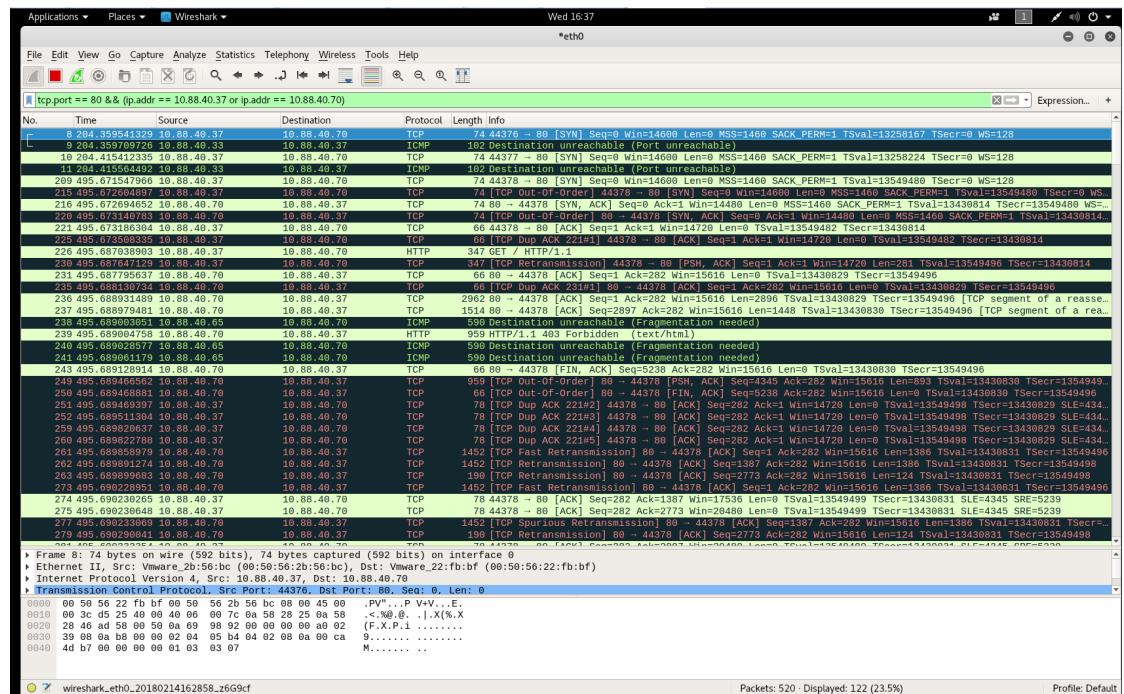
## Status wird erneut geprüft

```
[root@pnid4-mid-hh ipsec.d]# ipsec setup status
IPsec running - pluto pid: 3978
pluto pid 3978
1 tunnels up
some eroutes exist
```

## 2 Tunnels up

```
[root@ipn1d-cnt-bl ipsec.d]# ipsec setup reload
ipsec setup: Stopping Openswan IPsec...
ipsec setup: stop ordered, but IPsec appears to be already stopped!
ipsec setup: doing cleanup anyway...
ipsec setup: Starting Openswan U2.6.32/K2.6.32-431.e16.x86_64...
ipsec setup: no default routes detected
ipsec setup: /usr/libexec/ipsesd Non-FIPS mode set in /proc/sys/crypto/fips_enabled
117 "net-to-net-psk" #5: STATE QUICK I1: initiate
004 "net-to-net-psk" #5: STATE QUICK I2: sent O12, IPsec SA established tunnel mode (ESP=>0x4e051066 <0xc6077b18 xfrm=AES_128-HMAC_SHA1 NATOA=none NATD=none DPD=none)
[root@ipn1d-cnt-bl ipsec.d]# ipsec auto --add net-to-net-psk
[root@ipn1d-cnt-bl ipsec.d]# ipsec setup status
[root@ipn1d-cnt-bl ipsec.d]# ipsec setup status
IPsec running
pluto.pid: 4018
2 tunnels up
some routes exist
```

## HTTP-Pakete mittels Wireshark filtern



- iv) Secure the computer traffic by creating iptables rules to permit only IPSec traffic between the two gateways. All other, non-IPSec packets must be denied.

```
[root@pnid4-mid-hh ipsec.d]# iptables -F
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -j REJECT
[root@pnid4-mid-hh ipsec.d]# iptables-save > /etc/ipsec.d/iptables_rules
```

Abbildung 191: creating iptables rules

### iptable rules mid-hh

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_iptable_rules
Generated by iptables-save v1.4.7 on Wed Feb 14 16:02:17 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Wed Feb 14 16:02:17 2018
```

### iptable rules cnt-bln

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_iptable_rules
Generated by iptables-save v1.4.7 on Wed Feb 14 16:11:45 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Wed Feb 14 16:11:45 2018
```

v) After a successful IPSec connection has been established, create the following files using the given commands:

```
File ipsec_ifconfig_shared.dump
ifconfig > ipsec_ifconfig_Shared.dump
File ipsec_look_shared.dump
ipsec look > ipsec_look_shared.dump
File ipsec_route_shared.dump
route -n > ipsec_route_shared.dump
```

The following files must be included with the lab report:

- 1) /etc/ipsec.secrets (shared key)
- 2) /etc/ipsec.conf (shared key)
- 3) ipsec\_ifconfig\_shared.dump
- 4) ipsec\_look\_shared.dump
- 5) ipsec\_route\_shared.dump

#### Files from Berlin:

```
[root@pnid4-cnt-bln ipsec.d]# ifconfig > ipsec_ifconfig_shared.dump
[root@pnid4-cnt-bln ipsec.d]# ipsec look > ipsec_look_shared.dump
[root@pnid4-cnt-bln ipsec.d]# route -n > ipsec_route_shared.dump
```

#### 1) /etc/ipsec.secrets

```
[root@pnid4-cnt-bln ipsec.d]# cat net-to-net-psk.secrets
10.88.40.161 10.88.40.97 : PSK_0xab0e1199_27c7bf03_92efd7f3_f0903705_713c1411_6daleb40_e086901a_1b40a760
```

Abbildung 192: /etc/ipsec.secrets

## 2) /etc/ipsec.conf

```
[root@pnid4-cnt-bln ipsec.d]# cat net-to-net-psk.conf
conn net-to-net-psk
 type=tunnel
 auto=add
 authby=secret

 #GATEWAY Hamburg
 left=10.88.40.161
 leftsubnet=10.88.40.32/27
 leftnexthop=10.88.40.162

 #GATEWAY Berlin
 right=10.88.40.97
 rightsubnet=10.88.40.64/27
 rightnexthop=10.88.40.98
```

Abbildung 193: /etc/ipsec.conf

## 3) /etc/ipsec\_iptables - a file which contains your iptables rules

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_iptable_rules
Generated by iptables-save v1.4.7 on Wed Feb 14 16:11:45 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Wed Feb 14 16:11:45 2018
```

Abbildung 194: /etc/ipsec\_iptables

#### 4) ipsec\_ifconfig.dump

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_ifconfig_shared.dump
eth6 Link encap:Ethernet HWaddr 00:50:56:23:DC:D5
 inet addr:10.88.40.97 Bcast:10.88.40.127 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe23:dcd5/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:8876 errors:0 dropped:0 overruns:0 frame:0
 TX packets:7657 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:622049 (607.4 KiB) TX bytes:569771 (556.4 KiB)

eth7 Link encap:Ethernet HWaddr 00:50:56:27:AE:35
 inet addr:10.88.40.65 Bcast:10.88.40.95 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe27:ae35/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:3118 errors:0 dropped:0 overruns:0 frame:0
 TX packets:2705 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:258858 (252.7 KiB) TX bytes:169845 (165.8 KiB)

lo Link encap:Local Loopback
 inet addr:127.0.0.1 Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING MTU:16436 Metric:1
 RX packets:336 errors:0 dropped:0 overruns:0 frame:0
 TX packets:336 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:32920 (32.1 KiB) TX bytes:32920 (32.1 KiB)
```

Abbildung 195: ipsec\_ifconfig.dump

#### 5) ipsec\_look.dump

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_look_shared.dump
pnid4-cnt-bln Wed Feb 14 16:17:45 CET 2018
IPSEC TABLE
ROUTING TABLE
```

Abbildung 196: ipsec\_look.dump

## 6) ipsec\_route.dump

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_route_shared.dump
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
10.88.40.64 0.0.0.0 255.255.255.224 U 0 0 0 eth7
10.88.40.96 0.0.0.0 255.255.255.224 U 0 0 0 eth6
10.88.40.32 10.88.40.98 255.255.255.224 UG 0 0 0 eth6
10.88.40.128 10.88.40.98 255.255.255.224 UG 0 0 0 eth6
10.88.40.160 10.88.40.98 255.255.255.224 UG 0 0 0 eth6
169.254.0.0 0.0.0.0 255.255.0.0 U 1002 0 0 eth7
169.254.0.0 0.0.0.0 255.255.0.0 U 1003 0 0 eth6
```

Abbildung 197: ipsec\_route.dump

## Files from Hamburg:

```
[root@pnid4-mid-hh ipsec.d]# ifconfig > ipsec_ifconfig_shared.dump
[root@pnid4-mid-hh ipsec.d]# ipsec look > ipsec_look_shared.dump
[root@pnid4-mid-hh ipsec.d]# route -n > ipsec_route_shared.dump
```

## 1) /etc/ipsec.secrets

```
[root@pnid4-mid-hh ipsec.d]# cat net-to-net-psk.secrets
10.88.40.161 10.88.40.97 : PSK 0xab0e1199_27c7bf03_92efd7f3_f0903705_713c1411_6daleb40_e086901a_1b40a760
```

Abbildung 198: /etc/ipsec.secrets

## 2) /etc/ipsec.conf

```
[root@pnid4-mid-hh ipsec.d]# cat net-to-net-psk.conf
conn net-to-net-psk
 type=tunnel
 auto=add
 authby=secret

 #GATEWAY Hamburg
 left=10.88.40.161
 leftsubnet=10.88.40.32/27
 leftnexthop=10.88.40.162

 #GATEWAY Berlin
 right=10.88.40.97
 rightsubnet=10.88.40.64/27
 rightnexthop=10.88.40.98
```

Abbildung 199: /etc/ipsec.conf

## 3) /etc/ipsec\_iptables - a file which contains your iptables rules

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_iptable_rules
Generated by iptables-save v1.4.7 on Wed Feb 14 16:02:17 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p esp -j ACCEPT
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p udp -m udp --dport 500 -j ACCEPT
-A FORWARD -s 10.88.40.161/32 -d 10.88.40.97/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -s 10.88.40.97/32 -d 10.88.40.161/32 -p udp -m udp --dport 4500 -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
COMMIT
Completed on Wed Feb 14 16:02:17 2018
```

Abbildung 200: /etc/ipsec\_iptables

#### 4) ipsec\_ifconfig.dump

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_ifconfig_shared.dump
eth6 Link encap:Ethernet HWaddr 00:50:56:22:FB:BF
 inet addr:10.88.40.33 Bcast:10.88.40.63 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe22:fbbf/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:3266 errors:0 dropped:0 overruns:0 frame:0
 TX packets:2715 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:277230 (270.7 KiB) TX bytes:171084 (167.0 KiB)

eth7 Link encap:Ethernet HWaddr 00:50:56:33:17:07
 inet addr:10.88.40.161 Bcast:10.88.40.191 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe33:1707/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:8973 errors:0 dropped:0 overruns:0 frame:0
 TX packets:7638 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:630705 (615.9 KiB) TX bytes:576697 (563.1 KiB)

lo Link encap:Local Loopback
 inet addr:127.0.0.1 Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING MTU:16436 Metric:1
 RX packets:240 errors:0 dropped:0 overruns:0 frame:0
 TX packets:240 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:19984 (19.5 KiB) TX bytes:19984 (19.5 KiB)
```

Abbildung 201: ipsec\_ifconfig.dump

#### 5) ipsec\_look.dump

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_look_shared.dump
pnid4-mid-hh.localdomain Wed Feb 14 17:01:16 CET 2018
IPSEC TABLE
ROUTING TABLE
```

Abbildung 202: ipsec\_look.dump

## 6) ipsec\_route.dump

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_route_shared.dump
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
10.88.40.64 10.88.40.162 255.255.255.224 UG 0 0 0 eth7
10.88.40.96 10.88.40.162 255.255.255.224 UG 0 0 0 eth7
10.88.40.32 0.0.0.0 255.255.255.224 U 0 0 0 eth6
10.88.40.128 10.88.40.162 255.255.255.224 UG 0 0 0 eth7
10.88.40.160 0.0.0.0 255.255.255.224 U 0 0 0 eth7
169.254.0.0 0.0.0.0 255.255.0.0 U 1002 0 0 eth7
169.254.0.0 0.0.0.0 255.255.0.0 U 1003 0 0 eth6
```

Abbildung 203: ipsec\_route.dump

## Exercise 9: Set up a network-to-network VPN using RSA secrets keys

To create a network-to-network VPN as shown in Figure 2 using RSA keys both systems must have OPENS/WAN properly installed and tested. Next, you must ensure that IP networking is functioning. For this exercise you will capture http packets between the hosts. This capture will allow you to compare and prove that IPSec is functioning after the tunnel is created between the hosts. Make sure Apache and Wireshark are installed.

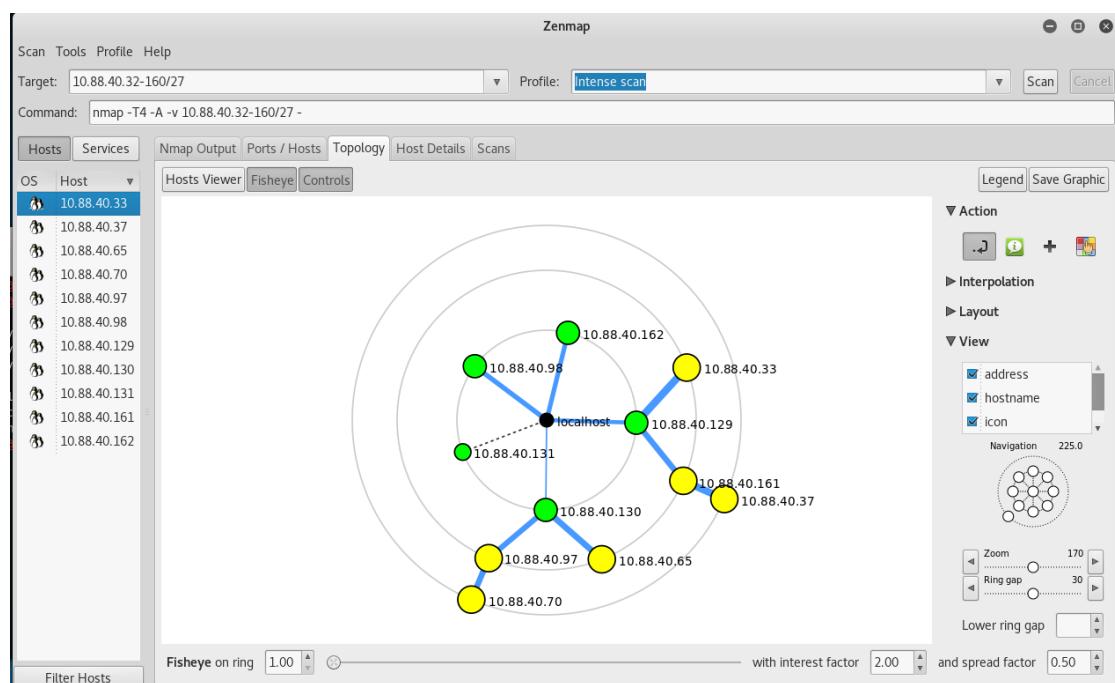
- i) Test connectivity between pnidX-mid-hh and pnidX-cnt-bln using ping, see figure 2 and save the result.

## Ping between pnid4-mid-hh and pnid4-cnt-bln

```
[root@pnid4-mid-hh ~]# ping -c 5 10.88.40.70
PING 10.88.40.70 (10.88.40.70) 56(84) bytes of data.
64 bytes from 10.88.40.70: icmp_seq=1 ttl=60 time=1.28 ms
64 bytes from 10.88.40.70: icmp_seq=2 ttl=60 time=1.87 ms
64 bytes from 10.88.40.70: icmp_seq=3 ttl=60 time=1.92 ms
64 bytes from 10.88.40.70: icmp_seq=4 ttl=60 time=1.60 ms
64 bytes from 10.88.40.70: icmp_seq=5 ttl=60 time=1.69 ms

--- 10.88.40.70 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4017ms
rtt min/avg/max/mdev = 1.287/1.677/1.927/0.227 ms
[root@pnid4-mid-hh ~]#
```

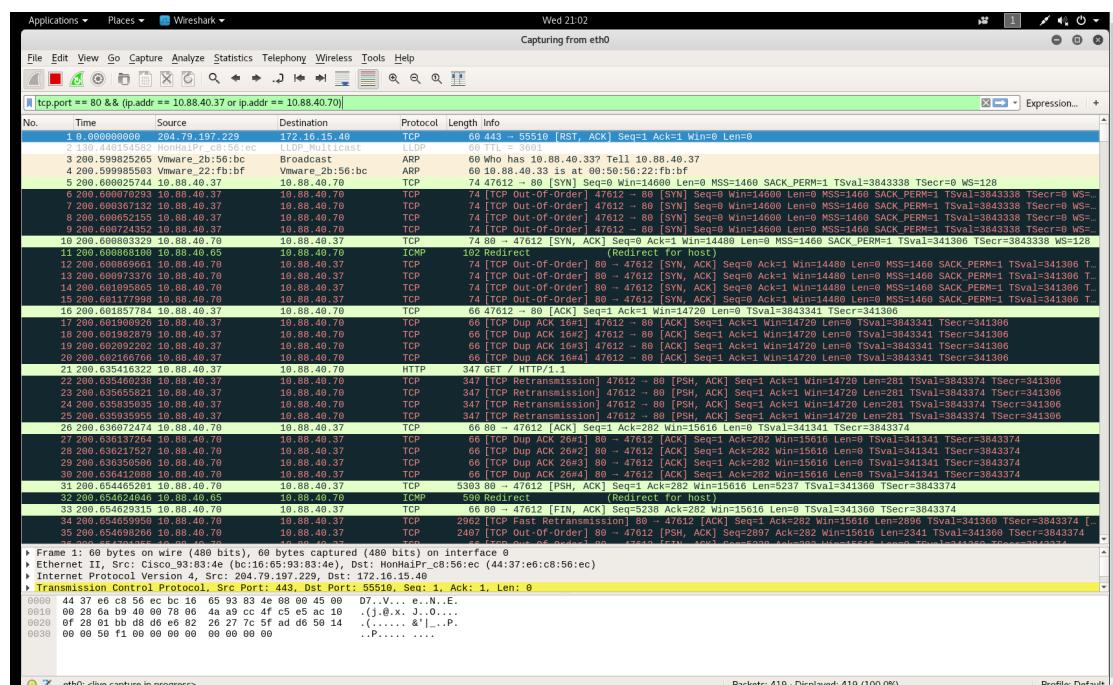
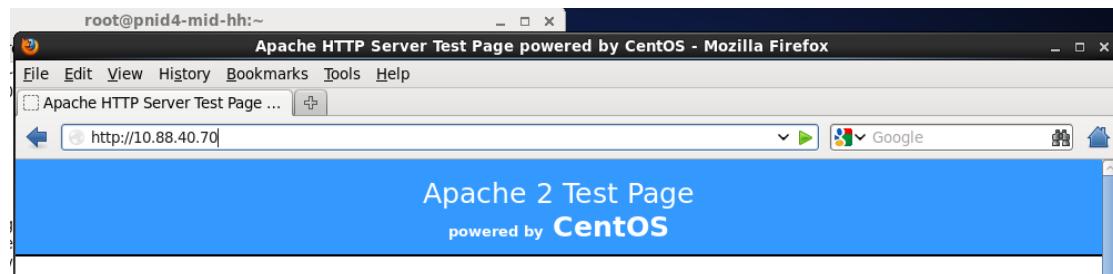
Zenmap between pnid4-mid-hh and pnid4-cnt-blw



- ii) Capture http packets from pnidX-mid-hh to pnidX-cnt-blن see figure 2, by using Wireshark on pnidX-kln-st before establishing a tunnel and save the packet captured.

Please include this with your lab report

## HTTP-Pakete mit Wireshark filtern dst 10.88.40.130 && tcp && port 80



iii) Establish a tunneled IPSec connection, using the RSA Method between the two gateways pnidX-gw-hh and pnidX-gw-bln

Confirm that the connection is established.

Capture again http packets from pnidX-mid-hh to pnidX-cnt-bln see figure 2, by using Wireshark on pnidX-kln-st after establishing a tunnel and save the packet captured. Please include this with your lab report.

#### Privaten RSA-Schlüssel generieren auf beiden Gateways

a) certutil -N -d /etc/ipsec.d

##### privater Schlüssel von Gateway-Berlin

```
[root@pnid4-cnt-bln ipsec.d]# certutil -N -d /etc/ipsec.d
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
```

##### privater Schlüssel von Gateway-Hamburg

```
[root@pnid4-mid-hh ipsec.d]# certutil -N -d /etc/ipsec.d
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
```

b) ipsec newhostkey --configdir /etc/ipsec.d/ --output /etc/ipsec.d/keys.secrets  
**Der Befehl generiert 2 RSA Authentifikationsschlüssel für die beiden Gateways.**

### net-to-net-rsa.secrets

```
[root@pnid4-cnt-bln ipsec.d]# ipsec newhostkey --configdir /etc/ipsec.d/ --output /etc/ipsec.d/net-to-net-rsa.secrets
Generated RSA key pair using the NSS database
[root@pnid4-cnt-bln ipsec.d]# cat net-to-net-rsa.secrets
: RSA {
 # RSA 2192 bits pnid4-cnt-bln Wed Feb 14 18:05:34 2018
 # for signatures only, UNSAFE FOR ENCRYPTION
 #pubkey=0xa0053kaqrW8k/NnhfqqqNv1475Vyx8dqqrw3mE65z/6UroQqe3H6T/XNEzDUENT18M19oTOKP+FeeFs0ZmCV5gjkEkFjyoq6
UjPVh1XtWe05cGYIHn+ph/Z6r6jSt8y4B8hzj8rVuf27rqetnKBN=NCoxHygB2RBF4R5D1vgCwQ0VTKUsbSzzyyf81CeAuHhQmaEca2Fl02r2iK9L37
NYZF0gyWjs2h7FFC81Ek39MLlf3a2W0IwEVfpTzdfR0YfjHdhqyS/3TKBaCM1865yGloaTMRqjRWKfVsLlmkUkhn9We5bKKu2lnIf5b1hftsHt61+K
03V671e0G+6pq531cv0LwhZvbCtz1tnJN
 Modulus: 0xb9de46aaad6fc93f3671df82aa8d5f5e3b495cb105daaad6de6106e73fffa511a06a9edc7e93fd7344cc350436dd7c335
f55a13d0a3fe16179e16cd19982579823904905872a2ae9488751d57b567b4e51982079fe821fd9eabe894af32e01f21ce3f2b567e9d9eeba9e
b6728137e34231bf21e0f644151e11e438af8025903954ca5126d2fc3cb27fc2027808b785099a11c6b6165d36af688a4fbdf3586450e0c96
26cda7ec5142f22124fd5a595fddad96408c0455ffa5365d8ddf1f181631dd86a612ff74ca05a08c9a2f3a4b21a5d1a4e4311a8745691f56c2cb9a4
52484df567b96ca92eda59c81526e5b611bd487b7ad7e2b0e27757a4c780le1ba9aacde20af5742d6859bdb0adce5b6724d
 PublicExponent: 0x03
 # everything after this point is CKA ID in hex format when using NSS
 PrivateExponent: 0x156557c2dde061418ff415fd61982e3946da63b3
 Prime1: 0x156557c2dde061418ff415fd61982e3946da63b3
 Prime2: 0x156557c2dde061418ff415fd61982e3946da63b3
 Exponent1: 0x156557c2dde061418ff415fd61982e3946da63b3
 Exponent2: 0x156557c2dde061418ff415fd61982e3946da63b3
 Coefficient: 0x156557c2dde061418ff415fd61982e3946da63b3
 CKAIDNS: 0x156557c2dde061418ff415fd61982e3946da63b3
}
do not change the indenting of that "}"
```

Abbildung 204: net-to-net-rsa.secrets

```
[root@pnid4-mid-hh ipsec.d]# ipsec newhostkey --configdir /etc/ipsec.d/ --output
/etc/ipsec.d/net-to-net-rsa.secrets
Generated RSA key pair using the NSS database
[root@pnid4-mid-hh ipsec.d]# cat net-to-net-rsa.secrets
: RSA {
 # RSA 2192 bits pnid4-mid-hh.localdomain Wed Feb 14 17:49:35 2018
 # for signatures only, UNSAFE FOR ENCRYPTION
 #pubkey=0xAQPkULJJL2+TeaTubW9hZ3kBcICBqOsZpmDANTIPr0Err2Kkge990tx80na+L
kVddQ3wRor+SxkR7II4TNS0QQZhwsH4gExLqfk008VfkLEUzv7QCi5Xost10xe2gkA1prCeCTRq19FkR
tgYg8CbfcjykeBtQGeonYXXfMBwyu7xvGghywG4lN/DgDIZ/ueBuzz2iI0p15ghSzvd4rhsh5t06k
lRHp+6F+eSoQa8iEB4FK-x6XXkNMwAFY7zsPJzBGvDl8EXX8vaKC+MPkmf2PTmpJowskDH/DpoSjVL
SZf2S05fYCKMdUyA7w4iWmxv9aa9qKKShDQ0iM8w3oeC5vSjEJLPpZH0YU8NNii7
 Modulus: 0xe45252492f6ff4de69351b5bd859de405c20206a3ac66998300d4c83eb38
4aebd8a9207bdf50b71f289dfa8b9157d437c11a2bf9293147b208e13352d10419856b07e201312
ea7e438ef151642c4533bf4028b95e8b2d74c5eda0900d69ac27824d1ab5f45911b60620f026df
723ca4cde2133c6b9ea276176177cc070cb2bbbc6f1a0872c06e2537f0e00c8cffb9e06ecf68883a
9239821499bde2b86c879b68ea49511e9fba17e792a106bc88407814afb1e975e434c59a00563bc
ec3c9cc11af0e5f045d7f2f68a0be30f926d9fd84e6a49a30b240c7fc3a684a354b4997f64b449f
60228c75c53203bc388969b1bfd69af6a28a4a10c3ab488cf30de8782e6f4a31092cfa591f4614f0
d3628bb
 PublicExponent: 0x03
 # everything after this point is CKA ID in hex format when using NSS
 PrivateExponent: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 Prime1: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 Prime2: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 Exponent1: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 Exponent2: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 Coefficient: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 CKAIDNS: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
}
do not change the indenting of that "}"
```

Abbildung 205: net-to-net-rsa.secrets

## net-to-net-rsa.conf

```
[root@pnid4-cnt-bln ipsec.d]# cat net-to-net-rsa.conf
conn net-to-net-rsa
 type=tunnel
 auto=add
 authby=rsa sig

 #GATEWAY Hamburg
 left=10.88.49.161
 leftsubnet=10.88.49.32/27
 leftrsa_sigkey=@AOPKU1JJL2/+TeaTUbW9hZ3kBICBqOsZpnDANTIPr0Err2Kkge990tx8ona+LvVdd3wRor+Skr7I14TNS0QZhwSHqExLqfk008VFKEUzv7QCi5Xost10xe2gkA1prCeCTRq19FrRtgYg8C
bfcjykeITPGueonXYXfMBwyu7xvGhyw64lN/DgBIz/ueBu211Op15gh5Zvd4rhsh5t06kRp+6F+e5o0a81EB4FK+x6XXNMWafY7zsPJzBgvd8EXX8vaKC+MPkmz12PTmpJowskDH/Dpo5jVL5zf2s0s5fYCKMdUyA7
w4iWmx9aa5qKShDq91Mw3ec5v5jEJLPpZH0YU8NNNi7N

 #GATEWAY Berlin
 right=10.88.49.97
 rightsubnet=10.88.49.64/27
 rightrsa_sigkey=@A0053kanrH8k/NnhfgqqNx147SVyxBdqqrW3mEG5z/6URoGqe3H6T/XNE20UENT18M19VoT0KP+FheeFs02mCV5gjkEkFjyoq6U1PVh1XtWe05cGYIHn+gh/Z6r635t8y4B8hzjBrVufZ7rqet
nKBN+NCOxv/yH02R8FR4R5D1vgCWQ0VTKUsbSzzyf81CeAuHhQmaEcazfl02r21k9L37NYZF0gyW3s2n7FFC81Ek9Wllf3a2WQ1wEVfpTZdJR8YFJdhqYS/3TK8aCMm1865yG10aTKMqHRWkfVsLLmkUhN9we9bKku2tn2F
SblhtFtSht61+kOJ3V6Tie0o+Gpq53icv0Lwh2vbCtzltN
```

Abbildung 206: net-to-net-rsa.conf

## ipsec wird gestartet auf beiden Gateways

```
[root@pnid4-cnt-bln ipsec.d]# ipsec setup reload
ipsec_setup: Stopping Openswan IPsec...
ipsec_setup: Starting Openswan IPsec U2.6.32/K2.6.32-431.el6.x86_64...
ipsec_setup: no default routes detected
ipsec_setup: /usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips_enabled
```

```
[root@pnid4-mid-hh ipsec.d]# ipsec setup reload
ipsec_setup: Stopping Openswan IPsec...
ipsec_setup: Starting Openswan IPsec U2.6.32/K2.6.32-431.el6.x86_64...
ipsec_setup: no default routes detected
ipsec_setup: /usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips_enabled
```

## Verbindung wird in Datenbank eingelesen

```
[root@pnid4-mid-hh ipsec.d]# ipsec auto --add net-to-net-rsa
/usr/libexec/ipsec/addconn Non-fips mode set in /proc/sys/crypto/fips_enabled
```

## Verbindung wird hergestellt und ein Tunnel baut sich auf

```
[root@pnid4-mid-hh ipsec.d]# ipsec auto --up net-to-net-rsa
104 "net-to-net-rsa" #1: STATE MAIN II: initiate
003 "net-to-net-rsa" #1: received Vendor ID payload [Openswan (this version) 2.6.32]
003 "net-to-net-rsa" #1: received Vendor ID payload [Dead Peer Detection]
003 "net-to-net-rsa" #1: received Vendor ID payload [RFC 3947] method set to=109
106 "net-to-net-rsa" #1: STATE MAIN I2: sent MI2, expecting MR2
003 "net-to-net-rsa" #1: NAT-Traversal: Result using RFC 3947 (NAT-Traversal): no NAT detected
108 "net-to-net-rsa" #1: STATE MAIN I3: sent MI3, expecting MR3
003 "net-to-net-rsa" #1: received Vendor ID payload [CAN-IKEv2]
004 "net-to-net-rsa" #1: STATE MAIN I4: ISAKMP SA established {auth=OAKLEY_RSA_SIG cipher=aes_128 prf=oakley_sha group=modp2048}
117 "net-to-net-rsa" #2: STATE QUICK I1: initiate
004 "net-to-net-rsa" #2: STATE QUICK I2: sent QI2, IPsec SA established tunnel mode {ESP=>0xc1608fea <0xb5b8102a xfrm=AES_128-HMAC_SHA1 NAT0A=none NATD=none DPD=none}
[root@pnid4-mid-hh ipsec.d]# ipsec setup status
IPsec running - pluto pid: 4896
1 tunnels up
some routes exist
```

## 2 Tunnels up

```
[root@pnid4-cnt-bln ipsec.d]# ipsec auto --up net-to-net-rsa
117 "net-to-net-rsa" #5: STATE QUICK_I2: initiate
004 "net-to-net-rsa" #5: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel mode {ESP=>0x1d86dbf3 <0xd95b6dfb xfrm
=AES_128-HMAC_SHA1_NATOA=none NATD=none DPD=none}
[root@pnid4-cnt-bln ipsec.d]# ipsec setup status
IPsec running - pluto pid: 4938
pluto pid 4938
2 tunnels up
some eroutes exist
```

- iv) Secure the computer traffic by creating iptables rules to permit only IPSec traffic between the two gateways. All other, non-IPSec packets must be denied.

### iptables rules für Berlin

```
[root@pnid4-cnt-bln ipsec.d]# iptables -F
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -j REJECT
[root@pnid4-cnt-bln ipsec.d]# iptables-save > /etc/ipsec.d/ipsec_iptable_rules
```

### iptable rules für Hamburg

```
[root@pnid4-mid-hh ipsec.d]# iptables -F
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -j REJECT
[root@pnid4-mid-hh ipsec.d]# iptables-save > /etc/ipsec.d/ipsec_iptable_rules
```

- v) After a successful IPSec connection has been established, create the following files using the given commands:

File ipsec\_ifconfig\_rsa.dump  
ifconfig > ipsec\_ifconfig\_rsa.dump  
File ipsec\_look\_rsa.dump  
ipsec look > ipsec\_look\_rsa.dump  
File ipsec\_route\_rsa.dump  
route - n > ipsec\_route\_rsa.dump

The following files must be included with the lab report:

- 1) /etc/ipsec.secrets (rsa key)
- 2) /etc/ipsec.conf (rsa key)
- 3) ipsec\_ifconfig\_rsa.dump
- 4) ipsec\_look\_rsa.dump
- 5) ipsec\_route\_rsa.dump

### Files from Berlin:

```
[root@pnid4-cnt-bln ipsec.d]# ifconfig > ipsec_ifconfig_rsa.dump
[root@pnid4-cnt-bln ipsec.d]# ipsec look > ipsec_look_rsa.dump
[root@pnid4-cnt-bln ipsec.d]# _route -n > ipsec_route_rsa.dump
```

### 1) /etc/ipsec.secrets

```
[root@pnid4-cnt-bln ipsec.d]# cat net-to-net-rsa.secrets
: RSA {
 # RSA 2192 bits pnid4-cnt-bln Wed Feb 14 18:05:34 2018
 # for signatures only, UNSAFE FOR ENCRYPTION
 #pubkey=0sA0053kaqrwD8k/NnhfqgnXN1475VyxBdqqrW3mEG5z/6URoGqe3H6T/XNEzDUENT18M19VoT0KP+FheeFs0ZmCV5gjKEkFjyoq6UiPVh1XtWe05cGYIHn+gh/Z6r6JSt8yB8hzj8rVufZ7rqetnKBn+NC
0xvyHgD2RBRFR45D1vgCW00VTKUsbSzzyyf81CeAuHh0maEca2Fl02r21K9L37NYZFdgYJ3s2n7FFC8iEk39Wlfsa2W0IwEVfpfZdjR8YFjHdhqYS/3TKBaCMMi1865yGl0aTkMRqHRwfvsLmkuKhN9We5bKku2ln1FSblthFt
SHT61+kO3V67IE0G-Gpq31Cv0LwNZvbCtzlnJN
 Modulus: 0xb9de46aaad06fc93f3671df82aa8d5f5e3b495cb105daaaaad06de6106e73fffa511a06a9edc7e93fd7344cc350436dd7c335f55a13d0a3fe16179e16cd199825798239049958f2a2ae9488f561d
57b56704e5c1982079fe821fd9eabe894af32e01f21ce3f2b567e9deeb9e6728137e3423bf126d2cf3c27fc2027808b785099a11c6b6165d36af688af4bdff35864
50e0c9626cda7ec5142f22124fd5a595fddad96408c0455fa5365d8d1f181613dd86a612ff74ca05a08c9a2f3a4b21a5d1a4e4311a8745691f56c2b9a452484df567b96ca92eda59c81526e5b6116d487b7ad7e28e
27757a4c478e1be1a9aecd20af5742d6859b0db0adce5b6724d
 PublicExponent: 0x03
 # everything after this point is CKA ID in hex format whee using NSS
 PrivateExponent: 0x156557c2dde061410ff415fd61982e3946da63b3
 Prime1: 0x156557c2dde061410ff415fd61982e3946da63b3
 Prime2: 0x156557c2dde061410ff415fd61982e3946da63b3
 Exponent1: 0x156557c2dde061410ff415fd61982e3946da63b3
 Exponent2: 0x156557c2dde061410ff415fd61982e3946da63b3
 Coefficient: 0x156557c2dde061410ff415fd61982e3946da63b3
 CKAIIDNS: 0x156557c2dde061410ff415fd61982e3946da63b3
}
do not change the indenting of that "}"
```

Abbildung 207: /etc/ipsec.secrets

### 2) /etc/ipsec.conf

```
[root@pnid4-cnt-bln ipsec.d]# cat net-to-net-rsa.conf
conn net-to-net-rsa
 type=tunnel
 auto=add
 authby=rsasig
 #GATEWAY Hamburg
 left=10.88.40.161
 leftsubnet=10.88.40.32/27
 leftsrasigkey=0sAQPKULJL2/+TeaTubW9hZ3kBcICBq0sZpmDANTIPr0Err2Kkgc990tx8ona+LkVddQ3wRor+5kxR7II4TNs00QZhWsH4gExLqfk008VfkLEUv7QCl5Xost10xe2gkA1prCeCTRq19FkRtgYg8C
bfcjykezTPGueonYXXYfmBwyuy7xvGgnyG4lN/Dg0DzI/ueBu2z1Op15ghSzd4rhsh5t06klRHp+6F+eSoQa81EB4FK+x6XXNM#oAFY7zsPJzbGvDl8EXX8vaKC+MPkm2f2PTmpJowskDH/DpoSjVLszf2505fYCKMdcUyA7
w41Wnxv9a9qKShD0qgiM8w3oeC5v5jEJLPPzH0YUBNNii7
 #GATEWAY Berlin
 right=10.88.40.97
 rightssubnet=10.88.40.64/27
 rightsrasigkey=0sA0053kaqrwD8k/NnhfqgnXN1475VyxBdqqrW3mEG5z/6URoGqe3H6T/XNEzDUENT18M19VoT0KP+FheeFs0ZmCV5gjKEkFjyoq6UiPVh1XtWe05cGYIHn+gh/Z6r6JSt8yB8hzj8rVufZ7rqet
nKBn+NC0xvyHgD2RBRFR45D1vgCW00VTKUsbSzzyyf81CeAuHh0maEca2Fl02r21K9L37NYZFdgYJ3s2n7FFC8iEk39Wlfsa2W0IwEVfpfZdjR8YFjHdhqYS/3TKBaCMMi1865yGl0aTkMRqHRwfvsLmkuKhN9We5bKku2ln1F
SblthFtSht61+kO3V67IE0G-Gpq31Cv0LwNZvbCtzlnJN
```

Abbildung 208: /etc/ipsec.conf

### 3) /etc/ipsec\_iptables - a file which contains your iptables rules

```
[root@pnid4-cnt-bln ipsec.d]# iptables -F
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-cnt-bln ipsec.d]# iptables -A FORWARD -j REJECT
[root@pnid4-cnt-bln ipsec.d]# iptables-save > /etc/ipsec.d/ipsec_iptable_rules
```

Abbildung 209: /etc/ipsec\_iptables

### 4) ipsec\_ifconfig.dump

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_ifconfig_rsa.dump
eth6 Link encap:Ethernet HWaddr 00:50:56:23:DC:D5
 inet addr:10.88.40.97 Bcast:10.88.40.127 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe23:dcd5/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:9321 errors:0 dropped:0 overruns:0 frame:0
 TX packets:7826 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:676932 (661.0 KiB) TX bytes:614032 (599.6 KiB)

eth7 Link encap:Ethernet HWaddr 00:50:56:27:AE:35
 inet addr:10.88.40.65 Bcast:10.88.40.95 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe27:ae35/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:3449 errors:0 dropped:0 overruns:0 frame:0
 TX packets:2766 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:313622 (306.2 KiB) TX bytes:178821 (174.6 KiB)

lo Link encap:Local Loopback
 inet addr:127.0.0.1 Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING MTU:16436 Metric:1
 RX packets:404 errors:0 dropped:0 overruns:0 frame:0
 TX packets:404 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:38340 (37.4 KiB) TX bytes:38340 (37.4 KiB)
```

Abbildung 210: ipsec\_ifconfig.dump

## 5) ipsec\_look.dump

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_look_rsa.dump
pnid4-cnt-bln Wed Feb 14 18:46:55 CET 2018
IPSEC TABLE
ROUTING TABLE
```

Abbildung 211: ipsec\_look.dump

## 6) ipsec\_route.dump

```
[root@pnid4-cnt-bln ipsec.d]# cat ipsec_route_rsa.dump
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
10.88.40.64 0.0.0.0 255.255.255.224 U 0 0 0 eth7
10.88.40.96 0.0.0.0 255.255.255.224 U 0 0 0 eth6
10.88.40.32 10.88.40.98 255.255.255.224 UG 0 0 0 eth6
10.88.40.128 10.88.40.98 255.255.255.224 UG 0 0 0 eth6
10.88.40.160 10.88.40.98 255.255.255.224 UG 0 0 0 eth6
169.254.0.0 0.0.0.0 255.255.0.0 U 1002 0 0 eth7
169.254.0.0 0.0.0.0 _ 255.255.0.0 U 1003 0 0 eth6
```

Abbildung 212: ipsec\_route.dump

## Files from Hamburg:

```
[root@pnid4-mid-hh ipsec.d]# ifconfig > ipsec_ifconfig_rsa.dump
[root@pnid4-mid-hh ipsec.d]# ipsec look > ipsec_look_rsa.dump
[root@pnid4-mid-hh ipsec.d]# route -n > ipsec_route_rsa.dump
```

## 1) /etc/ipsec.secrets

```
[root@pnid4-mid-hh ipsec.d]# cat net-to-net-rsa.secrets
: RSA {
 # RSA 2192 bits pnid4-mid-hh.localdomain Wed Feb 14 17:49:35 2018
 # for signatures only, UNSAFE FOR ENCRYPTION
 #pubkey=<0xAPKULJL2/+TeaTub9hZ3kBcICBq0s2pmDANTIPr0Err2Kkge990tx8ona+LkVdd03wRor+5kxR7II4TNs0QZhWsH4gExLqfk008VfLEUzv70C15Xost10xe2gkA1prCeCTRq19FkRtgYg8Cbfcjyk
zeITPGueonXXXFMBwyv0xGhyw641N/Dg0Iz/uEbu2110p15ghS2vd4rhsh5t06k1RHp+6f+e5o0a81EB4FK+x6XXkNMwAFY7zsPjZBgDl8EXX8vaKc+MPkm2f2PTmpJowskDH/Dpo5jVLsZf2505fyCKMdciyA7
v9aa9qKKh0Dg01M9o3oeC5v5jEJLp0ZhvYUHvYUNNii7
 Modulus: 0x45252492f6fffa4de9351b5bd859de405c20206a3c669983004c83eb384ceb8d9297bd50b71f289af8b91575d437c1la2bf929147b208e13352d10419856b07e201312ea7e438ef151
642c4533fb4028b95eb2dd74c5eda0980d69a9c782441ab5f45911b60620f26df723ca4de213c6b9ea2761776cc076cb2bbc6f1a8872c0ee2537f0e00c0cffb9e06ecf68883a9239821499bbde2b86c879b6
8ea49511e9fb17e792a106b8c8407814afbe975e434c59a080563bcecc39cc11af0e5f045d7f2f68a00e30f92bd9fd8f4e6a9a38b240c7fc7fc684a354b4997f64b449f760228c75c53203bc388969b1bf9a76a28a
4a10c3bb488cf30ade87826f14a31092cfa591f4614f0d36288b
 PublicExponent: 0x03
 # everything after this point is CKA ID in hex format when using NSS
 PrivateExponent: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 Prime1: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 Prime2: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 Exponent1: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 Exponent2: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 Coefficient: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
 CKADNS: 0x3d748af8cffda9c7f832da692638a6b9a5f8a6e9
}
do not change the indenting of that "}"
```

Abbildung 213: /etc/ipsec.secrets

## 2) /etc/ipsec.conf

```
[root@pnid4-mid-hh ipsec.d]# cat net-to-net-rsa.conf
conn net-to-net-rsa
 type=tunnel
 auto=add
 authby=rsasig
 #GATEWAY_Hamburg
 left=10.88.40.161
 leftsubnet=10.88.40.32/27
 leftrsasigkey=0xA0PKULJL2/+TeaTub9hZ3kBcICBq0s2pmDANTIPr0Err2Kkge990tx8ona+LkVdd03wRor+5kxR7II4TNs0QZhWsH4gExLqfk008VfLEUzv70C15Xost10xe2gkA1prCeCTRq19FkRtgYg8C
bfcjykezITPGueonXXXFMBwyv0xGhyw641N/Dg0Iz/uEbu2110p15ghS2vd4rhsh5t06k1RHp+6f+e5o0a81EB4FK+x6XXkNMwAFY7zsPjZBgDl8EXX8vaKc+MPkm2f2PTmpJowskDH/Dpo5jVLsZf2505fyCKMdciyA7
w41Wmxv9aa9qKKh0Dg01M9o3oeC5v5jEJLp0ZhvYUHvYUNNii7
 #GATEWAY_Berlin
 right=10.88.40.97
 rightsubnet=10.88.40.64/27
 rightsrasigkey=0xA0053kaqrW8k/NnHfgqqNx1475VyxBdqqrW3mEG5z/6URoGqe3H6T/XNEzDUEnt18M19V0t0KP+FheeFs0ZmCV5gjkEkFjyoq6UiPVh1XtWe05GyIHn+gh/Z6r6J5t8y4B8hzj8rVufZ7rqet
nKBN+NCoxvyHgD2RBFR4RS5D1vgCW00VTkUsb5zzyf8IceAuUhQmaEca2Fl02r21k9L37NYZFdgysJ5n7FFF81Ek39Wllf3a2W0IwEVfpTzdjR8YFjDhdhqy5/3TBCBaCmm1865yG10aTkMRqHRwfVsLLmkukhN9We5bKku2lnIN
SblthFtSHt61+K0J3V6Ti0eG+6pq531cv0LwhZvbCtzlnJN
```

Abbildung 214: /etc/ipsec.conf

## 3) /etc/ipsec\_iptables - a file which contains your iptables rules

```
[root@pnid4-mid-hh ipsec.d]# iptables -F
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p esp -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 500 -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.161 -d 10.88.40.97 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -p udp --dport 4500 -s 10.88.40.97 -d 10.88.40.161 -j ACCEPT
[root@pnid4-mid-hh ipsec.d]# iptables -A FORWARD -j REJECT
[root@pnid4-mid-hh ipsec.d]# iptables-save > /etc/ipsec.d/ipsec iptable rules
```

Abbildung 215: /etc/ipsec\_iptables

#### 4) ipsec\_ifconfig.dump

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_ifconfig_rsa.dump
eth6 Link encap:Ethernet HWaddr 00:50:56:22:FB:BF
 inet addr:10.88.40.33 Bcast:10.88.40.63 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe22:fbbf/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:3476 errors:0 dropped:0 overruns:0 frame:0
 TX packets:2788 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:296168 (289.2 KiB) TX bytes:194378 (189.8 KiB)

eth7 Link encap:Ethernet HWaddr 00:50:56:33:17:07
 inet addr:10.88.40.161 Bcast:10.88.40.191 Mask:255.255.255.224
 inet6 addr: fe80::250:56ff:fe33:1707/64 Scope:Link
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
 RX packets:9285 errors:0 dropped:0 overruns:0 frame:0
 TX packets:7814 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:1000
 RX bytes:687244 (671.1 KiB) TX bytes:606772 (592.5 KiB)

lo Link encap:Local Loopback
 inet addr:127.0.0.1 Mask:255.0.0.0
 inet6 addr: ::1/128 Scope:Host
 UP LOOPBACK RUNNING MTU:16436 Metric:1
 RX packets:308 errors:0 dropped:0 overruns:0 frame:0
 TX packets:308 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:0
 RX bytes:25648 (25.0 KiB) TX bytes:25648 (25.0 KiB)
```

Abbildung 216: ipsec\_ifconfig.dump

#### 5) ipsec\_look.dump

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_look_rsa.dump
pnid4-mid-hh.localdomain Wed Feb 14 18:38:34 CET 2018
IPSEC TABLE
ROUTING TABLE
```

Abbildung 217: ipsec\_look.dump

## 6) ipsec\_route.dump

```
[root@pnid4-mid-hh ipsec.d]# cat ipsec_route_rsa.dump
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
10.88.40.64 10.88.40.162 255.255.255.224 UG 0 0 0 eth7
10.88.40.96 10.88.40.162 255.255.255.224 UG 0 0 0 eth7
10.88.40.32 0.0.0.0 255.255.255.224 U 0 0 0 eth6
10.88.40.128 10.88.40.162 255.255.255.224 UG 0 0 0 eth7
10.88.40.160 0.0.0.0 255.255.255.224 U 0 0 0 eth7
169.254.0.0 0.0.0.0 255.255.0.0 U 1002 0 0 eth7
169.254.0.0 0.0.0.0 255.255.0.0 U 1003 0 0 eth6
```

Abbildung 218: ipsec\_route.dump

# Abbildungsverzeichnis

|                                                                                |    |
|--------------------------------------------------------------------------------|----|
| Abbildung 1: aktivierte bzw. deaktivierte Dienste eines runlevels . . . . .    | 4  |
| Abbildung 2: Runlevel, in denen iptables eingeschaltet bzw. ausgeschaltet sind | 4  |
| Abbildung 3: runlevel 2 wird ausgeschaltet . . . . .                           | 4  |
| Abbildung 4: Runlevel 2,3,4,5 werden deaktiviert . . . . .                     | 5  |
| Abbildung 5: chkconfig iptables on   off . . . . .                             | 5  |
| Abbildung 6: vi /etc/yum.repos.d/local.repo . . . . .                          | 6  |
| Abbildung 7: mounten . . . . .                                                 | 6  |
| Abbildung 8: yum install tftp . . . . .                                        | 6  |
| Abbildung 9: yum install tftp . . . . .                                        | 7  |
| Abbildung 10: service xinetd start . . . . .                                   | 7  |
| Abbildung 11: xinet tftp config . . . . .                                      | 7  |
| Abbildung 12: vi tftp service . . . . .                                        | 8  |
| Abbildung 13: chkconfig . . . . .                                              | 9  |
| Abbildung 14: yum install vsftpd . . . . .                                     | 9  |
| Abbildung 15: Runlevel 2 von vsftpd wird deaktiviert . . . . .                 | 10 |
| Abbildung 16: chkconfig –level . . . . .                                       | 10 |
| Abbildung 17: vsftpd runlevels . . . . .                                       | 10 |
| Abbildung 18: chkconfig –level . . . . .                                       | 10 |
| Abbildung 19: vsftpd runlevels . . . . .                                       | 10 |
| Abbildung 20: Netzwerk Interface aktivieren . . . . .                          | 11 |
| Abbildung 21: Netzwerk figure 1 . . . . .                                      | 13 |
| Abbildung 22: ifconfig Server München (Routing-Tabelle) . . . . .              | 14 |
| Abbildung 23: Server München Routing-Tabelle anwenden . . . . .                | 14 |
| Abbildung 24: ifconfig Webserver Hannover (Routing-Tabelle) . . . . .          | 15 |
| Abbildung 25: Webserver Hannover Routing-Tabelle anwenden . . . . .            | 15 |
| Abbildung 26: ifconfig Server Berlin (Routing-Tabelle) . . . . .               | 16 |

|                                                                   |    |
|-------------------------------------------------------------------|----|
| Abbildung 27: Server Berlin Routing-Tabelle anwenden . . . . .    | 16 |
| Abbildung 28: ifconfig Server Hamburg (Routing-Tabelle) . . . . . | 16 |
| Abbildung 29: Server Hamburg Routing-Tabelle anwenden . . . . .   | 17 |
| Abbildung 30: ifconfig Router 1 (Routing-Tabelle) . . . . .       | 17 |
| Abbildung 31: Router 1 Routing-Tabelle anwenden . . . . .         | 18 |
| Abbildung 32: ifconfig Router 2 (Routing-Tabelle) . . . . .       | 18 |
| Abbildung 33: Router 2 Routing-Tabelle anwenden . . . . .         | 18 |
| Abbildung 34: ifconfig Router 3 (Routing-Tabelle) . . . . .       | 19 |
| Abbildung 35: Router 3 Routing-Tabelle anwenden . . . . .         | 19 |
| Abbildung 36: Netz: 10.88.40.32/27 . . . . .                      | 20 |
| Abbildung 37: Netz: 10.88.40.64/27 . . . . .                      | 21 |
| Abbildung 38: Netz: 10.88.40.96/27 . . . . .                      | 21 |
| Abbildung 39: Netz: 10.88.40.128/27 . . . . .                     | 22 |
| Abbildung 40: Konfig. Server München mit ip addr . . . . .        | 24 |
| Abbildung 41: Konfig. Router 3 mit ip addr . . . . .              | 24 |
| Abbildung 42: Konfig. Router 3 mit nmcli . . . . .                | 25 |
| Abbildung 43: Konfig. Router 2 mit ip addr . . . . .              | 25 |
| Abbildung 44: Konfig. Router 2 mit nmcli . . . . .                | 26 |
| Abbildung 45: Konfig. Router 1 mit ip addr . . . . .              | 26 |
| Abbildung 46: Konfig. Router 1 mit nmcli . . . . .                | 26 |
| Abbildung 47: Konfig. Webserver Hannover mit ip addr . . . . .    | 27 |
| Abbildung 48: Konfig. Server Berlin mit nmcli . . . . .           | 27 |
| Abbildung 49: Konfig. Server Hamburg mit ip addr . . . . .        | 28 |
| Abbildung 50: figure 1 LAN . . . . .                              | 44 |
| Abbildung 51: Zenmap Subnetz 64 . . . . .                         | 45 |
| Abbildung 52: Zenmap Subnetz 96 . . . . .                         | 46 |
| Abbildung 53: Zenmap Subnetz 128 . . . . .                        | 46 |
| Abbildung 54: Zenmap Subnetz 160 . . . . .                        | 47 |
| Abbildung 55: Zenmap Subnetz 32 . . . . .                         | 47 |
| Abbildung 56: Zenmap alle Subnetze . . . . .                      | 48 |
| Abbildung 57: nmap command 1 . . . . .                            | 49 |
| Abbildung 58: nmap command 2 . . . . .                            | 50 |
| Abbildung 59: nmap externes Logfile . . . . .                     | 50 |

|                                                                       |    |
|-----------------------------------------------------------------------|----|
| Abbildung 60: nmap command 3 . . . . .                                | 51 |
| Abbildung 61: nmap command 4 . . . . .                                | 51 |
| Abbildung 62: nmap command 5 . . . . .                                | 52 |
| Abbildung 63: nmap command 6 . . . . .                                | 52 |
| Abbildung 64: nmap command 7 . . . . .                                | 52 |
| Abbildung 65: nmap command 8 . . . . .                                | 53 |
| Abbildung 66: nmap command 9 . . . . .                                | 53 |
| Abbildung 67: nmap command 10 . . . . .                               | 54 |
| Abbildung 68: installation Nessus . . . . .                           | 56 |
| Abbildung 69: Konfiguration Nessus . . . . .                          | 58 |
| Abbildung 70: Registrierung Nessus . . . . .                          | 58 |
| Abbildung 71: Aktivierung Nessus . . . . .                            | 59 |
| Abbildung 72: License Nessus . . . . .                                | 59 |
| Abbildung 73: Scan Hosts in Subnetzt 10.88.40.32/27 . . . . .         | 62 |
| Abbildung 74: Ergebnis des Scans im Subnetz 10.88.40.32/27 . . . . .  | 62 |
| Abbildung 75: Scan Hosts in Subnetzt 10.88.40.64/27 . . . . .         | 63 |
| Abbildung 76: Ergebnis des Scans im Subnetz 10.88.40.64/27 . . . . .  | 63 |
| Abbildung 77: Scan Hosts in Subnetzt 10.88.40.96/27 . . . . .         | 64 |
| Abbildung 78: Ergebnis des Scans im Subnetz 10.88.40.96/27 . . . . .  | 64 |
| Abbildung 79: Scan Hosts in Subnetzt 10.88.40.128/27 . . . . .        | 65 |
| Abbildung 80: Ergebnis des Scans im Subnetz 10.88.40.128/27 . . . . . | 65 |
| Abbildung 81: Scan Hosts in Subnetzt 10.88.40.160/27 . . . . .        | 66 |
| Abbildung 82: Ergebnis des Scans im Subnetz 10.88.40.160/27 . . . . . | 66 |
| Abbildung 83: Ergebnis des Scans aller Subnetze . . . . .             | 67 |
| Abbildung 84: Netzwerkkonfiguration über die GUI . . . . .            | 68 |
| Abbildung 85: erfolgreiche Internetverbindung . . . . .               | 69 |
| Abbildung 86: apt-get update . . . . .                                | 74 |
| Abbildung 87: apt-get upgrade . . . . .                               | 74 |
| Abbildung 88: OpenVas Installation . . . . .                          | 74 |
| Abbildung 89: OpenVas Konfiguration . . . . .                         | 75 |
| Abbildung 90: OpenVas user anlegen . . . . .                          | 75 |
| Abbildung 91: OpenVas Login . . . . .                                 | 76 |
| Abbildung 92: OpenVas Scan des Subnetzes 10.88.40.32/27 . . . . .     | 77 |

|                                                                    |     |
|--------------------------------------------------------------------|-----|
| Abbildung 93: OpenVas Scan des Subnetzes 10.88.40.64/27 . . . . .  | 77  |
| Abbildung 94: OpenVas Scan des Subnetzes 10.88.40.96/27 . . . . .  | 78  |
| Abbildung 95: OpenVas Scan des Subnetzes 10.88.40.128/27 . . . . . | 78  |
| Abbildung 96: OpenVas Scan des Subnetzes 10.88.40.160/27 . . . . . | 79  |
| Abbildung 97: OpenVas Scan alle Subnetze . . . . .                 | 80  |
| Abbildung 98: Figur 1 . . . . .                                    | 82  |
| Abbildung 99: Figur 2 . . . . .                                    | 83  |
| Abbildung 100: install nmap-frontend . . . . .                     | 84  |
| Abbildung 101: install nmap-frontend 2 . . . . .                   | 84  |
| Abbildung 102: pnid4-mid-hh nach pnid4-cnt-bln . . . . .           | 85  |
| Abbildung 103: pnid4-mid-hh nach pnid4-cnt-bln 2 . . . . .         | 85  |
| Abbildung 104: pnid4-cnt-bln nach pnid4-mid-hh . . . . .           | 86  |
| Abbildung 105: pnid4-cnt-bln nach pnid4-mid-hh 2 . . . . .         | 86  |
| Abbildung 106: installation wireshark . . . . .                    | 88  |
| Abbildung 107: installation libsmi . . . . .                       | 89  |
| Abbildung 108: installation httpd . . . . .                        | 89  |
| Abbildung 109: installation httpd 2 . . . . .                      | 89  |
| Abbildung 110: installation tcpdump . . . . .                      | 90  |
| Abbildung 111: installation telnet . . . . .                       | 90  |
| Abbildung 112: installation wireshark . . . . .                    | 90  |
| Abbildung 113: Apache aktivieren . . . . .                         | 91  |
| Abbildung 114: Verbindung zum http Server . . . . .                | 91  |
| Abbildung 115: Wireshark Analyse . . . . .                         | 92  |
| Abbildung 116 Tcpdump Analyse . . . . .                            | 92  |
| Abbildung 117: tcpdump -i ens33 . . . . .                          | 93  |
| Abbildung 118: tcpdump -c 12 -i ens33 . . . . .                    | 93  |
| Abbildung 119: tcpdump -c 12 -i ens33 . . . . .                    | 94  |
| Abbildung 120: Telnet mid-hh to cnt-bln . . . . .                  | 97  |
| Abbildung 121: tcpdump -vvv . . . . .                              | 98  |
| Abbildung 122: telnet trump4 . . . . .                             | 98  |
| Abbildung 123: pnidX-mid-hh-dump.1 . . . . .                       | 98  |
| Abbildung 124: ssh 10.88.40.37 . . . . .                           | 99  |
| Abbildung 125: pnidX-mid-hh-dump.2 . . . . .                       | 100 |

|                                                                     |     |
|---------------------------------------------------------------------|-----|
| Abbildung 126: pnidX-mid-hh-dump.2 . . . . .                        | 100 |
| Abbildung 127: pnidX-mid-hh-dump.2 . . . . .                        | 101 |
| Abbildung 128: pnidX-mid-hh-dump.3 . . . . .                        | 101 |
| Abbildung 129: Wireshark Filter 1 . . . . .                         | 103 |
| Abbildung 130: Wireshark Filter 2 . . . . .                         | 104 |
| Abbildung 131: Wireshark Filter 3 . . . . .                         | 104 |
| Abbildung 132: Wireshark Filter 4 . . . . .                         | 105 |
| Abbildung 133: Wireshark Filter 5 . . . . .                         | 105 |
| Abbildung 134: Wireshark Filter 7 . . . . .                         | 106 |
| Abbildung 135: Wireshark Filter 8 . . . . .                         | 107 |
| Abbildung 136: Wireshark ftp Login Passwort . . . . .               | 108 |
| Abbildung 137: Wireshark Filter 9 . . . . .                         | 109 |
| Abbildung 138: Wireshark ssh Datenpaket . . . . .                   | 109 |
| Abbildung 139: nmap offene Ports anzeigen . . . . .                 | 110 |
| Abbildung 140: Telnet login . . . . .                               | 110 |
| Abbildung 141: Installation Openswan . . . . .                      | 112 |
| Abbildung 142: Figure 1: HOST-TO-HOST-VPN . . . . .                 | 113 |
| Abbildung 143: dst 10.88.40.130 && tcp && port 80 . . . . .         | 114 |
| Abbildung 144: http://10.88.40.130 . . . . .                        | 114 |
| Abbildung 145: Filter: dst 10.88.40.130 && tcp && port 80 . . . . . | 115 |
| Abbildung 146: ipsec ranbits 256 > . . . . .                        | 115 |
| Abbildung 147: psk.secrets . . . . .                                | 115 |
| Abbildung 148: psk.conf . . . . .                                   | 116 |
| Abbildung 149: ipsec setup reload . . . . .                         | 118 |
| Abbildung 150: ipsec auto –add . . . . .                            | 118 |
| Abbildung 151: ipsec auto –up . . . . .                             | 118 |
| Abbildung 152: iptables rules . . . . .                             | 119 |
| Abbildung 153: /etc/ipsec.secrets . . . . .                         | 121 |
| Abbildung 154: /etc/ipsec.conf . . . . .                            | 122 |
| Abbildung 155: /etc/ipsec_iptables . . . . .                        | 122 |
| Abbildung 156: ipsec_ifconfig.dump . . . . .                        | 123 |
| Abbildung 157: ipsec_look.dump . . . . .                            | 123 |
| Abbildung 158: ipsec_route.dump . . . . .                           | 123 |

|                                                               |     |
|---------------------------------------------------------------|-----|
| Abbildung 159: /etc/ipsec.secrets . . . . .                   | 124 |
| Abbildung 160: /etc/ipsec.conf . . . . .                      | 124 |
| Abbildung 161: /etc/ipsec_iptables . . . . .                  | 125 |
| Abbildung 162: ipsec_ifconfig.dump . . . . .                  | 125 |
| Abbildung 163: ipsec_look.dump . . . . .                      | 126 |
| Abbildung 164: ipsec_route.dump . . . . .                     | 126 |
| Abbildung 165: dst 10.88.40.130 && tcp port 80 . . . . .      | 127 |
| Abbildung 166: Berlin: certutil -N -d /etc/ipsec.d . . . . .  | 128 |
| Abbildung 167: Hamburg: certutil -N -d /etc/ipsec.d . . . . . | 128 |
| Abbildung 168: ipsec newhostkey 1 . . . . .                   | 129 |
| Abbildung 169: ipsec newhostkey 2 . . . . .                   | 129 |
| Abbildung 170: rsa.conf . . . . .                             | 129 |
| Abbildung 171: ipsec setup reload . . . . .                   | 130 |
| Abbildung 172: ipsec setup reload . . . . .                   | 130 |
| Abbildung 173: ipsec auto -- add rsa . . . . .                | 130 |
| Abbildung 174: ipsec auto -- up rsa . . . . .                 | 130 |
| Abbildung 175: /etc/ipsec.secrets . . . . .                   | 132 |
| Abbildung 176: /etc/ipsec.conf . . . . .                      | 133 |
| Abbildung 177: /etc/ipsec_iptables . . . . .                  | 133 |
| Abbildung 178: ipsec_ifconfig.dump . . . . .                  | 134 |
| Abbildung 179: ipsec_look.dump . . . . .                      | 134 |
| Abbildung 180: ipsec_route.dump . . . . .                     | 135 |
| Abbildung 181: /etc/ipsec.secrets . . . . .                   | 135 |
| Abbildung 182: /etc/ipsec.conf . . . . .                      | 136 |
| Abbildung 183: /etc/ipsec_iptables . . . . .                  | 136 |
| Abbildung 184: ipsec_ifconfig.dump . . . . .                  | 137 |
| Abbildung 185: ipsec_look.dump . . . . .                      | 137 |
| Abbildung 186: ipsec_route.dump . . . . .                     | 138 |
| Abbildung 187: ping-Ergebnis . . . . .                        | 139 |
| Abbildung 188: zenmap for testing the connectivity . . . . .  | 139 |
| Abbildung 189: net-to-net-psk.secrets . . . . .               | 141 |
| Abbildung 190: net-to-net-psk.conf . . . . .                  | 141 |
| Abbildung P5 creating iptables rules . . . . .                | 144 |

|                                                        |     |
|--------------------------------------------------------|-----|
| Abbildung 191: /etc/ipsec.secrets . . . . .            | 145 |
| Abbildung 192: /etc/ipsec.conf . . . . .               | 146 |
| Abbildung 193: /etc/ipsec_iptables . . . . .           | 146 |
| Abbildung 194: ipsec_ifconfig.dump . . . . .           | 147 |
| Abbildung 195: ipsec_look.dump . . . . .               | 147 |
| Abbildung 196: ipsec_route.dump . . . . .              | 148 |
| Abbildung 197: /etc/ipsec.secrets . . . . .            | 148 |
| Abbildung 198: /etc/ipsec.conf . . . . .               | 149 |
| Abbildung 199: /etc/ipsec_iptables . . . . .           | 149 |
| Abbildung 200: ipsec_ifconfig.dump . . . . .           | 150 |
| Abbildung 201: ipsec_look.dump . . . . .               | 150 |
| Abbildung 202: ipsec_route.dump . . . . .              | 151 |
| Abbildung P5 Berlin: net-to-net-rsa.secrets . . . . .  | 155 |
| Abbildung P5 Hamburg: net-to-net-rsa.secrets . . . . . | 155 |
| Abbildung P5 net-to-net-rsa.conf . . . . .             | 156 |
| Abbildung 203: /etc/ipsec.secrets . . . . .            | 158 |
| Abbildung 204: /etc/ipsec.conf . . . . .               | 158 |
| Abbildung 205: /etc/ipsec_iptables . . . . .           | 159 |
| Abbildung 206: ipsec_ifconfig.dump . . . . .           | 159 |
| Abbildung 207: ipsec_look.dump . . . . .               | 160 |
| Abbildung 208: ipsec_route.dump . . . . .              | 160 |
| Abbildung 209: /etc/ipsec.secrets . . . . .            | 161 |
| Abbildung 210: /etc/ipsec.conf . . . . .               | 161 |
| Abbildung 211: /etc/ipsec_iptables . . . . .           | 161 |
| Abbildung 212: ipsec_ifconfig.dump . . . . .           | 162 |
| Abbildung 213: ipsec_look.dump . . . . .               | 162 |
| Abbildung 214: ipsec_route.dump . . . . .              | 163 |