Linux am Dienstag

Firewalls

Gestern und Heute

LAD: Firewalls

Freigabe:

Dieser Foliensatz darf explizit von jedem zum eigenen Vortrag genutzt werden.

LAD: Firewalls

Vorwort

Es ist hilfreich sich das Internet wie eine Großstadt mit Straßen, Hausnummern und Wohnungsnummer vorzustellen.

LAD: Firewalls

Damit wir verstehen, was die Aufgabe einer Firewall ist, müssen wir uns mit dem Netzwerk vertraut machen.

LAD: Firewalls

Ein Datenpaket soll von A nach B gesendet werden

LAD: Firewalls

Wenn man A und B direkt mit einem Netzwerkkabel verbindet, braucht man im Prinzip die Datenpakete nur auf das Kabel leiten, also in physische Form bringen.

LAD: Firewalls

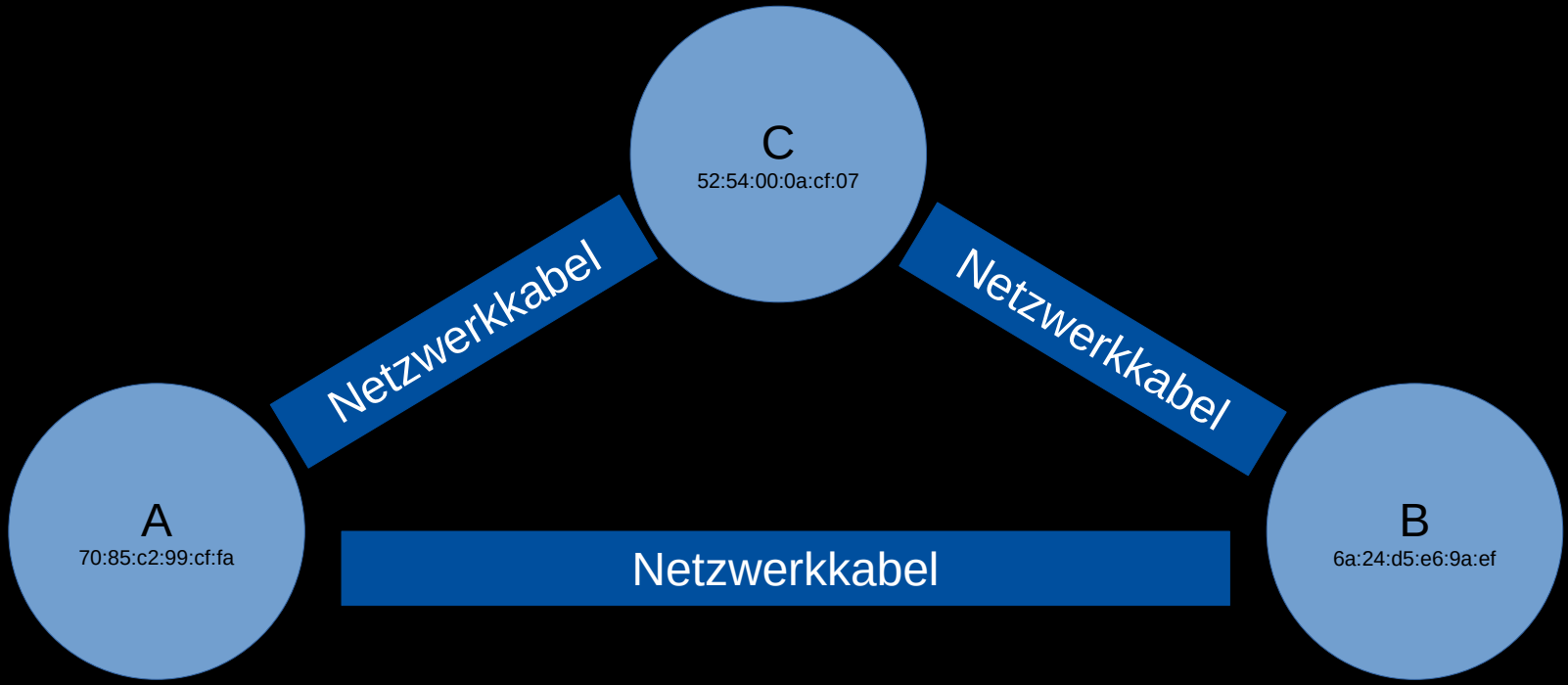
Da beide Seiten jederzeit Daten auf das Kabel schieben könnten, brauchte es eine eindeutige Kennung von wem, bzw. an wen, ein Paket gehen sollte:

Die MAC-Adresse wurde geboren

LAD: Firewalls

Jede Netzwerkkarte hat eine eindeutige MAC-Adresse

LAD: Firewalls



Für jeden weiteren PC bräuchte man immer mehr Kabel

LAD: Firewalls

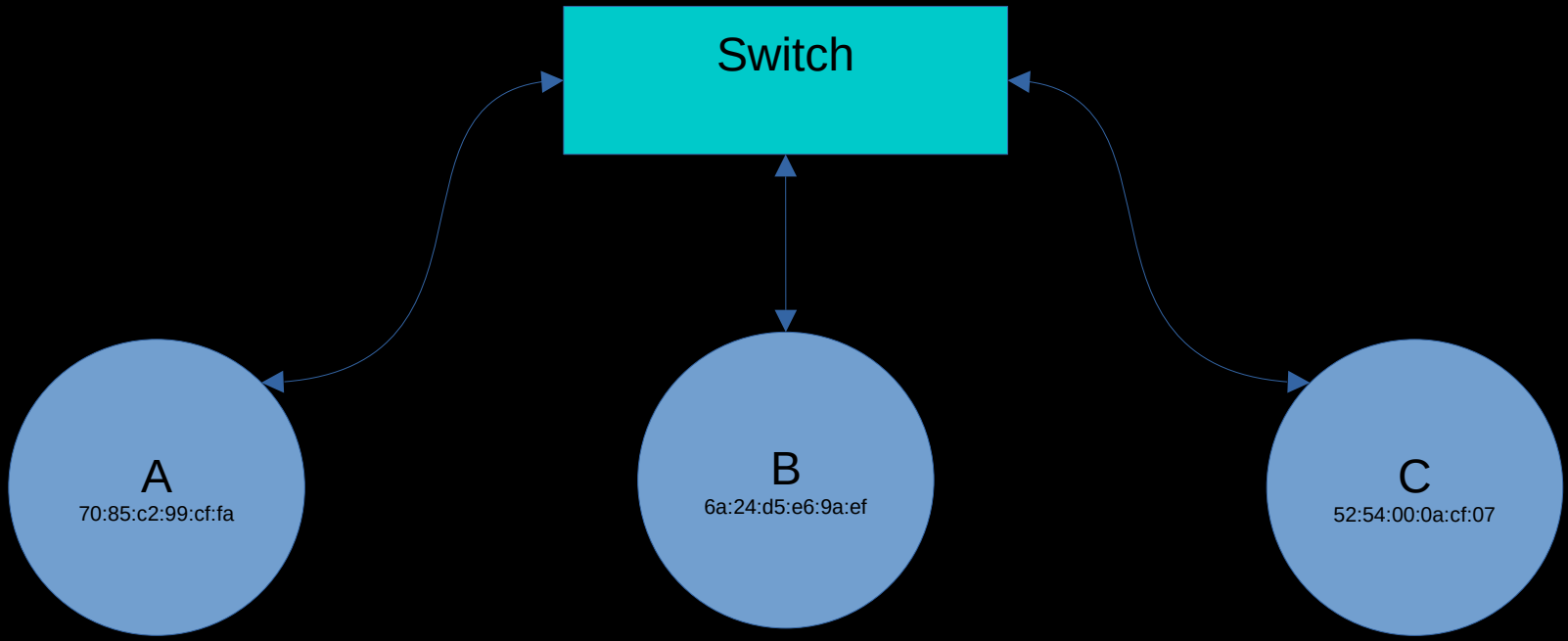
Jeder PC bräuchte soviele Netzwerkanschlüsse,
wie er mit PC’s reden will.

LAD: Firewalls

Jeder PC bräuchte soviele Netzwerkanschlüsse,
wie er mit PC’s reden will.

Das wäre sehr unpraktisch.

LAD: Firewalls



Also baut man einen Switch ein, der mit jedem PC verbunden wird.

LAD: Firewalls

Vorteil:

Jetzt braucht man nur noch soviele Kabel,
wie es PCs gibt.

LAD: Firewalls

Problem:

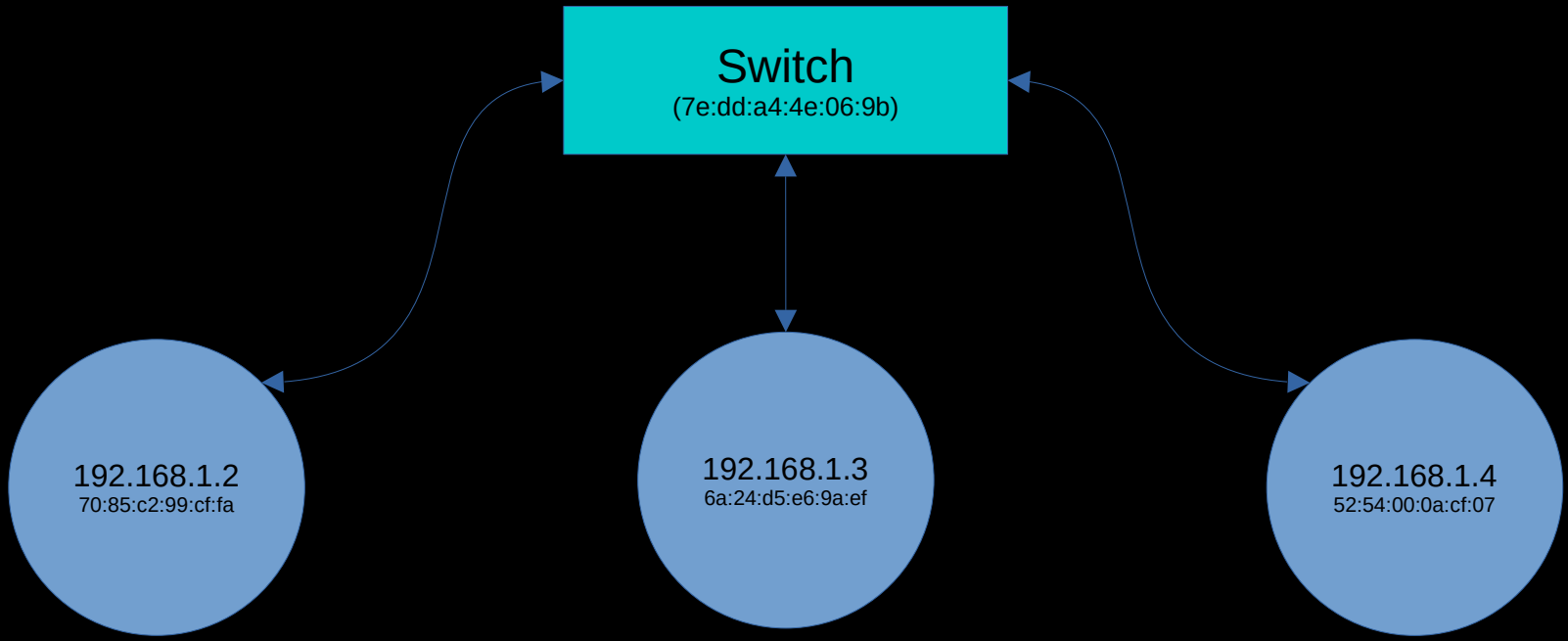
Wie bekommt jetzt die Daten von A nach B,
wenn ein Switch dazwischen ist?

LAD: Firewalls

Lösung:

jeder PC bekommt eine eindeutige Nummer

LAD: Firewalls



Die IP Adresse wurde geboren

Schlaumeier Disclaimer

Natürlich wissen wir auch,
daß die IP schon bei der Direktverbindung
von A und B benutzt wird,
aber nötig ist sie da technisch nicht.

LAD: Firewalls

Da der Switch Datenpakete von einem Anschluß zum nächsten sendet, aber nicht den Inhalt des Paketes kennt, brauchen wir einen Weg, wie ein PC die MAC von einer IP erfährt...

LAD: Firewalls

tada...das AR-Protokoll wurde erfunden.

Address Resolution Protocol (ARP)

LAD: Firewalls

Wir stellen „im Netz“ die Frage:

Wer auch immer die IP X.X.X.X hat,
schickt mir bitte die MAC dazu.

LAD: Firewalls

Der Switch verteilt das Fragepaket
an alle angeschlossenen Geräte
und transportiert (ggf.) die Antwort zurück.

LAD: Firewalls

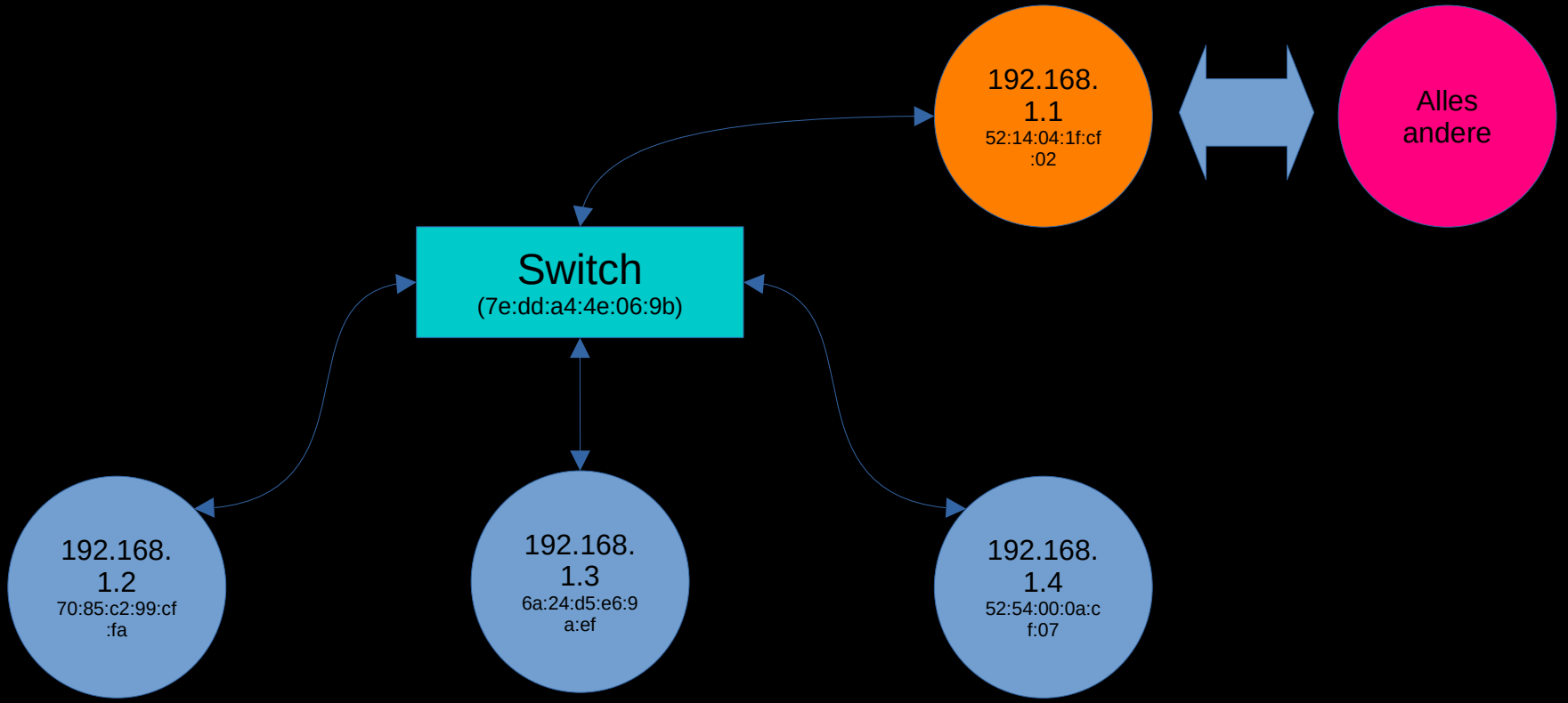
PC A kodiert dann das Paket an B/C mit der jeweiligen MAC und schickt es auf die Reise.

Der Switch sieht die MAC und weiß auf welches Kabel er das Paket schieben muß.

LAD: Firewalls

Der Weg ins Internet

LAD: Firewalls



Der Router wurde geboren

LAD: Firewalls

Was und Wie auch immer die Daten außerhalb des eigenen Netzes transportiert werden, ist den Netzwerkteilnehmern egal.

LAD: Firewalls

„Alles andere“

Es funktioniert alles nach genau diesem Prinzip,
welches aber ein bisschen „optimiert“ wird,
was die Routen von Aa nach Bb betrifft.

LAD: Firewalls

Begriffbestimmung

routebare IP – ist im ganzen Internet (WAN) erreichbar
Beispiel: 83.246.80.133 (LAD)

private IP – nur im eigenen privaten LAN
Beispiel: 192.168.178.1 (Fritz!Box)

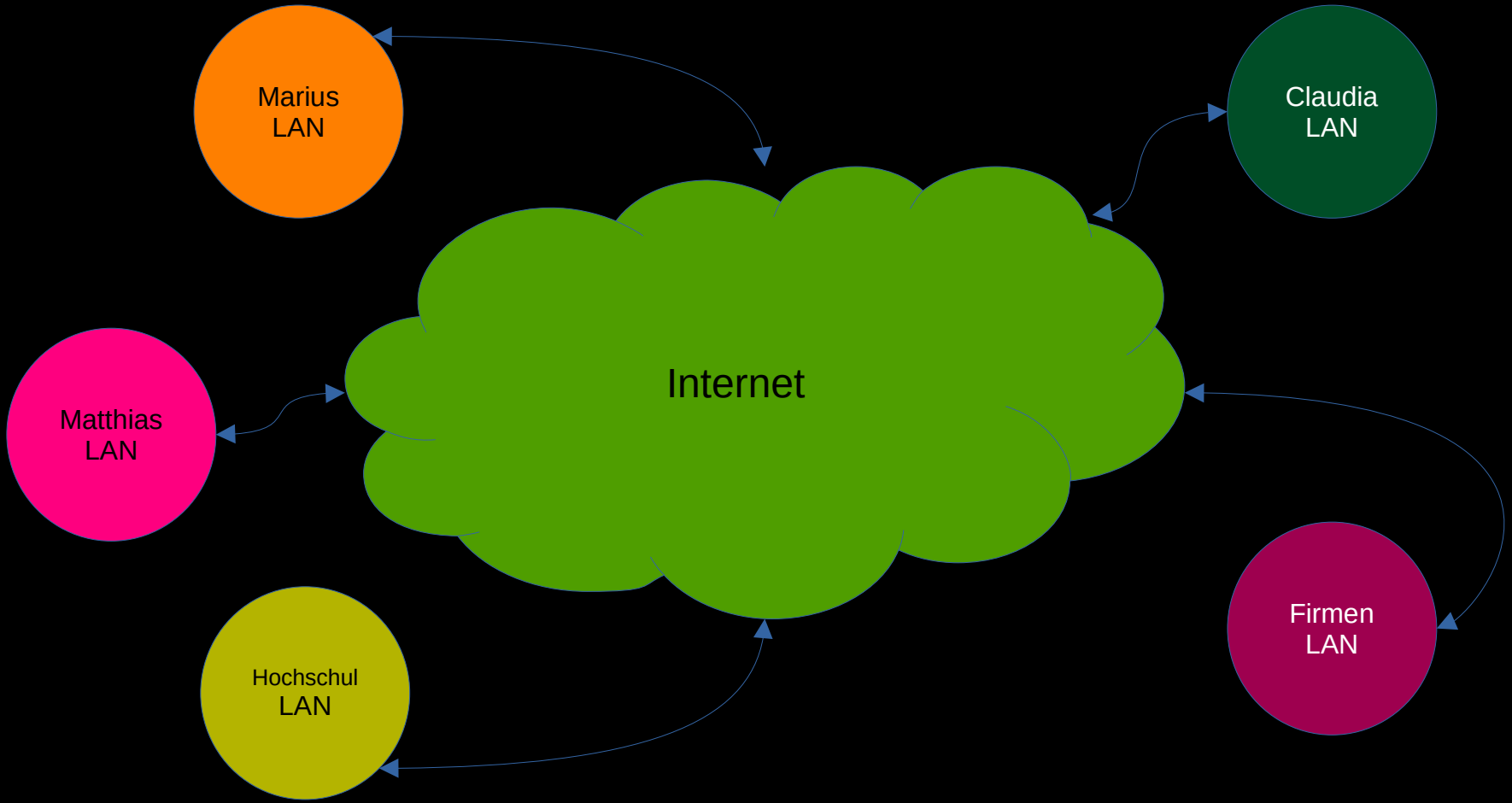
LAD: Firewalls

LAN – LOKAL AREA NETWORK

WAN – WIDE AREA NETWORK

Das „Internet“ ist ein WAN,
an das unzählige LANs angehängt sind.

LAD: Firewalls



LAD: Firewalls

Schlaumeier Intervention

„Haha! Wenn ich in meinem LAN eine Private IP habe, die im WAN nicht routbar ist, wie kommen die Daten aus dem WAN dann dahin?“

LAD: Firewalls

Antwort:

Eurer Router macht das für Euch,
indem er dem Datenpakt fürs Internet seine externe IP Adresse leiht und wenn die Antwort kommt, weiß, von welcher internen IP das kam.

LAD: Firewalls

Der Vorgang nennt sich NAT:

Network Address Translation

LAD: Firewalls

NAT wird später noch wichtig.

LAD: Firewalls

Es heißt zwar immer, dass Internet besteht nur aus Daten, aber eigentlich müßte es heißen, das Internet besteht aus lauter Datendiensten!

LAD: Firewalls

Jeder Netzwerkteilnehmer mit einer IP kann Datendienste erbringen.

Z.B.
Webserver, Fileserver, Datenbank, Mailserver

LAD: Firewalls

Jeder dieser Dienste hat einen eigenen PORT
auf dem er erreichbar ist.

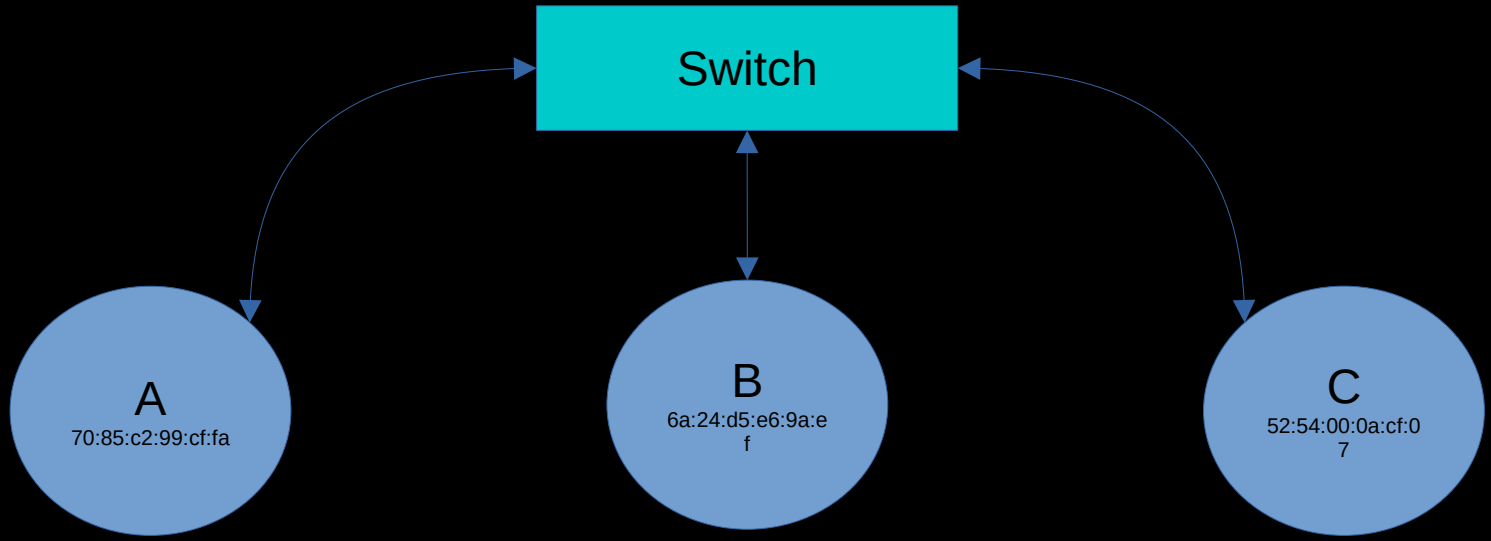
z.B. Webserver auf Port 80 und 443

LAD: Firewalls

Wir haben jetzt Seite 38 erreicht und nicht ein Wort über Firewalls und wozu man die braucht.

LAD: Firewalls

Kommen wir zurück zu unserem einfachen Netz von Seite 11:



LAD: Firewalls

Praktisches Beispiel

Der virenzerfressene Windows PC (C) soll nicht mehr auf die NAS (A) zugreifen können, um die Dateien nicht mit Viren verseuchen zu lassen.

LAD: Firewalls

Wir brauchen … eine Firewall!

LAD: Firewalls

Da „C“ unter der Kontrolle von Angreifern ist,
sonst wäre ja kein Virus drauf,
scheidet eine Zugriffsregelung auf C als
Sicherheitsmaßnahme aus.

LAD: Firewalls

Ergo, muß A selbst eine Firewall/Sicherheitsschranken-Regel aufstellen, in der er alle Pakete von C ablehnt.

LAD: Firewalls

Firewalls gibt viele...

LAD: Firewalls

Wir beschränken uns auf

IPTables & nfTables

LAD: Firewalls

Portzuordnung:

21 FTP
22 SSH
25 SMTP
53 DNS
80 HTTP
443 HTTPS

LAD: Firewalls

Was macht eine Firewall genau?

LAD: Firewalls

Begriffsklärung:

„RULE“ => eine REGEL
„RULES“ => mehrere REGELN
„CHAIN“ => Kette von Regeln
“POLICY“ => Verhalten, wenn die Chain passiert wurde und nichts festgelegt hat aka Default

LAD: Firewalls

Abstrakt

Die Firewall prüft jedes Paket, dass die Netzwerkkarte passiert auf die Einhaltung (oder Nichteinhaltung) von Regeln.

LAD: Firewalls

Damit die das gut kann, ist sie intern in verschiedene Ketten aufgeteilt.

Wenn eine Kette zu dem Schluß kam, sie sei mit dem Paket fertig, kann die nächste Kette arbeiten, was bedeutet, dass nicht jede Regel einer Kette geprüft werden muß bevor es weitergeht.

LAD: Firewalls

Außerdem gibt es noch Spezialketten,

z.B. INPUT / OUTPUT / FORWARD

die schon eine grobe Vorfilterung sind.

LAD: Firewalls

Außerdem kann man abstrakte Strukturen bauen, wie :

Jeder in Kette Zeta wird geblockt,
weil er zu SSH will, aber nicht darf.

Ist das Paket nicht für SSH gedacht,
braucht man die Kette gar nicht beachten.

LAD: Firewalls

Ketten können

Regeln und Unterketten

enthalten!

LAD: Firewalls

Die drei wichtigsten Hauptketten:

INPUT
OUTPUT
FORWARD

LAD: Firewalls

Durch INPUT müssen alle Pakete,
die in den Computer rein wollen.

Durch OUTPUT müssen alle Pakete,
die aus dem PC raus wollen.

FORWARD bearbeitet alles, was zwischen
Netzwerkinterfaces hin und her soll.

LAD: Firewalls

Für alle Beispiele gilt immer:

A = 192.168.1.2
B = 192.168.1.3
C = 192.168.1.4
...
Z = 192.168.1.27

A=NAS, B=Linux PC, C=Windows PC

LAD: Firewalls

Erstes Beispiel:

C darf nicht zu A

LAD: Firewalls

realisiert via IPTABLES auf A:

iptables -A INPUT -s 192.168.1.4 -j **DROP**

LAD: Firewalls

realisiert via nfTables auf A:

```
nft 'add rule ip filter INPUT ip saddr 192.168.1.3 counter accept'
nft 'add rule ip filter INPUT counter drop'
```

LAD: Firewalls

Beispiel 2

Ok, C ist jetzt komplett geblockt,
aber was ist, wenn NUR B
und sonst niemand auf A zugreifen können soll?

LAD: Firewalls

realisiert via IPTABLES auf A:

```
iptables -A INPUT -s 192.168.1.3 -j ACCEPT
iptables -A INPUT -j DROP
```

nimm alles von B an,
aber ALLES ANDERE wird abgelehnt.

LAD: Firewalls

realisiert via IPTABLES auf A:

```
iptables -A INPUT -s 192.168.1.3 -j ACCEPT
iptables -A INPUT -s 192.168.1.5 -j ACCEPT
iptables -A INPUT -j DROP
```

Nimm B an, Nimm D an,
aber ALLES ANDERE wird abgelehnt.

LAD: Firewalls

Beispiel DNS

Jetzt wollen wir allen erlauben,
auf den DNS Dienst zuzugreifen.

Ohne DNS könnte C z.b. nie ein gutes
Antivirenprogramm runterladen.

LAD: Firewalls

realisiert via IPTABLES auf A:

```
iptables -A INPUT -s 192.168.1.3 -j ACCEPT
iptables -A INPUT -s 192.168.1.5 -j ACCEPT
iptables -A INPUT -p tcp -m tcp --dport 53 -j ACCEPT
iptables -A INPUT -p udp -m udp --dport 53 -j ACCEPT
iptables -A INPUT -j DROP
```

Nimm B an, Nimm D an,
nimm alle für DNS an
aber ALLES ANDERE wird abgelehnt.

LAD: Firewalls

Unterschied DROP und REJECT

Jetzt wollen wir C sagen,
das er gesperrt ist,
aber alle anderen sollen nicht mal ahnen,
daß es A gibt.

LAD: Firewalls

realisiert via IPTABLES auf A:

```
iptables -A INPUT -s 192.168.1.4 -j REJECT --reject-with icmp-port-unreachable
iptables -A INPUT -j DROP
```

DROP verwirft ein ankommendes Paket kommentarlos,

REJECT meldet dem Sender, dass es verboten war.

LAD: Firewalls

Jetzt das Gleiche per nfTables

LAD: Firewalls

realisiert via nfTables auf A:

```
nft 'add rule ip filter INPUT ip saddr 192.168.1.3 counter accept'
nft 'add rule ip filter INPUT counter drop'
```

nimm alles von B an,
aber ALLES ANDERE wird abgelehnt.

LAD: Firewalls

realisiert via nfTables auf A:

```
nft 'add rule ip filter INPUT ip saddr 192.168.1.3 counter accept'
nft 'add rule ip filter INPUT ip saddr 192.168.1.5 counter accept'
nft 'add rule ip filter INPUT counter drop'
```

Nimm B an, Nimm D an,
aber ALLES ANDERE wird abgelehnt.

LAD: Firewalls

Beispiel DNS

Jetzt wollen wir allen erlauben,
auf den DNS Dienst zuzugreifen.

Ohne DNS könnte C z.b. nie ein gutes
Antivirenprogramm runterladen.

LAD: Firewalls

realisiert via nfTables auf A:

```
nft 'add rule ip filter INPUT ip saddr 192.168.1.3 counter accept'
nft 'add rule ip filter INPUT ip saddr 192.168.1.5 counter accept'
nft 'add rule ip filter INPUT tcp dport 53 counter accept'
nft 'add rule ip filter INPUT udp dport 53 counter accept'
nft 'add rule ip filter INPUT counter drop'
```

Nimm B an, Nimm D an,
nimm alle für DNS an
aber ALLES ANDERE wird abgelehnt.

LAD: Firewalls

Unterschied DROP und REJECT

Jetzt wollen wir C sagen,
das er gesperrt ist,
aber alle anderen sollen nicht mal ahnen,
daß es A gibt.

LAD: Firewalls

realisiert via IPTABLES auf A:

```
nft 'add rule ip filter INPUT ip saddr 192.168.1.4 counter REJECT with icmp type port-unreachable'
nft 'add rule ip filter INPUT counter drop'
```

DROP verwirft ein ankommendes Paket kommentarlos,

REJECT meldet dem Sender, dass es verboten war.

LAD: Firewalls

Bei DROP gibt es keine Antwort an den Absender. Er hat so keine Chance festzustellen, ob der PC für Ihn da ist, oder gar nicht existiert.

LAD: Firewalls

Bei REJECT gibt es eine Antwort an den Absender. Er bekommt so eine Fehlermeldung vom Ziel über seinen Versuch und kann passend handeln.

LAD: Firewalls

ABER

LAD: Firewalls

Weil er eine Antwort bekommt, weiß er, daß der PC da ist und antworten kann.

Wenn das Ziel des Senders ein DOS Angriff ist, darf man das dem Angreifer natürlich nicht mitteilen.

LAD: Firewalls

CHAINS / KETTEN

LAD: Firewalls

Für alle IPv4 iptables Anweisungen,
kann man auch ip6tables benutzen,
um Regeln für die IPv6 Firewall zu machen.

LAD: Firewalls

Wie legen wir eigene Ketten an?

LAD: Firewalls

Anlegen

iptables -N chainname

nft add chain <ip|ip6|inet> <tablename> <chainname> '{ type <chain_type> hook <hook_type> priority <priority_value> ; }'

LAD: Firewalls

Leeren

iptables -F chainname

nft flush chain <ip|ip6|inet> <table_name> <chain_name>

LAD: Firewalls

Löschen

iptables -X chainname

nft delete chain <ip|ip6|inet> <table_name> <chain_name>

LAD: Firewalls

Policy ändern

iptables -P chainname ACCEPT|DROP

nft chain <ip|ip6|inet> <table_name> <chain_name> '{ policy drop ; }'

LAD: Firewalls

Namen ändern

iptables -E alternname neuername

keine nft Entsprechung vorhanden

LAD: Firewalls

Statistiken zurücksetzen

iptables -Z chainname [rule]

nft reset counters <table_name> <ip|ip6|inet> <counter_name>

LAD: Firewalls

Anzeigen lassen

iptables -L [chainname]

(mit Hostnamen drin)

iptables -n -L [chainname]

(ohne Hostnameauflösung)

iptables -n -v -L [chainname]

(ohne Hostnameauflösung / mit Statistiken)

LAD: Firewalls

Anzeigen lassen

nft list ruleset

LAD: Firewalls

Was man meisten braucht:

APPEND/INSERT/DELETE

LAD: Firewalls

Beispiel APPEND:

```
$ iptables -N test
$ iptables -A test -s 192.168.0.1 -j ACCEPT
$ iptables -A test -s 192.168.0.2 -j DROP
```

```
# iptables -L test -n
Chain test (0 references)
target     prot opt source               destination
ACCEPT     all  --  192.168.0.1          0.0.0.0/0
DROP       all  --  192.168.0.2          0.0.0.0/0
```

LAD: Firewalls

Beispiel INSERT:

```
$ iptables -N test
$ iptables -A test -s 192.168.0.1 -j ACCEPT
$ iptables -A test -s 192.168.0.2 -j DROP
$ iptables -I test -s 192.168.0.3 -j DROP

# iptables -L test -n
Chain test (0 references)
target     prot opt source               destination
DROP       all  --  192.168.0.3          0.0.0.0/0
ACCEPT     all  --  192.168.0.1          0.0.0.0/0
DROP       all  --  192.168.0.2          0.0.0.0/0
```

LAD: Firewalls

Beispiel DELETE:

```
$ iptables -N test
$ iptables -A test -s 192.168.0.1 -j ACCEPT
$ iptables -A test -s 192.168.0.2 -j DROP
$ iptables -I test -s 192.168.0.3 -j DROP
$ iptables -D test -s 192.168.0.1 -j ACCEPT

# iptables -L test -n
Chain test (0 references)
target     prot opt source               destination
DROP       all  --  192.168.0.3          0.0.0.0/0
DROP       all  --  192.168.0.2          0.0.0.0/0
```

LAD: Firewalls

Bei nftables läuft das Spiel etwas anders ab,

da exportiert man sich den Ruleset,
ändert ihn und
läd ihn wieder rein.

LAD: Firewalls

```
nft list ruleset > ruleset.nft
vim ruleset.nft
nft -f ruleset.nft -c
nft -f ruleset
```

LAD: Firewalls

Nützliche Anweisungen

LAD: Firewalls

Mehr als einen Port angeben:

```
iptables -A INPUT -p tcp -m multiport --dports 25,26,587,465 -j smtp

nft 'add rule ip filter INPUT ip protocol tcp tcp dport { 25, 26, 587, 465 } counter jump smtp'
```

LAD: Firewalls

Mehr als einen Port angeben:

```
iptables -A INPUT -p tcp -m multiport --dports 25,26,587,465 -j smtp
iptables -A INPUT -p tcp -m multiport --dports 25,26,587,465 -j smtp2

nft 'add rule ip filter INPUT ip protocol tcp tcp dport { 25, 26, 587, 465 } counter jump smtp'
nft 'add rule ip filter INPUT ip protocol tcp tcp dport { 25, 26, 587, 465 } counter jump smtp2'
```

LAD: Firewalls

Alles was NICHT ist:

```
iptables -A INPUT ! -s 127.0.0.1/32 -p tcp -m tcp --dport 3306 -j REJECT --reject-with icmp-port-unreachable

nft 'add rule ip filter INPUT ip saddr != 127.0.0.1 tcp dport 3306 counter reject'
```

LAD: Firewalls

Man kann zusätzliche Module laden

LAD: Firewalls

Zuviele Verbindungen von einer IP?

LAD: Firewalls

Zuviele Verbindungen von einer IP?

Kein Problem:

```
iptables -A INPUT -p tcp --syn --dport 80 -m connlimit --connlimit-above 50 -j REJECT --reject-with tcp-reset
iptables -A INPUT -p tcp --syn --dport 443 -m connlimit --connlimit-above 50 -j REJECT --reject-with tcp-reset
```

LAD: Firewalls

Zuviele Verbindungen von einer IP?

Kein Problem:

```
iptables -A INPUT -p tcp --syn --dport 80 -m connlimit --connlimit-above 50 -j REJECT --reject-with tcp-reset
iptables -A INPUT -p tcp --syn --dport 443 -m connlimit --connlimit-above 50 -j REJECT --reject-with tcp-reset
```

Der Connectionlimiter macht das schon :)

LAD: Firewalls

nftables

```
nft 'add set ip filter connlimit0 { type ipv4_addr; flags dynamic; }'
nft 'add rule ip filter INPUT tcp dport 80 tcp flags syn / fin,syn,rst,ack add @connlimit0 { ip saddr ct count
over 50 } counter reject with tcp reset'
nft 'add set ip filter connlimit1 { type ipv4_addr; flags dynamic; }'
nft 'add rule ip filter INPUT tcp dport 443 tcp flags syn / fin,syn,rst,ack add @connlimit1 { ip saddr ct count
over 50 } counter reject with tcp reset'
```

LAD: Firewalls

Der User darf das nicht:

```
iptables -A OUTPUT ! -d 127.0.0.1/32 -p tcp -m owner ! --uid-owner 0-93 -m tcp --dport 25 -j DROP

  nft 'add rule ip filter OUTPUT ip daddr != 127.0.0.1 skuid != 0-93 tcp dport 25 counter drop'
```

LAD: Firewalls

Wieso „DROP“ und nicht „REJECT“ ?

LAD: Firewalls

Das „DROP“ sorgt dafür, daß der Prozess des Benutzers sehr lange warten muß.

Das garantiert die Aufmerksamkeit des NIDS oder des regulären Benutzers für das Problem.

LAD: Firewalls

Eine Frage haben wir nicht gestellt:

Wo kommt eigentlich das tables in iptables her?

LAD: Firewalls

Bislang kannten wir Ketten/Chains

als oberste Ebene,

LAD: Firewalls

Bislang kannten wir Ketten/Chains

als oberste Ebene,

aber das stimmt nicht.

LAD: Firewalls

Tabellen / Tables sind die oberste Stufe

LAD: Firewalls

filter:

Standard Tabelle
vordefinierte Ketten: **INPUT**, **FORWARD** und **OUTPUT**.

nat:

Diese Tabelle wird für Pakete benutzt die neue Verbindungen erzeugen
vordefinierte Ketten: **PREROUTING**, **INPUT**, **OUTPUT** und **POSTROUTING**

mangle:

Damit lassen sich spezial Operationen an Paketen vornehmen (selten)
vordefinierte Ketten: **PREROUTING**, **INPUT**, **OUTPUT** und **POSTROUTING**

raw:

Diese Tabelle greift früher, wenn NOTRACK im Spiel ist. **Braucht man nur ganz seltenst**.
vordefinierte Ketten: **PREROUTING**, **OUTPUT**

security:

Kommt direkt nach Filter zum Einsatz und ist für Mandatory Access Control (MAC) Regeln.
vordefinierte Ketten: **INPUT**, **FORWARD** und **OUTPUT**.

LAD: Firewalls

Wir haben bisher nur mit filter gearbeitet.

LAD: Firewalls

Kommen wir zu NAT

LAD: Firewalls

Network Address Translation

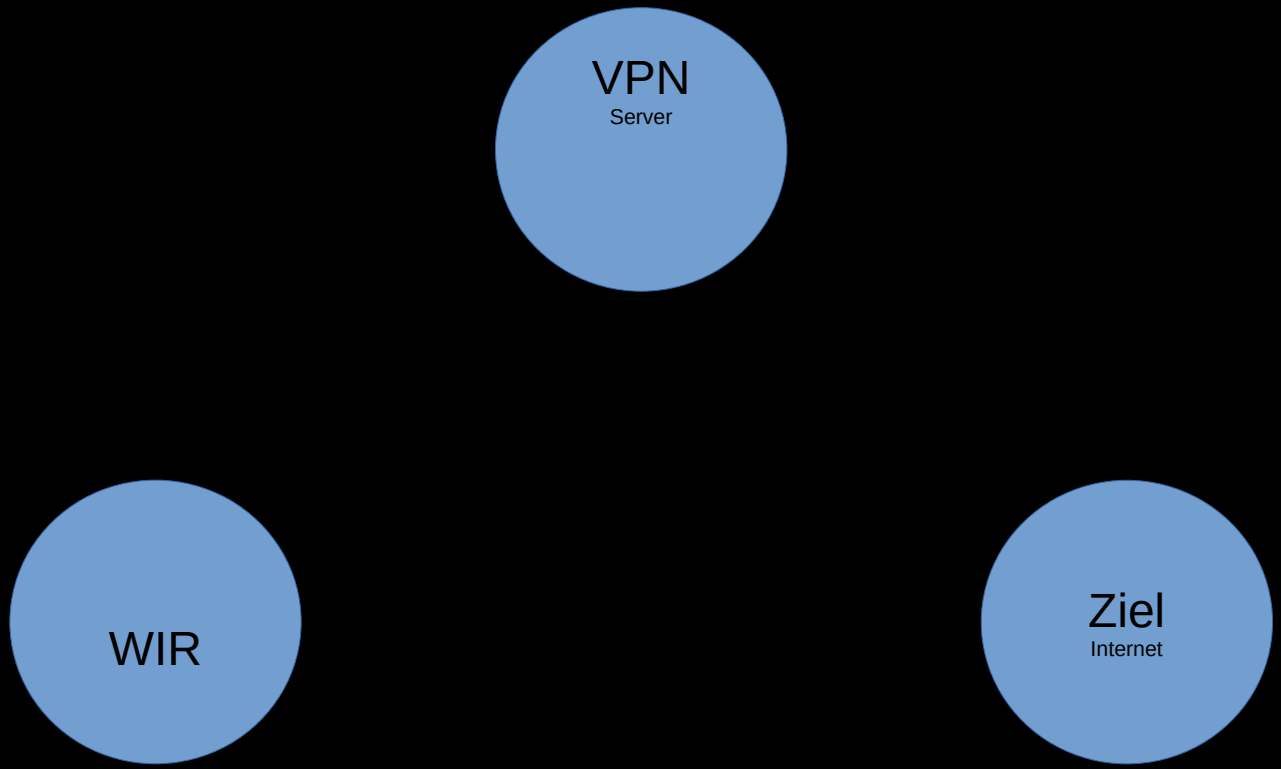
am Beispiel: SSH VPN

LAD: Firewalls

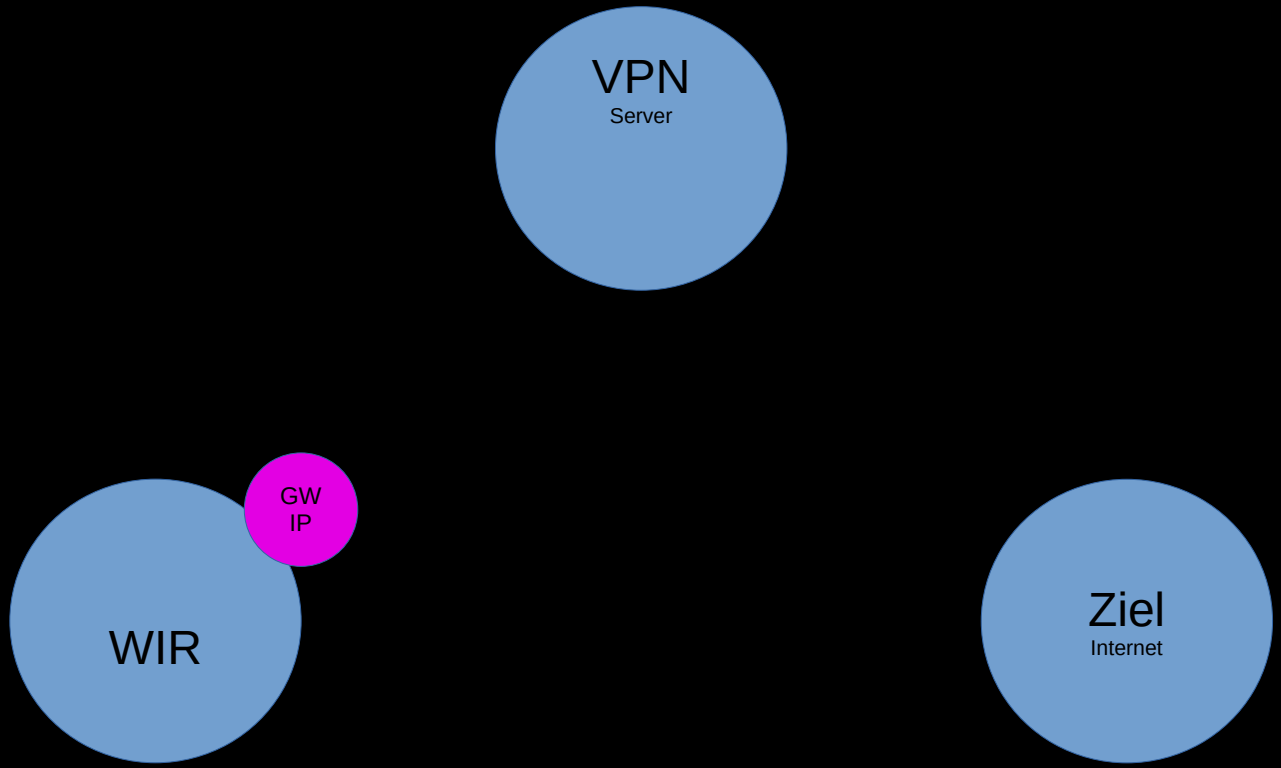
Hinweis

Wir machen der einfachheithalber IPv4,
es geht aber auch IPv6.

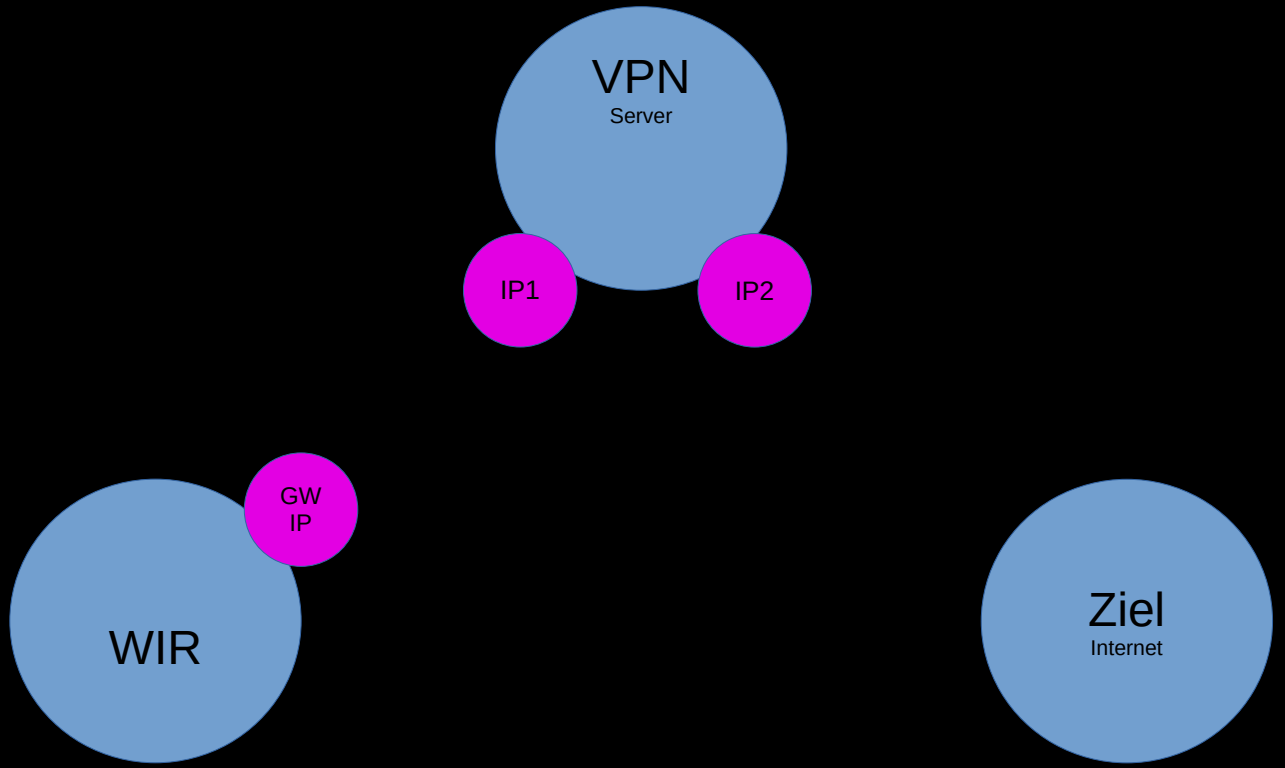
LAD: Firewalls



LAD: Firewalls

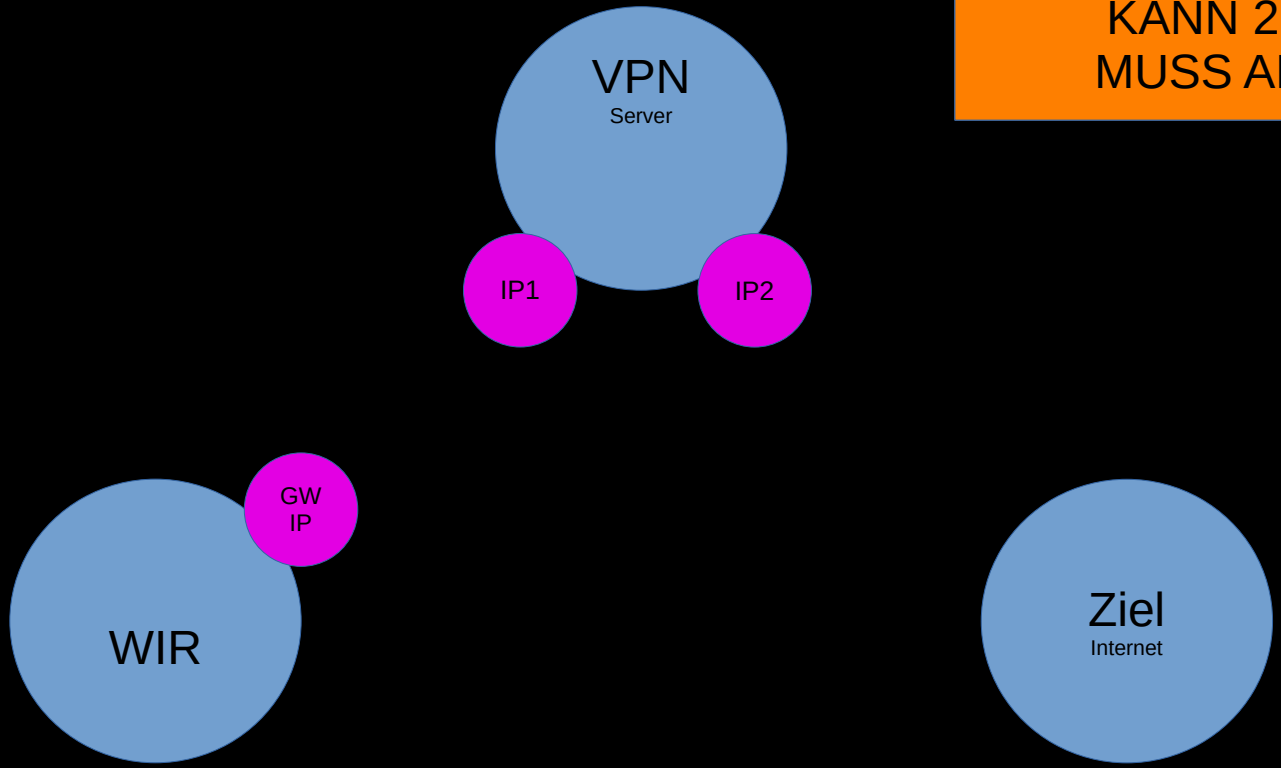


LAD: Firewalls

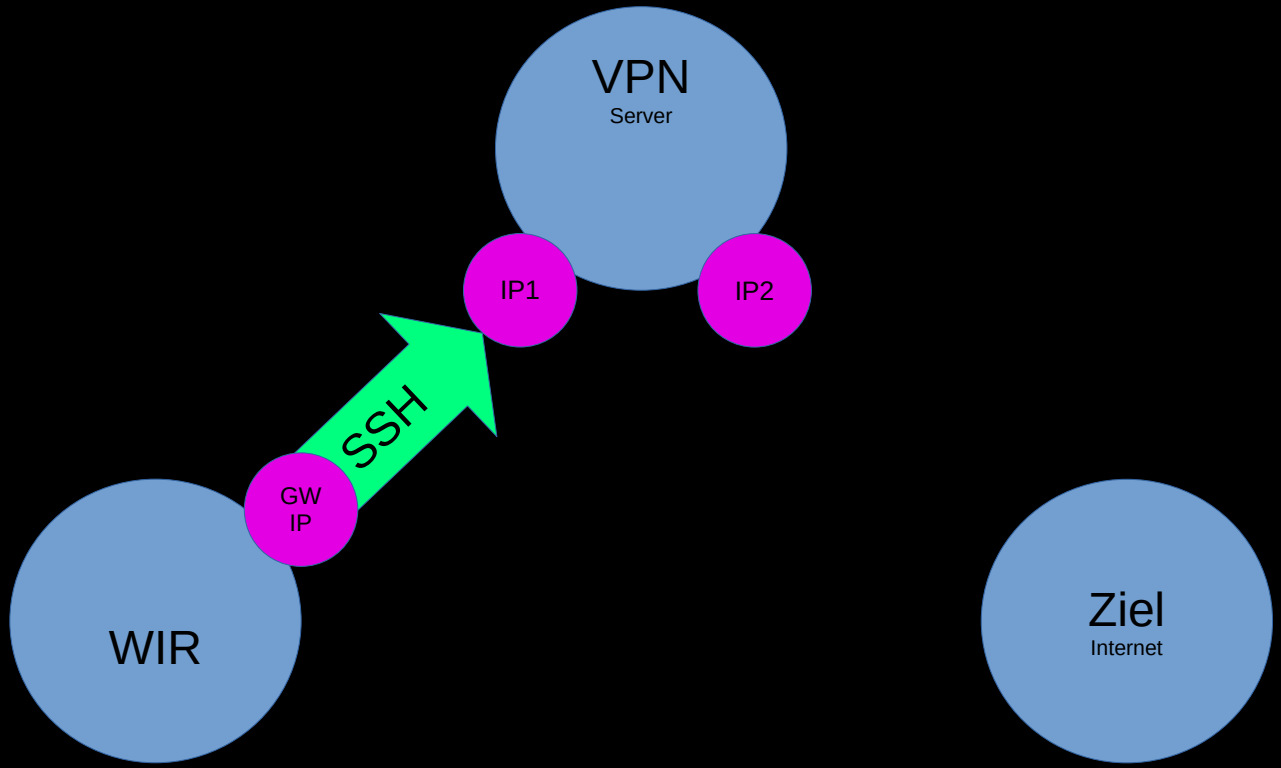


LAD: Firewalls

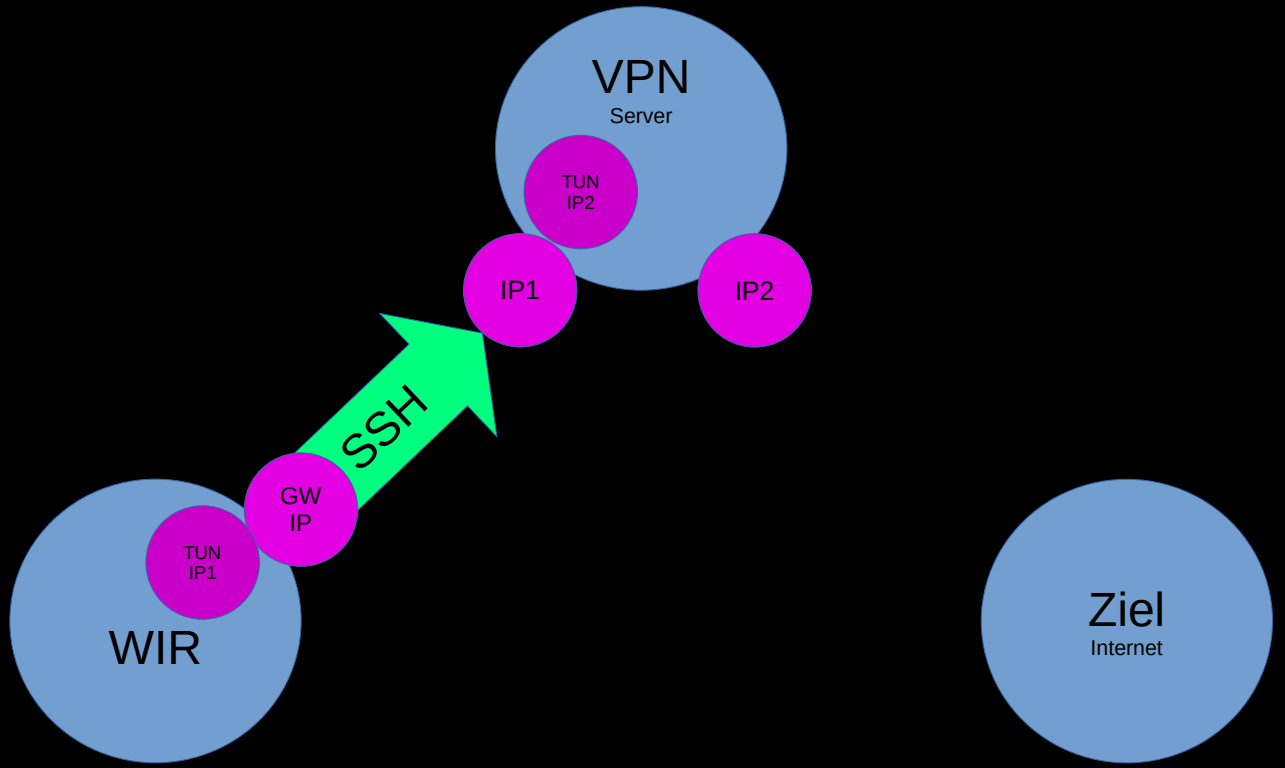
KANN 2 IPs haben,
MUSS ABER NICHT



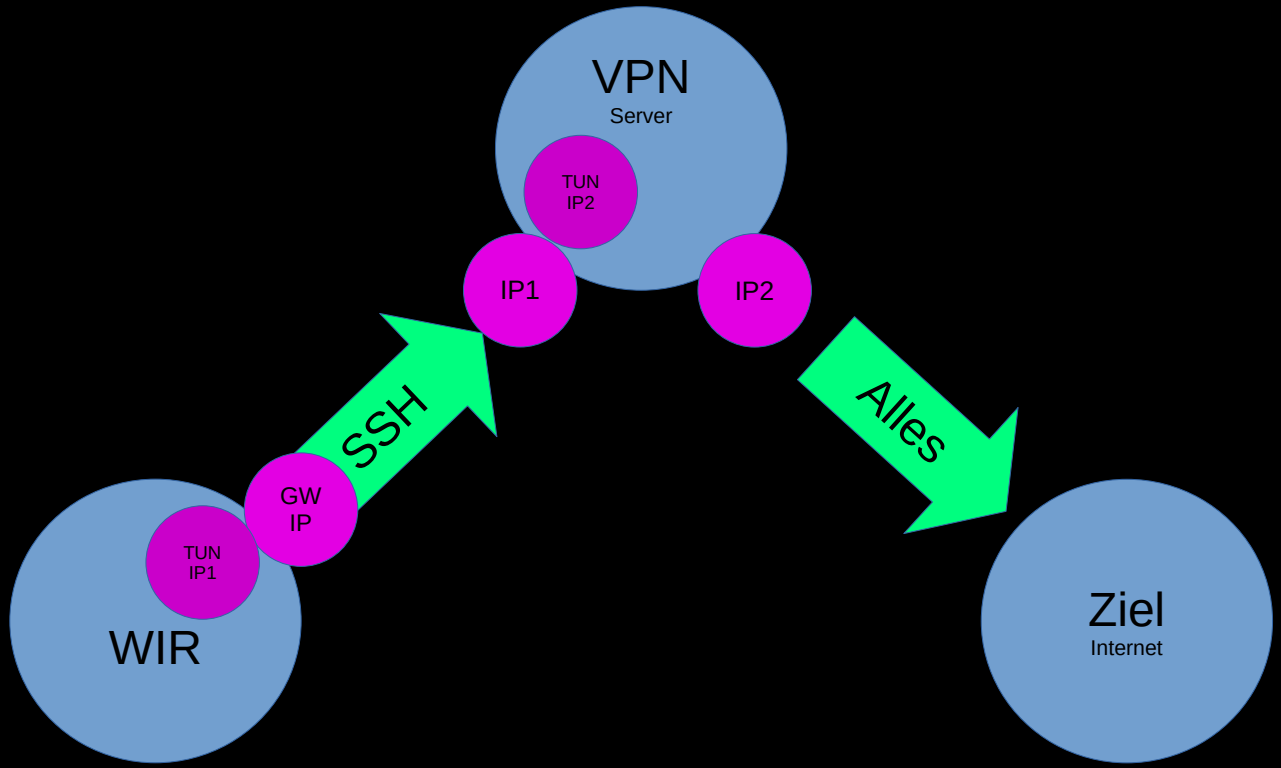
LAD: Firewalls



LAD: Firewalls



LAD: Firewalls



LAD: Firewalls

In Software sieht das so aus...

LAD: Firewalls

Schritt 1: Server

SSHD Optionen setzen:

GatewayPorts yes
PermitTunnel yes

LAD: Firewalls

Schritt 2: Client

```
modprobe tun
tunctl -t tun0
ssh -NTCf -w 0:0 root@2te.vpn.server.ip
```

LAD: Firewalls

Schritt 3: Server

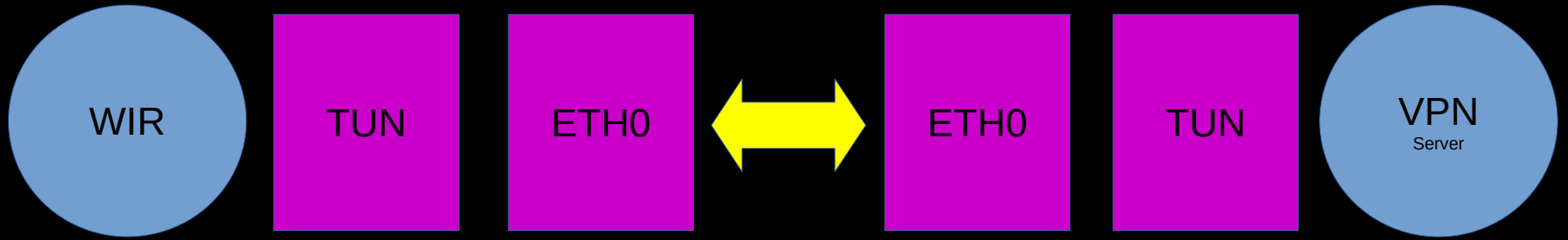
```
modprobe tun
tunctl -t tun0
ip link set tun0 up;
ip addr add 10.0.1.1/32 peer 10.0.1.2 dev tun0
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
alternativ für nftables:
nft 'add rule ip nat POSTROUTING oifname "eth0" counter masquerade'
```

LAD: Firewalls

Schritt 4: Client

```
ip link set tun0 up;
ip addr add 10.0.1.2/32 peer 10.0.1.1 dev tun0
route add 2te.vpn.server.ip gw alte.gw.ip;
route del default gw alte.gw.ip;
route add default gw 10.0.1.1 dev tun0;
```

LAD: Firewalls



LAD: Firewalls

LIVE Vorführung mit Kommentaren!

LAD: Firewalls

- wieso ist ip_forward nötig?

Damit Pakete von einem Netzwerkinterface zu anderen zu gelangen.

LAD: Firewalls

- was macht das NAT jetzt für uns?

Der VPN Server leiht dem Paket von uns seine IP, weil der kontaktierte Server ja nichts in unser Netz senden kann ODER SOLL!

LAD: Firewalls

- was ist eine DEFAULT Route?

Da gehen Pakete hin, wenn nichts andere definiert ist.

LAD: Firewalls

- was ist ein **G**ate**W**ay?

Da gehen Pakete hin, wenn sie das Netz verlassen sollen und keine eigene Netzwerkkarte das Zielnetz bieten kann.

LAD: Firewalls

Den Rest gibts LIVE bei

Linux am Dienstag