

Praktikum Rechnernetze

Versuch 5 vom 19.5.2020 Firewall

Protokoll Gruppe 1

Yannick Möller (ym018)

Bernd Maier (bm075)

Michael Vanhee (mv068)

Rebecca Mombrei (rm048)

Vorbereitung

Wir verbinden uns per SSH auf den Server **92.217.17.65** mit dem User root und dem uns zur Verfügung gestellten Passwort.

Server: 95.217.17.65

user: root

pw: 9bb50d4bda2c477ef0fc9766c4b9a025

Aufgabe 1: Wordpress konfigurieren

Auf ihrem Server ist Wordpress vorinstalliert / vorkonfiguriert. Führen Sie die Einrichtung durch und stellen Sie die korrekte Funktion sicher.

Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Don't worry, you can always change these settings later.


Site Title	<input type="text" value="Praktikum Rechnernetze Grupp"/>
Username	<input type="text" value="root"/> <small>Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.</small>
Password	<input type="password" value="sagichnicht"/> Hide <div>Strong</div> <p>Important: You will need this password to log in. Please store it in a secure location.</p>
Your Email	<input type="text" value="ym018@hdm-stuttgart.de"/> <small>Double-check your email address before continuing.</small>
Search Engine Visibility	<input checked="" type="checkbox"/> Discourage search engines from indexing this site <small>It is up to search engines to honor this request.</small>

Install WordPress

Anschließend per Klick auf **Install Wordpress** und nach einigen Sekunden kann man sich mit dem konfigurierten User einloggen.




Bernd Maier (bm075), Yannick Möller (ym018), Rebecca Mombrei (rm048), Michael Vanhee (mv068)

Über den Browser ist nun unter <http://95.217.17.65/> die folgende Seite aufrufbar.


Praktikum Rechnernetze Gruppe 1 Just another WordPress site [Sample Page](#)  Search

UNCATEGORIZED

Hello world!

 By root  May 19, 2020  1 Comment

Welcome to WordPress. This is your first post. Edit or delete it, then start writing!

 Edit

Archives

May 2020

Recent Posts

Aufgabe 2: Portscan durchführen

Überprüfen Sie mit einem Portscanner welche Ports an Ihrem Server öffentlich erreichbar sind. Welche Ports/Services sind das? Müssen diese Services öffentlich erreichbar sein?

Wir schauen erst einmal nach auf welcher Distribution sind: **uname -a**

Wir installieren **nmap** via **apt install nmap** und starten anschließend einen Scan:.

nmap localhost (als einfache Möglichkeit in der tmux Session, für die Windows User)

```
root@gruppe1:~# nmap localhost

Starting Nmap 7.40 ( https://nmap.org ) at 2020-05-19 14:42 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000017s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
3306/tcp  open  mysql
9100/tcp  open  jetdirect

Nmap done: 1 IP address (1 host up) scanned in 1.64 seconds
root@gruppe1:~#
```

Es sind mehrere Ports geöffnet:

- Port 22/tcp: ssh (nötig, da wir uns über ssh auf den Server verbinden)
- Port 80/tcp: http (Seitenaufruf über den Browser)
- Port 3306/tcp: mysql (Dies ist die an WordPress angebundene Datenbank; sie sollte von außen nicht erreichbar sein)
- Port 9100/tcp: Eigentlich ein WLAN-Drucker (auch dieser sollte nicht öffentlich verfügbar sein), in diesem Versuch jedoch missbraucht worden für einen anderen Dienst (Prometheus).

Ein **nmap 95.217.17.65** von extern auf den Server liefert das folgende Ergebnis.

```
blauwiggie@CXMP: ~
Last login: Tue May 19 14:32:53 on ttys001
> nmap 95.217.17.65
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-19 14:48 CEST
Nmap scan report for static.65.17.217.95.clients.your-server.de (95.217.17.65)
Host is up (0.050s latency).
Not shown: 993 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
135/tcp   filtered msrpc
139/tcp   filtered netbios-ssn
445/tcp   filtered microsoft-ds
3306/tcp  open  mysql
9100/tcp  open  jetdirect

Nmap done: 1 IP address (1 host up) scanned in 1.88 seconds
```

Folgende Ports werden durch eine FRITZ!Box "angehängt" und sind nicht relevant.

- Port 135/tcp: msrpc
- Port 139/tcp: netbios-ssn
- Port 445/tcp: microsoft-ds

Lediglich Port 22 und 80 (bei Webseiten per HTTPS auch 443) müssen hier geöffnet sein.

Bernd Maier (bm075), Yannick Möller (ym018), Rebecca Mombrei (rm048), Michael Vanhee (mv068)

Aufgabe 3: Blockieren von Services

Sie haben in Aufgabe 2 mindestens einen Service identifiziert, der nicht öffentlich verfügbar sein muss. Blockieren Sie den externen Zugriff auf diesen Service in Ihrer Firewall (Blacklist-Ansatz).

Mithilfe des Befehls **iptables -P INPUT ACCEPT** wird die Policy gesetzt, also das defaultmäßige Verhalten der INPUT Chain. ACCEPT ist hierbei der **Blacklist-Ansatz**: Alle Pakete werden grundsätzlich durchgelassen. Mithilfe von **iptables -L** lassen sich die verschiedenen Chains anzeigen:

```
root@gruppel:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source                               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                               destination
root@gruppel:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                               destination
```

Es können nun neue Regeln hinzugefügt werden, um den Zugriff auf die Ports zu blockieren.

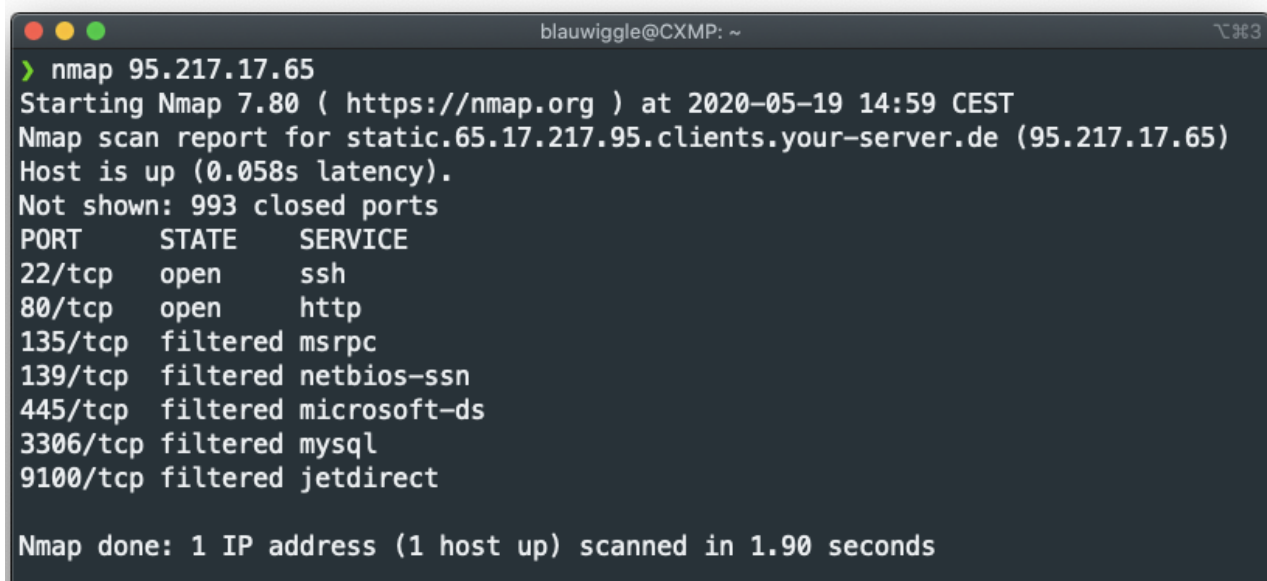
```
iptables -A INPUT -p tcp -d 95.217.17.65 --dport 3306 -j DROP
iptables -A INPUT -p tcp -d 95.217.17.65 --dport 9100 -j DROP
```

```
root@gruppel:~# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source                               destination      tcp dpt:3306
DROP       tcp  --  0.0.0.0/0                             95.217.17.65      tcp dpt:9100

Chain FORWARD (policy ACCEPT)
target     prot opt source                               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                               destination
```

Der STATE der Ports hat sich von **open** auf **filtered** geändert.



```
blauwiggle@CXMP: ~  
> nmap 95.217.17.65  
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-19 14:59 CEST  
Nmap scan report for static.65.17.217.95.clients.your-server.de (95.217.17.65)  
Host is up (0.058s latency).  
Not shown: 993 closed ports  
PORT      STATE      SERVICE  
22/tcp    open       ssh  
80/tcp    open       http  
135/tcp   filtered   msrpc  
139/tcp   filtered   netbios-ssn  
445/tcp   filtered   microsoft-ds  
3306/tcp  filtered   mysql  
9100/tcp  filtered   jetdirect  
  
Nmap done: 1 IP address (1 host up) scanned in 1.90 seconds
```

Aufgabe 4: Whitelist-Ansatz per Shell-Skript

Stellen Sie den gleichen Zustand der Firewall her wie in Aufgabe 3, allerdings verfolgen Sie jetzt den Whitelist-Ansatz.

Im Nachfolgenden unsere Whitelist-Konfiguration für den Server.

Zunächst verwerfen wir alle alten Regeln und setzen die Policy auf DROP (also Whitelist; alle Pakete werden grundsätzlich gelöscht).

Im Anschluss erlauben wir Port 80 & Port 22 um den SSH & HTTP-Zugriff explizit zu erlauben.

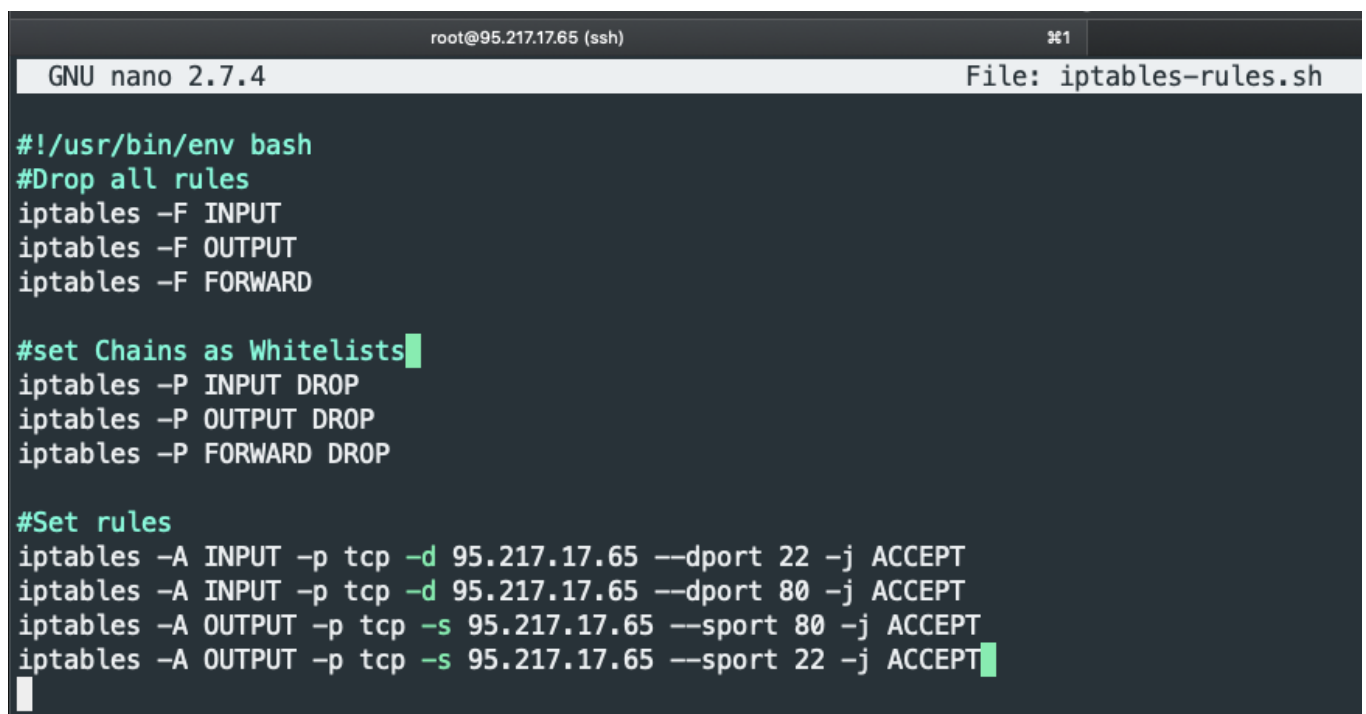
```
-----  
#!/usr/bin/env bash  
#Drop all rules  
iptables -F  
  
#set Chains as Whitelists  
iptables -P INPUT DROP  
iptables -P OUTPUT DROP  
iptables -P FORWARD DROP  
  
#set rules for allowed services  
iptables -A INPUT -p tcp -d 95.217.17.65 --dport 22 -j ACCEPT  
iptables -A INPUT -p tcp -d 95.217.17.65 --dport 80 -j ACCEPT  
iptables -A OUTPUT -p tcp -s 95.217.17.65 --sport 80 -j ACCEPT  
iptables -A OUTPUT -p tcp -s 95.217.17.65 --sport 22 -j ACCEPT
```

Bernd Maier (bm075), Yannick Möller (ym018), Rebecca Mombrei (rm048), Michael Vanhee (mv068)

```
#Drop all IPv6 rules
ip6tables -F

#set Chains for IPv6 as Whitelists
ip6tables -P INPUT DROP
ip6tables -P OUTPUT DROP
ip6tables -P FORWARD DROP

#set IPv6 rules for allowed services
ip6tables -A INPUT -p tcp -d 2a01:4f9:c010:7b40::1 --dport 22 -j ACCEPT
ip6tables -A INPUT -p tcp -d 2a01:4f9:c010:7b40::1 --dport 80 -j ACCEPT
ip6tables -A OUTPUT -p tcp -s 2a01:4f9:c010:7b40::1 --sport 80 -j ACCEPT
ip6tables -A OUTPUT -p tcp -s 2a01:4f9:c010:7b40::1 --sport 22 -j ACCEPT
```



```
root@95.217.17.65 (ssh)
GNU nano 2.7.4 File: iptables-rules.sh

#!/usr/bin/env bash
#Drop all rules
iptables -F INPUT
iptables -F OUTPUT
iptables -F FORWARD

#set Chains as Whitelists
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

#Set rules
iptables -A INPUT -p tcp -d 95.217.17.65 --dport 22 -j ACCEPT
iptables -A INPUT -p tcp -d 95.217.17.65 --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp -s 95.217.17.65 --sport 80 -j ACCEPT
iptables -A OUTPUT -p tcp -s 95.217.17.65 --sport 22 -j ACCEPT
```

Im Anschluss speichern wir das Script und testen es mit dem Befehl:

./iptables-rules.sh && sleep 60 && reboot

Sollte es ein unerwünschtes Verhalten gegeben haben, merken wir es und der Server startet anschließend neu. Damit verhindern wir zum Beispiel, dass wir uns vom Server aussperren.

Folgendes Änderungen in den Chains können beobachtet werden:

```
ssh root@95.217.17.65
root@95.217.17.65 (ssh)
root@gruppe1:~# iptables -L
Chain INPUT (policy DROP)
target     prot opt source                destination          tcp dpt:ssh
ACCEPT     tcp  --  anywhere              95.217.17.65         tcp dpt:ssh
ACCEPT     tcp  --  anywhere              95.217.17.65         tcp dpt:http

Chain FORWARD (policy DROP)
target     prot opt source                destination

Chain OUTPUT (policy DROP)
target     prot opt source                destination          tcp spt:http
ACCEPT     tcp  --  95.217.17.65         anywhere             tcp spt:ssh
ACCEPT     tcp  --  95.217.17.65         anywhere
root@gruppe1:~#
```

Ein Portscan mit nmap ergibt das folgende Ergebnis:

```
blauwiggle@CXMP: ~
> nmap 95.217.17.65
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-19 15:46 CEST
Nmap scan report for static.65.17.217.95.clients.your-server.de (95.217.17.65)
Host is up (0.014s latency).
Not shown: 995 closed ports
PORT      STATE      SERVICE
22/tcp    open       ssh
80/tcp    open       http
135/tcp    filtered  msrpc
139/tcp    filtered  netbios-ssn
445/tcp    filtered  microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 8.91 seconds
```


Aufgabe 5: ICMP und Prometheus Node-Exporter

Der Prometheus Node-Exporter liefert Metriken für Prometheus (<https://prometheus.io/>). Konfigurieren Sie ihre Firewall so, dass diese Metriken nur von Ihren IP-Adressen aus erreichbar sind. Dasselbe gilt für ICMP. Die Angriffsvektoren für ICMP sind zwar ziemlich eingeschränkt, trotzdem reicht es, wenn Sie in der Lage sind Probes an den Server zu senden.

Wir nutzen das Whitelist-Script aus Aufgabe 4 weiter und erlauben nun für 2 Gruppenmitglieder (1x mit IPv4 und 1x mit IPv6) den Zugriff auf den Port **9100** und **ICMP**:

```
#allow Port 9100 for ym018
iptables -A INPUT -p tcp -d 95.217.17.65 -s 109.250.136.134 --dport 9100 -j
ACCEPT
iptables -A OUTPUT -p tcp -s 95.217.17.65 -d 109.250.136.134 --sport 9100
-j ACCEPT

#allow ICMP for ym018
iptables -A INPUT -p icmp -d 95.217.17.65 -s 109.250.136.134 --icmp-type
255 -j ACCEPT
iptables -A OUTPUT -p icmp -s 95.217.17.65 -d 109.250.136.134 --icmp-type
255 -j ACCEPT
#-----
#allow Port 9100 for mv068
ip6tables -A INPUT -p tcp -d 2a01:4f9:c010:7b40::1 -s
2a02:8070:4c2:4200:e013:6628:a95f:4439 --dport 9100 -j ACCEPT
ip6tables -A OUTPUT -p tcp -s 2a01:4f9:c010:7b40::1 -d
2a02:8070:4c2:4200:e013:6628:a95f:4439 --sport 9100 -j ACCEPT

#allow ICMP for mv068
ip6tables -A INPUT -p icmpv6 -d 2a01:4f9:c010:7b40::1 -s
2a02:8070:4c2:4200:e013:6628:a95f:4439 --icmpv6-type 255 -j ACCEPT
ip6tables -A OUTPUT -p icmpv6 -s 2a01:4f9:c010:7b40::1 -d
2a02:8070:4c2:4200:e013:6628:a95f:4439 --icmpv6-type 255 -j ACCEPT
```

Der Zugriff über ICMPv6 funktionierte trotz der Regeln bisher nicht.

In den Chains stehen nun folgende Regeln:

IPv4:

```
Chain INPUT (policy DROP)
target     prot opt source                               destination
ACCEPT     tcp  --  0.0.0.0/0                            95.217.17.65      tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0                            95.217.17.65      tcp dpt:80
ACCEPT     tcp  --  109.250.136.134                      95.217.17.65      tcp dpt:9100
ACCEPT     icmp --  109.250.136.134                      95.217.17.65      icmp type 255

Chain FORWARD (policy DROP)
target     prot opt source                               destination

Chain OUTPUT (policy DROP)
target     prot opt source                               destination
ACCEPT     tcp  --  95.217.17.65                         0.0.0.0/0         tcp spt:80
ACCEPT     tcp  --  95.217.17.65                         0.0.0.0/0         tcp spt:22
ACCEPT     tcp  --  95.217.17.65                         109.250.136.134   tcp spt:9100
ACCEPT     icmp --  95.217.17.65                         109.250.136.134   icmp type 255
```

IPv6:

```
Chain INPUT (policy DROP)
target     prot opt source                               destination
ACCEPT     tcp  --  ::/0                                2a01:4f9:c010:7b40::1 tcp dpt:22
ACCEPT     tcp  --  ::/0                                2a01:4f9:c010:7b40::1 tcp dpt:80
ACCEPT     tcp  --  2a02:8070:4c2:4200:e013:6628:a95f:4439 2a01:4f9:c010:7b40::1 tcp dpt:9100
ACCEPT     icmpv6 --  2a02:8070:4c2:4200:e013:6628:a95f:4439 2a01:4f9:c010:7b40::1 ipv6-icmp type 255

Chain FORWARD (policy DROP)
target     prot opt source                               destination

Chain OUTPUT (policy DROP)
target     prot opt source                               destination
ACCEPT     tcp  --  2a01:4f9:c010:7b40::1 ::/0                        tcp spt:80
ACCEPT     tcp  --  2a01:4f9:c010:7b40::1 ::/0                        tcp spt:22
ACCEPT     tcp  --  2a01:4f9:c010:7b40::1 2a02:8070:4c2:4200:e013:6628:a95f:4439 tcp spt:9100
ACCEPT     icmpv6 --  2a01:4f9:c010:7b40::1 2a02:8070:4c2:4200:e013:6628:a95f:4439 ipv6-icmp type 255
```

Ein **ping** oder **tracert** vom Heimrechner von ym018 auf den Server ist nun möglich:

```
C:\Users\Yannick>ping 95.217.17.65
```

```
Ping wird ausgeführt für 95.217.17.65 mit 32 Bytes Daten:
```

```
Antwort von 95.217.17.65: Bytes=32 Zeit=52ms TTL=53
```

```
Antwort von 95.217.17.65: Bytes=32 Zeit=50ms TTL=53
```

```
Antwort von 95.217.17.65: Bytes=32 Zeit=53ms TTL=53
```

```
Antwort von 95.217.17.65: Bytes=32 Zeit=48ms TTL=53
```

```
Ping-Statistik für 95.217.17.65:
```

```
  Pakete: Gesendet = 4, Empfangen = 4, Verloren = 0  
  (0% Verlust),
```

```
Ca. Zeitangaben in Millisek.:
```

```
  Minimum = 48ms, Maximum = 53ms, Mittelwert = 50ms
```

```
C:\Users\Yannick>tracert 95.217.17.65
```

```
Routenverfolgung zu static.65.17.217.95.clients.your-server.de [95.217.17.65]  
Über maximal 30 Hops:
```

1	2 ms	2 ms	2 ms	fritz.box [192.168.178.1]
2	15 ms	18 ms	25 ms	stu1903aihr001.versatel.de [62.214.63.95]
3	20 ms	12 ms	17 ms	62.214.38.173
4	29 ms	27 ms	27 ms	62.214.37.202
5	28 ms	43 ms	33 ms	80.249.209.55
6	51 ms	50 ms	57 ms	core3.sto.hetzner.com [213.239.245.38]
7	52 ms	44 ms	42 ms	core32.hel1.hetzner.com [213.239.224.21]
8	48 ms	43 ms	45 ms	static.88-198-242-250.clients.your-server.de [88.198.242.250]
9	*	*	*	Zeitüberschreitung der Anforderung.
10	51 ms	51 ms	50 ms	10977.your-cloud.host [95.216.129.220]
11	*	*	*	Zeitüberschreitung der Anforderung.
12	63 ms	55 ms	65 ms	static.65.17.217.95.clients.your-server.de [95.217.17.65]

```
Ablaufverfolgung beendet.
```

Aufgabe 6: Besprechung, Musterlösung und Einbindung als System-Service

Erkenntnisse aus der Besprechung:

- Bei einem Fehler beim Ausführen des Skripts sollte die Ausführung komplett abgebrochen werden, damit keine Firewall mit unbekanntem State entsteht
- Es ist sinnvoll, Variablen zu definieren, um die Austauschbarkeit zu vereinfachen und Tippfehler zu vermeiden, außerdem verkürzen sich die Regeln dadurch etwas und sind leichter lesbar
- Dem Localhost sollte bei einer Whitelist alles erlaubt werden
- Die Output-Chain kann auf ACCEPT gesetzt werden, wenn wir allen Programmen auf unserem Server vertrauen, dies gibt uns den Vorteil, für Output keine zusätzlichen Regeln aufstellen zu müssen
- Wenn IPv6 nicht genutzt wird, sollte daran gedacht werden, die Chains auf DROP zu setzen, um nicht autorisierten Zugriff zu vermeiden, ansonsten müssen natürlich auch alle **ip6tables** entsprechend konfiguriert werden
- bei stateful Firewalls:
 - für alle neuen Verbindungen (NEW) werden Regeln festgelegt
 - danach kann für alle bereits bestehenden Verbindungen Freigabe erteilt werden

Aufsetzen eines System Service:

systemctl status iptables zeigt den aktuellen Status unseres Services an.

Wenn der Service mit **systemctl start iptables-rules.sh** aktiviert wird, wird das Skript direkt beim Booten des Servers ausgeführt.

Anhang: Komplettes Shell-Script

```
#!/usr/bin/env bash
#Drop all IPv4-rules
iptables -F

#set Chains as Whitelists
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

#Set rules
iptables -A INPUT -p tcp -d 95.217.17.65 --dport 22 -j ACCEPT
iptables -A INPUT -p tcp -d 95.217.17.65 --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp -s 95.217.17.65 --sport 80 -j ACCEPT
iptables -A OUTPUT -p tcp -s 95.217.17.65 --sport 22 -j ACCEPT

#allow Port 9100 for ym018
iptables -A INPUT -p tcp -d 95.217.17.65 -s 109.250.136.134 --dport 9100 -j
ACCEPT
iptables -A OUTPUT -p tcp -s 95.217.17.65 -d 109.250.136.134 --sport 9100
-j ACCEPT

#allow ICMP for ym018
iptables -A INPUT -p icmp -d 95.217.17.65 -s 109.250.136.134 --icmp-type
255 -j ACCEPT
iptables -A OUTPUT -p icmp -s 95.217.17.65 -d 109.250.136.134 --icmp-type
255 -j ACCEPT

#-----

#Drop all IPv6-Rules
ip6tables -F

#set Chains for IPv6 as Whitelists
ip6tables -P INPUT DROP
ip6tables -P OUTPUT DROP
ip6tables -P FORWARD DROP

#set IPv6 rules for allowed services
ip6tables -A INPUT -p tcp -d 2a01:4f9:c010:7b40::1 --dport 22 -j ACCEPT
ip6tables -A INPUT -p tcp -d 2a01:4f9:c010:7b40::1 --dport 80 -j ACCEPT
ip6tables -A OUTPUT -p tcp -s 2a01:4f9:c010:7b40::1 --sport 80 -j ACCEPT
ip6tables -A OUTPUT -p tcp -s 2a01:4f9:c010:7b40::1 --sport 22 -j ACCEPT
```

Bernd Maier (bm075), Yannick Möller (ym018), Rebecca Mombrei (rm048), Michael Vanhee (mv068)

```
#allow Port 9100 for mv068
iptables -A INPUT -p tcp -d 2a01:4f9:c010:7b40::1 -s
2a02:8070:4c2:4200:e013:6628:a95f:4439 --dport 9100 -j ACCEPT
iptables -A OUTPUT -p tcp -s 2a01:4f9:c010:7b40::1 -d
2a02:8070:4c2:4200:e013:6628:a95f:4439 --sport 9100 -j ACCEPT

#allow ICMP for mv068
iptables -A INPUT -p icmpv6 -d 2a01:4f9:c010:7b40::1 -s
2a02:8070:4c2:4200:e013:6628:a95f:4439 --icmpv6-type 255 -j ACCEPT
iptables -A OUTPUT -p icmpv6 -s 2a01:4f9:c010:7b40::1 -d
2a02:8070:4c2:4200:e013:6628:a95f:4439 --icmpv6-type 255 -j ACCEPT
```