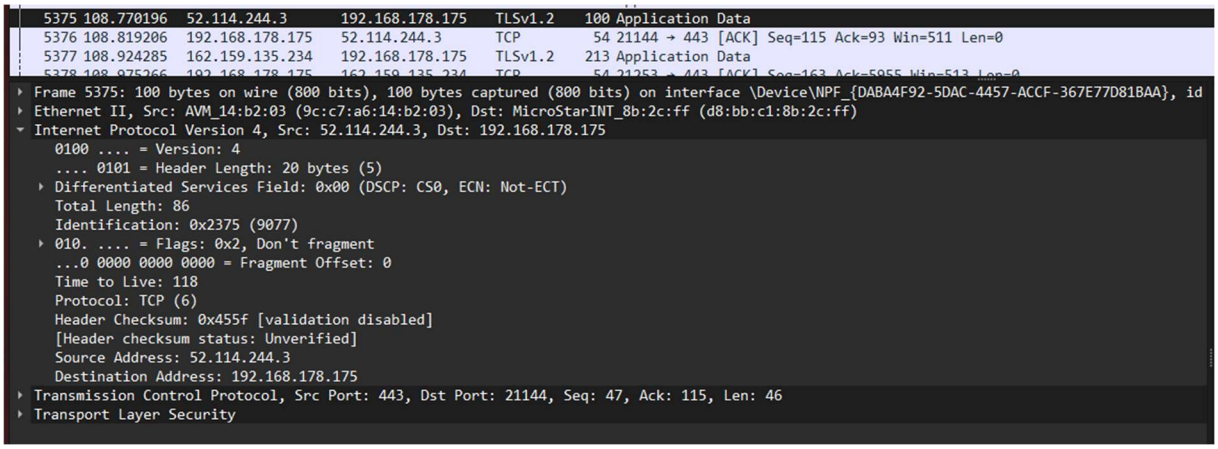
Aufgabe 1:

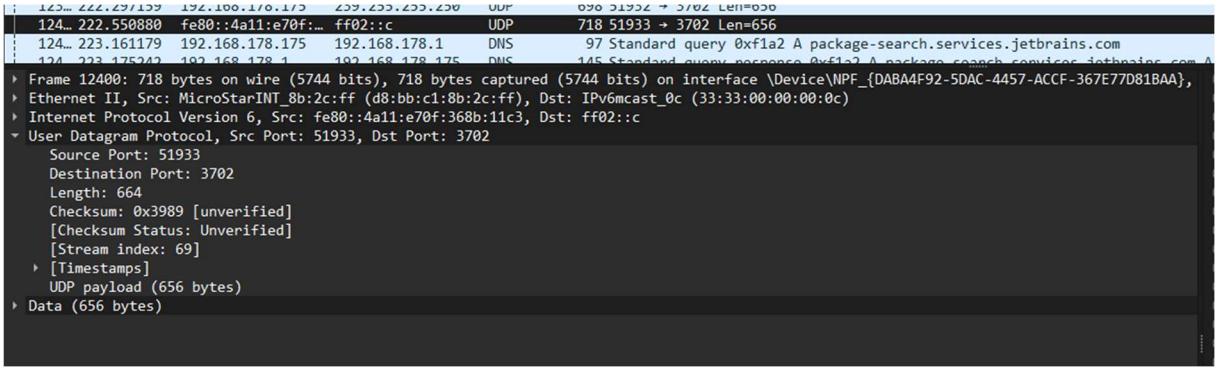
Um ein beliebiges IPv4-Protokoll zu finden, kann der Suchstring „ip“ verwendet werden. Ein beliebiges Paket ist:



Es können die folgenden Daten ausgelesen werden:

- Version: 4
- Header Length: 20 bytes (5)
- Identification: 0x2375 (9077)
- Time to Live: 118
- Protocol: TCP (6)
- Header Checksum: 0x455f [validation disabled]
- Source Address: 52.114.244.3
- Destination Address: 192.168.178.175
- Differentiated Services Field: 0x00 (In der VL als „tos“ betitelt)

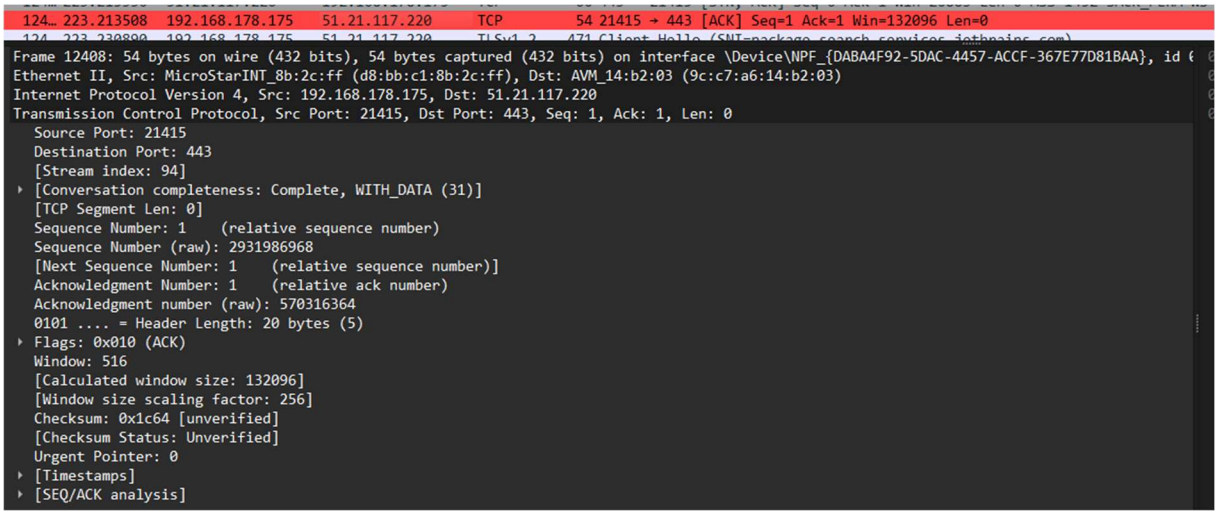
Für UPD wird analog verfahren:



Es können die folgenden Daten ausgelesen werden:

- Source Port: 51933
- Destination Port: 3702
- Length: 664
- Checksum: 0x3989 [unverified]

Für TCP wird analog verfahren:



Es können die folgenden Daten ausgelesen werden:

- Source Port: 21415
- Destination Port: 443
- Sequence Number: 1 (relative sequence number)
- Sequence Number (raw): 2931986968
- Acknowledgment Number: 1 (relative ack number)
- Acknowledgment number (raw): 570316364
- Window: 516
- Checksum: 0x1c64 [unverified]
- Urgent Pointer: 0

Aufgabe 2:

Bei der Adresse 103.161.122.83 handelt es sich um eine IPv4-Adresse.
Die 18 gibt an, dass es sich hier um die CIDR-Notation handelt, das bedeutet, dass die ersten 18 Bits der Subnetzmaske für das Netzwerk verwendet werden und die restlichen 14 Bits werden für Hosts im Netzwerk verwendet.

Die **Subnetzmaske** kann wie folgt ermittelt werden:

Mit der CIDR-Notation geht für die Subnetzmaske einher, dass die ersten 18 Bits auf 1 gesetzt, die folgenden 14 Bits auf 0 gesetzt werden. Die 32 Bits werden dann in 8 Bit-Blöcke unterteilt, damit die Subnetzmaske ermittelt werden kann:

11111111 = 255

11111111 = 255

11000000 = 192

00000000 = 0

Die Subnetzmaske ist also: 255.255.192.0

Die **Netzwerkadresse** erhält man, wenn man ein logisches UND zwischen IP-Adresse und der Subnetzmaske nutzt.

IP-Adresse: 103.161.122.83 → 01100111.10100001.01111010.01010011

Subnetzmaske: 255.255.192.0 → 11111111.11111111.11000000.00000000

Durch ein logisches UND der beiden erhalten wir:

01100111.10100001.01000000.00000000 → 103.161.64.0

Die Broadcastadresse erhält man, indem die Host-Bits auf 1 gesetzt werden (die letzten 14 Bits der Adresse):

Netzwerkadresse: 103.161.64.0 → 01100111.10100001.01000000.00000000

Setzen der Host-Bits auf 1: 01100111.10100001.01111111.11111111

Das ergibt die Broadcastadresse:

01100111.10100001.01111111.11111111 → 103.161.127.255

Um herauszufinden, ob 103.161.122.83/18 und 103.161.193.83/18 im gleichen Netzwerk liegen, werden die Netzwerkadressen ermittelt.

Aus der vorigen Beschreibung ist bekannt, dass die Subnetzmaske für /18 wie folgt ist:

255.255.192.0

Damit kann die Netzwerkadresse für 103.161.122.83/18 berechnet werden, diese ist nach obigen Verfahren:

103.161.64.0

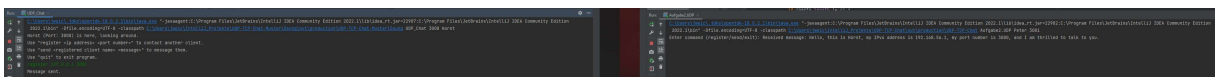
Gleiches wird durchgeführt für 103.161.193.83/18, wir erhalten die Netzwerkadresse:

103.161.192.0

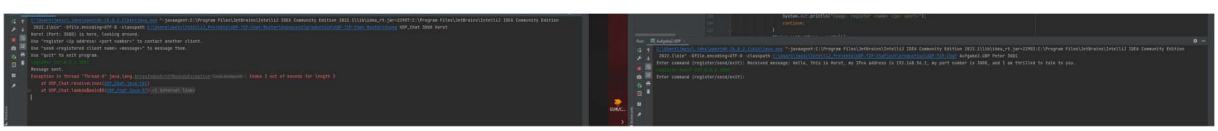
Daraus resultiert, dass 103.161.122.83/18 und 103.161.193.83/18 **nicht** im gleichen Netzwerk liegen, da sie nicht identisch sind.

Aufgabe 3:

UDP: Problematisch ist der Verbindungsaufbau. In meinem Versuch nutze ich die Implementierung von Herrn Sturm und meine eigene. Die Registrierung eines Chatteilnehmers von meiner Implementierung (Peter) bei der Sturm-Implementierung (Horst) ist kein Problem:



Versuche ich jedoch den Sturm-Chatpartner bei mir zu registrieren, so wird eine IndexOutOfBoundsException in der Sturm-Implementierung geworfen:



Dies liegt an unterschiedlichen Registrierungsimplementierungen. Das Verfahren von Sturm erhält 2 Argumente und registriert sich dann bei mir. Meine Implementierung nutzt 3 Argumente und registriert sich dann bei der Sturm-Implementierung. Problematisch ist hier, dass die Sturm-Implementierung nur 2 Argumente erwartet und daher eine IOOB geworfen wird.

Verläuft die Registrierung einwandfrei, dann macht das Senden auch kein Problem. Wichtig ist, dass *register* und *send* bei beiden Implementierungen gleich aufgebaut sind.

TCP: Ebenso wie bei UDP ist der ausschlaggebende Punkt, dass die Registrierung bei beiden Implementierungen gleich ist. Steht die Registrierung, so kann gechattet werden.