Challenge 5

After some (pretty devastating) trial-and-error (~4 hrs), we managed to send a correct IPv6-packet with an empty TCP packet as payload. For some reason, Wireshark did not track any packets until we registered our IPv6-adress in the server. This was very surprising because intuitively, sending a packet should work independently from the receiving server.
We then created a working SYN-packet. We received the SYN/ACK from the server and responded with a final ACK packet.
We then tried to build the HTTP-GET-request inside the TCP-packet. We managed to figure out that we needed the „ISO-8859-1“ encoding and were able to get a response to our first HTTP request. It says „400- Bad Request“, and we did not have any clue why our request could be corrupted, since we basically copied it from the instructions. It was also displayed correctly within Wireshark. After some extended research, we found the source of our problem in the HTTP- specification: We need to two a CLFR at the end of the request, so normal ‚\n‘ (in Java) does not suffice. We added the required ‚\r‘ and finally got our first hidden keys.

We then proceeded to work on the next port/server. In order to receive the whole content of the website (which was artificially extended), we edited our receiving routine so that we send back appropriate ACK-packets upon receiving new data. The ACK-number is calculated as the sequence number of the received packet plus the data length of this packet. We had some problems with converting ints into byte arrays (caused by the fact that Java uses signed bytes and IP/TCP use unsigned), but finally managed to receive the secret phrases from the second server.

These are the received phrases:

7710:
Daniel (416096): The Gorilla Experiment
Jonas (428292): The Zarnecki Incursion

7711:
Daniel (416096): The Plimpton Stimulation
Jonas (428292): The Rhinitis Revelation

Since we already invested ~7 hrs to get to the second server, and our exam weeks already start in about 10 days, we decided to stop here.

Jonas Becker (428292) & Daniel Beckmann (416096)