Praktikum Rechnernetze

Protokoll zu Versuch 5 (Paketfilter-Firewall unter Linux)
von Gruppe 1

Jakob Waibel, Daniel Hiller, Elia Wüstner, Felix Pojtinger

2021-11-16

Inhaltsverzeichnis

1 Einführung **3**
1.1 Mitwirken 3
1.2 Lizenz 3

2 Wordpress Konfigurieren **4**

3 Portscan durchführen **7**

4 Blockieren von Services **8**

5 Whitelist-Ansatz per Shell-Skript **10**

6 ICMP und Prometheus Node-Exporter **11**

7 Besprechung, Musterlösung und Einbindung als System-Service **13**

1 Einführung

1.1 Mitwirken

Diese Materialien basieren auf Professor Kiefers "Praktikum Rechnernetze"-Vorlesung der HdM Stuttgart.

Sie haben einen Fehler gefunden oder haben einen Verbesserungsvorschlag? Bitte eröffnen Sie ein Issue auf GitHub (github.com/pojntfx/uni-netpractice-notes):

Abbildung 1: QR-Code zum Quelltext auf GitHub

Wenn ihnen die Materialien gefallen, würden wir uns über einen GitHub-Stern sehr freuen.

1.2 Lizenz

Dieses Dokument und der enthaltene Quelltext ist freie Kultur bzw. freie Software.

Abbildung 2: Badge der AGPL-3.0-Lizenz

Uni Network Practice Notes (c) 2021 Jakob Waibel, Daniel Hiller, Elia Wüstner, Felix Pojtinger

SPDX-License-Identifier: AGPL-3.0

2 Wordpress Konfigurieren

Auf ihrem Server ist Wordpress vorinstalliert / vorkonfiguriert. Lediglich die abschließende Einrichtung ist noch nicht erfolgt… Führen Sie die Einrichtung durch und stellen Sie die Funktion sicher. Rufen Sie dazu die IP der Servers in einem Web-Browser auf.

Zur Fertigstellung der Konfiguration muss zuerst folgender Dialog ausgefüllt werden:

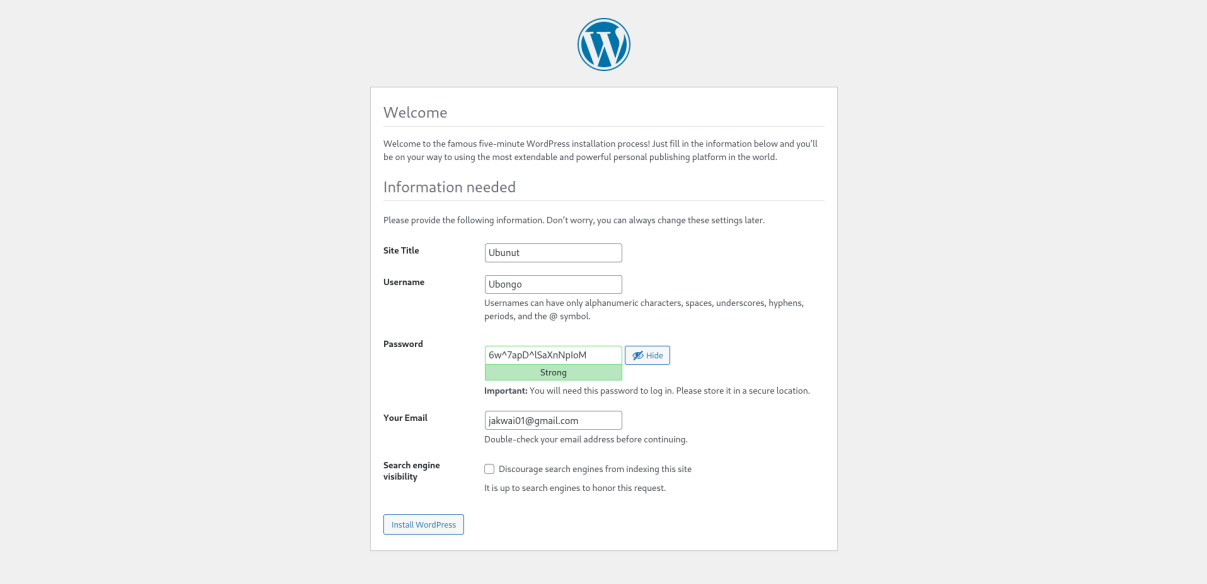

Abbildung 3: Wordpress-Einrichtungsbildschirm

Im darauffolgenden Dashboard kann im `Pages` Reiter eine neue page erstellt werden.

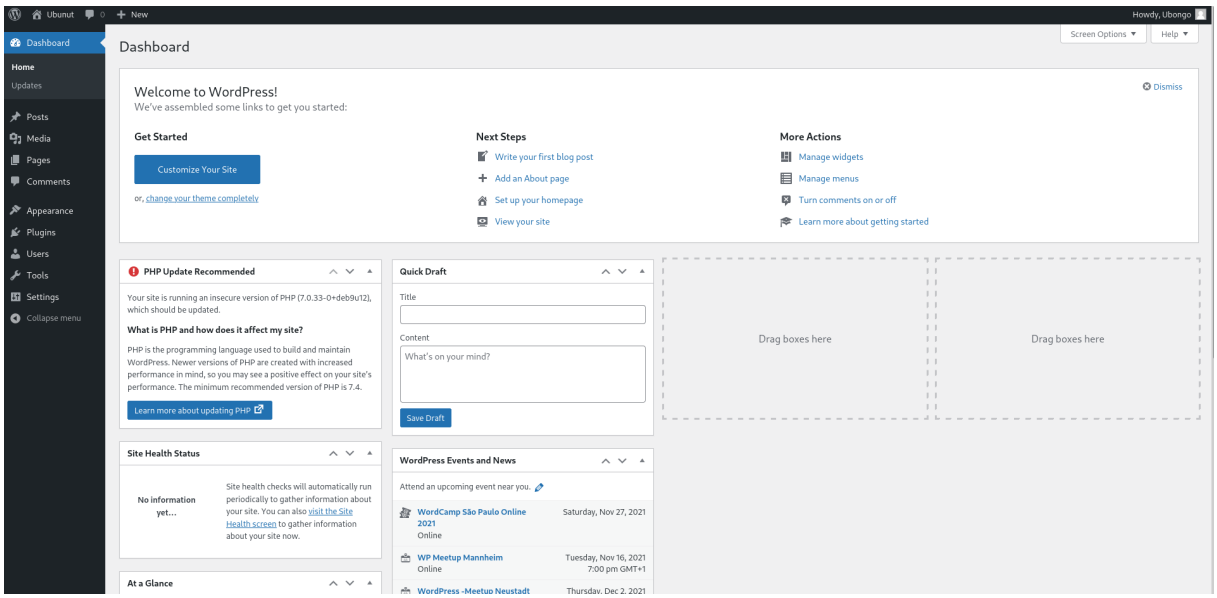

Abbildung 4: Wordpress-Dashboard

Diese Page kann nun mit arbiträrem Inhalt gefüllt werden.

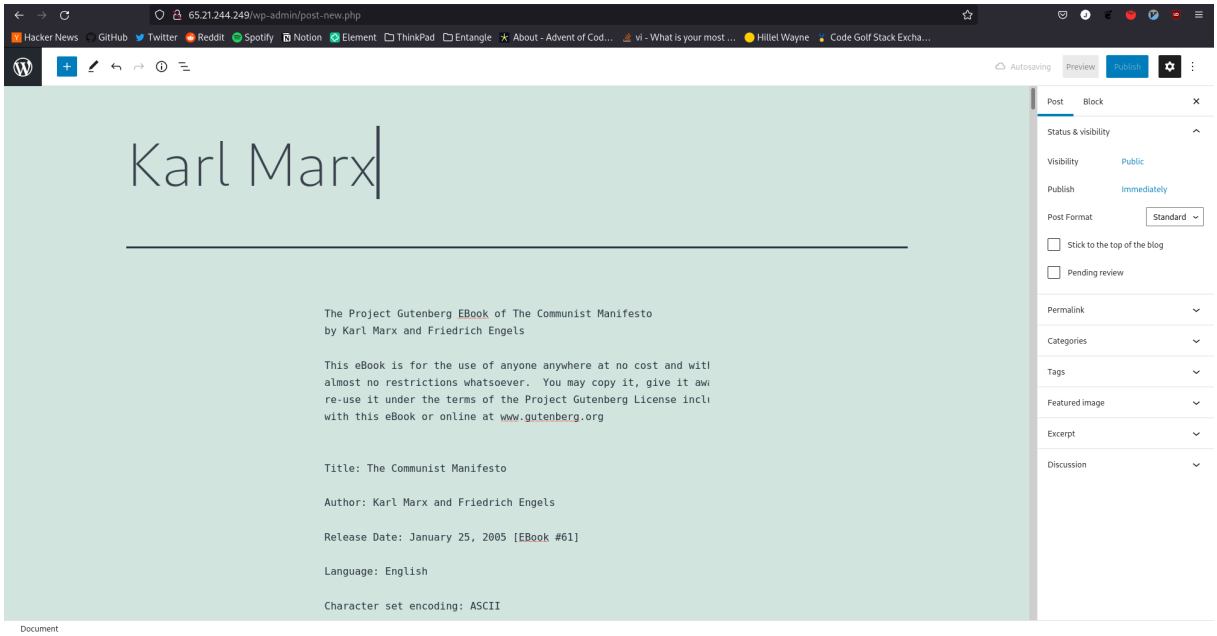

Abbildung 5: Wordpress-Post

Der Resultierende Post kann nun im Web gefunden werden. Eine Eingabe der IP führt auf eine kleine Übersicht mit diesem Post und einem weiteren “Hello-World” Post.

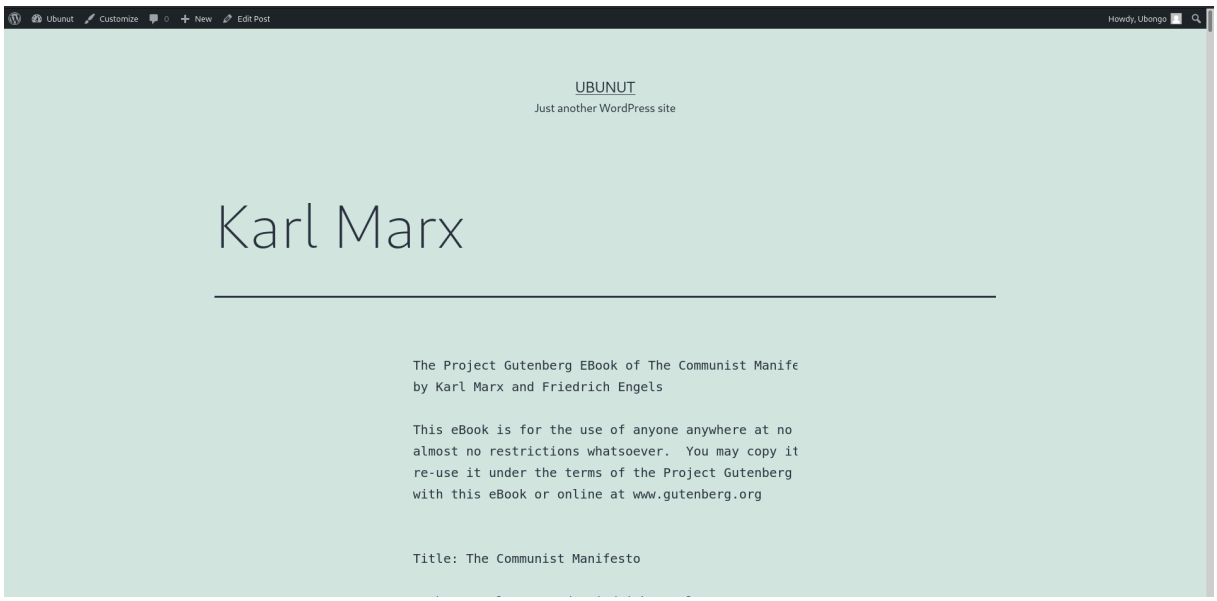

Abbildung 6: Wordpress-Home

3 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?

Zur Sicherheit starten wir bevor wir mit dem Portscanning beginnen den VPN unseres Vertrauens.

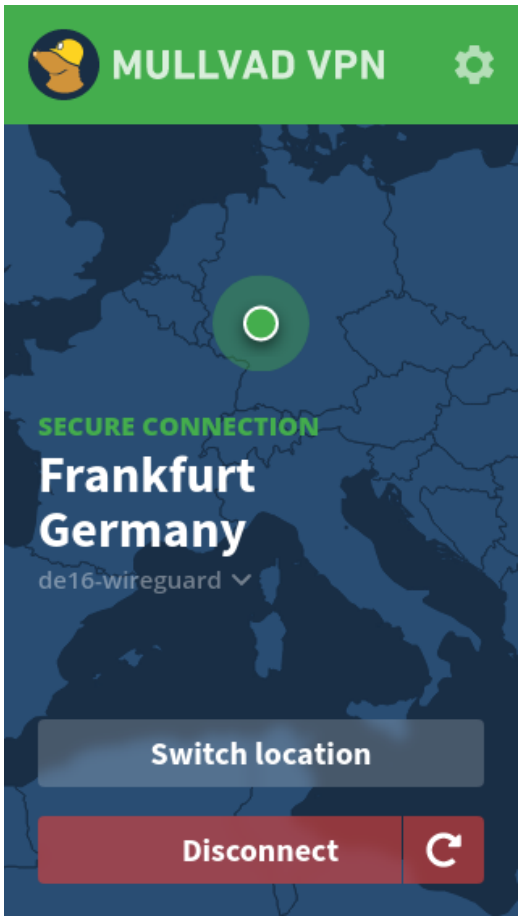

Abbildung 7: VPN-Einrichtung

`nmap` zeigt die offenen Ports:

```
1 $ sudo nmap 65.21.244.249
2 Starting Nmap 7.91 ( <https://nmap.org> ) at 2021-11-16 14:59 CET
3 sendto in send*ip*packet_sd: sendto(4, packet, 44, 0, 65.21.244.249,
      16) => Operation not permitted
4 Offending packet: TCP 10.108.48.108:61668 > 65.21.244.249:53 S ttl=55
      id=22537 iplen=44  seq=1930045695 win=1024 <mss 1460>
5 sendto in send*ip*packet_sd: sendto(4, packet, 44, 0, 65.21.244.249,
      16) => Operation not permitted
```

```
6  Offending packet: TCP 10.108.48.108:61669 > 65.21.244.249:53 S ttl=50
       id=21891 iplen=44  seq=1930111230 win=1024 <mss 1460>
7  Nmap scan report for static.249.244.21.65.clients.your-server.de
       (65.21.244.249)
8  Host is up (0.035s latency).
9  Not shown: 990 closed ports
10 PORT     STATE    SERVICE
11 22/tcp   open     ssh
12 25/tcp   filtered smtp
13 53/tcp   filtered domain
14 80/tcp   open     http
15 139/tcp  filtered netbios-ssn
16 445/tcp  filtered microsoft-ds
17 1900/tcp filtered upnp
18 2869/tcp filtered icslap
19 3306/tcp open     mysql
20 9100/tcp open     jetdirect
```

Ein Auszug des Wireshark-Captures zeigt hier zum Beispiel die Ergebnisse des Port-Scans.

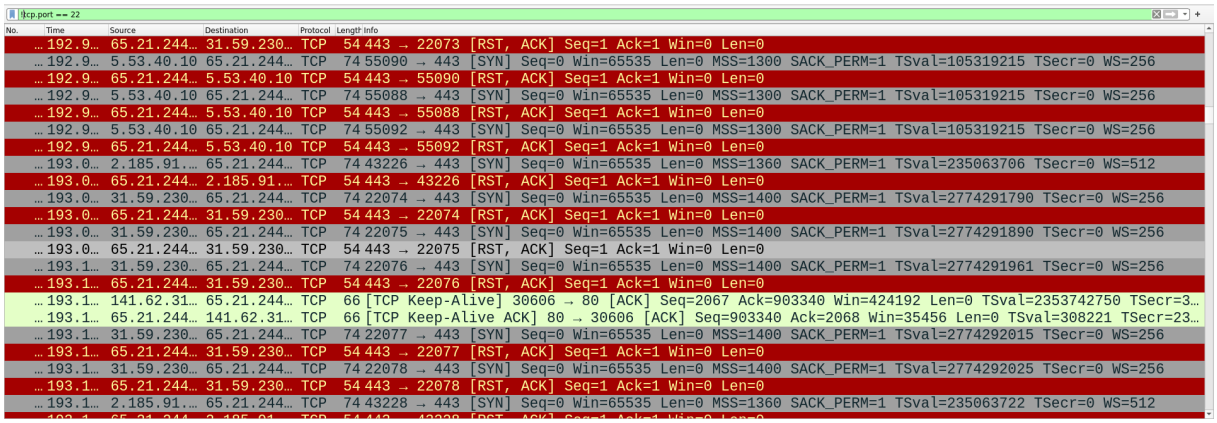

Abbildung 8: Wireshark-Capture von nmap

Da der Aufgabenstellung zu entnehmen ist, dass wir im Moment nur einen Webserver betreiben, können wir getrost allen externen Zugriff auf alle Ports bis auf Port 80 und Port 22 für den SSH-Zugriff blockieren. Für spätere Aufgabenstellungen können die Regeln dann angepasst werden.

4 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).

Der “Blacklist-Ansatz” bedeutet, dass mit einer ACCEPT policy und negativen Regeln gearbeitet wird, sodass alles erlaubt ist, sofern es nicht durch eine Regel explizit verboten wird.

Blocken aller Ports neben 22 und 80:

```
$ sudo iptables -F INPUT
$ sudo iptables -A INPUT -p tcp --dport 25 -j DROP
$ sudo iptables -A INPUT -p tcp --dport 53 -j DROP
$ sudo iptables -A INPUT -p tcp --dport 139 -j DROP
$ sudo iptables -A INPUT -p tcp --dport 445 -j DROP
$ sudo iptables -A INPUT -p tcp --dport 1900 -j DROP
$ sudo iptables -A INPUT -p tcp --dport 2869 -j DROP
$ sudo iptables -A INPUT -p tcp --dport 3306 -j DROP
$ sudo iptables -A INPUT -p tcp --dport 9100 -j DROP
$ sudo iptables -L INPUT
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  anywhere             anywhere             tcp dpt
    :9100
DROP       tcp  --  anywhere             anywhere             tcp dpt:
    smtp
DROP       tcp  --  anywhere             anywhere             tcp dpt:
    domain
DROP       tcp  --  anywhere             anywhere             tcp dpt:
    netbios-ssn
DROP       tcp  --  anywhere             anywhere             tcp dpt:
    microsoft-ds
DROP       tcp  --  anywhere             anywhere             tcp dpt
    :1900
DROP       tcp  --  anywhere             anywhere             tcp dpt
    :2869
DROP       tcp  --  anywhere             anywhere             tcp dpt:
    mysql
```

Check auf dem lokalen System:

Es kann erkannt werden, dass nun alle blockierten Ports als “filtered” angezeigt werden.

```
$ nmap 65.21.244.249
Starting Nmap 7.91 ( https://nmap.org ) at 2021-11-16 15:23 CET
Nmap scan report for static.249.244.21.65.clients.your-server.de
    (65.21.244.249)
Host is up (0.079s latency).
Not shown: 991 closed ports
PORT     STATE    SERVICE
22/tcp   open     ssh
25/tcp   filtered smtp
80/tcp   open     http
139/tcp  filtered netbios-ssn
445/tcp  filtered microsoft-ds
1900/tcp filtered upnp
```

```
2869/tcp filtered icslap
3306/tcp filtered mysql
9100/tcp filtered jetdirect

Nmap done: 1 IP address (1 host up) scanned in 3.39 seconds
```

5 Whitelist-Ansatz per Shell-Skript

Stellen Sie den gleichen Zustand der Firewall (Damit meine ich, dass die gleichen Services erreichbar sind) her wie in Aufgabe 3, allerdings verfolgen Sie jetzt den Whitelist-Ansatz.

Der "Whitelist-Ansatz" bedeutet, dass die default policy DROP verwendet wird und dass alles verboten ist, was nicht explizit durch eine Regel erlaubt wurde.

Inhalt von `iptables-rules.sh`:

```
# $HOME/iptables-rules.sh

#!/usr/bin/env bash

sudo iptables -F INPUT

sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

Zur Sicherheit wurde hier noch gecheckt, ob auch wirklich die richtigen Regeln angewandt wurden:

```
$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:
    ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:
    http
```

Nun wurde noch die Default Deny Rule aktiviert:

```
$ sudo iptables -P INPUT DROP
```

Und die korrekte Anwendung sichergestellt:

```
$ sudo iptables -L INPUT
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:
    ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:
    http
```

`nmap` zeigt nun auch die Ports nicht mehr als `filtered` an:

```
$ nmap 65.21.244.249
Starting Nmap 7.91 ( https://nmap.org ) at 2021-11-16 15:48 CET
Nmap scan report for static.249.244.21.65.clients.your-server.de
    (65.21.244.249)
Host is up (0.053s latency).
Not shown: 997 filtered ports
PORT   STATE  SERVICE
22/tcp open   ssh
53/tcp closed domain
80/tcp open   http

Nmap done: 1 IP address (1 host up) scanned in 11.56 seconds
```

6 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 (Nutzen Sie https://ifconfig.co/ um Ihre öffnetliche IP-Adresse in Erfahrung zu bringen). Das selbe 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.

Unsere IP-Addresse:

```
$ curl https://ifconfig.io
193.27.14.134
```

Nun müssen zwei `ACCEPT`-Rules erstellt werden; zuerst für den Prometheus Node-Exporter:

```
$ sudo iptables -A INPUT -p tcp --dport 9100 -j ACCEPT -s 193.27.14.134
$ curl http://65.21.244.249:9100
<html>
            <head><title>Node Exporter</title></head>
            <body>
            <h1>Node Exporter</h1>
            <p><a href="/metrics">Metrics</a></p>
            </body>
            </html>
$ nmap 65.21.244.249
Starting Nmap 7.91 ( https://nmap.org ) at 2021-11-16 15:54 CET
Nmap scan report for static.249.244.21.65.clients.your-server.de
    (65.21.244.249)
Host is up (0.039s latency).
Not shown: 996 filtered ports
```

```
PORT     STATE  SERVICE
22/tcp   open   ssh
53/tcp   closed domain
80/tcp   open   http
9100/tcp open   jetdirect

Nmap done: 1 IP address (1 host up) scanned in 5.74 seconds
```

Wie zu sehen ist, ist der Port von dieser öffentlichen IP zu erreichen. Von einem anderen Host ist dies nicht der Fall:

```
$ curl https://ifconfig.io
141.62.31.100
$ nmap 65.21.244.249
Starting Nmap 7.80 ( https://nmap.org ) at 2021-11-16 15:55 CET
Nmap scan report for static.249.244.21.65.clients.your-server.de
    (65.21.244.249)
Host is up (0.024s latency).
Not shown: 996 filtered ports
PORT    STATE  SERVICE
22/tcp  open   ssh
25/tcp  closed smtp
80/tcp  open   http
445/tcp closed microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 5.18 seconds
```

Nun die Rule für ICMP:

```
$ sudo iptables -A INPUT -p icmp -j ACCEPT -s 193.27.14.134
```

Nun lässt sich von unserem System aus der Host anpingen:

```
$ ping 65.21.244.249
PING 65.21.244.249 (65.21.244.249) 56(84) bytes of data.
64 bytes from 65.21.244.249: icmp_seq=1 ttl=52 time=51.3 ms
64 bytes from 65.21.244.249: icmp_seq=2 ttl=52 time=51.5 ms
64 bytes from 65.21.244.249: icmp_seq=3 ttl=52 time=52.5 ms
64 bytes from 65.21.244.249: icmp_seq=4 ttl=52 time=29.2 ms
64 bytes from 65.21.244.249: icmp_seq=5 ttl=52 time=32.4 ms
^C
--- 65.21.244.249 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 29.212/43.383/52.477/10.323 ms
```

Von einer anderen Workstation (`193.27.14.134`) aus ist dies nicht der Fall:

```
$ ping 65.21.244.249
PING 65.21.244.249 (65.21.244.249) 56(84) bytes of data.
^C
--- 65.21.244.249 ping statistics ---
```

```
5 9 packets transmitted, 0 received, 100% packet loss, time 8226ms
```

Diese beiden Regeln wurden noch zu `$HOME/iptables-rules.sh` hinzugefügt, damit sie im folgenden auch nach dem Reboot persistiert werden. Ebenso wurde die Default-Deny-Rule in das Skript mit aufgenommen.

7 Besprechung, Musterlösung und Einbindung als System-Service

Diese Aufgabe führen wir zusammen durch, dokumentieren Sie trotzdem die Schritte und Ergebnisse!

```
$ systemctl cat iptables
# /etc/systemd/system/iptables.service
[Unit]
Description=firewall service
Before=network.target
AssertPathExists=/root/iptables-rules.sh

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/root/iptables-rules.sh
StandardOutput=syslog
StandardError=syslog

[Install]
WantedBy=multi-user.target
$ sudo systemctl enable --now iptables.service
$ sudo systemctl reboot
```

Wie zu sehen ist, wurden die Regeln auch nach dem Reboot angewandt:

```
$ sudo iptables -L INPUT
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:
    ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:
    http
ACCEPT     icmp --  193.27.14.134        anywhere
ACCEPT     tcp  --  193.27.14.134        anywhere             tcp dpt
    :9100
```

Auf einer anderen Workstation:

```
$ nmap 65.21.244.249
Starting Nmap 7.80 ( https://nmap.org ) at 2021-11-16 16:14 CET
```

```
Nmap scan report for static.249.244.21.65.clients.your-server.de
    (65.21.244.249)
Host is up (0.049s latency).
Not shown: 996 filtered ports
PORT STATE SERVICE
22/tcp open ssh
25/tcp closed smtp
80/tcp open http
445/tcp closed microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 10.74 seconds
```

Auf der allowlisteten Workstation:

```
$ nmap 65.21.244.249
Starting Nmap 7.91 ( https://nmap.org ) at 2021-11-16 16:14 CET
Nmap scan report for static.249.244.21.65.clients.your-server.de
    (65.21.244.249)
Host is up (0.036s latency).
Not shown: 996 filtered ports
PORT STATE SERVICE
22/tcp open ssh
53/tcp closed domain
80/tcp open http
9100/tcp open jetdirect

Nmap done: 1 IP address (1 host up) scanned in 5.43 seconds
```

Letztendlich sieht unser überarbeitetes Skript wie folgt aus:

```
#!/usr/bin/env bash

# exit on error
set -e

IPT="/sbin/iptables"

$IPT -F INPUT

$IPT -A INPUT -p tcp --dport 22 -j ACCEPT
$IPT -A INPUT -p tcp --dport 80 -j ACCEPT
$IPT -A INPUT -p icmp -j ACCEPT -s 193.27.14.134
$IPT -A INPUT -p tcp --dport 9100 -j ACCEPT -s 193.27.14.134

# default targets
$IPT -P INPUT DROP # we want to block all incoming traffic (whitelist-
    approach)
$IPT -P OUTPUT ACCEPT # we trust the installed software
$IPT -P FORWARD DROP # we don't want to forward traffic at all
```