IEEE BlackSeaCom™

UNIVERSITY OF South Carolina
Molinaroli College of Engineering and Computing

Real-Time Detection of Encrypted DGA and DNS Tunneling Using DPDK on SmartNICs

Sergio Elizalde, Ali AlSabeh, Samia Choueiri, Jaime Galán-Jiménez, Elie Kfoury, Jorge Crichigno

University of South Carolina
https://research.cec.sc.edu/cyberinfra/

IEEE International Black Sea Conference on Communications and Networking
June 11, 2026
Bucharest, Romania

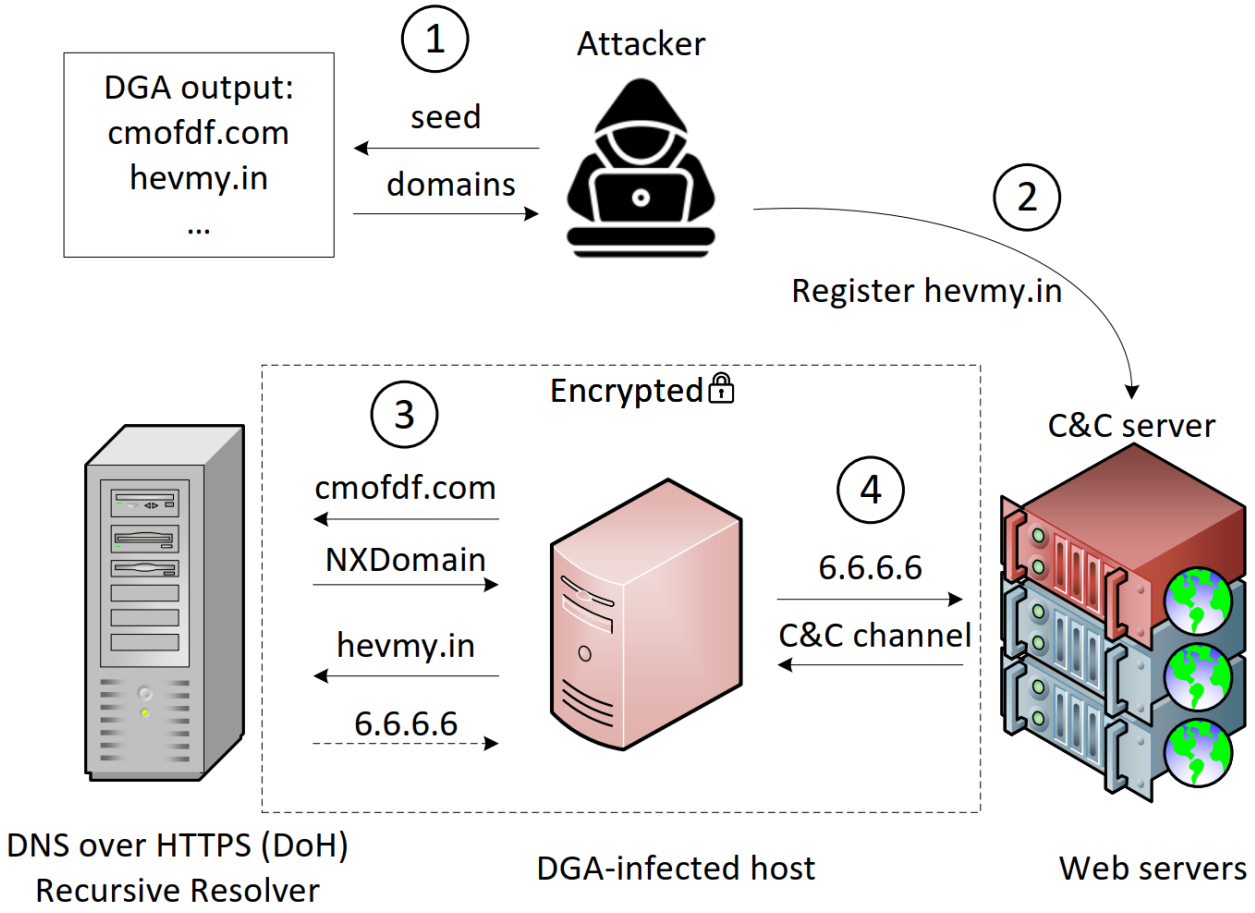
Motivation

- DNS over HTTPS (DoH) improves privacy by encrypting DNS queries
- The same encryption hides malicious DNS activity from defenders
- Attackers can use DoH for command-and-control and data exfiltration
- ML can classify encrypted traffic using observable flow metadata
- Real-time deployment requires packet processing close to the network interface

Goal: detect encrypted DNS threats without decrypting packets

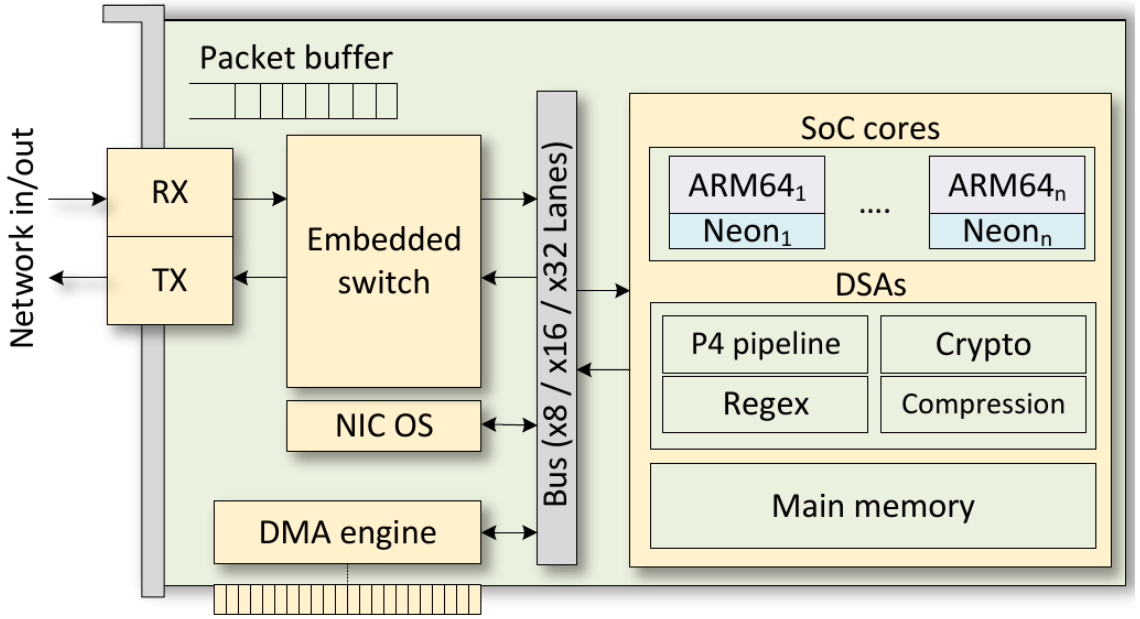
Encrypted DNS threats

- We considered DNS tunneling and DGA-based malware



SmartNICs

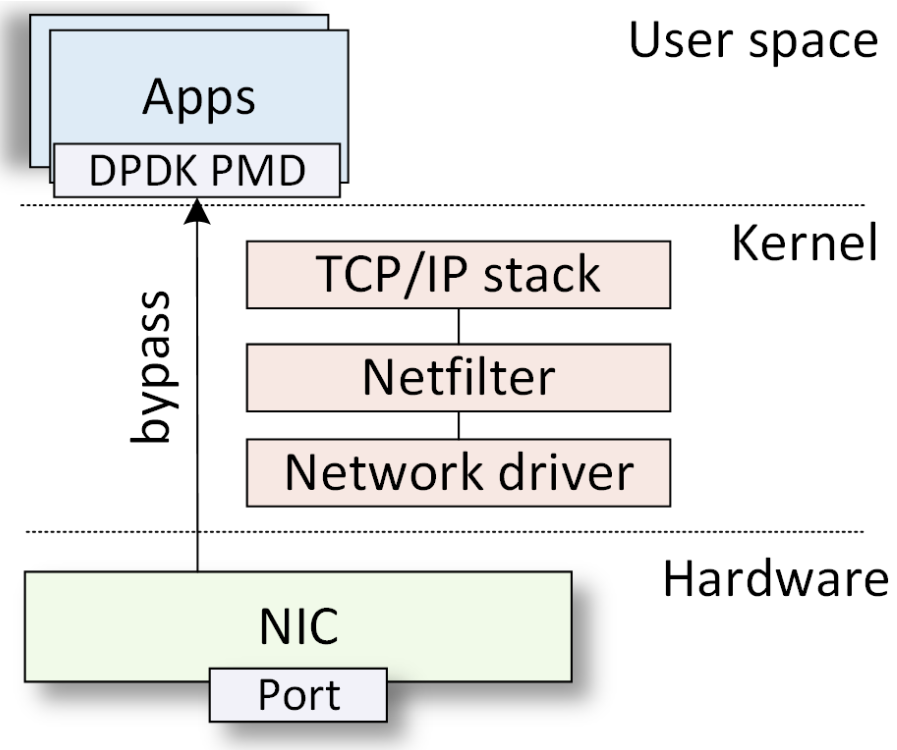
- SmartNICs include domain-specific accelerators (DSAs) and CPU cores
- Offload infrastructure workload from the host CPU
- Provide isolation when monitored host may be compromised



SmartNIC architecture

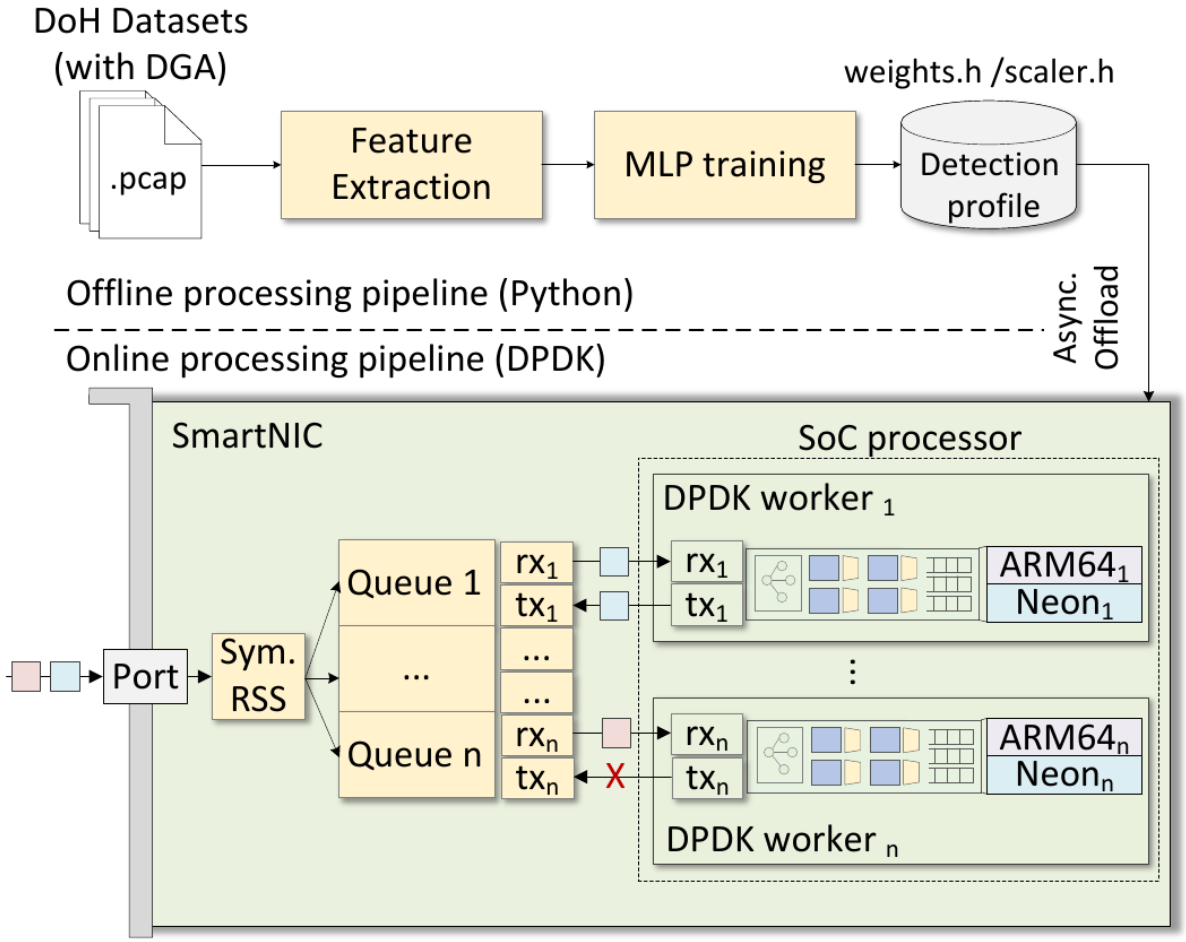
Data Plane Development Kit (DPDK)

- Data Plane Development Kit (DPDK) is a software-based acceleration technique
- DPDK applications can run directly on the computing units of Smart Network Interface Cards (SmartNICs)



Proposed system

- **Offline stage**
  - Extract flow-level features from DoH and Non-DoH traces
  - Train the multiclass MLP and export weights/scalers as header files
- **Online stage**
  - DPDK workers process packets on ARM64 cores
  - Symmetric RSS keeps each bidirectional flow on the same core
  - MLP inference runs after the first N packets of each flow



Feature Set

- Each flow is summarized using the first N packets (e.g., N=16)
- Size features: minimum, maximum, and average IPv4 packet size
- Context features: packet counts, byte counts, direction switches, packet fractions, and byte fractions
- Direction switches refer to the number of times the communication changes direction between Client and Server in a sequence of packets

Features		Stats	Scope		
			Client	Server	Global
Size	IPv4 Packet	Min	●	●	●
		Max	●	●	●
		Avg.	●	●	●
Context	Dir. switches	Count	○	○	●
	Packets	Count	●	●	○
	Packet fraction	Ratio	●	○	○
	Bytes	Count	●	●	○
	Bytes fraction	Ratio	●	○	○

● Selected ○ Not used

DPDK implementation

- Initialization:
  - DPDK ports, memory pools, workers, MLP model, and flow table
- Loop:
  - Poll packets in bursts
  - Inspect TCP flows on port 443
  - Use only TLS Application Data records for feature updates
  - Normalize features and classify once a flow reaches N packets
  - Forward traffic while logging per-flow predictions

Algorithm 1: Online DoH classification in DPDK

Input: Port ID, MLP weights, StandardScaler parameters
Output: Classified flows

```
1  Init: initialize DPDK, ports, memory pools, flow table.
2  Launch workers: start worker cores.
3  Main loop (per worker): while running do
4      pkts ← rx_burst(port);
5      foreach packet p in pkts do
6          if p is TCP and involves port 443 then
7              if TLS Application Data then
8                  update flow stats(flowID, pkt_len, direction);
9                  if flow has ≥ N packets and not yet
                     classified then
10                     feats ← extract_features(flow_entry);
11                     feats_norm ← normalize(features);
12                     pred ← MLP_inference(features_norm);
13                     log(flowID, features, prediction);
14                     mark flow finalized;
15         tx_burst(port, pkts);     // forward packet
```

DPDK implementation

- Initialization:
  - DPDK ports, memory pools, workers, MLP model, and flow table
- Loop:
  - Poll packets in bursts
  - Inspect TCP flows on port 443
  - Use only TLS Application Data records for feature updates
  - Normalize features and classify once a flow reaches N packets
  - Forward traffic while logging per-flow predictions

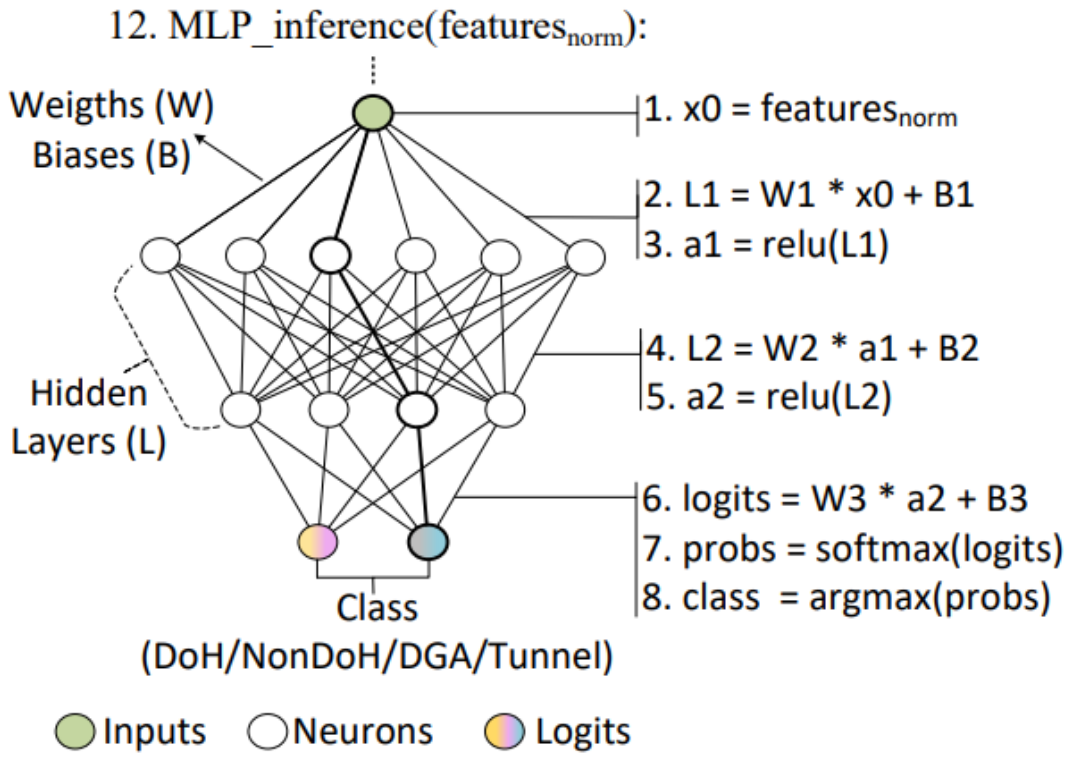
Algorithm 1: Online DoH classification in DPDK

Input: Port ID, MLP weights, StandardScaler parameters
Output: Classified flows

```
1  Init: initialize DPDK, ports, memory pools, flow table.
2  Launch workers: start worker cores.
3  Main loop (per worker): while running do
4      pkts ← rx_burst(port);
5      foreach packet p in pkts do
6          if p is TCP and involves port 443 then
7              if TLS Application Data then
8                  update flow stats(flowID, pkt_len, direction);
9                  if flow has ≥ N packets and not yet
                     classified then
10                     feats ← extract_features(flow_entry);
11                     feats_norm ← normalize(features);
12                     pred ← MLP_inference(features_norm);
13                     log(flowID, features, prediction);
14                     mark flow finalized;
15         tx_burst(port, pkts);     // forward packet
```

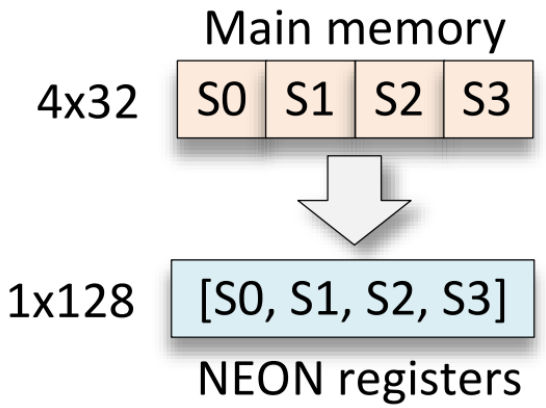
NEON Acceleration

- ARM's NEON is a parallel arithmetic accelerator
  - Single Instruction Multiple Data (SIMD) architecture
- NEON can process 4 32-bit words per instruction (i.e., 128bit registers)
- Multilayer Perceptron (MLP) can be implemented with matrix–vector operations
- Inference with 16 inputs, 4 outputs, and hidden layers whose sizes are multiples of 4



NEON Acceleration

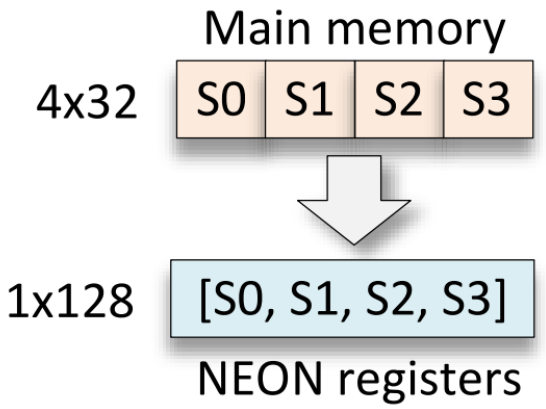
- NEON intrinsic operations:
  - load_operand(): `vld1q_f32()`

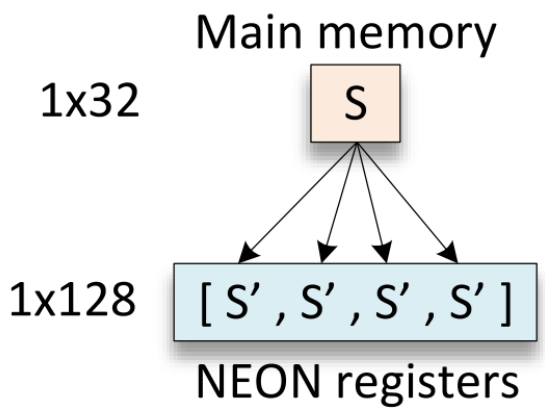


S Single word V Vector

NEON Acceleration

- NEON intrinsic operations:
  - load_operand(): `vld1q_f32()`
  - vector_duplicate(): `vdupq_n_f32()`



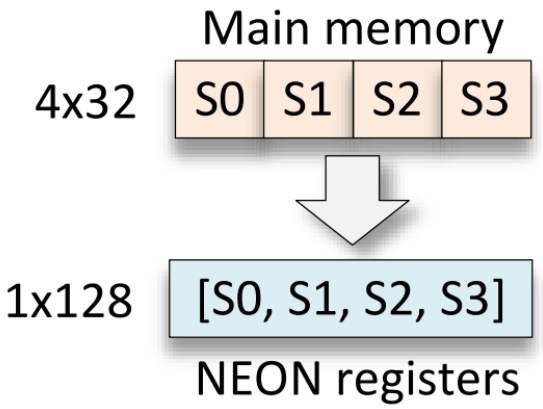


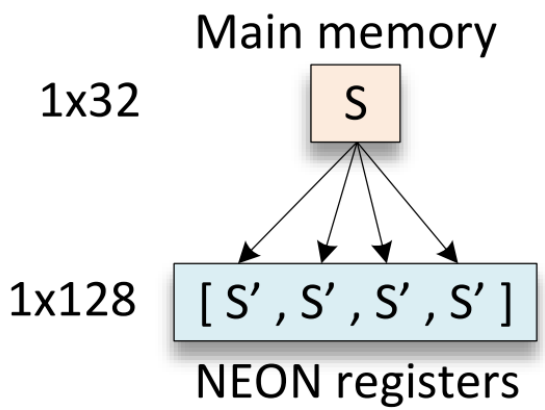
S Single word

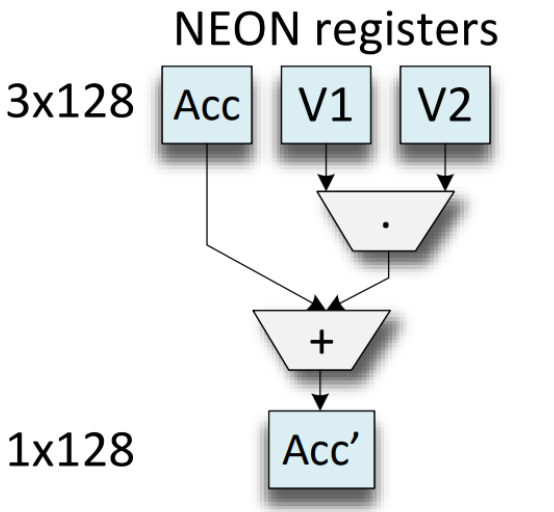
V Vector

NEON Acceleration

- NEON intrinsic operations:
  - load_operand(): `vld1q_f32()`
  - vector_duplicate(): `vdupq_n_f32()`
  - vector_multiply(): `vfmaq_f32()`







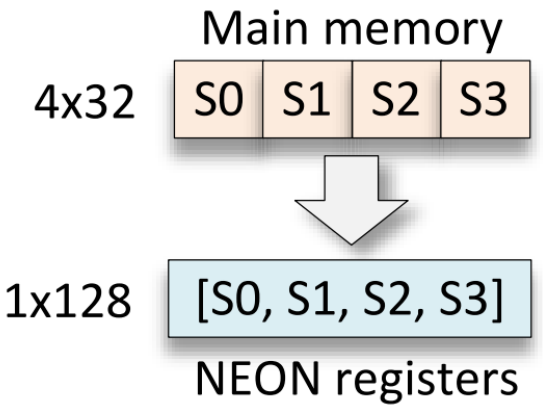
Acc' = Acc + (V1.V2)

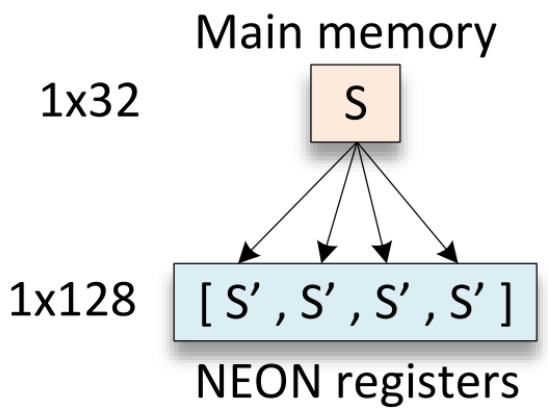
S Single word

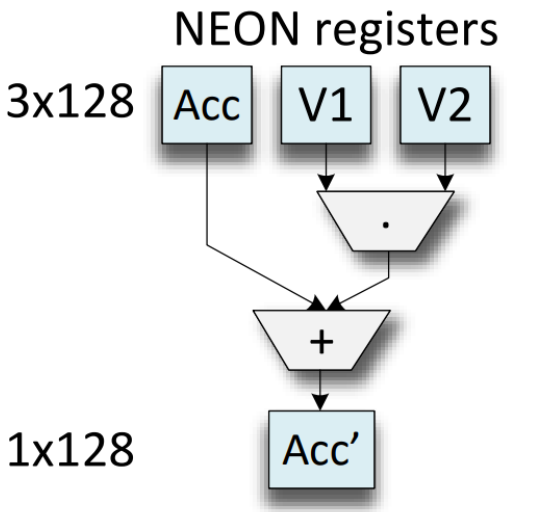
V Vector

NEON Acceleration

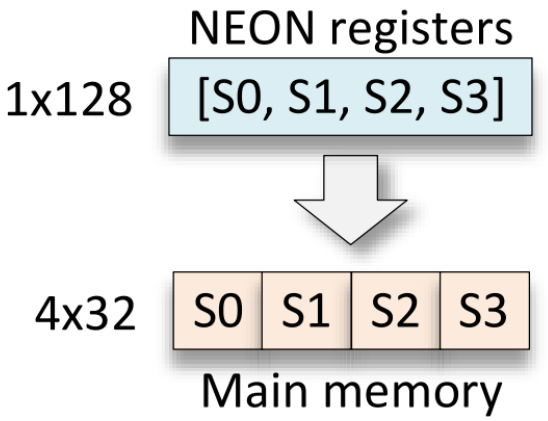
- NEON intrinsic operations:
  - load_operand(): `vld1q_f32()`
  - vector_duplicate(): `vdupq_n_f32()`
  - vector_multiply(): `vfmaq_f32()`
  - vector_store(): `vst1q_f32()`

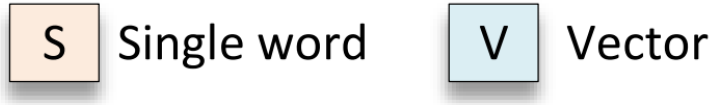






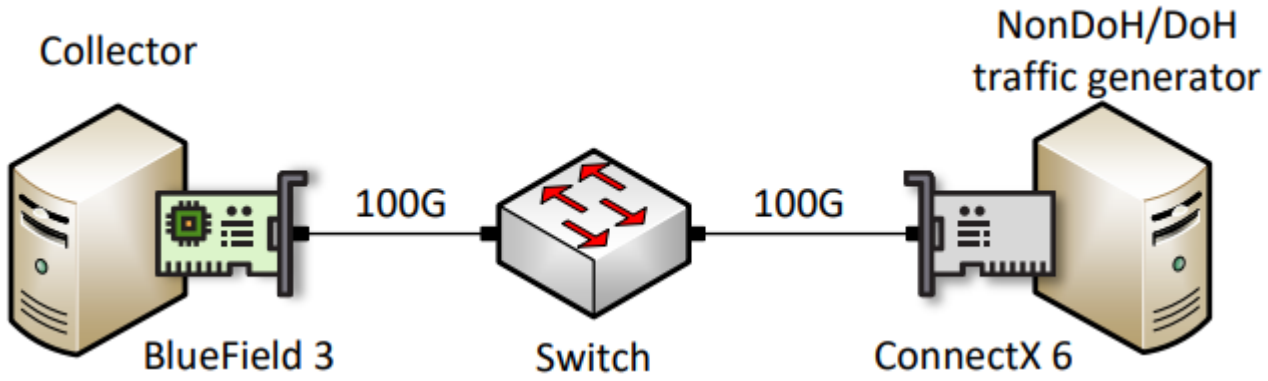
Acc' = Acc + (V1.V2)





Experimental Setup

- Datasets:
  - CIRA-CIC-DoHBrw-2020[1]
  - DoH-DGA-Malware-Traffic-HKD[2]
  - Augmented DoH DGA dataset: plaintext DNS PCAPs replayed as DoH

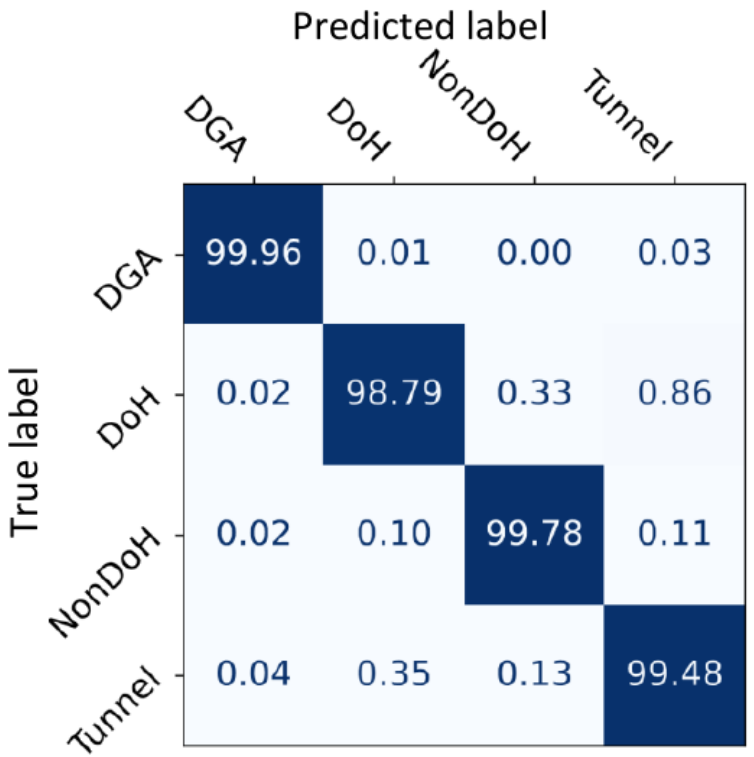


1. M. MontazeriShatoori, L. Davidson, G. Kaur, and A. H. Lashkari, “Detection of doh tunnels using time-series classification of encrypted traffic,” in IEEE DASC/PiCom/CBDCom/CyberSciTech Conference, 2020.
2. R. Mitsuhashi, Y. Jin, K. Iida, T. Shinagawa, and Y. Takai, “Detection of DGA-based malware communications from doh traffic using machine learning analysis,” in 2023 IEEE 20th Consumer Communications & Networking Conference (CCNC), 2023.

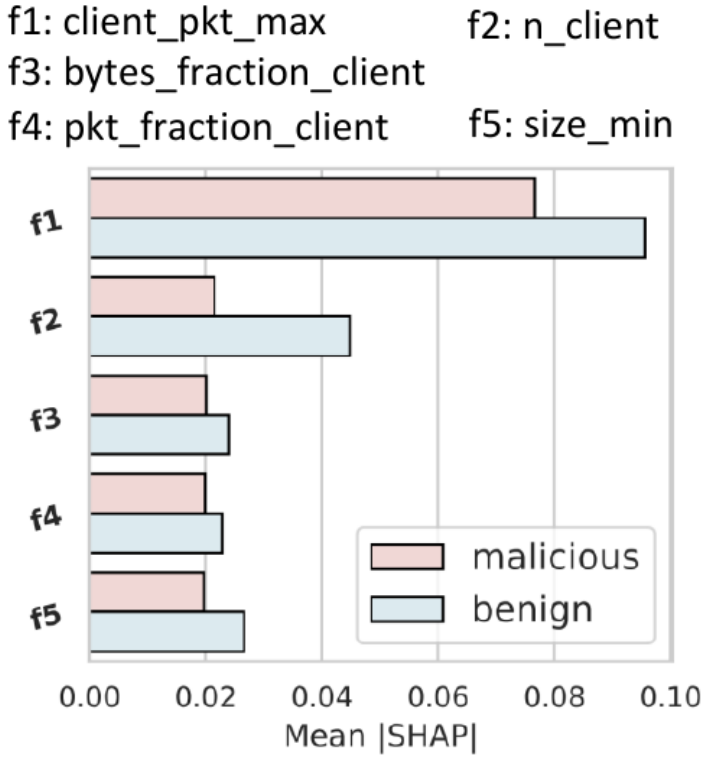
OFFLINE RESULTS

Result 1: Multiclass detection

- Four-class task: DGA, DoH, Non-DoH, Tunnel
- MLP with hidden layers of 64 and 32 neurons
- Most confusion occurs between DoH and Tunnel traffic



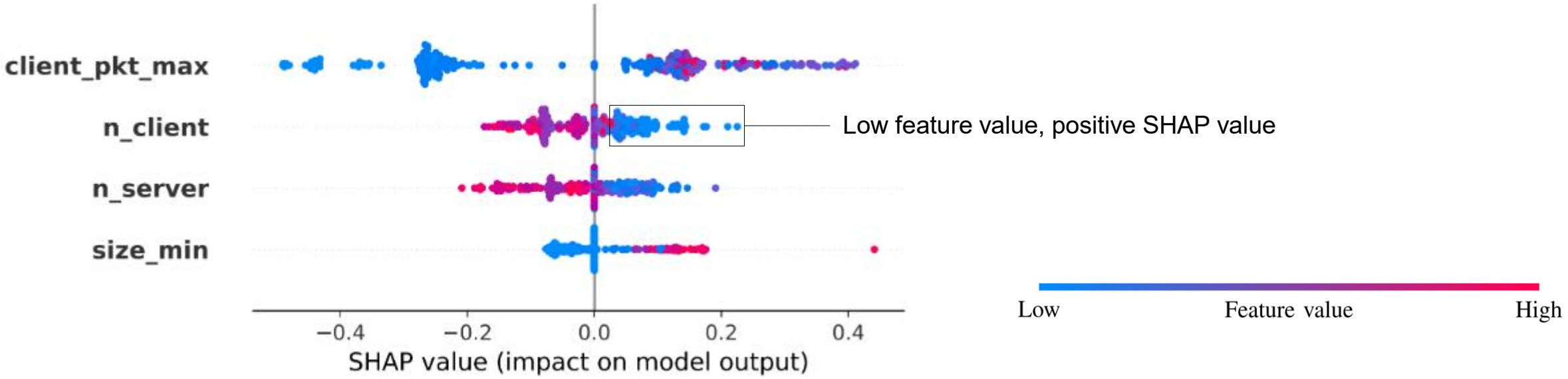
(a) Confusion matrix



(b) Feature importance

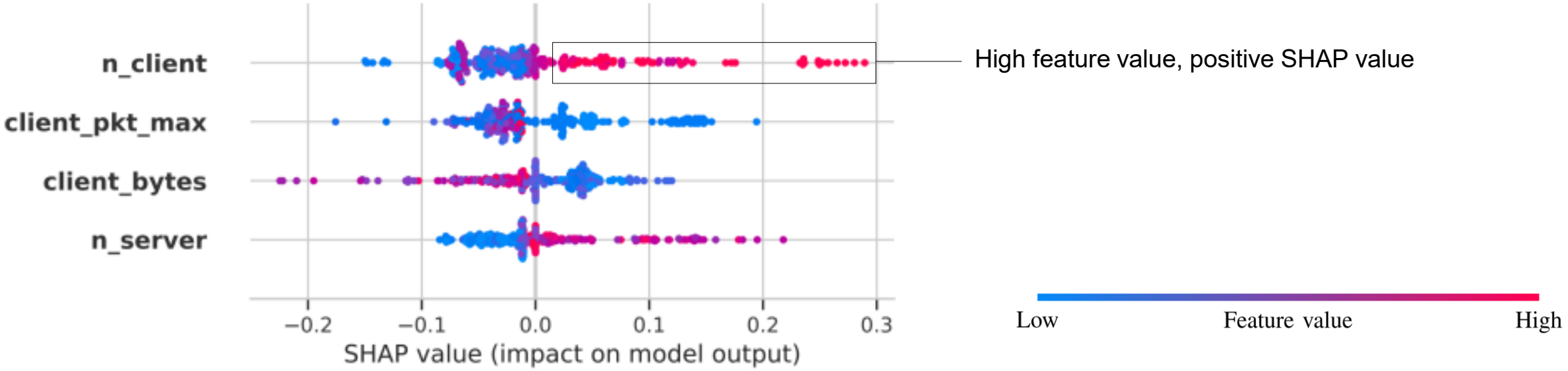
Result 2: Explainability via SHAP

- Benign case: NonDoH
- Features ordered by importance
- Negative SHAP values push the prediction away
- Positive SHAP values push the prediction towards the selected class



Result 2: Explainability via SHAP

- Malicious case: DGA
- Features ordered by importance
- Negative SHAP values push the prediction away
- Positive SHAP values push the prediction towards the selected class



ONLINE RESULTS

Result 4: Threat sensitivity

- Malicious PCAP files replayed using tcpreplay
- Recall measures how well we catch all the malicious flows

TABLE II: Sensitivity on DNS tunneling and DGAs.

Dataset	Resolver	Medium	Recall
CIC-DoHBrw tunnels	Adguard, Cloudflare Google, Quad9	Tool	98.73%
HKD DGA	Suspicious DoH (Private IP, NXDOMAIN)	Malware	100%
Own DGA	Adguard, Cloudflare Google, Quad9	Malware	99.81%

98.73%

CIC DoH tunnels

100%

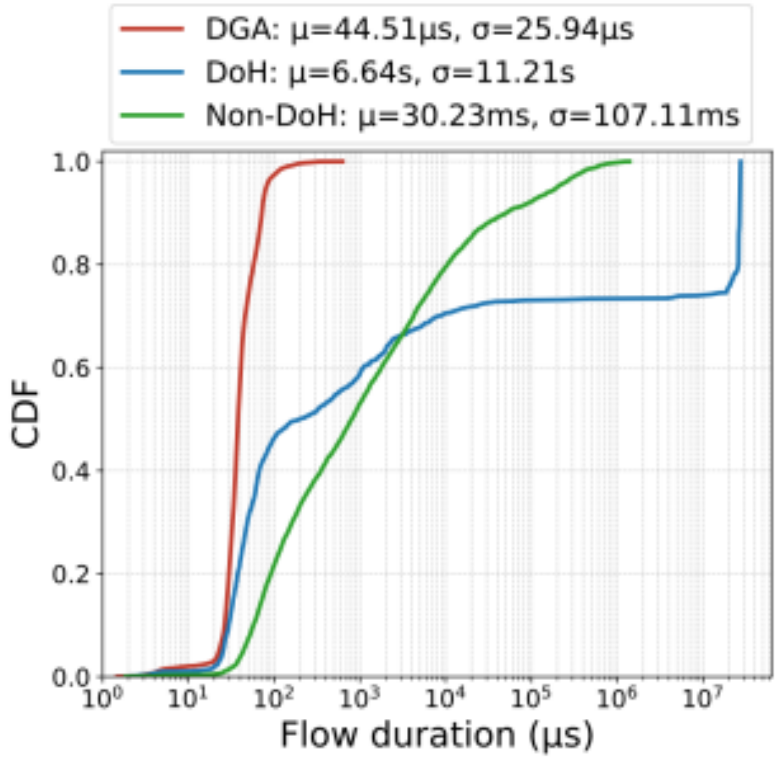
HKD DGA malware

99.81%

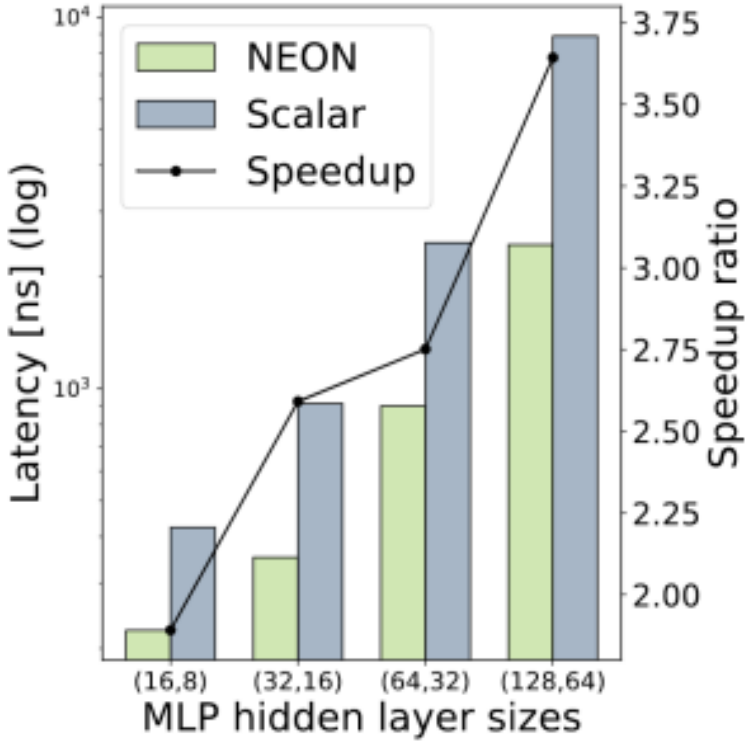
augmented DGA dataset

Result 5: ML latency

- (a) Feature collection latency is measured from packet 1 to packet 16 of each flow
- (b) NEON reduces inference latency compared with scalar processing



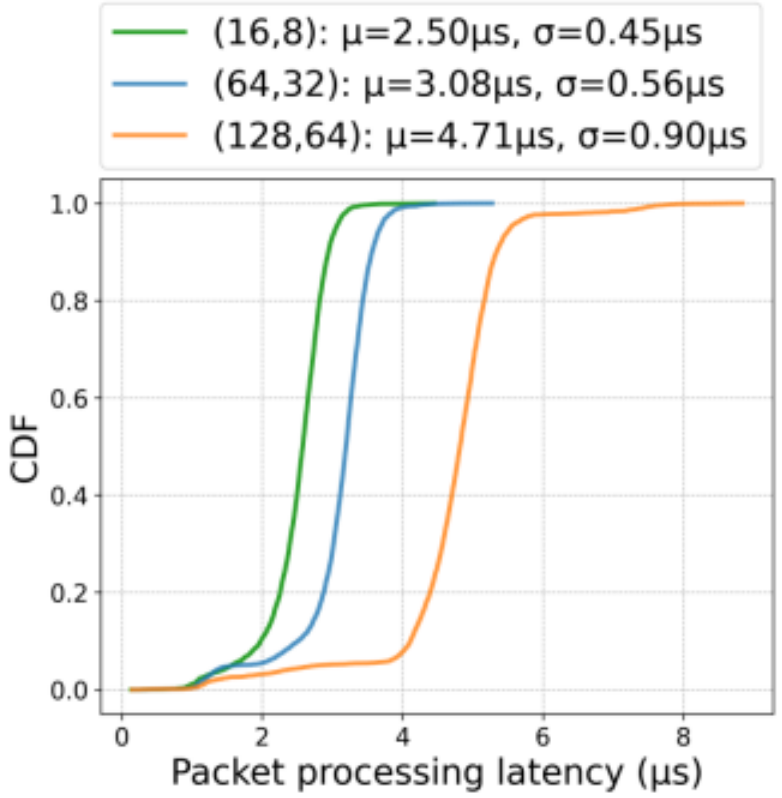
(a) Distribution for feature collection



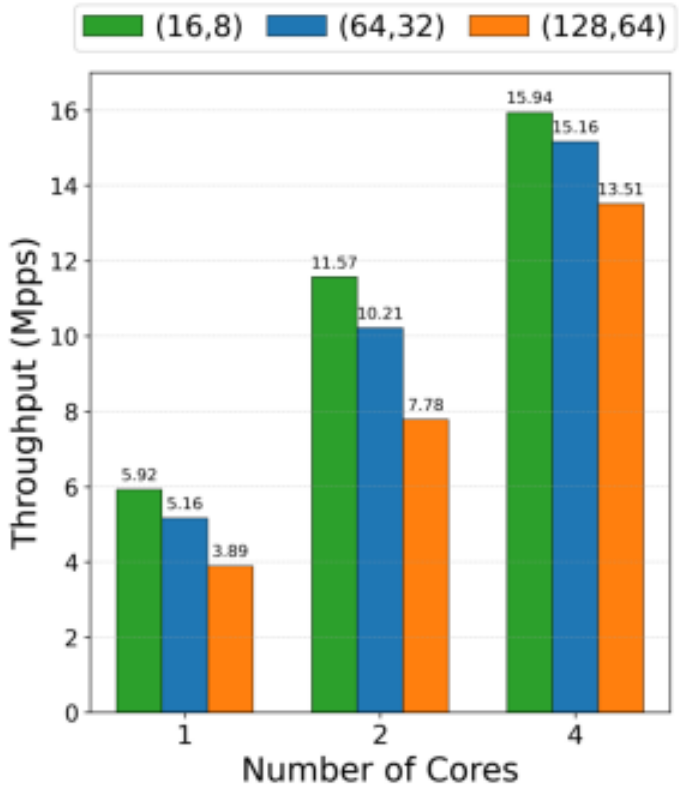
(b) Average inference latency

Result 6: End-to-End latency

- Larger MLPs increase per-core latency and heavier tails
- Throughput scales as additional ARM cores are used
- More cores reduce the impact of model size on throughput



(a) Per-core latency distribution



(b) Average throughput

16 Mpps

maximum throughput

2.5 us

average packet latecy

Conclusion

- Presented a SmartNIC-based framework for encrypted DNS threat detection
- Classifies DoH, Non-DoH, DGA-based DoH, and tunneling DoH traffic
- Achieves over 98% offline accuracy and over 98.5% online threat recall
- Maintains real-time performance with 1 us inference latency and up to 16 Mpps
- Future work includes:
  - Collaborative offloading across switches, SmartNICs, and hosts
  - Analysis of machine-learning model vulnerabilities

UNIVERSITY OF
South Carolina

Aiken
UNIVERSITY OF SOUTH CAROLINA

This work was supported by the U.S. National Science Foundation (NSF), under awards 2346726, 2417823, and 2403360

For additional information, please refer to
https://research.cec.sc.edu/cyberinfra

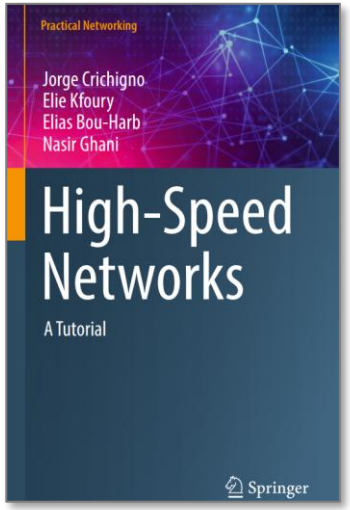
Email: elizalds@email.sc.edu

Cybertraining Material

Information about lab libraries is available at https://research.cec.sc.edu/cyberinfra/cybertraining

Programmable data plane devices

1. P4-DPDK Security
2. Introduction to P4-DPDK
3. Cybersecurity Apps with P4 Programmable Data Planes
4. P4 Programmable Data Planes: Applications, Stateful Elements, Custom Packet Processing
5. Intro to P4 Programmable Switches
6. Intro to P4 Programmable Switches with Intel's Tofino
7. P4 Monitoring Applications



Traditional Protocols

8. Software-defined Networking (SDN)
9. Open vSwitch
10. Network Tools and Protocols
11. Introduction to IPv6
12. Open Shortest Path First (OSPF)
13. Introduction to BGP
14. MPLS and Advanced BGP

Cybersecurity, Management, and Science DMZs

15. Zeek IDS/IPS
16. Cybersecurity Fundamentals
17. perfSONAR 5
18. P4-perfSONAR
19. Network Management Tools
20. High-speed Networks: A Tutorial