

Praktikum Rechnernetze

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

Jakob Waibel Daniel Hiller Elia Wüstner Felix Pojtinger

2021-10-26

Einführung

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

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

Wireshark

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.

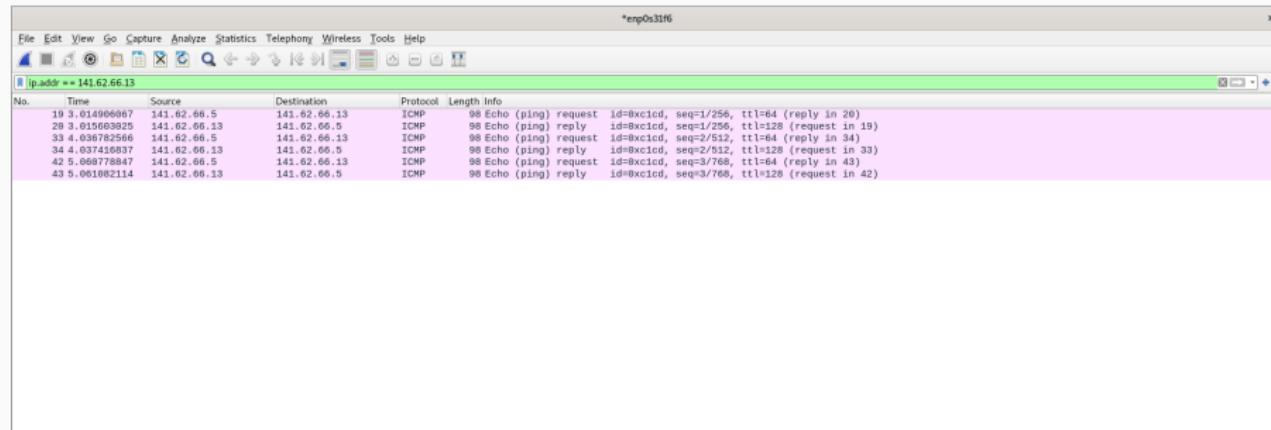
Starten Sie Wireshark und capturern Sie den aktuellen Traffic. Dokumentieren Sie zunächst, was alles auf Wireshark einprasselt.



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:



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.

No.	Time	Source	Destination	Protocol	Length	Info
47	36.248724335	0.0.0.0	255.255.255.255	DHCP	59	DHCP Discover - Transaction ID 0x620e53eb
48	36.248844227	opnsense-router.rml...	255.255.255.255	DHCP	343	DHCP Offer - Transaction ID 0x620e53eb
55	49.259252500	0.0.0.0	255.255.255.255	DHCP	598	DHCP Request - Transaction ID 0x620e53eb
56	49.259252529	opnsense-router.rml...	255.255.255.255	DHCP	348	DHCP ACK - Transaction ID 0x620e53eb
57	49.259797973	linux.local	Broadcast	ARP	60	Who has 141.62.66.236? Tell 141.62.66.4
58	49.278416173	linux.local	Broadcast	ARP	60	Who has 141.62.66.250? Tell 141.62.66.4
63	49.476659439	fog.rnlabor.hds-stu...	linux.local	ARP	60	Who has 141.62.66.4? Tell 141.62.66.236
65	49.592657513	fog.rnlabor.hds-stu...	linux.local	ARP	60	Who has 141.62.66.4? Tell 141.62.66.236
79	49.526653895	fog.rnlabor.hds-stu...	linux.local	ARP	60	Who has 141.62.66.4? Tell 141.62.66.236
72	49.497126304	0.0.0.0	255.255.255.255	DHCP	451	DHCP Discover - Transaction ID 0xc1470931
73	49.497126305	opnsense-router.rml...	255.255.255.255	DHCP	343	DHCP Offer - Transaction ID 0xc1470931
79	49.526653895	0.0.0.0	255.255.255.255	DHCP	343	DHCP Request - Transaction ID 0xc1470931
88	50.531124982	opnsense-router.rml...	255.255.255.255	DHCP	348	DHCP ACK - Transaction ID 0xc1470931
81	50.531125138	linux.local	Broadcast	ARP	60	ARP Announcement for 141.62.66.4
82	50.584564498	linux.local	Broadcast	ARP	60	Who has 141.62.66.236? Tell 141.62.66.4
85	54.826519700	linux.local	Broadcast	ARP	60	Who has 141.62.66.236? Tell 141.62.66.4
92	66.340215769	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0xadc90d58
93	66.342356749	0.0.0.0	255.255.255.255	DHCP	345	DHCP Request - Transaction ID 0xadc90d58
95	66.629416649	linux.local	Broadcast	ARP	60	Who has 141.62.66.250? Tell 141.62.66.4

Abbildung 9: Gesamter Bootprozess

Dokumentieren Sie den Ablauf bei einer DNS-Abfrage

Fall 1: DNS-Server 141.62.66.250:

Mittels folgendem Command wurde eine DNS-Abfrage gemacht:

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

No.	Time	Source	Destination	Protocol	Length	Info
11	1.357358800	rn05.rn1abor.hdm-st... opnsense-router.rnL	DNS	93	Standard query 0xa276	A google.com OPT
12	1.371050270	opnsense-router.rnL	rn05.rn1abor.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:

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.

Ats	Time	Source	Dest	Type	Protocol	Length	Info
		214.110.5155670211	linux-2.local	broadcast	ARP	46	who has 141.62.66.6? Tell 141.62.66.5
		215.110.5155667208	linux-3.local	linux-2.local	ARP	60	141.62.66.6 is at 4c:52:0e:54:2b
		231.115.673164798	linux-3.local	linux-2.local	ARP	60	who has 141.62.66.5? Tell 141.62.66.6
		232.115.673180788	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

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
178	1993-07-23T06:48:00.000Z	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
178	1993-07-23T06:48:00.000Z	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
175 72 .8867515887	Linux-3.local	224.0.0.251	MHDN	82 Standard query @0x8000 PTR _opkey-hkp._tcp.local. "Qn" question	
178	1993-07-23T06:48:00.000Z	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
178	1993-07-23T06:48:00.000Z	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
178	1993-07-23T06:48:00.000Z	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
179	1993-07-23T06:48:00.000Z	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
181	1993-07-23T06:48:00.000Z	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
182	84.699546741	Librennes-226.rnlabor.de	Broadcast	ARP	60 Who has 141.62.66.29? Tell 141.62.66.220
183	84.731177897	Librennes-226.rnlabor.de	Broadcast	ARP	60 Who has 141.62.66.227? Tell 141.62.66.226
184	85.674657721	Librennes-226.rnlabor.de	Broadcast	ARP	60 Who has 141.62.66.29? Tell 141.62.66.226
185	85.674657721	Librennes-226.rnlabor.de	Broadcast	ARP	60 Who has 141.62.66.29? Tell 141.62.66.226
186	85.954876527	Linux-2.local	opensec_rnlabor.hds	DNS	86 Standard query @0x9e2a PTR 226.66.62.141.in-addr.arpa
187	85.956236998	opensec_rnlabor.hds	Linux-2.local	DNS	137 Standard query response 0x9e2a PTR 226.66.62.141.in-addr.arpa PTR Librennes-226.rnlabor.hdm-stuttgart.de
188	86.721457449	Librennes-226.rnlabor.de	Broadcast	ARP	60 Who has 141.62.66.29? Tell 141.62.66.226
189	86.758478381	Librennes-226.rnlabor.de	Broadcast	ARP	60 Who has 141.62.66.227? Tell 141.62.66.226
191	87.899791212	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/9/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
192	88.6269704988	Linux-3.local	224.0.0.251	MHDN	81 Standard query @0x8000 PTR _newa_0183._tcp.local. "Qn" question
193	89.899791212	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/9/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
194	90.8676590494	Linux-2.local	opensec_rnlabor.hds	DNS	42 Who has 141.62.66.29? Tell 341.62.66.5
195	91.8997172080	opensec_rnlabor.hds	Linux-2.local	ARP	60 141.62.66.29b is at 00:00:09:4fb:14
197	93.886571335	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
198	95.899796112	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
199	95.899796112	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
200	95.899796112	NewlettPc_as-80-be	Spanning-tree-(For...)	STP	119 MST Root = 32768/8/0/0.1a:c1:Se0/0:0 Cost = 228020 Port = 8x8002
Frames 302: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface em0fa0, id 9					

HTTP und TCP

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

Zuerst wird ein DNS-Request getätigt. Daraufhin folgt der 3-Way-Handshake. Dieser ist an der charakteristischen Abfolge SYN, SYN-ACK, ACK zu erkennen.

No.	Time	Source	Destination	Protocol	Length	Info
714	7.5,98625	100.64.84.66	141.70.124.5	DNS	88	Standard query 0x189d A news.ycombinator.com
715	7.5,98881	100.64.84.66	141.70.124.5	DNS	88	Standard query 0x58df AAAA news.ycombinator.com
716	7.6,08834	141.70.124.5	100.64.84.66	DNS	158	Standard query response 0x58df AAAA news.ycombinator.com SOA ns-225.awsdns-28.com
717	7.6,13971	141.70.124.5	100.64.84.66	DNS	233	Standard query response 0x189d A news.ycombinator.com A 209.216.230.240 NS ns-1411.awsdns-48.org NS ns-1914.awsdns-47.co.l
718	7.6,14386	100.64.84.66	209.216.230.248	TCP	78	49314 -> 443 [SYN, ECR, CWR] Seq=0 Win=65535 Len=0 MSS=1468 Win=65535 TSecr=> SACK_PERM=1
719	7.6,75218	209.216.230.248	100.64.84.66	TCP	74	443 -> 49314 [SYN, ACK, ECR] Seq=1 Win=65535 Len=0 MSS=1468 Win=65535 TSecr=> SACK_PERM=1
720	7.6,75334	209.216.230.248	100.64.84.66	TCP	66	49314 -> 443 [ACK] Seq=1 Win=131712 Len=0 TSecr=>2045828466
721	7.7,65826	100.64.84.66	209.216.230.248	TLSv1_	583	Client Hello
722	7.7,91740	209.216.230.248	100.64.84.66	TLSv1_	1514	New Session
723	7.7,91748	209.216.230.248	100.64.84.66	TLSv1_	1514	443 -> 49314 [ACK] Seq=448 Ack=518 Win=65664 Len=1448 TSecr=>2045828632 TSecr=>2512581211 [TCP segment of a reassembled PDU]
724	7.7,91749	209.216.230.248	100.64.84.66	TLSv1_	1063	Certificate, Certificate Status, Server Key Exchange, Server Hello Done
725	7.7,91751	100.64.84.66	209.216.230.248	TCP	66	49314 -> 443 [ACK] Seq=189 Win=13083 Win=127972 Len=0 TSecr=>2045828612
726	7.9,17726	100.64.84.66	209.216.230.248	TCP	66	[TCP Window Update] 49314 -> 443 [ACK] Seq=1812 Ack=3893 Win=131972 Len=0 TSecr=>2512581363 TSecr=>2045828612
727	7.9,37248	100.64.84.66	209.216.230.248	TLSv1_	192	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
728	7.9,37649	100.64.84.66	209.216.230.248	TLSv1_	786	Application Data
729	7.9,88785	209.216.230.248	100.64.84.66	TCP	66	443 -> 49314 [ACK] Seq=8931 Ack=1364 Win=64832 Len=0 TSecr=>2045828783 TSecr=>2512581383
730	7.9,89369	209.216.230.248	100.64.84.66	TLSv1_	324	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
731	7.9,89397	100.64.84.66	209.216.230.248	TCP	66	49314 -> 443 [ACK] Seq=1364 Ack=1364 Win=130752 Len=0 TSecr=>2045828788
732	7.9,89625	209.216.230.248	100.64.84.66	TCP	1514	443 -> 49314 [ACK] Seq=4151 Ack=1513 Win=65664 Len=1448 TSecr=>2512581383 [TCP segment of a reassembled PDU]
733	7.9,89626	209.216.230.248	100.64.84.66	TCP	1514	443 -> 49314 [ACK] Seq=5999 Ack=1364 Win=65664 Len=1448 TSecr=>2045828789 TSecr=>2512581383 [TCP segment of a reassembled PDU]
734	7.9,89626	209.216.230.248	100.64.84.66	TCP	1514	443 -> 49314 [ACK] Seq=8474 Ack=1364 Win=65664 Len=1448 TSecr=>2045828789 TSecr=>2512581383 [TCP segment of a reassembled PDU]
735	7.9,89627	209.216.230.248	100.64.84.66	TCP	1514	443 -> 49314 [ACK] Seq=8495 Ack=1364 Win=65664 Len=1448 TSecr=>2045828789 TSecr=>2512581383 [TCP segment of a reassembled PDU]
736	7.9,89628	209.216.230.248	100.64.84.66	TLSv1_	681	Application Data
737	7.9,89631	100.64.84.66	209.216.230.248	TCP	66	49314 -> 443 [ACK] Seq=1364 Ack=18588 Win=124688 Len=0 TSecr=>2512581542 TSecr=>2045828789
738	7.9,89648	100.64.84.66	209.216.230.248	TCP	66	[TCP Window Update] 49314 -> 443 [ACK] Seq=1364 Ack=18588 Win=131872 Len=0 TSecr=>2512581542 TSecr=>2045828789
739	7.9,22322	100.64.84.66	209.216.230.248	TLSv1_	691	Change Cipher Spec, Encrypted Handshake Message
740	7.9,37698	100.64.84.66	209.216.230.248	TCP	78	49314 -> 443 [SYN, ECR, CWR] Seq=0 Win=65535 Len=0 MSS=1468 Win=65535 TSecr=>2045828789 TSecr=>2512581669 [TCP segment of a reassembled PDU]
741	7.9,37485	209.216.230.248	100.64.84.66	TCP	1514	443 -> 49314 [ACK] Seq=18558 Ack=1989 Win=65664 Len=1448 TSecr=>2045829076 TSecr=>2512581669 [TCP segment of a reassembled PDU]
742	7.9,37487	209.216.230.248	100.64.84.66	TLSv1_	823	Application Data
743	7.9,37463	100.64.84.66	209.216.230.248	TCP	66	49314 -> 443 [ACK] Seq=1989 Ack=12763 Win=128832 Len=0 TSecr=>2512581828 TSecr=>2045829076
744	7.9,37691	100.64.84.66	209.216.230.248	TLSv1_	674	Application Data
750	7.9,41934	209.216.230.248	100.64.84.66	TCP	74	443 -> 49314 [SYN, ACK, ECR] Seq=1 Win=65535 Len=0 MSS=1468 Win=65535 TSecr=>1535768379 TSecr=>3827897587
751	7.9,41956	100.64.84.66	209.216.230.248	TCP	66	49314 -> 443 [ACK] Seq=1 Win=131712 Len=0 TSecr=>382789754 TSecr=>1535768379
752	7.9,42437	100.64.84.66	209.216.230.248	TLSv1_	585	Client Hello
759	7.9,52767	209.216.230.248	100.64.84.66	TCP	1514	443 -> 49314 [ACK] Seq=12763 Ack=597 Win=65664 Len=1448 TSecr=>2045829221 TSecr=>2512581821 [TCP segment of a reassembled PDU]
766	7.9,52768	209.216.230.248	100.64.84.66	TLSv1_	793	Application Data
761	7.9,52715	100.64.84.66	209.216.230.248	TCP	66	49314 -> 443 [ACK] Seq=1989 Ack=12763 Win=128832 Len=0 TSecr=>2512581972 TSecr=>2045829221
762	7.9,59143	209.216.230.248	100.64.84.66	TLSv1_	222	Server Hello, Change Cipher Spec, Encrypted Handshake Message
763	7.9,59147	100.64.84.66	209.216.230.248	TCP	66	49315 -> 443 [ACK] Seq=157 Win=131586 Len=0 TSecr=>3827897926 TSecr=>1535768058
764	7.9,59169	100.64.84.66	209.216.230.248	TLSv1_	117	Change Cipher Spec, Encrypted Handshake Message

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
170	63. 999710934	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
171	65. 999832879	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
172	67. 999832879	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
173	70. 999817336	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
174	71. 999817336	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
178	72. 999729543	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
177	73. 999729543	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
178	74. 999729543	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
179	75. 999806699	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
178	76. 999806699	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
179	77. 999806699	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
180	78. 999806699	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
181	80. 999802388	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
182	83. 999531792	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
183	84. 999531792	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
184	85. 999531792	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
185	86. 999807785	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
186	89. 9998034042	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
188	91. 9998071526	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
189	93. 9998071526	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
190	95. 9998071526	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
206	97. 9995306051	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
201	100. 980216873	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
203	103. 980216873	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
204	103. 999772305	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
205	104. 999772305	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
212	108. 9802409879	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
213	108. 999891429	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
191	87. 9997191212	HewlettPc_aa:bb:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002

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
393 102.000115690	HeuvelTP.an.Bb:be	Spanning-tree-(For-	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
394 104.000105982	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
395 106.000056817	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
397 109.000202936	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
398 110.000202937	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
406 192.000560847	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
407 194.000877110	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
408 196.000399860	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
411 200.000399861	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
412 208.000287489	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
413 292.000187163	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
417 204.000254351	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
418 206.000015959	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
420 210.000015960	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
424 218.000028987	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
425 212.000027773	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
426 214.000080847	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
427 216.000078690	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
428 218.000078691	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
430 720.000140895	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002
433 222.000177264	HeuvelTP.an.Bb:be	Spanning-tree-(For- STP	119 MST	Root = 32768/0/0/0:1a:cl:Se:eb:c9	Cost = 226029	Port = 0x80002

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.

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 28: 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 mit der Ziel-IP 192.168.1.100 und der FIN- und ACK-Flag gesetzt.

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.

No.	Time	Source	Destination	Protocol	Length	Info
53	13.371899779	141.62.66.5	141.62.66.207	TELNET	69	Telnet Data ...
55	13.371964177	141.62.66.207	141.62.66.5	TELNET	69	Telnet Data ...
57	13.372108043	141.62.66.5	141.62.66.207	TELNET	69	Telnet Data ...
59	13.372108043	141.62.66.5	141.62.66.207	TELNET	69	Telnet Data ...
61	13.372108043	141.62.66.5	141.62.66.207	TELNET	69	Telnet Data ...
65	15.536484921	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
67	15.537258875	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
69	15.537258875	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
71	15.537258875	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
73	15.784452662	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
74	15.784992429	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
76	15.864385854	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
77	15.865698282	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
79	15.992584487	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	16.056366088	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
83	16.057270317	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
86	16.176481343	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
87	16.176481343	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
89	16.444256688	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
90	16.453501998	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...

Frame 61: 88 bytes on wire (648 bits), 88 bytes captured (648 bits) on interface enp3s0f0, id = 0
Ethernet II, Src: rnlabor (62:39:f6:7b:b8:87) [ethernet], Dst: rn6.rnlabor.hdm-stuttgart.de (4c:52:82:0e:54:8b)
Internet Protocol Version 4, Src: 141.62.66.207, Dst: 141.62.66.5
Transmission Control Protocol, Src Port: 23, Dst Port: 30234, Seq: 78, Ack: 163, Len: 34
Telnet
Data: telnet login:

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
25	255	1.444955667	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
26	255	1.444955673	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
31	255	1.451978337	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
89	255	1.498643116	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
98	255	1.3.500598080	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
112	255	1.4.354393556	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
120	255	1.4.354393557	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
1527	255	1.21.51668853	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
1567	255	1.21.654196641	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2831	255	1.25.443188947	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2844	255	1.25.456619749	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2850	255	1.25.456619750	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2849	255	1.25.509882269	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2858	255	1.25.509882265	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2851	255	1.25.509882264	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2852	255	1.25.509882265	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
1180	255	1.451373020	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QU" quest.
12818	255	75.507569660	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QM" quest.
12561	255	78.567487619	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QM" quest..
13269	255	87.681387937	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QM" quest..
18851	255	1.134.49841999	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
18852	255	1.134.49841999	100.64.154.254	felixx-xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
19848	255	340.929138747	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QU" quest..
19852	255	141.955810993	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QM" quest..
23834	255	144.924217109	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QM" quest..
21865	255	154.339292380	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QM" quest..
21539	255	154.372588368	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QM" quest..
22148	255	158.441338164	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QM" quest..
22784	255	167.657466409	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QM" quest..
22852	255	168.579565631	100.64.154.245	224.0.0.251	MDNS	198	Standard query 0x0000 PTR lb_.dns-sd._udp.local. "QM" question PTR companion-link._tcp.local. "QM" quest..

Abbildung 34: 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