

Übung 4

Shehata Abd El Rahaman

Inhaltsverzeichnis

1	Automatische Adressvergabe	2
1.1	Aufbau des virtuellen Netzwerks – Basis	2
1.1.1	Konfiguration von Ü3	2
1.1.2	Ipv6 Adressen der Subinterfaces	2
1.2	Erstellung des IPv4 DHCP Pools	3
1.2.1	DHCP aktivieren	3
1.2.2	Frage 1	3
1.3	Stateless IPv6 Address Autoconfiguration (SLAAC)	5
1.3.1	IPv6 auto Adresse	5
1.3.2	Ping Test	6
1.3.3	Frage 2	7
2	Port Mirroring und Sniffing	8
2.1	Aufbau des virtuellen Netzwerks – Erweiterung	8
2.2	Konfiguration des Mirror Ports	8
2.3	Sniffen von DHCP Nachrichten	9
2.3.1	Frage 3	10
3	Wireshark Traces	11
3.1	Sniffing von ICMP Paketen	11
3.1.1	Ping	11
3.1.2	Ping mit Fragmentation	11
3.1.3	Traceroute\Tracert	13

1 Automatische Adressvergabe

1.1 Aufbau des virtuellen Netzwerks – Basis

1.1.1 Konfiguration von Ü3

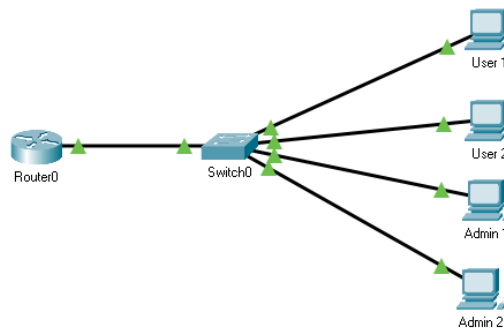


Abbildung 1: Netz

1.1.2 Ipv6 Adressen der Subinterfaces

```
1 Router>en
2 Router#conf t
3 Router(config)#ipv6 unicast-routing
4 Router(config)#int gig0/0.1
5 Router(config-subif)#ipv6 address 2a0c:2343:0:1::254/64
6 Router(config-subif)#exit
7 Router(config)#int gig0/0.2
8 Router(config-subif)#ipv6 address 2a0c:2343:0:2::254/64
9 Router(config-subif)#end
```

```
interface GigabitEthernet0/0.1
 encapsulation dot1Q 10
 ip address 192.168.1.254 255.255.255.0
 ipv6 address 2A0C:2343:0:1::254/64
!
interface GigabitEthernet0/0.2
 encapsulation dot1Q 20
 ip address 192.168.2.254 255.255.255.0
 ipv6 address 2A0C:2343:0:2::254/64
!
```

Abbildung 2: Running Config nach Ipv6 Einstellung

1.2 Erstellung des IPv4 DHCP Pools

1.2.1 DHCP aktivieren

```
1 Router>en
2 Router#conf t
3 Router(config)#ip dhcp excluded-address 192.168.1.1 192.168.1.10
4 Router(config)#ip dhcp excluded-address 192.168.2.1 192.168.2.10
5 Router(config)#ip dhcp pool Users
6 Router(dhcp-config)#network 192.168.1.0 255.255.255.0
7 Router(dhcp-config)#default-router 192.168.1.254
8 Router(dhcp-config)#dns-server 192.168.1.254
9 Router(dhcp-config)#exit
10 Router(config)#ip dhcp pool Admins
11 Router(dhcp-config)#network 192.168.2.0 255.255.255.0
12 Router(dhcp-config)#default-router 192.168.2.254
13 Router(dhcp-config)#dns-server 192.168.2.254
14 Router(dhcp-config)#end
```

Resultate

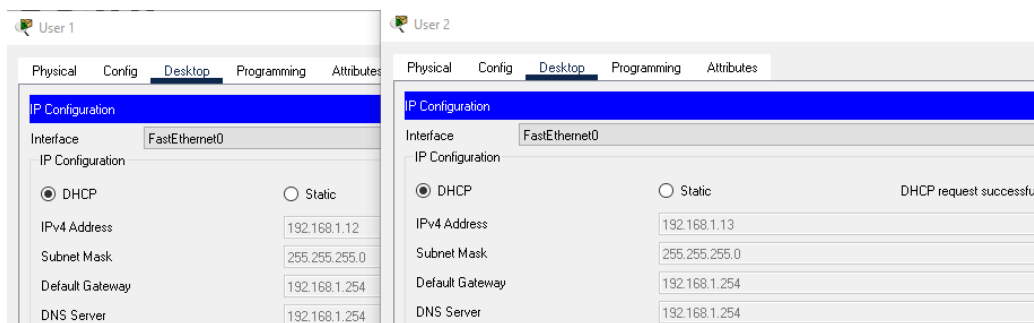
1.2.2 Frage 1

Frage

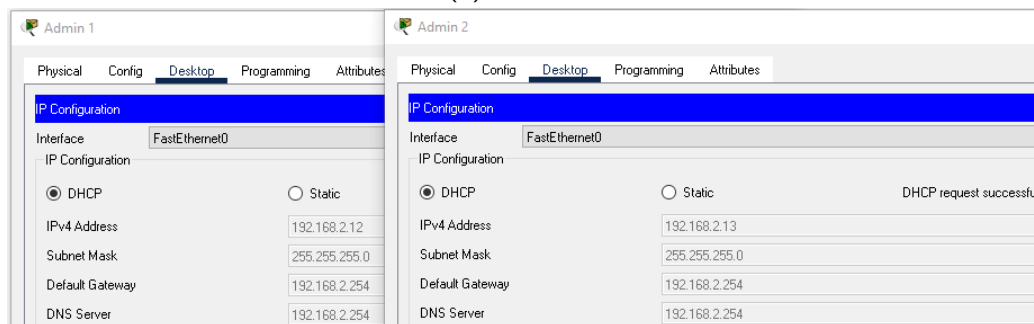
Woher weiß der Router, aus welchem Pool er die Adressen an die anfragenden Clients (also PCs) zuweisen soll?

Antwort

Bei der letzten Übung wurden die "dot1q encapsulation" beim Router aktiviert. Dadurch wurde jedem Subinterface eine bestimmte vlan-id zugewiesen. Jeder Subinterface weiß dadurch welchen Geräten er welche Adresse zuweisen muss.



(a) User PCs



(b) Admin PCs

Abbildung 3: DHCP an den Endsystemen

1.3 Stateless IPv6 Address Autoconfiguration (SLAAC)

1.3.1 IPv6 auto Adresse

IPv6 Configuration

☒ Automatic ☐ Static IPv6 request successful.

IPv6 Address /

Link Local Address

Default Gateway

DNS Server

(a) User 1

IPv6 Configuration

☒ Automatic ☐ Static IPv6 request successful.

IPv6 Address /

Link Local Address

Default Gateway

DNS Server

(b) User 2

IPv6 Configuration

☒ Automatic ☐ Static IPv6 request successful.

IPv6 Address /

Link Local Address

Default Gateway

DNS Server

(c) Admins 1

IPv6 Configuration

☒ Automatic ☐ Static IPv6 request successful.

IPv6 Address /

Link Local Address

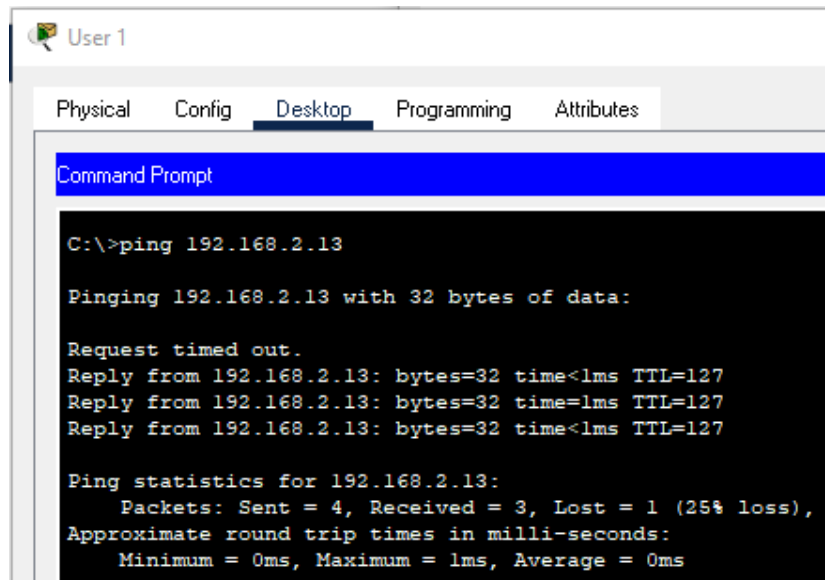
Default Gateway

DNS Server

(d) Admin 2

Abbildung 4: Auto IPv6 an den Endsystemen

1.3.2 Ping Test



The screenshot shows a Command Prompt window titled "User 1" with tabs for Physical, Config, Desktop, Programming, and Attributes. The "Desktop" tab is selected. The command prompt shows the execution of the command `C:\>ping 192.168.2.13`. The output indicates a successful ping with 32 bytes of data, showing four replies from 192.168.2.13 with a time of 1ms and TTL of 127. The ping statistics for 192.168.2.13 show 4 packets sent, 3 received, and 1 lost (25% loss), with approximate round trip times in milliseconds: Minimum = 0ms, Maximum = 1ms, Average = 0ms.

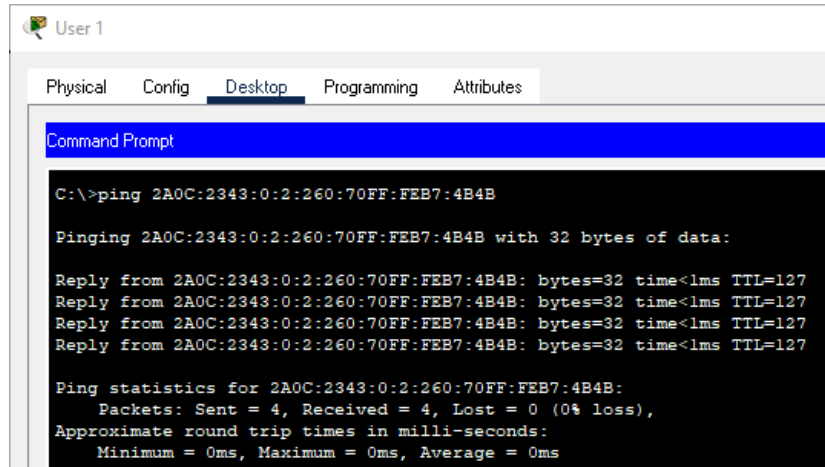
```
C:\>ping 192.168.2.13

Pinging 192.168.2.13 with 32 bytes of data:

Request timed out.
Reply from 192.168.2.13: bytes=32 time<1ms TTL=127
Reply from 192.168.2.13: bytes=32 time=1ms TTL=127
Reply from 192.168.2.13: bytes=32 time<1ms TTL=127

Ping statistics for 192.168.2.13:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

(a) IPv4



The screenshot shows a Command Prompt window titled "User 1" with tabs for Physical, Config, Desktop, Programming, and Attributes. The "Desktop" tab is selected. The command prompt shows the execution of the command `C:\>ping 2A0C:2343:0:2:260:70FF:FEB7:4B4B`. The output indicates a successful ping with 32 bytes of data, showing four replies from 2A0C:2343:0:2:260:70FF:FEB7:4B4B with a time of 1ms and TTL of 127. The ping statistics for 2A0C:2343:0:2:260:70FF:FEB7:4B4B show 4 packets sent, 4 received, and 0 lost (0% loss), with approximate round trip times in milliseconds: Minimum = 0ms, Maximum = 0ms, Average = 0ms.

```
C:\>ping 2A0C:2343:0:2:260:70FF:FEB7:4B4B

Pinging 2A0C:2343:0:2:260:70FF:FEB7:4B4B with 32 bytes of data:

Reply from 2A0C:2343:0:2:260:70FF:FEB7:4B4B: bytes=32 time<1ms TTL=127
Reply from 2A0C:2343:0:2:260:70FF:FEB7:4B4B: bytes=32 time<1ms TTL=127
Reply from 2A0C:2343:0:2:260:70FF:FEB7:4B4B: bytes=32 time<1ms TTL=127
Reply from 2A0C:2343:0:2:260:70FF:FEB7:4B4B: bytes=32 time<1ms TTL=127

Ping statistics for 2A0C:2343:0:2:260:70FF:FEB7:4B4B:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

(b) IPv6

Abbildung 5: Ping von User zu Admin

1.3.3 Frage 2

Frage

Warum funktioniert die Autokonfiguration bei IPv6 ohne Erstellung eines DHCP Services, und worin besteht der Unterschied zu DHCPv6?

Antwort

Weil IPv6 automatisch SLAAC nutzt und die Adresse einfach ohne Vergleiche ausgewählt wird. IPv6 wurde so designt, dass man keinen DHCP Server benötigt. Wenn die Adresse nach der Zuweisung doch vorhanden sein sollte, wird eine neue vergeben. Bei (Stateful) DHCPv6 gibt es einen DHCP Server der alles zuweist und auch gleich die richtige Adresse den Host gibt. Hier können zb. Adressen ausgeschlossen werden, was bei SLAAC nicht geht.

2 Port Mirroring und Sniffing

2.1 Aufbau des virtuellen Netzwerks – Erweiterung

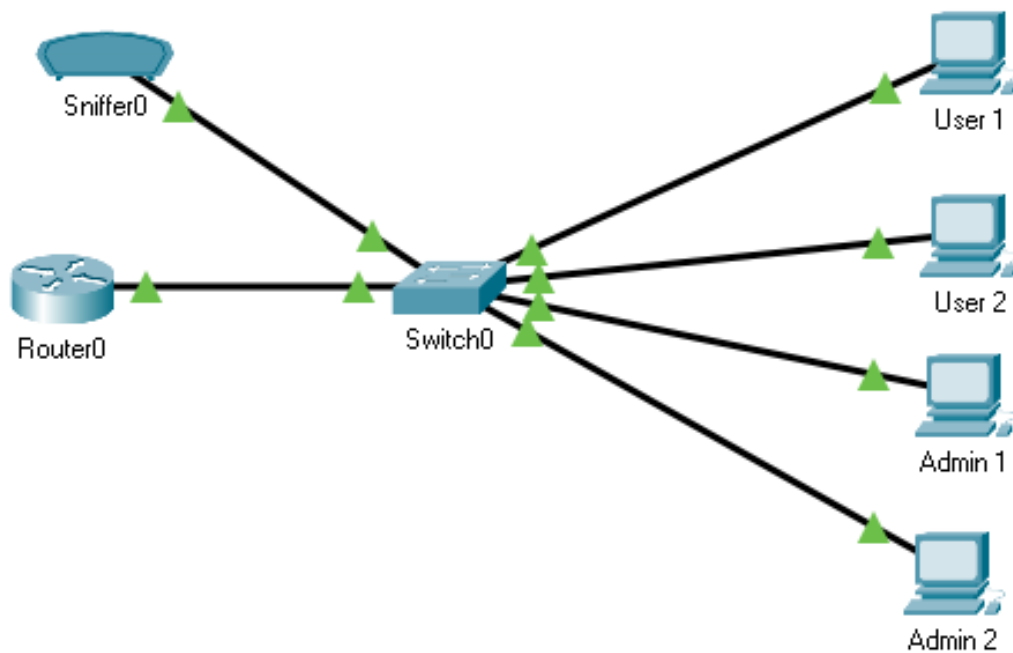


Abbildung 6: Erweitertes Netzwerk

2.2 Konfiguration des Mirror Ports

```
1 Switch>en
2 Switch#conf t
3 Switch(config)#no monitor session 1
4 Switch(config)#monitor session 1 source int gig0/1
5 Switch(config)#monitor session 1 destination int gig0/2
6 Switch(config)#end
```


2.3 Sniffen von DHCP Nachrichten

Ausschalten von DHCP

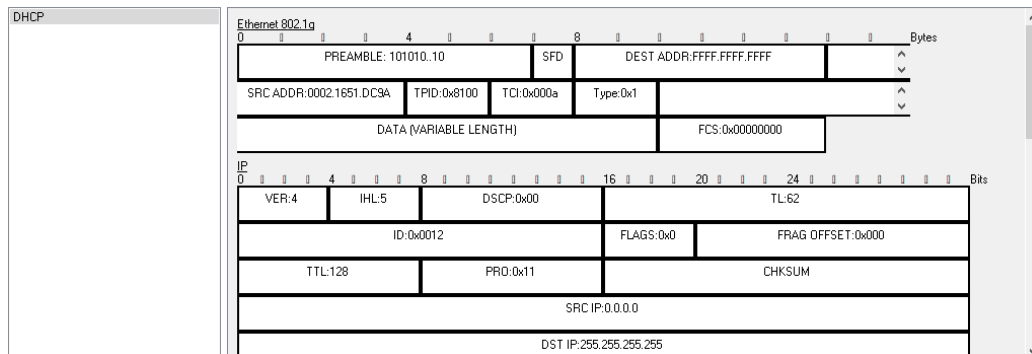


Abbildung 7: Sniffer

Sobald beim Endsystem auf static geschaltet wurde, hat das Endsystem ein Broadcast gemacht um die eigene Adresse zu erhalten, welche aber nicht mehr von DHCP Server kommt. Bei einem DHCP wird immer mit 0.0.0.0 als Src und 255.255.255.255 als Dest nach einer Adresse angefragt.

Einschalten von DHCP

- 1.DHCP Discover: Der Client fragt nach einer Adresse in dem er einen Broadcast mit 0.0.0.0 als Src und 255.255.255.255 als Dest macht.
- 2.DHCP Offer: Als Antwort bekommt er dann eine Adresse vom Server "geoffert".
- 3.DHCP Request: Wenn der Client die Adresse bekommt, schickt er dem Server ein Request um diese Adresse für ihn zu übernehmen/allokieren.
- 4.DHCP ACK: Der Server sendet die Infos, die der Client braucht um konfiguriert zu werden.

2.3.1 Frage 3

Frage

Frage 3: Warum scheinen im Trace des Sniffers alle Nachrichten doppelt auf?

Antwort

Die Nachrichten tauchen nur einmal auf.

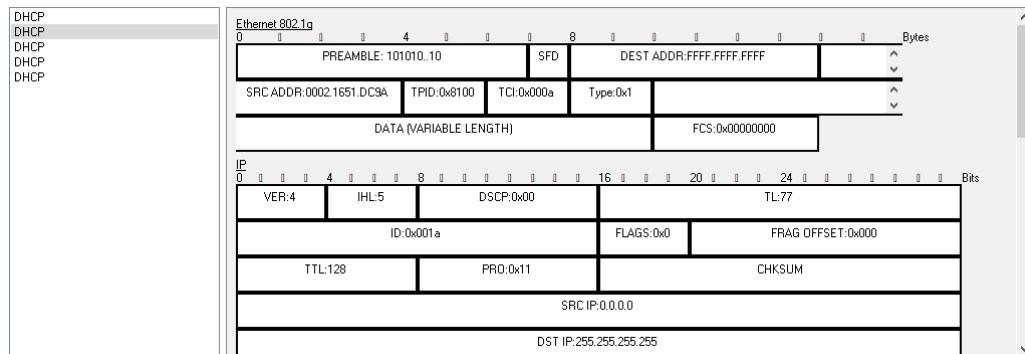


Abbildung 8: Sniffer

3 Wireshark Traces

Die Linux Tests wurden im Windows Subsystem Linux durchgeführt mit Ubuntu als Distro.

3.1 Sniffing von ICMP Paketen

3.1.1 Ping

Shell

```
1 $ ping 8.8.8.8 -c 2
2 PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
3 64 bytes from 8.8.8.8: icmp_seq=1 ttl=116 time=22.6 ms
4 64 bytes from 8.8.8.8: icmp_seq=2 ttl=116 time=19.6 ms
5
6 --- 8.8.8.8 ping statistics ---
7 2 packets transmitted, 2 received, 0% packet loss, time 1002ms
8 rtt min/avg/max/mdev = 19.639/21.124/22.610/1.485 ms
```

Listing 1: Ping Google Public DNS

Wireshark

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.111.43	8.8.8.8	ICMP	98	Echo (ping) request id=0x00ee, seq=1/256, ttl=64 (reply in 2)
2	0.022459	8.8.8.8	192.168.111.43	ICMP	98	Echo (ping) reply id=0x00ee, seq=1/256, ttl=116 (request in 1)
3	1.001753	192.168.111.43	8.8.8.8	ICMP	98	Echo (ping) request id=0x00ee, seq=2/512, ttl=64 (reply in 4)
4	1.021218	8.8.8.8	192.168.111.43	ICMP	98	Echo (ping) reply id=0x00ee, seq=2/512, ttl=116 (request in 3)

Abbildung 9: Erweitertes Netzwerk

Wie erwartet gingen 2 Echo-requests raus und es kamen 2 Echo-replies an.

3.1.2 Ping mit Fragmentation

Shell

```
1 $ ping 8.8.8.8 -s 1500
2 PING 8.8.8.8 (8.8.8.8) 1500(1528) bytes of data.
3 ^C
4 --- 8.8.8.8 ping statistics ---
5 3 packets transmitted, 0 received, 100% packet loss, time 2078ms
```

Listing 2: Linux ping with More Fragment

```

1 $ ping 8.8.8.8 -l 1500
2
3 Pinging 8.8.8.8 with 1500 bytes of data:
4 Request timed out.
5 Request timed out.
6
7 Ping statistics for 8.8.8.8:
8     Packets: Sent = 2, Received = 0, Lost = 2 (100% loss),
9 Control-C

```

Listing 3: Window ping with More Fragment

Das Ergebnis ist wie erwartet, denn die Daten dürfen nur 1472 bytes groß sein, da Ethernet maxmial 1500 Bytes (1518 mit Header und FCS) erlaubt. Der Grund dafür ist, dass der IP Header 20 Bytes und der ICMP Header 8 Bytes ist. $1500 - 28 = 1472$.

Der More Fragment Header war jedoch gesetzt, aber die meisten Provider, Server, Router, etc., blocken ICMP fragmentation aus historischen Gründen und weil fragmentation keine Paketverluste erlaubt, ansonsten kann das Paket nicht zusammengesetzt werden.

Da mein lokales Netzwerk dies jedoch nicht hat, kann ich Pakete mit dem DF flag senden und bekomme eine Antwort zurück.

Wireshark

```

> ping 192.168.1.1 -l 1500

Pinging 192.168.1.1 with 1500 bytes of data:
Reply from 192.168.1.1: bytes=1500 time=1ms TTL=64
Reply from 192.168.1.1: bytes=1500 time<1ms TTL=64
Reply from 192.168.1.1: bytes=1500 time<1ms TTL=64
Reply from 192.168.1.1: bytes=1500 time=1ms TTL=64

Ping statistics for 192.168.1.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

~ took 3s

```

(a) Terminal

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.118	192.168.1.1	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=fb6f) [Reassembled in #2]
2	0.000000	192.168.1.118	192.168.1.1	ICMP	62	Echo (ping) request id=0x0001, seq=192/49152, ttl=128 (reply in 4)
3	0.000075	192.168.1.1	192.168.1.118	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=90b7) [Reassembled in #4]
4	0.000075	192.168.1.1	192.168.1.118	ICMP	62	Echo (ping) reply id=0x0001, seq=192/49152, ttl=64 (request in 2)
5	1.015519	192.168.1.118	192.168.1.1	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=fb70) [Reassembled in #6]
6	1.015519	192.168.1.118	192.168.1.1	ICMP	62	Echo (ping) request id=0x0001, seq=193/49408, ttl=128 (reply in 8)

(b) Wireshark Trace

Abbildung 10: Ergebnisse

3.1.3 Traceroute\Tracert

Shell

```
1 $ traceroute -m 1 8.8.8.8
2 traceroute to 8.8.8.8 (8.8.8.8), 1 hops max, 60 byte packets
3 1 DESKTOP-DJVEGIR (192.168.96.1) 0.211 ms 0.216 ms 0.214 ms
```

Listing 4: Linux traceroute

```
1 $ traceroute -m 1 8.8.8.8
2 traceroute to 8.8.8.8 (8.8.8.8), 1 hops max, 60 byte packets
3 1 192.168.1.1 (192.168.1.1) 0.211 ms 0.216 ms 0.214 ms
```

Listing 5: Darwin traceroute

```
1 $ tracert -h 1 8.8.8.8
2 Tracing route to dns.google [8.8.8.8]
3 over a maximum of 1 hops:
4 1 <1 ms <1 ms <1 ms 192.168.1.1
5 Trace complete.
```

Listing 6: Windows tracert

Wireshark

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.96.1	192.168.111.43	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
2	0.000000	192.168.96.1	192.168.111.43	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
3	0.000001	192.168.96.1	192.168.111.43	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)

(a) Linux

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.1	192.168.1.126	ICMP	94	Time-to-live exceeded (Time to live exceeded in transit)
2	0.014947	192.168.1.1	192.168.1.126	ICMP	94	Time-to-live exceeded (Time to live exceeded in transit)
3	0.016035	192.168.1.1	192.168.1.126	ICMP	94	Time-to-live exceeded (Time to live exceeded in transit)

(b) Darwin

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.118	8.8.8.8	ICMP	106	Echo (ping) request id=0x0001, seq=104/26624, ttl=1 (no response found!)
2	0.000288	192.168.1.1	192.168.1.118	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
3	0.000683	192.168.1.118	8.8.8.8	ICMP	106	Echo (ping) request id=0x0001, seq=105/26880, ttl=1 (no response found!)
4	0.000860	192.168.1.1	192.168.1.118	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
5	0.001164	192.168.1.118	8.8.8.8	ICMP	106	Echo (ping) request id=0x0001, seq=106/27136, ttl=1 (no response found!)
6	0.001343	192.168.1.1	192.168.1.118	ICMP	134	Time-to-live exceeded (Time to live exceeded in transit)
7	0.013857	192.168.1.1	192.168.1.118	ICMP	120	Destination unreachable (Port unreachable)
8	1.522190	192.168.1.1	192.168.1.118	ICMP	120	Destination unreachable (Port unreachable)
9	3.023785	192.168.1.1	192.168.1.118	ICMP	120	Destination unreachable (Port unreachable)

(c) Windows

Abbildung 11: Traceroute auf den verschiedenen Betriebssystemen

Da Darwin und Linux auf Unix basieren, haben beide den gleichen traceroute und wie erwartet entstand ein ttl exceeded. Es werden immer 3 Packets gesendet,

damit man einen Durchschnitt hat.

Windows tracert verhält sich jedoch anders und sendet ein Echo-request bei einem Traceroute raus. Da der echo nie die Destination erreicht, bekommt man zusätzlich zu ttl exceeded auch ein Destination unreachable(Port unreachable)