An Overview of P4 Programmable Switches and Applications

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

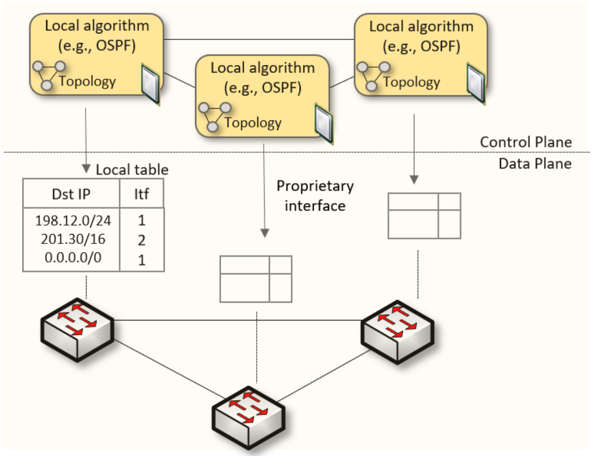
EECE 797 Graduate Webinar
American University of Beirut
Tuesday March 1st, 2022

Agenda

- Motivation
- Overview of P4 programmable switches
- Application examples
  - Offloading an application to the data plane
  - Router’s buffer sizing in real time
- PhD opportunities at the University of South Carolina (USC)

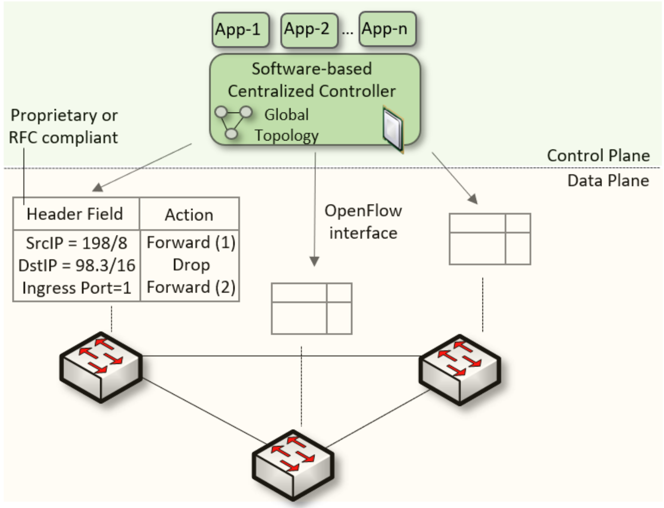
Traditional (Legacy) Networking

- 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 network owners
  - ➢ A router is a monolithic unit built and internally accessed by the manufacturer only



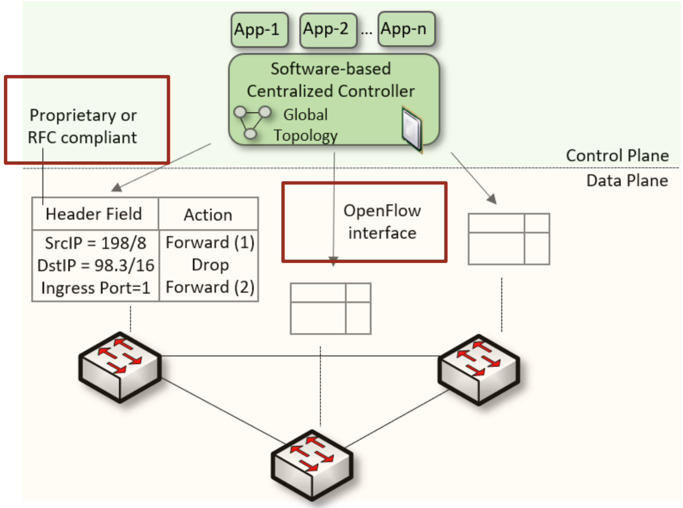
SDN

- Protocol ossification has been challenged first by SDN
- SDN explicitly separates the control and data planes, and implements the control plane intelligence as a software outside the switches
- The function of populating the forwarding table is now performed by the controller



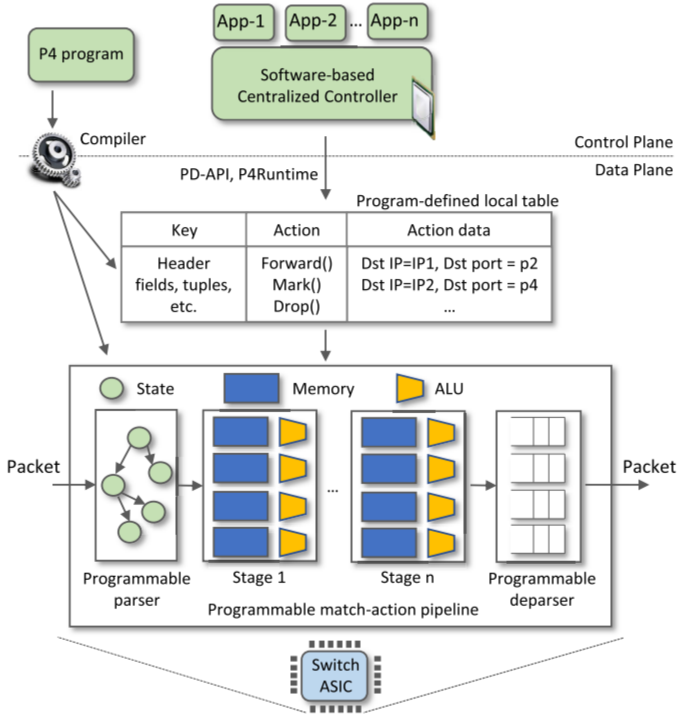
SDN Limitation

- 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



P4 Programmable Switches

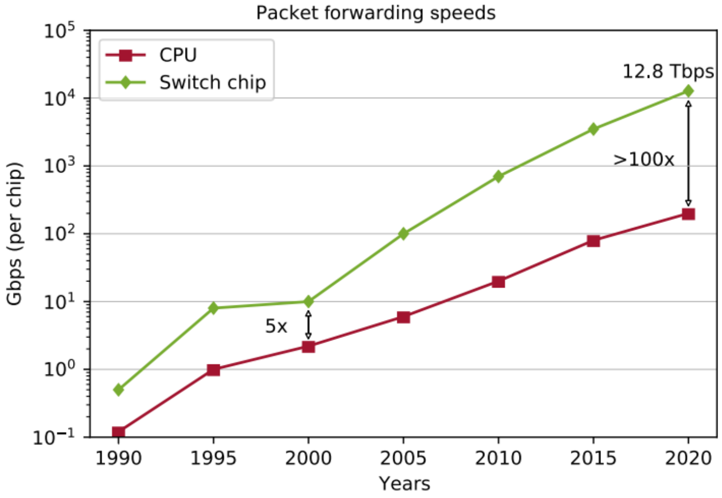
- P4[1] programmable switches permit a programmer to program the data plane
  - ➢ Define and parse new protocols
  - ➢ Customize packet processing functions
  - ➢ Measure events occurring in the data plane with high precision
  - ➢ Offload applications to the data plane



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

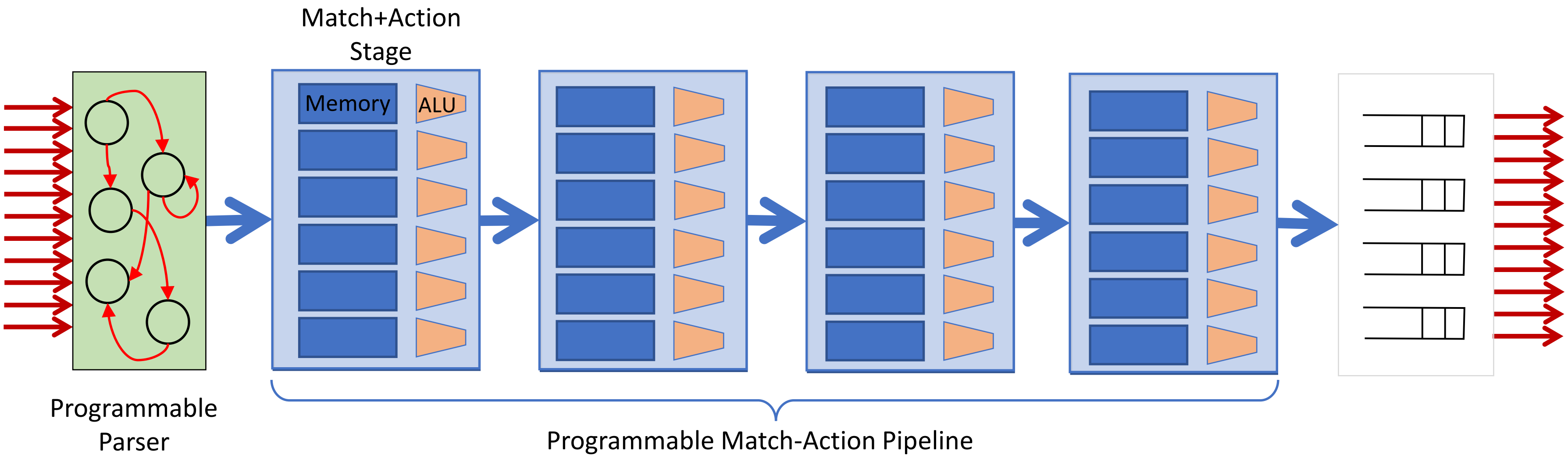
P4 Programmable Switches

- P4[1] programmable switches permit a programmer 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



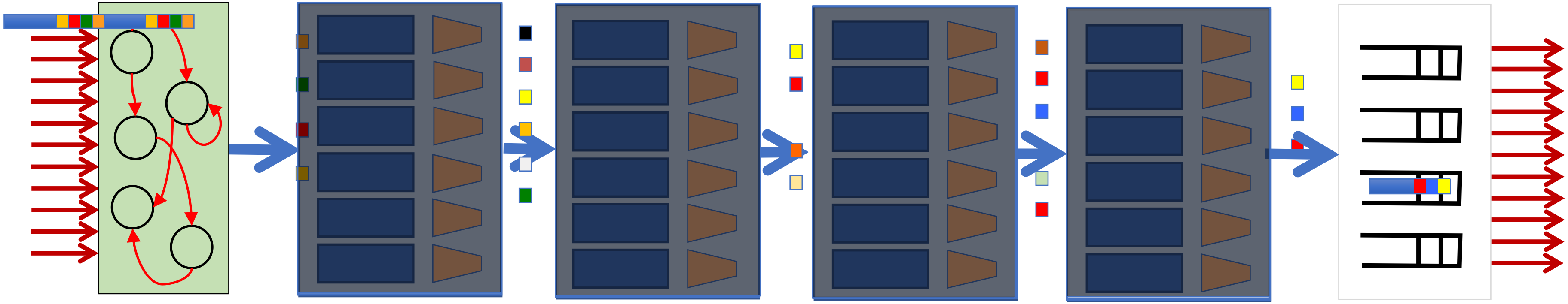
Reproduced from N. McKeown. Creating an End-to-End Programming Model for Packet Forwarding. Available: **https://www.youtube.com/watch?v=fiBuao6YZl0&t=4216s**

PISA: Protocol Independent Switch Architecture



Reproduced from N. McKeown. Creating an End-to-End Programming Model for Packet Forwarding. Available: **https://www.youtube.com/watch?v=fiBuao6YZl0&t=4216s**

PISA: Protocol Independent Switch Architecture



Reproduced from N. McKeown. Creating an End-to-End Programming Model for Packet Forwarding. Available: **https://www.youtube.com/watch?v=fiBuao6YZl0&t=4216s**

Example P4 Program

Parser Program

```
parser parse_ethernet {
   extract(ethernet);
   return switch(ethernet.ethertype) {
      0x8100 : parse_vlan_tag;
      0x0800 : parse_ipv4;
      0x8847 : parse_mpls;
      default: ingress;
}
```

Header and Data Declarations

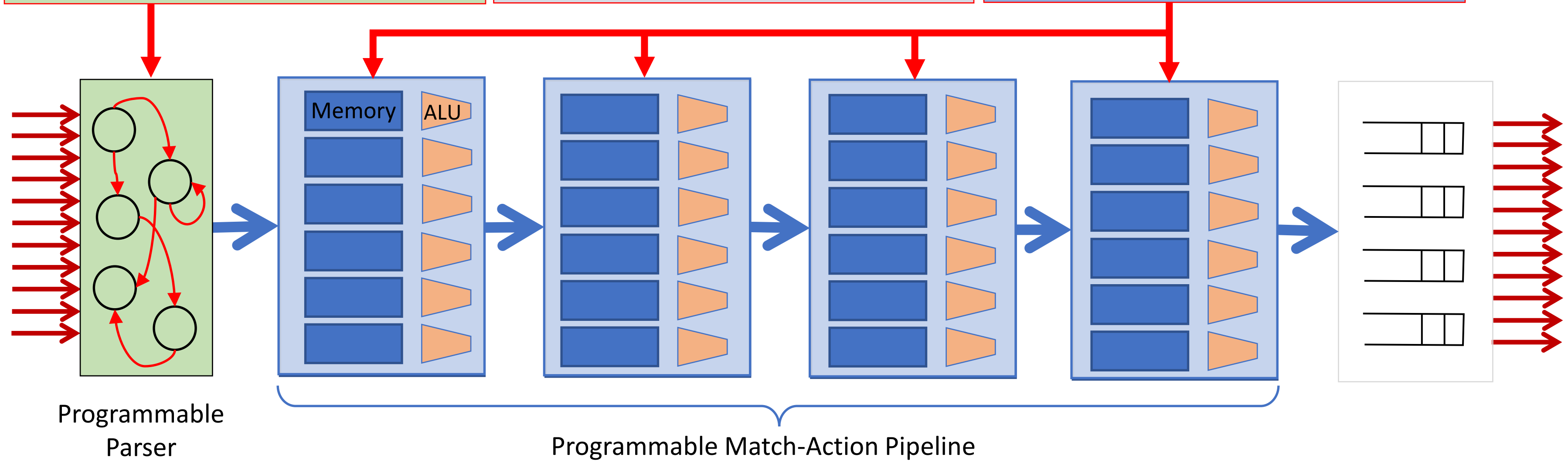
```
header_type  ethernet_t    { … }
header_type  l2_metadata_t { … }

header     ethernet_t     ethernet;
header     vlan_tag_t     vlan_tag[2];
metadata   l2_metadata_t  l2_meta;
```

Tables and Control Flow

```
table port_table { … }

control ingress {
    apply(port_table);
    if (l2_meta.vlan_tags == 0) {
        process_assign_vlan();
    }
}
```



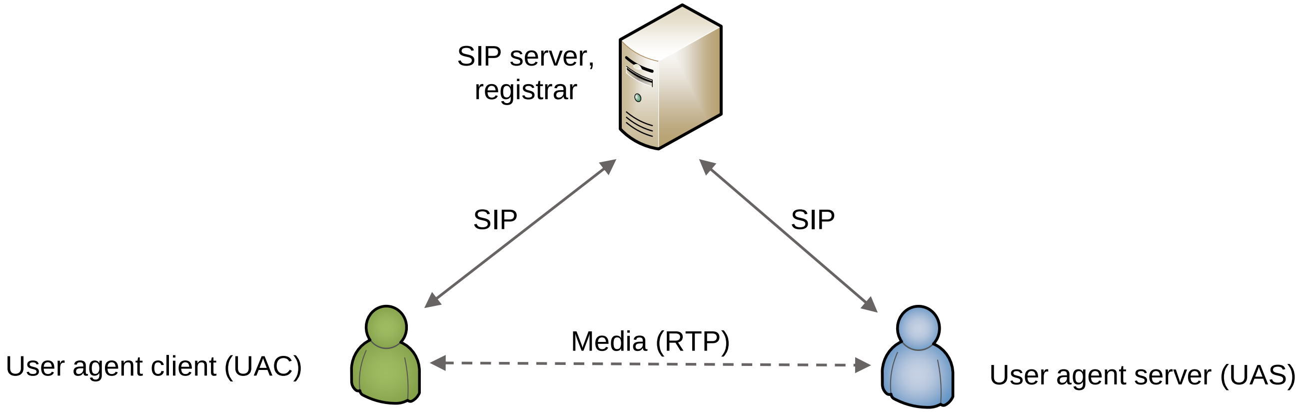
Reproduced from N. McKeown. Creating an End-to-End Programming Model for Packet Forwarding. Available: https://www.youtube.com/watch?v=fiBuao6YZl0&t=4216s

Offloading Media Traffic to P4 Programmable Data Plane Switches

E. Kfoury, J. Crichigno, E. Bou-Harb
IEEE International Conference on Communications (ICC)
June 2020

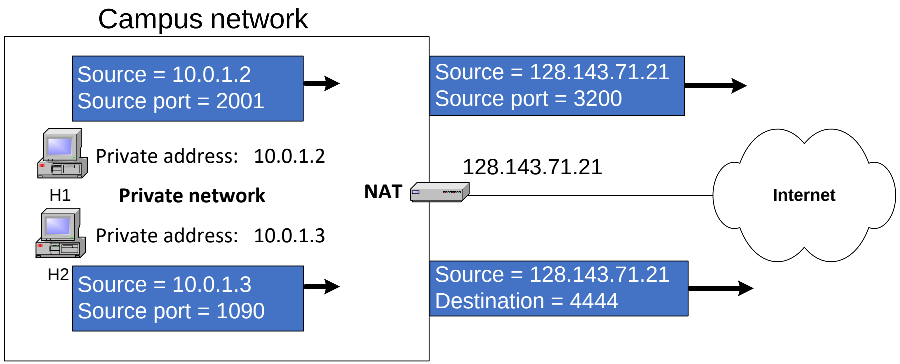
Voice and Video

- 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



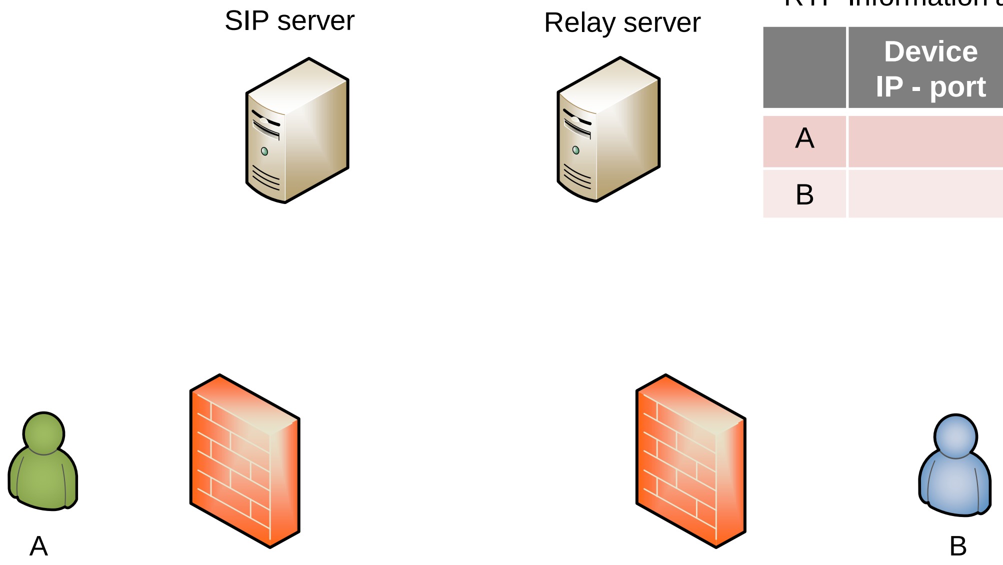
Network Address Translation (NAT)

- 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

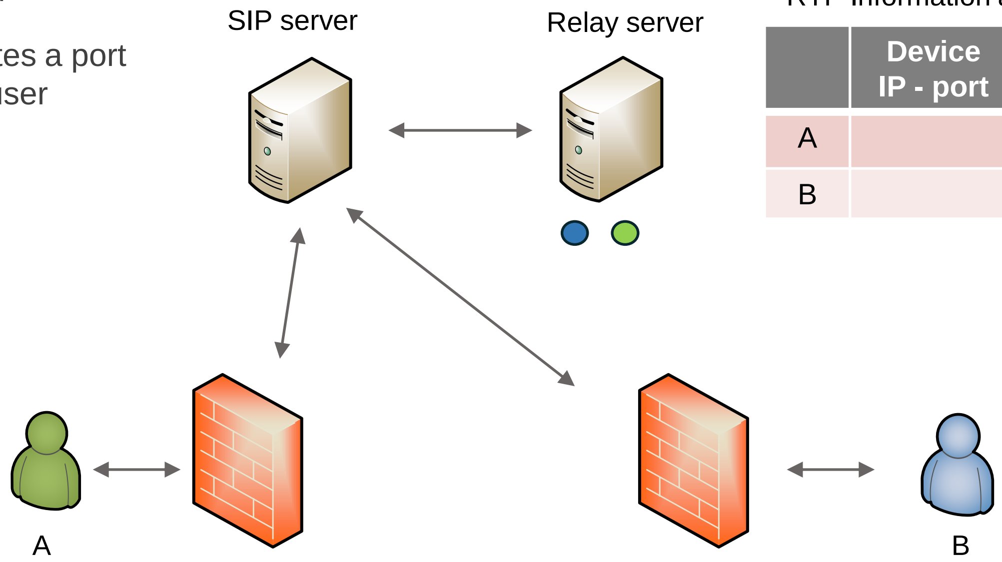


RTP Information at relay server

	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

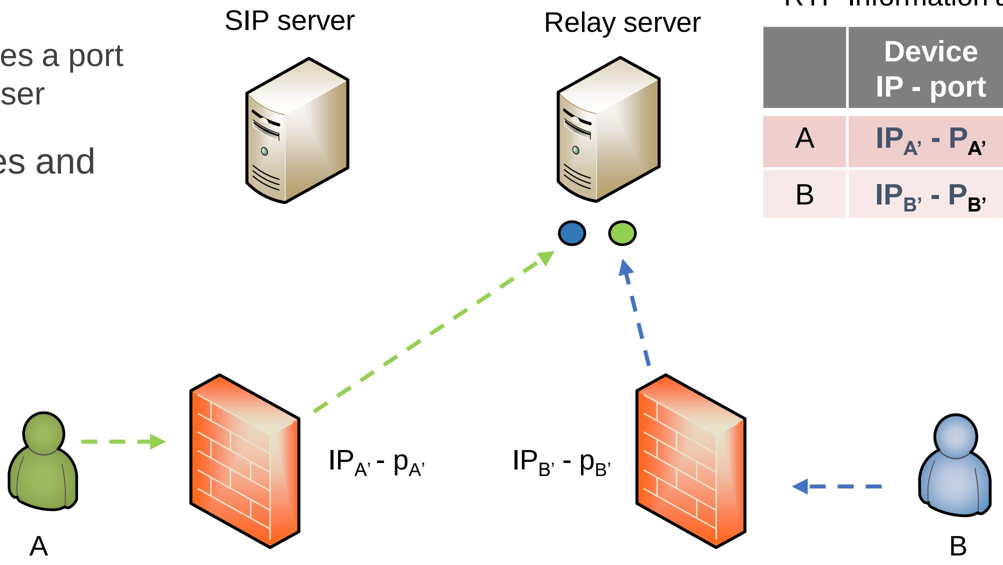


RTP Information at relay server

	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

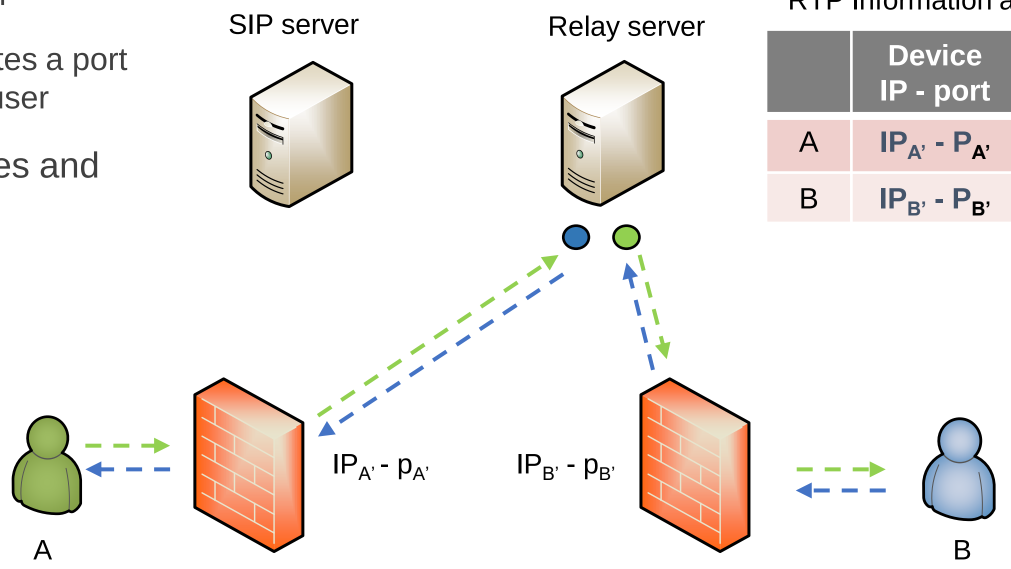


RTP Information at relay server

	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

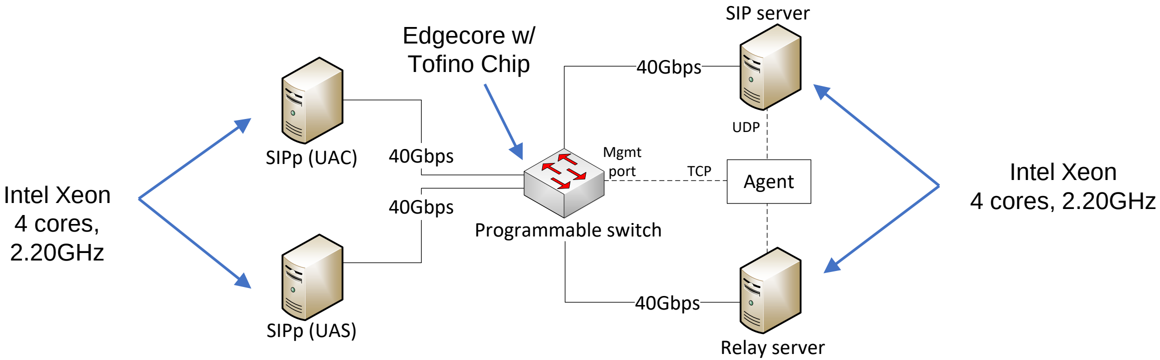


RTP Information at relay server

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

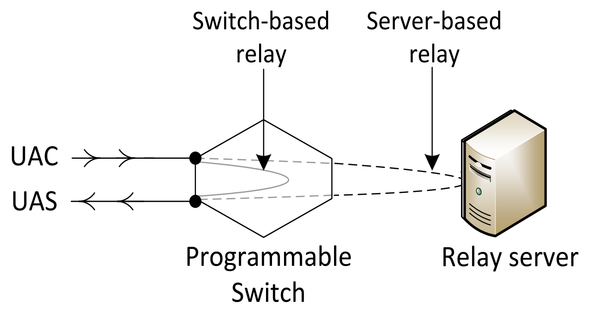
Implementation and Evaluation

- 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



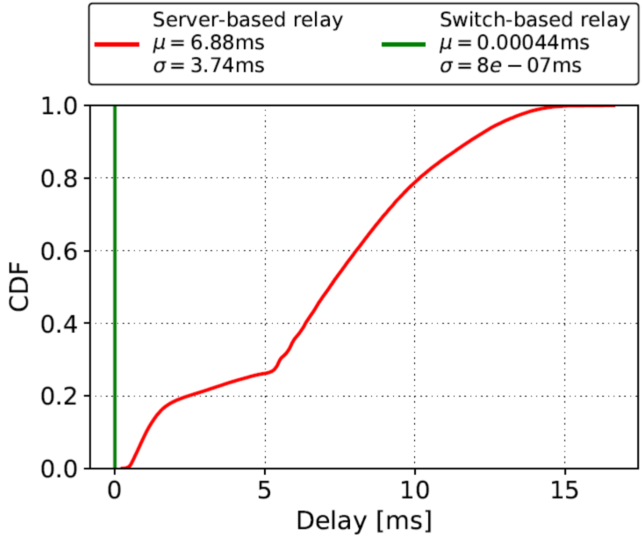
Implementation and Evaluation

- 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)



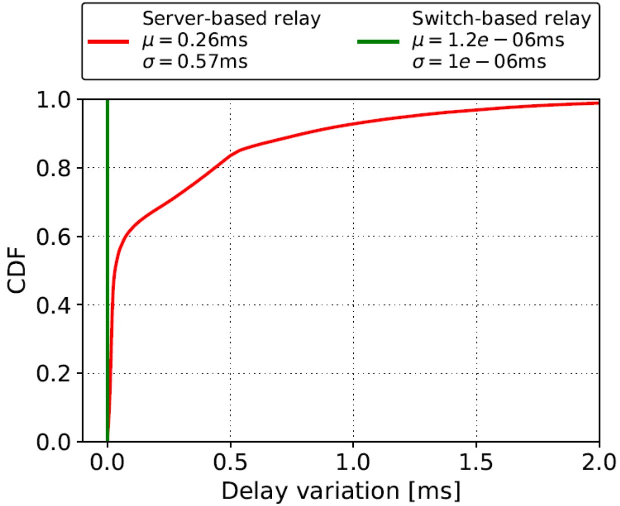
Results

- 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



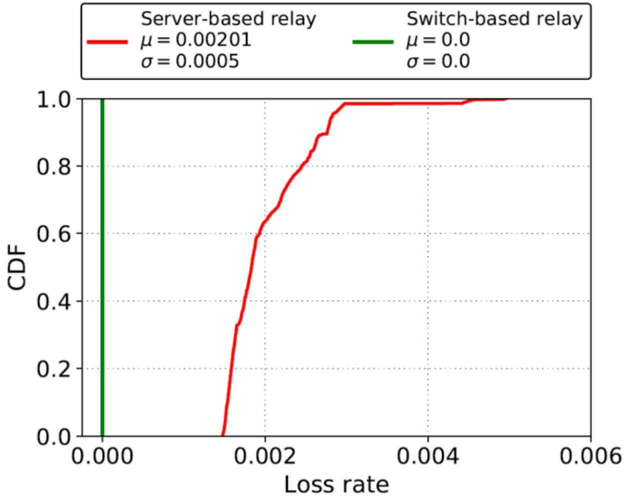
Results

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



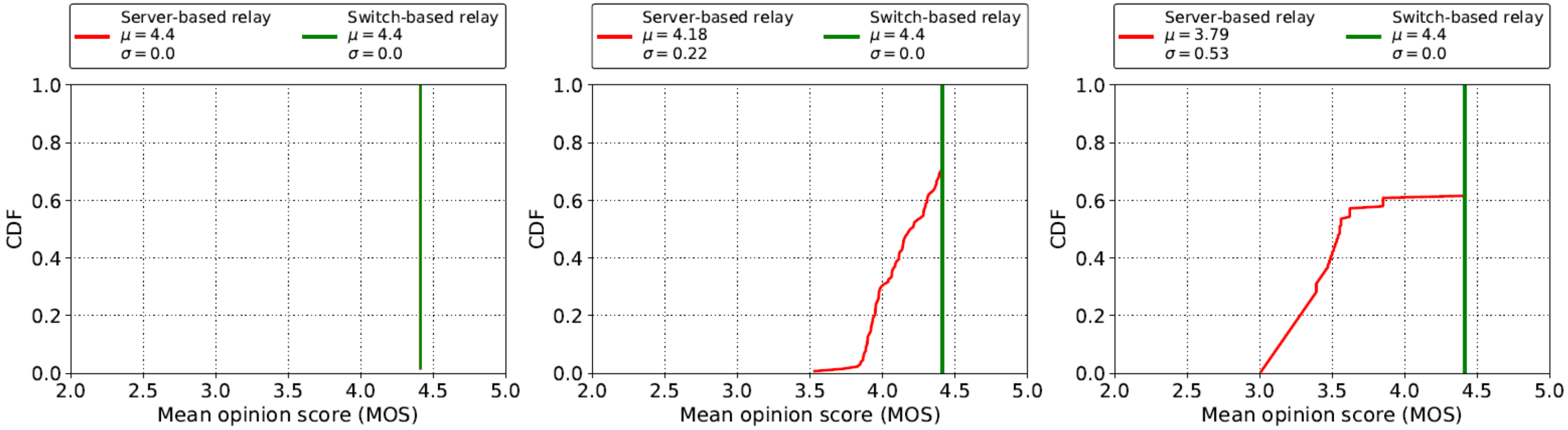
Results

- 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

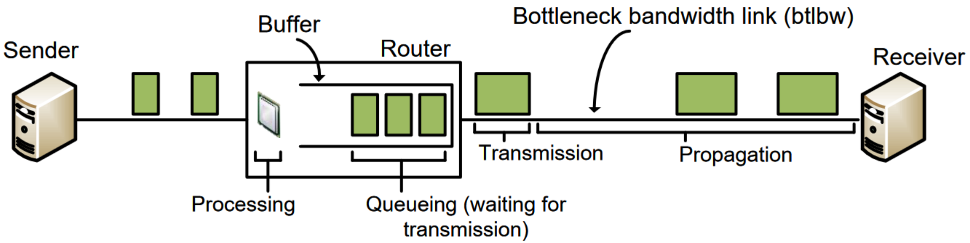
- Advantages of offloading relay application to the data plane:
  - Performance: ~1,000,000 sessions vs ~1,000 sessions per core
  - Optimal QoS parameters: delay, delay variation, packet loss rate
- Limited resources
- Avoid complex application logic

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

E. Kfoury, J. Crichigno, E. Bou-Harb, G. Srivastava
IEEE Global Communications Conference (Globecom)
December 2021, Madrid - Spain

Buffer Sizing Problem

- Routers and switches have a memory referred to as packet buffer
- The size of the buffer impacts the network performance
  - Large buffers -> excessive delays, bufferbloat
  - Small buffers -> packet drops, potential low link utilization



Buffer Sizing Rules

- General rule-of-thumb: bandwidth-delay product (older rule)
  - Buffer = $C * RTT$
  - C is the capacity of the link and RTT is the average round-trip time (RTT)
- Stanford rule
  - Buffer = $\frac{C * RTT}{\sqrt{N}}$
  - N is the number of long (persistent over time) flows traversing the link

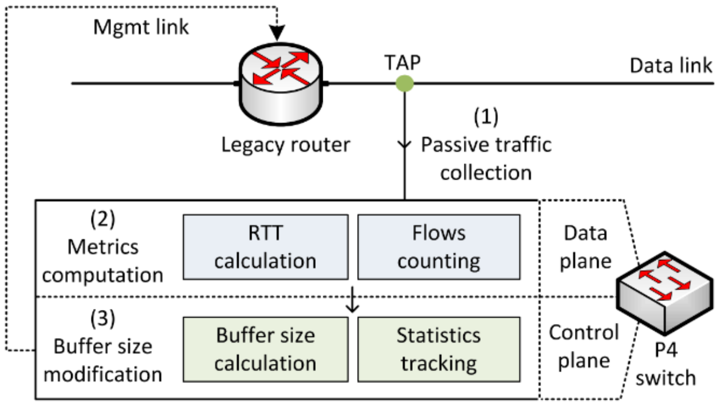
Stanford Rule Applicability

- Setting the router's buffer size to BDP/√N would require determining the current average RTT and the number of flows
- A general-purpose CPU cannot cope with high traffic rates
- Sampling techniques (e.g., NetFlow) are not accurate enough[1]

[1]Spang, Bruce, and Nick McKeown. "On estimating the number of flows." *Stanford Workshop on Buffer Sizing*. 2019.

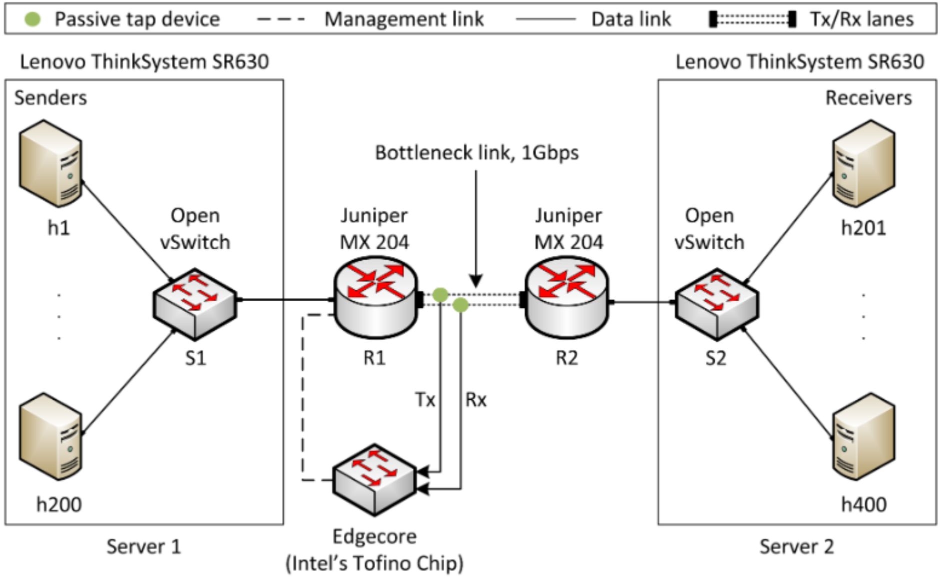
Proposed System

- Dynamically modify the buffer size of routers based on measurements collected on programmable switches
    1. Copy of the traffic is forwarded to a programmable switch by passively tapping router’s ports
    2. The programmable switch identifies, tracks, and computes the RTT of long flows
    3. The programmable switch modifies the legacy router’s buffer size



Implementation and Evaluation

- Different congestion control algorithms[1]
- iPerf3
- Default buffer size of the router is 200ms[2]



[1]Mishra et al. "The great Internet TCP congestion control census," ACM on Measurement and Analysis of Computing Systems, 2019

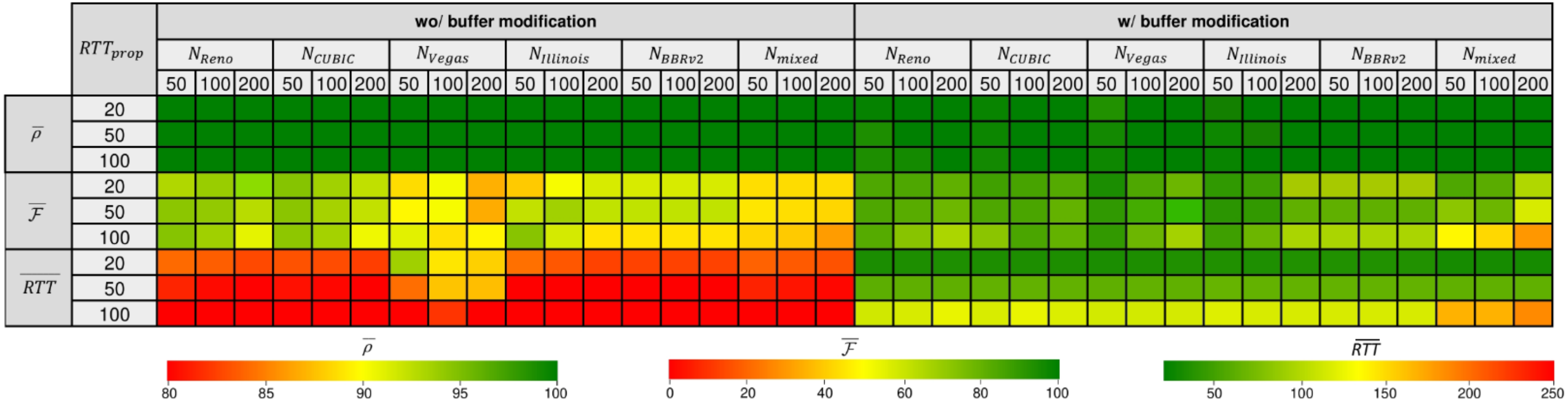
[2]N. McKeown et al. "Sizing router buffers (redux)," ACM SIGCOMM Computer Communication Review, vol. 49, no. 5

Implementation and Evaluation

- Two scenarios are considered:
  1. Default buffer size on the router, without any dynamic modification
  2. P4 switch measures and modifies the buffer size of the router

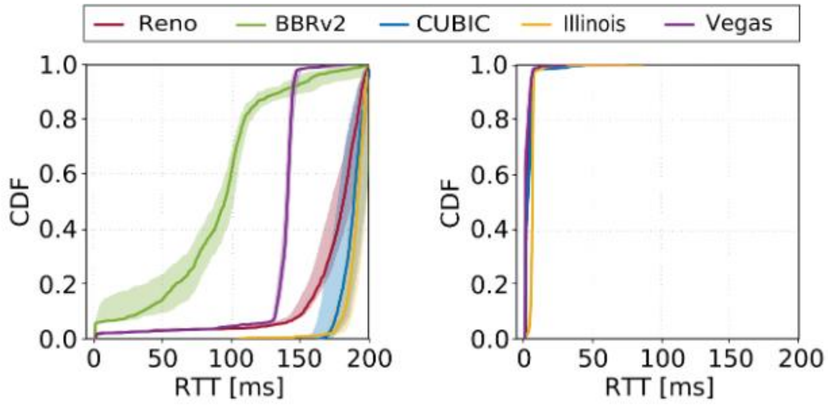
Results

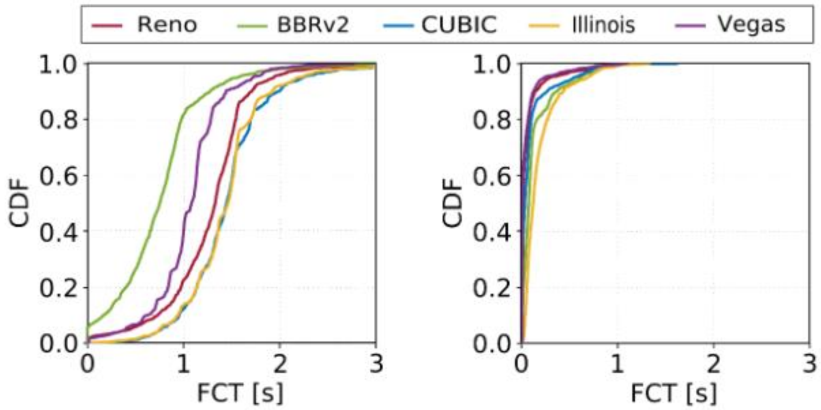
- Multiple long flows, CCAs, and propagation delays
- Average link utilization $(\overline{\rho})$
- Average fairness index $(\overline{\mathcal{F}})$
- Average RTT $(\overline{RTT})$



Results

- Performance of short flows sharing the bottleneck with long flows
- 1000 short flows are arriving according to a Poisson process
- Flow size distribution resembles a web search workload (10KB to 1MB)
- Background traffic: 200 long flows, propagation delay = 50ms





Lessons Learned

- The data plane can precisely measure flow information at line rate (e.g., RTT, number of flows)
- Measurements are used to close the control loop and modify the network (e.g., buffer size)
  - Better performance is obtained in terms of RTT, packet loss rate, fairness, FCT
- Limited resources
- Avoid complex application logic

Opportunities at the University of South Carolina

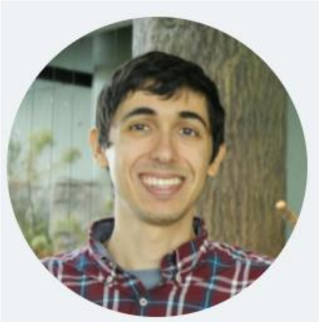
Lessons Learned

- Founded in 1801, University of South Carolina (USC) is the flagship institution of the University of South Carolina System
- More than 350 programs of study, leading to bachelor's, master's, and doctoral degrees
- Total enrollment of approximately 50,000 students, with over 33,000 on the main Columbia campus

Cyberinfrastructure Lab at USC

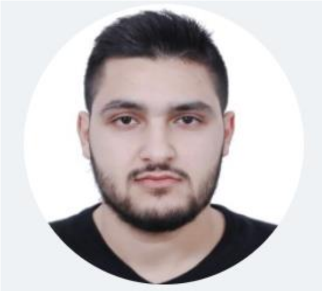
- http://ce.sc.edu/cyberinfra/
- Currently 5 PhD students, 1 Master student, 8-12 undergraduate students

Elie Kfoury



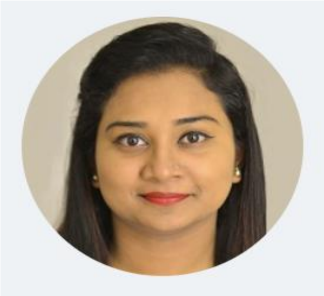
Jose Gomez

Ali AlSabeh



Ali Mazloum

Christian Vega



Shahrin Sharif

Cyberinfrastructure Lab at USC

- Students are supported via funded projects (salary, tuition, insurance)
- Typical student life
  - Two courses per semester (6 credits)
  - Work consists of 20 hours per week on funded projects / research
  - Other extra-curricular activities

Cyberinfrastructure Lab at USC

- Students are supported via funded projects (salary, tuition, insurance)
- Typical student life
  - Two courses per semester (6 credits)
  - Work consists of 20 hours per week on funded projects / research
  - Other extra-curricular activities

PhD in Informatics

- Information available at https://tinyurl.com/2pnnzpu4
- A total of 60 credit-hours beyond the bachelor's degree, or 48 credit-hours beyond the masters, is required for the Ph.D. in Informatics
- Currently there are positions available to work in the Cyberinfrastructure Lab
- Applications are accepted throughout the year
- Required documents: CV, GRE or GMAT scores (optional for admissions through Fall 2023), official transcripts, personal statement, 2 letters of recommendation

Domain-specific Processor

- Analogy between networks and other computing domains

Domain	Year	Processing Unit	Main Language/s
General computing	1971	Central Processing Unit (CPU)	C, Java, Phyton, etc.
Signal processing	1979	Digital Signal Processor (DSP)	Matlab
Graphics	1994	Graphics Processing Unit (GPU)	Open Computing Language
Machine learning	2015	Tensor Processing Unit (TPU)	Tensor Flow
Computer networks	2016	Protocol Independent Switch Architecture (PISA)	P4