Masterproef industriële ingenieurs wetenschappen 2021-2022

Improving QUIC's performance with hardware offload

Lowie Deferme
Master IW electronica-ict

Situering

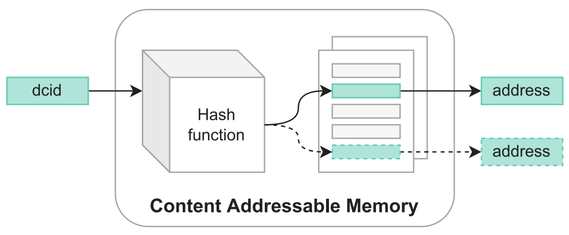
Iedere internetgebruiker maakt veelvuldig gebruik van het *Hypertext Transfer Protocol* (HTTP). De overgang naar de nieuwste versie van HTTP (HTTP/3) brengt de overgang van *Transmission Control Protocol* (TCP) naar QUIC op *User Datagram Protocol* (UDP) als onderliggend transportprotocol met zich mee.

Het relatief nieuwe QUIC belooft het maken van connecties sneller te laten verlopen en biedt meer encryptie. Deze verbeterde encryptie kost echter veel tijd op de processor. Dit heeft tot gevolg dat de ***throughput* per connectie snel daalt** bij meerdere concurrente connecties. Daardoor moeten services sneller opschalen en meer servers/hardware gebruiken als ze grote hoeveelheden clients willen bedienen over QUIC.

Daarom tracht deze masterproef om een deel van de processorintensieve berekeningen uit te voeren in ***dedicated hardware***. Op die manier kunnen de server *resources* beter benut worden.

Proefopstelling

In de proefopstelling is ervoor gekozen om **enkel binnenkomende pakketten te behandelen**, zo kan het systeem zelf definiëren wat de DCID van de pakketten is en dus ook wat de lengte van de DCID's is. Op deze manier moet de *DCID-Lookup*-module slechts werken voor **DCID's met een vaste, zelf gekozen, lengte**. Verder is, om praktische redenen, de *DCID-Lookup* uit het ideale ontwerp uitgevoerd als twee aparte modules: *DCID-Lookup* en *Key-memory*. Ook de *Crypto*-module is opgesplitst weergegeven in Figuur 3.

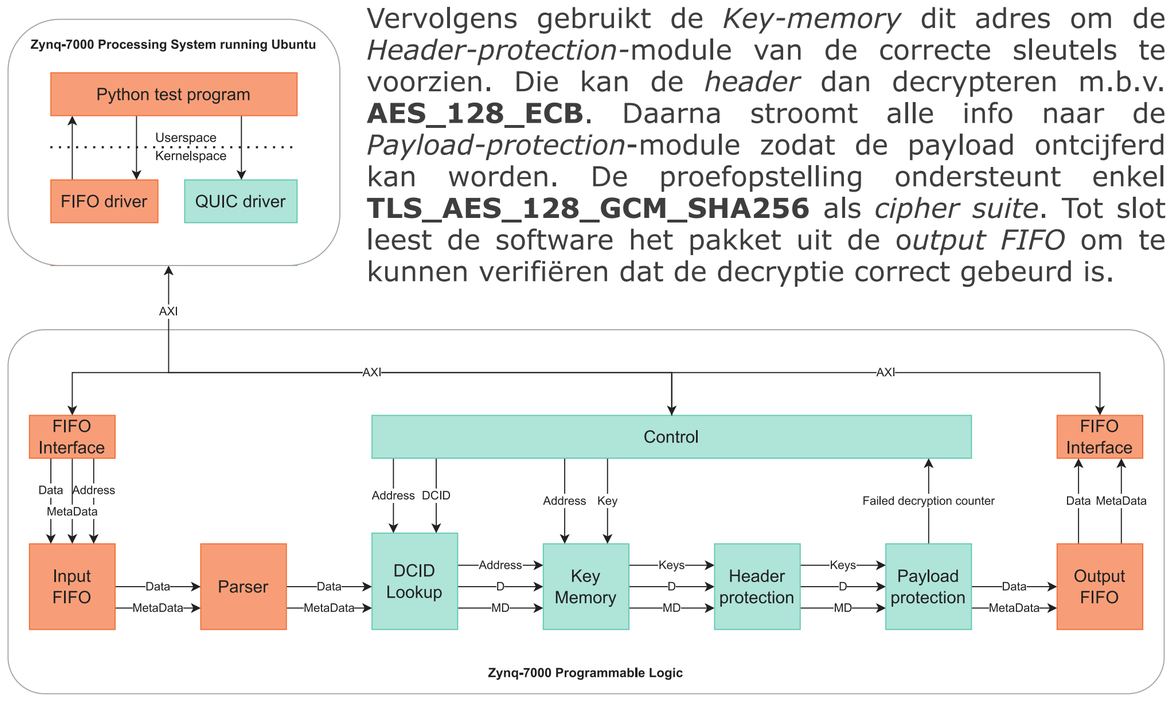


Figuur 2: Principe van de CAM

Data wordt elke stijgende klokflank per 32-bits parallel doorgegeven. Aangezien er zich nergens in de *accelerator* buffers bevinden is de ***throughput* enkel afhankelijk van de kloksnelheid**.

Een pakket is voorbereid in software en wordt via een AXI-stream bus naar een *input-FIFO* gekopieerd. Daarna markeert de *parser* delen van de data als IP, UDP, etc. Dankzij deze markering kan de *DCID-Lookup*-module de DCID uit de data filteren en het **adres van de sleutels opzoeken.** Figuur 2 illustreert dit: de DCID wordt door een **hash-functie** gevoerd die aangeeft waar het adres van de sleutels opgeslagen is in een lokaal geheugen.

Vervolgens gebruikt de *Key-memory* dit adres om de *Header-protection*-module van de correcte sleutels te voorzien. Die kan de *header* dan decrypteren m.b.v. **AES_128_ECB**. Daarna stroomt alle info naar de *Payload-protection*-module zodat de payload ontcijferd kan worden. De proefopstelling ondersteunt enkel **TLS_AES_128_GCM_SHA256** als *cipher suite*. Tot slot leest de software het pakket uit de o*utput FIFO* om te kunnen verifiëren dat de decryptie correct gebeurd is.

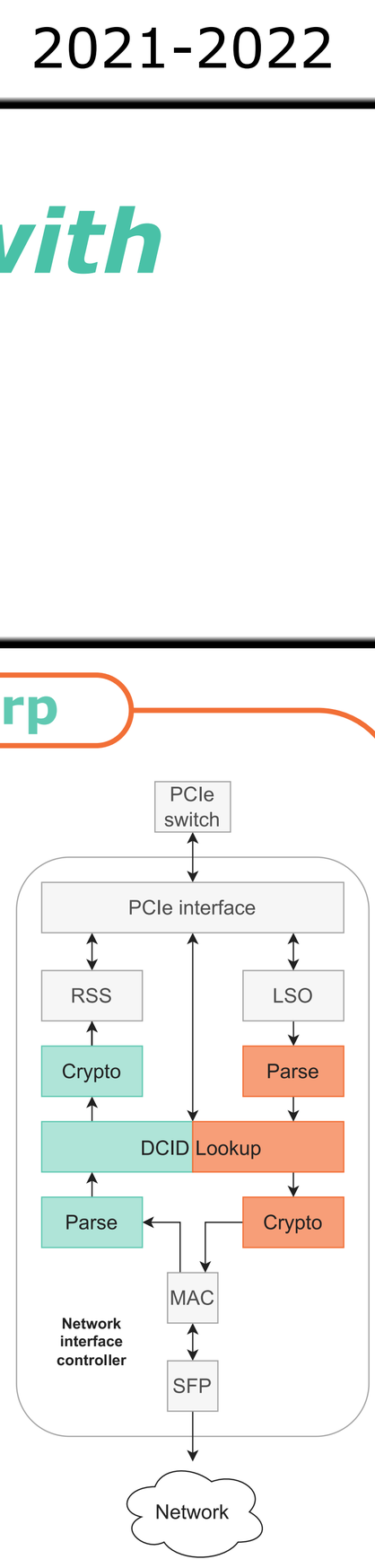


Figuur 3: Overzicht van het data- en metadatapad in de testopstelling

Ideaal ontwerp

Een moderne server communiceert met het netwerk met behulp van een *Network Interface Controller* (NIC). Een eenvoudige NIC interpreteert de signalen van het fysieke medium (*shielded twisted pair cable*, glasvezel, …) zodat de processor met de data aan de slag kan. Complexere NIC's proberen het werk voor de processor te verlichten door extra functies aan te bieden: twee voorbeelden hiervan zijn *Receive Side Scaling* (RSS) en *Large Send Offload* (LSO). Bij RSS probeert de NIC te achterhalen voor welke *processor core* de aangekomen data bedoeld is, zo kan de data efficiënter doorgegeven worden. In het geval van LSO krijgt de NIC een grote hoeveelheid data die het dan zelf opdeelt in pakketten alvorens te verzenden.

Aangezien de NIC het server-onderdeel is dat zich het dichtst bij het netwerk bevindt is het een goede plaats om een stuk van QUIC te *offloaden*. Figuur 1 geeft een NIC weer met een **QUIC-accelerator**. De accelerator maakt een onderscheid tussen inkomende en uitgaande pakketten.

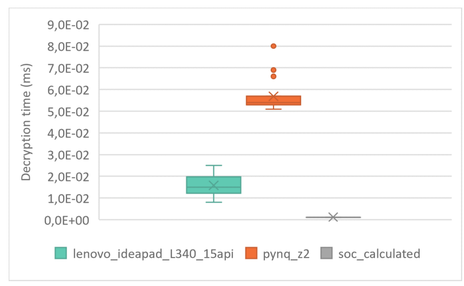


Figuur 1: Blockshema van de NIC in het ideale ontwerp

Inkomende pakketten doorlopen het linkse, blauwe, pad: eerst worden verschillende delen van het pakket gemarkeerd in de *Parse*-blok. Daarna controleert de *DCID-Lookup*-module of de *Destination Connection Identifier* (DCID) gekend is zodat eventuele *keys* doorgegeven kunnen worden aan de *Crypto*-module. Decryptie van een QUIC-pakket gebeurt in twee stappen: eerst moet de *header* ontcijferd worden omdat deze info bevat die nodig is om vervolgens de *payload* te decryptereren. Het oranje pad, voor **uitgaande pakketten**, is gelijkaardig aan het blauwe maar met een belangrijk verschil in de *Crypto*-module. Hierin moet nu eerst de *payload* versleuteld worden voordat de *header* geëncrypteerd kan worden.

Resultaten

De maximale klokfrequentie van de proefoptelling, bepaald voor een **Zync-7000 SoC** m.b.v. Xilinx Vivado, is **83,33 MHz**. Dit levert een *throughput* van **2,67 Gbit/s** op. De proefopstelling introduceert 63 klokcycli, oftewel **756 ns** *latency* tussen de *input* en *output* FIFO.



Figuur 4: Decryptie tijd van een 80 bytes lang 1RTT packet

Figuur 4 geeft de tijd weer die nodig is om één pakket te decrypteren op twee verschillende Linux-systemen m.b.v. software en in de proefosptelling m.b.v. hardware. In de proefopstelling is dit 1,16 µs, terwijl de softwaresystemen er gemiddeld 1,59 µs en 5,69 µs over doen. Het pakket was opgebouwd uit: 18 *bytes ethernet header*, 20 *bytes ip header*, 8 *bytes udp header* en 88 *bytes* QUIC waarvan 64 *bytes payload* waren.

Verder is deze **tijd** in de proefopstelling **lineair afhankelijk van de grootte** van de **payload** en is dit steeds kleiner dan de uitgevoerde softwaretesten met deze grootte.

Tot slot kan er dus besloten worden dat de proefopstelling **aantoont dat *hardware offload* van decryptie voor 1RTT-QUIC-pakketten een snelheidswinst oplevert**. Verder onderzoek is nodig om deze oplossing inzetbaar te maken *in the field*. Zo heeft de SoC in de proefopstelling geen integratie met een QUIC-*stack* en ondersteundt het geen variabele DCID-lengtes waardoor het enkel bruikbaar is voor uitgaande pakketten.

Promotoren dr. ing. Jo Vliegen, dr. Robin Marx