

Praktikum Rechnernetze

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

Jakob Waibel Daniel Hiller Elia Wüstner Felix Pojtinger

2021-10-26

Einführung

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

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.

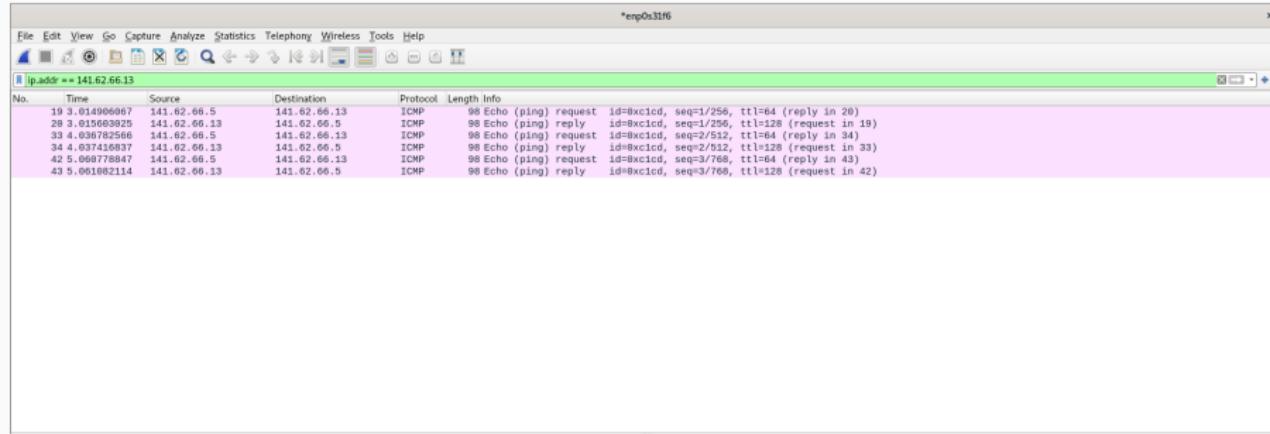
Starten Sie Wireshark und capturern Sie den aktuellen Traffic. Dokumentieren Sie zunächst, was alles auf Wireshark einprasselt.

*ens31:6										
No.	Time	Source	Destination	Protocol	Length	Info	Frame	HTTP	Raw	Display Filter
1	0.000000000	192.168.1.100	192.168.1.100	ICMP	80	ICMP Echo Request (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
2	0.000519572	192.168.1.100	192.168.1.100	Spanning-tree (Forw)	80	Spanning-tree (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
3	2.994421339	192.168.1.100	192.168.1.100	Spanning-tree (Forw)	80	Spanning-tree (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
4	4.994472633	192.168.1.100	192.168.1.100	Spanning-tree (Forw)	80	Spanning-tree (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
5	6.994523927	192.168.1.100	192.168.1.100	Spanning-tree (Forw)	80	Spanning-tree (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
6	8.994572336	192.168.1.100	192.168.1.100	Spanning-tree (Forw)	80	Spanning-tree (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
7	9.975649665	141.62.06.19	141.62.06.255	BROWSER	216	Become Backup Browser	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
8	10.994429210	192.168.1.100	192.168.1.100	Spanning-tree (Forw)	80	Spanning-tree (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
9	12.994478513	192.168.1.100	192.168.1.100	Spanning-tree (Forw)	80	Spanning-tree (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
10	14.994527807	192.168.1.100	192.168.1.100	Spanning-tree (Forw)	80	Spanning-tree (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
11	16.994697521	192.168.1.100	192.168.1.100	Spanning-tree (Forw)	80	Spanning-tree (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		
12	18.994490784	192.168.1.100	192.168.1.100	Spanning-tree (Forw)	80	Spanning-tree (Forw)	119	MGT_Root = 32768/0/001:1e:1be:c0 Cost = 220020 Port = 0x00002		

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	590	DHCP Discover - Transaction ID 0x6269e53eb
48	36.248724335	ognsense-router.rml...	255.255.255.255	DHCP	348	DHCP Offer - Transaction ID 0x6269e53eb
55	46.250842623	0.0.0.0	255.255.255.255	DHCP	540	DHCP Request - Transaction ID 0x6269e53eb
56	48.2598518738	ognsense-router.rml...	255.255.255.255	DHCP	348	DHCP ACK - Transaction ID 0x6269e53eb
57	48.259797973	linux.local	Broadcast	ARP	60	Who has 141.62.66.236? Tell 141.62.66.4
58	48.278416173	linux.local	Broadcast	ARP	60	Who has 141.62.66.250? Tell 141.62.66.4
63	45.478669439	fog.rnlabor.hds-stu...	linux.local	ARP	60	Who has 141.62.66.47? Tell 141.62.66.236
65	46.582657513	fog.rnlabor.hds-stu...	linux.local	ARP	60	Who has 141.62.66.47? Tell 141.62.66.236
79	47.526653895	fog.rnlabor.hds-stu...	linux.local	ARP	60	Who has 141.62.66.47? Tell 141.62.66.236
72	48.250842623	0.0.0.0	255.255.255.255	DHCP	348	DHCP Discover - Transaction ID 0xc1478931
73	48.498452075	ognsense-router.rml...	255.255.255.255	DHCP	348	DHCP Offer - Transaction ID 0xc1478931
79	50.529353459	0.0.0.0	255.255.255.255	DHCP	463	DHCP Request - Transaction ID 0xc1478931
88	58.531124992	ognsense-router.rml...	255.255.255.255	DHCP	348	DHCP ACK - Transaction ID 0xc1478931
81	50.531125138	linux.local	Broadcast	ARP	60	ARP Announcement for 141.62.66.4
82	50.584564928	linux.local	Broadcast	ARP	60	Who has 141.62.66.236? Tell 141.62.66.4
85	54.628510700	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 0xadc90d59
93	66.342356749	0.0.0.0	255.255.255.255	DHCP	345	DHCP Request - Transaction ID 0xadc90d59
95	66.629416649	linux.local	Broadcast	ARP	60	Who has 141.62.66.250? Tell 141.62.66.4

Abbildung 9: Gesamter Bootprozess

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:

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

dns && frame.number<20						
No.	Time	Source	Destination	Protocol	Length	Info
11	1.3513558000	rn05.rnLabor.hdm-st... opnsense-router.rnL...	DNS	93	Standard query 0xa278 A google.com OPT	
12	1.371692078	opnsense-router.rnL...	rn05.rnLabor.hdm-st... DNS	97	Standard query response 0xa278 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.

No.	Time	Source	Destination	Protocol	Length	Info
214	110.515570213	Linux-2.local	Broadcast	ARP	42	who has 141.02.66.6? Tell 141.02.66.5
215	110.515867208	Linux-3.local	Linux-2.local	ARP	60	141.02.66.6 is at 4c:52:62:0e:54:2b
231	110.678164795	Linux-3.local	Linux-2.local	ARP	60	who has 141.02.66.6? Tell 141.02.66.6
232	110.678386798	Linux-2.local	Linux-3.local	ARP	42	141.02.66.5 is at 4c:52:62:0e: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 Rahmenzyklus wird für die Anfrage verwendet?

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	70.000137340	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
174	71.000137378	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
175	72.000751507	Linux-3.local	224.0.0.255	PDNS	82	Standard query 0x0009 PTR _ppkey-Mp._tcp.local. "Qn question
176	72.000751507	Linux-3.local	224.0.0.255	PDNS	82	Standard query 0x0009 PTR _ppkey-Mp._tcp.local. "Qn question
177	75.000536089	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
178	77.000639082	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
179	79.000639082	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
180	80.000632388	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
181	83.000533792	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
182	84.000540741	Libremes-226.rnlabor.de	Broadcast	ARP	69	Who has 141.02.66.29? Tell 141.02.66.226
183	84.731177879	Libremes-226.rnlabor.de	Broadcast	ARP	69	Who has 141.02.66.227? Tell 141.02.66.226
184	84.731177879	Libremes-226.rnlabor.de	Broadcast	ARP	69	Who has 141.02.66.228? Tell 141.02.66.228
185	85.761491539	Libremes-226.rnlabor.de	Broadcast	ARP	69	Who has 141.02.66.227? Tell 141.02.66.226
186	85.954876527	linux-2.local	openseNSE.rnlabor.hub	DNS	88	Standard query 0x9e2a PTR 226.66.62.141.in-addr.arpa
187	85.955623699	openseNSE.rnlabor.hub	linux-2.local	DNS	137	Standard query response 0x9e2a PTR 226.66.62.141.in-addr.arpa PTR libremes-226.rnlabor.hub.hdm-stuttgart.de
188	86.721547449	Libremes-226.rnlabor.de	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
189	86.721547449	Libremes-226.rnlabor.de	Broadcast	ARP	69	Who has 141.02.66.29? Tell 141.02.66.226
190	86.785467391	Libremes-226.rnlabor.de	Broadcast	ARP	69	Who has 141.02.66.227? Tell 141.02.66.226
191	87.009791212	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
192	88.620794508	Linux-3.local	224.0.0.255	PDNS	81	Standard query 0x0009 PTR _neww-0183._tcp.local. "Qn question
193	88.620794508	Linux-3.local	224.0.0.255	PDNS	81	Standard query 0x0009 PTR _neww-0183._tcp.local. "Qn question
194	91.0007505494	Linux-2.local	openseNSE.rnlabor.hub	ARP	42	Who has 141.02.66.258? Tell 141.02.66.5
195	91.0007505494	openseNSE.rnlabor.hub	linux-2.local	ARP	69	141.02.62.66.258 is at 00:00:00:4f:bb:14
196	91.0006340482	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
197	93.000571935	HewlettPc.an:00:be	LLDP_Multicast	LLDP	312	Ma/0/4:0x973:an:80:08:BF LA/2 128 SysName:219-HP-2929-240-91424 SystID:9726A_2929-240 Switch, revision wB.10.10.0015, ROM wB.10.00.03 -
198	93.000571935	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002
199	95.000795615	HewlettPc.an:00:be	Spanning-tree-(For-)	STP	119	MST. Root = 32768/0/00:1a:c1:8e:b0:c Cost = 228020 Port = 0x80002

Ergebnis: 5673. All frames on wire (2000 bytes) All bytes captured (2000 bytes) on interface analysis_id 8

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	7.598625	100.64.84.66	141.78.124.5	DNS	88	Standard query 0x189d A news.ycombinator.com
715	7.598881	100.64.84.66	141.78.124.5	DNS	88	Standard query 0x85d1f AAAA news.ycombinator.com
716	7.608834	141.78.124.5	100.64.84.66	DNS	158	Standard query response 0x58df AAAA news.ycombinator.com SOA ns-225.awsdns-28.com
717	7.613971	141.78.124.5	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.tl
718	7.614386	100.64.84.66	209.216.230.248	TCP	78	49314 - 443 [SYN, ECN, CWR] Seq=0 Win=65535 Len=8 MSS=1468 Len=8 TSeqval=2512581059 TSeqcr=8 SACK_PERM=1
719	7.615218	209.216.230.248	100.64.84.66	TCP	74	443 - 49314 [SYN, ACK, ECN] Seq=0 Win=65532 Len=8 MSS=64 TSeqval=2845828468 TSeqcr=2512581059
720	7.765334	100.64.84.66	209.216.230.248	TCP	66	49314 - 443 [ACK] Seq=1 Ack=1 Win=131712 Len=8 TSeqval=2512581211 TSeqcr=2045828468
721	7.765826	100.64.84.66	209.216.230.248	TLSv1_	583	Client Hello
722	7.917493	209.216.230.248	100.64.84.66	TLSv1_	1514	Server Hello
723	7.917494	209.216.230.248	100.64.84.66	TCP	1514	443 - 49314 [ACK] Seq=1498 Ack=518 Win=65664 Len=1448 TSeqval=2045828612 TSeqcr=2512581211 [TCP segment of a reassembled PDU]
724	7.917495	209.216.230.248	100.64.84.66	TLSv1_	1862	Certificate [Certificate Status, Server Key Exchange, ServerHelloDone]
725	7.917581	100.64.84.66	209.216.230.248	TCP	66	49314 - 443 [ACK] Seq=518 Ack=3983 Win=127672 Len=8 TSeqval=2512581363 TSeqcr=2045828612
726	7.917585	100.64.84.66	209.216.230.248	TCP	66	[TCP Window Update] 49314 - 443 [ACK] Seq=518 Ack=3983 Win=31072 Len=8 TSeqval=2512581363 TSeqcr=2045828612
727	7.937248	100.64.84.66	209.216.230.248	TLSv1_	192	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
728	7.937649	100.64.84.66	209.216.230.248	TLSv1_	786	Application Data
729	8.88785	209.216.230.248	100.64.84.66	TCP	66	443 - 49314 [ACK] Seq=8993 Ack=1364 Win=64832 Len=8 TSeqval=2045828783 TSeqcr=2512581383
730	8.93869	209.216.230.248	100.64.84.66	TLSv1_	324	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
731	8.93957	100.64.84.66	209.216.230.248	TCP	66	49314 - 443 [ACK] Seq=1364 Ack=1364 Win=138752 Len=8 TSeqval=2512581359 TSeqcr=2045828788
732	8.96295	209.216.230.248	100.64.84.66	TCP	1514	443 - 49314 [ACK] Seq=1514 Ack=1364 Win=65664 Len=1448 TSeqval=2045828789 TSeqcr=2512581383 [TCP segment of a reassembled PDU]
733	8.96296	209.216.230.248	100.64.84.66	TCP	1514	443 - 49314 [ACK] Seq=5599 Ack=1364 Win=65664 Len=1448 TSeqval=2045828789 TSeqcr=2512581383 [TCP segment of a reassembled PDU]
734	8.96296	209.216.230.248	100.64.84.66	TCP	1514	443 - 49314 [ACK] Seq=8747 Ack=1364 Win=65664 Len=1448 TSeqval=2045828789 TSeqcr=2512581383 [TCP segment of a reassembled PDU]
735	8.896297	209.216.230.248	100.64.84.66	TCP	1514	443 - 49314 [ACK] Seq=8495 Ack=1364 Win=1448 TSeqval=2045828789 TSeqcr=2512581383 [TCP segment of a reassembled PDU]
736	8.96298	209.216.230.248	100.64.84.66	TLSv1_	681	Application Data
737	8.96371	100.64.84.66	209.216.230.248	TCP	66	49314 - 443 [ACK] Seq=1364 Ack=18558 Win=124688 Len=8 TSeqval=2512581542 TSeqcr=2045828789
738	8.96484	100.64.84.66	209.216.230.248	TCP	66	[TCP Window Update] 49314 - 443 [ACK] Seq=1364 Ack=18558 Win=131872 Len=8 TSeqval=2512581542 TSeqcr=2045828789
739	8.223352	100.64.84.66	209.216.230.248	TLSv1_	691	Application Data
740	8.250009	100.64.84.66	209.216.230.248	TCP	70	49314 - 443 [ACK] Seq=1364 Win=65664 Len=1448 TSeqval=2045828789 TSeqcr=2512581542 [TCP segment of a reassembled PDU]
741	8.474585	209.216.230.248	100.64.84.66	TCP	1514	443 - 49314 [ACK] Seq=1809 Win=65664 Len=1448 TSeqval=2045829070 TSeqcr=2512581669 [TCP segment of a reassembled PDU]
742	8.374587	209.216.230.248	100.64.84.66	TLSv1_	823	Application Data
743	8.374653	100.64.84.66	209.216.230.248	TCP	66	49314 - 443 [ACK] Seq=1989 Ack=12763 Win=128832 Len=8 TSeqval=2512581829 TSeqcr=2045829076
744	8.376901	100.64.84.66	209.216.230.248	TLSv1_	674	Application Data
750	8.419434	209.216.230.248	100.64.84.66	TCP	74	443 - 49314 [SYN, ACK, ECN] Seq=0 Ack=1 Win=65535 Len=8 MSS=64 SACK_PERM=1 TSeqval=1535768379 TSeqcr=3827897587
751	8.419586	100.64.84.66	209.216.230.248	TCP	66	49315 - 443 [ACK] Seq=1 Ack=1 Win=131712 Len=8 TSeqval=3827897754 TSeqcr=1535768379
752	8.424337	100.64.84.66	209.216.230.248	TLSv1_	585	Client Hello
759	8.527067	209.216.230.248	100.64.84.66	TCP	1514	443 - 49314 [ACK] Seq=12763 Ack=2597 Win=65664 Len=1448 TSeqval=2045829221 TSeqcr=2512581821 [TCP segment of a reassembled PDU]
766	8.527068	209.216.230.248	100.64.84.66	TLSv1_	793	Application Data
761	8.527151	100.64.84.66	209.216.230.248	TCP	66	49314 - 443 [ACK] Seq=19338 Win=128896 Len=8 TSeqval=2512581972 TSeqcr=2045829221
762	8.591413	209.216.230.248	100.64.84.66	TLSv1_	222	Server Hello, Change Cipher Spec, Encrypted Handshake Message
763	8.591467	100.64.84.66	209.216.230.248	TCP	66	49315 - 443 [ACK] Seq=528 Ack=136 Win=131896 Len=8 TSeqval=3827897926 TSeqcr=1535768058
764	8.591689	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
176 03.99873034		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
173 05.99938275		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
172 07.99949448		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
173 79.99813730		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
174 12.99908220		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
176 73.999729543		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
177 75.999566699		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
178 77.999639982		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
179 79.999639982		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
180 81.999692369		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
181 83.999531792		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
180 85.999023094		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
193 87.999791212		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
194 91.999639982		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
196 93.999871926		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
199 95.99979642		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
200 97.999550051		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
202 101.999639982		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
203 101.999550734		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
204 103.99977330		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
206 105.999642785		HewlettP_aa:8b:be	Spanning-tree-(For...)	STP	119 MST. Root = 32768/0/08:1a:c1:5e:eb:c0 Cost = 220020 Port = 0x8002
212 108.999870479		HewlettP_aa:8b: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	182.0000115609	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
394	184.0010050820	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
395	186.0000050817	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
396	188.0000050817	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
398	190.0000150340	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
406	192.0000050827	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
407	194.0000071180	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
409	196.0000039980	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
411	198.0000039980	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
412	200.000005267849	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
413	202.0000051763	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
417	204.000005267849	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
418	206.0000051952	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
420	208.0000051952	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
424	210.000005260871	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
425	212.000005260871	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
426	214.0000050847	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
427	216.0000050847	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
428	218.0000050847	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
430	220.000005140854	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002
432	222.0001177244	HeuletCP_aa:0b:be	Spanning-tree-(for-bridges)	STP	119	MST. Root = 32768/0/0/0:la:c1:5e:eb:c9 Cost = 220629 Port = 0x8002

Frame 426: 119 bytes on wire (952 bits), 119 bytes captured (952 bits) on interface emps31f0, id 0

Interface: 0 (Ethernet)

Destination: Spanning-tree-(for-bridges).00 (01:00:c0:00:00:00)

Address: Spanning-tree-(for-bridges).00 (01:00:c0:00:00:00)

... .0.0.0. = 0 bit Globally unique address (factory default)

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

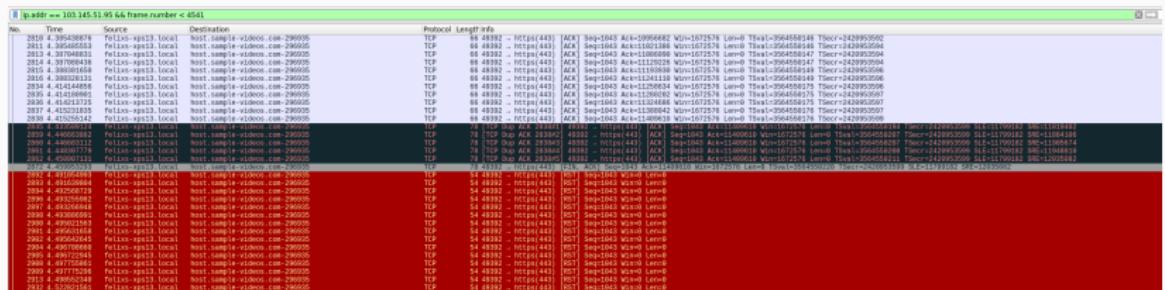


Abbildung 28: Capture beim Canceln des eines Downloads über HTTPS

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, in welchem hier die FIN- und ACK-Flags gesetzt wurden, ist dementsprechend zu erkennen.

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	13.371899778	141.62.66.5	141.62.66.207	TELNET	69	Telnet Data ...
55	13.371900177	141.62.66.207	141.62.66.5	TELNET	69	Telnet Data ...
57	13.371900575	141.62.66.5	141.62.66.207	TELNET	69	Telnet Data ...
58	13.372142487	141.62.66.207	141.62.66.5	TELNET	86	Telnet Data ...
61	13.373263991	141.62.66.207	141.62.66.5	TELNET	69	Telnet Data ...
65	15.536484821	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
67	15.537220775	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
69	15.537221173	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
71	15.713143890	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
73	15.784452602	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	15.804300534	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
77	15.804300932	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
79	15.991754757	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	16.056306088	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
83	16.056306486	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
85	16.176481695	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
87	16.177386417	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...
89	16.344425088	141.62.66.5	141.62.66.207	TELNET	67	Telnet Data ...
90	16.34581998	141.62.66.207	141.62.66.5	TELNET	67	Telnet Data ...

Abbildung 30: Capture des Telnet-Logins

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	1.4.77085609	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
29	255	1.4.77088579	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
31	255	1.4.819793372	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
80	255	1.4.898431133	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
98	255	1.4.930200000	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
112	255	1.4.954393550	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
113	255	1.4.954393675	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
1527	255	1.21.511686375	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
1867	255	1.614986013	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2030	255	1.35.100000000	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2044	255	1.25.456639749	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2045	255	1.25.456639783	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2049	255	1.25.509882269	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2060	255	1.25.509882268	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2081	255	1.25.509882264	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
2092	255	1.25.509882262	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
11826	255	74.573789520	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" quest.
12018	255	75.597595960	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" quest.
12201	255	78.597595960	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" quest.
13269	255	80.681387937	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _Companion-link._tcp.local. "Q" quest.
16051	255	1.134.498471999	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
16066	255	1.134.622131475	100.64.154.254	felix.xps13.local	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
16046	255	140.929138747	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" quest.
16052	255	140.929138748	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" quest.
20394	255	144.934237189	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" quest.
21865	255	154.345939268	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" quest.
21935	255	158.472517384	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" quest.
22148	255	158.474133816	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" quest.
22798	255	160.474133816	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" quest.
22852	255	166.579566333	100.64.154.245	224.0.251	MDNS	198	Standard query 0x0000 PTR lb._dns-sd._udp.local. "Q" question PTR _companion-link._tcp.local. "Q" 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 „ungeübliche“ Pakete wie z.B.