An Overview of P4 Programmable Switches and Applications

Jorge Crichigno
College of Engineering and Computing, University of South Carolina
jcrichigno@cec.sc.edu
https://research.cec.sc.edu/cyberinfra

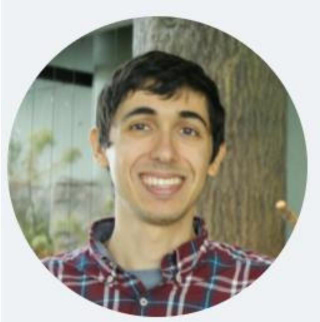
Electrical and Computer Engineering Seminar
Brigham Young University
Provo, Utah – April 11, 2024

Agenda

- Introduction to P4 Programmable Switches
- DGA Family Classification using DNS Deep Packet Inspection on P4 Switches
- Dynamic Router’s Buffer Sizing using P4 Switches
- Conclusion

Cyberinfrastructure Lab (CI) at USC

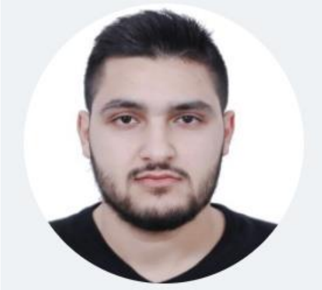
Elie Kfoury
Assistant Professor



Jose Gomez
PhD Student

Samia Choueiri
PhD Student

Ali AlSabeh
PhD Student



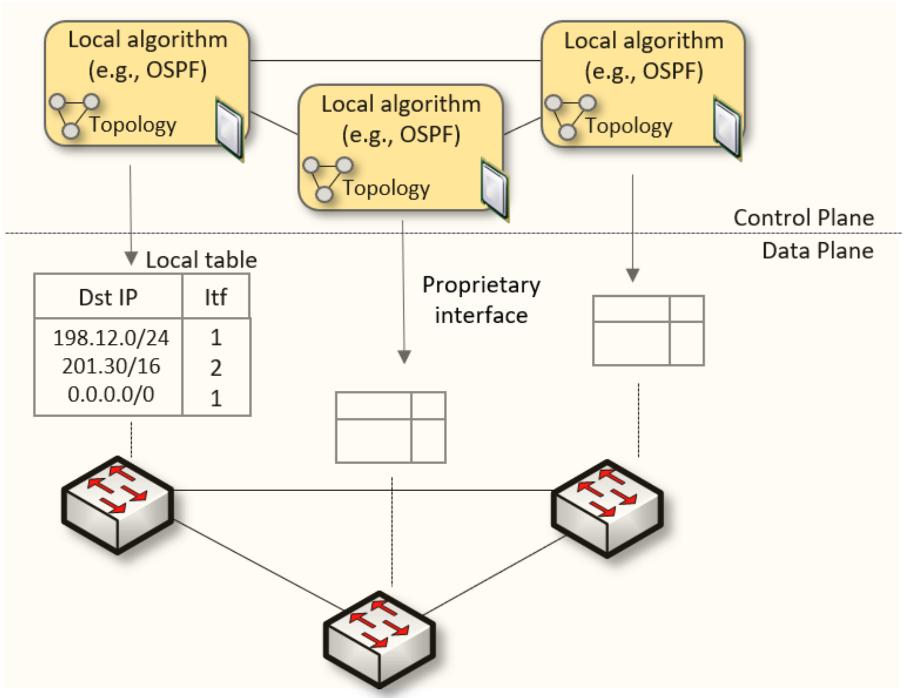
Ali Mazloum
PhD Student

Christian Vega
PhD Student

Introduction to P4 Programmable Switches

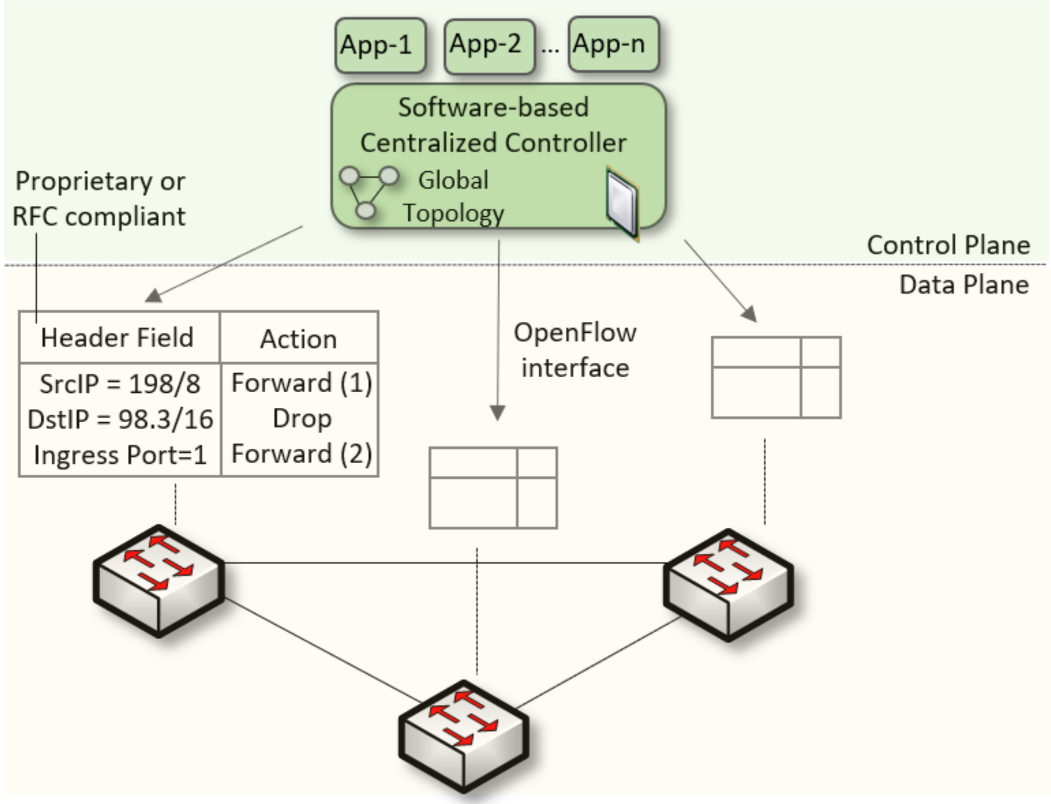
Traditional (Legacy) Networking

- Since the explosive growth of the Internet in the 1990s, the networking industry has been dominated by closed and proprietary hardware and software
- The interface between control and data planes has been historically proprietary
  - ➢ Vendor dependence: slow product cycles of vendor equipment, no innovation from **programmers**



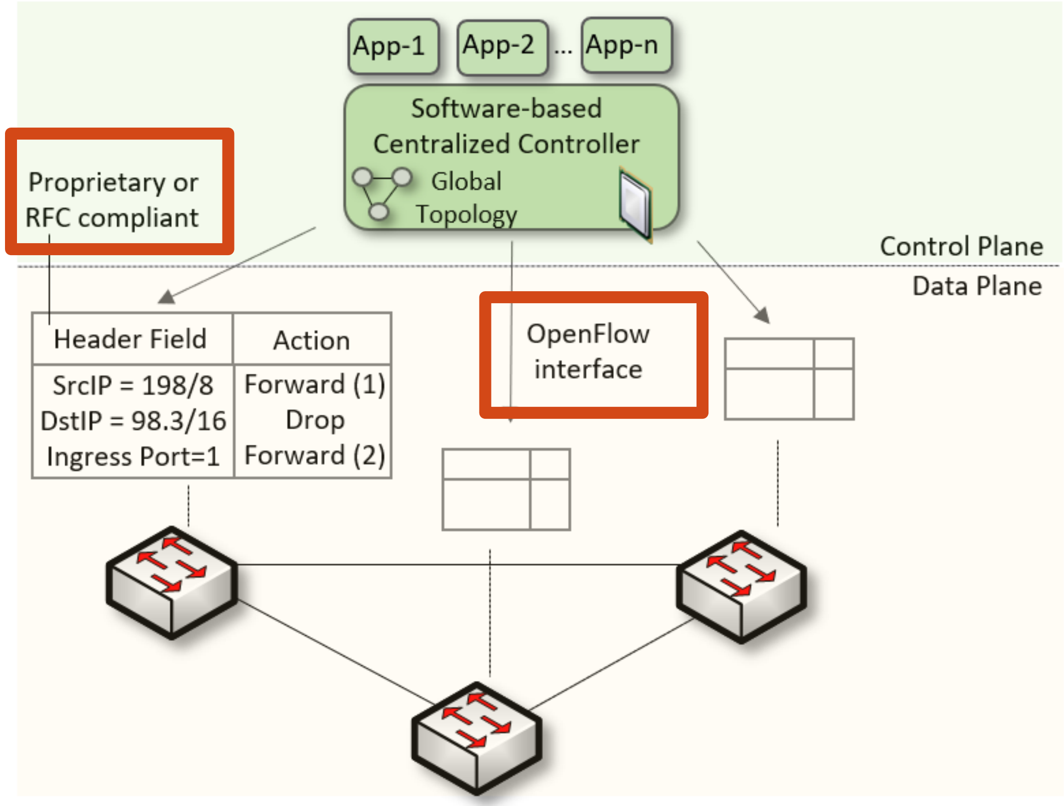
Software-Defined Networking (SDN)

- Protocol ossification has been challenged first by SDN
- SDN (1) explicitly separates the control and data planes, and (2) enables the control plane intelligence to be implemented as a software outside the switches by **end programmers**
- The function of populating the forwarding table is now performed by the controller



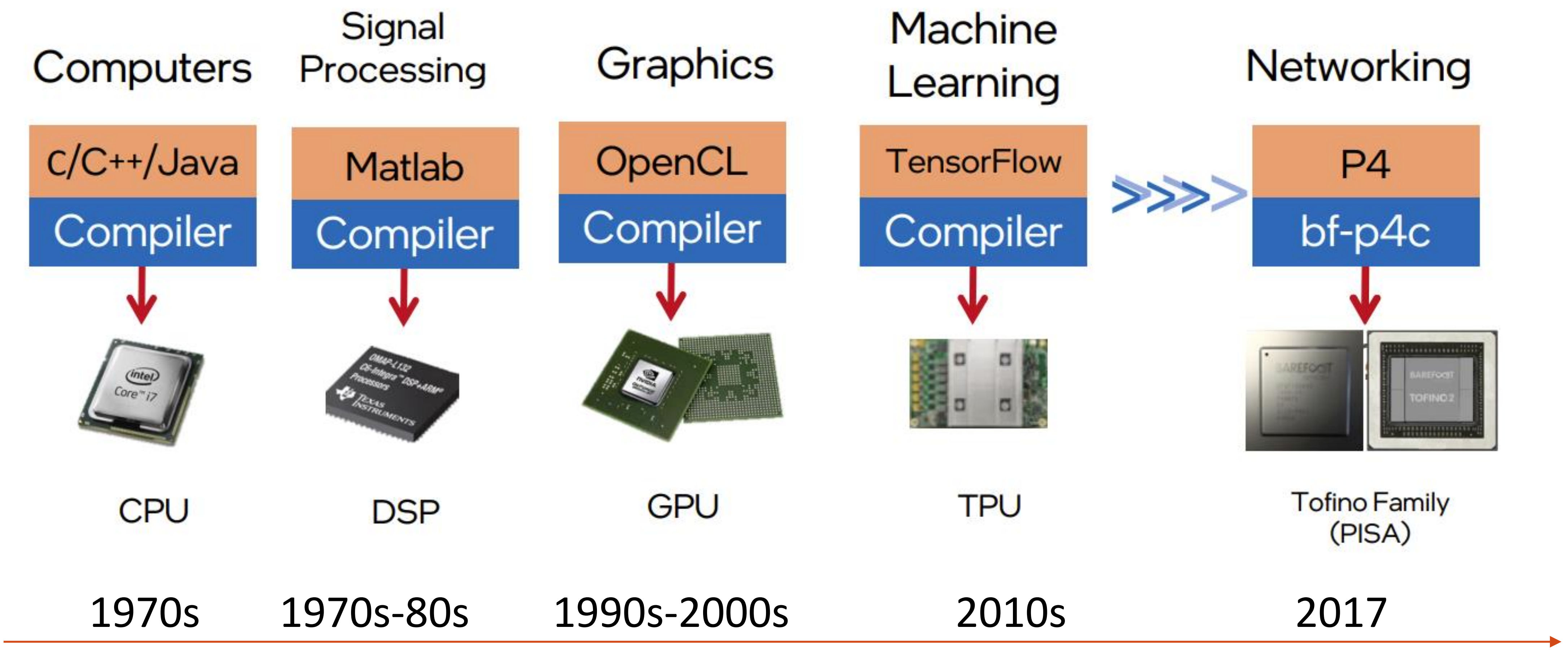
SDN Limitation

- SDN is limited to the OpenFlow specifications
  - ➢ Forwarding rules are based on a fixed number of protocols / header fields (e.g., IP, Ethernet)
- The data plane is designed with fixed functions (hard-coded)
  - ➢ Functions are implemented by the chip designer



Can the Data Plane be Programmable?

- Evolution of the computing industry



1. Vladimir Gurevich, "Introduction to P4 and Data Plane Programmability," https://tinyurl.com/2p978tm9.

P4 Programmable Switches

- P4[1] programmable switches permit **programmers** to program the data plane

```
/*************************************************************************
*********************** P A R S E R  ***********************************
*************************************************************************/

    state parse_ethernet {
        packet.extract(hdr.ethernet);
        transition select(hdr.ethernet.etherType) {
            TYPE_IPV4: parse_ipv4;
            default: accept;
        }
    }

    state parse_ipv4 {
        packet.extract(hdr.ipv4);
        verify(hdr.ipv4.ihl >= 5, error.IPHeaderTooShort);
        transition select(hdr.ipv4.ihl) {
            5             : accept;
            default       : parse_ipv4_option;
        }
    }
```

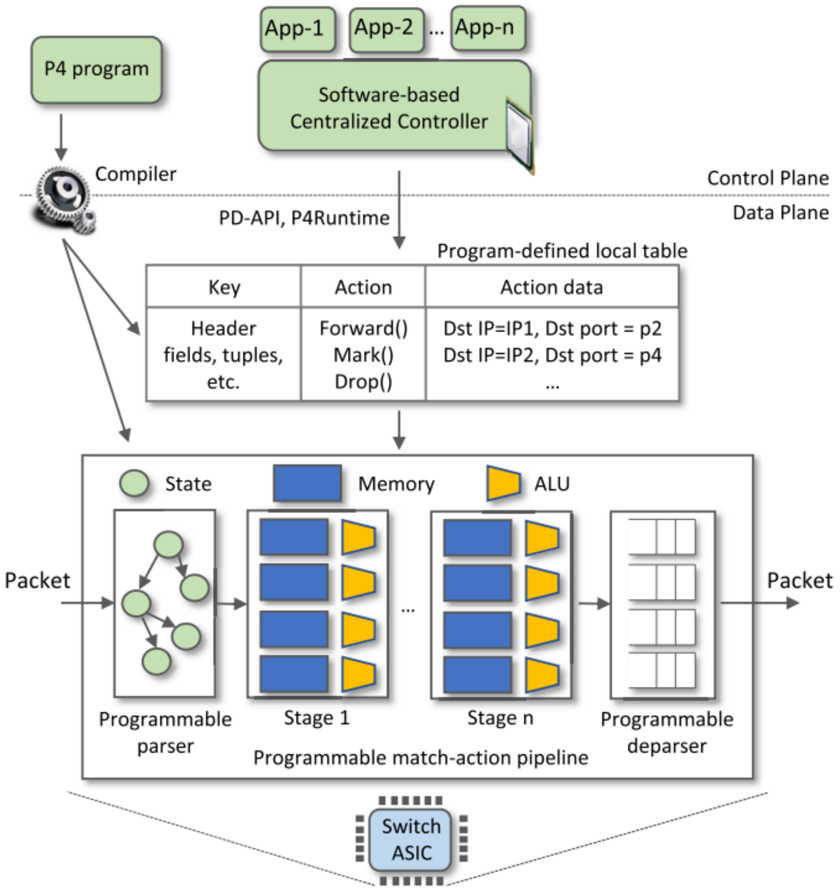
P4 code

Programmable chip

1. P4 stands for stands for Programming Protocol-independent Packet Processors

P4 Programmable Switches

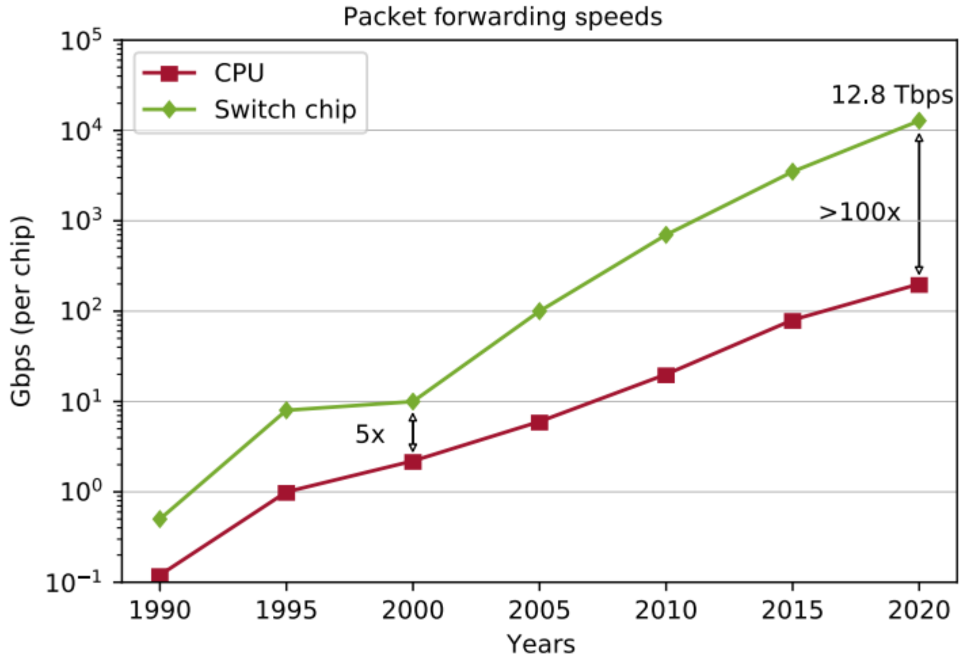
- P4[1] programmable switches permit **programmers** to program the data plane
  - Define and parse new protocols
  - Customize packet processing functions
  - Measure events occurring in the data plane with high precision
  - Offload applications to the data plane



1. P4 stands for stands for Programming Protocol-independent Packet Processors

P4 Programmable Switches

- P4[1] programmable switches permit **programmers** to program the data plane
  - ➢ Define and parse new protocols
  - ➢ Customize packet processing functions
  - ➢ Measure events occurring in the data plane with high precision
  - ➢ Offload applications to the data plane



Reproduced from N. McKeown. Creating an End-to-End Programming Model for Packet Forwarding. Available: https://www.youtube.com/watch?v=fiBuao6YZl0&t=631s

DGA Family Classification using DNS Deep Packet Inspection on P4 Programmable Switches

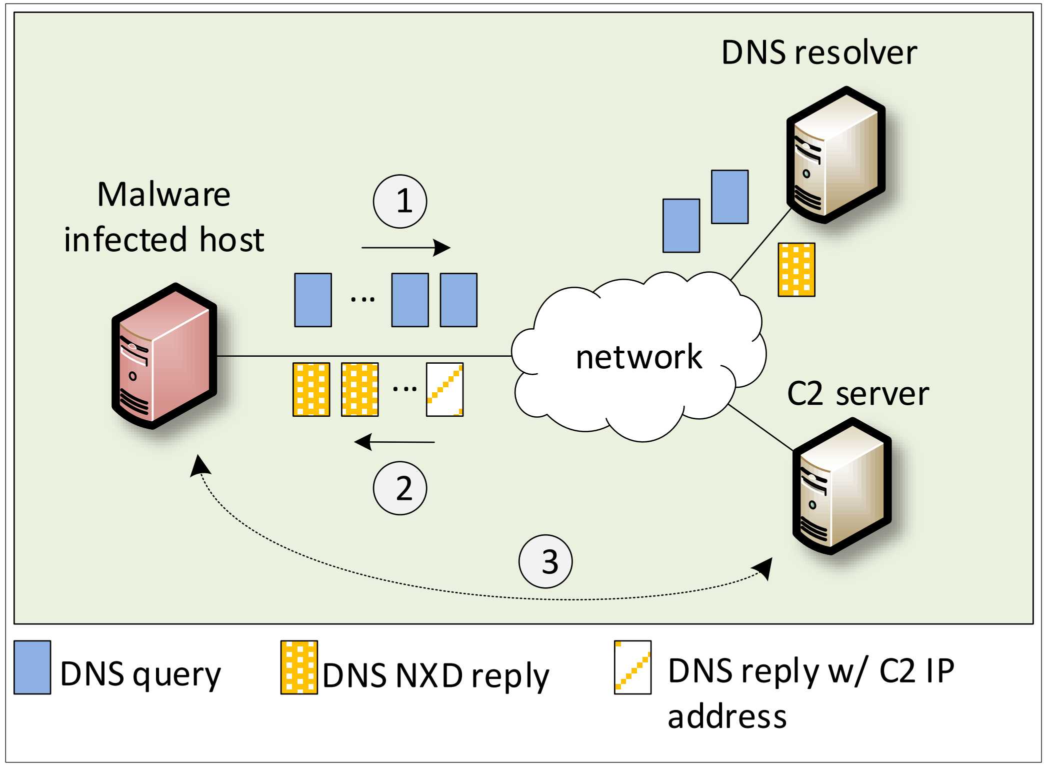
NSF Cybertraining 2118311: “Cybertraining on P4 Programmable Devices using an Online Scalable Platform with Physical and Virtual Switches and Real Protocol Stacks”

Introduction to DGAs

- Attackers often use a Command and Control (C2) server to establish communication between infected host/s and bot master
- Domain Generation Algorithms (DGAs) are the *de facto* dynamic C2 communication method used by malware, including botnets, ransomware, and many others

Introduction to DGAs

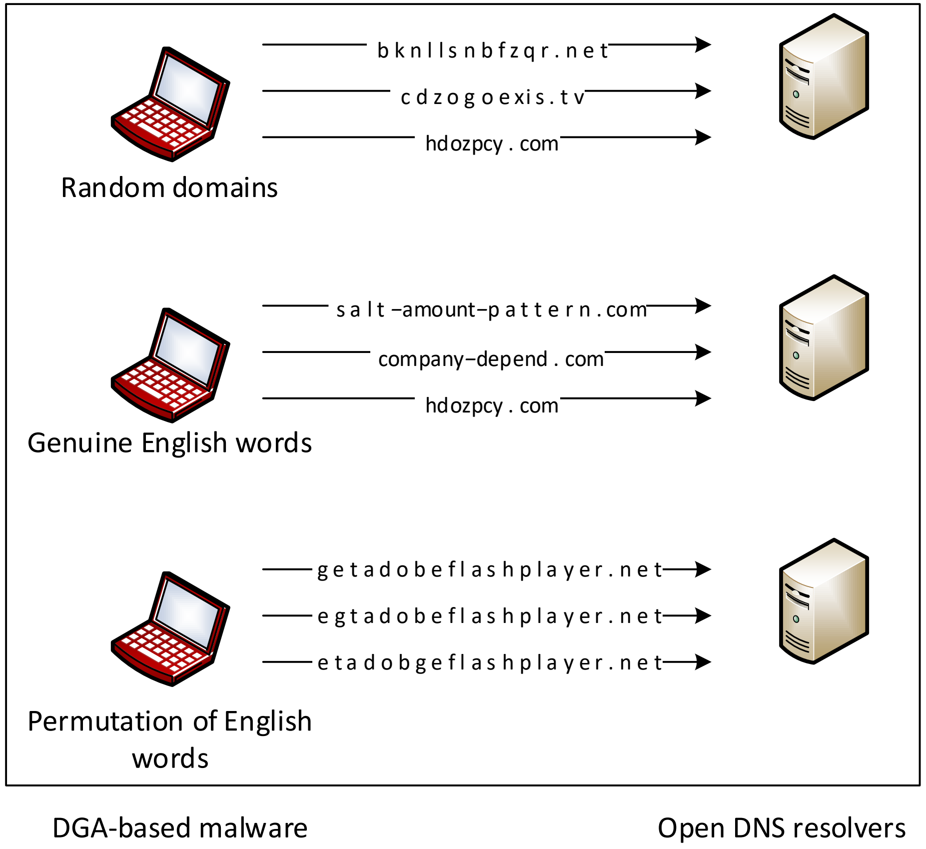
- DGAs evade firewall controls by frequently changing the domain name selected from a large pool of candidates
- The malware makes DNS queries to resolve the IP addresses of these generated domains
- Only a few of these queries will be successful; most of them will result in Non-Existent Domain (NXD) responses



(1) DNS queries. (2) (NXD) replies. (3) Eventually, a query for the actual domain is sent and malware-C2 communication starts.

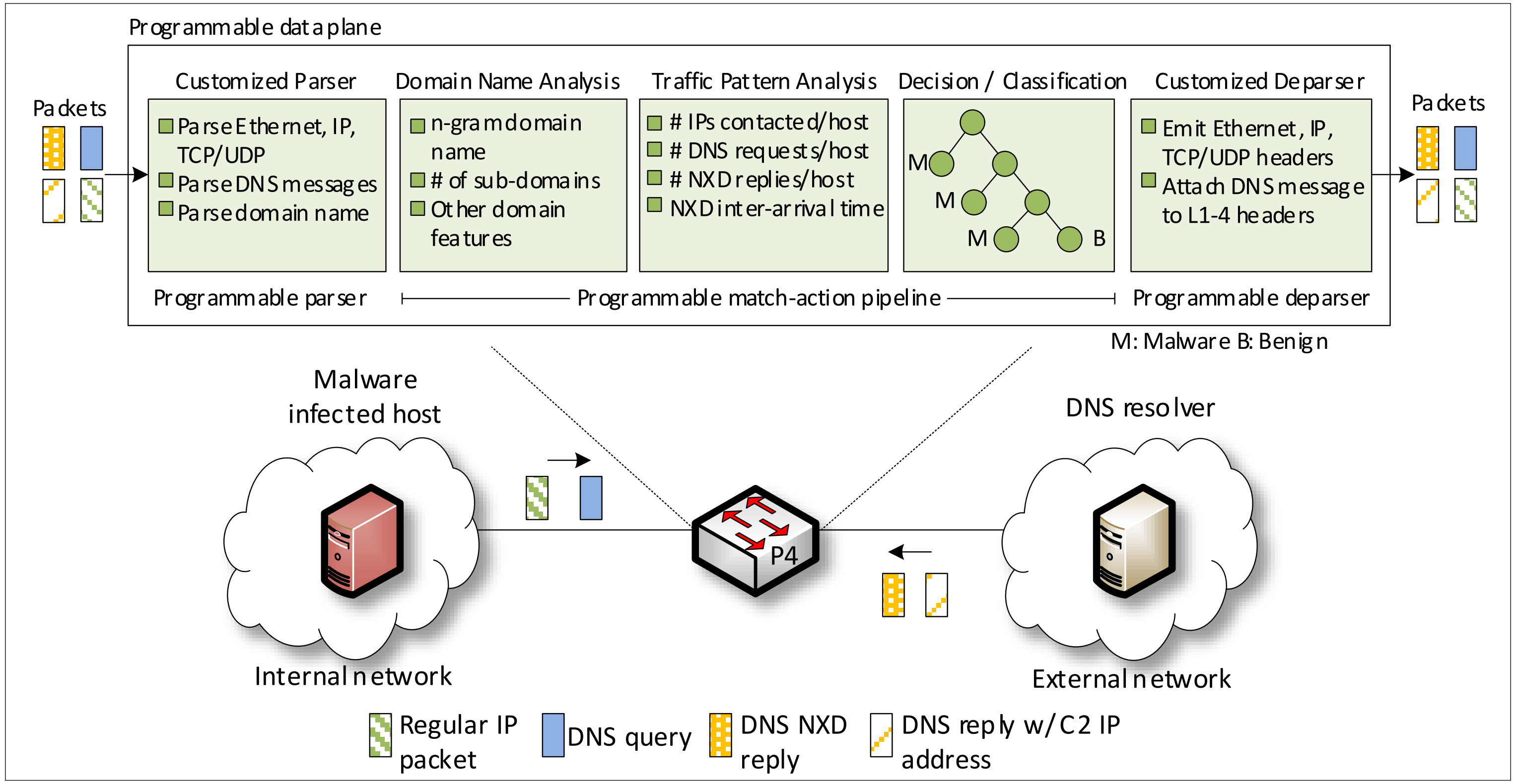
Introduction to DGAs

- DGAs evade firewall controls by frequently changing the domain name selected from a large pool of candidates
- The malware makes DNS queries to resolve the IP addresses of these generated domains
- Only a few of these queries will be successful; most of them will result in Non-Existent Domain (NXD) responses



Proposed System

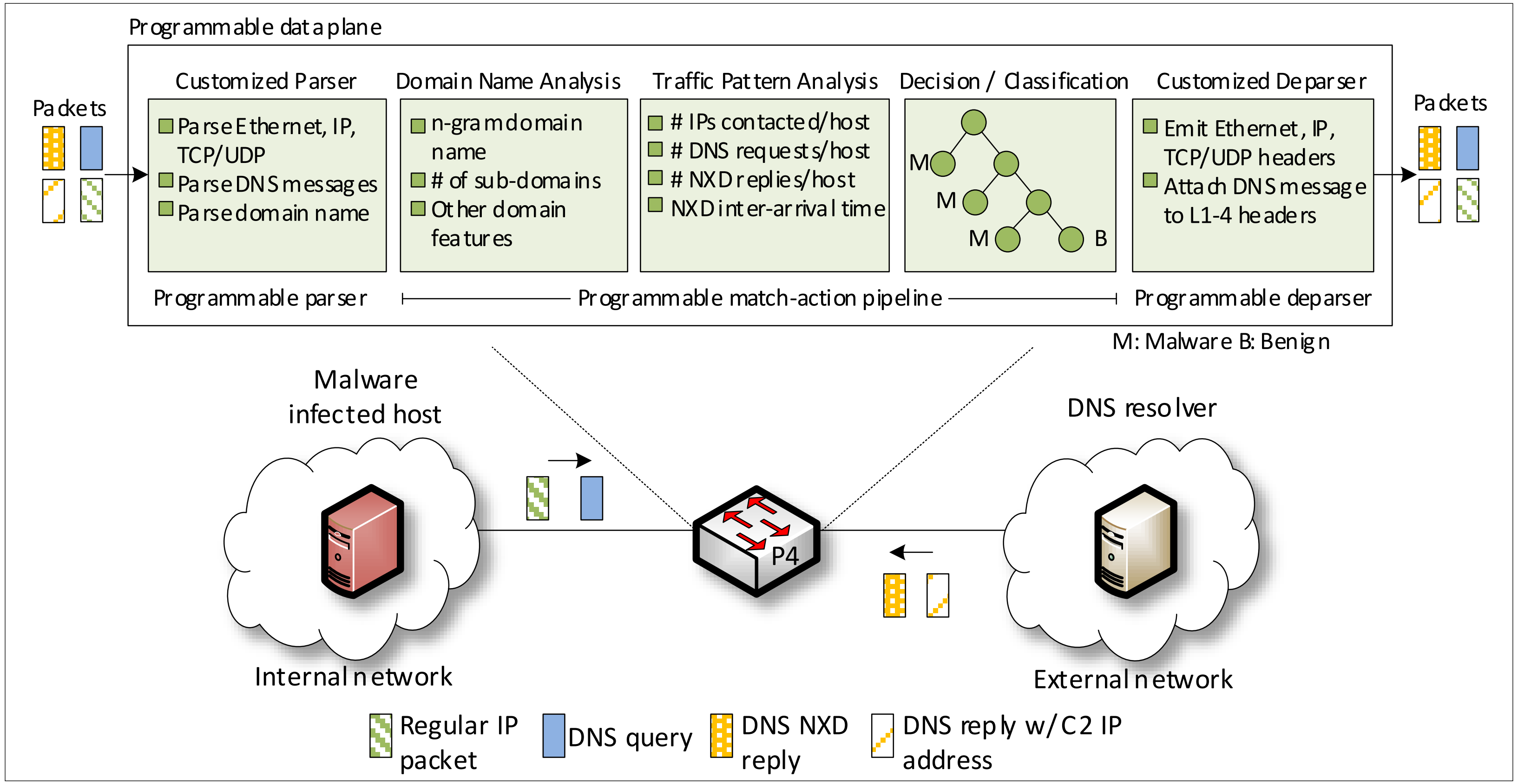
- The proposed system uses P4 programmable data plane switches to
  - Run a customized packet parser
  - Collect fine-grained measurements
  - Perform per-packet inspection
  - Process packets at line rate



1. A. AlSabeh, K. Friday, E. Kfoury, J. Crichigno, E. Bou-Harb, “On DGA Detection and Classification using P4 Programmable Switches,” under review, Journal of Computers and Security.

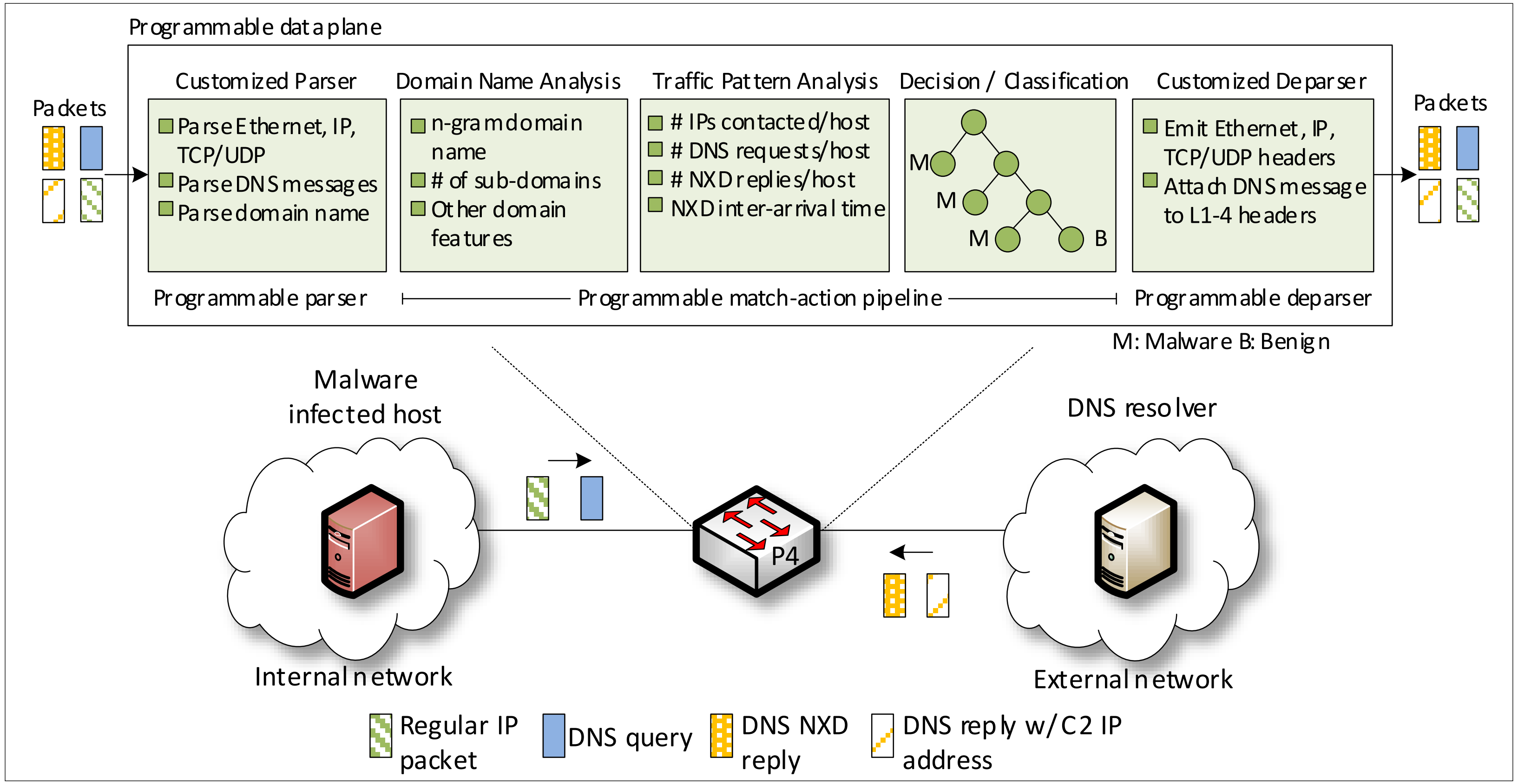
Proposed System

- The switch collects and stores the **traffic features** of the hosts (Traffic Pattern Analysis)
  - Number of IP addresses contacted, number of DNS requests made, Inter-arrival Time (IAT) between consecutive IP packets, time it takes for the first NXD response to arrive, IAT between subsequent NXD responses



Proposed System

- When an NXD response is received, the switch performs deep-packet inspection (DPI) on the domain name to extract **domain features** (**Domain Name Analysis**)
  - For classification, the data plane sends the collected features to the control plane, which runs the intelligence to classify the DGA family and initiate the appropriate response



Proposed System

- The scheme uses the bigram technique for the domain name analysis:
  - ➢ It computes the bigram of the domain name; a bigram model may suffice to predict whether a domain name is a legitimate human readable domain

$$score\,(d) = \sum_{\forall\ subdomain\ s \in d} \left(\sum_{\forall\ bigram\ b \in s} f_s^b \right)$$

Where f_S^b is the frequency of the bigram b in the subdomain *s*

  - ➢ The frequency value of a bigram b is pre-computed and stored in a Match-Action Table (MAT)
  - ➢ Example: the bigrams of “google” are: “go”, “oo”, “og”, “gl”, “le”
  - ➢ The lower the score, the more random the domain name

Evaluation

- Experimental setup
  - Hundreds of GB of malware samples; 1,311 samples containing 50 DGA families[1]
  - We used samples that receive NXD responses containing domain names generated by DGAs[1]
  - The collected dataset was used to train ML models offline on a general-purpose CPU
  - 80% of data was used for training and 20% for testing

[1] D. Lohmann, "DGArchive." [Online]. Available: https://tinyurl. com/yc6whwrc.

Evaluation

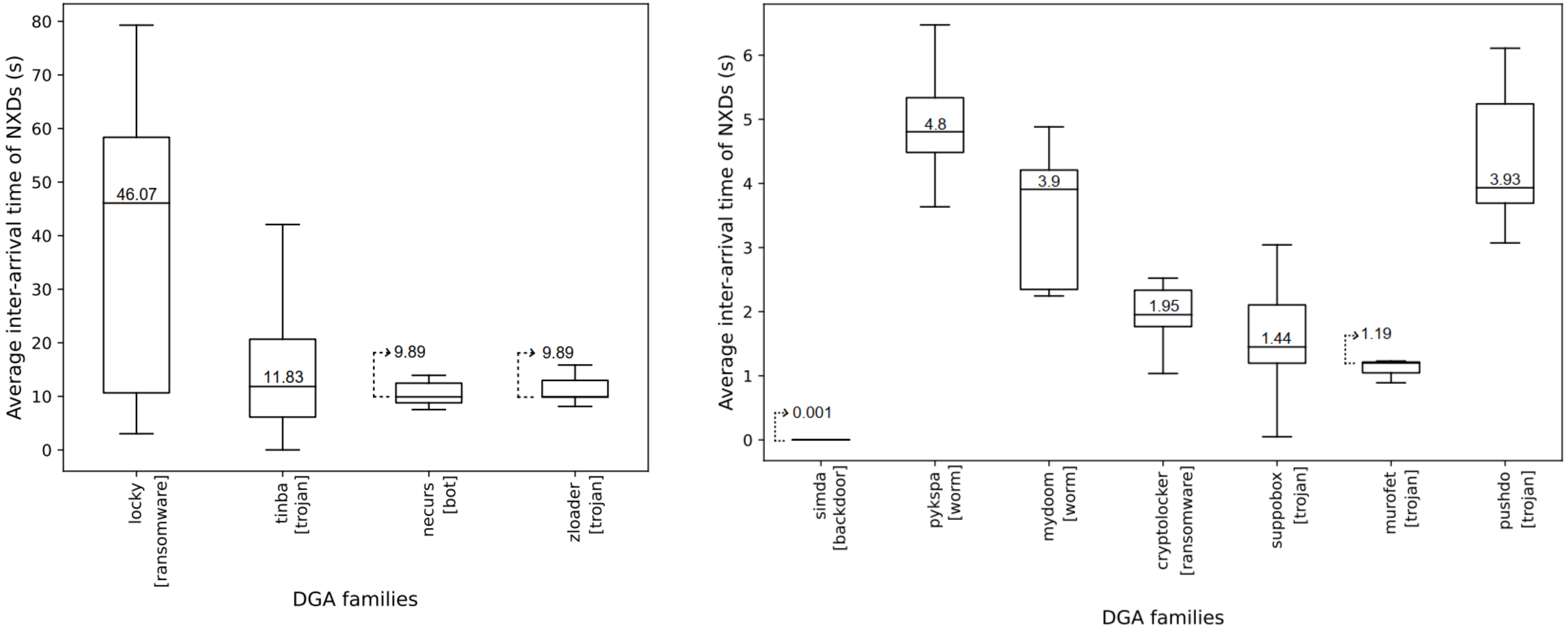
- The evaluation reports the accuracy (ACC) of different ML classifiers during the first 50 NXD responses
  - P4-DGAD RF (detection) is fully implemented in the data plane
  - For detection, all algorithms have an ACC > 0.9 with four or more NXD responses
  - For classification, the ACC of the proposed scheme is comparable to that of CPU-based schemes (with minimal control-plane intervention)

Approach		Model	Accuracy with respect to the number of NXD responses received												
			2	3	4	5	6	7	8	9	10	20	30	40	50
Detection	P4-DGAD	RF	0.903	0.908	0.918	0.927	0.933	0.933	0.935	0.942	0.944	0.960	0.971	0.973	0.977
	DGAD	RF	0.991	0.994	0.994	0.995	0.996	0.997	0.997	0.996	0.997	0.998	0.998	0.998	0.999
		SVM	0.968	0.963	0.961	0.963	0.972	0.964	0.966	0.960	0.969	0.964	0.972	0.976	0.979
		MLP	0.991	0.994	0.993	0.99	0.994	0.995	0.994	0.996	0.996	0.996	0.997	0.997	0.998
		GNB	0.826	0.896	0.94	0.943	0.955	0.960	0.957	0.955	0.955	0.955	0.960	0.957	0.957
		LR	0.956	0.967	0.969	0.968	0.967	0.972	0.967	0.972	0.970	0.977	0.977	0.971	0.973
Classification	DGAMC	RF	0.894	0.900	0.921	0.927	0.934	0.938	0.945	0.946	0.951	0.965	0.972	0.976	0.979
		SVM	0.836	0.866	0.863	0.875	0.874	0.890	0.881	0.896	0.892	0.880	0.901	0.906	0.915
		MLP	0.866	0.877	0.888	0.917	0.905	0.904	0.921	0.927	0.933	0.943	0.952	0.962	0.961
		GNB	0.769	0.716	0.696	0.611	0.666	0.596	0.630	0.640	0.641	0.672	0.709	0.722	0.722
		LR	0.799	0.806	0.818	0.818	0.828	0.818	0.840	0.834	0.836	0.800	0.822	0.841	0.849

RF: Random Forest; SVM: Support Vector Machine; MLP: Multilayer perceptron; LR: Logistic Regression; GNB: Gaussian Naive Bayes
P4-DGAD: DGA detection algorithm runs fully in the data plane
DGAD: Detection algorithm runs in the control plane
DGAMC: Classifier algorithm runs in the control plane

Evaluation

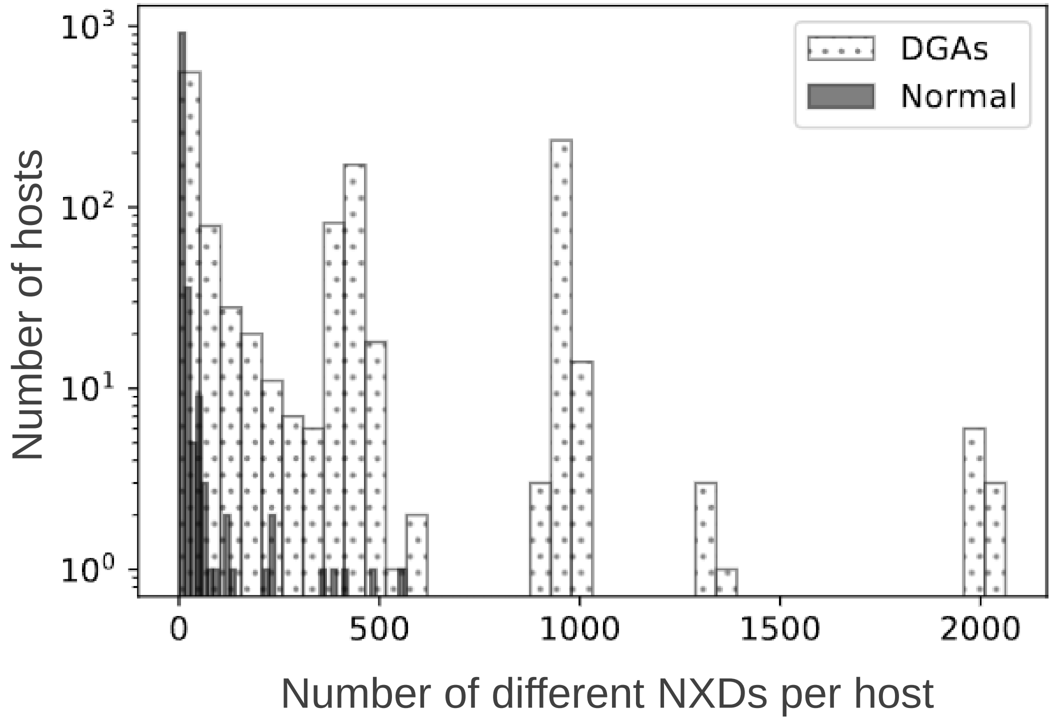
- The scheme can accurately characterize traffic flows (traffic features)



Interarrival times between NXDs of DGA families with the largest number of samples

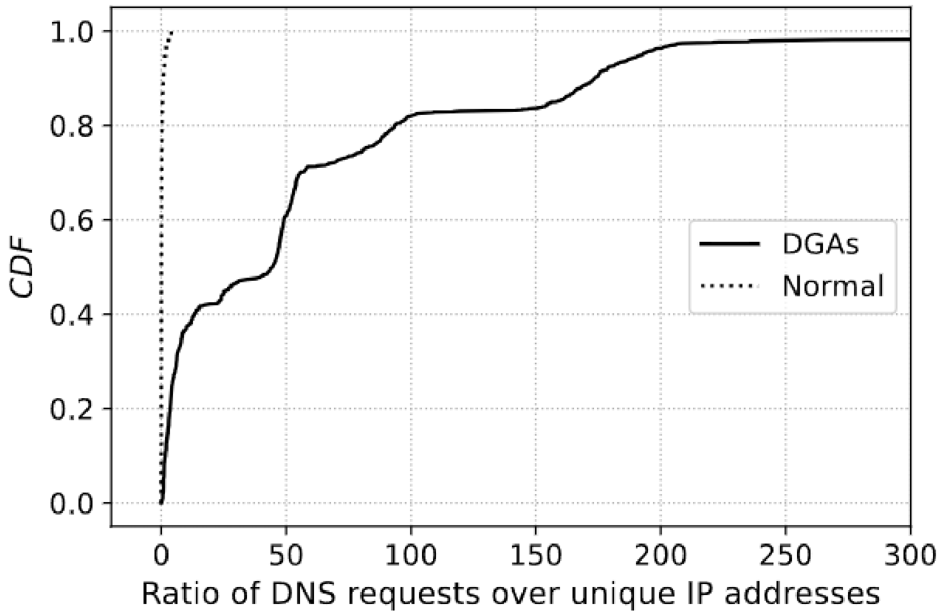
Evaluation

- The scheme can accurately characterize traffic flows (traffic features)
  - Normal (benign) hosts typically generates a few NXDs (at most)



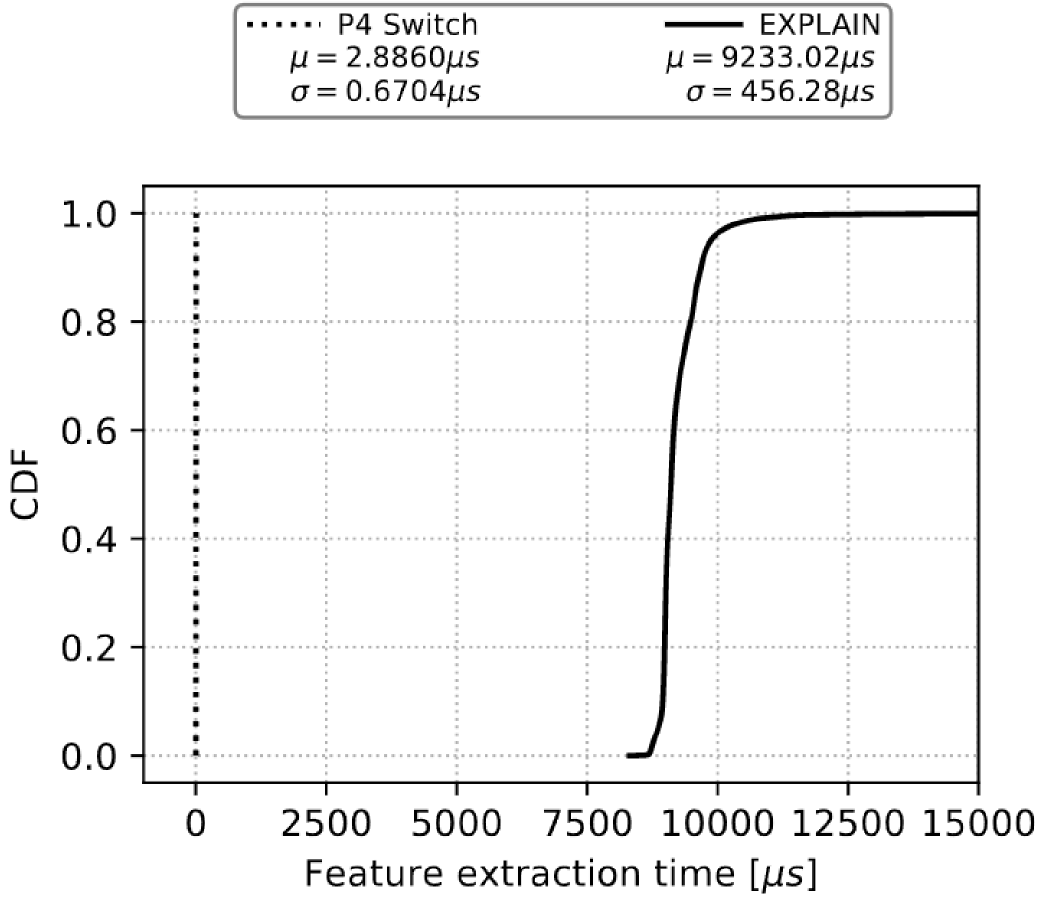
Evaluation

- The scheme can accurately characterize traffic flows (traffic features)
  - When a normal (benign) hosts queries a given domain, the DNS system returns a corresponding IP address (ratio of DNS requests to IP addresses is approximately 1)
  - DGAs often query hundreds of domains; only few queries return an IP address (at best) (ratio of DNS requests to IP addresses > 1)



Evaluation

- Comparison of the feature extraction time of the proposed approach vs EXPLAIN[1]
  - The proposed approach runs on the switch data plane
  - EXPLAIN runs on a general-purposed CPU with 64 GB RAM, 2.9 GHz processor with eight cores



[1]A. Drichel, N. Faerber, U. Meyer, "First step towards explainable DGA multiclass classification," in the 16th International Conference on Availability, Reliability and Security, pp. 1–13, 2021.

DEMO – High-resolution Measurements

https://youtu.be/cWaWxsqVAgc

DEMO – DoS

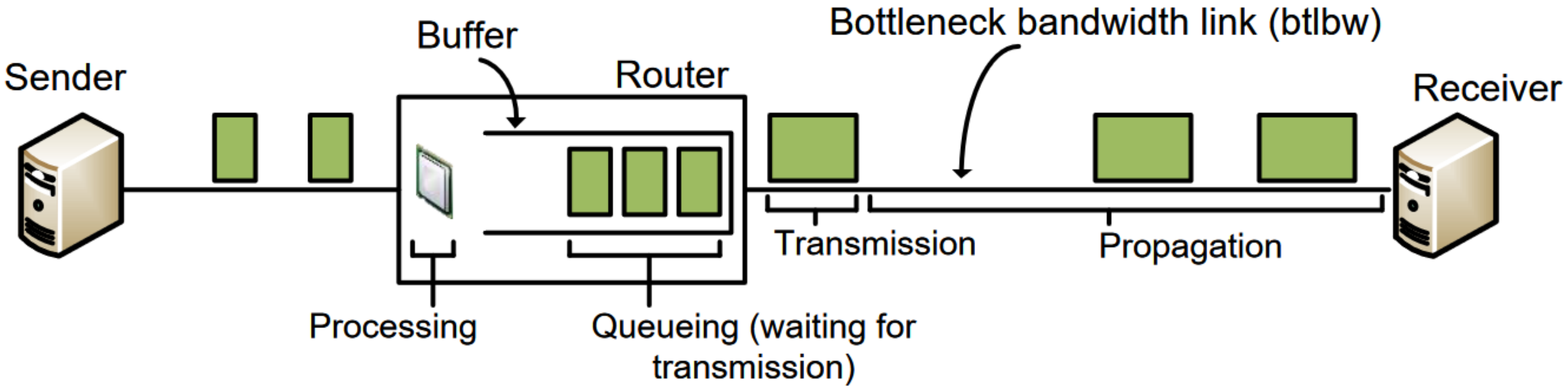
https://youtu.be/EGQHUdrQ80M

Dynamic Router's Buffer Sizing using Passive Measurements and P4 Programmable Switches

NSF CC* 2346726: "CC* Integration-Small: Enhancing Data Transfers by
Enabling Programmability and Closed-loop Control in a Non-programmable Science DMZ"

Buffer Sizing Problem

- Routers and switches are designed to include packet buffers
- The size of buffers impacts the performance of the network
- If the buffer allocated to an interface is
  - Very large, then packets may experience excessive delay (“bufferbloat”)
  - Very small, then there may be a large packet drop rate and low link utilization

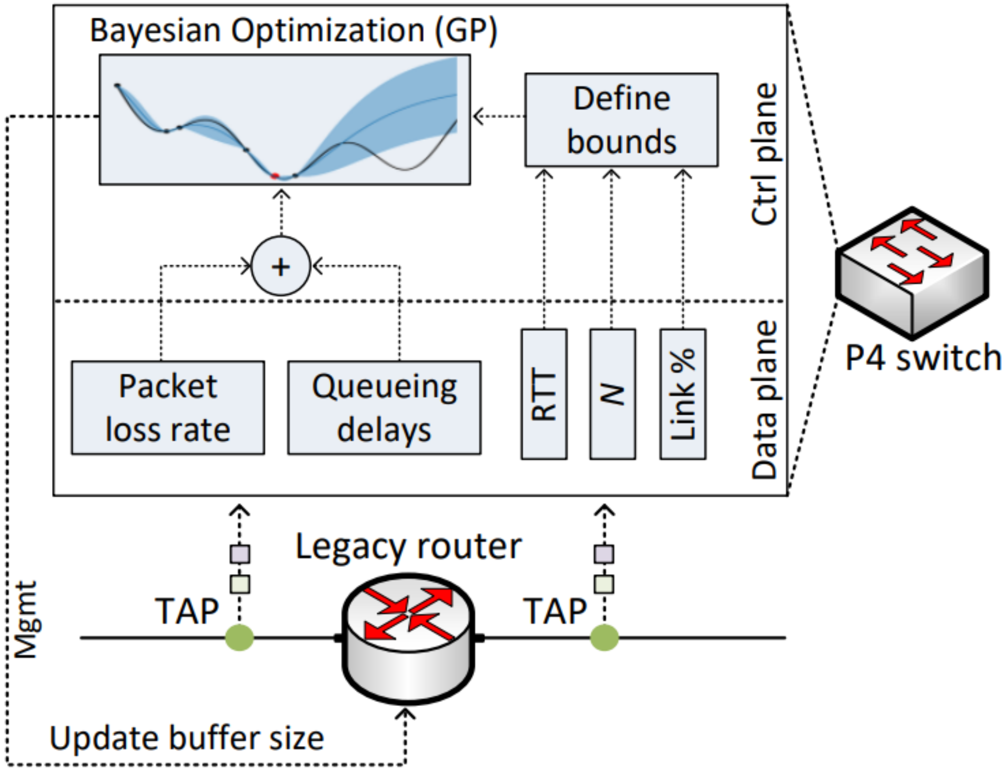


Buffer Sizing Problem

- Th General rule-of-thumb in the 90s was that the buffer size must equal the Bandwidth-delay product (BDP)
  - Buffer = C * RTT
  - C is the capacity of the port and RTT is the average round-trip time (RTT)
- The “Stanford Rule” corrected the previous rule
  - Buffer = $(C * RTT)/(\sqrt{N})$
  - N is the number of long (persistent over time) flows traversing the port
- Operators hardcode the buffer size based on the typical traffic pattern

Proposed System

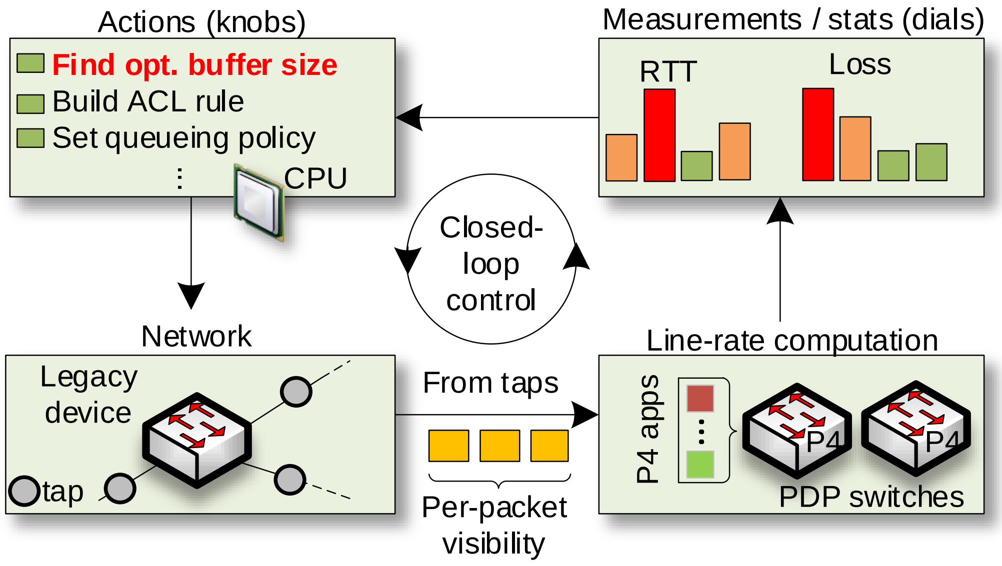
- The buffer size is dynamically modified
- A P4 switch is deployed passively to compute:
  - Number of long flows
  - Average RTT
  - Queueing delays
  - Packet loss rates
- The control plane sequentially searches for a buffer that minimizes delays and losses
- The searching algorithm is Bayesian Optimization (BO) with Gaussian Processes[1]



[1] E. Kfoury, J. Crichigno, E. Bou-Harb, "P4Tune: Enabling Programmability in Non-Programmable Networks," IEEE Communications Magazine, Vol. 61, Issue 3, Jun. 2023

Proposed System

- The buffer size is dynamically modified
- A P4 switch is deployed passively to compute:
    - Number of long flows
    - Average RTT
    - Queueing delays
    - Packet loss rates
- The control plane sequentially searches for a buffer that minimizes delays and losses
- The searching algorithm is Bayesian Optimization (BO) with Gaussian Processes

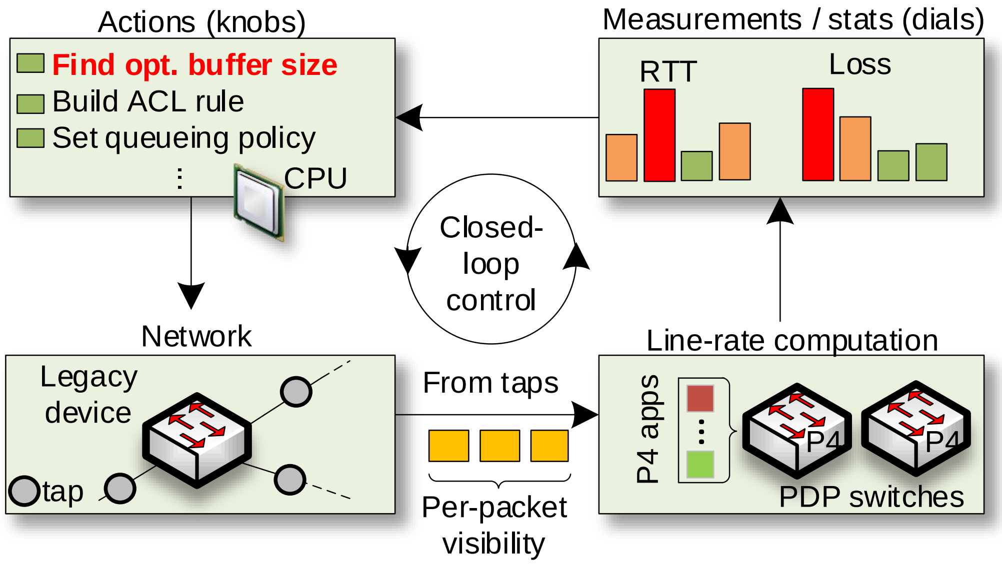


Closed-loop control system

[1] E. Kfoury, J. Crichigno, E. Bou-Harb, “P4Tune: Enabling Programmability in Non-Programmable Networks,” IEEE Communications Magazine, Vol. 61, Issue 3, Jun. 2023

Proposed System

- The system incorporates
  - Customized packet processing
  - Nanosecond resolution measurements
  - Per-packet visibility
  - Packet processing at line rate

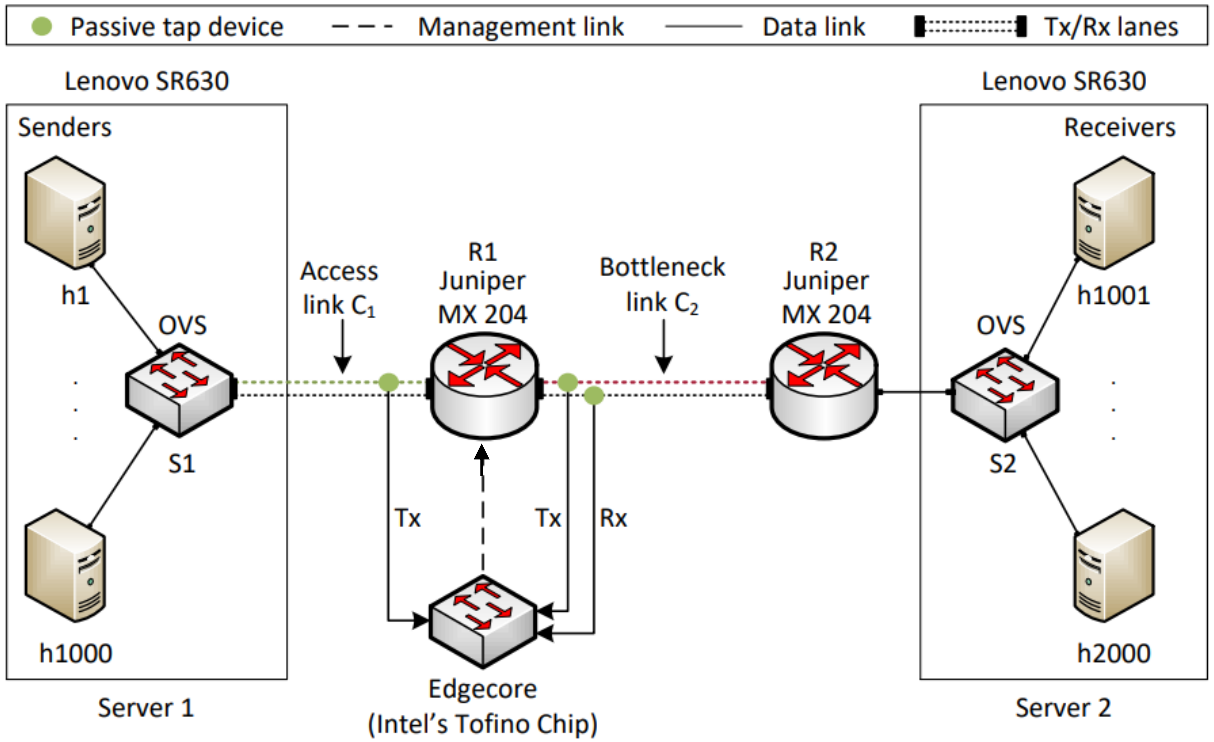


Closed-loop control system

[1] E. Kfoury, J. Crichigno, E. Bou-Harb, "P4Tune: Enabling Programmability in Non-Programmable Networks," IEEE Communications Magazine, Vol. 61, Issue 3, Jun. 2023

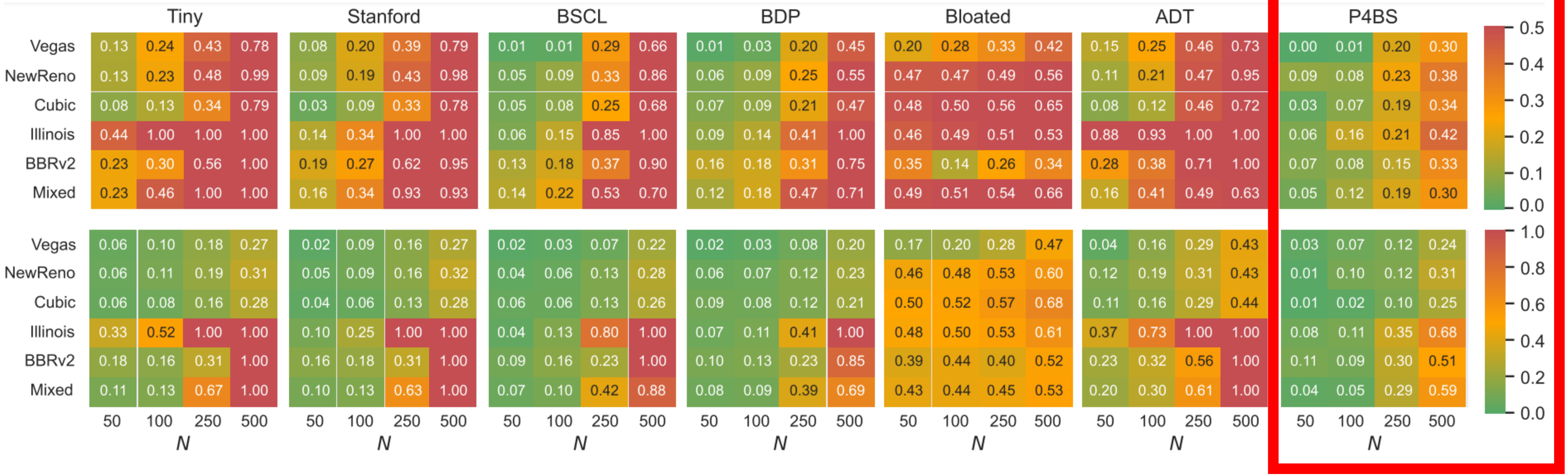
Evaluation

- 1000 senders
- P4 switch: Wedge100BF-32X with Intel's Tofino ASIC
- Legacy router: Juniper router MX-204
- Different congestion control algorithms
- Access network:
  - ➢ C_1 = 40Gbps, C_2 = 1Gbps
- Core network:
  - ➢ C_1 = 10Gbps, C_2 = 2.5Gbps



Results

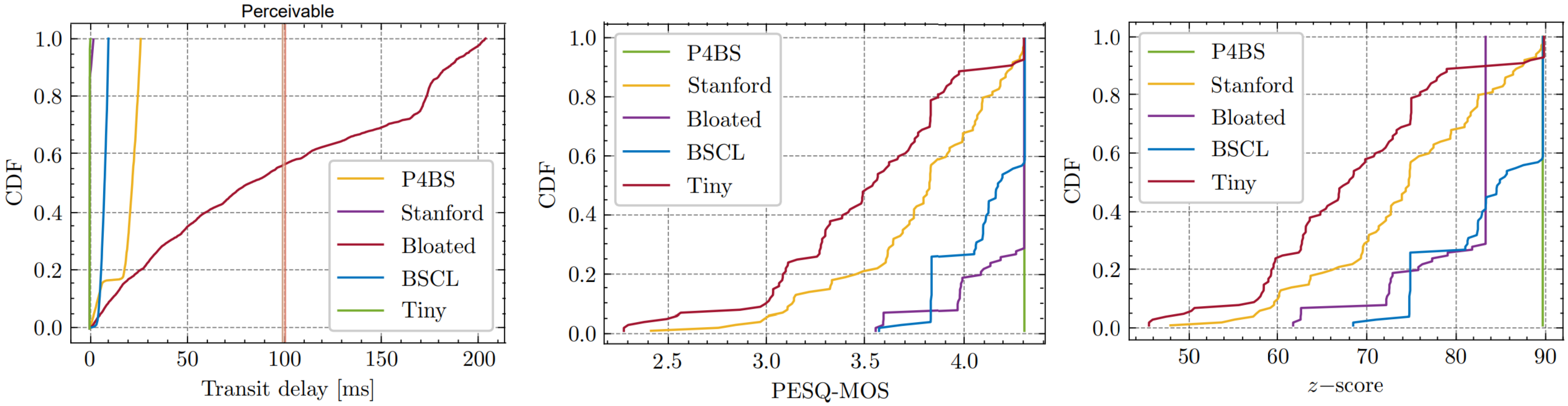
- Combined metric accounting for packet loss and delay [0, 1] (the lower, the better)
- Top heatmaps: access network
- Bottom heatmaps: core network
- The Mixed scenario combines multiple congestion control algorithms[1]



1. E. Kfoury, J. Crichigno, E. Bou-Harb, "P4BS: Leveraging Passive Measurements From P4 Switches to Dynamically Modify a Router's Buffer Size," IEEE Transactions on Network and Service Management, February 2024.

Results

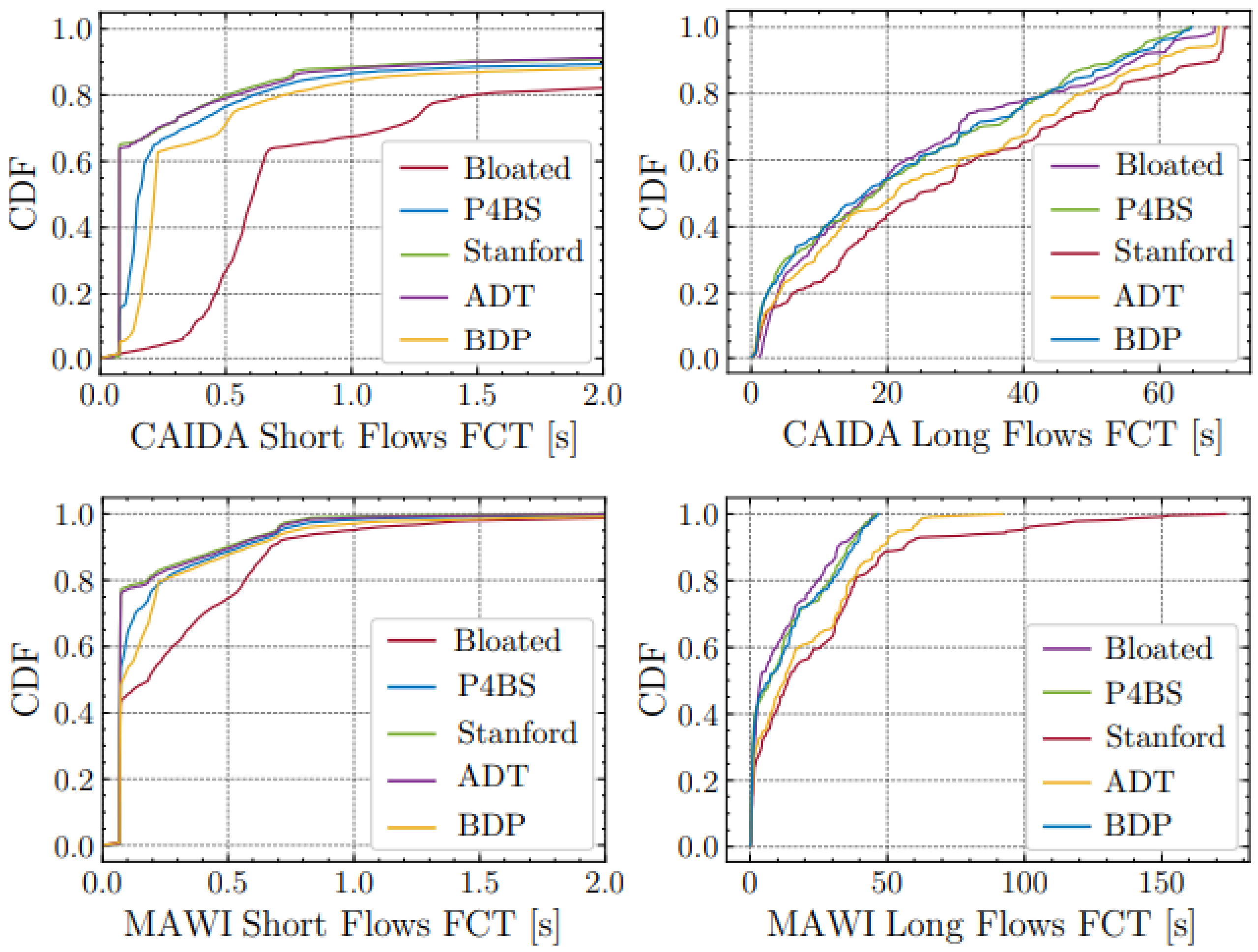
- 100 VoIP calls playing 20 reference speech samples (G.711.a)
- The Perceptual Evaluation of Speech Quality (PESQ) compares an error-free audio signal to a degraded one (the higher, the better)
- The z-score considers both the delay and the PESQ (the higher, the better)



1. E. Kfoury, J. Crichigno, E. Bou-Harb, “P4BS: Leveraging Passive Measurements From P4 Switches to Dynamically Modify a Router’s Buffer Size,” IEEE Transactions on Network and Service Management, February 2024.

Results

- These results use real traffic traces from CAIDA[1] and MAWI[2]
- They include long and short flows
- P4BS found a balance such that:
  - ➢ The FCT of short flows is close to that of the Stanford buffer
  - ➢ The FCT of long flows is close to that of the bloated buffer



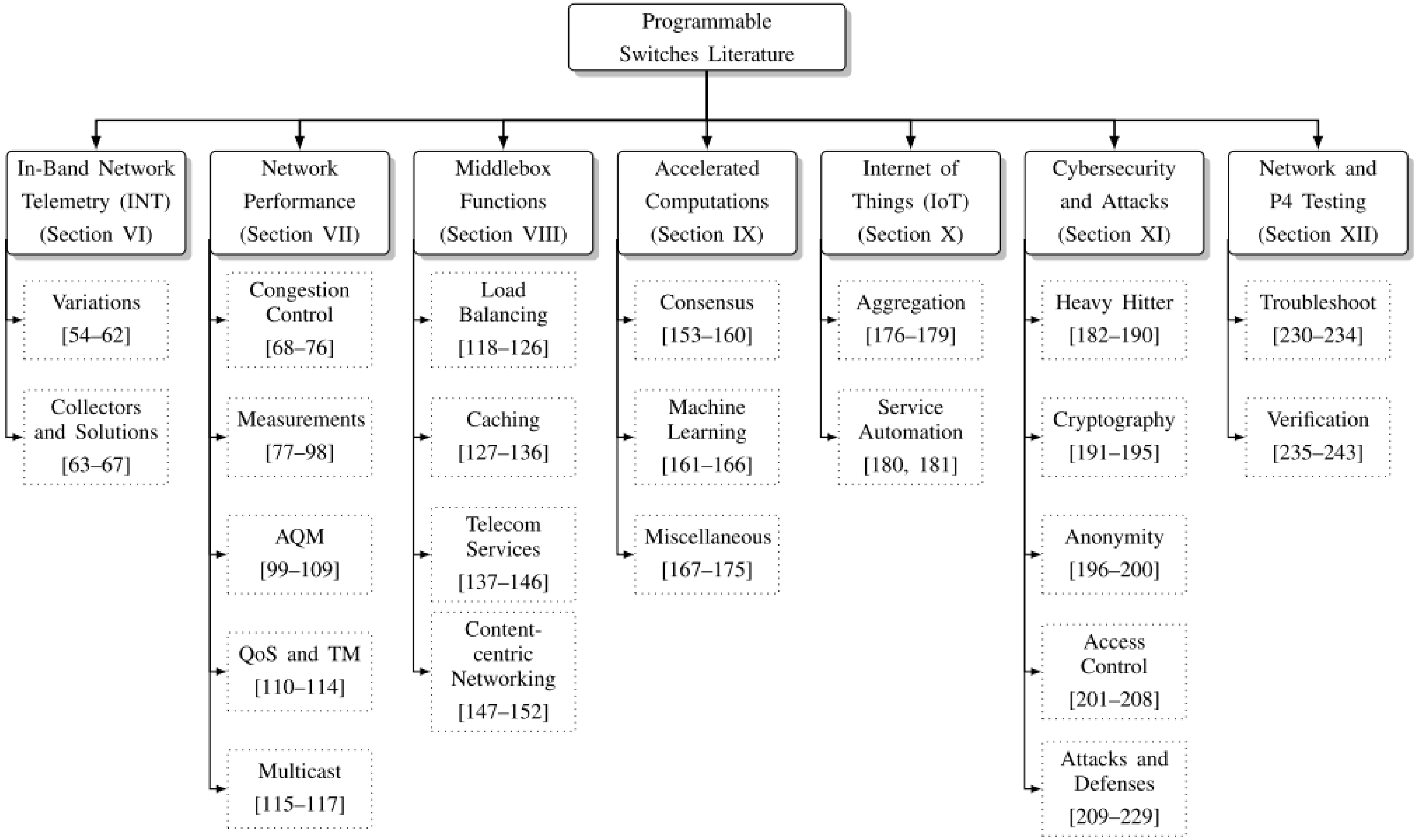
[1] Center for Applied Internet Data Analysis (CAIDA). https://www.caida.org/
[2] MAWI Working Group Traffic Archive. https://mawi.wide.ad.jp/mawi/

Conclusion

- P4 programmable switches enable programmers to control how packets are processed, produce fine-grained measurements, customize parsers and functions, and compute at line rate
- Such capabilities can be used to solve a variety of problems, e.g.,
  - Buffer sizing problem, where programmability is enabled in non-programmable devices
  - DGA problem, where the P4 application can detect DGAs using a combination of DNS deep packet inspection and traffic characterization

Conclusion

- Data plane programmability is enabling a wave of innovation



[1]E. Kfoury, J. Crichigno, E. Bou-Harb, "An Exhaustive Survey on P4 Programmable Data Plane Switches: Taxonomy, Applications, Challenges, and Future Trends", IEEE Access, June 2021.

Contact Information

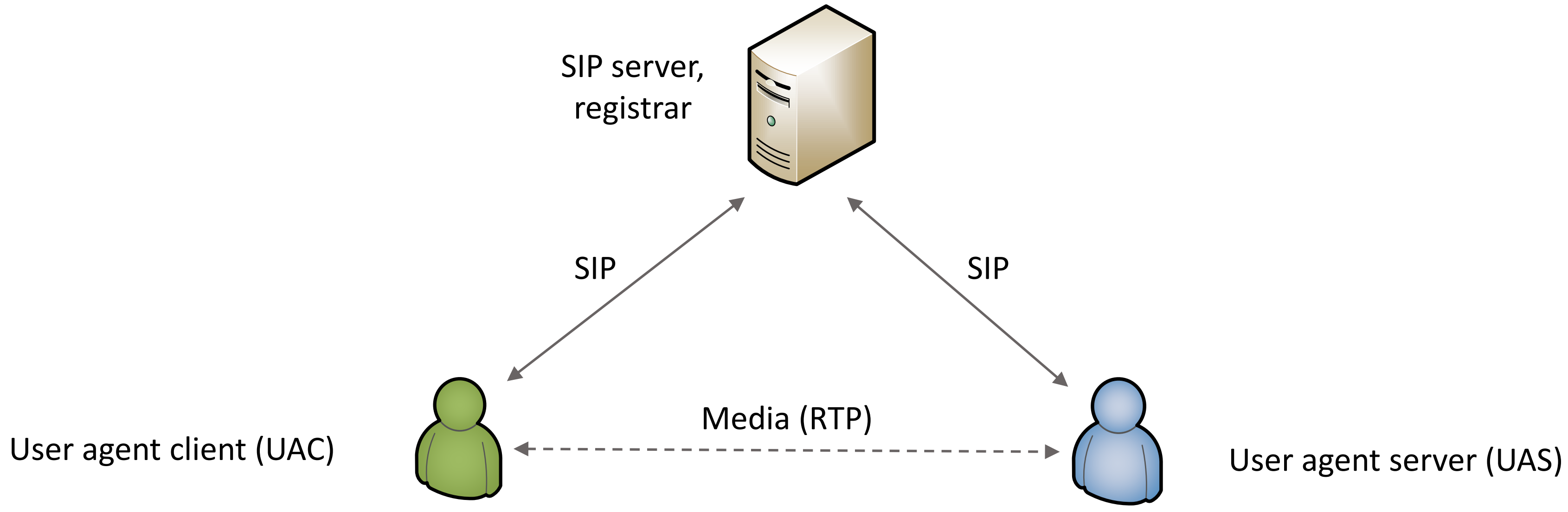
Jorge Crichigno
College of Engineering and Computing, University of South Carolina
jcrichigno@cec.sc.edu
http://ce.sc.edu/cyberinfra

UNIVERSITY OF SOUTH CAROLINA

Offloading Media Traffic to P4 Programmable Data Plane Switches

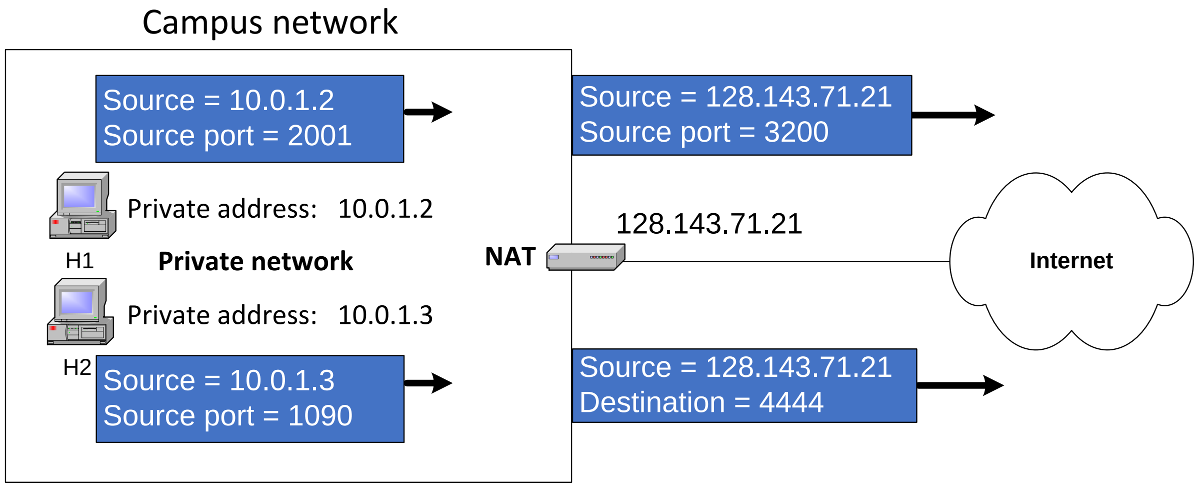
Voice and Video

- Supporting protocols are divided into two main categories
  - Signaling protocols: establish and manage the session; e.g., Session Initiation Protocol (SIP)
  - Media protocols: transfer actual audio and video streams; e.g., Real Time Protocol (RTP)
- Desirable Quality-of-Service (QoS) characteristics
  - Delay- and jitter-sensitive, low values
  - Occasional losses are tolerated



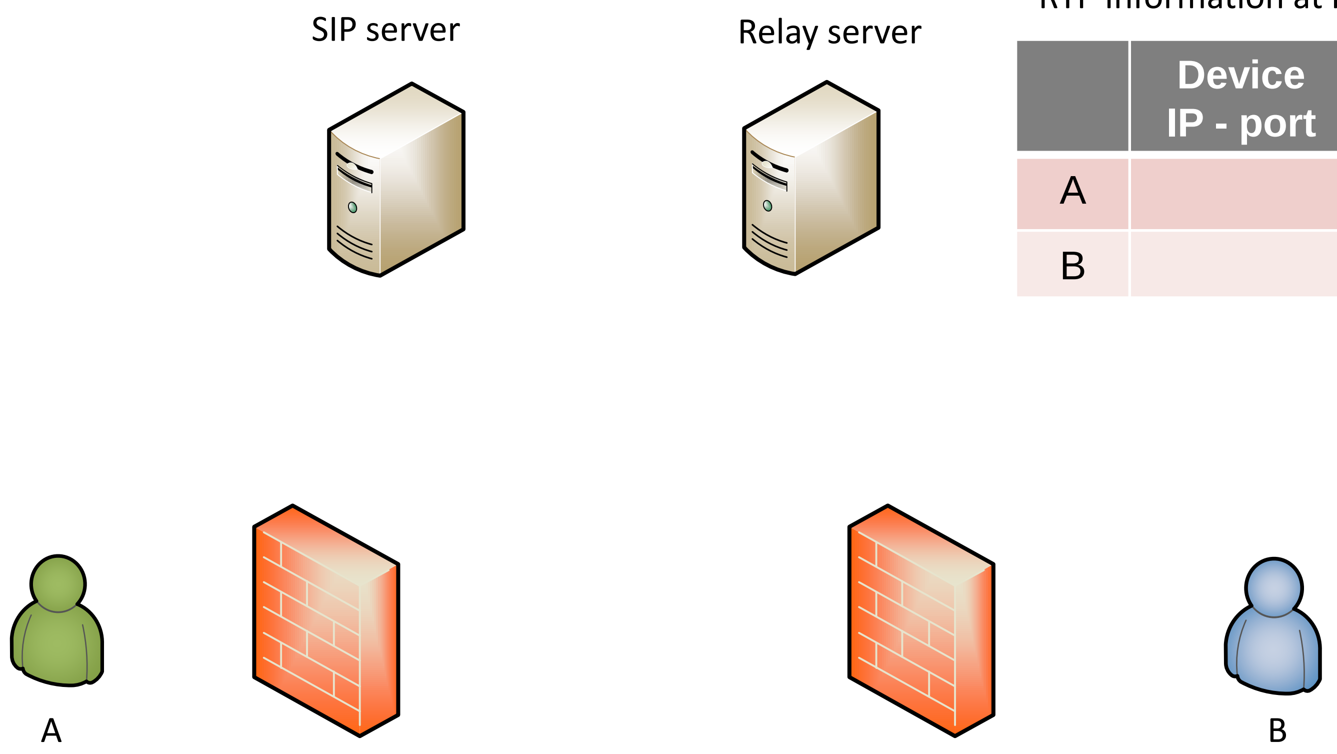
Network Address Translation (NAT)

- NAT maps ports and private IP addresses to ephemeral ports and public IP addresses
  - Used in campus / enterprise networks, operators[1]
- NAT introduces various issues
  - NAT prevents a user from outside from initiating a session
  - If both users are behind NAT, then cannot communicate



Relay Server for Media Traffic

- Intermediary device

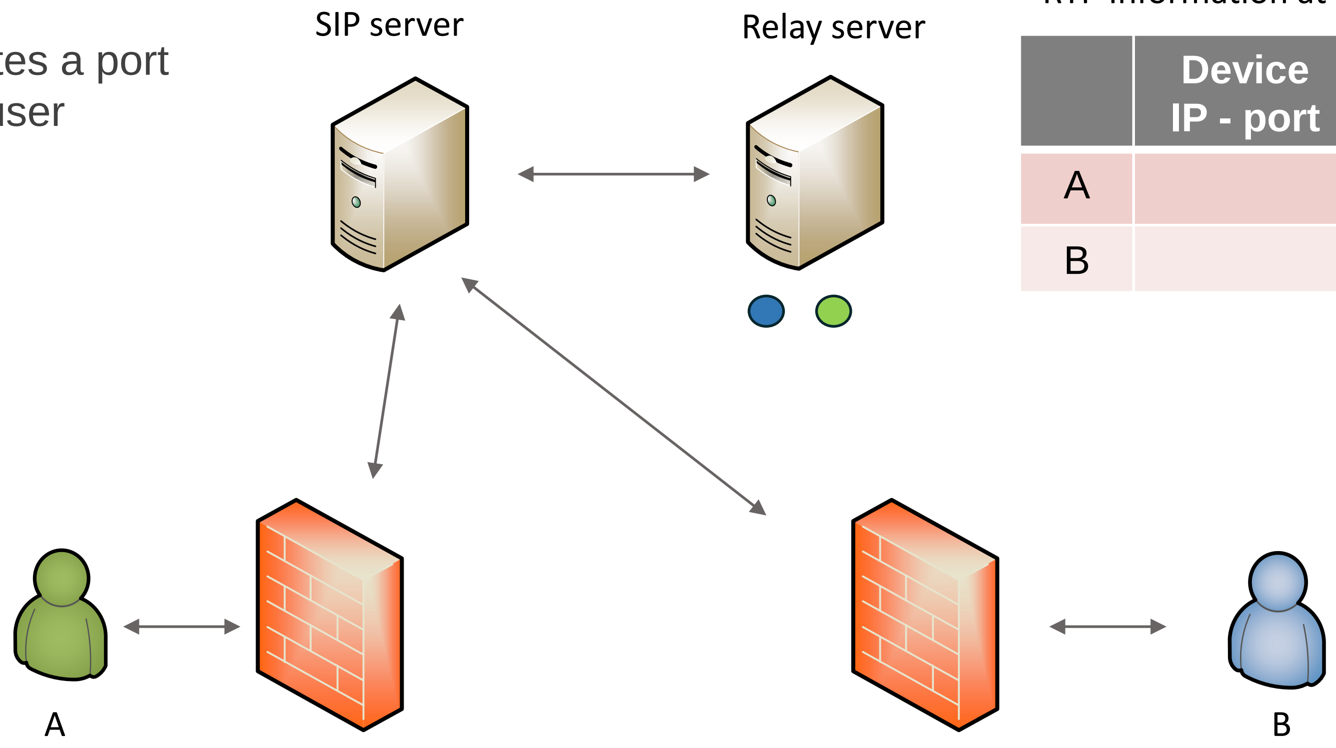


RTP Information at relay server

	Device IP - port	Allocated IP - port
A		
B		

Relay Server for Media Traffic

- Intermediary device
- SIP establishes the session
  - RTP ports are unknown
  - The relay server allocates a port on behalf of each end user



RTP Information at relay server

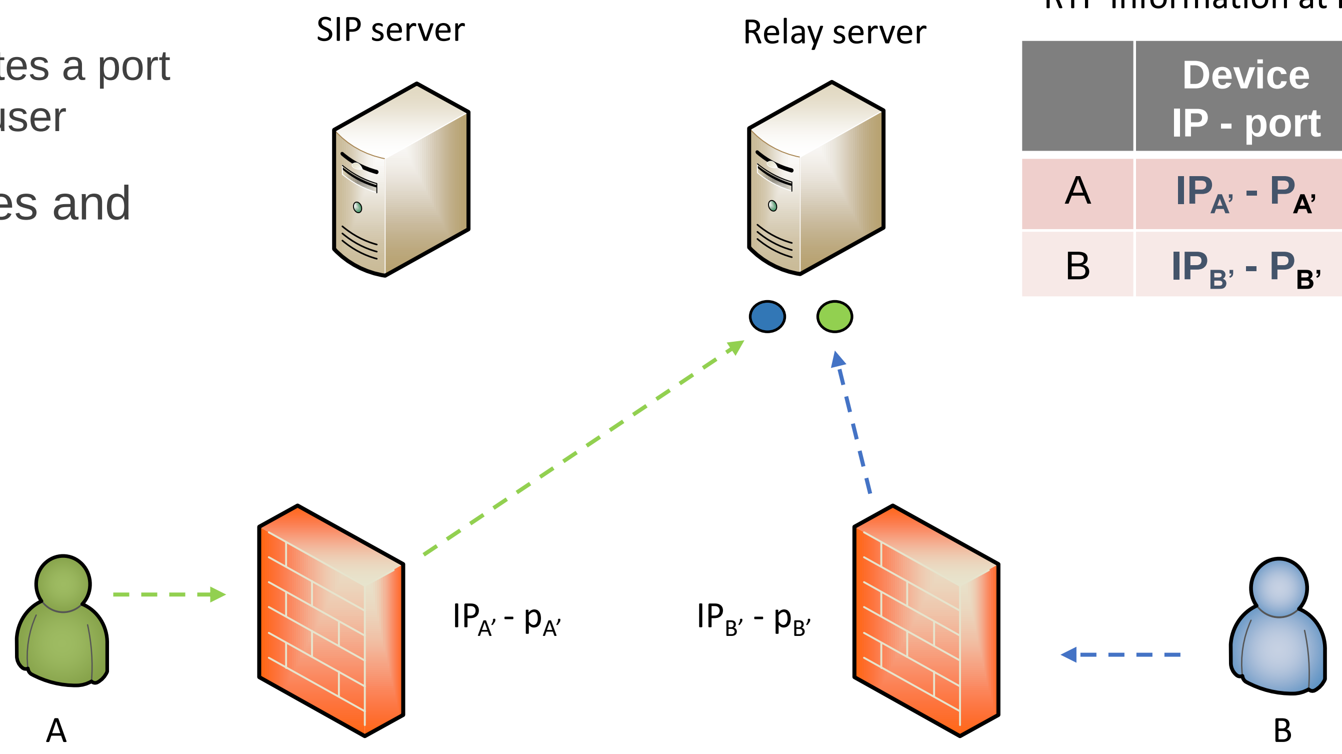
	Device IP - port	Allocated IP - port
A		IP_R - P_{RA}
B		IP_R - P_{RB}

Relay Server for Media Traffic

- Intermediary device
- SIP establishes the session
  - RTP ports are unknown
  - The relay server allocates a port on behalf of each end user
- The relay server receives and relays the RTP traffic

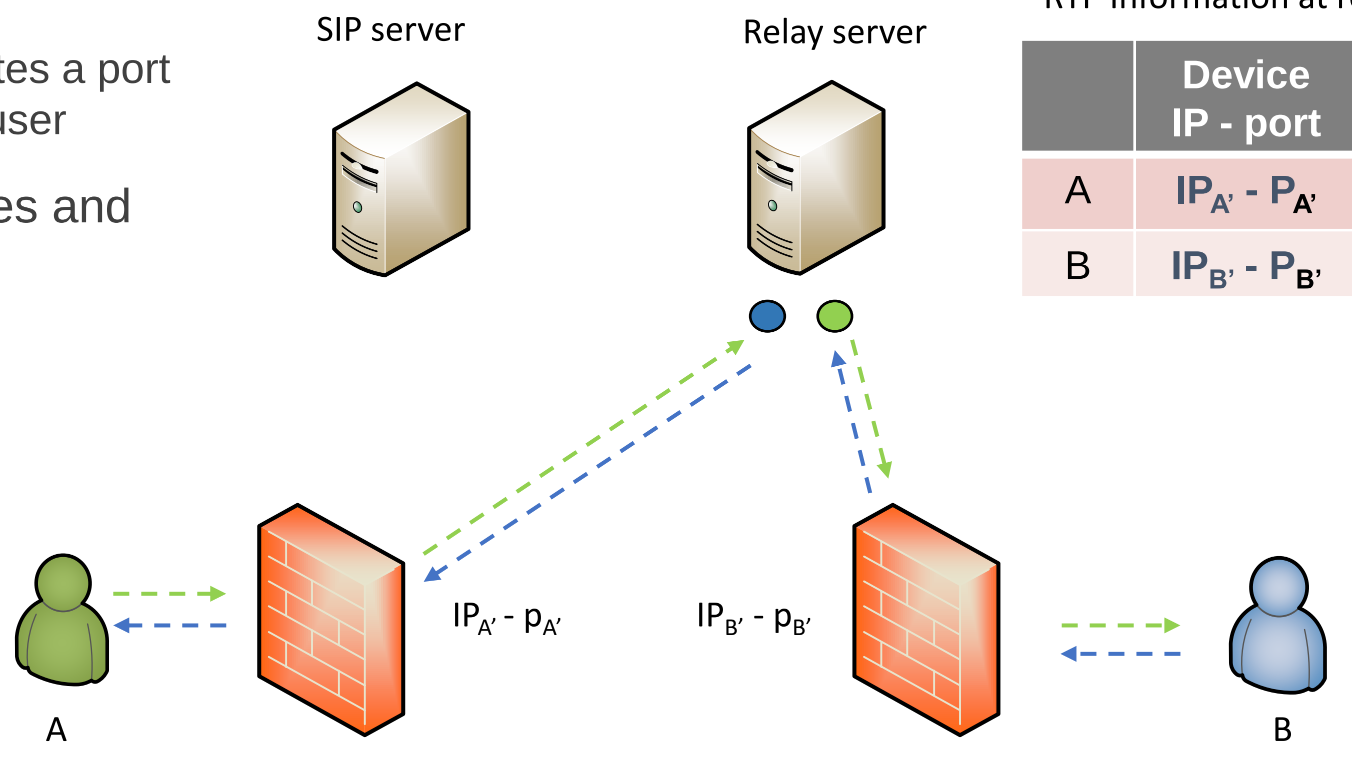
RTP Information at relay server

	Device IP - port	Allocated IP - port
A	$IP_{A'}$ - $P_{A'}$	IP_R - P_{RA}
B	$IP_{B'}$ - $P_{B'}$	IP_R - P_{RB}



Relay Server for Media Traffic

- Intermediary device
- SIP establishes the session
  - RTP ports are unknown
  - The relay server allocates a port on behalf of each end user
- The relay server receives and relays the RTP traffic

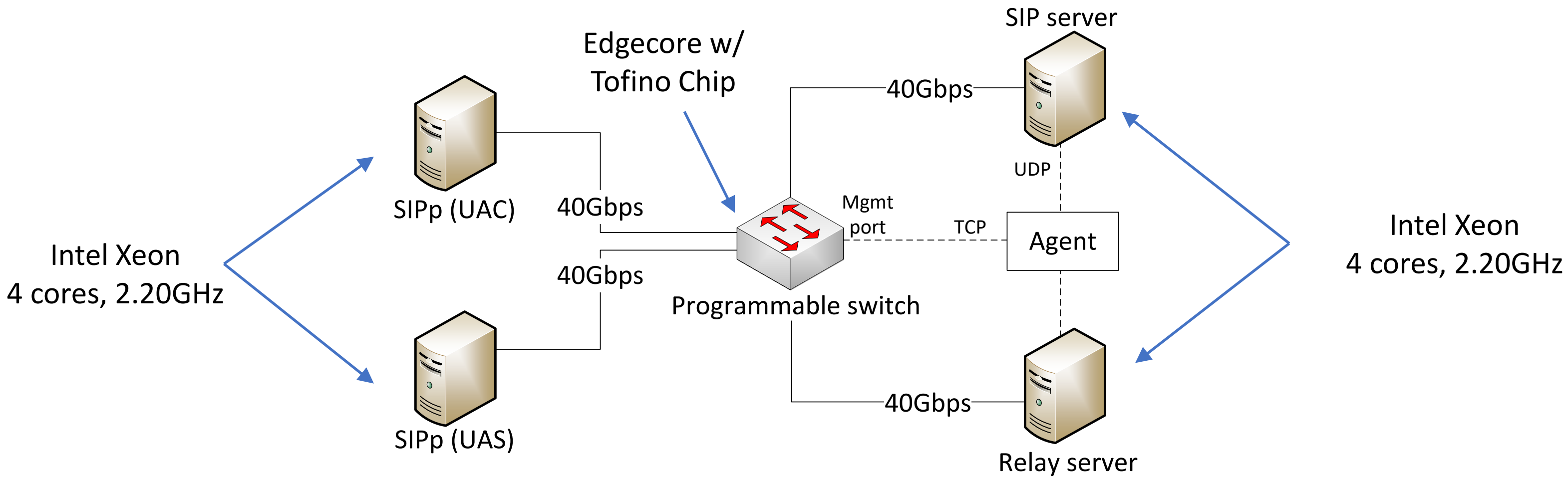


RTP Information at relay server

	Device IP - port	Allocated IP - port
A	$IP_{A'}$ - $P_{A'}$	IP_R - P_{RA}
B	$IP_{B'}$ - $P_{B'}$	IP_R - P_{RB}

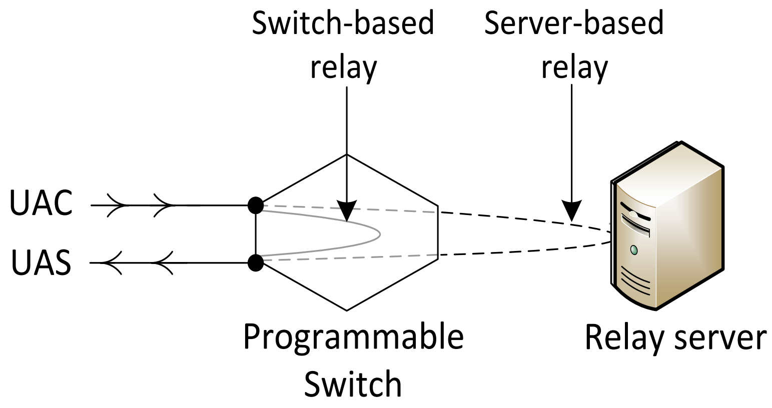
Implementation and Evaluation

- OpenSIPS, an open-source implementation of a SIP server
- RTPProxy, a high-performance relay server for RTP streams
- SIPp: an open-source SIP traffic generator that can establish multiple concurrent sessions and generate media (RTP) traffic
- Iperf3: traffic generator used to generate background UDP traffic
- Edgecore Wedge100BF-32X: programmable switch



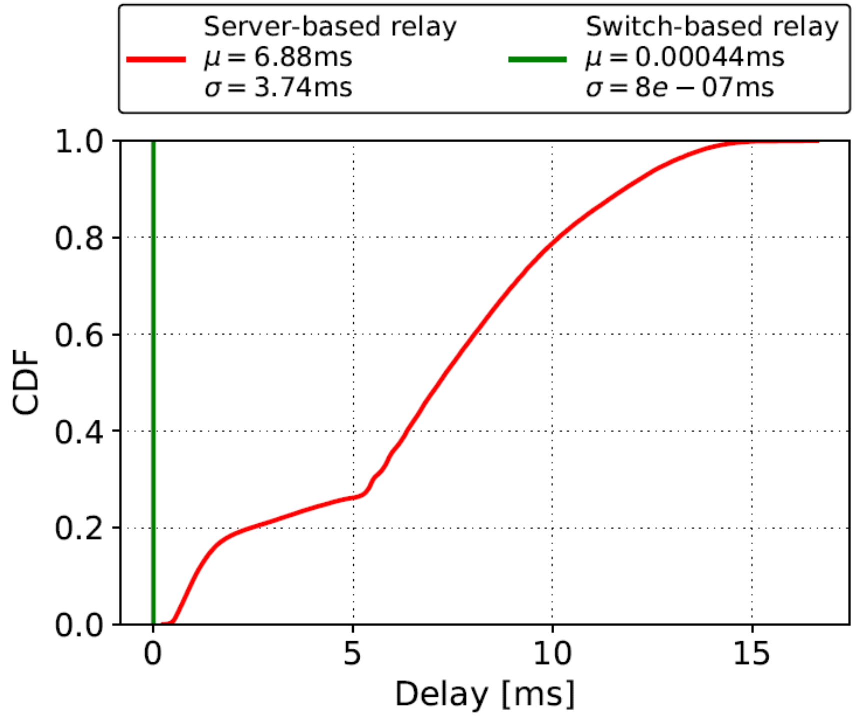
Implementation and Evaluation

- Two scenarios are considered:
  - "Server-based relay": relay server is used to relay media between end devices
  - "Switch-based relay": the switch is used to relay media
- UAC (SIPp) generates 900 media sessions, 30 per second
- The test lasts for 300 seconds
- G.711 media encoding codec (160 bytes every 20ms)



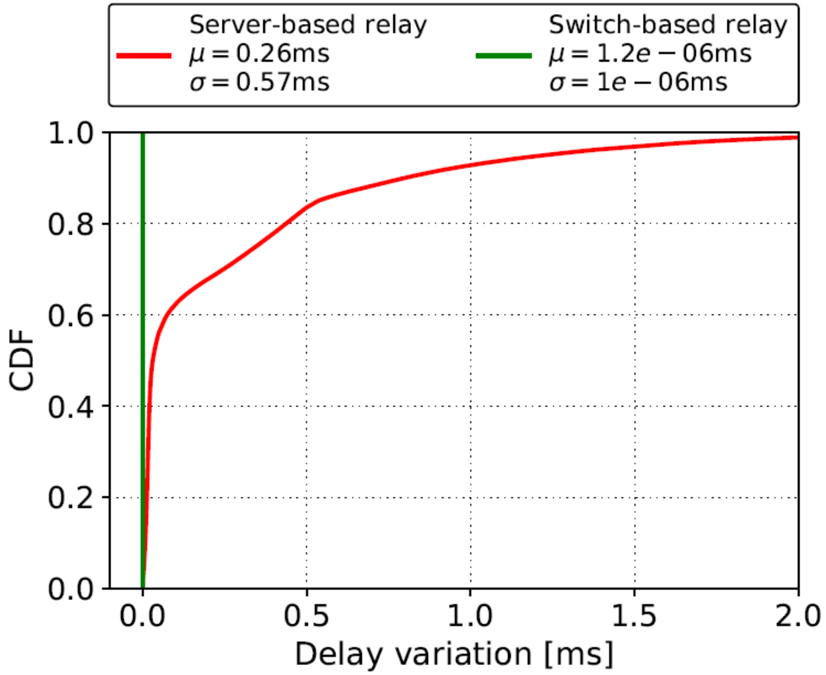
Results

- Delay: time interval starting when a packet is received from the UAC by the switch's ingress port and ending when the packet is forwarded by the switch's egress port to the UAS
  - Delay contributions of the switch and the relay server



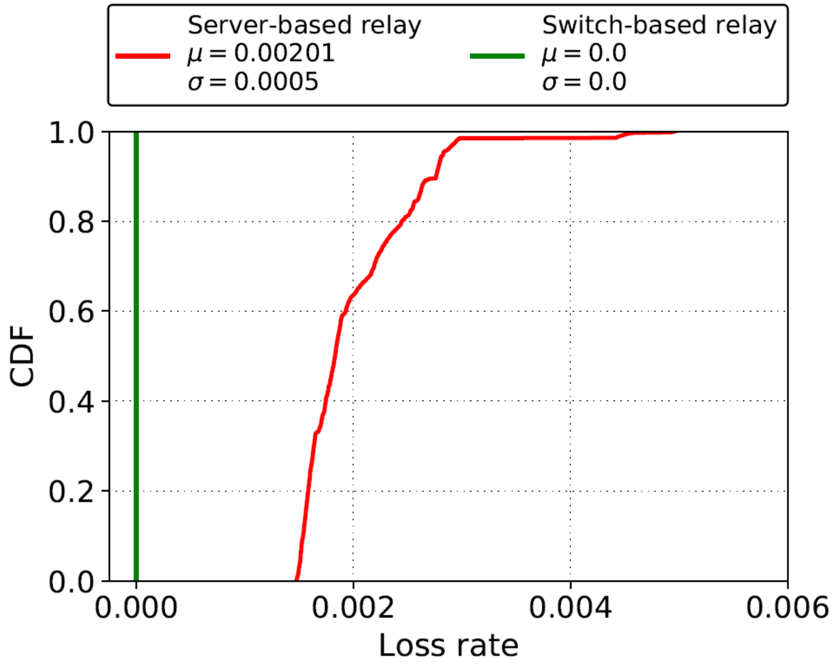
Results

- Delay variation: the absolute value of the difference between the delay of two consecutive packets
  - Analogous to jitter, as defined by RFC 4689



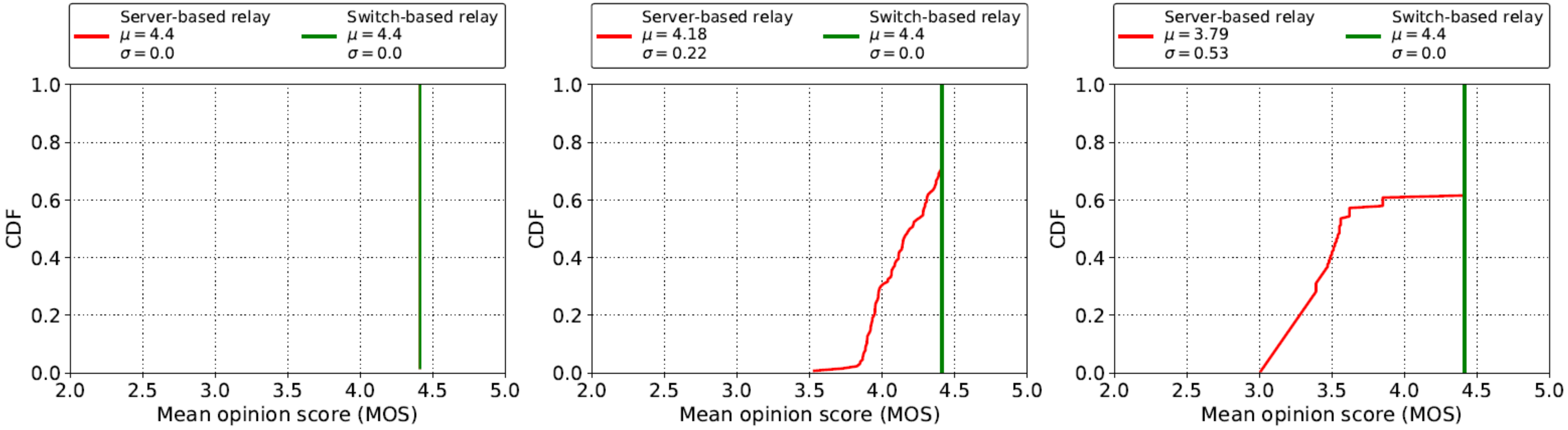
Results

- Loss rate: number of packets that fail to reach the destination
    - Calculation is based on the sequence number of the RTP header



Results

- Mean Opinion Score (MOS): estimation of the quality of the media session
  - ➢ A reference quality indicator standardized by ITU-T
  - ➢ Maximum for G.711 is ~4.4



(a) 750 simultaneous sessions. (b) 1500 simultaneous sessions. (c) 1800 simultaneous sessions.

Lessons Learned

- Advantages of offloading relay application to the data plane:
  - Performance: ~1,000,000 sessions vs ~1,000 sessions per core
  - Optimal QoS parameters: delay, delay variation, packet loss rate
- Limited resources
- Avoid complex application logic