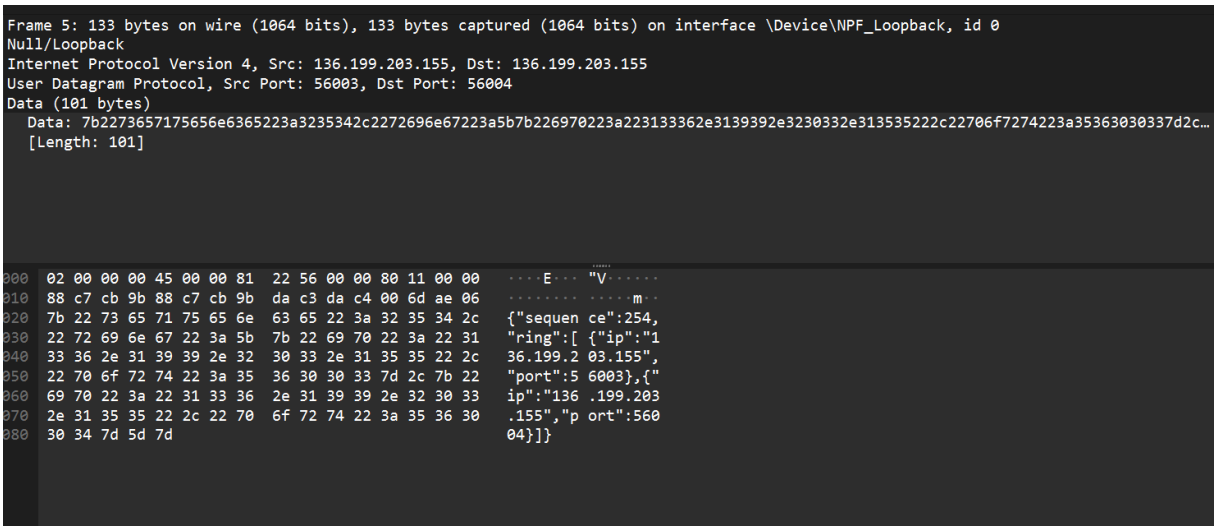
Protokoll Leonard Richertz 1560736

Lokal:

Beim Analysieren des TokenRings lokal auf meinem Rechner, konnte ich folgendes feststellen. Zum einen sieht man in Wireshark die IP-Adresse des Senders und Empfängers der Pakete, was lokal offensichtlich die gleiche Adresse ist. Ebenso findet man, dass der Destination Port sowie der Source Port immer zwischen den spezifizierten alterniert. Beim genaueren Betrachten des Pakets erkennt man ebenfalls, dass die sequence, in diesem Fall 254 erhalten ist, welche auch in der Konsole ausgegeben wird.



Mittels des „Adapter for Loopback Traffic capture“ konnte ich den Traffic des Java Programms isolieren und dann analysieren. Mittels der Display filter Syntax wäre dies wahrscheinlich auch möglich gewesen.

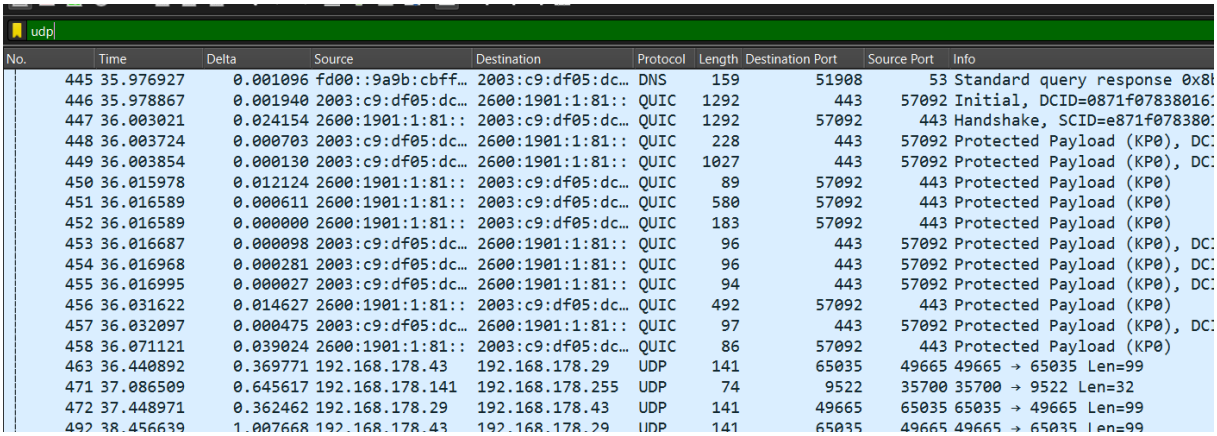
Mit mehreren PCs:

Um das Programm mit mehren PCs zu testen musste ich zunächst die Firewall auf beiden PCs abschalten und dann hat der Datenaustausch auch funktioniert.

Hierbei lässt sich wieder erkennen, dass beide Pakete mit dem UDP Protokoll jeweils an die richtigen Ports geschickt wurden (65035 und 49665)

0.296194	192.168.178.29	192.168.178.43	UDP	141	49665	65035
0.174141	fe80::b2bd:82a5…	ff02::fb	MDNS	191	5353	5353
0.000239	192.168.178.43	224.0.0.251	MDNS	186	5353	5353
0.015092	2003:c9:df05:dc…	2603:1026:2405:…	TLSv…	109	443	56863
0.070611	2603:1026:2405:…	2003:c9:df05:dc…	TCP	74	56863	443
0.084138	2003:c9:df05:dc…	2603:1026:2405:…	TLSv…	109	443	56866
0.064142	2603:1026:2405:…	2003:c9:df05:dc…	TCP	74	56866	443
0.420903	Intel_01:df:3d	Broadcast	ARP	42		
0.175688	192.168.178.43	192.168.178.29	UDP	141	65035	49665

Hierbei konnte ich auch bereits die Display Filter Syntax von Wireshark ausprobieren und somit nur alle UDP Pakete untersuchen.



udp

No.	Time	Delta	Source	Destination	Protocol	Length	Destination Port	Source Port	Info
445	35.976927	0.001096	fd00::9a9b:cbff…	2003:c9:df05:dc…	DNS	159	51908	53	Standard query response 0x8
446	35.978867	0.001940	2003:c9:df05:dc…	2600:1901:1:81::	QUIC	1292	443	57092	Initial, DCID=0871f078380161
447	36.003021	0.024154	2600:1901:1:81::	2003:c9:df05:dc…	QUIC	1292	57092	443	Handshake, SCID=e871f078380
448	36.003724	0.000703	2003:c9:df05:dc…	2600:1901:1:81::	QUIC	228	443	57092	Protected Payload (KP0), DC
449	36.003854	0.000130	2003:c9:df05:dc…	2600:1901:1:81::	QUIC	1027	443	57092	Protected Payload (KP0), DC
450	36.015978	0.012124	2600:1901:1:81::	2003:c9:df05:dc…	QUIC	89	57092	443	Protected Payload (KP0)
451	36.016589	0.000611	2600:1901:1:81::	2003:c9:df05:dc…	QUIC	580	57092	443	Protected Payload (KP0)
452	36.016589	0.000000	2600:1901:1:81::	2003:c9:df05:dc…	QUIC	183	57092	443	Protected Payload (KP0)
453	36.016687	0.000098	2003:c9:df05:dc…	2600:1901:1:81::	QUIC	96	443	57092	Protected Payload (KP0), DC
454	36.016968	0.000281	2003:c9:df05:dc…	2600:1901:1:81::	QUIC	96	443	57092	Protected Payload (KP0), DC
455	36.016995	0.000027	2003:c9:df05:dc…	2600:1901:1:81::	QUIC	94	443	57092	Protected Payload (KP0), DC
456	36.031622	0.014627	2600:1901:1:81::	2003:c9:df05:dc…	QUIC	492	57092	443	Protected Payload (KP0)
457	36.032097	0.000475	2003:c9:df05:dc…	2600:1901:1:81::	QUIC	97	443	57092	Protected Payload (KP0), DC
458	36.071121	0.039024	2600:1901:1:81::	2003:c9:df05:dc…	QUIC	86	57092	443	Protected Payload (KP0)
463	36.440892	0.369771	192.168.178.43	192.168.178.29	UDP	141	65035	49665	49665 → 65035 Len=99
471	37.086509	0.645617	192.168.178.141	192.168.178.255	UDP	74	9522	35700	35700 → 9522 Len=32
472	37.448971	0.362462	192.168.178.29	192.168.178.43	UDP	141	49665	65035	65035 → 49665 Len=99
492	38.456639	1.007668	192.168.178.43	192.168.178.29	UDP	141	65035	49665	49665 → 65035 Len=99

Allgemein konnte ich mit der Übung nicht nur das PacketSniffer Tool Wireshark ausprobieren, sondern auch die Funktionsweise der Firewall an einem praktischen Beispiel sehen.