Table of Contents

Linux Security

Jochen Metzger
(3-4 days)

Nmap

Lab 1: Nmap - Install

```
# Ubuntu / Debian
sudo apt-get install nmap
# Centos / Redhat
yum install nmap
```

Nmap - Some basics

```
* Nmap needs root-privileges to scan properly
* By default it scans IPv4
* -6 to scan IPv6
* -O: Enable OS detection
```

Nmap - Using predefined Timing Templates

- -T0 = paranoid
- -T1 = sneaky (tries to fool IDS)
- -T2 = polite (tries to fool IDS)
- -T3 = Does not change the timing (Default)
- -T4 = aggressive (accelerates scans)
- -T5 = insane (quickest option, less accuracy, o.k ?)

Nmap - Scan for the most common ports

- By default nmap scans for the most 1000 common ports
- -F = scan for the most 100 common ports

Lab 2: Scanning a specific IP/Host

```
# Detect OS
sudo nmap -O <target_ip/target_hostname>
# -A (OS-Detection + Version Detection + Script Scanning + Traceroute
# -T4
sudo nmap -A -F -T4 <target-ip>
```

Nmap - Scan a network

- -sP does a simple scan on the network by pinging
- example:

```
nmap -sP 192.168.1.0/24
```

OpenVAS

OpenVAS - History/Basics

- Nessus went non-Open Source
- A fork was created → OpenVAS
- Uses the same plugins as nessus
- Client / Server architecture

OpenVAS - CVE

- CVE = Common Vulnerabilities and Exposures (CVE)

OpenVAS 9 - Documentation

- Greenbone OS 4 corresponds with OpenVAS-9
- Greenbone OS 3.1 corresponds to OpenVAS-8.
- Ref: http://docs.greenbone.net/#user_documentation

OpenVAS9 - Debian Jessie

```
https://avleonov.com/2017/04/10/installing-openvas-9-from-the-sources/
wget
https://raw.githubusercontent.com/leonov-av/openvas-commander/master/openvas_commander.sh
apt install curl
chmod +x openvas_commander.sh

# Important. This script does not work on Debian Stretch
```

OpenVAS8 - Debian Stretch

```
http://enricorossi.org/blog/2017/OpenVAS_scanner_on_Debian_Stretch/

openvas-mkcert
# -ni is important /
# The error you get is not problem !!!
openvas-mkcert-client -ni
```

```
# change port of manager
# edit it after listen
systemctl edit .service
systemctl restart openvas-manager.service
# see if the right configuration was loaded
# you should see something like /etc/systemd/system in the first line
#[Service]
#ExecStart=
#ExecStart=.... put your new version here
systemctl status openvas-manager.service
```

OpenVAS - Install (Ubuntu 16.04)

- One way is to work with OpenVAS using a virtual machine
  e.g. virtualbox
- Another way is to install it on your OS (e.g. Linux Ubuntu 16.04)
- https://www.vultr.com/docs/how-to-install-openvas-vulnerability-scanner-on-ubuntu-16-04

Lab 3: Install OpenVAS9 (Ubuntu 16.04)

- Follow the install steps and install OpenVAS (Version for Ubuntu 16.04)

```
sudo apt-get update -y
sudo apt-get upgrade -y
# reboot, because we have some kernel changes
sudo reboot

# install prerequisites
sudo apt-get install python-software-properties
sudo apt-get install sqlite3

sudo add-apt-repository ppa:mrazavi/openvas
# Update the repository.
sudo apt-get update

# Install openvas9
sudo apt-get install openvas9

#
sudo greenbone-nvt-sync
sudo greenbone-scapdata-sync
sudo greenbone-certdata-sync

sudo service openvas-scanner restart
sudo service openvas-manager restart
sudo openvasmd --migrate #only required when upgrading from an older
version
sudo openvasmd --rebuild --progress

# Enable pdf-reports
```

```
sudo apt-get install texlive-latex-extra --no-install-recommends
# Without that, there will be no fonts in pdf-document !!
sudo apt-get install texlive-fonts-recommended

# You can now navigate to the interface
# https://localhost:4000
# user: admin
# pass: admin
```

OpenVAS - Install (Ubuntu 18.04)

- Hurrah ! OpenVAS is within the Ubuntu - Repo by default

Installation (Ubuntu 18.04)

```
apt update
# about 150 packages !!
apt install openvas
# Download the signatures
# If you go there with the same ip + multiple people
# the second try is blocked
greenbone-nvt-sync
greenbone-scapdata-sync
greenbone-certdata-sync
# now starting
systemctl start openvas-scanner
```

OpenVAS - Change Port

```
You can change the web interface port number by modifying
/etc/default/openvas-gsa. Then, restart its service by issuing "sudo service
openvas-gsa restart".
```

Lab: Install OpenVAS 9 on Debian Stretch (from Linux Kali)

```
# Install killall command / use for kali installation
apt install psmisc

# /etc/apt/sources.list
deb http://http.kali.org/kali kali-rolling main

apt-get update
apt-get install -t kali-rolling openvas

# 4. Run the following command to configure the OpenVAS and to download the
initial database:
openvas-setup
```

```
# Finally, open a web browser and access the address https://127.0.0.1:9392
(use https!!!).

# Create new administrative user (one is already created with openvas-setup
openvasmd --create-user test --role Admin

# Update nvt-database
openvasmd --update
openvasmd --rebuild
service openvas-scanner restart

# To solve the message "Login failed. Waiting for OMP service to become
available":
openvas-start

# Ref:
http://collab.debian.net/portal/planet-debian/eriberto-mota-openvas-9-from-k
ali-linux-2017.1-to-debian-9
```

OpenVAS 9 (Kali/Debian Stretch) - Change Interface

```
# /etc/default/greenbone-security-assistant
export SYSTEMD_EDITOR=vi
systemctl edit greenbone-security-assistant.service
# Enter the following data
[Service]
ExecStart=
ExecStart=ExecStart=/usr/sbin/gsad --foreground --listen=0.0.0.0 --port=9392
--mlisten=127.0.0.1 --mport=9390

#
systemctl restart greenbone-security-assistant.service
```

OpenVAS 9 - Services running

- ```
  ps aux | grep openvas
  root       677  2.0 18.2 260196 92200 ?        SL   18:18   0:01
  openvasmd
  root      1222 46.2  2.4 133300 12352 ?        Ds   18:19   0:07
  openvassd: Reloaded 26550 of 56798 NVTs (46% / ETA: 00:15)
  root      1223  0.0  0.3 125880  1520 ?        S    18:19   0:00
  openvassd (Loading Handler)
  root      1243  0.0  0.1  12720   944 pts/0    S+   18:19   0:00 grep
  openvas
  ```

- ```
  # gsad = greenbone security assistant daemon
  # That's the webinterface
  ```

```
ps aux | grep gsad
```

OpenVAS 9 - Parts

- gsad = greenbone securtiy assistant daemon
- openvasmd = openvas management daemon
- openvassd = openvas scanner daemon

OpenVAS 9 - Logs

- Directory: /var/log/openvas/
- gsad.log = Log of Webinterface
- openvasmd.log = log of management daemon
- openvassd.messages = log of scanner

OpenVAS 9 - User Management

- Change Passwort of user:

```
openvasmd --user=test2 --new-password=11dortmund22
```

- Delete user:

```
openvasmd --delete-user=test2
```

OpenVAS - Install nasl utiltity

```
To install openvas-nasl utility:
sudo apt-get install libopenvas9-dev

# Why ?
# This make it possible to debug and use nasl - scripts
# NASL = Nessus Attack Scripting Language

# Usage: Example
# -T Trace output  -t <target-ip's>
openvas-nasl -T -t 127.0.0.1 /var/lib/openvas/plugins/ping_host.nasl

#
http://www.openvas.org/trusted-nvts.html
```

OpenVAS - OpenVAS Manager

- CLI - Interface to manager openvas
- openvasmd –help

OpenVAS - Create new admin user + password

- openvasmd –create-user=admin –role=Admin
- openvasmd –user=admin –new-password=NewPW

Lab 4: OpenVAS 9 - check installation

```
sudo su
cd /usr/local/bin
wget --no-check-certificate
https://svn.wald.intevation.org/svn/openvas/trunk/tools/openvas-check-setup
chmod u+x openvas-check-setup
openvas-check-setup --v9
exit
```

OpenVAS - references

- https://hackertarget.com/openvas-9-install-ubuntu-1604/

OpenVAS - Management GUI

- For creating a new role/schedule a.s.o click on the “*” on the left
- e.g. Configuration → Schedule → *

OpenVAS - Create new scan

- Easiest way is to use the wizard
  - Scan → Task → Click on the wizard icon on the left of '*'

tcpdump -> scan environment while scanning

- ```
  # See what happens trafficwise
  apt install tcpdump
  # find out the interface with 'ip addr'
  tcpdump -i eth0
  # filter specific entry
  tcpdump -i eth0 not ssh
  ```

OpenVAS hints on scan-types

- Full and Fast
  - Fast = fast & intelligent (vs. slow)
- Slow
  - Throws all the scans on a target (no matter if they fit or not)
  - e.g. Testing Shellshock exploits on a Windows SMB Port
- Full and fast ultimate
  - contains scans that can crash the target system
  ```

OpenVAS - alerts

- Configuration → Alerts
- Trigger after e.g. Task has run with certain conditions
- Different actions can be triggered:
  - Upload report by scp
  - Open an url (GET)
  - Send Email
  - Trigger a new (different scan)

OpenVAS - Delta Reports

- OpenVAS makes it possible to generate delta-reports
- These delta reports only show the differences since the last scan
- Ref: http://docs.greenbone.net/GSM-Manual/gos-4/en/reports.html#delta-reports

OpenVAS - Delta Reports (Howto)

- Start the same task once again (Scans → Tasks → Task in List → Start (>) - Button
- After the task has finished click
  - In Reports → Total → Click on the number (after 2 scans → 2)
  - Now you will see 2 reports
  - In the line of the first report → in Actions (column) → click on ^ (Blue background = Delta Icon)
  - Now the clicked icon (^) is greyed out
  - Now click on the (^ - icon / actually to ^^ icons ;o) in the line (report) you want to compare
- You will now see → Report: Delta Results
  - Now either:
    - Click on the vulnerabilities one by one
    - or: Create a report (on top) of the page (next to ? sign)

arp

arpwatch - Install

```
sudo apt install arpwatch
```

basic arpwatch - usage

```
arpwatch -i eth0
#
# You will notice syslog entries as follows /var/log/syslog file (or
/var/log/message file) when #changes are made i.e MAC/IP address pair is
changed:
tail -f /var/log/syslog
```

arpwatch - howto

https://tournasdimitrios1.wordpress.com/2011/01/09/how-to-detect-arp-spoofing-under-unix-or-linux/

arping - spoof mac - addresses

- it tells the destination that it is this source - ip (-S)
- ```
  # on debian stretch
  arping -S 10.10.10.104 10.10.10.122
  # this seems no to work on Ubuntu 16.04
  ```

Intrusion Detection Systems (IDS)

IDS - Types

- Host Based IDS (=HIDS)
- Network Based IDS (=NIDS)

NIDS: Snort

Snort Ecosystem

- Graylog (GUI for showing logs and stats)
  - https://www.graylog.org
- + Ref: https://www.snort.org/downloads → Additional Downloads (Third Party)
  - Pulled Pork (Managing Ruleset, incl. Downloads with OINK - Code)
  - Barnyard (Open Source Interpreter of unified2 - logs)
    - ```
      Barnyard2 is an open source interpreter for Snort unified2
      binary output files.
      Its primary use is allowing Snort to write to disk in an
      efficient manner and
      leaving the task of parsing binary data into various formats
      to a separate
      process that will not cause Snort to miss network traffic.
      ```
  - Snorby (GUI to snort stuff)
  - OpenFPC (Get the complete traffic associated with a network security event and put it in a pcap file) - e.g. An incident on your maillog

Lab 5: Snort - Install

```
# Debian 9 (Debian Stretch)
apt update
apt upgrade
apt install snort
```

```
# check if Snort runs properly
systemctl status snort
```

Snort - Debugging - Startup

- Check /var/log/syslog for errors
- Common Error:

```
Starting Network Intrusion Detection System: snort (eth0 using
/etc/snort/snort.conffork: Cannot allocate memory
```

- This might happen on systems, that have no swap - partition

Snort - Create swap - file

- **Snort needs a swap-partition or swap-file to work properly**

```
# locate about the size of the memory
fallocate -l 1G /swapfile
ls -lh /swapfile
# swap will complain about permission otherwice
chmod 600 /swapfile
ls -lh /swapfile
mkswap /swapfile
swapon /swapfile
swapon --show
# after that you can try to restart snort again
# systemctl start snort
```

Snort: Test configuration

```
sudo snort -T -c /etc/snort/snort.conf
```

Snort: Setup configuration for functional test

- We want to set up an icmp ruleset

```
# sudo nano /etc/snort/rules/local.rules
# add
alert icmp any any -> $HOME_NET any (msg:"ICMP test"; sid:10000001;
rev:001;)
```

Snort: Structure of a rule

- action: alert
- protocol: icmp
- source ip: any

- source port: any
- dest ip: $HOME_NET
- dest port: any
- msg: message to log
- sid: unique rule identifier
    - **1000001 or higher**
- rev: rule version number

Log files activate

- After changing output directives “output” in /etc/snort/snort.conf we need to restart snort
- systemctl restart snort
- log files are in unified2 - format
- and can be read with
    - u2spewfoo /var/log/snort/snort.log

HIDS: Tripwire

Lab: Tripwire Install

```
# Debian
apt install tripwire
# Answer the questions as follows:
# Site
# Schlüssel erzeugen -> Ja
# Lokalen Schlüssel erzeugen -> Ja
# Tripwire - Konfigurationsdatei erzeugen -> Ja
# Policies -> Ja
```

Tripwire - What is where ?

- Binaries: /usr/sbin
- Database: /var/lib/tripwire

Tripewire - Keys

- site key : Secure configuration files (may not be modified)
- local key: Protect binary files

Tripwire - configuration file

- If you have not created that during installation

```
# Creates encrypted twpol - file
sudo twadmin --create-polfile /etc/tripwire/twpol.txt
# create database
sudo tripwire --init
```

Tripwire - check (document)

- We want to document what gets scanned

```
tripwire --check | grep Filename > test_results'
#If we view this file, we should see entries that look like this:
less /etc/tripwire/test_results
# ...
Filename: /etc/rc.boot
Filename: /root/mail
Filename: /root/Mail
Filename: /root/.xsession-errors
```

Tripwire - adjust twpol.txt

```
# replace /proc by /proc/devices
# was:
#/proc                   -> $(Device) ;
# now
/proc/devices           -> $(Device) ;

# remove all /root/* entries that are not present
# e.g.
# /root/.sawfish

# uncomment /var/lock and /var/run
#/var/lock              -> $(SEC_CONFIG) ;
#/var/run               -> $(SEC_CONFIG) ; # daemon PIDs
```

Tripwire - recreate pol file + re-init db

```
# polfile
sudo twadmin -m P /etc/tripwire/twpol.txt
# re-init database
sudo tripwire --init
```

Tripwire - rerun check

```
sudo tripwire --check
```

Tripwire - remove sensitive information

```
sudo rm /etc/tripwire/test_results
sudo rm /etc/tripwire/twpol.txt
# recreate it
sudo twadmin --print-polfile > /etc/tripwire/twpol.txt
```

```
sudo rm /etc/tripwire/twpol.txt
```

OpenSSH

Hardening the SSH-Server

```
Port 22
Protocol 2
AllowUsers user1 user2
LoginGraceTime 2m
PermitRootLogin no
StrictModes yes
MaxAuthTries 1
PubkeyAuthentication yes
# no authentication by ip only
RHostsAuthentication no
PasswordAuthentication no
PermitEmptyPasswords no
```

OpenSSH: About security

```
It is now well-known that (some) SSH sessions can be decrypted
(potentially in real time) by an adversary with sufficient resources.
SSH best practice has changed in the years since the protocols were
developed,
and what was reasonably secure in the past is now entirely unsafe.
```

OpenSSH: More about security

- The server and the client choose a set of algorithms supported by both,
  - then proceed with the key exchange.
- The key exchange ensures that the server and the client shares a secret no one else knows.
  - We also have to make sure that they share this secret with each other and not an NSA analyst
- Some of the supported algorithms are not so great and should be disabled completely.
- Ref: https://stribika.github.io/2015/01/04/secure-secure-shell.html

OpenSSH: Kex-algorithms, Ciphers, HMACS

- Kex-algorithms:
  - the key exchange methods that are used to generate per-connection keys
- Ciphers:
  - the ciphers to encrypt the connection
- HMACs:
  - the message authentication codes used to detect traffic modification
- PubkeyAcceptedKeyTypes
  - the public key algorithms that the server can use to authenticate itself to the client

OpenSSH: Key-Exchange-Protocols

- In general there are 2 safe methods:
  - Diffie Hellman and Elliptic Curve Diffie Hellmann
- OpenSSH supports 11 Key Exchange Protocols (as of 2015)
  - (12 including @libssh.org)

```
(1) curve25519-sha256: ECDH over Curve25519 with SHA2
(2) diffie-hellman-group1-sha1: 1024 bit DH with SHA1
(3) diffie-hellman-group14-sha1: 2048 bit DH with SHA1
(4) diffie-hellman-group14-sha256: 2048 bit DH with SHA2
(5) diffie-hellman-group16-sha512: 4096 bit DH with SHA2
(6) diffie-hellman-group18-sha512: 8192 bit DH with SHA2
(7) diffie-hellman-group-exchange-sha1: Custom DH with SHA1
(8) diffie-hellman-group-exchange-sha256: Custom DH with SHA2
(9) ecdh-sha2-nistp256: ECDH over NIST P-256 with SHA2
(10) ecdh-sha2-nistp384: ECDH over NIST P-384 with SHA2
(11) ecdh-sha2-nistp521: ECDH over NIST P-521 with SHA2
```

OpenSSH - Key-Exchange Considerations

- We have to look at 3 things here
- ECDH curve choice:
  - This eliminates 9-11 because NIST curves suck.
  - They leak secrets through timing side channels and off-curve inputs.
  - Also, NIST is considered harmful and cannot be trusted.
- Bit size of the DH modulus:
  - This eliminates 2 because the NSA has supercomputers and possibly unknown attacks.
  - 1024 bits simply don’t offer sufficient security margin.
- Security of the hash function:
  - This eliminates 2, 3, and 7 because SHA1 is broken.
    - We don’t have to wait for a second preimage attack that takes 10 minutes on a cellphone to disable it right now.
- We are left with 1 and 8,
  - as well as 4-6 which were added in OpenSSH 7.3.
  - 1 is better and it’s perfectly OK to only support that
  - but for interoperability (with Eclipse, WinSCP), 8 can be included.

OpenSSH: Secure Kex-algorithms

- In the first phase the both side create a “number” based
  - on the diffie-hellmann - process
  - (=Exchanging a safe key, over an insecure channel)
- After that this key is hashed
  - That's the last part in the Kex-algorithm

```
# Safe on Debian Jessie, should work on Debian Stretch too
KexAlgorithms curve25519-sha256@libssh.org,diffie-hellman-group-
exchange-sha256
```

- See here for other ssh-services: https://github.com/stribika/stribika.github.io/wiki/Secure-Secure-Shell

OpenSSH: Ciphers (I)

- Symmetric ciphers are used to encrypt the data
  - after the initial key exchange and authentication is complete.

```
 # there are quite some ciphers
1. 3des-cbc
2. aes128-cbc
3. aes192-cbc
4. aes256-cbc
5. aes128-ctr
6. aes192-ctr
7. aes256-ctr
8. aes128-gcm@openssh.com
9. aes256-gcm@openssh.com
10. arcfour
11. arcfour128
12. arcfour256
13. blowfish-cbc
14. cast128-cbc
15. chacha20-poly1305@openssh.com
```

OpenSSH - Cipher considerations

- Security of the cipher algorithm:
  - This eliminates 1 and 10-12 →
  - both DES and RC4 are broken.
  - Again, no need to wait for them to become even weaker, disable them now.
- Key size: At least 128 bits, the more the better.
- Block size: At least 128 bits.
  - This eliminates 13 and 14 because those have a 64 bit block size.
- Cipher mode: The recommended approach here is to prefer AE modes
  - and optionally allow CTR for compatibility. CTR with Encrypt-then-MAC is provably secure.

OpenSSH - Ciphers - safe settings

```
Ciphers chacha20-poly1305@openssh.com,aes256-gcm@openssh.com,aes128-
gcm@openssh.com,aes256-ctr,aes192-ctr,aes128-ctr
```

OpenSSH - HMAC/MAC

- What is MAC/HMAC for ?
  - Each message that is sent after the encryption is negotiated must contain a MAC
  - so that the other party can verify the packet integrity.
  - The MAC is calculated from
    - the symmetrical shared secret

- the packet sequence number of the message
- and the actual message content.

OpenSSH - Encrypt/HMAC

- MAC = Message authentication codes
- Two ways:
  - First encrypt then mac
  - First mac then encrypt
- Best: First encrypt then mac
- Why ?
  - Using MAC-then-encrypt have lead to many attacks on TLS
  - while Encrypt-and-MAC have lead to not quite that many attacks on SSH
- Good: SSH uses Encrypt/HMAC by default

OpenSSH - available HMACs

- Possible hmacs

```
1. hmac-md5
2. hmac-md5-96
3. hmac-sha1
4. hmac-sha1-96
5. hmac-sha2-256
6. hmac-sha2-512
7. umac-64
8. umac-128
9. hmac-md5-etm@openssh.com
10. hmac-md5-96-etm@openssh.com
11. hmac-sha1-etm@openssh.com
12. hmac-sha1-96-etm@openssh.com
13. hmac-sha2-256-etm@openssh.com
14. hmac-sha2-512-etm@openssh.com
15. umac-64-etm@openssh.com
16. umac-128-etm@openssh.com
```

OpenSSH - HMAC considerations

- Security of the hash algorithm:
  - No MD5 and SHA1.
  - Tag size: At least 128 bits. This eliminates umac-64-etm.
  - Key size: At least 128 bits. This doesn’t eliminate anything at this point.

OpenSSH - HMAC secure settings

```
MACs hmac-sha2-512-etm@openssh.com,hmac-sha2-256-
etm@openssh.com,umac-128-etm@openssh.com,hmac-sha2-512,hmac-
sha2-256,umac-128@openssh.com
```

OpenSSH: List different Settings

- Settings for the client, when it connects to another server
- Connection is not done !!!
- checks /etc/ssh_config ~/.ssh/config

```
ssh -Q cipher       # List supported ciphers
ssh -Q mac          # List supported MACs
ssh -Q key          # List supported public key types
ssh -Q kex          # List supported key exchange algorithms
```

OpenSSH: List all settings after connecting

```
# Lists all settings, including cipher, mac a.s.o.
ssh -G foo@server
```

OpenSSH: Supported Kex-algorithms for client-side

- Important: This command does **NOT !!** connect to another server
- It just checks ssh_config and ~/.ssh/config

```
ssh -Q kex server

diffie-hellman-group1-sha1
diffie-hellman-group14-sha1
diffie-hellman-group-exchange-sha1
diffie-hellman-group-exchange-sha256
ecdh-sha2-nistp256
ecdh-sha2-nistp384
ecdh-sha2-nistp521
diffie-hellman-group1-sha1
curve25519-sha256@libssh.org
```

Create Pubkey Authentication

```
ssh-keygen -t dsa
scp .ssh/id_dsa.pub >> .ssh/authorized_keys2
# correct permissions
chmod 700 .ssh
chmod 600 .ssh/authorized_keys2
```

Debugging SSH

```
# the more v's the more verbose
ssh -vvv ...
```

Configuring ssh - client

```
# ssh_config configure the general
# behaviour of the ssh - client
# on the system
```

ssh-client (user specific settings)

```
$HOME/.ssh/config
Host github-project1
    User git
    HostName github.com
    IdentityFile ~/.ssh/github.project1.key
Host github-org
    User git
    HostName github.com
    IdentityFile ~/.ssh/github.org.key
Host github.com
    User git
    IdentityFile ~/.ssh/github.key
```

Local ssh - port - forwarding

```
# What is does ?
# -L -> enable Local Port Forwarding
# Open an ssh-connection to user@example.com
# Traffic on local machine
#  -> is tunneled trough ssh
#    -> and redirected to :80 on the other side

# ---- Steps ----
# Step 1: Open tunnel
ssh -L 9000:imgur.com:80 user@example.com

# Step 2: Locally open browser
http://localhost:9000
```

Local Port Forwarding - ssh - hints

```
ssh -L LocalPort:RemoteIP:RemotePort User@RemoteIP
```

ssh - Setting Local Port Forwarding in .ssh/config

```
# .ssh/config
Host tunnel
    HostName database.example.com
    IdentityFile ~/.ssh/coolio.example.key
```

```
    LocalForward 9906 127.0.0.1:3306
    User coolio

# use if with
# -f start in the background
# -N do not execute a command
ssh -f -N tunnel
```

Setting up sftp

How ?

- Use subsystem (added since openssh 6.4) internal-sftp instead
- This includes all necessary files for a chroot environment

Setting sshd_config

- ```
  Subsystem sftp internal-sftp
  And then block other uses:

  Match group sftponly
       ChrootDirectory /home/%u
       X11Forwarding no
       AllowTcpForwarding no
       ForceCommand internal-sftp
  ```

Settings for /home/%u

- The directory is not allowed to be writeable
  by any other user than root

- ```
  adduser bob
  addgroup sftponly
  chmod g-w,o-w /home/bob
  chown root:sftponly /home/bob
  usermod -aG sftponly bob
  ```

Specific settings in user account

- ```
  # not necessary - also work without
  # home directory
  # The most important part are the permissions above
  usermod -d / bob
  # not really needed, but:
  # to be sure user cannot use shell
  ```
  ```

```
usermod -s /usr/bin/nologin bob
```

To debug

- `grep -ir ssh /var/log/*`
- Try to login

```
sftp -vv bob@localhost
```

- This show addtional debug information

ip vs. ifconfig (deprecated)

Working with the command 'ip' (Basics)

- Why ?
  - ip is present on debian strech / centos 7
  - ifconfig not anymore (by default! package: net-tools)
- Syntax:
  - ip [<option>] <object> [<commmand> | help]
  - ip [-force] -batch filename # running batch files to manipulate object

What can the 'ip' - command do ? (Overview)

- Answer: Which interfaces are configured on a system
- Answer: Status of a network interface
- Configure: Network interfaces (including local loop-back, and Ethernet)
- Bring up/down: an interface
- Configure: Both default and static routing
- Configure: Tunnel over IP
- Configure: ARP or NDISC cache entry

ifconfig / ip - cheatsheet

What	**ifconfig**	**ip**
List interfaces	ifconfig	ip a ip addr ip address
add ip address	ifconfig eth0 add 192.168.80.174	ip a add 192.168.80.174 dev eth0
del ip address	ifconfig eth0 del 192.168.80.174	ip a del 192.168.80.174 dev eth0
change communicated Hardware (Ethernet) address	ifconfig eth0 hw ether 00:0c:29:33:4e:aa	ip link set dev eth0 address 00:0c:29:33:4e:aa
change mtu	ifconfig eth0 mtu 2000	ip link set dev eth0 mtu 2000
enable/disable multicast	ifconfig eth0 multicast	ip link set dev eth0 multicast on

What	ifconfig	ip
enable/disable promisc - mode	ifconfig eth0 promisc	ip link set dev eth0 promisc on

Reuse ifconfig/route

- On Debian Stretch / Centos 7
  - ifconfig/route - command is not available
- Install (package is not deprecated, you not there)
  - Debian Stretch:
    - apt install net-tools

ip - list/set route

- ip route show / ip route list
- ip route add default via 192.168.81.1
- sent all packets to the local network 192.168.1.0 directly through → eth0
  - ip route add 192.168.1.0/24 dev eth0
- delete route entry
  - ip route delete 192.168.1.0/24 dev eth0

ip (command) -> tunnel

- http://ask.xmodulo.com/create-gre-tunnel-linux.html

```
## prerequisites on both machines
sudo modprobe ip_gre
lsmod | grep gre

## Machine settings
Host A: 192.168.233.204
Host B: 172.168.10.25

## Machine a
sudo ip tunnel add gre1 mode gre remote 172.168.10.25 local
192.168.233.204 ttl 255
sudo ip link set gre1 up
sudo ip addr add 10.10.10.1/24 dev gre1
# important to set the routing
sudo ip route add 172.168.10/24 dev gre1

# verify the route
ip route show

## Machine b
sudo ip tunnel add gre1 mode gre remote 192.168.233.204 local
172.168.10.25 ttl 255
sudo ip link set gre1 up
sudo ip addr add 10.10.10.2/24 dev gre1
# important to set the routing
sudo ip route add 192.168.233/24 dev gre1
```

```
# Test
ping 10.10.10.2 (from host A)

# Tear down
sudo ip link set gre0 down
sudo ip tunnel del gre0
```

Firewalling: iptables (netfilter)

iptables / default chains

- iptables offers the following builtin chains
- INPUT
- OUTPUT
- FORWARD
- PREROUTING (Inspect packets as soon as they come in (table → nat = -t nat))

iptables - cheatsheet

```
manage chain:
# iptables -N new_chain             // create a chain
# iptables -E new_chain old_chain       // edit a chain
# iptables -X old_chain             // delete a chain

redirecting packet to a user chain:
# iptables -A INPUT -p icmp -j new_chain

listing rules:
# iptables -L                   // list all rules of all tables
# iptables -L -v                // display rules and their counters
# iptables -L -t nat                // display rules for a specific tables
# iptables -L -n --line-numbers         // listing rules with line number
for all tables
# iptables -L INPUT -n --line-numbers       // listing rules with line
number for specific table

manage rules:
# iptables -A chain             // append rules to the bottom of the chain
# iptables -I chain [rulenum]           // insert in chain as rulenum
(default at the top or 1)
# iptables -R chain rulenum         // replace rules with rules specified
for the rulnum
# iptables -D chain rulenum           // delete rules matching rulenum
(default 1)
# iptables -D chain             // delete matching rules

change default policy:
# iptables -P chain target          // change policy on chain to target
```

```
# iptables -P INPUT DROP            // change INPUT table policy to DROP
# iptables -P OUTPUT DROP           // change OUTPUT chain policy to DROP
# iptables -P FORWARD DROP          // change FORWARD chain policy to DROP
```

iptables / netfilter documentation

- https://www.netfilter.org/documentation/

iptables - What traffic on what chains ?

- or: is the forward chain used ?

```
# if 0 bytes .. simply try a ping
iptables -L -v
```

iptables - Match modules (the basics)

- https://www.netfilter.org/documentation/HOWTO/de/netfilter-extensions-HOWTO-3.html

iptables (match module: owner)

- owner
- Act on packages set by a specific user

```
iptables -A OUTPUT -m owner --uid-owner 0 -j LOG
iptables -A OUTPUT -m owner --uid-owner ftp -j DROP
```

- options:
  - –uid-owner (user)
  - –gid-owner (group)
  - –sid-owner (session id)
  - –pid-owner (process id)

iptables (match module: iplimit)

- Limit parallel connections

```
iptables -A INPUT -p tcp --dport http -m iplimit --iplimit-above 4 -j
REJECT
```

iptables (match module: string)

- Filter/React based on packet content data

```
# Prevent typical IIS - Webserver - attack
iptables -A INPUT -p tcp --dport http -m string --string ".exe?/c+tftp"
-j drop
```

iptables (match module: quota)

- Filter based on amount of traffic

```
# Allow traffic up to 50 MB
iptables -A INPUT -p tcp --dport 80 -m quota --quota 52428800 -j ACCEPT
iptables -A INPUT -p tcp --dport 80 -j DROP
```

iptables - work with additional chain / +ipfilter -> limit

```
# Chain to prevent denial of service
# Create syn-flood chain
iptables -t nat -N syn-flood

# Limit 12 connections per second (burst to 24)
# using module 'limit' (-m -> match (extension module limit will be used))
iptables -t nat -A syn-flood -m limit --limit-burst 24 \
  -j RETURN
iptables -t nat -A syn-flood -j DROP

# Enable DDOS - attack
# Variable $EXT_IFACE, $DEST_IP need to be set before
# -p protocol
# -d destination
# -j jump to rule
# --syn ?
# -t -> table / as is nat,filter(default),mangle,raw,security
iptables -t nat -A PREROUTING -i $EXT_IFACE -d $DEST_IP -p tcp --syn -j syn-
flood
```

iptables - send packets -> port 80 -> to -> port 3128 (squid)

```
# $INT_IFACE -> internal interface
# -p protocol
# --dport destination port

iptables -t nat -A PREROUTING -i $INT_IFACE -p tcp --dport 80 \
  -j REDIRECT --to-port 3128
```

OpenVPN

OpenVPN - Installation (Centos 7)

```
sudo su
# enable epel repository
# add this repo to yum
```

```
yum -y install epel-release
# openvpn + easy-rsa (for easy certificate creation)
yum -y install openvpn easy-rsa
```

OpenVPN - --mode

- –mode m
- Set OpenVPN major mode.
- By default, OpenVPN runs in point-to-point mode (“p2p”).
- OpenVPN 2.0 introduces a new mode (“server”)
  which implements a multi-client server capability.

OpenVPN --remote --nobind

- Client stuff
- –remote connect to a remote server (public ip)
- –nobind - do not bind to a specific port
  - relevant for clients only, where the port for sending packages is not relevant
  - important especially if you run a client and a server on the same port

Lab 1: OpenVPN - PreShared - Key / P-to-P - Setup

```
# Server
openvpn --genkey --secret secret.key
# securely transfer the key to the client
# scp secret.key jmetzger@192.168.33.11:/tmp
sudo openvpn --ifconfig 10.200.0.1 10.200.0.2  --dev tun --secret secret.key

# Client
sudo openvpn --ifconfig 10.200.0.2 10.200.0.1  --dev tun --secret secret.key
--remote 192.168.33.10

# Try to ping the other server with 10.200.0.2 / 10.200.0.1
```

Lab 2: OpenVPN - PreShared - Key / P-to-P Setup with TCP

```
# Server
openvpn --genkey --secret secret.key
# securely transfer the key to the client
# scp secret.key jmetzger@192.168.33.11:/tmp
sudo openvpn --ifconfig 10.200.0.1 10.200.0.2  --dev tun --proto tcp-server
--secret secret.key

# Client
sudo openvpn --ifconfig 10.200.0.2 10.200.0.1  --dev tun --proto tcp-client
--secret secret.key --remote 192.168.33.10

# Try to ping the other server with 10.200.0.2 / 10.200.0.1
```

OpenVPN - pre-shared key - Fixing the weak ciphers

```
# message:
# WARNING: INSECURE cipher with block size less than 128 bit (64 bit).
# This allows attacks like SWEET32.
# Mitigate by using a --cipher with a larger block size (e.g. AES-256-CBC)

# Server
sudo openvpn --ifconfig 10.200.0.1 10.200.0.2  --dev tun --proto tcp-server
--secret secret.key --cipher AES-256-CBC
# Client
sudo openvpn --ifconfig 10.200.0.2 10.200.0.1  --dev tun --proto tcp-client
--secret secret.key --cipher AES-256-CBC --remote 192.168.33.10

# try pinging the server or ssh them
```

OpenVPN - static key - How it works ? (I)

- By default: OpenVPN uses 2 keys
    - when setting up point-to-point connection
- a cipher key to encrypt the data of the packages (being exchanged)
- an HMAC - Key to sign packages
- when packages arrive
    - that are not signed with the appropriate HMAC - Key
    - they are dropped immediately
    - FIRST LINE OF DEFENSE
        - against a denial-of-service attack

OpenVPN - Disadvantages static key

- Limited scalability – one client, one server
- Lack of perfect forward secrecy
    - key compromise results in total disclosure of previous sessions
- Secret key must be exchanged using a pre-existing secure channel
- Secret key must exist in plaintext form on each VPN peer

OpenVPN - configuration file

```
* Coniguration options when starting openvpn are...
  * read from left to right
  * and.. top to bottom
* <code>Example
```

openvpn –config client.conf –port 10000 port 10000 will override setting in config BUT: Some settings cannot be overridden</code>

OpenVPN - config that cannot get overridden

- Example

```
# Example
remote openvpnserver.example.com 1194
# this can be written instead of the following 2 configuration-settings
remote openvpnserver.example.com
port 1194
# BUT -> It is seen as connection blocks
# AND -> Ports in connection blocks
# --> CAN NOT
#     --> be overwritten with --port
```

What is perfect forward secrecy ?

- Data cannot get decrypted later
- This is the case for synchronous encryption

OpenVPN - Basics Diffie Hellmann (DHM)

- Diffie Hellmann key exchange is used for OpenVPN
- In 1976 Martin Hellman, Whitfield Diffie and Ralph Merkle
  developed a protocol that allows
  secure information exchange (key) over an insecure channel.
- Key is then used for synchronous encryption.
- There are some numbers exchanged (Server starts with that)
  - g + p (p is a prime number)
  - It is really hard to compute those (cpu-intense)
    - so computing them on each new connecition would be a bad idea
    - so they are pre-computed and stored in the filesystem
    - with easy-rsa (we use that) it is done with ./build-sh

OpenVPN - tcp or udp

- udp faster (no error correction)
- tcp more reliable (but slower), because of error corrections
- **Recommendation: Use udp if you are not experiencing connection problems**

Lab 3a: OpenVPN - copy key-scripts(easy-rsa)

```
mkdir -m 700 /etc/openvpn/training
cd /etc/openvpn/training
cp -a /usr/share/easy-rsa/2.0/* .
```

Lab 3b: OpenVPN - adjust vars - file

- ```
  # /etc/openvpn/training/vars
  # adjust the following lines
  # or just add them add the end of file
  export KEY_COUNTRY="DE"
  export KEY_PROVINCE="BERLIN"
  export KEY_CITY="Berlin"
  export KEY_OU="IT"
  export KEY_ORG="Kathrein"
  export KEY_EMAIL="openvpn@company.de"
  ```

Lab 3c: OpenVPN - Build CA-Certificate

```
cd /etc/openvpn/training/
. ./vars
# cleanup old keys
./clean-all
# we use a stronger certificate
# Answer all the questions with default answer
# !! IMPORTANT: You need to remembers the password !!
KEY_SIZE=4096 ./build-ca --pass
```

Lab 3d: OpenVPN - Build Server-Certificate

```
export KEY_EMAIL=
./build-key-server openvpnserver
# answer all question
# set no password for the certificate itself
# Enter the password for the ca.key
# When ask for signing answer "y"
```

Lab 3e: OpenVPN - Create all the client certificates

- Create one certificate for every client
  - use different name for every client, e.g.
    - client1
    - client2
    - client3

- ```
  ./build-key client1
  ```

Lab 3f: OpenVPN - Create the DiffieHellmann - Parameters file

- Explanation - see Diffie Hellmann section in this document

- `./build-dh`

Lab 3g: OpenVPN - Create the ta.key

- `openvpn --genkey --secret ta.key`

Lab 3h: OpenVPN - Create server config file

- Create file /etc/openvpn/server.conf
- ```
  proto udp
  port 1194
  dev tun
  server 10.200.0.0 255.255.255.0

  log-append /var/log/openvpn.log

  ca /etc/openvpn/training/keys/ca.crt
  cert /etc/openvpn/training/keys/openvpnserver.crt
  key /etc/openvpn/training/keys/openvpnserver.key
  dh /etc/openvpn/training/dh2048.pem
  ```

Lab 3k: OpenVPN - Launch Server

- `openvpn --config server.conf`

Lab 3l: OpenVPN - Create Client-config

```
* Create on client:
  * /etc/openvpn/client.conf
* <code>
```

client proto udp remote 192.168.33.10 port 1194 dev tun nobind

ca /etc/openvpn/ca.crt cert /etc/openvpn/client1.crt key /etc/openvpn/client1.key

daemon log-append /var/log/openvpn.log

</code>

Lab 3m: OpenVPN - Copy client-files

- on client: mkdir /etc/openvpn/training
- Securely copy (scp) client1.crt,client1.csr,client1.key and ca.crt to /etc/openvpn/
```

    - from server (ca-authority)

Lab 3n: OpenVPN - Start client

- ```
  cd /etc/openvpn/training
  openvpn --config client.conf
  ```

Lab 3o: OpenVPN - Fix certificate problem

- ```
  # Error
  WARNING: No server certificate verification method has been enabled.
  See http://openvpn.net/howto.html#mitm for more info.
  ```

- ```
  # Fix: add to clienttest.conf
  remote-cert-tls server
  ```

OpenVPN - List of keys

Filename	Needed By	Purpose	Secret
ca.crt	server + all clients	Root CA certificate	NO
ca.key	key signing machine only	Root CA key	YES
dh{n}.pem	server only	Diffie Hellman parameters	NO
server.crt	server only	Server Certificate	NO
server.key	server only	Server Key	YES
client1.crt	client1 only	Client1 Certificate	NO
client1.key	client1 only	Client1 Key	YES

OpenVPN - Changing --topology

- Stop Server
- Possible –topology is:

- ```
  net30 (default)
  subnet
  p2p
  ```

- subnet is the most effective one
  (but not set by default because of downward compability)

OpenVPN - net30

- Each time an address is given a multiple of 4 is used
- Example:
    - 10.200.0.[0-3] ... 10.200.20.1 will be the server address
        - Normally this block is for the OpenVPN - Server itself

- 10.200.0.[4-7] ... 10.200.20.6 is the client ip. Normally this is the block that is used for the first client
- 10.200.0.[8-11] .. [12-15] .. [16-19] are the blocks for the next clients

OpenVPN - example ip-only network (one server, multi clients) Client-Server IP-only network

```
# Server
proto udp
port 1194
dev tun
server 10.200.0.0 255.255.255.0
ca /etc/...
cert /etc/...
key /etc/...
dh /etc..
tls-auth /etc...

# important if you work with nobody
persist-key
persist-tun
keepalive 10 60

# All traffic to 10.198.0.x is redirected through openvpn server
push "route 10.198.0.0 255.255.255.0"
topology subnet

user nobody
group nobody

daemon
log-append /var/log/openvpn.log
```

(Step 2)

```
cp keys/ta.key ta.key
# copy ta.key to client
```

(Step 3) client config

```
client
proto udp
remote openvpnserver.example.com
port 1194
dev tun
nobind

ca
cert
```

```
key
tls-auth /etc/openvpn/training/ta.key 1

remote-cert-tls server
```

OpenVPN - Example of server.conf

```
port 1137
# use tcp or udp
proto udp
# dev tun will create a routed ip tunnel
dev tun
# certificate config
# ca certificate
ca /etc/openvpn/keys/ca.crt
# server certificate
cert /etc/openvpn/keys/server.crt
# server key and keep this a secret
key /etc/openvpn/keys/server.key

# Refers to the size in /etc/openvpn/keys/
dh /etc/openvpn/keys/dh1024.pem

# Internal IP will get when already connected
# Please replace x.x.x.x with the public ip address
# -> of the linux - server where the openvpnserver runs
#
server x.x.x.x 255.255.255.0

# This line will redirect all traffic through our OpenVPN
push "redirect-gateway def1"

# Provide DNS servers to the client, you can use google DNS
push "dhcp-option DNS 8.8.8.8"
push "dhcp-option DNS 8.8.4.4"

# enable multiple clients to connect with same key
duplicate-cn

#
keepalive 20 60
complzo
persist-key
persist-tun
daemon

# eanble log
log-append /var/log/openvpn.log

# Log Level
```

```
verb 3

## Save that one
```

OpenVPN - logfile

```
mkdir -p /var/log/myvpn/
touch /var/log/myvpn/openvpn.log
```

OpenVPN - Eventually disable firewall

```
# mask disable firewall
# makes it impossible to restart the firewall
systemctl mask firewalld
systemctl stop firewalld

# Disable selinux
setenforce permissive
# set for next boot
vim /etc/sysconfig/selinux
#
SELINUX=disabled
```

OpenVPN - dev->tun & --topology

- topology defines how to setup up the virtual devices and virtual ip's
- only together with –dev tun
  - in –dev tap no additional value
- possible options are:
  - net30 (default in openvpn 2.3)
    - one /30 subnet per client (point-to-point)
  - p2p
  - subnet (default in openvpn 2.4)
    - one ip per client
    - –topology subnet changes the interpretation
      of the arguments of –ifconfig to mean “address netmask”, no longer “local remote”.

OpenVPN - Client config with embedded certificates

- blog: https://www.brainfart.sg/index.php/2012/05/embedding-certificate-into-openvpn-config/
- client config with embedded certificates
- https://gist.github.com/ssinyagin/b196da5234c57de71bcfb44041274a15

OpenVPN - What is ccd ?

- Special configuration-files that are read after the client as connected
- DEFAULT or client-name (as in certificate) can be used.

OpenVPN - Background - Why iroute (in CCD)

- http://backreference.org/2009/11/15/openvpn-and-iroute/

OpenVPN - Connection Profiles

- OpenVPN will try to connect to these connection one after eachother
  - –> till one connection is reached

```
client
dev tun

<connection>
remote 198.19.34.56 1194 udp
</connection>

<connection>
remote 198.19.34.56 443 tcp
</connection>

<connection>
remote 198.19.34.56 443 tcp
http-proxy 192.168.0.8 8080
</connection>

<connection>
remote 198.19.36.99 443 tcp
http-proxy 192.168.0.8 8080
</connection>

persist-key
persist-tun
pkcs12 client.p12
remote-cert-tls server
verb 3
```

OpenVPN: Resigning & Revoking certificates

- Client - Certificates do have a certain time they are valids (defined in vars)
- When a certificate gets invalid you eventually want to create a new one

```
cd /etc/openvpn/training
# first you need to revoke the old one
# This removes it from the index (index.txt)
./revoke-full client1
# Now you sign a new certificate
# get all data
source vars
./pkitool --sign client1
```

```
# now you can transfer the new client1.crt to the client
```

OpenVPN: Working with revokation list

- Normally certificates cannot be revoked (by default)
- To be able to do so, openvpn needs to now about the list

```
cd /etc/openvpn/training
# revoke user1
source vars
./revoke-full client1
# show the revokation list
list-crl
# vi /etc/openvpn/training/servertest.conf
crl-verify /etc/openvpn/training/keys/crl.pem
# now restart the server CTRL+C
# Start openvpn again
# Try to connect with client
```

- Ref: https://blog.remibergsma.com/2013/02/27/improving-openvpn-security-by-revoking-unneeded-certificates/

OpenVPN - Routing scenario / additional network on server

- For the purpose of this example,
  - we will assume that the server-side LAN uses a subnet of 10.66.0.0/24 and the VPN IP address pool uses 10.8.0.0/24 as cited in the server directive in the OpenVPN server configuration file.
  - First, you must advertise the 10.66.0.0/24 subnet to VPN clients as being accessible through the VPN. This can easily be done with the following server-side config file directive:
- push “route 10.66.0.0 255.255.255.0”
- Only if Gateway and OpenVPN - Server are different:
  - ```
    Next, you must set up a route on the server-side LAN gateway
    to route the VPN client subnet (10.8.0.0/24) to the OpenVPN server
    (this is only necessary if the OpenVPN server and the LAN gateway
    are different machines)
    ```

OpenVPN - iroute / route / push "route...."

- **All this is only needed when you have**
  - more clients behind a vpn-server (a network)
  - more clients behind a vpn-client (a network)
- route / traffic from kernel to vpn
- iroute / routing within openvpn
  - Helps openvpn to understand to which vpn client a network belongs to
- push “route” - routing for client
    ```

- pushed to client and set there in his routing table

OpenVPN - Multiple machines on the OpenVPN client side

- make sure ip/ & tun/tap - forwarding isset
  - echo 1 > /proc/sys/net/ipv4/ip_forward
- client LAN is using the 192.168.4.0/24
- certificate with a common name of client2
- Our goal is to set up the VPN so that any machine on the client LAN can communicate with any machine on the server LAN through the VPN.
- **Important:** Every subnet which is joined to the VPN via routing must be unique.
- The client must have a unique Common Name in its certificate
  - (“client2” in our example),
  - and the duplicate-cn flag must not be used in the OpenVPN server configuration file.
- client-config-dir ccd
  - Directory for settings client-based basis after connecting
- ccd/client2: iroute 192.168.4.0 255.255.255.0
  - This will tell the OpenVPN server that the 192.168.4.0/24 subnet should be routed to client2.
- Main config openvpn: route 192.168.4.0 255.255.255.0
  - Why redundant (route / iroute)?
    - route controls controls the routing from kernel to openvpn
    - iroute controls the routing from the openvpnserver to the openvpnclients
  - –
  - Next: Next, ask yourself if you would like to allow network traffic between client2's subnet (192.168.4.0/24) and other clients of the OpenVPN server.
    If so, add the following to the server config file.

```
client-to-client
push "route 192.168.4.0 255.255.255.0"
```

  - Route on Server-Lan-Gateway: (if gateway is present)
    - add a route to the server's LAN gateway which directs 192.168.4.0/24 to the OpenVPN server box
    - 192.168.4.0 netmask 255.255.255.0 gw x.x.x.x
      - x.x.x.x ip (not openvpn - ip) of openvpn -server

OpenVPN - Howto

- For a lot of commonly known scenarios, there is a good howto
  - https://openvpn.net/index.php/open-source/documentation/howto.html

Securing Passwords (Linux)

Secure Passwords - Password Length

- 4 characters: 456.976 combinations
- 5 characters: 11,8 million combinatons

- 6 characters: 308,9 million combinations
- 7 characters: 8 billion combinations
- 8 characters: 200 billion combinations
- 9 characters: 5,4 trillion combinations
- 10 characters: 141 trillion combinations
- 12 characters: 95 quadrillion combinations

Secure passwords - cracking time based on length

- Prerequisites: a million guesses per second (not unlikely with todays systems)
- crack .... a password of
  - 6 chars → in → 5 minutes
  - 8 chars → in → 2,5 days
  - 12 chars → 3026 years
- THIS is why attackers like dictionaries !!

Security Scanning

Security Scan (Webserver) with nicto

```
# debian stretch
apt install nikto
perl nikto.pl -host http://www.google-no-dont-do-that.com
```

Malware Detection

Malware detect with maldetect

```
https://www.rfxn.com/projects/linux-malware-detect/
```

Prevent DDOS attacks / Restrict Connections

fail2ban - Debian stretch

```
apt install fail2ban
systemctl status fail2ban.service
# The will be a new chain
# Chain f2b-sshd (1 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere
root@stretch:/etc/fail2ban#

iptables -L
Chain f2b-sshd (1 references)
target     prot opt source               destination
```

```
RETURN     all  --  anywhere             anywhere
root@stretch:/etc/fail2ban#

# ssh is already set on installation
# but not activated here
# /etc/fail2ban/fail2ban.conf
# that one is considered as a jail

[sshd]

port    = ssh
logpath = %(sshd_log)s
backend = %(sshd_backend)s

# But in the jail.d
# defaults-debian.conf activates it

[sshd]
enabled = true

# what it jails is done by the filter
# normally the same name as the jail
# -> jail.conf
# -> filter = %(__name__)s
# so /etc/fail2ban/filter.d/sshd.conf
```

fail2ban -> fail2ban-client

- fail2ban-client
- fail2ban-client -i # interactive
- https://www.fail2ban.org/wiki/index.php/Commands

fail2ban -> sshd -> status/banned ip's

- fail2ban-client status sshd
- Important: totally banned... means all time (not necessarily now)
- currently banned .. means what is banned now

Logs of fail2ban

- /var/log/fail2ban.log

Alternative

- sshguard
- Available in Repo in Ubuntu/Debian ?

Local Security

Advanced Unix Permissions (POSIX capabilities)

- Has nothing to do with SELinux
- setcap (to modify capabilities)
- getcap (to monitor capabilities)
- great way to reduce setuid or sudo usage

Check capabilities of executable

- ```
  getcap /sbin/ping
  /sbin/ping = cap_net_raw+ep
  ```

Show all capabilities

- man capabilities

Set in pam

```
# generally allow user to use it
# but he has to set it
# setcap cap_net_raw+p anotherping
# vim /etc/pam.d/system-login
auth required pam_cap.so

# vim /etc/security/capability.conf
cap_net_raw    user1
```

See set capabilities

```
#filecap
file                capabilities
/bin/anotherping    net_raw
```

Test with ping(test)

```
# on ubuntu 18.04
# as root
ls -la /bin/ping
# setuid set
cd /bin
cp -a ping pingtest
ls -la pingtest
chmod u-s pingtest
pingtest 127.0.0.1
# no permission
setcap cap_net_raw+p /bin/pingtest
```
```

```
# now we have permission
# without needing suid
ping pingtest
```

Capabilities - the modes

e: Effective	This means the capability is “activated”
p: Permitted	This means the capability can be used/is allowed.
i: Inherited	The capability is kept by child/subprocesses upon execve() for example.

* more info: man cap_from_text

Capabilities - ref

- https://www.insecure.ws/linux/getcap_setcap.html

Mandatory Access Control (MACs)

SELinux - Debian Stretch (Install)

- Basic introduction for Debian:
  - https://wiki.debian.org/SELinux/Setup
- Install
  - `apt install selinux-basics selinux-policy-default auditd`

SELinux - Debian Stretch (Configure)

- Configure pam and grub + /.autorelabel
  - ```
    selinux-activate
    # System will be set to permissive mode
    # Output:
    #SELinuxfs mount: /sys/fs/selinux
    #SELinux root directory: /etc/selinux
    #Loaded policy name: default
    #Current mode: enforcing
    #Mode from config file: permissive
    #Policy MLS status: enabled
    #Policy deny_unknown status: allowed
    #Max kernel policy version: 30
    # SE Linux is activated. You may need to reboot now.
    ```

SELinux - Debian Stretch (autorelabel ?)

- What does the .autorelabel file located at the file root / do in Linux?
    ```

- Next time when you will reboot the system, it will relabel the filesystem for SElinux automatically
- This needs to be done when enabling SELinux

```
A relabel walks all of the mounted file systems that support labelling,
and compares the file context on the file to the system default,
if they differ, the process will fix the label.
```

SELinux - Check current SELinux mode

```
getenforce
```

SELinux - Change current SELinux mode (runtime)

```
# same as
# setenforce 0
setenforce permissive
getenforce
sestatus
# same as
# setenforce 1
setenforce enforcing
getenforce
sestatus
```

SELinux - sestatus

*

```
# Reflects the current runtime and configuration state
sestatus
```

SELinux - set mode for next boot

*

```
# /etc/selinux/config
SELINUX=permissive

# For this change to take effect you need to reboot
reboot
```

SELinux - Prevent to switch to permissive mode (permanently)

```
# -P => persistent
setsebool -P secure_mode_policyload 1
```

SELinux - Reallow to switch to permissive mode

- Attention: This only works when system is not in enforcing mode
- setsebool -P secure_mode_policyload 0

SELinux - Debian Stretch (Check)

- Check the configuration

```
# on Debian Stretch, this script does not work properly
# because we are using systemd not initscripts
# this can help to fake
touch /etc/default/rcS
check-selinux-installation
```

SELinux - Check per file

```
ls -Z /etc/passwd
```

SELinux - Context

- Context:
  - Every file and process is labelled with addditional information
    - ps -Z
    - ls -Z
  - a set of rules that define security and access rights
    - for everything in the system
- Everything = Users, Roles, Process, Files

SELinux - Users

General

- Users ⇒ In SELinux → Subjects
- There are some pre-built Users in SELinux

Context of my login user

```
id -Z
```

Login User to SELinux User mapping

- Each Linux User is mapped to a SELinux user
- Done via SELinux policy

List Mappings -> Linux Login User -> SELinux User mapping

```
# show all the linux users and which seuser they mare mapped to
semanage login -l
```

List all SELinux - Users

- List all available selinux users in the system
- ```
  seinfo -u
  ```

Mapping a Login User to an SELinux User

- ```
  # Eventually add the user before
  # adduser training
  semanage login -a -s user_u training
  ```

SELinux - Roles

- Roles like a gateway between
  - Users and a process
  - Users ⇐ Role ⇒ Process
- A role defines which users can access that process
- Roles are like filters

See all roles in the system

```
seinfo -r
```

Which types can a role access

- Shows, which types are assigned to a role
- ```
  seinfo -ruser_r -x
  ```

Built-In set of roles

Role	Description
user_r	The regular user role, which is meant to only allow user applications and other non-privileged domains
staff_r	Similar to the user_r role, but might be allowed to receive more system information than a regular user. This role is mostly given to users that should be allowed to switch towards other roles.
  ```

Role	Description
sysadm_r	System administrative role; this is a very powerful role as it is allowed most target domains, including privileged domains. Use with care.
system_r	System role, not meant to be switched to directly (and newrole will even disallow it as it doesn't have a default user domain associated with it - something we'll talk about later)

SELinux - Subjects and Objects

- A subject is a process and can potentially affect an object.
- An object in SELinux is anything that can be acted upon:
  - a file
  - a directory
  - a port
  - a tcp socket
- The actions that a subject can perform on an object are the subject's permissions.

SELinux - Types are for Objects

- e.g. context may dictate → it's a web page
- or: file belongs to /etc directory
- or: a files owner is a specific SELinux User
- File's context:
  - in .. SELinux → Type

SELinux - Policies

What is a policy ?

- ```
  User -> Role -> Domain -> File
  ```

- A role defines what users may access
- Domains determines what roles are authorized
- Domains can access certain types of files

SELinux - How policies work ?

- Steps
  - Step 1: User has to be authorized to enter a role
  - Step 2: role has to be authorized to access the domain.
  - Step 3: The domain in turn is restricted to access only certain types of files.

SELinux - Policy store & policy modules

- sestatus | grep “policy name”

- ```
  # Output
  Loaded policy name:             default
  ```

- refers to: /etc/selinux/default/

- ```
  # shows all modules loaded in memory
  ```
  ```

```
semodule -l | less
```

SELinux - Modules

SELinux - semodule

- can be used for:
  - installing
  - removing
  - reloading
  - upgrading
  - enabling
  - disabling
- → SELinux policy modules.

SELinux - see setting of modules

```
semanage boolean -l | less
```

SELinux - enable / disable a module properties

```
getsebool allow_ftpd_anon_write
# output
allow_ftpd_anon_write --> off

setsebool allow_ftpd_anon_write on
getsebool allow_ftpd_anon_write
# output
# allow_ftpd_anon_write --> on
```

SELinux - security contexts

See context of files

```
ls -Z /etc/*.conf
system_u:object_r:etc_t:s0 /etc/nscd.conf
system_u:object_r:etc_t:s0 /etc/nsswitch.conf
system_u:object_r:ntp_conf_t:s0 /etc/ntp.conf
```

SELinux - security contexts (concepts->files)

```
system_u:object_r:etc_t:s0

part1: (system_u) => _u = user
part2: (object_r) => _r = role
part3: (etc_t)    => _t = type or domain
part4: (s0)       => s = sensitivity (use for multilevel security or MLS)
```

Lab: Processes and Apache (Debian Stretch)

- in processes the third entry (httpd_t) is the domain

```
apt install apache2
systemctl. status apache2
ps -efZ | grep httpd
# output
# ...
system_u:system_r:httpd_t:s0    root      9967     1  0 04:18 ?
00:00:00 /usr/sbin/apache2 -k start
```

SELinux - processes = domains ?

- Command: ps -efZ
- What does a domain do for a process:
  - It gives the process a context to run within.
  - It's like a bubble around the process that confines it.
  - It tells the process what it can do and what it can't do.
  - This confinement makes sure each process domain can act on only certain types of files and nothing more.

SELinux - How processes access resources ..

- access rule in a policy
- structure:
  - allow <domain> <type>:<class> { <permissions> };
    - process → in domain ?
    - access of certain type & class ?
      - Then → Allow access
      - Else → Deny access

SELinux - Lab: Permission on files

- ```
  # be sure selinux is activated
  setenforce 1
  ps -efZ | grep apache2
  system_u:system_r:httpd_t:s0    root      9967     1  0 04:18 ?
  00:00:00 /usr/sbin/apache2 -k start

  touch /var/www/html/index.html
  ls -Z /var/www/html/*
  # output
  unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/index.html

  # So is http_t - domain allowed to access ?
  sesearch --allow --source httpd_t --target httpd_sys_content_t --class
  file
  # Yes !
  ```
  ```

```
# output
allow httpd_t httpd_sys_content_t:file { lock ioctl read getattr open
};
allow httpd_t httpdcontent:file { create link open append rename write
ioctl lock getattr unlink setattr read }; [ ( httpd_builtin_scripting
&& httpd_unified && httpd_enable_cgi ) ]:True
...

# so let's check
echo "<html><body>hello</body></html>" > /var/www/html/index.html
chmod 775 /var/www/html/index.html

# open in browser:
# e.g.
# http://<yourip>

# you should get an output -> hello ;o)

# Now change the type of the file
# ONLY changes temporarily
# NEXT restorecon breaks it.
chcon --type var_t /var/www/html/index.html
ls -Z /var/www/html/index.html

# open in browser again
# http://<yourip>
# NOW -> you should have a permission denied
# Why ? -> var_t is not one of the context the webserver domain
(http_t) is not authorized to connect to

# Doublecheck
sesearch --allow --source httpd_t --target var_t --class file
# -> no output here -> no access

# Restore again
restorecon -v /var/www/html/index.html
# output
# Relabeled /var/www/html/index.html from
unconfined_u:object_r:var_t:s0 to
unconfined_u:object_r:httpd_sys_content_t:s0
ls -Z /var/www/html/index.html
# output
unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/index.html

# open in browser again
# http://<yourip>
# Now testpage works again
```

SELinux - context inheritance

```
cd /var/www/
ls -Z
# output
#/var/www/
ls -Z  .
system_u:object_r:httpd_sys_content_t:s0 html

# If you create a file within it will have the same context
cd html
touch page.html
ls -Z page.html
# httpd_sys_content_t
unconfined_u:object_r:httpd_sys_content_t:s0 page.html
```

SELinux - Copying data (context change ?)

```
cp /var/www/html/index.html /var/
/var/www/html# cd /var
/var# ls -Z index.html
# it will have the context of its parent directory -> /var
unconfined_u:object_r:var_t:s0 index.html
```

SELinux - Copying data (preserve context)

```
cp -a --preserve=context /var/www/html/index.html /var/index_preserved.html
```

SELinux - Moving data

```
mv /var/index.html /etc/index.html
# context is preserved by default
ls -Z /var/index.html
ls -Z /etc/index.html
```

SELinux - restorecone => stored context

- restorecon - will apply settings from here:
  - head /etc/selinux/default/contexts/files/file_contexts
  - ```
    /.* system_u:object_r:default_t:s0
    /a?quota.(user|group)   --    system_u:object_r:quota_db_t:s0
    /sys(/.*)?  system_u:object_r:sysfs_t:s0
    /xen(/.*)?  system_u:object_r:xen_image_t:s0
    /mnt(/[^/]*)    -l    system_u:object_r:mnt_t:s0
    /mnt(/[^/]*)?   -d    system_u:object_r:mnt_t:s0
    /usr/.* system_u:object_r:usr_t:s0
    /var/.* system_u:object_r:var_t:s0
    ```
    ```

```
/run/.* system_u:object_r:var_run_t:s0
/tmp/.* <<none>>
```

SELinux - Lab - new folder

```
mkdir -p /www/html
ls -Z /www/
# output
unconfined_u:object_r:root_t:s0
#
cp /var/www/html/index.html /www/html/
```

SELinux - Labelling of the Apache - Webserver

```
# Example of how all the parts of the apache2 - webserver might be labelled
Binary: /usr/sbin/httpd→httpd_exec_t
Configuration directory: /etc/httpd→httpd_config_t
Logfile directory: /var/log/httpd → httpd_log_t
Content directory: /var/www/html → httpd_sys_content_t
Startup script: /usr/lib/systemd/system/httpd.service → httpd_unit_file_d
Process: /usr/sbin/httpd -DFOREGROUND → httpd_t
Ports: 80/tcp, 443/tcp → httpd_t, http_port_t
```

SELinux - Domain transition

Search for configure transition

```
sesearch -T
sesearch -T | grep "process httpd_t"
```

Lab: Domain transitions

```
apt install vsftpd
service vsftpd start
# Let's look into systemd the ancestor of all processes
ps -eZ  | grep init
# system_u:system_r:init_t:s0         1 ?        00:00:02 systemd
# systemd will invoke vsftpd - binary
#
ls -Z /usr/sbin/vsftpd
#-rwxr-xr-x. root root system_u:object_r:ftpd_exec_t:s0 /usr/sbin/vsftpd

# checking the process
ps -eZ | grep vsftpd
# output
system_u:system_r:ftpd_t:s0-s0:c0.c1023 7708 ? 00:00:00 vsftpd
#
# init_t -> ftpd_exec_t -> ftpd_t
```

```
# -> this transition is done through the policies of selinux
# -> that are loaded at startup of selinux into memory
```

Theory: Rules for domain transition

- Domain transition is subject to three rules
  - Rule 1:
    - source domain init needs to have execute permission on the entrypoint application with the ftpdexec_t
    - verify:
      - sesearch -s init_t -t ftpd_exec_t -c file -p execute -A
      - `# output`
  - Rule 2:
    - Next, we check if the binary file is the entrypoint for the target domain ftpd_t:
    - verify:
      - sesearch -s ftpd_t -t ftpd_exec_t -c file -p entrypoint -A
      - `#output`
  - Rule 3:
    - And finally, the source domain initt needs to have permission to transition to the target domain ftpdt:
    - verify:
      - sesearch -s init_t -t ftpd_t -c process -p transition -A

Theory When is a transition allowed ?

- 3 requirements
  1. origin domain has execute permissions on the file
  2. file context is defined as entry point for the target domain
  3. origin domain is allowed to transition to the target domain.
- Only when 3 requirements meet, transition will be done

SELinux - Does an app use selinux directly

- Application is then: selinux-aware
- Behaviour is probably different then for non-aware apps
  when

switching from enforcing to permissive

- ```
  [root@localhost selinux]# ldd /sbin/sshd
      linux-vdso.so.1 =>  (0x00007ffcbdfd6000)
      libfipscheck.so.1 => /lib64/libfipscheck.so.1 (0x00007fc237816000)
          ...
      libselinux.so.1 => /lib64/libselinux.so.1 (0x00007fc236fab000)
  ```
  ```

SELinux - Logs

Where ?

- On Debian Stretch (if auditd is installed) and in Centos 7 / Redhat 7
- Logs are in /var/log/audit/audit.log
- ausearch -m avc -c httpd
- AVC = access vector scan

Forwarding log with audisp-remote plugin

```
# on remote - server
# configure remote system to listen on port 60
# vi /etc/audit/auditd.conf
tcp_listen_port = 60

# on local - server
# vi /etc/audisp/audisp-remote.conf
remote_server = <targethost-name>
port = 60

# enable the audisp-remote plugin:
# yum install audispd-plugins
# vi /etc/audisp/plugins.d/au-remote.conf
active = yes
```

Example of creating a report based on audit logs

```
aureport --avc --start recent
```

ausearch / audit.log

Prerequisites

- Is auditd running ?
  - Check for systemctl status auditd
  - If not running event will be logged to syslog → kernel.*
    - → /var/log/messages

Raw

- Audit information from auditd is normally saved in ...
  - /var/log/audit/audit.log
- This is text and you can look into it

ausearch

- Goes through the audit log and extracts data

Example: Activity of a specific user being denied

```
# search with user [id]
```

```
ausearch -ua training
```

Filter by time

```
# Show all events that happened in the last 10 minutes
ausearch -ts recent
# today
ausearch -ts today
# yesterday
ausearch -ts yesterday
```

Show only AVC (Access Vector)- Events

```
ausearch -m AVC -ts recent
```

SELinux - Stats of policy file

- seinfo –stats

SELinux - Dontaudit

- In a rule the creator can set rule to dontaudit

then it will not shown in logs

- ```
  # See how many Dontaudit rules are active
  seinfo --status | grep -i audit
  ```

SELinux - Dontaudit -> Audit - Debugging

- To get more debugging infos disable the **Dontaudit** rules

- ```
  # disable dontaudit
  semodule --disable_dontaudit --build
  # Reenable it again
  semodule -B
  ```

SELinux - Disable selinux at boot

- grub: selinux=0
  - Start linux with selinux deactivated + touches ./autorelabel file in /
  - By having an ./autorelabel file in place, all necessary contexts for files will be set

SELinux - Enforcing/Permissive set at boot

- grub: enforcing=0
- grub: enforcing=1

SELinux - Protecting grub at boot

- It is in general a good idea to protect grub with a password
  when you use selinux, to preventing other to boot in permissive mode or disable selinux altogether.

SELinux - Common Tasks well explained

- https://opensource.com/article/18/7/sysadmin-guide-selinux

SELinux - Creating a module

- http://www.billauer.co.il/selinux-policy-module-howto.html
- https://debian-handbook.info/browse/de-DE/stable/sect.selinux.html

Kernel Vulnerabilities Networking

- The kernel supports on-demand loading of kernel modules
- Prevent all unnecessary protocols to be loaded:

```
To Disable All Unnecessary Protocol Stacks
Modify the following lines to the /etc/modules.conf file:

alias net-pf-4 off # IPX
alias net-pf-5 off # Appletalk
alias net-pf-10 off # IPv6
alias net-pf-12 off # Decnet
```

- Ref: http://www.informit.com/articles/article.aspx?p=101181&seqNum=2

Apparmor

How it works ?

```
In practice
  -> the kernel queries AppArmor before each system call
      ->to know whether the process is authorized to do the given
operation.
```

Set up utilities you need for management

```
sudo apt-get install apparmor-utils
```

Show the current status of apparmor

```
sudo apparmor_status
# or
sudo aa_status
```

Set up additional profiles

- Within the core installation
  - there are only a minimal number of profiles
- So:
  - apt install apparmor-profiles

Disable a profile altogether

```
sudo ln -s /etc/apparmor.d/<profile> /etc/apparmor/disable/
# rereads that single profile
sudo apparmor_parser -R /etc/apparmor.d/<profile>
```

Re-Enable a disabled profile

```
sudo rm /etc/apparmor.de/disable/<profile>
cat /etc/apparmor.d/<profile> | sudo apparmor_parser -a
```

Set a specific profile to complain mode

- Similar to 'permissive' in selinux
- `sudo aa-complain nginx`

Set a specific profile to enforce mode

- `sudo aa-enforce nginx`

Find out which services are not protected

```
# in checks with netstats what ports are open
# and compares it with the given profiles
sudo aa-unconfined

## Example output:
5460 /usr/sbin/avahi-daemon not confined
5460 /usr/sbin/avahi-daemon not confined
5806 /sbin/dhclient3 not confined
18367 /usr/sbin/cupsd confined by '/usr/sbin/cupsd (enforce)'
```

Ref

https://www.digitalocean.com/community/tutorials/how-to-create-an-apparmor-profile-for-nginx-on-ubuntu-14-04

Remote Attacks and Tools

Syn-Flooding (tcp - Layer 3/4)

Find out syn-flood attack against webserver

```
netstat -tuna | grep :80 | grep SYN_RECV
# open syn, without ack from attacker (normally: syn, syn-ack, ack)
```

Tool: hping3 (e.g. for syn flooding)

- Free packet generator and analyzer
- Can not only ping icmp
- Part of Kali Linux
- Helpful tool to spoof ip (source ip)

Example dos-attack (syn-flood) - random source ip

- ```
  hping3 -c 10000 -d 120 -S -w 64 -p 21 --flood --rand-source
  www.hping3testsite.com
  hping3 = Name of the application binary.
  -c 100000 = Number of packets to send.
  -d 120 = Size of each packet that was sent to target machine.
  -S = I am sending SYN packets only.
  -w 64 = TCP window size.
  -p 21 = Destination port (21 being FTP port). You can use any port
  here.
  --flood = Sending packets as fast as possible, without taking care to
  show incoming replies. Flood mode.
  --rand-source = Using Random Source IP Addresses. You can also use -a
  or –spoof to hide hostnames. See MAN page below.
  www.hping3testsite.com = Destination IP address or target machines IP
  address. You can also use a website name here. In my case resolves to
  127.0.0.1 (as entered in /etc/hosts file)
  ```

Example dos-attack (syn-flood) - simple version

```
hping3 -S --flood -V www.hping3testsite.com
```
```

Harden Kernel - Prevent syn flooding

- net.ipv4.tcp_syncookies = 1

- ```
  This is the most effective method of defending from SYN Flood attack.
  The use of SYN cookies allow a server to avoid dropping connections
  when the SYN queue fills up. Instead, the server behaves
  as if the SYN queue has been enlarged.
  The server sends back the appropriate SYN+ACK response to the client
  but discards the SYN queue entry. If the server then receives
  a subsequent ACK response from the client, it is able to reconstruct
  the SYN queue entry using information encoded in the TCP sequence
  number.
  ```

Hacking ;o)

Get Metasploitable 2

- Ready vulnerable machine to do testing
- https://metasploit.help.rapid7.com/docs/metasploitable-2
- You will get a virtual machine there, you can also use with virtualbox

Set up Metasploitable 2 machine with VirtualBox

- https://pdrcybersecurity.com/six-steps-install-metasploitable-2-virtualbox/

Get Kali Linux

- Get your virtual machine for kali linux

Metasploit: Work with db and hosts

- https://www.offensive-security.com/metasploit-unleashed/using-databases/

Metasploit: Run an exploit (Shellshock)

- https://null-byte.wonderhowto.com/how-to/exploit-shellshock-web-server-using-metasploit-0186084/
  ```