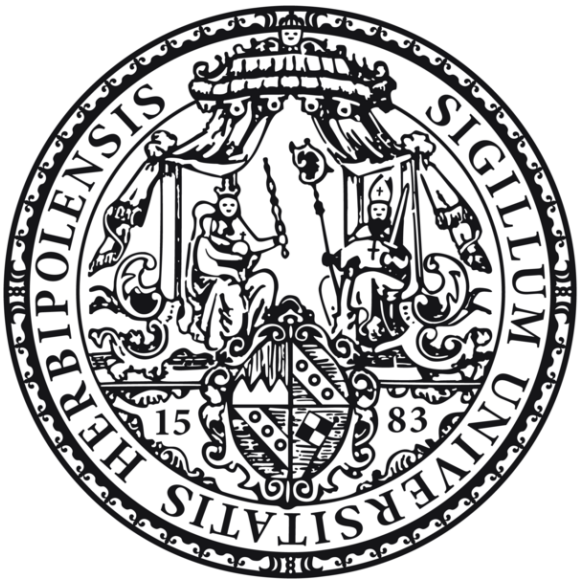
Hardwarepraktikum Internet-Technologien

Comprehension Questions of Task 7: Packet-loss and Latency



Julius-Maximilians-Universität Würzburg

Chair of Computer Science III

A project report submitted by **Group 11**

Enrique Mesonero Ronco

Manuel Calvo Martín

Pablo E. Ortega Ureña

Summer term 2022

July, 2022

Contents

7. Packet-loss and Latency

7.2. Network configuration

In order to emulate the different network conditions, we connect the NetEm to both the switch (on port 4) and the router (on the Internet port). We use the USB adapter to connect to the router and the native Ethernet interface to connect to the switch.

In addition, we configure the routing back to the static routes of section 5. For us to change the conditions, we change the way of reaching the Raspberry from the PC, changing the physical connections.

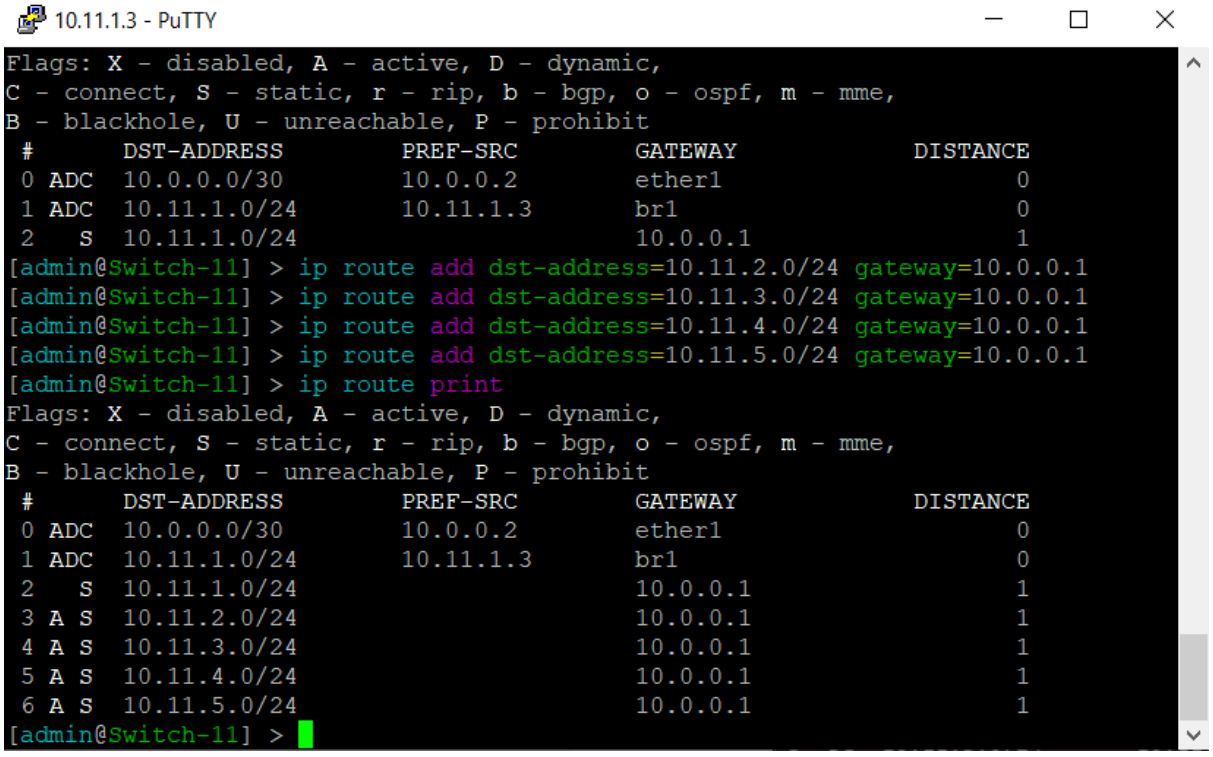


Figure 7.1. Switch Configuration

7.3. Subjective Influence of Latency and Packet Loss

The first thing we do is try a ping test. With the direct connection, our pings happen at a rate of around 0.6ms. However, with the NetEm connected, this almost triples, to around 1.6ms. The effects of the NetEm are already noticeable by the SSH connection, which is much slower.

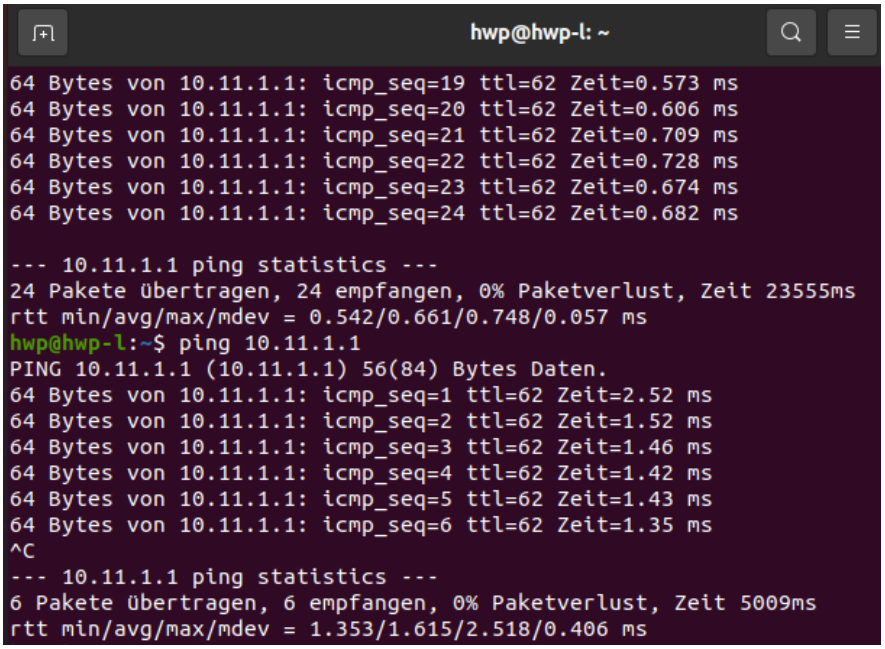


Figure 7.2. ICMP without and with the NetEm

Our second test is establishing a connection using Netcat. For that, we put the Raspberry listening on port 3333, using the command nc -l 3333. By default, Netcat uses TCP for the connections. If we wanted to use UDP, we should have put the flag -u.

On our end device, we connect to the Raspberry using nc 10.11.1.1 3333, in other words, the Raspberry's IP address and listening port. We write some lines on both devices, without any issue.

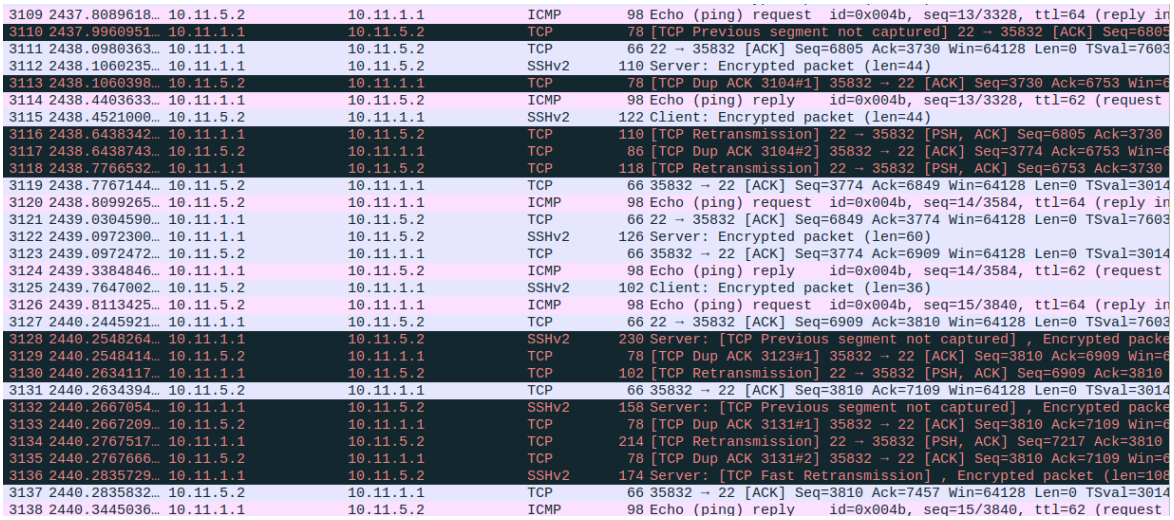


Figure 7.3. Packet loss on SSH with NetEm connected

We also try the same thing using the -u flag to do it via UDP. Again, no issues on either end, and no variation detected. On Wireshark, we see that

TCP used three kinds of packet in order to keep the synchronization, the first one being SYN, sent by the end device to the Raspberry, the second one the Raspberry's response, SYN ACK to do the handshake. Then we have PSH and ACK packets to acknowledge the reception of packets in order and request more. None of those packets are seen on the UDP connection.

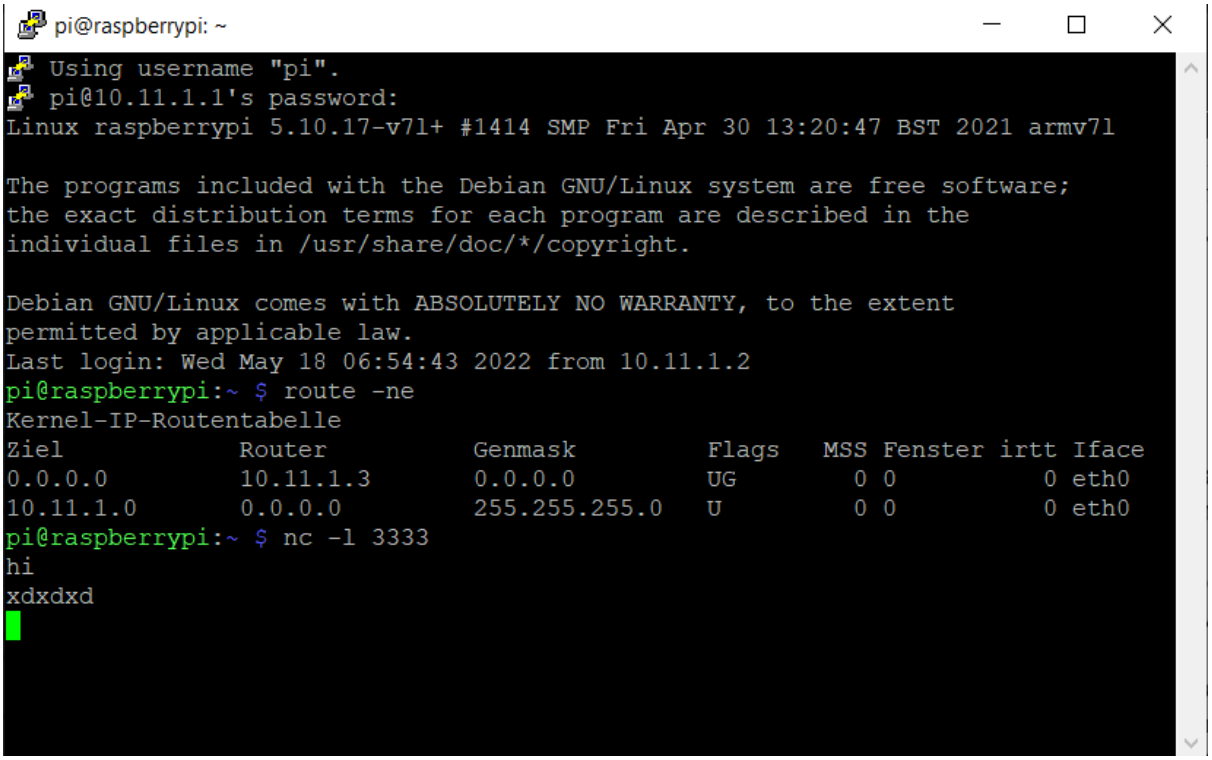


Figure 7.4. Raspberry receiving and sending messages

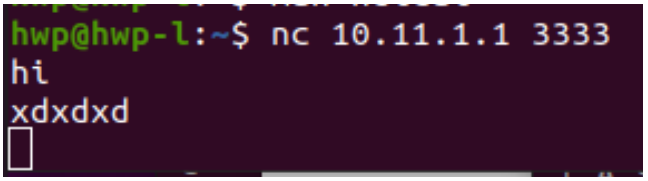


Figure 7.5. End device connection with the Raspberry

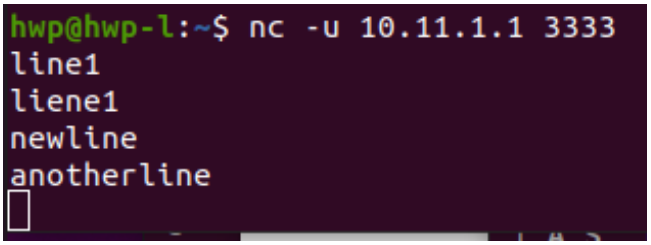


Figure 7.6. UDP communication

Next up, we try forwarding the USB microphone recording from the Raspberry to the end device using the network. We forward it using Netcat, and using the arecord/aplay interface.

On the sending device, which is the Raspberry, we need the arecord -f S16_LE -r 44100 -t raw command piped along the nc 10.11.5.2 3333 command. This sends a raw audio signal on a 16 bit little endian with a 44100 Hz sampling rate, to the server located on the IP address and port designated (our end device).

Meanwhile, on our end device, we open our 3333 port first using Netcat, with nc -l 3333 and we pipe that with the aplay -f S16_LE -r 44100 command, exactly what we had on the sending device in order to process the signal correctly.

We could establish a real-time audio transmission, both using TCP and UDP. There was no noticeable difference in the quality and delay, this last one being about half a second.

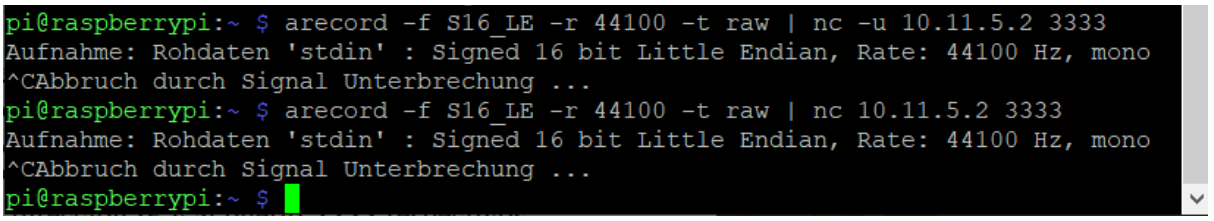


Figure 7.7. Sending audio via UDP and TCP

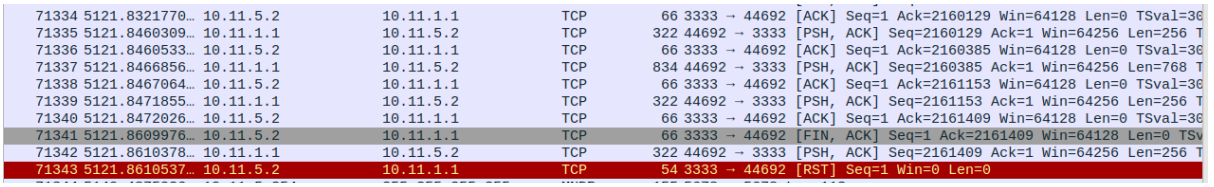


Figure 7.8. Audio through TCP

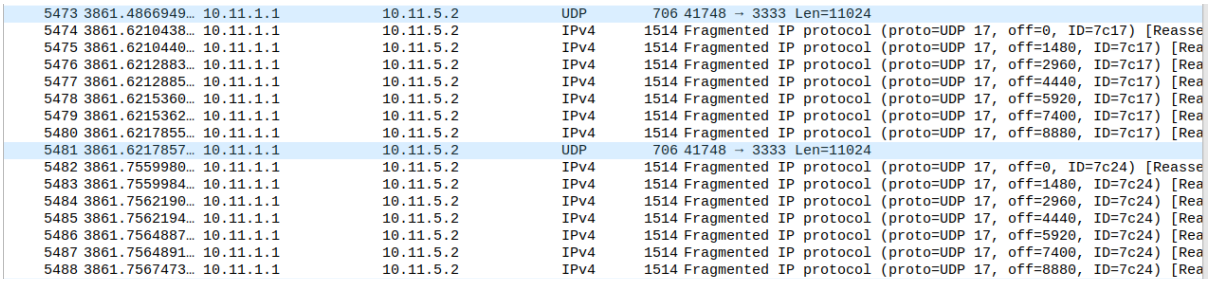


Figure 7.9. Audio through UDP

7.3.1. Latency minimisation and buffers

The default buffer time arecord and aplay use is 500ms, which was around the time we experienced when listening. In order to tweak this time, the buffer-time (-B) flag can be used. It measures in ms.

We tried lowering it from 500ms, however, no difference was noticed, it did not lower the reception time, using both TCP and UDP. We only detected anomalies using 0 as a buffer, where we could hear cuts in the audio.

7.3.2. Packet loss and latency

Now, we connected the NetEm to the network and ran the same commands. At first, we tried TCP, and the difference was highly noticeable. We heard the complete signal, but with a lot of stops and cuts, which was not pleasant.

When trying later with UDP, the result was similar in the amount of cuts, however, there was not as much delay. Instead, the audio was either lost or scrambled. However, with the default buffer, it was mostly unintelligible.

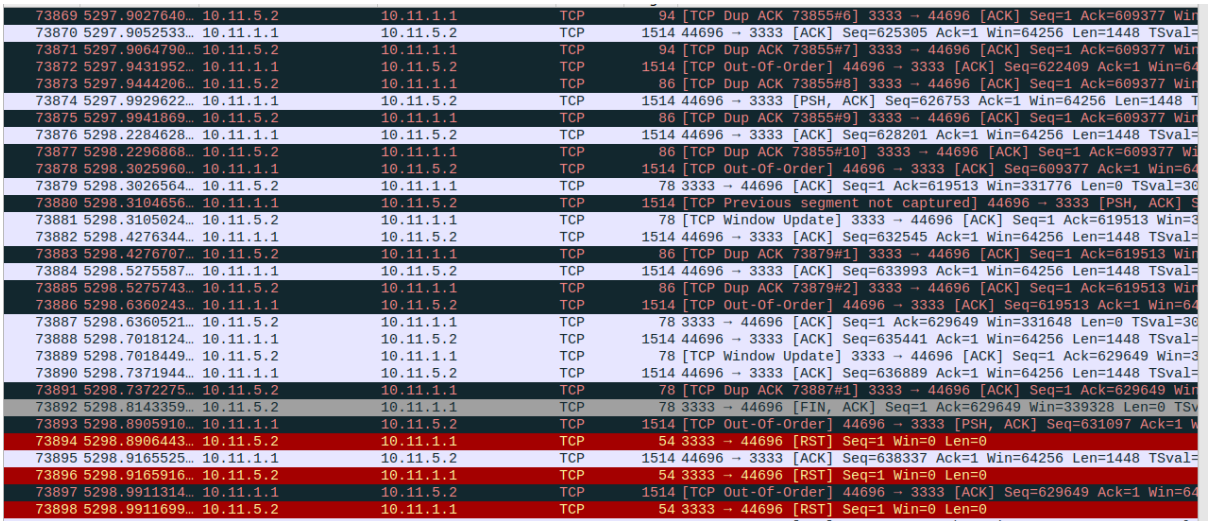


Figure 7.10. Audio through TCP using NetEm

```
hwp@hwp-l:~$ nc -u -l 3333 | aplay -f S16_LE -r 44100 -B 45000
Wiedergabe: Rohdaten 'stdin' : Signed 16 bit Little Endian, Rate: 44100 Hz, mono
Unterlauf!!! (mindestens 53,394 ms)
Unterlauf!!! (mindestens 214,517 ms)
Unterlauf!!! (mindestens 914,209 ms)
Unterlauf!!! (mindestens 378,894 ms)
Unterlauf!!! (mindestens 142,103 ms)
Unterlauf!!! (mindestens 118,247 ms)
Unterlauf!!! (mindestens 392,838 ms)
Unterlauf!!! (mindestens 883,585 ms)
Unterlauf!!! (mindestens 166,746 ms)
Unterlauf!!! (mindestens 762,234 ms)
Unterlauf!!! (mindestens 242,852 ms)
Unterlauf!!! (mindestens 873,072 ms)
Unterlauf!!! (mindestens 480,980 ms)
Unterlauf!!! (mindestens 315,377 ms)
^CAbbruch durch Signal Unterbrechung ...
aplay: pcm_write:2061: Schreibfehler: Unterbrechung während des Betriebssystemau
frufs
```

Figure 7.11. Warnings received on audio through UDP using NetEm

We experimented with various buffer sizes, obtaining different results for TCP and UDP.

TCP:

Buffer Size	Experience
0	Same as using 500
1	Lots of cuts, nothing can be heard
500	Bits are able to understand, but really delayed
3000	Slight improvement, but still really delayed
20000	Perfect point, audio is a bit delayed but is more constant and intelligible
100000	Cuts start to be experienced again

UDP:

Buffer Size	Experience
0	Same as using 500
1	Nothing intelligible
500	Barely anything intelligible
3000	Slight improvement, but still scrambled
20000	Biggest improvement from previous sizes, but still unpleasant
40000	Plateau point, from this point onwards, there is no improvement

We need to highlight that while with TCP we could reach a buffer time in which the packet loss and latency did not matter as much, as the audio was perfectly intelligible, with UDP that could not be reached. We avoided the audio scrambling with high buffer sizes. However, there were still some lost packets that made the experience worse.