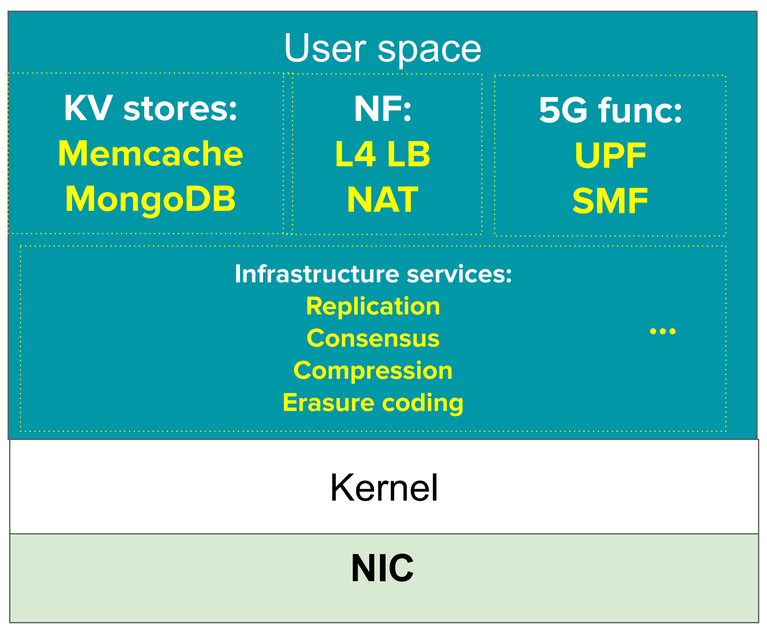
Microservice-based in-network security framework for FPGA NICs
(Poster 10)

Lasani Hussain, Mayank Rawat, Neeraj Kumar Yadav, Sumit Darak, Praveen Tammana* and Rinku Shah
IIIT Delhi, IIT Hyderabad*

The Problem

Traditional server in a datacenter network

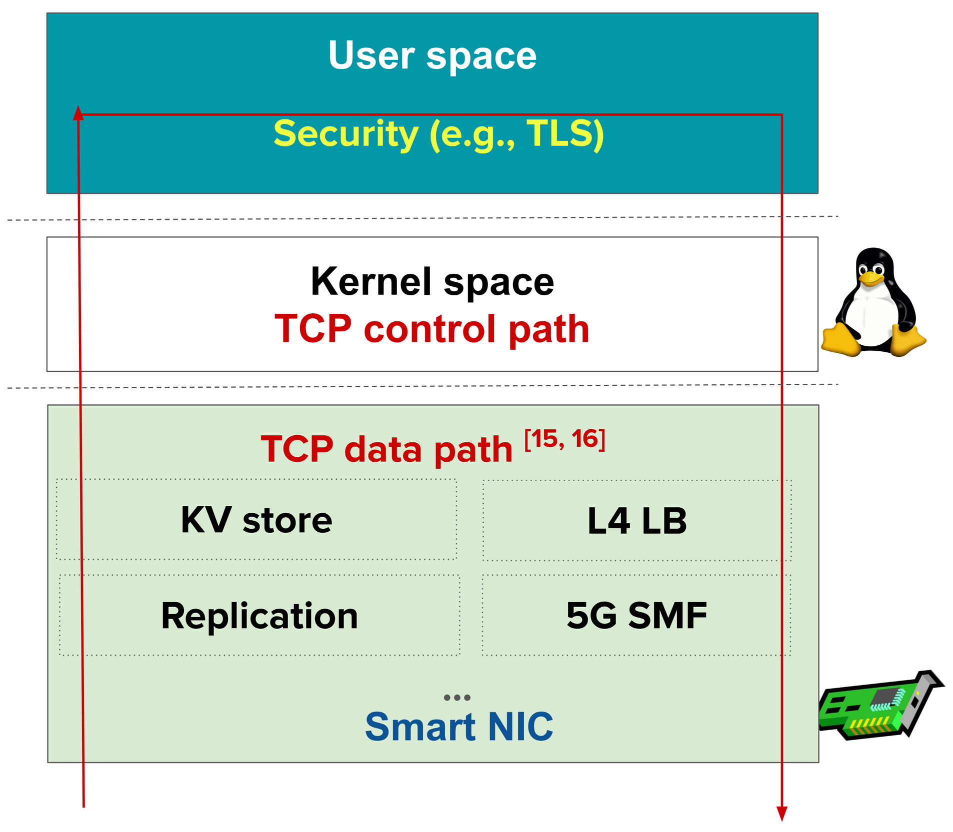


Application SLOs [1,2,3]
- Throughput: ~billions of req/sec
- Latency: < 100's of milliseconds

Per-server traffic demand [4]
- few 100s of Gbps

CPU cycles spent for network stack [5]
- ~ 24 CPUs for 400 Gbps

Server accelerates datacenter applications by leveraging smart NICs



Datacenter offload apps to smart NICs
- Satisfies application throughput/latency SLOs

Offloaded application requires crypto processing?
- Latency overheads due to traversal to host's user-space library

Need for in-network crypto processing!

Crypto algorithms used by datacenter and telecom network applications

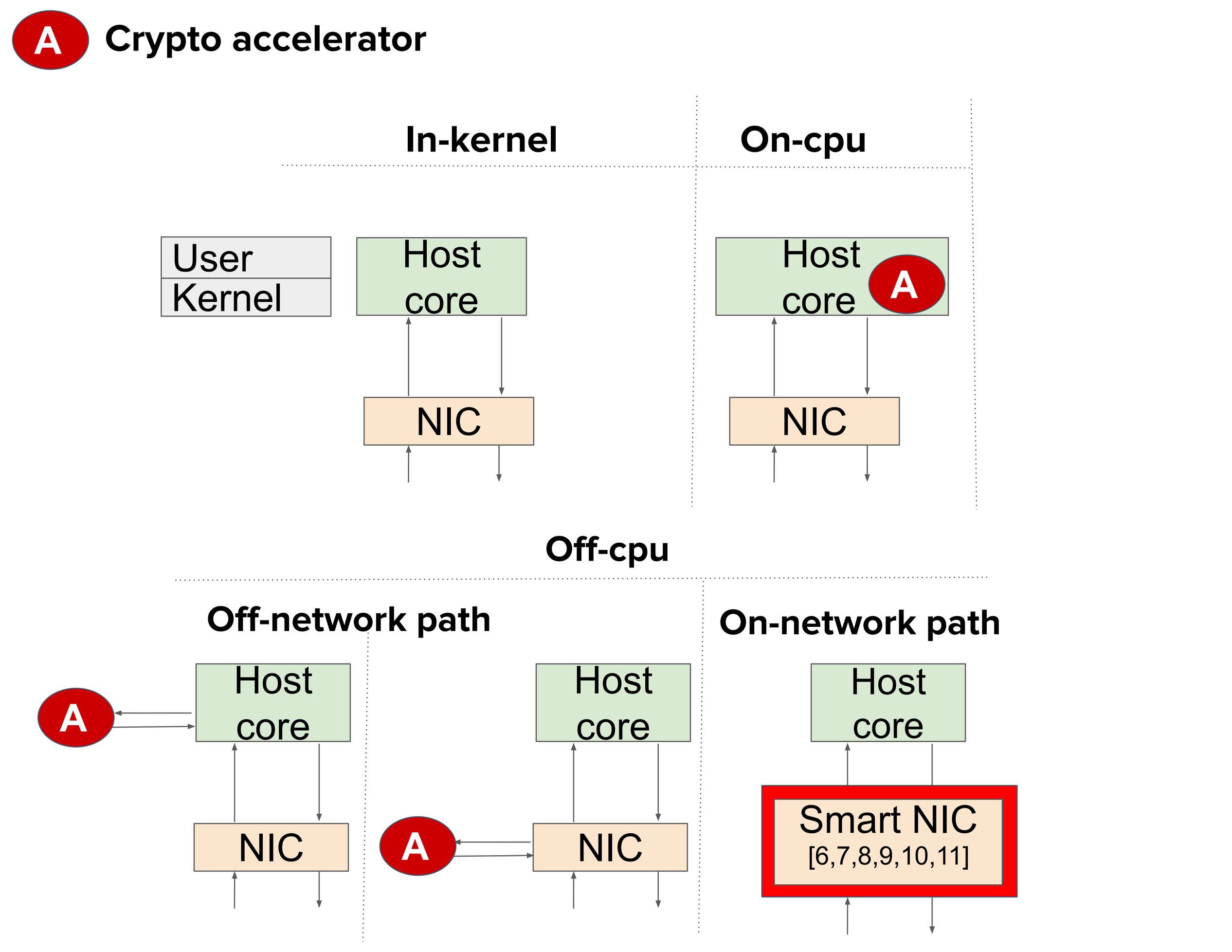
TLS cipher suites
TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA

5G cipher suites	
128-NEA1	Snow 3G
128-NEA2	AES-CTR
128-NEA3	ZUC
128-NIA1	Snow 3G
128-NIA2	AES-CMAC
128-NIA3	ZUC

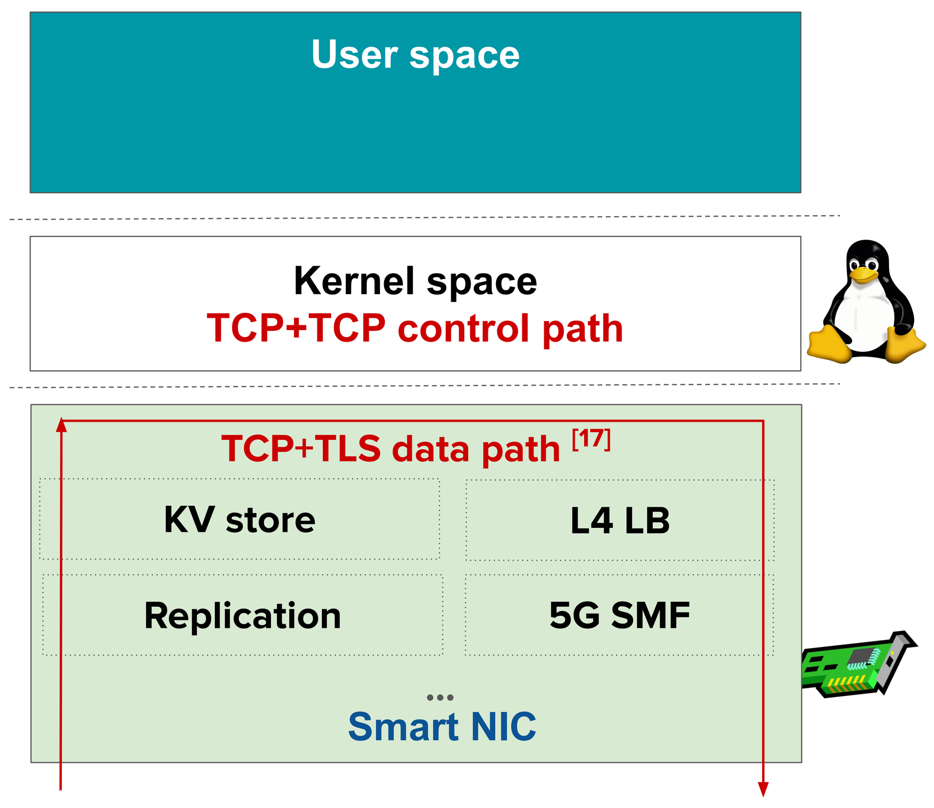
...

Host CPU	**Handshake Path** • Algorithm, Keys, IV, AAD, nonce, ...
Security accelerator	**Data-path** • Encryption, Decryption,Authentication

Crypto Accelerator Classification



State-of-the-art solutions



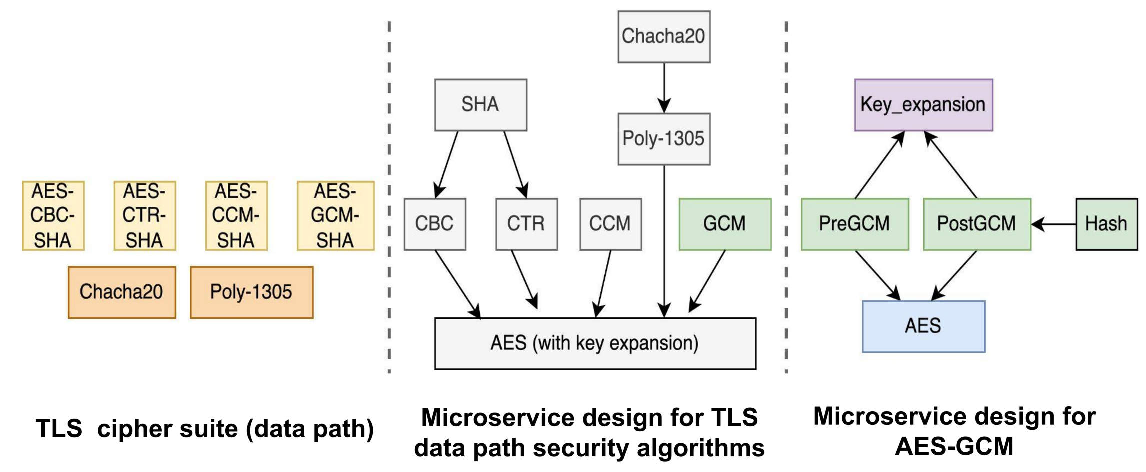
Fixed Function ASIC solutions NOT Flexible	Reconfigurable solutions Flexible
• Nvidia's Bluefield [12] • Pensando's DSC [13] • Netronome CX NICs [14]	• FPGA-based solutions [6, 7, 8, 10] • Chacha on programmable NIC [9] • AES on programmable Tofino switch [11]

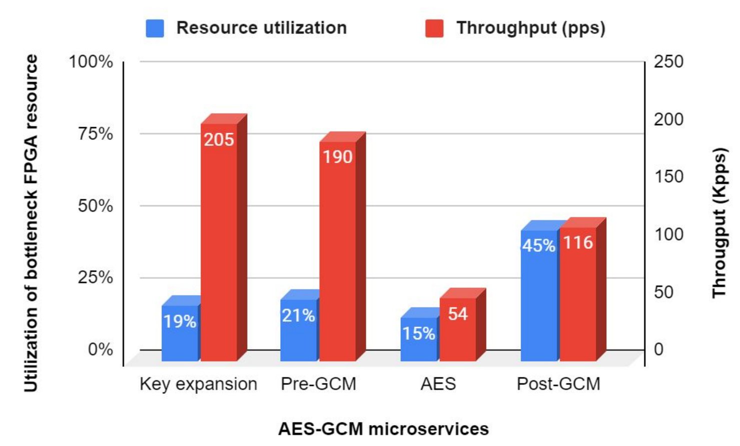
GAPS
- **CPUs** are general-purpose, may lead to **performance bottlenecks**
- **ASICs** are **fast but fixed function**
- Programmable **smart NICs**
  - **Reconfigurable** but have **limited resources**
  - Focus on **individual algorithm**

The Problem Statement

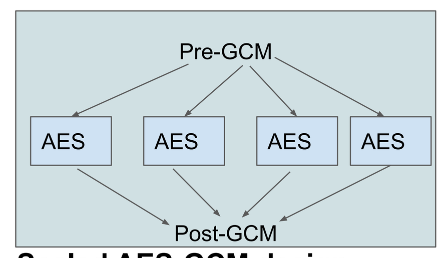
Design a **workload-aware**, **in-network crypto framework** that dynamically adapts based on workload parameters such as:
- Packet size
- Flow size
- Load per algorithms

Our Approach





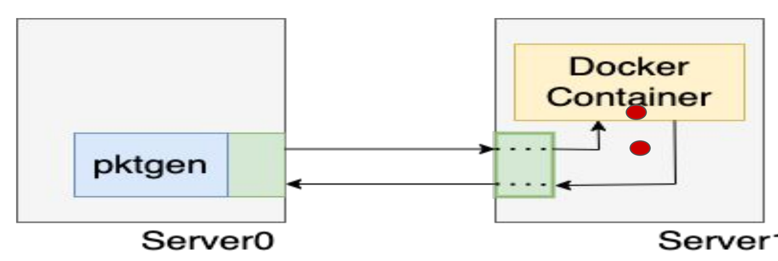
- **AES** is the **bottleneck** resource
  - Only scale AES microservice
- AES & key expansion are **reusable**



Scaled AES-GCM design

Experimental Setup

- **Baseline: Container setup**

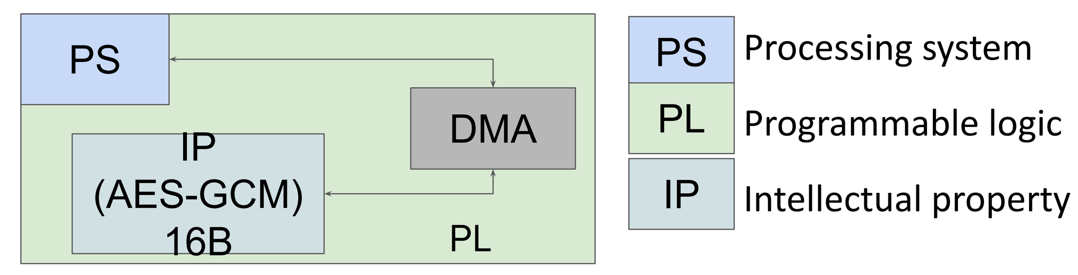


- AES-NI enabled (on-CPU acceleration)
- AES-NI disabled

AMD Ryzen 9 5950X (3.4 GHz, 16 cores, 32 threads) processor and 32GB RAM

Netronome Agilio CX 40 Gbit/s dual-port SmartNIC

- **FPGA setup**

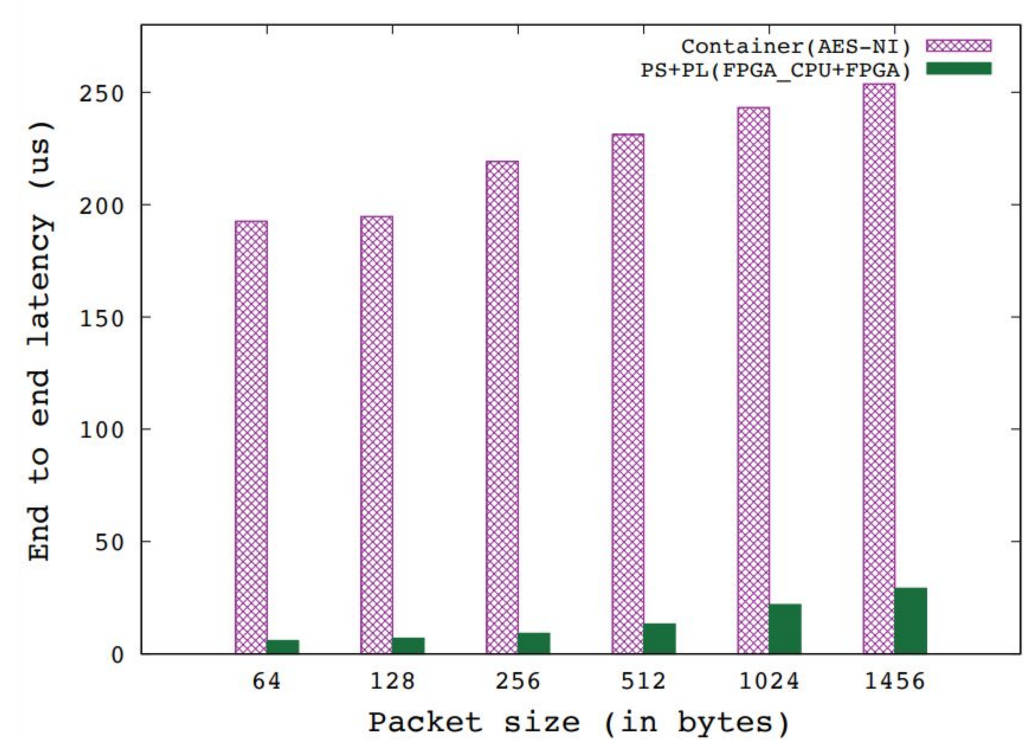


- Zynq UltraScale+ MPSoC ZCU106
  - Quad-core ARM cortex A53 processing system (PS)
  - 504K system logical cell
  - 38Mb distributed PL memory
- **~ 1K lines of hardware code**
  - HLS source for AES-GCM
  - HLS testbench for verification
  - Driver code for PS-PL communication on SDK

Ongoing Work

- **Design scalable FPGA hardware**
  - Variable-sized AES-GCM hardware
  - Scale bottleneck crypto function (e.g., AES)
  - Eliminate optional functions from the microservice chain (e.g., key expansion)
  - Port the design to Ethernet-based FPGA
- Design **monitoring** and **adaptation framework** for workload-aware dynamic scaling
- Design **crypto primitive APIs** that can be leveraged by application programmers for acceleration

Initial Results



Message size (in bytes)	Latency (in μs) Container (AES-NI) Compute	Container (AES-NI) End-to-end	FPGA End-to-end
64	0.65	192.57	5.89
128	0.67	194.65	6.94
256	0.69	219.18	9.07
512	0.75	231.17	13.29
1024	0.83	243.21	21.87
1464	0.93	253.73	29.16

- **AES-GCM processing using host's AES-NI accelerator**
  - Crypto processing is fast
  - Network stack and container engine overheads
- **AES-GCM offload to FPGA hardware**
  - Latency reduction: 88.5% to 97.3%

References

[1] Be Fast, Cheap and in Control with SwitchKV, NSDI'16
[2] NetCache: Balancing Key-Value Stores with Fast In-Network Caching, SOSP'17
[3] https://www.ngmn.org/wp-content/uploads/NGMN_5G_White_Paper_V1_0.pdf
[4] https://www.theregister.com/2022/03/06/400gbps-switching-demand/
[5] Understanding Host Network Stack Overheads, SIGCOMM 2021
[6] FlexDriver: A Network Driver for Your Accelerator, ASPLOS 2022
[7] Lightweight Cryptography for FPGAs https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5382056
[8] FPGA-based Cryptography for Internet Security. http://vhdl.pbworks.com/f/dandalisOSEE00.pdf
[9] Implementing ChaCha Based Crypto Primitives on Programmable SmartNICs, SIGCOMM FFSPIN 2022
[10] AES on FPGA from the fastest to the smallest. https://iacr.org/archive/ches2005/031.pdf
[11] Implementing AES Encryption on Programmable Switches via Scrambled Lookup Tables, SIGCOMM SPIN 2020
[12] https://www.nvidia.com/en-us/networking/products/data-processing-unit/
[13] https://www.amd.com/en/accelerators/pensando
[14] https://www.netronome.com/products/agilio-cx/
[16] FlexTOE: Flexible TCP Offload with Fine-Grained Parallelism, NSDI 2022
[17] AccelTCP: Accelerating Network Applications with Stateful TCP Offloading, NSDI 2020
[18] Autonomous NIC Offloads, ASPLOS 2021