
Praktikum Rechnernetze

Protokoll zu Versuch 2 (Protokollanalyse mit Wireshark)
von Gruppe 1

Jakob Waibel, Daniel Hiller, Elia Wüstner, Felix Pojtinger

2021-10-19

Inhaltsverzeichnis

1 Einführung	3
1.1 Mitwirken	3
1.2 Lizenz	3
2 Wireshark	4
2.1 Einführung	4
2.2 Ping	6
2.3 DHCP	7
2.4 DNS	9
2.5 ARP	10
2.6 Layer-2-Protokolle	12
2.7 HTTP und TCP	13
2.8 MAC	14
2.9 STP	16
2.10 SNMP	17
2.11 Streaming and Downloads	17
2.12 Telnet und SSH	19
2.13 Wireshark-Filter	20

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 Wireshark

2.1 Einführung

An welchem Koppelement im Systemschrank sollte der Hardware-Analysator/Netzwerk-Sniffer sinnvollerweise angeschlossen werden und warum? Welche grundsätzlichen Möglichkeiten gibt es noch?

- Switch, damit Nachrichten auf Layer 2 auch abgefangen werden können
- Grundsätzlich könnte, vor allem auch in Heimnetzwerken, der Router hierzu verwendet werden, da hier oft Router und Switch zu einem Gerät kombiniert sind.

Starten Sie Wireshark und capturern Sie den aktuellen Traffic. Dokumentieren Sie zunächst, was alles auf Wireshark einprasselt.



Abbildung 3: Screenshot von Wireshark

Zu erkennen sind Pakete von mehreren Protokollen:

- LLDP
- Spanning-Tree-Protokoll (STP)
- DNS
- TCP

- HTTP

Die letzten beiden Protokolle (TCP, HTTP) lassen sich durch das Öffnen des Browsers erklären.

Wie lautet der Filter, mit dem Sie ihre eigene Verbindung ins Labor ausklammern? Welche Möglichkeiten gibt es?

Hierzu gibt es mehrere Optionen:

```
1 !ip.addr == 141.62.66.5
2 not ip.addr == 141.62.66.5
3 !ip.addr eq 141.62.66.5
```



Abbildung 4: Ausklammern der eig. IP, Option 1



Abbildung 5: Ausklammern der eig. IP, Option 2

2.2 Ping

Senden Sie einen Ping zu nachfolgenden Empfängern und zeichnen Sie die entsprechenden Protokolle gezielt mit Wireshark auf. Vergleichen Sie die Protokollabläufe: wer sendet welches Protokoll warum an wen? Pingen Sie an

Einen Rechner Ihrer Wahl im Labornetz:



Abbildung 6: Wireshark-Output zu einem Rechner im Labornetz

Einen beliebigen Server im Internet (Google)

Wir haben hierzu die Namensauflösung aktiviert, damit die IPs zur Domain google.com zugeordnet werden können.



Abbildung 7: Wireshark-Output zu einem Ping nach google.com

Eine beliebige nicht existierende IP-Adresse



Abbildung 8: Wireshark-Output zu einem Ping nach 137.69.12.69

2.3 DHCP

Analysieren Sie die Abläufe bei DHCP (im Labor installiert). Ihre Teilgruppe am Nachbartisch bootet den PC am Arbeitsplatz, protokollieren Sie die DHCP-Abläufe sowie sonstigen Netzverkehr, den der PC bis zum Erhalt der IP-Adresse erzeugt.

Während des Startens werden drei DHCP-Requests für verschiedene Komponenten abgehandelt.



Abbildung 9: Gesamter Bootprozess



Abbildung 10: Bootprozess: DHCP-Requests des BIOS zum Netzwerkboot, damit der Netzwerbootloader über i.e. TFTP geladen werden kann

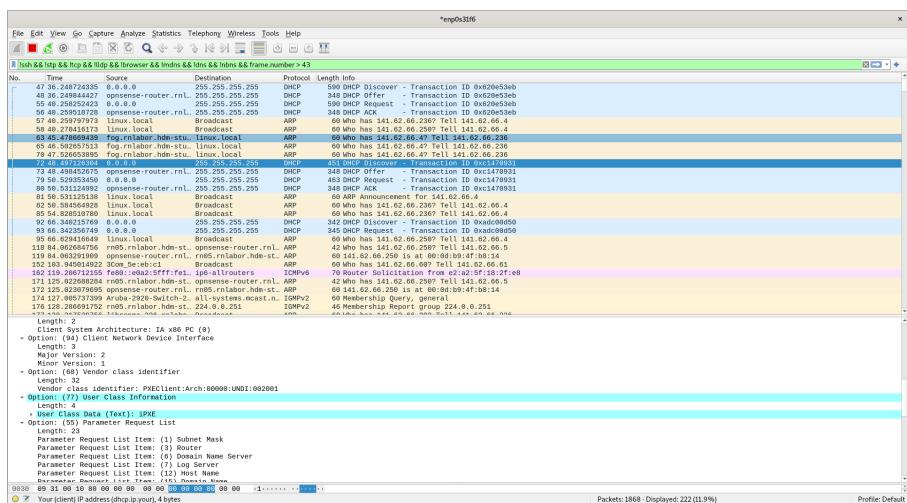


Abbildung 11: Bootprozess: DHCP-Requests des Netzwerbootloaders iPXE

Strukturieren Sie die DHCP-Abläufe und beschreiben Sie, wie DHCP im Detail funktioniert.

Durch Booten des PCs wird dem Rechner mittels DHCP eine IP zugewiesen. Ergänzend kommen noch Standard-Gateway-Adresse und DNS Adresse hinzu. DHCP ermöglicht damit erst, dass verschiedene Rechner in einem Netzwerk kommunizieren können, da dafür jeder Computer eine eigene IP benötigt.

Grundlegend funktioniert DHCP mithilfe von vier Nachrichtentypen. Es gibt den DHCP-Discover, welcher den DHCP-Server in erster Linie benachrichtigen will, dass eine neue IP verlangt wird. Der Server antwortet daraufhin mit einer Offer, welche eine IP reserviert und diese dem Client anbietet. Außerdem

enthält die Offer die IP des DHCP-Servers, die Subnetzmaske und die Lease-Time. Danach kann der Client mit einer DHCP-Request die angebotene IP anfordern. Wenn das in Ordnung ist, antwortet der DHCP-Server mit einem DHCP-Acknowledge.

Vergleicht Sie den Ablauf, wenn Sie den DHCP-Ablauf per ipconfig /release und ipconfig /renew initiiieren

Mittels der folgenden Commands wurde eine IP-Adresse freigegeben und eine neue angefordert.

```
1 # dhclient -r # Release der IP-Adresse
2 # dhclient # Anfrage einer neuen IP-Adresse
```

No.	Time	Source	Destination	Protocol	Length Info
1	19.15.392845861	0.0.0.0	255.255.255.255	DHCP	342 DHCP Discover - Transaction ID 0x70ef81d
2	20.15.39351726	0.0.0.0	255.255.255.255	DHCP	342 DHCP Request - Transaction ID 0x70ef81d
21	15.408801806	linux.local	Broadcast	ARP	68 Who has 141.62.66.250? Tell 141.62.66.4

Dem bereits hochgefahrenen Rechner wird eine neue IP zugeordnet. Wenn wir die IP Zuweisung auf diese weise neu initiieren dann ist der DHCP Ablauf deutlich kürzer, da beim Booten unter der Haube noch deutlich mehr gemacht werden muss (es muss e.g. keine DHCP-Request des BIOS zum Netzwerkboot getätigigt werden).

2.4 DNS

Dokumentieren Sie den Ablauf bei einer DNS-Abfrage

Fall 1: DNS-Server 141.62.66.250:

Mittels folgendem Command wurde eine DNS-Abfrage gemacht:

```
1 $ dig @141.62.66.250 google.com
2 google.com. 163 IN A 142.250.186.174
```

No.	Time	Source	Destination	Protocol	Length Info
11	1.357358000	rn05.rnlabor.hdm-st.	opnsense-router.rnl...	DNS	93 Standard query 0xa276 A google.com OPT
12	1.371692878	opnsense-router.rnl...	rn05.rnlabor.hdm-st.	DNS	97 Standard query response 0xa276 A google.com A 142.250.186.174 OPT

Abbildung 12: Ablauf der Anfrage

Hier nutzten wir den internen DNS Server und machen eine Anfrage auf google.com.

Fall 2: DNS-Server 1.1.1.1 (Cloudflare):

Mittels folgendem Command wurde eine DNS-Abfrage gemacht:

```
1 $ dig @1.1.1.1 +noall +answer google.com
2 google.com. 231 IN A 142.250.185.110
```

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	rn05.rnlabor.hdm-st..	one.one.one.one	DNS	93	Standard query 0x6247 A google.com OPT
2	1.205820789	rn05.rnlabor.hdm-st..	opnsense-router.rnl...	DNS	84	Standard query 0xd2ab PTR 5.66.62.141.in-addr.arpa
5	1.205849397	rn05.rnlabor.hdm-st..	opnsense-router.rnl...	DNS	88	Standard query 0x8083 PTR 1.1.1.1.in-addr.arpa
6	1.207179251	opnsense-router.rnl...	rn05.rnlabor.hdm-st..	DNS	127	Standard query response 0xd2ab PTR 5.66.62.141.in-addr.arpa PTR rn05.rnlabor.hdm-stuttgart.de
7	1.207611338	opnsense-router.rnl...	rn05.rnlabor.hdm-st..	DNS	109	Standard query response 0x8083 PTR 1.1.1.1.in-addr.arpa PTR one.one.one

Abbildung 13: Ablauf der Anfrage

Bei der DNS Anfrage über Cloudflare erscheinen weitere DNS-Requests über DNS Reverse-Zones. Dies wird daran liegen, dass wir über den Router mit dem Internet kommunizieren.

Fall 3: DNS-Server 8.8.8.9 (DNS-Dienst ist dort nicht installiert):

Mittels folgendem Command wurde eine DNS-Abfrage gemacht:

```
1 $ dig @8.8.8.9 +noall +answer google.com
2 ;; connection timed out; no servers could be reached
```

No.	Time	Source	Destination	Protocol	Length	Info
3	0.572493872	rn05.rnlabor.hdm-st..	8.8.8.9	DNS	93	Standard query 0x73f9 A google.com OPT
5	1.088436116	rn05.rnlabor.hdm-st..	opnsense.rnlabor.hd...	DNS	84	Standard query 0xce6b PTR 5.66.62.141.in-addr.arpa
6	1.089061823	rn05.rnlabor.hdm-st..	opnsense.rnlabor.hd...	DNS	88	Standard query 0x73f9 PTR 5.66.62.141.in-addr.arpa
8	1.089026625	opnsense.rnlabor.hd...	rn05.rnlabor.hdm-st..	DNS	127	Standard query response 0xd2ab PTR 5.66.62.141.in-addr.arpa PTR rn05.rnlabor.hdm-stuttgart.de
13	2.087996807	rn05.rnlabor.hdm-st..	opnsense.rnlabor.hd...	DNS	88	Standard query 0xf4fb PTR 250.66.62.141.in-addr.arpa
17	2.089268813	opnsense.rnlabor.hd...	rn05.rnlabor.hdm-st..	DNS	163	Standard query response 0x4fb PTR 250.66.62.141.in-addr.arpa PTR opnsense-router.rnlabor.hdm-stuttgart.de PTR opnsense.rnlabor.hdm...
22	2.089268813	opnsense.rnlabor.hd...	rn05.rnlabor.hdm-st..	DNS	88	Standard query 0x4fb PTR 250.66.62.141.in-addr.arpa
23	3.087954583	rn05.rnlabor.hdm-st..	opnsense.rnlabor.hd...	DNS	84	Standard query 0x73f9 PTR 5.66.62.141.in-addr.arpa
24	3.087959318	rn05.rnlabor.hdm-st..	opnsense.rnlabor.hd...	DNS	88	Standard query 0xf124 PTR 255.255.254.169.in-addr.arpa
25	3.088893145	opnsense.rnlabor.hd...	rn05.rnlabor.hdm-st..	DNS	145	Standard query response 0x59b No such name PTR 19.75.254.169.in-addr.arpa SOA localhost
26	3.089011764	opnsense.rnlabor.hd...	rn05.rnlabor.hdm-st..	DNS	141	Standard query response 0xfc0 No such name PTR 251.0.0.224.in-addr.arpa SOA sns.dns.icann.org
27	3.089125772	opnsense.rnlabor.hd...	rn05.rnlabor.hdm-st..	DNS	147	Standard query response 0xf124 No such name PTR 255.255.254.169.in-addr.arpa SOA localhost

Abbildung 14: Ablauf der Anfrage

Wie im Bild zu sehen ist, bekommen wir den Response **No such name PTR 9.8.8.8.**

Wie erkennen Sie mit Wireshark, dass “versehentlich” ein falscher DNS-Server eingetragen wurde?

Es gibt eine Antwort, welche auf eine nicht gültige IP-Adresse hinweist (Siehe oben).

2.5 ARP

Lösen Sie eine ARP-Anfrage aus und protokollieren Sie die Datenpakete.

Hierzu wurde ein Rechner, welcher zuvor nicht im lokalen ARP-Cache war, neu gestartet.

No.	Time	Source	Destination	Protocol	Length	Info
214	110.515578213	linux-2.local	Broadcast	ARP	42	Who has 141.62.66.67 Tell 141.62.66.5
215	110.5155867298	linux-3.local	linux-2.local	ARP	68	141.62.66.6 is at 4c:52:0e:54:2b
231	115.675164735	linux-3.local	linux-2.local	ARP	68	Who has 141.62.66.5? Tell 141.62.66.6
262	116.673186703	linux-2.local	linux-3.local	ARP	42	141.62.66.5 is at 4c:52:0e:54:8b

Abbildung 15: Ablauf der Anfrage

Wann wird eine ARP-Anfrage gestartet?

Sobald ein Paket an die Zieladresse (in unserem Fall 141.62.66.6) gesendet werden soll, wird eine ARP-Anfrage in Form eines Broadcasts gestartet, um das Zielgerät im Netzwerk zu ermitteln, sofern sich diese nicht bereits im ARP-Cache befindet. Dieser kann mit `ip neigh show` ausgelesen werden. Mit `ip neigh flush all` kann der ARP-Cache geleert werden.

Welcher Rahmentyp wird für die Anfrage verwendet?

Als Rahmentyp wird Ethernet II verwendet.

No.	Time	Source	Destination	Protocol	Length	Info
214	110.515578213	linux-2.local	Broadcast	ARP	42	Who has 141.62.66.6? Tell 141.62.66.5
215	110.5155807208	linux-3.local	Linux-2.local	ARP	68	141.62.66.6 is at 4c:52:62:0e:54:2b
231	115.673164735	linux-3.local	Linux-2.local	ARP	68	Who has 141.62.66.5? Tell 141.62.66.6
232	115.673186793	linux-2.local	Linux-3.local	ARP	42	141.62.66.5 is at 4c:52:62:0e:54:8b

Frame 234: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface enp0s31f6, id 0
Ethernet II, Src: Linux-2.local (4c:52:62:0e:54:8b), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
 » Destination: Broadcast (ff:ff:ff:ff:ff:ff)
 » Source: Linux-2.local (4c:52:62:0e:54:8b)
 Type: ARP (0x0806)
 » Address Resolution Protocol (request)

Abbildung 16: Verwendetes Ethernet-Frame

Beobachten Sie die Veränderung in der ARP-Tabelle Ihres Rechners

Zuvor:

```
1 $ ip neigh show
2 141.62.66.6 dev enp0s31f6 lladdr 4c:52:62:0e:54:2b STALE
3 141.62.66.250 dev enp0s31f6 lladdr 00:0d:b9:4f:b8:14 STALE
4 141.62.66.13 dev enp0s31f6 lladdr 4c:52:62:0e:54:5d STALE
5 141.62.66.236 dev enp0s31f6 lladdr 26:c5:04:8a:fa:eb STALE
```

Danach:

```
1 $ ip neigh show
2 141.62.66.6 dev enp0s31f6 lladdr 4c:52:62:0e:54:2b STALE
3 141.62.66.250 dev enp0s31f6 lladdr 00:0d:b9:4f:b8:14 STALE
4 141.62.66.4 dev enp0s31f6 lladdr 4c:52:62:0e:53:eb STALE
5 141.62.66.13 dev enp0s31f6 lladdr 4c:52:62:0e:54:5d STALE
6 141.62.66.236 dev enp0s31f6 lladdr 26:c5:04:8a:fa:eb STALE
```

2.6 Layer-2-Protokolle

Gelegentlich werden vom Analyzer Broadcasts erkannt. Wer sendet sie, warum und in welchen zeitlichen Abständen?

Die Broadcasts sind ARP-Requests. Sie entstehen dadurch, da Geräte versuchen Daten an andere Geräte zu übertragen, für welche sie keinen Eintrag in ihrem ARP-Cache haben, deshalb muss eine ARP-Anfrage in Form eines Broadcasts gesendet werden, da jeder Host potenziell der gesuchte Host sein kann. Dieser besitzt gesuchte IP X und antwortet daraufhin mit seiner Mac.

No.	Time	Source	Destination	Protocol	Length Info
173 70 .000137330		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
174 71 .999585770		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
175 72 .000137507		linux-3.local	224.0.0.251	MDNS	82 Standard query 0x0000 PTR _ppkey-hkp._tcp.local. "QNAME" question
176 73 .999572543		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
177 74 .999566690		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
178 77 .999563982		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
179 79 .999588860		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
180 81 .999582308		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32758/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
181 82 .000137507		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32758/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
182 84 .000540741		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.20? Tell 141.62.66.226
183 84 .731177879		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.227? Tell 141.62.66.226
184 85 .697465721		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.20? Tell 141.62.66.226
185 85 .697419531		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.227? Tell 141.62.66.226
186 85 .954876527		linux-2.local	opnsense.rnlabor.hd.	DNS	86 Standard query 0x0e2a PTR 226.66.62.141.in-addr.arpa
187 85 .955623690		opnsense.rnlabor.hd.	Linux-2.local	DNS	137 Standard query response @0xe2a PTR 226.66.62.141.in-addr.arpa PTR librenms-226.rnlabo.hdm-stuttgart.de
188 86 .000137507		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32758/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
189 86 .721454740		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.20? Tell 141.62.66.226
190 86 .885478391		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.227? Tell 141.62.66.226
191 87 .999781212		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
192 88 .620784508		linux-3.local	224.0.0.251	MDNS	81 Standard query 0x0000 PTR _nmea-0183._tcp.local. "QNAME" question
193 88 .999588785		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
194 91 .067595494		linux-2.local	opnsense.rnlabor.hd.	ARP	42 Who has 141.62.66.250? Tell 141.62.66.5
195 91 .069717204		opnsense.rnlabor.hd.	Linux-2.local	ARP	60 141.62.66.250 is at 00:0d:b9:4f:b8:14
196 91 .999534042		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
197 93 .00053715353		HewlettP_aa:0b:be	LLDP_Multicast	LLDP	312 MA/04:09:73:a8:80:80 LA/2 120 SysN-213-MP-2920-24G-R142A SysD-HDM-3726A 2920-24G Switch, revision WB.16.10.0015, ROM WB.16.03 ..
198 93 .999571926		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
199 95 .999786412		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002

Abbildung 17: Aufzeichnung der ARP-Requests

Haben Sie noch weitere Protokolle “eingefangen”, die offensichtlich im Labor Rechnernetze keinen Sinn machen?

Aus dem Screenshot lässt sich aus der MDNS-Nachricht der `_nmea-0183._tcp.local` Service-String entnehmen. NMEA 0183 ist ein Standard, welcher für die Kommunikation zwischen Navigationsgeräten auf Schiffen definiert wurde. Da es mitunter für die Kommunikation zwischen GPS-Empfänger und PCs verwendet wird, macht es in unserem Netzwerk wenig Sinn.

No.	Time	Source	Destination	Protocol	Length Info
173 70 .000137330		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
174 71 .999585770		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
175 72 .000137507		linux-3.local	224.0.0.251	MDNS	82 Standard query 0x0000 PTR _ppkey-hkp._tcp.local. "QNAME" question
176 73 .999725453		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
177 75 .999566690		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
178 77 .999563982		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
179 78 .999588860		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
180 79 .999582308		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32758/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
181 80 .000137507		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32758/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
182 84 .000540741		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.20? Tell 141.62.66.226
183 84 .731177879		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.227? Tell 141.62.66.226
184 85 .697465721		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.20? Tell 141.62.66.226
185 85 .697419531		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.227? Tell 141.62.66.226
186 85 .954876527		linux-2.local	opnsense.rnlabor.hd.	DNS	86 Standard query 0x0e2a PTR 226.66.62.141.in-addr.arpa
187 85 .955623690		opnsense.rnlabor.hd.	Linux-2.local	DNS	137 Standard query response @0xe2a PTR 226.66.62.141.in-addr.arpa PTR librenms-226.rnlabo.hdm-stuttgart.de
188 86 .000137507		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
189 86 .721454740		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.20? Tell 141.62.66.226
190 86 .885478391		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.227? Tell 141.62.66.226
191 88 .620784508		linux-3.local	224.0.0.251	MDNS	81 Standard query 0x0000 PTR _nmea-0183._tcp.local. "QNAME" question
192 88 .999588785		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
193 89 .999599785		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
194 91 .067595494		linux-2.local	opnsense.rnlabor.hd.	ARP	42 Who has 141.62.66.250? Tell 141.62.66.5
195 91 .069717204		opnsense.rnlabor.hd.	Linux-2.local	ARP	60 141.62.66.250 is at 00:0d:b9:4f:b8:14
196 91 .999531792		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
197 92 .000540741		librenms-226.rnlabo.	Broadcast	ARP	60 Who has 141.62.66.20? Tell 141.62.66.226
198 92 .999571926		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
199 95 .999786412		HewlettP_aa:0b:be	Spanning-tree-(for..)	STP	119 MST. Root = 32768/0:00:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002

Abbildung 18: Aufzeichnung der ARP-Requests; hier ist das Protokoll zu sehen

Wie sieht es mit UPnP im Labor aus? Auf welchen Maschinen von welchem Hersteller läuft der Dienst? Mit welchem Wireshark-Filter „fischen“ Sie den Traffic heraus?

Es existiert ein Gerät von AVMAudio im Netzwerk, welches über UPnP angesteuert wird. Dies wird immer von demselben Gerät angesteuert, welches über eine Link-Lokale Adresse verfügt, was dafür sorgt, dass es nur innerhalb des Netzwerkes erreicht werden kann. Diese Adressen werden nicht geroutet, sprich die Geräte müssen durch einen Switch etc. verbunden sein. Es kann über den Display-Filter „herausgefischt werden“, indem man nach SSDP filtert.

No.	Time	Source	Destination	Protocol	Length	Info
827	23:55:11.9864589	fe80::5e49:79ff:fe6...ff02::c	SSDP	325	NOTIFY * HTTP/1.1	
828	235.115520628	fe80::5e49:79ff:fe6...ff02::c	SSDP	375	NOTIFY * HTTP/1.1	
829	235.117651013	fe80::5e49:79ff:fe6...ff02::c	SSDP	411	NOTIFY * HTTP/1.1	
830	240.110104287	fe80::5e49:79ff:fe6...ff02::c	SSDP	369	NOTIFY * HTTP/1.1	
841	240.110442125	fe80::5e49:79ff:fe6...ff02::c	SSDP	372	NOTIFY * HTTP/1.1	
842	240.113785421	fe80::5e49:79ff:fe6...ff02::c	SSDP	435	NOTIFY * HTTP/1.1	
843	240.114125309	fe80::5e49:79ff:fe6...ff02::c	SSDP	372	NOTIFY * HTTP/1.1	
844	249.117673673	fe80::5e49:79ff:fe6...ff02::c	SSDP	411	NOTIFY * HTTP/1.1	
845	249.118051257	fe80::5e49:79ff:fe6...ff02::c	SSDP	372	NOTIFY * HTTP/1.1	
846	249.120316833	fe80::5e49:79ff:fe6...ff02::c	SSDP	409	NOTIFY * HTTP/1.1	
847	249.1224718594	fe80::5e49:79ff:fe6...ff02::c	SSDP	399	NOTIFY * HTTP/1.1	
848	249.124712671	fe80::5e49:79ff:fe6...ff02::c	SSDP	443	NOTIFY * HTTP/1.1	
849	249.126997425	fe80::5e49:79ff:fe6...ff02::c	SSDP	427	NOTIFY * HTTP/1.1	
850	249.128981088	fe80::5e49:79ff:fe6...ff02::c	SSDP	425	NOTIFY * HTTP/1.1	
851	241.110212915	fe80::5e49:79ff:fe6...ff02::c	SSDP	408	NOTIFY * HTTP/1.1	
852	241.110541617	fe80::5e49:79ff:fe6...ff02::c	SSDP	364	NOTIFY * HTTP/1.1	
853	241.110892234	fe80::5e49:79ff:fe6...ff02::c	SSDP	373	NOTIFY * HTTP/1.1	
854	241.114269272	fe80::5e49:79ff:fe6...ff02::c	SSDP	436	NOTIFY * HTTP/1.1	
855	241.114269195	fe80::5e49:79ff:fe6...ff02::c	SSDP	373	NOTIFY * HTTP/1.1	
					412	NOTIFY * HTTP/1.1

Frame 1: 2021-10-19 23:55:11.9864589 (2320 bytes on wire (1856 bits), 2320 bytes captured (1856 bits), id 0)
 Ethernet II, Src: AVMAudio_6a:97:78 (5c:49:79:6a:97:78), Dst: IPv4mcast_0c (33:33:00:00:00:00)
 Internet Protocol Version 6, Src: fe80::5e49:79ff:fe6a:9778, Dst: ff02::c
 User Datagram Protocol, Src Port: 1980, Dst Port: 1980
 Simple Service Discovery Protocol

Abbildung 19: Aufzeichnung des SSDP-Protokolls

2.7 HTTP und TCP

Initieren Sie eine HTTP-TCP-Sitzung (beliebige Website) und zeichnen Sie die Protokollabläufe auf

Zuerst wird eine DNS-Request getätig. Daraufhin folgt der 3-Way-Handshake.

TODO: Add valid image

Können Sie den 3-Way-Handshake erkennen? Markieren Sie ihn in der Dokumentation. Welche TCP-Optionen sind beim Handshake aktiviert und welche Bedeutung haben sie?

TODO: Add valid image

TODO: Add valid image

TODO: Add valid image

TODO: Add valid image

Dokumentieren und erläutern Sie die Verwendung der Portnummern bei der Dienstanfrage und der Beantwortung des Dienstes durch den Server.

Unser Computer sendet von Port 49314 an Port 443, welcher für HTTPS genutzt wird. Unser Port ist dabei arbiträr vom System gewählt, der HTTPS Port ist allerdings fest für HTTPS reserviert. Mit einem

Port ist ein Dienst eines Rechners gekennzeichnet. Die Kombination aus Port und IP ergibt einen Socket. Wir senden unsere Nachrichten also an den Socket 209.216.230.240:443.

Klicken Sie auf der Website ein anderes Bild / Link an. Beobachten und dokumentieren Sie: wie verändert sich der TCP-Ablauf?

TODO: Add valid image

2.8 MAC

Wie lauten die MAC-Adressen der im Labor befindlichen Ethernet-Switches? Wie haben Sie die Switches identifizieren können. Welche Möglichkeiten der Identifizierung gibt es?

Beim Spanning-Tree-Protocol lässt sich sehen, dass die Quelle der Nachrichten immer ein HP-Gerät ist. Dieses muss ein fähiges Kopplungselement des Netzwerkes sein, welches das Spanning-Tree-Protocol unterstützt. Daher wird dies mit hoher Wahrscheinlichkeit der Ethernet-Switch sein.

MAC-Adresse: 04:09:73:aa:8b:be

No.	Time	Source	Destination	Protocol	Length Info
178 63.999719934	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
173 65.999382875	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
172 67.999494849	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
174 69.999500770	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
174 71.999585778	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
176 73.999729543	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
177 75.999566699	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
178 77.999639980	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
179 79.999639980	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
180 81.999692309	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
181 83.999531792	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
188 85.999523999	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
191 87.999791219	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
193 89.999699785	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
198 91.999621620	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
198 93.999671926	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
199 95.999796412	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
200 97.999559095	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
201 100.9996216073	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
204 102.999773302	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
206 103.999642753	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
212 109.9996240079	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
213 109.999891439	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
220 113.999728041	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
226 113.999732041	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	
233 115.999656097	HeulettP_aa:8b:be	Spanning-tree-(for-)	STP	119 MST, Root = 32768/0/0:1a:c1:5e:eb:c0 Cost = 228020 Port = 0x8002	

Frame 191: 110 bytes on wire (952 bits), 119 bytes captured (952 bits) on interface enp0s31f6, id 0
 IEEE 802.3 Ethernet
 -> Destination: Spanning-tree-(for-bridges).00 (01:80:c2:00:00:00)
0.0.0. = LG Bit: Globally unique address (factory default)
1.1.1. = IG Bit: Individual address (unicast)
 -> Source: HeulettP_aa:8b:be (04:09:73:aa:8b:be)
 Address: HeulettP_aa:8b:be (04:09:73:aa:8b:be)
0.0.0. = LG Bit: Globally unique address (factory default)
0.0.0. = IG Bit: Individual address (unicast)
 Length: 105
 -> Logical-Link Control
 -> Spanning Tree Protocol

Abbildung 20: Aufzeichnung des STP-Protokolls

Welche MAC-Adresse hat ihr Nachbarrechner?

Durch einen [ping](#) konnten wir die MAC-Adresse des Switches herausfinden.

MAC-Adresse: 4c:52:62:0e:54:2b

No.	Time	Source	Destination	Protocol	Length	Info
216	110.515881773	linux-2.local	linux-3.local	ICMP	98	Echo (ping) request id=0x6c3f, seq=1/256, ttl=64 (reply in 217)
217	110.515881781	linux-3.local	linux-2.local	ICMP	98	Echo (ping) reply id=0x6c3f, seq=1/256, ttl=64 (request in 216)
218	110.515881781	linux-2.local	linux-3.local	ICMP	98	Echo (ping) request id=0x6c3f, seq=2/252, ttl=64 (reply in 220)
219	110.515881783	linux-3.local	linux-2.local	ICMP	98	Echo (ping) reply id=0x6c3f, seq=2/252, ttl=64 (request in 219)
220	110.515881785	linux-2.local	linux-3.local	ICMP	98	Echo (ping) request id=0x6c3f, seq=3/768, ttl=64 (reply in 223)
221	110.515881785	linux-3.local	linux-2.local	ICMP	98	Echo (ping) reply id=0x6c3f, seq=3/768, ttl=64 (request in 223)
Frame 216: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface enp0s31f6, id 0						
Ethernet II, Src: linux-2.local (4c:52:62:0e:54:2b), Dst: linux-3.local (4c:52:62:0e:54:2b)						
Destination: linux-3.local (4c:52:62:0e:54:2b)						
Address: linux-3.local (4c:52:62:0e:54:2b)						
.....0..... = 16 bit: Globally unique address (factory default)						
.....0..... = 16 bit: Individual address (unicast)						
Source: linux-2.local (4c:52:62:0e:54:2b)						
Address: linux-2.local (4c:52:62:0e:54:2b)						
.....0..... = 16 bit: Globally unique address (factory default)						
.....0..... = 16 bit: Individual address (unicast)						
Type: IPv4 (0x0800)						
Internet Protocol Version 4, Src: linux-2.local (141.62.66.5), Dst: linux-3.local (141.62.66.6)						
0100 ..0.. = Version: 4						
0100 ..0.. = Header Length: 20 bytes (5)						
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)						
Total Length: 84						
Identification: 0xe003 (57443)						
Flags: 0x40 (Don't fragment)						
Fragment Offset: 0						
Time to Live: 64						
Protocol: ICMP (1)						
Header Checksum: 0xbbbb [validation disabled]						
[Header checksum status: Unverified]						
Source Address: linux-2.local (141.62.66.5)						
Destination Address: linux-3.local (141.62.66.6)						
Internet Control Message Protocol						

Abbildung 21: MAC-Adresse des Nachbarrechners**Welche MAC-Adresse hat der Labor-Router?**

Durch einen [ping](#) konnten wir die MAC-Adresse des Routers herausfinden.

MAC-Adresse: 00:0d:b9:4f:b8:14

No.	Time	Source	Destination	Protocol	Length	Info
2	0.000000000	rn05.rnlabor.hdm-stuttgart.de (00:0d:b9:4f:b8:14)	opnsense-router.rnlabor.hdm-stuttgart.de (00:0d:b9:4f:b8:14)	ICMP	98	Echo (ping) request id=0x6e92, seq=1/256, ttl=64 (reply in 3)
3	0.327512724	rn05.rnlabor.hdm-st... opnsense-router.rnlabor.hdm-st... ICMP	98 Echo (ping) reply id=0x6e92, seq=4/1024, ttl=64 (request in 2)			
4	0.935109731	opnsense-router.rnlabor.hdm-st... rn05.rnlabor.hdm-st... DNS	84 Standard query 0x7dfa PTR 5.66.62.141.in-addr.arpa			
5	0.935109905	rn05.rnlabor.hdm-st... opnsense-router.rnlabor.hdm-st... DNS	50 Standard query 0x1bb9 PTR 259.66.62.141.in-addr.arpa			
6	0.936033635	opnsense-router.rnlabor.hdm-st... rn05.rnlabor.hdm-st... DNS	127 Standard query response 0x7dfa PTR 5.66.62.141.in-addr.arpa PTR rn05.rnlabor.hdm-stuttgart.de			
7	0.936041401	opnsense-router.rnlabor.hdm-st... rn05.rnlabor.hdm-st... DNS	163 Standard query response 0x1bb9 PTR 259.66.62.141.in-addr.arpa PTR opnsense-router.rnlabor.hdm-stuttgart.de PTR opnsense-router.rnlabor.hdm-st... DNS			
8	0.936041408	opnsense-router.rnlabor.hdm-st... rn05.rnlabor.hdm-st... ICMP	98 Echo (ping) reply id=0x6e92, seq=5/1280, ttl=64 (request in 8)			
9	1.375018075	rn05.rnlabor.hdm-st... opnsense-router.rnlabor.hdm-st... ICMP	98 Echo (ping) request id=0x6e92, seq=6/1536, ttl=64 (reply in 12)			
10	2.375450812	opnsense-router.rnlabor.hdm-st... rn05.rnlabor.hdm-st... ICMP	98 Echo (ping) reply id=0x6e92, seq=6/1536, ttl=64 (request in 11)			
Frame 2: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface enp0s31f6, id 0						
Ethernet II, Src: rn05.rnlabor.hdm-stuttgart.de (00:0d:b9:4f:b8:14), Dst: opnsense-router.rnlabor.hdm-stuttgart.de (00:0d:b9:4f:b8:14)						
Destination: opnsense-router.rnlabor.hdm-stuttgart.de (00:0d:b9:4f:b8:14)						
.....0..... = 16 bit: Globally unique address (factory default)						
.....0..... = 16 bit: Individual address (unicast)						
Source: rn05.rnlabor.hdm-stuttgart.de (00:0d:b9:4f:b8:14)						
Address: rn05.rnlabor.hdm-stuttgart.de (00:0d:b9:4f:b8:14)						
.....0..... = 16 bit: Globally unique address (factory default)						
.....0..... = 16 bit: Individual address (unicast)						
Type: IPv4 (0x0800)						
Internet Protocol Version 4, Src: rn05.rnlabor.hdm-stuttgart.de (141.62.66.5), Dst: opnsense-router.rnlabor.hdm-stuttgart.de (141.62.66.250)						
0100 ..0.. = Version: 4						
0100 ..0.. = Header Length: 20 bytes (5)						
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)						
Total Length: 84						
Identification: 0x6007 (26089)						
Flags: 0x40 (Don't fragment)						
Fragment Offset: 0						
Time to Live: 64						
Protocol: ICMP (1)						
Header Checksum: 0x3266 [validation disabled]						
[Header checksum status: Unverified]						
Source Address: rn05.rnlabor.hdm-stuttgart.de (141.62.66.5)						
Destination Address: opnsense-router.rnlabor.hdm-stuttgart.de (141.62.66.250)						
Internet Control Message Protocol						

Abbildung 22: MAC-Adresse des Labor-Routers**Welche MAC-Adresse hat der Server 141.62.1.5 (außerhalb des Labor-Netzes)?**

Da der Rechner außerhalb des Labor-Netzes ist, kann dessen Mac nicht bestimmt werden.

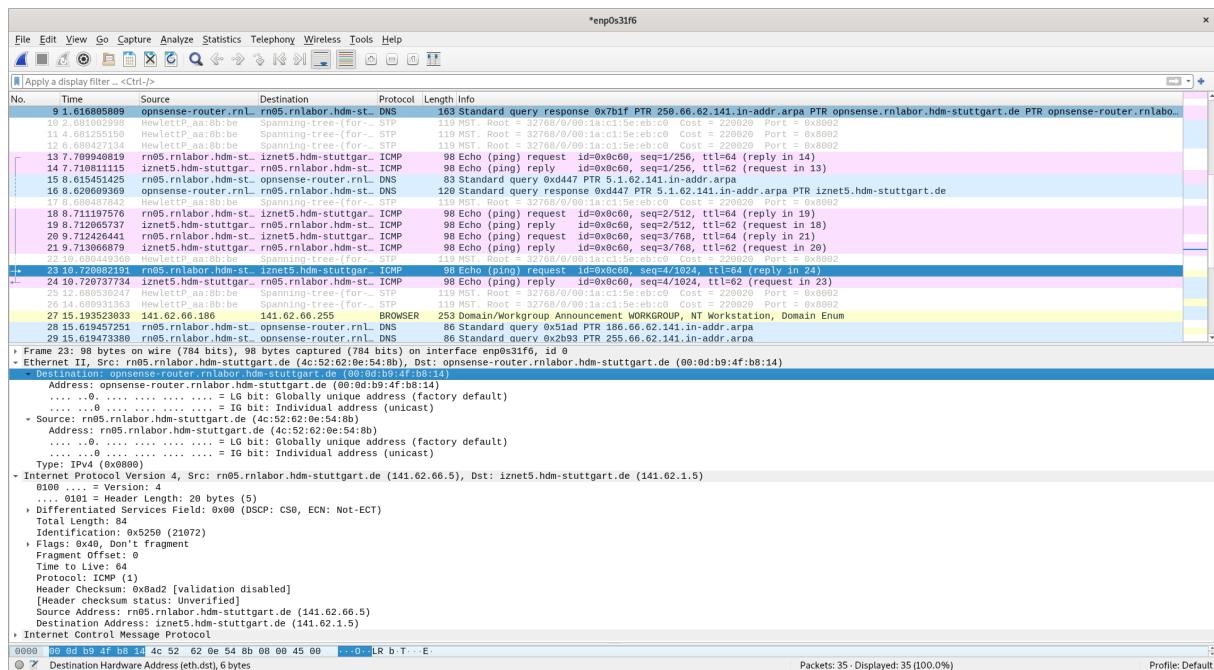


Abbildung 23: MAC-Adresse des externen Rechners

2.9 STP

Filtern Sie auf das Protokoll BPDU/STP. Wer sendet es und welchen Sinn hat dieses Protokoll?

Das STP-Protokoll ist das Spanning Tree Protocol. Das STP-Protokoll verhindert Schleifenbildung; dies ist besonders dann von Nutzen, wenn Redundanzen vorhanden sind. Beim STP-Protokoll werden durch alle am Netz beteiligten Switches eine "Root Bridge" gewählt und redundante Links werden deaktiviert. Wie anhand der OUI der MAC-Adresse erkannt werden kann wird dieses hier von einem HP-Switch verwendet.

No.	Time	Source	Destination	Protocol	Length	Info
393	182.000115690	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
394	184.001059920	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
395	188.000115690	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
397	188.000282308	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
398	190.000183548	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
400	192.000566047	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
407	194.000115690	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
408	196.000399863	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
411	198.000115690	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
412	200.000282749	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
413	202.000197163	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
417	204.000423451	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
418	206.000195952	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
423	208.000093795	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
424	210.000144080	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
425	212.000144080	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
426	214.001080472	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
427	216.000676867	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
429	218.000206922	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
430	220.000144080	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002
431	222.000144080	HewlettP_aa:8b:be	Spanning-tree-(for- STP)	119 MST	Root = 32768/0/00:1a:c1:5ee:b:cb	Cost = 220020 Port = 0x8002

Abbildung 24: Capture mit Filter für STP

2.10 SNMP

Auf welchen Komponenten im Netzwerk wird das Protokoll SNMP ausgeführt?

Es konnte kein SNMP-Traffic im Netzwerk gefunden werden. SNMP, das Simple Network Management Protocol, wird jedoch meist zur Wartung von verbundenen Geräte im Network verwendet, woraus sich schließen lässt, dass es auf Komponenten wie Switches, Routern oder Servern zum Einsatz kommen würde.

2.11 Streaming and Downloads

Starten Sie einen Download einer größeren Datei aus dem Internet und stoppen Sie ihn während der Übertragung. Dokumentieren Sie, wie der Stop-Befehl innerhalb der Protokolle umgesetzt wird

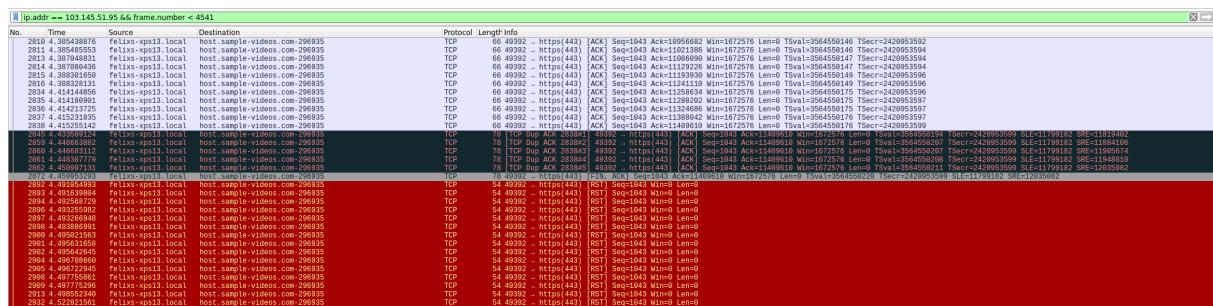


Abbildung 25: Capture beim Canceln des eines Downloads über HTTPS

Da der Download hier via HTTPS durchgeführt wurde, kann erkannt werden, dass die darunterliegende TCP-Verbindung unterbrochen wurde, indem die **RST**-Flag gesetzt wurde. Auch ein TCP-Segment, in welchem hier die **FIN**- und **ACK**-Flags gesetzt wurden, ist dementsprechend zu erkennen.

Protokollieren sie ein Video-Streaming Ihrer Wahl. Welche TCP-Ports werden wozu benutzt? Filtern Sie alle Rahmen, in denen sich das TCP-Window geändert hat

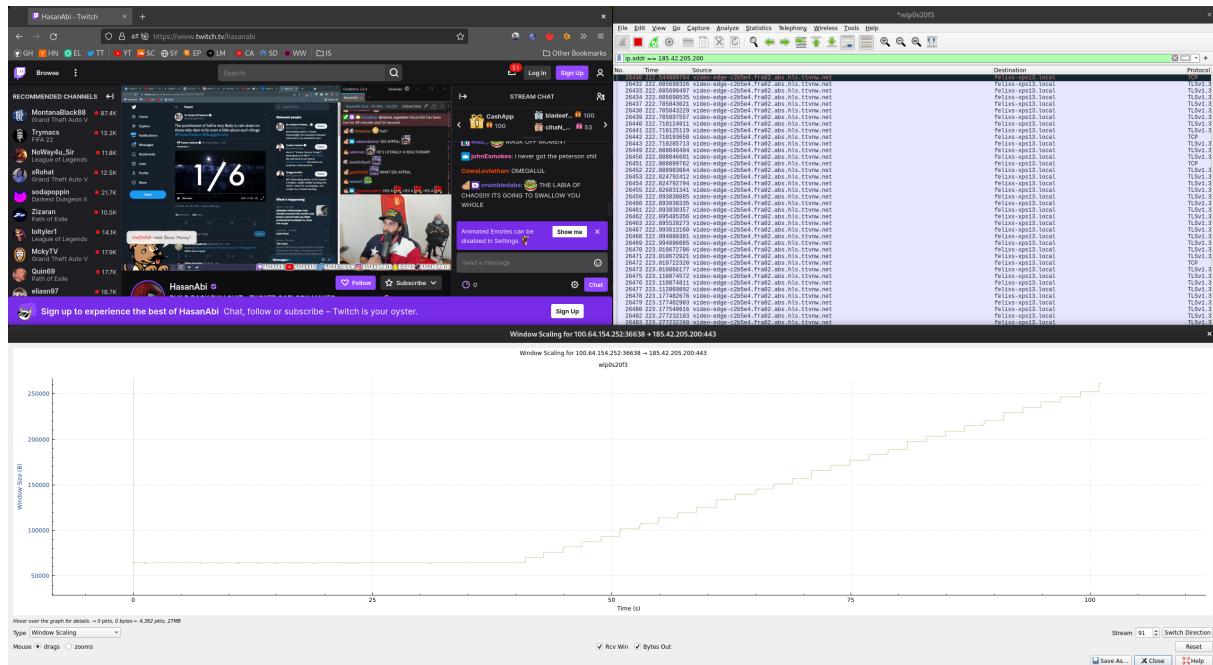


Abbildung 26: Verlauf der TCP-Window-Size beim Streaming von Twitch

Hier wurde ein Stream von Twitch konsumiert; wie zu erkennen ist, wird die Window-Size stetig erhöht. Es wird Port 443, der Standard-Port für HTTPS, verwendet. Seitens des Clients wird vom TCP-Stack des Kernels ein temporärer Port zugewiesen.

2.12 Telnet und SSH

Protokollieren Sie den Ablauf einer TELNET-Verbindung zur IP-Adresse 141.62.66.207 (login: praktikum; passwd: versuch). Können Sie Passwörter im Wireshark-Trace identifizieren? Wie verhält sich im Vergleich dazu eine SSH-Verbindung zum gleichen Server?

Wie zu erkennen ist, wird für eine Telnet-Verbindung eine TCP-Verbindung aufgebaut. Die Passwörter sind zu erkennen.

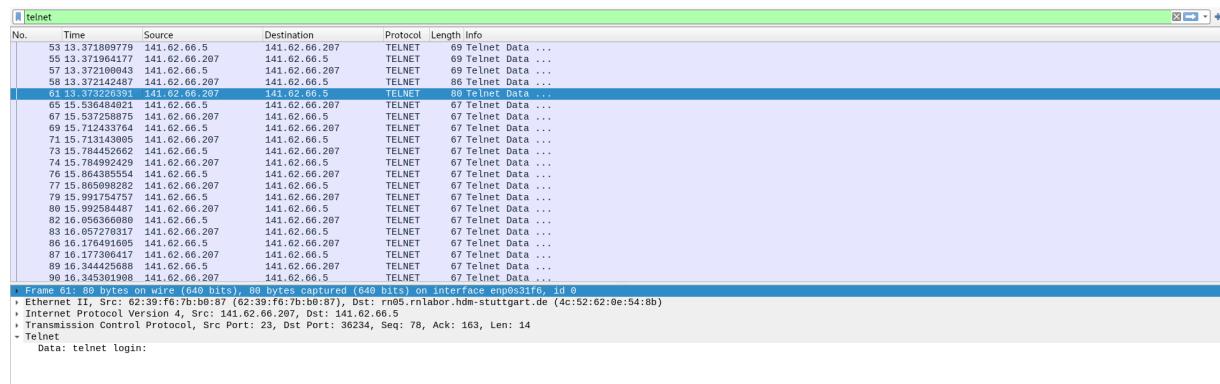


Abbildung 27: Capture des Telnet-Logins

Können Sie Passwörter im Wireshark-Trace identifizieren?

Da Telnet unverschlüsselt ist, können Passwörter identifiziert und ausgelesen werden.

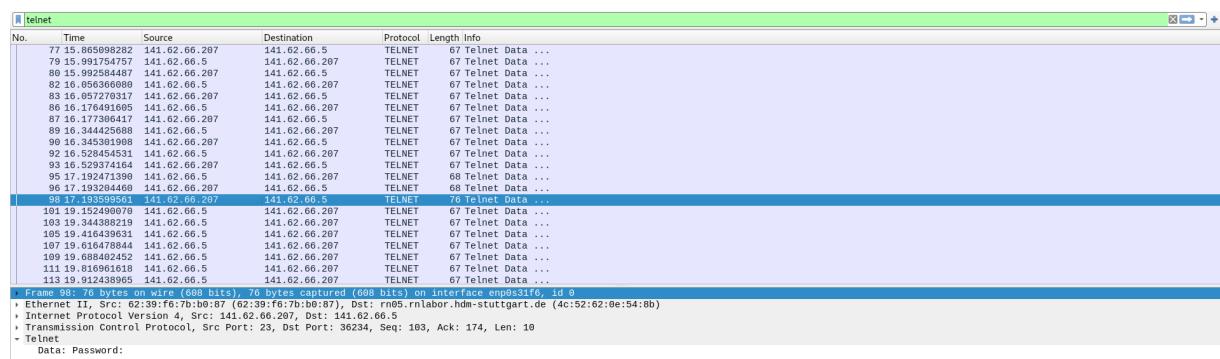


Abbildung 28: Capture des Telnet-Passwords

No.	Time	Source	Destination	Protocol	Length	Info
77	15.865988282	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
78	15.891754757	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
80	15.992584487	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
82	16.056360688	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
83	16.057278313	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
87	16.119205545	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
89	16.344425688	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
90	16.528454531	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
93	16.529374161	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
95	16.530284469	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
97	17.193284469	141.62.66.207	141.62.66.5	TELNET	68	Telnet Data ...
98	17.193599561	141.62.66.207	141.62.66.5	TELNET	76	Telnet Data ...
101	19.152490870	141.62.66.207	141.62.66.207	TELNET	67	Telnet Data ...
103	19.344388219	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
104	19.401247793	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
107	19.410478844	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
109	19.689402452	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
111	19.816961616	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
113	19.912438966	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...

Frame 101: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interface enp0s31f6, id 8
 Ethernet II, Src: rn05.rnlabor.hdm-stuttgart.de (4c:52:62:0e:54:8b), Dst: 62:39:f6:7b:b0:87 (62:39:f6:7b:b0:87)
 Internet Protocol Version 4, Src: 141.62.66.5, Dst: 141.62.66.207
 Transmission Control Protocol, Src Port: 30234, Dst Port: 23, Seq: 174, Ack: 113, Len: 1
 Telnet
 Data: v

Abbildung 29: Capture eines Charakters des Telnet-Passwords**Wie verhält sich im Vergleich dazu eine SSH-Verbindung zum gleichen Server?**

Die SSH-Verbindung ist verschlüsselt; Passwörter, Logins etc. können hier nicht mitgelesen werden.

No.	Time	Source	Destination	Protocol	Length	Info
202	65.784067321	138.68.70.72	141.62.66.5	SSH	126	Server: Encrypted packet (len=60)
204	65.784229966	141.62.66.5	138.68.70.72	SSH	182	Client: Encrypted packet (len=36)
279	119.032310634	138.68.70.72	141.62.66.5	SSH	126	Server: Encrypted packet (len=60)
319	119.032477959	141.62.66.5	138.68.70.72	SSH	182	Client: Encrypted packet (len=36)
459	177.247677878	141.62.66.5	138.68.70.72	SSH	142	Client: Encrypted packet (len=50)
440	174.482509357	138.68.70.72	141.62.66.5	SSH	198	Server: Encrypted packet (len=132)
448	177.2240986626	141.62.66.5	138.68.70.72	SSH	158	Client: Encrypted packet (len=92)
450	177.230961808	138.68.70.72	141.62.66.5	SSH	182	Server: Encrypted packet (len=116)
452	177.237044982	141.62.66.5	138.68.70.72	SSH	126	Client: Encrypted packet (len=60)
454	177.237128457	141.62.66.5	138.68.70.72	SSH	126	Client: Encrypted packet (len=60)
456	177.237144747	141.62.66.5	138.68.70.72	SSH	126	Client: Encrypted packet (len=60)
458	177.243289805	138.68.70.72	141.62.66.5	SSH	206	Server: Encrypted packet (len=140)
460	177.244314401	141.62.66.5	138.68.70.72	SSH	119	Client: Encrypted packet (len=52)
461	177.259592845	138.68.70.72	141.62.66.5	SSH	1514	Server: Encrypted packet (len=1448)
463	177.259674112	138.68.70.72	141.62.66.5	SSH	862	Server: Encrypted packet (len=796)
465	177.259744034	141.62.66.5	138.68.70.72	SSH	118	Client: Encrypted packet (len=52)
466	177.259763894	141.62.66.5	138.68.70.72	SSH	141	Server: Encrypted packet (len=52)
467	177.258776376	141.62.66.5	138.68.70.72	SSH	119	Client: Encrypted packet (len=52)
468	177.264904430	138.68.70.72	141.62.66.5	SSH	134	Server: Encrypted packet (len=68)
469	177.285330770	141.62.66.5	138.68.70.72	SSH	118	Client: Encrypted packet (len=52)
470	177.285533968	141.62.66.5	138.68.70.72	SSH	118	Client: Encrypted packet (len=52)

Frame 400: 126 bytes on wire (1008 bits), 126 bytes captured (1008 bits) on interface enp0s31f6, id 8
 Ethernet II, Src: rn05.rnlabor.hdm-stuttgart.de (4c:52:62:0e:54:8b), Dst: rn05.rnlabor.hdm-stuttgart.de (4c:52:62:0e:54:8b)
 Internet Protocol Version 4, Src: 138.68.70.72, Dst: 141.62.66.5
 Transmission Control Protocol, Src Port: 22, Dst Port: 47480, Seq: 1, Ack: 1, Len: 60
 SSH Protocol
 Packet Length (encrypted) = 900ff09e4
 Encrypted Packet: 6bcb15349d582f55930da2caccb0c73e84abeb992378514580fe2c0b2d9dab4f820ad3e...
 [Direction: server-to-client]

Abbildung 30: Capture eines verschlüsselten SSH-Pakets**2.13 Wireshark-Filter**

Entwickeln, testen und dokumentieren Sie Wireshark-Filter zur Lösung folgender Aufgaben:

Nur IP-Pakete, deren TTL größer ist als ein von Ihnen sinnvoll gewählter Referenzwert

No.	TTL	Time	Source	Destination	Protocol	Length	Info
29	255	1 1.441955699	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
31	255	1 1.519733772	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
40	255	1 1.519733772	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
90	255	1 3.508595900	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
112	255	1 4.554393555	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
113	255	1 4.554393575	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
127	255	1 21.619196641	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
203	255	1 21.619196641	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2044	255	1 25.456197949	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2049	255	1 25.500932266	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2050	255	1 25.500932266	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2051	255	1 25.500932266	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
11826	255	74.573785928	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest.
12018	255	75.597569666	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest.
12561	255	87.681397937	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
13269	255	134.622113475	100.64.154.245	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
18666	255	1 134.622113475	100.64.154.245	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
19846	255	149.929118747	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest.
19852	255	141.955810991	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
20394	255	144.924217109	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
21965	255	150.598595900	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
22035	255	155.472517794	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
22249	255	158.441318164	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
22784	255	167.657466049	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
22852	255	168.579565631	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..

Abbildung 31: Capture der TTL-Werte ab 200

Der Linux-Kernel stellt standardmäßig die TTL auf 64; hier wurde ab 200 gefiltert, damit ausschließlich “ungewöhnliche” Pakete wie z.B. Type: 11 (Time-to-live exceeded)-ICMP-Pakete angezeigt werden.

Nur IP-Pakete, die fragmentiert sind

Mittels eines Filters auf “Must Fragment” wurde hier die TTL eingestellt.

No.	TTL	Time	Source	Destination	Protocol	Length	Info
29	255	1 1.441955699	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
31	255	1 1.519733772	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
40	255	1 1.519733772	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
89	255	1 3.498431116	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
90	255	1 3.508595900	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
112	255	1 4.554393555	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
113	255	1 4.554393575	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
117	255	1 21.619196641	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
1567	255	1 25.456197949	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2031	255	1 25.441398947	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2044	255	1 25.456197949	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2049	255	1 25.500932266	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2050	255	1 25.500932266	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2051	255	1 25.500932266	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
2052	255	1 25.500932266	100.64.154.254	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
11826	255	74.573785928	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest.
12018	255	75.597569666	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
12561	255	78.567487619	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
13269	255	87.681397937	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
18666	255	134.622113475	100.64.154.245	felix-xpos13.local	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
19846	255	149.929118747	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest.
19852	255	141.955810991	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
20394	255	144.924217109	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
21965	255	150.598595900	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
22035	255	155.472517794	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
22249	255	158.441318164	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
22784	255	167.657466049	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..
22852	255	168.579565631	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local "Q" question PTR_companion-link._tcp.local, "Q" quest..

Abbildung 32: Capture von fragmentierten IP-Paketen

Beim Login-Versuch auf <ftp.bellevue.de> mit von Ihnen wählbaren Account-Daten nur Rahmen herausfiltern, die das gewählte Passwort im Ethernet-Datenfeld enthalten

Mittels des Filters `ftp.request.command == "PASS"` werden nur Pakete angezeigt, welche das Passwort enthalten.

No.	Time	Source	Destination	Protocol	Length	Info
3057 651	572178872	212.77.241.212	141.62.66.5	FTP	99	Response: 220 OMNnet FTP Daemon
3784 766	888412363	141.62.66.5	212.77.241.212	FTP	78	Request: USER jakob
3786 766	843474062	141.62.66.5	212.77.241.212	FTP	99	Response: 331 Password required for jakob
3713 715	293446818	141.62.66.5	212.77.241.212	FTP	85	Request: user password=jakob
3714 715	293446818	141.62.66.5	212.77.241.212	FTP	89	Response: 530 Login incorrect.
3716 715	332461233	141.62.66.5	212.77.241.212	FTP	72	Request: SYST
3717 715	331546915	212.77.241.212	141.62.66.5	FTP	85	Response: 215 UNIX Type: L8

```
[Frame 3751: 89 bytes on wire (712 bits), 89 bytes captured (712 bits) on interface em0, link layer type Ethernet II (Ethernet II), Src: rnlabor.hdm-stuttgart.de (4c:9d:62:9e:54:b8), Dst: opnsense.rnlabor.hdm-stuttgart.de (00:0d:b9:4f:b8:14)
Internet Protocol Version 4, Src: 141.62.66.5, Dst: 212.77.241.212
Transmission Control Protocol, Src Port: 51798, Dst Port: 21, Seq: 13, Ack: 57, Len: 23
File Transfer Protocol (FTP)
[Current working directory: ]
```

Abbildung 33: Capture eines FTP-Pakets, welches ein Password enthält

Nur den Port 80-Verkehr zu Ihrer IP-Adresse (ankommend und abgehend)

Mittels eines Filters wurde ausschließlich TCP-Traffic auf Port 80 dargestellt. Mittels `|| udp.port == 80` hätte auch noch UDP-Traffic auf diesem Port dargestellt werden können.

No.	TTL	Time	Source	Destination	Protocol	Length	Info
6508	64	11.367453746	felix-xps13.local	news.ycombinator.com	TCP	74	41206 - http(80) [SYN] Seq=0 Win=64240 Len=0 MSS=1466 SACK_PERM=1 TSval=366326180 TSeq=3732855280
6644	64	11.60931374	felix-xps13.local	news.ycombinator.com	TCP	66	41206 - http(80) [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=366326422 TSeq=3732855280
6645	64	11.609418667	felix-xps13.local	news.ycombinator.com	HTTP	150	GET / HTTP/1.1
6774	64	11.814666182	felix-xps13.local	news.ycombinator.com	TCP	66	41206 - http(80) [ACK] Seq=5 Ack=376 Win=64000 Len=0 TSval=366326627 TSeq=3732855522
6775	64	11.814783601	felix-xps13.local	news.ycombinator.com	TCP	66	41206 - http(80) [FIN, ACK] Seq=85 Ack=376 Win=64126 Len=0 TSval=366326628 TSeq=3732855522
6888	64	12.019299384	felix-xps13.local	news.ycombinator.com	TCP	66	41206 - http(80) [ACK] Seq=377 Win=64128 Len=0 TSval=366326822 TSeq=3732855728

Abbildung 34: Capture aller TCP-Segmente auf Port 80

Nur Pakete mit einer IP-Multicast-Adresse

Mittels eines Filters werden nur IPs > 224.0.0.0 dargestellt, was IP-Multicast-Adressen sind.

No.	TTL	Time	Source	Destination	Protocol	Length	Info
6508	64	11.367453746	felix-xps13.local	news.ycombinator.com	TCP	74	41206 - http(80) [SYN] Seq=0 Win=64240 Len=0 MSS=1466 SACK_PERM=1 TSval=366326180 TSeq=3732855280
6644	64	11.60931374	felix-xps13.local	news.ycombinator.com	TCP	66	41206 - http(80) [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=366326422 TSeq=3732855280
6645	64	11.609418667	felix-xps13.local	news.ycombinator.com	HTTP	150	GET / HTTP/1.1
6774	64	11.814666182	felix-xps13.local	news.ycombinator.com	TCP	66	41206 - http(80) [ACK] Seq=5 Ack=376 Win=64000 Len=0 TSval=366326627 TSeq=3732855522
6775	64	11.814783601	felix-xps13.local	news.ycombinator.com	TCP	66	41206 - http(80) [FIN, ACK] Seq=85 Ack=376 Win=64126 Len=0 TSval=366326628 TSeq=3732855522
6888	64	12.019299384	felix-xps13.local	news.ycombinator.com	TCP	66	41206 - http(80) [ACK] Seq=377 Win=64128 Len=0 TSval=366326822 TSeq=3732855728

Abbildung 35: Capture aller IP-Pakete mit Multicast-Adressen