Master's Thesis Engineering Technology

2021-2022

Hardware/Software co-design for accelerating decryption of 1-RTT QUIC packets

Purnal Lennert

Master of Electronics and ICT Engineering Technology

Introduction and problem

The QUIC protocol is a relatively new reliable transport layer protocol which aims to improve on TCP with TLS. Most implementations are executed in user space to improve flexibility of the protocol. This makes the protocol processor intensive and slower than TCP with TLS, with the main bottlenecks being **cryptography** and **kernel/user space data copy**. Thus, reducing the amount of times that both are executed on a processor is an important step to achieving better performance.

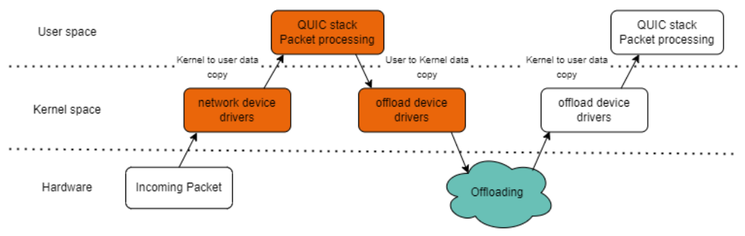

Figure 1: Architecture for crypto offload of incoming packets with extra user/kernel space data copy (excessive steps in red)

Objectives

The target is to create a hardware/software co-design that handles **packet protection**. The architecture should minimalize kernel/user space data copy. This has to be balanced with hardware complexity and resource usage. A general workflow for this architecture is shown in Figure 2. Parts of this design should then be implemented as a proof of concept (PoC) for **1 roundtrip time (1-RTT)** packets, which make up the main bulk of QUIC traffic, using the **TLS_AES_128_GCM_SHA256** cipher suite.

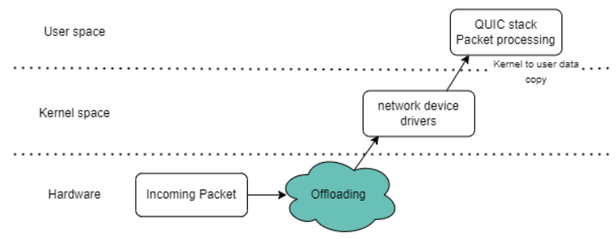

Figure 2: Architecture for cryptographic offloads of incoming packets immediately after arrival

Method and materials

Figure 3 shows the flow of the PoC hardware/software co-design for **arriving** packets.

The following steps are done in hardware:

- **Parsing** of the incoming Ethernet frame to **generate metadata** for the downstream modules
- **Connection ID lookup** to determine whether the packet belongs to an **existing connection**
- **Key lookup** based on the connection ID lookup to **gather** the **connection secrets**
- **Header decryption** based on the **AES_128_ECB** cipher
- **Payload decryption and packet authentication** based on the **AEAD_AES_128_GCM** cipher

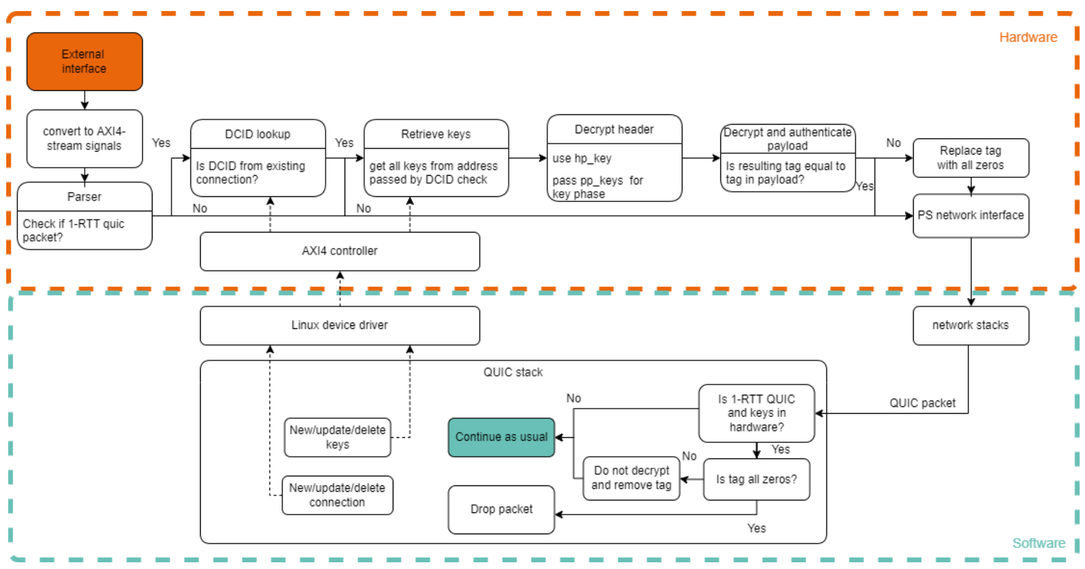

Figure 3: Flowchart of the hardware/software co-design for received ethernet frames

Following this, the packet is passed to the processor through a networking device. A check is done if the QUIC packet is 1-RTT and its DCID and keys were set in hardware. If so, the packet was decrypted and the **last 16 bytes** of the packet indicates whether or not it was succesfully authenticated. Finally, the stack has to **update the DCID table and key memory** with an **AXI-interface** controlled by a Linux device driver.

To **verify** the design, the POC was tested on a **Zynq 7000 FPGA** with an ARM Cortex-A9 processor on a **PYNQ-Z2** development board. A FIFO at the input is filled with packet data and metadata to simulate the **AXI4-stream** protocol and the DCIDs and keys are written to their respective modules. Using an enable signal, the packet is passed through the POC system and the output FIFO is filled with the decrypted packet. In software, the output FIFO is read and the packet is compared with a verification packet.

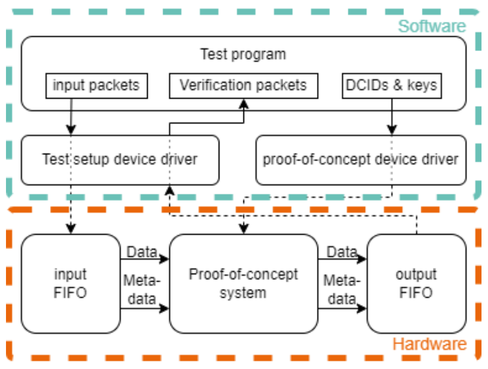

Figure 4: Test setup for verification of the proof-of-concept design

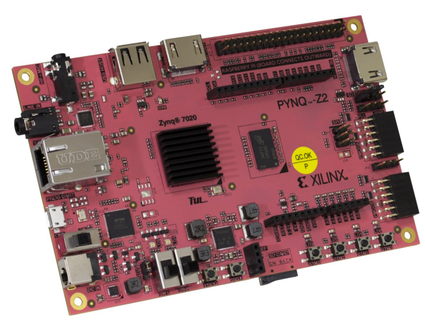

Figure 5: PYNQ-Z2 board, used for verification and performance measurements [1]

To measure the performance benefit of the POC design, the decryption speed and CPU resource usage was measured for both **software** decryption using the ***picoquic* stack with openSSL** and hardware decryption using the PoC design. The FPGA resource usage and timing constraints of the synthesized design are shown in Table 1. The **maximum raw network throughput** of the PoC design is **2.7 Gbit/s**.

Table 1: Resource utilisation and timing constraints after synthesis of the proof-of-concept on FPGA

Max. clock frequency	83.333 MHz
Delay (data in to data out)	756 ns
Slice lookup tables	24208
Slice registers	9687

Results

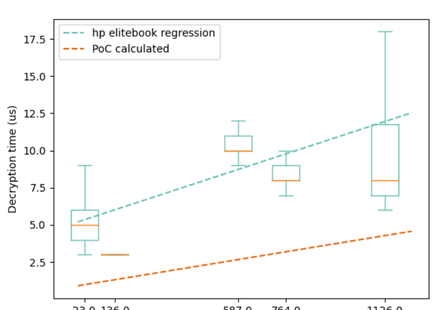

Figure 6: QUIC packet decryption duration for both hardware and software as a function of packet size

Figure 6 shows that the **time required** to decrypt a QUIC packet is **2 times faster** in hardware compared to software decryption with the picoquic stack. Both methods show that this duration increases **linearly** with increasing packet size. Note that with the hardware/software co-design, the software decryption is **bypassed** and the processor can use those resources for other tasks.

Conclusion

To conclude, a hardware/software co-design for offloading decryption of 1-RTT QUIC packets was made and verified. **Initial performance tests** showed that the decryption is **2 times faster** with this design than in a software-only setup on consumer grade hardware. The hardware decryption also frees up processor time and resources which are otherwise used for decryption in software. The maximum raw network throughput of the hardware is 2.7 Gbit/s. In future work, the PoC design should be **fully integrated** with a QUIC software stack to perform more thorough measurements such as maximum QUIC data throughput, network latency and other parameters while including overheads such as data copy. Finally, the functionality should be expanded for a real-world system.

Supervisors / Cosupervisors: Dr. Robin Marx
Dr. Ing. Jo Vliegen

[1] "Products - PYNQ-Z2" *Tulembedded.com*, 2022. https://www.tulembedded.com/FPGA/ProductsPYNQ-Z2.html (accessed Jul. 15, 2022).

UHASSELT

KU LEUVEN