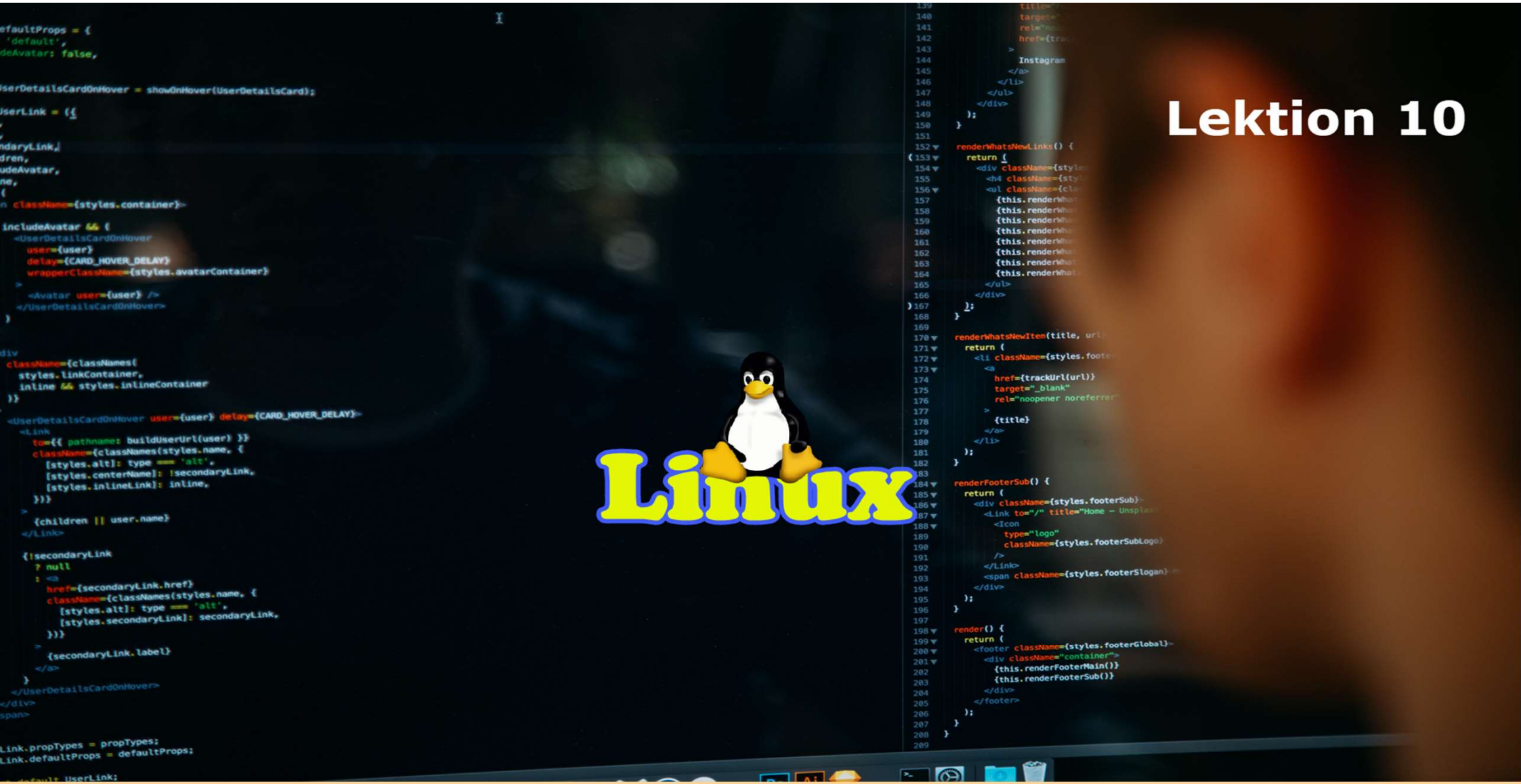
Lektion 10
Linux

Sicherheit

- Alle bisherigen Lektionen haben schon den Sicherheitsgedanken mitgetragen
  - Denken Sie beispielsweise an die verschlüsselte ssh-Verbindung oder
  - an unsere Backup-Strategie
- Sowohl der (die) Server als auch die Clientrechner sollten noch intensiver vor Malware und Missbrauch geschützt werden
- Dafür gibt es verschiedene Ansätze. Folgende werden wir hier behandeln
  - UFW-Firewall – der grundlegende Schutz für Client und Server
  - nmap – als Portscanner, um offene, ungenutzte Ports zu erkennen
  - fail2Ban – Intrusion Detektion IDS und Intrusion Protektion IPS (Server)
  - chage – der Zwang zur Passwort-Änderung (Client)

Firewall

- Eine Firewall ist nur ein Teil des Sicherheitskonzepts für einen Rechner (Server)
- Sie dient dazu gewünschten von ungewünschten Netzwerkverkehr zu unterscheiden und den ungewünschten Netzverkehr zu unterbinden
- Um diese Aufgaben zu erfüllen, gibt es verschiedene Lösungsansätze für die Nutzung von Firewalls
- Den größten Umfang bietet in diesem Zusammenhang iptables bzw. nftables die meisten Funktionen an
- Diese Firewall kann unter anderem als Personal- und als Netzwerkfirewall eigesetzt werden

Netfilter/iptables

- Netfilter ist ein modulares Softwarepaket, welches innerhalb des Linux-Kernels vor allem TCP/IP Netzwerkpakete analysiert und manipuliert
- Sie ist die zentrale Komponente für die Linux-Firewall
- Zur Konfiguration der Software wird das Programm iptables oder das leichter zu bedienende Programm UFW eingesetzt
- Im Sprachgebrauch wird Netfilter mit iptables gleichgesetzt, so dass nur iptables genannt wird
- Iptables wird momentan auf einigen Distributionen durch das modernere Nftables ersetzt

Iptables die Linux-Firewall

- Die Konfiguration von Iptables gehört nicht zu den leichten Aufgaben im Sicherheitskonzept von Linux
- Zum Glück gibt es ein leicht zu bedienendes Frontend zu Iptables mit dem Namen UFW (Uncomplicated Firewall)
- In Debian kann man das Paket aus den Paketquellen herunterladen und installieren
  - `# apt install ufw`
- Die Einrichtung erfolgt sowohl auf dem Client als Personal-Firewall und auch auf dem Server als Netzwerkfirewall

UFW auf dem Client

- Nach der unspektakulären Installation wird der Status von ufw abgefragt

```
root@debian:~# ufw status
Status: inactive
root@debian:~# systemctl status ufw
● ufw.service - Uncomplicated firewall
   Loaded: loaded (/lib/systemd/system/ufw.service; enabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:ufw(8)
```

- Sowohl die eigene Statusabfrage, als auch die Abfrage mit systemctl zeigen an, dass ufw inaktiv ist

ufw auf dem Client (debian) aktivieren

- Im nächsten Schritt werden wir die Firewall aktivieren. Mit dem Zusatz verbose macht man die Statusanzeige „geschwätzig“

```
root@debian:~# ufw status verbose
Status: inactive
root@debian:~# ufw enable
Firewall is active and enabled on system startup
root@debian:~# ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
```

- Eingehende Verbindungen (incoming) sind gesperrt, nach außen gehend (outgoing) sind erlaubt. So soll es sein!

Client belassen, aber der Server ...

- Ein ping vom Server zum Client wird jetzt unterbunden
- Ein Server muss für seine eingerichteten Dienste von außen, also von der Clientseite erreichbar sein
- Für diese Dienste müssen die verwendeten Ports bekannt sein und es muss geklärt sein, ob der Server für die Dienste auch vom Internet erreichbar sein soll. Für unsere Dienste ist das nicht der Fall
- Ausnahmen bilden natürlich Web-Server und Mail-Server, die vom Internet aus erreichbar sein sollen

ufw auf dem Server installieren

- Die Installation auf dem Server vollzieht sich wie auf dem Client, aber über die vorhandene ssh-Verbindung. Auch auf dem Server benötigen wir zur Installation selbstverständlich root-Rechte. Hier ein Ausschnitt aus der Installation

```
root@deb-s1:~# apt install ufw
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
Die folgenden NEUEN Pakete werden installiert:
  ufw
```

- Auch auf dem Server ist ufw anfangs inaktiv. Das gibt uns die Möglichkeit weiterhin ssh zu nutzen und sofort dafür die erste Freigabe zu erstellen

ssh auf dem Server erlauben

- Für alle Regeln, die eingerichtet werden gilt, dass sie so streng wie möglich und so frei wie nötig formuliert werden
- Für unsere ssh-Verbindung heißt das, dass es nur aus unserem lokalen Subnetz (also 192.168.111.0/24) genutzt werden soll.
- Nur Port 22 und nur das Protokoll tcp werden von ssh verwendet. Die dafür passende Freigaberegel lautet:

```
root@deb-s1:~# ufw allow from 192.168.111.0/24 to any port 22 proto tcp
Rules updated
root@deb-s1:~#
```

- Nachdem <Enter> gedrückt wurde, erscheint „Rules updated“

ssh auf dem Server erlauben

- Durch die Adresseingabe 192.168.111.0/24 kann man theoretisch von jedem PC im genannten Subnetz die ssh-Verbindung aufbauen
- Wollen Sie nur einen speziellen PC dafür nutzen, sollten Sie nur dessen IP-Adresse angeben z.B. 192.168.111.16. Das würde so aussehen:

```
root@deb-s1:~# ufw allow from 192.168.111.16 to any port 22 proto tcp
```

- Bedenken Sie, dass dieser PC seine IP-Adresse nicht wechseln darf
  - Das lässt sich im DHCP-Server für diesen PC konfigurieren (z.B. in der FritzBox) oder
  - durch Zuweisung einer statischen IP erreichen
- Wir belassen es hier auf dem gesamten Subnetz

Die Firewall aktivieren

- Für das Einrichten weiterer Regeln können wir nun ssh benutzen und zwar bei aktivierter Firewall
- Um die Firewall zu aktivieren, geben Sie ein:

```
root@deb-s1:~# ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
root@deb-s1:~# ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       192.168.111.0/24

root@deb-s1:~#
```

- Wenn die ssh-Verbindung stabil bleibt, hat die Regel funktioniert ☺

Übersicht der Protokolle und Ports

- Um herauszufinden, welche Transportprotokolle und Ports für die laufenden Prozesse genutzt werden bedient man sich des Kommandos
  - `# netstat –tulpn` (oder als Alternative: `# ss –tulpn`)
  - Für die Freigabe interessant sind die unspezifischen Adressen (0.0.0.0:<Port>) und deren Protokolle

```
root@deb-s1:~# netstat -tulpn
Aktive Internetverbindungen (Nur Server)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      482/sshd: /usr/sbin
tcp        0      0 0.0.0.0:139             0.0.0.0:*               LISTEN      494/smbd
tcp        0      0 0.0.0.0:445             0.0.0.0:*               LISTEN      494/smbd
tcp6       0      0 :::22                   :::*                    LISTEN      482/sshd: /usr/sbin
tcp6       0      0 :::139                  :::*                    LISTEN      494/smbd
tcp6       0      0 :::445                  :::*                    LISTEN      494/smbd
udp        0      0 192.168.111.255:137     0.0.0.0:*                           484/nmbd
udp        0      0 192.168.111.2:137       0.0.0.0:*                           484/nmbd
udp        0      0 0.0.0.0:137             0.0.0.0:*                           484/nmbd
udp        0      0 192.168.111.255:138     0.0.0.0:*                           484/nmbd
udp        0      0 192.168.111.2:138       0.0.0.0:*                           484/nmbd
udp        0      0 0.0.0.0:138             0.0.0.0:*                           484/nmbd
```

Port 22 für sshd ist schon frei gegeben!

Wenn SAMBA installiert wäre ...

```
root@deb-s1:~# ufw allow from 192.168.111.0/24 to any port 139 proto tcp
Rule added
root@deb-s1:~# ufw allow from 192.168.111.0/24 to any port 445 proto tcp
Rule added
root@deb-s1:~# ufw allow from 192.168.111.0/24 to any port 137 proto udp
Rule added
root@deb-s1:~# ufw allow from 192.168.111.0/24 to any port 138 proto udp
Rule added
root@deb-s1:~#
```

- Sollte eine Regel mit falschen Parametern erstellt worden sein z. B. ein falsches Protokoll, so können Sie diese Regel mit
- `# ufw status numbered` in numerierter Form anzeigen lassen und danach mit
- `# ufw delete mit der angezeigten Nummer z.B. ufw delete 7` wieder löschen
- Entfernen Sie alle nicht benötigten Regeln!

Weitere ufw-Einstellungen

- Wollen Sie die Firewall zwischenzeitlich einmal ausschalten, so geben Sie ein:
  - `# ufw disable` (wieder einschalten mit # ufw enable)
- Falls Änderungen nicht erkannt werden sollten, geben Sie ein:
  - `# ufw reload`
- Um sicher zu gehen, dass alle Ports, die Sie nicht freigegeben haben, auch wirklich gesperrt sind, geben Sie ein:
  - `# ufw default reject`
- Damit ist die Firewall für die beiden Dienste optimal eingestellt
- Wenn weitere Dienste installiert werden, müssen Sie die Firewall darauf anpassen z.B. Apache-Webserver als Intranet: Port 80/tcp freigeben

nmap

- nmap gehört, wie wireshark, zu den Netzwerksniffern und unterliegt damit dem Hackerparagraph § 202c StGB
  - Daher nutzen Sie beide Programme bitte nur in Ihrem(n) Subnetz(en). Es ist dort (wie wireshark auch) ausdrücklich erlaubt, denn man muss in der Lage sein, sein eigenes Netz zu scannen
  - Installation: `# apt install nmap`
- Wir nutzen nmap hier nur als Portscanner, um zu kontrollieren, welche Ports auf den Clients bzw. Servern offen sind
  - `# nmap –F  192.168.111.0/24`
- Mit diesem Kommando werden die 100 wichtigsten Ports (-F) auf allen Knoten im Subnetz gescannt
- nmap reagiert auch auf die Firewall-Regeln

nmap

```
root@deb-s1:~# nmap -F 192.168.111.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2022-07-03 19:38 CEST
Nmap scan report for 192.168.111.1
Host is up (0.00031s latency).
Not shown: 97 closed ports
PORT    STATE SERVICE
53/tcp  open  domain
81/tcp  open  hosts2-ns
444/tcp open  snpp
MAC Address: 08:00:27:D4:84:8B (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.111.15
Host is up (0.00024s latency).
All 100 scanned ports on 192.168.111.15 are filtered
MAC Address: 08:00:27:AF:16:BE (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.111.16
Host is up (0.00015s latency).
All 100 scanned ports on 192.168.111.16 are filtered
MAC Address: 08:00:27:AF:16:BE (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.111.2
Host is up (0.0000060s latency).
Not shown: 97 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Nmap done: 256 IP addresses (4 hosts up) scanned in 4.12 seconds
root@deb-s1:~#
```

- Alle Clients und Server im Subnetz 192.168.111.0 liefern Ihre Portlisten
- Nur die genutzten Ports von IPFire und deb-s1 sind offen
  - 192.168.111.1 – IPFIRE green – alles ok
  - 192.168.111.2 – deb-s1 – alles ok
  - 192.168.111.15 – Win10 – alles ok
  - 192.168.111.16 – debian – alles ok
  - (Als Virtualisierungssystem lief bei mir Virtualbox und nicht Hyper-V)

fail2ban

- Fail2ban ist ein Sicherheitsprogramm, welches alle Logins überwacht
- Mehrfache fehlerhafte Logins führen dazu, dass die ausführende IP-Adresse für eine festgelegte Zeit gesperrt wird
- Nachdem fail2ban eingerichtet ist, kann mit dem fail2ban-client die Login-Überwachung ausgeführt und Parameter verändert werden.
- Fail2ban ist ein sehr effektives Intrusion Detektion (IDS) und Intrusion Protection System (IPS) und sollte auf jedem Server aktiv sein
- Wir konzentrieren uns hier nicht auf die Sicherheit eines Dienstes, sondern auf alle Dienste, die mit f2b vor brute-force-Angriffen abgesichert werden können.

fail2ban

- **Installation auf deb-s1:**
  - `# apt install fail2ban`
- **Nach der Installation sollte geprüft werden, ob der Dienst funktioniert, z.B. mit:**
  - `# ps ax | grep fail2ban`
- **Natürlich besteht auch die Gefahr, sich selbst auszusperren, wenn man ein falsches Passwort eingibt. Deshalb ist es möglich, die aufrufende IP auf eine White-List zu setzen. Dazu später mehr**

fail2ban

```
root@deb-s1:~# apt install fail2ban
Paketlisten werden gelesen… Fertig
Abhängigkeitsbaum wird aufgebaut… Fertig
Statusinformationen werden eingelesen… Fertig
Vorgeschlagene Pakete:
  mailx monit sqlite3
Die folgenden NEUEN Pakete werden installiert:
  fail2ban
0 aktualisiert, 1 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.
Es müssen 451 kB an Archiven heruntergeladen werden.
Nach dieser Operation werden 2.142 kB Plattenplatz zusätzlich benutzt.
Holen:1 http://deb.debian.org/debian bullseye/main amd64 fail2ban all 0.11.2-2 [451 kB]
Es wurden 451 kB in 0 s geholt (3.790 kB/s).
Vormals nicht ausgewähltes Paket fail2ban wird gewählt.
(Lese Datenbank ... 80219 Dateien und Verzeichnisse sind derzeit installiert.)
Vorbereitung zum Entpacken von .../fail2ban_0.11.2-2_all.deb ...
Entpacken von fail2ban (0.11.2-2) ...
fail2ban (0.11.2-2) wird eingerichtet ...
Created symlink /etc/systemd/system/multi-user.target.wants/fail2ban.service → /lib/systemd/system/fail2ban.service.
Trigger für man-db (2.9.4-2) werden verarbeitet ...
root@deb-s1:~# ps ax | grep fail2ban
   6067 ?        Ssl    0:00 /usr/bin/python3 /usr/bin/fail2ban-server -xf start
   6149 pts/0    S+     0:00 grep fail2ban
root@deb-s1:~#
```

fail2ban

- Viele der weiteren Aktionen finden im Verzeichnis /etc/fail2ban statt. Daher ist ein Wechsel in dieses Verzeichnis sinnvoll

```
root@deb-s1:~# cd /etc/fail2ban/
root@deb-s1:/etc/fail2ban# ls
action.d        fail2ban.d  jail.conf  paths-arch.conf    paths-debian.conf
fail2ban.conf   filter.d    jail.d     paths-common.conf  paths-opensuse.conf
root@deb-s1:/etc/fail2ban#
```

- Im Unter-Verzeichnis filter.d findet man alle Konfigurationsdateien für die unterstützten Dienste (nächste Folie)

fail2ban

```
root@deb-s1:/etc/fail2ban# ls filter.d/
3proxy.conf                courier-auth.conf   haproxy-http-auth.conf  perdition.conf          solid-pop3d.conf
apache-auth.conf           courier-smtp.conf   horde.conf              phpmyadmin-syslog.conf  squid.conf
apache-badbots.conf        cyrus-imap.conf     ignorecommands          php-url-fopen.conf      squirrelmail.conf
apache-botsearch.conf      directadmin.conf    kerio.conf              portsentry.conf         sshd.conf
apache-common.conf         domino-smtp.conf    lighttpd-auth.conf      postfix.conf            stunnel.conf
apache-fakegooglebot.conf  dovecot.conf        mongodb-auth.conf       proftpd.conf            suhosin.conf
apache-modsecurity.conf    dropbear.conf       monit.conf              pure-ftpd.conf          tine20.conf
apache-nohome.conf         drupal-auth.conf    murmur.conf             qmail.conf              traefik-auth.conf
apache-noscript.conf       ejabberd-auth.conf  mysqld-auth.conf        recidive.conf           uwimap-auth.conf
apache-overflows.conf      exim-common.conf    nagios.conf             roundcube-auth.conf     vsftpd.conf
apache-pass.conf           exim.conf           named-refused.conf      screensharingd.conf     webmin-auth.conf
apache-shellshock.conf     exim-spam.conf      nginx-botsearch.conf    selinux-common.conf     wuftpd.conf
assp.conf                  freeswitch.conf     nginx-http-auth.conf    selinux-ssh.conf        xinetd-fail.conf
asterisk.conf              froxlor-auth.conf   nginx-limit-req.conf    sendmail-auth.conf      znc-adminlog.conf
bitwarden.conf             gitlab.conf         nsd.conf                sendmail-reject.conf    zoneminder.conf
botsearch-common.conf      grafana.conf        openhab.conf            sieve.conf
centreon.conf              groupoffice.conf    openwebmail.conf        slapd.conf
common.conf                gssftpd.conf        oracleims.conf          softethervpn.conf
counter-strike.conf        guacamole.conf      pam-generic.conf        sogo-auth.conf
root@deb-s1:/etc/fail2ban#
```

fail2ban

- Die Datei jail.conf enthält die globalen Einstellungen die für alle nutzbaren Dienste gleich sind
- Diese Datei soll man für das eigene System als jail.local kopieren und die jail.local an die gewünschten Einstellungen anpassen

```
root@deb-s1:/etc/fail2ban# cp jail.conf jail.local
root@deb-s1:/etc/fail2ban# nano jail.local
```

- Unter anderem kann man seinen eigenen IP-Adressraum oder nur eine IP-Adresse aus dem eigenen Subnetz aus den Bann-Einstellungen entfernen. Dazu aktiviert man den kommentierten Eintrag ignoreip

```
# "ignoreip" can be a list of IP addresses, CIDR masks or DNS hosts. Fail2ban
# will not ban a host which matches an address in this list. Several addresses
# can be defined using space (and/or comma) separator.
ignoreip = 127.0.0.1/8 ::1 192.168.111.0/24
```

fail2ban

- In unserer jail.local können wir die bantime (Zeit bis zum nächsten Eingabeversuch), die find-time (Zeit zwischen den Eingabeveruchen) und maxretry (Anzahl der möglichen Eingabeversuche bis die bantime aktiv ist, bestimmen
- Die Voreinstellungen sind :
  - bantime 10m (10 Minuten)
  - Findtime 10m (10 Minuten)
  - maxretry 5 (5 Fehlversuche sind möglich)
- Da wir selbst geschützt sind, ist eine bantime von einem Tag 86400 Sekunden (1d) (oder länger) und maxretry von 2 durchaus sinnvoll. Die findtime setze ich auf die doppelte bantime, also 172800 Sekunden (2d)

fail2ban

- Meine Einstellungen in der Datei jail.local:

```
# "bantime" is the number of seconds that a host is banned.
bantime  = 86400

# A host is banned if it has generated "maxretry" during the last "findtime"
# seconds.
findtime  = 172800

# "maxretry" is the number of failures before a host get banned.
maxretry = 2
```

- Es soll nicht unerwähnt bleiben, dass jeder Dienst separate Einstellungen erhalten kann, was den Rahmen hier aber sprengen würde
- Nach der Anpassung startet man den fail2ban.service neu:

```
root@deb-s1:/etc/fail2ban# systemctl reload fail2ban.service
root@deb-s1:/etc/fail2ban#
```

fail2ban

- Kontrolle, ob nicht autorisierte Anmeldeversuche stattgefunden haben:

```
root@deb-s1:/etc/fail2ban# fail2ban-client status
Status
|- Number of jail:      1
`- Jail list:   sshd
root@deb-s1:/etc/fail2ban# fail2ban-client status sshd
Status for the jail: sshd
|- Filter
|  |- Currently failed: 0
|  |- Total failed:     0
|  `- File list:        /var/log/auth.log
`- Actions
   |- Currently banned: 0
   |- Total banned:     0
   `- Banned IP list:
root@deb-s1:/etc/fail2ban#
```

- Solange die jeweilige Anzahl auf null steht und keine gebannte IP zu sehen ist, wurde das System nicht angegriffen

chage

- Mit chage stellt man ein, wie lange ein Benutzerkonto (Account) verwendet werden darf, wann das Passwort abläuft und verändert werden muss und wie oft das Passwort verändert werden darf bzw. muss.
- Unsere Benutzer Herr Korn, Frau Meier, Herr Huber und Frau Schmidt haben bisher keine Einschränkungen bezüglich ihres Accounts
- Mit chage ändert sich das ☺
- Die Vorgehensweise werde ich exemplarisch für Herrn Korn beschreiben
- Die Einstellungen sollten natürlich für alle User eingerichtet werden

chage

- Folgende Einstellungen sind vom Unternehmen gefordert (oder von uns)
  - Jeder Benutzer soll sein Passwort sofort nach dem ersten Login ändern müssen
  - Nach 3 Monaten (90 Tagen) läuft das bisherige Passwort aus
  - 8 Tage vor dem Ablaufdatum des Passworts soll der Benutzer die erste Warnung über den Ablauf erhalten
- Derartige Konstellationen gelten für dauerhaft Angestellte im Unternehmen
- Mit chage ist es aber auch möglich ein Benutzerkonto zu einem konkreten Datum ablaufen zu lassen, z.B. für Praktikanten
- Nur root darf die chage-Einstellungen vornehmen

chage

- Für alle Mitarbeiter gilt momentan, dass sie ihr Passwort nie ändern müssen (99999 Tage), was in der Datei /etc/shadow eingetragen ist
- Um die geforderten Werte zu aktivieren, geben Sie auf deb-s1 ein:

```
root@deb-s1:~# chage -d 0 -M 90 -W 8 korn
```

- Die Optionen:
  - -d 0 (Null) Der Benutzer muss sein Passwort sofort ändern, wenn er sich zum ersten Mal einwählt
  - -M 90 legt fest, nach wieviel Tagen das Passwort geändert werden muss.
  - -W 8 gibt an wann der Benutzer zum ersten Mal gewarnt wird, dass das Passwort abläuft. Der Standardwert sind 7 Tage
- Geben Sie die gleichen Werte für alle anderen Benutzer ein

chage

- chage legt die neuen Richtlinien zur Passwort-Änderung fest
- root meldet sich mit exit ab. Der Benutzer korn meldet sich an (su – korn)
- Anschließend muss korn sofort sein Passwort ändern
  - Die Wiederverwendung des alten Passworts funktioniert nicht
  - Ein einfaches Passwort wird nicht akzeptiert

```
root@deb-s1:~# chage -d 0 -M 90 -W 8 korn
root@deb-s1:~# exit
Abgemeldet
helmut@deb-s1:~$ su - korn
Passwort:  korn
Sie müssen Ihr Passwort sofort ändern (von root erzwungen).
Ändern des Passworts für korn.
Geben Sie das aktuelle Passwort ein: korn
Geben Sie ein neues Passwort ein: Passwort01
Geben Sie das neue Passwort erneut ein: Passwort01
korn@deb-s1:~$
```