GNA-G community VC - Q4 #2: December 6th, 2023

PolKA routing approach to support traffic engineering for data-intensive science

Magnos Martinello[2], ***Rafael S. Guimarães***[1], *Everson Borges*[2], *Cristina Klippel Dominicini*[1], *Diego Maffioletti*[1], *Jordi Ros-Giralt*[3], *Edgard Cunha*[2] *and Harvey Newman*[4]

[1]Federal Institute of Espírito Santo, [2]Federal University of Espírito Santo, [3]Qualcomm Europe, Inc. [4]Caltech - California Institute of Technology

Contact: rafaelg@ifes.edu.br

INSTITUTO FEDERAL Espírito Santo

CONNECT

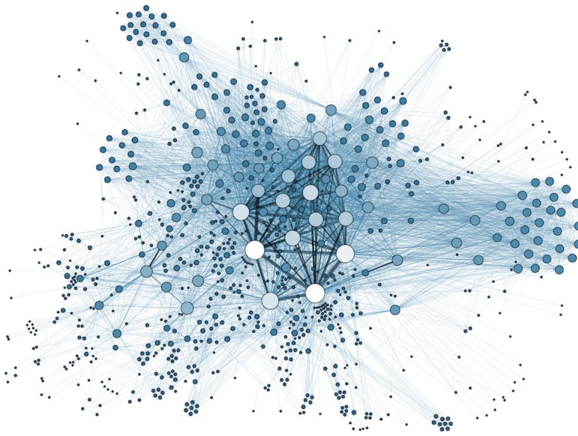
UFES

Agenda

- **Motivation**
- Proposal
- Design
- Deployment
- Demonstration
- Conclusions

Motivation

- **Data-Intensive Science (DIS) requirements** :
    - High-speed WAN networks
    - Massive data transfer & Large number of flows
    - E2E reliability and performance (traffic engineering)
    - Multiple domains



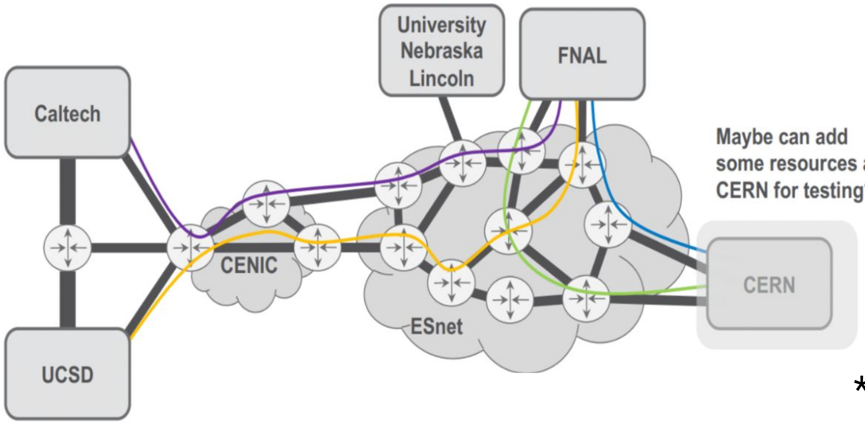
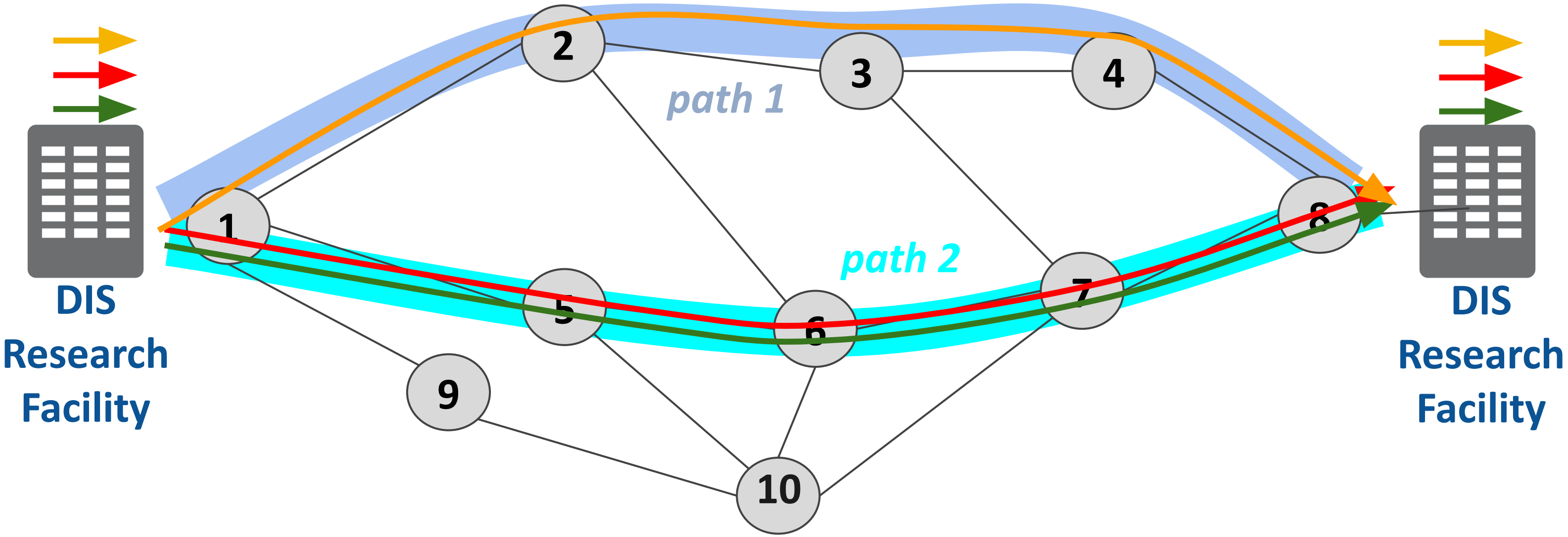


*Figure from prof. Harvey Newman

Data Intensive Science Requirements

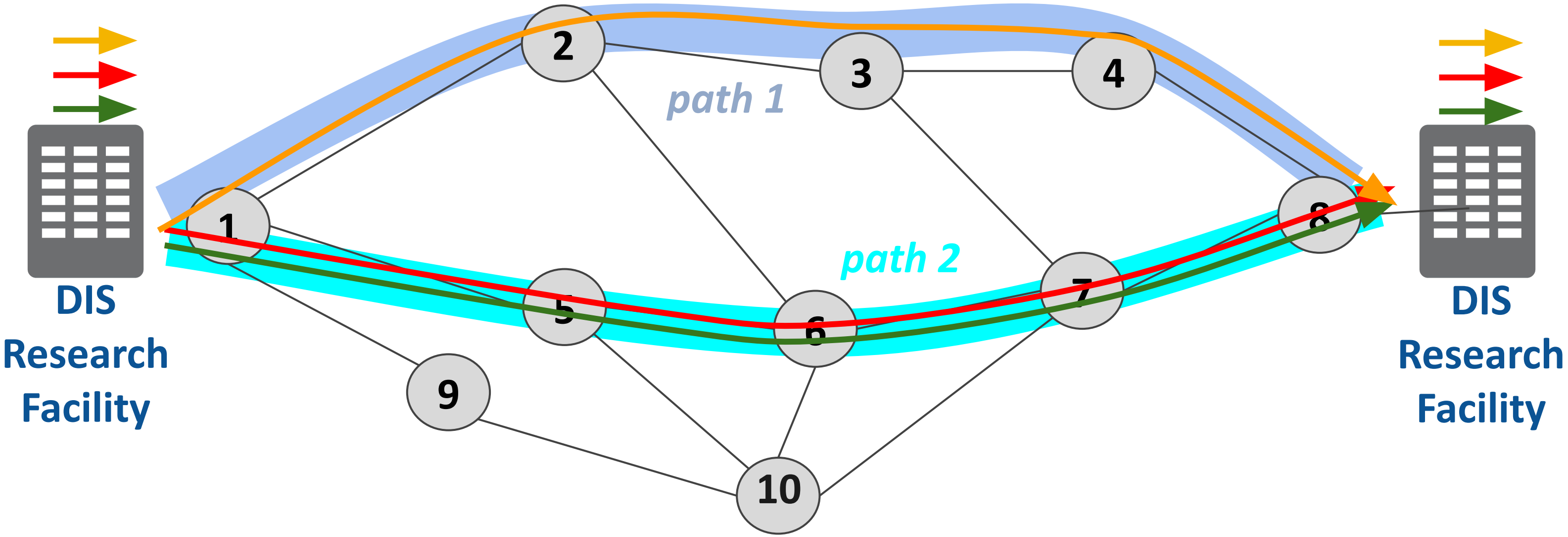
- **High Speed Networks (>= 100Gbps)**
- **Big Data Streams**
- **Multiple Flows Aggregation**



Data Intensive Science Requirements

- **High Speed Networks (>= 100gbps)**
- **Big Data Streams**
- **Multiple Flows Aggregation**

How can we dynamically configure…
… big pipes/tunnels
… in the underlay network
… to support these requirements?



Bottlenecks in traditional solutions

- **DIS requirements**:
  - High-speed WAN networks
  - Massive data transfer & Large number of flows
  - E2E reliability
  - Multiple domains
- **Table-based forwarding bottlenecks:**
  - Set of shortest paths ⟶ Traffic Engineering 🙁
  - Large number of states ⟶ Scalability 🙁
  - Latency for path configuration ⟶ Agility 🙁

Motivation

- **DIS requirements**:
  - High-speed WAN networks
  - Massive data transfer & Large number of flows
  - E2E reliability
  - Multiple domains
- **Table-based forwarding bottlenecks:**
  - Set of shortest paths ⟶ Traffic Engineering 🙁
  - Large number of states ⟶ Scalability 🙁
  - Latency for path configuration ⟶ Agility 🙁

Sub Utilization

Ossification

Endpoints with no control over paths

Bad Congestion Detection/Avoidance

Motivation

- **DIS requirements**:
  - High-speed WAN networks
  - Massive data transfer & Large number of flows
  - E2E reliability
  - Multiple domains
- **Table-based forwarding bottlenecks:**
  - Set of shortest paths → Traffic Engineering
  - Large number of states → Scalability
  - Latency for path configuration → Agility
- Alternative to tackle this: **Source Routing (SR)**
  - A source specifies a path and adds a route label to the packet header.

Subutilization

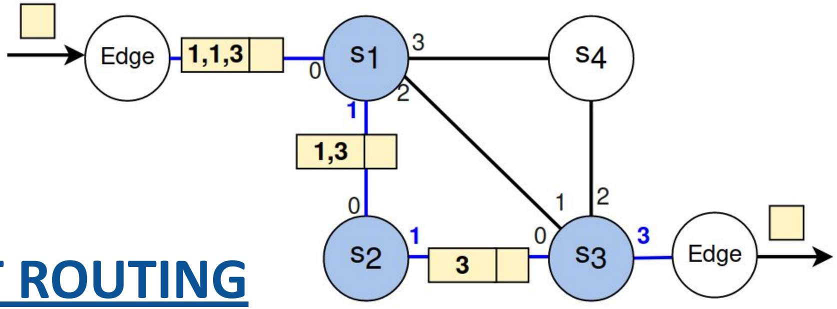
Ossification

No endpoint control over paths

Bad Congestion Detection/Avoidance

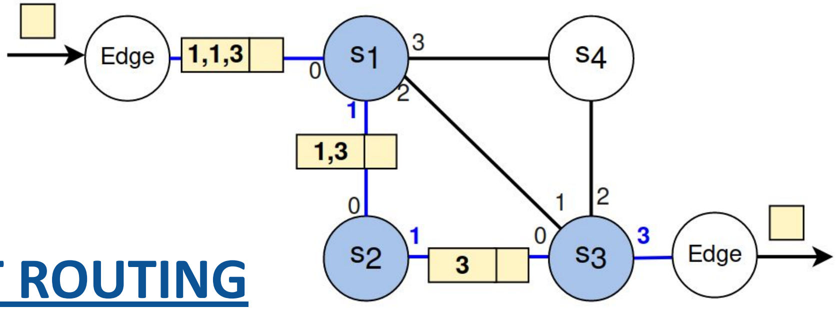
Source Routing (SR)

- Traditional way: **List-based SR (LSR)**
  - Path: a list of ports or addresses.
  - Each node performs a pop.
- Most remarkable protocol: **SEGMENT ROUTING**



Source Routing (SR)

- Traditional way: **List-based SR (LSR)**
  - Path: a list of ports or addresses.
  - Each node performs a pop.



- Most remarkable protocol: [SEGMENT ROUTING]
- **Limitations** :
  - Expensive equipment & proprietary implementations
  - Still depends on tables in the core nodes (MPLS)
  - Variable-length of headers (and big headers for both SRV4 and SRv6)
  - No multicast* *https://www.ciscolive.com/c/dam/r/ciscolive/emea/docs/2019/pdf/BRKIPM-2249.pdf*

Agenda

- Motivation
- **Proposal**
- Design
- Deployment
- Demonstration
- Conclusions

PolKA Proposal

- A Source Routing approach that meets the requirements:

open source/ interoperable	no tables in the core	support in prog. switches	fixed length header	topology agnostic multipath routing

- PolKA: Polynomial Key-based Architecture for Source Routing
  - Polynomial Residue Number System (**RNS**)
  - Chinese Remainder Theorem (**CRT**)
  - Packet forwarding based on mod operation: **remainder of division**

Agenda

- Motivation
- Proposal
- **Design**
- Deployment
- Demonstration
- Conclusions

How does **Polynomial Key-based Architecture** work?

- Three polynomials:
  - **routeID**: a route identifier calculated using the CRT.
  - **nodeID**: to identify each core node.

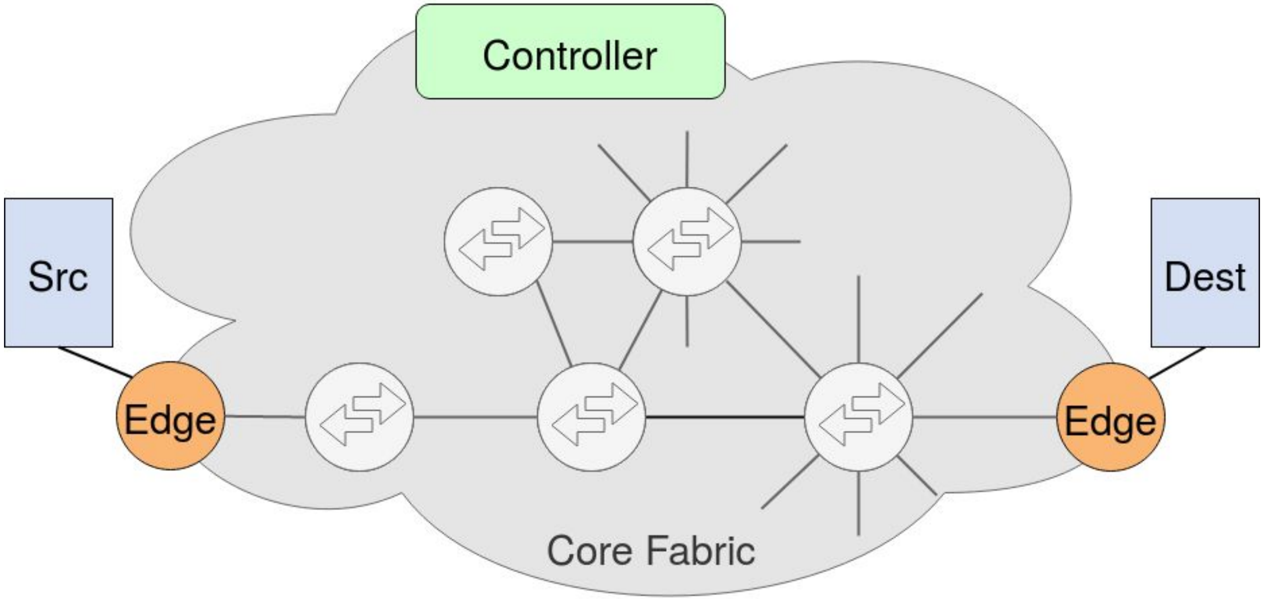
    - Irreducible polynomial which is a prime number representation in GF2
  - **portID**: to identify the port or a set of ports on each core node.

- The forwarding uses a **mod** operation (remainder of division):

$$\mathbf{portID} = \langle \mathbf{routeID} \rangle_{\mathbf{nodeID}}$$

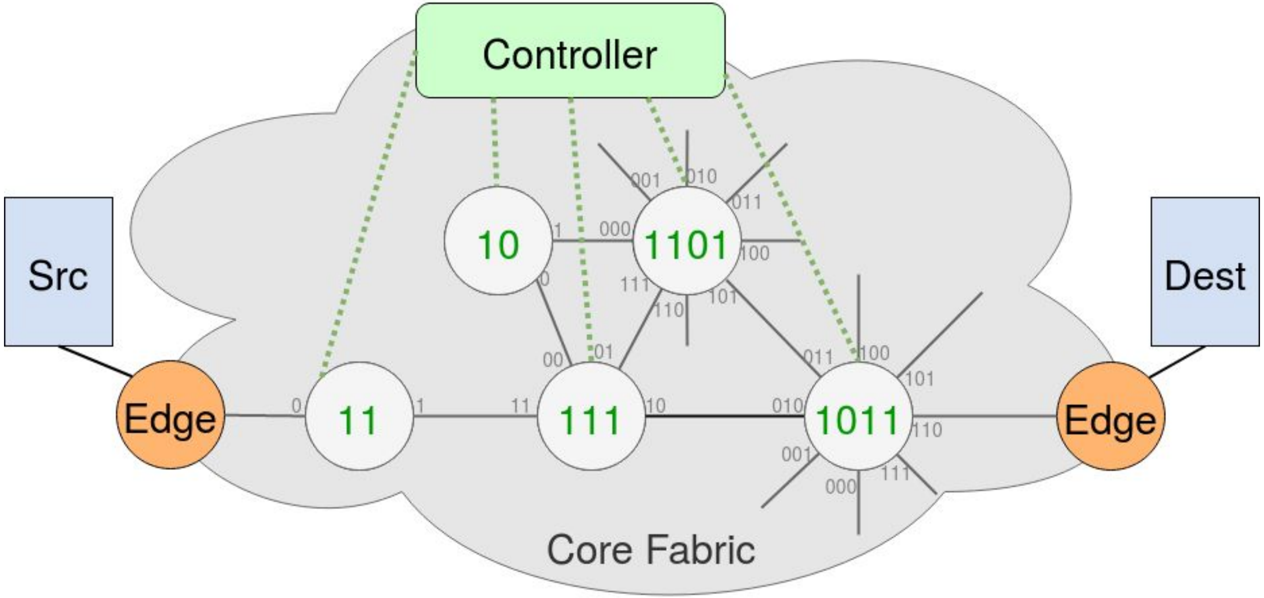
Simple example of how PolKA works

- Hosts are connected to **edge switches.**
- Edges are connected to a fabric of **core switches.**



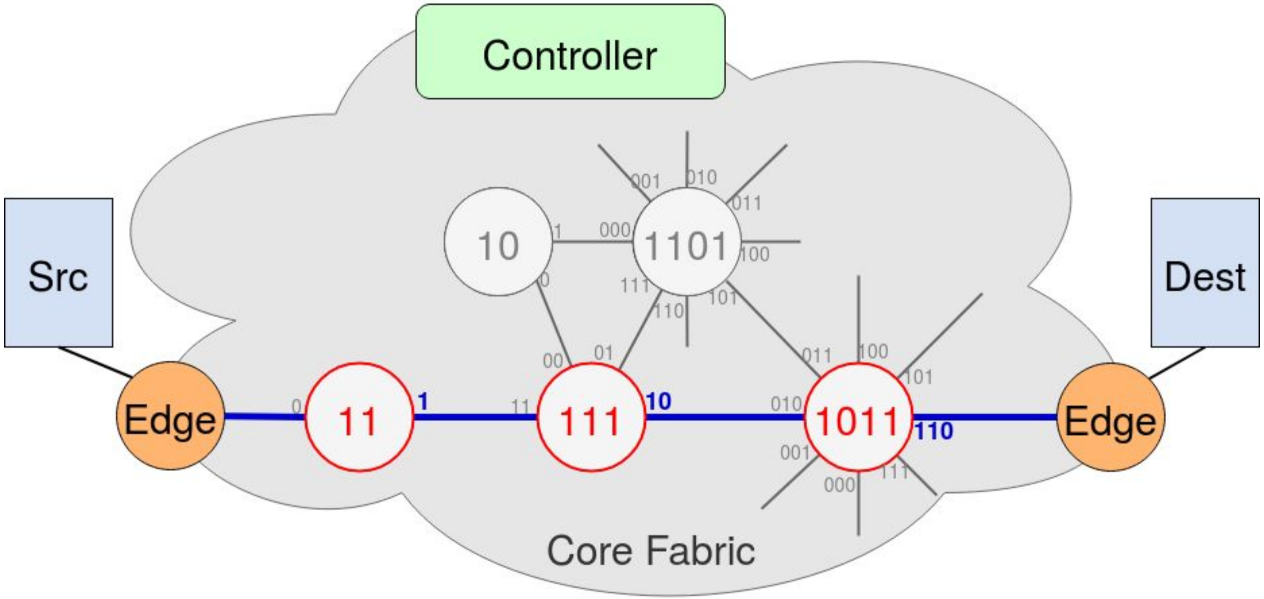
Configuration phase of PolKA network

- In a network set up phase, the **Controller** assigns irreducible polynomials to core switches (***nodeIDs***).
- Port labels are represented as binary polynomials (***portIDs***).



Selecting a path for flow assignment

- The **Controller** chooses a **path** for a specific flow (proactively or reactively):
  - A set of switches: {0011,0111,1011}
  - and their output ports: {1 , 10, 110}



Nodes and ports in their polynomial representation

- The **Controller** chooses a **path** for a specific flow:
  - A set of switches: {0011,0111,1011}
  - and their output ports: {1 , 10, 110}

nodeID polynomials

$s_1(t) = t + 1 = 11$

$s_2(t) = t^2 + t + 1 = 111$

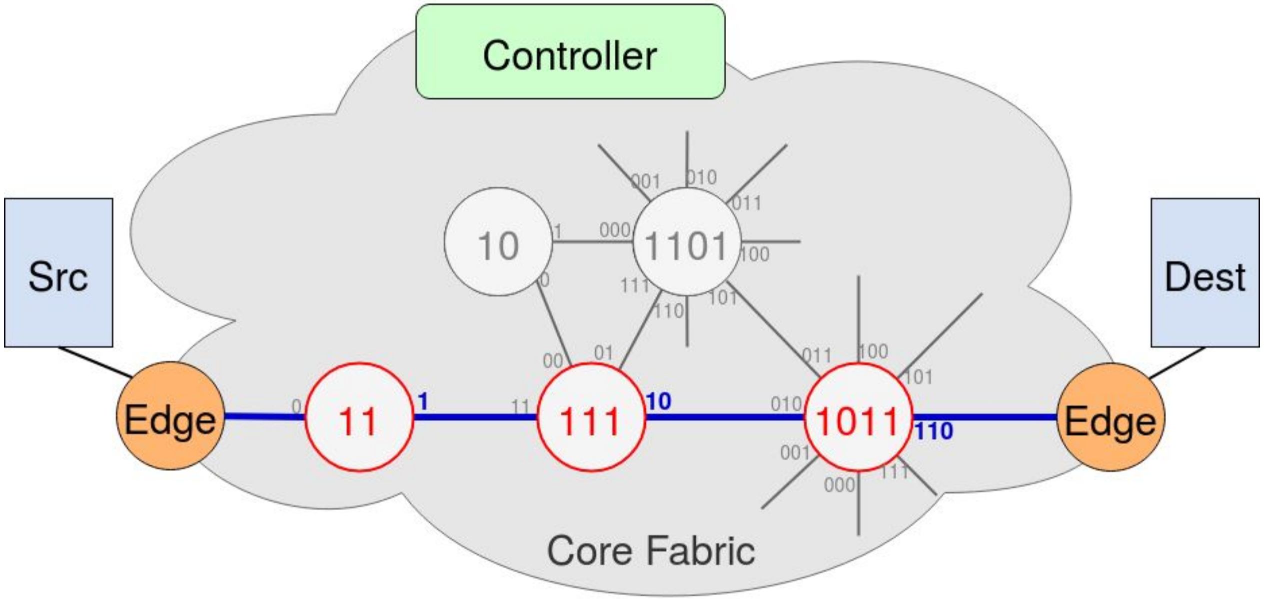
$s_3(t) = t^3 + t + 1 = 1011$

portID polynomials

$o_1(t) = 1$

$o_2(t) = t = 10$

$o_3(t) = t^2 + t = 110$



Computing the route-id with CRT

- The **Controller** calculates the ***routeID*** using CRT:
  - Complexity: $\mathcal{O}(len(M)^2)$, where $M(t) = \prod_{i=1}^{N} s_i(t)$

R = 10000

routeID

nodeID polynomials

$s_1(t) = t + 1 = 11$

$s_2(t) = t^2 + t + 1 = 111$

$s_3(t) = t^3 + t + 1 = 1011$

portID polynomials

$o_1(t) = 1$

$o_2(t) = t = 10$

$o_3(t) = t^2 + t = 110$

Calculate routeID with CRT

$t^4 \equiv 1 \mod (t+1)$

$t^4 \equiv t \mod (t^2+t+1)$

$t^4 \equiv (t^2+t) \mod (t^3+t+1)$

$t^4 = 10000$

Packet forwarding by mod operation

- The **Controller** calculates the ***routeID*** using CRT:
  - Complexity: $\mathcal{O}(len(M)^2)$, where $M(t) = \prod_{i=1}^{N} s_i(t)$

R = 10000

routeID

- Forwarding:

portID = < routeID >$_{\text{nodeID}}$

$1 = <10000>_{0011}$

$10 = <10000>_{0111}$

$110 = <10000>_{1011}$

nodeID polynomials

$$s_1(t) = t + 1 = 11$$
$$s_2(t) = t^2 + t + 1 = 111$$
$$s_3(t) = t^3 + t + 1 = 1011$$

portID polynomials

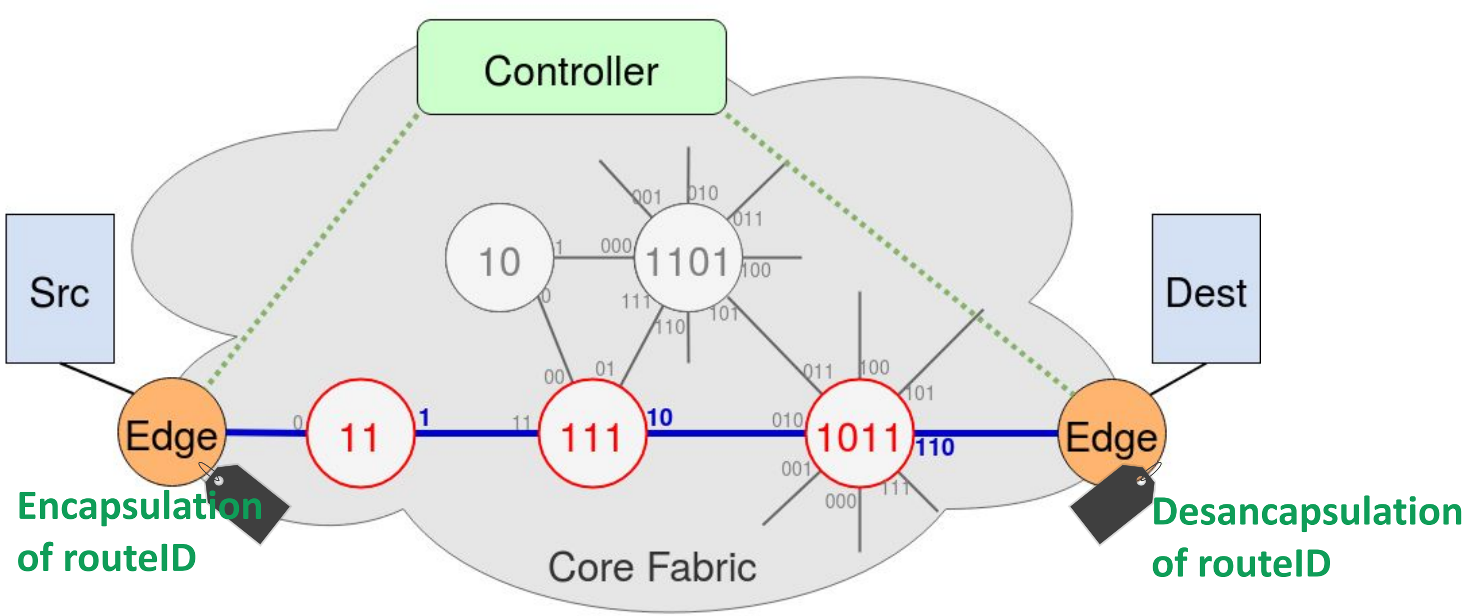
$$o_1(t) = 1$$
$$o_2(t) = t = 10$$
$$o_3(t) = t^2 + t = 110$$

Calculate routeID with CRT

$$t^4 \equiv 1 \mod (t+1)$$
$$t^4 \equiv t \mod (t^2+t+1)$$
$$t^4 \equiv (t^2+t) \mod (t^3+t+1)$$
$$t^4 = 10000$$

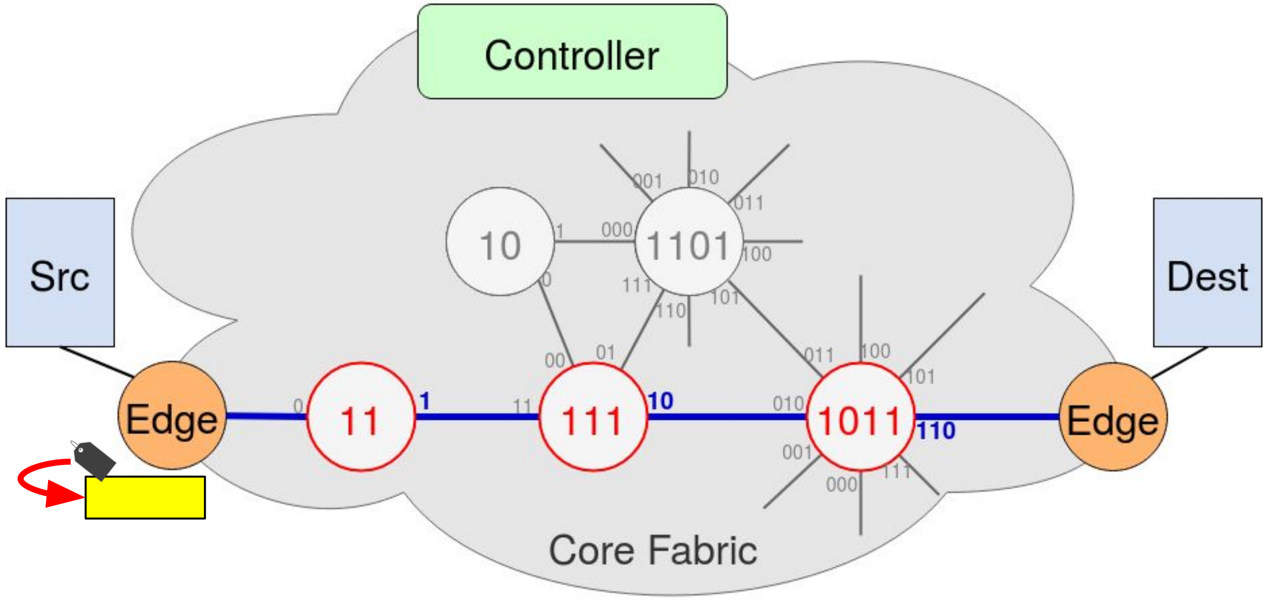
Installation of rules at the edges

- The **Controller** installs **rules** at the edges to add/remove ***routeIDs***.



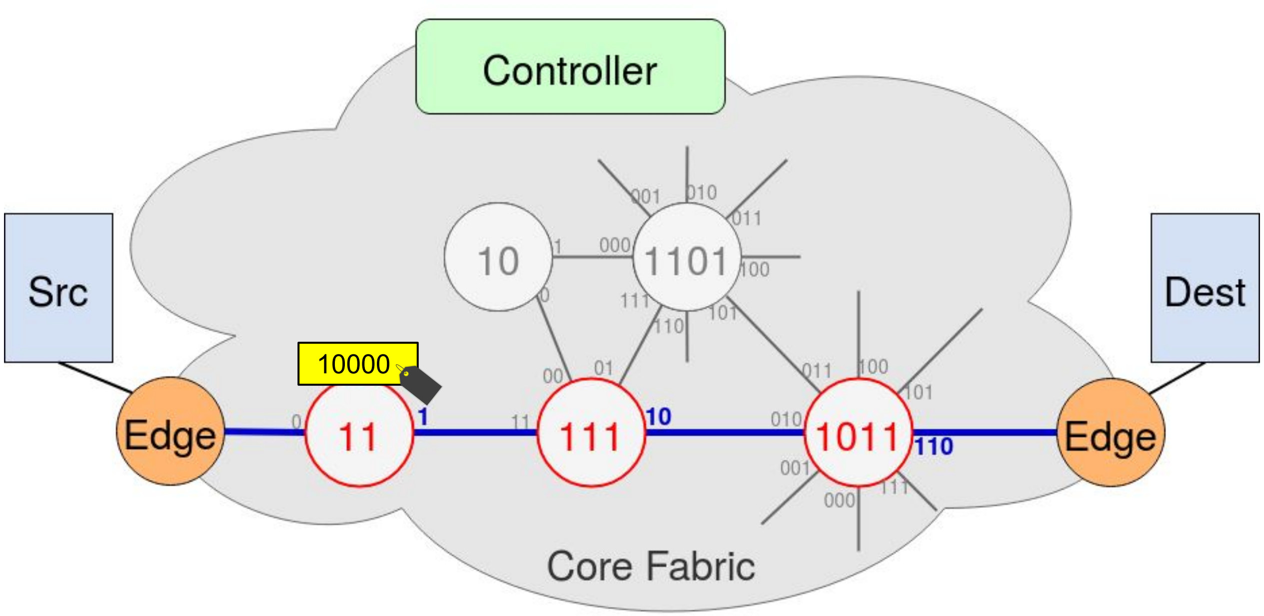
Ingress edge adds the labels

- When packets arrive, an action at ingress embeds *routeID* into the packets.



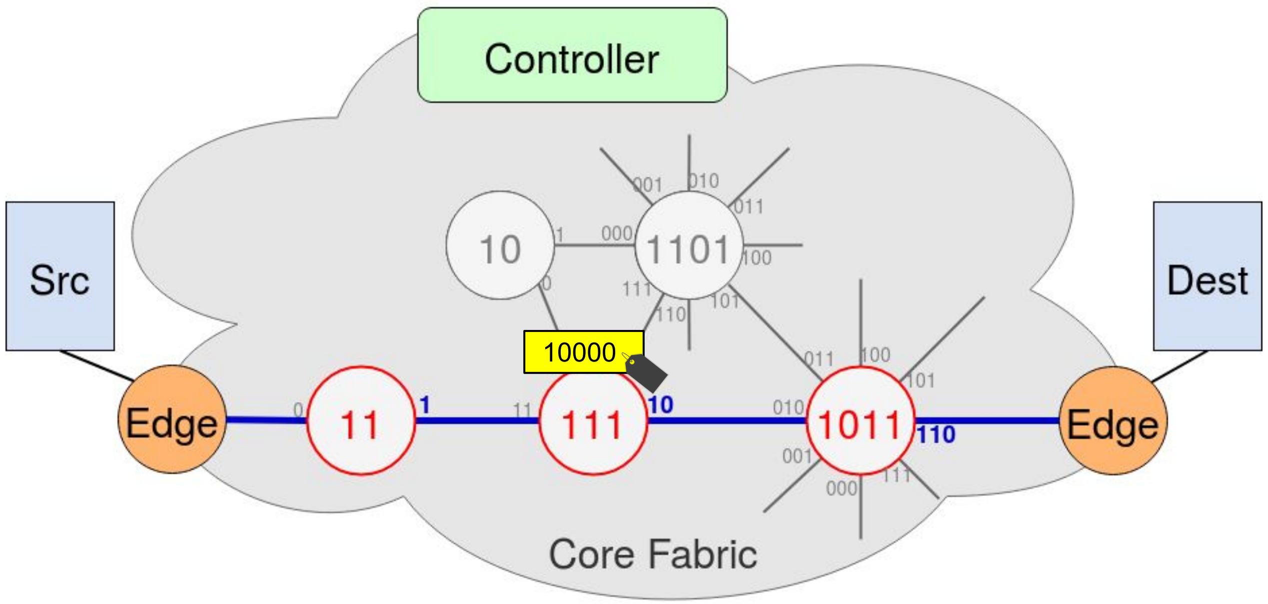
Packet forwarding at the core node

- Forwarding using **mod** operation: $\langle 10000 \rangle_{0011}$ = 1 → output port
- Stateless core nodes with no *routeID* rewrite! No tables !



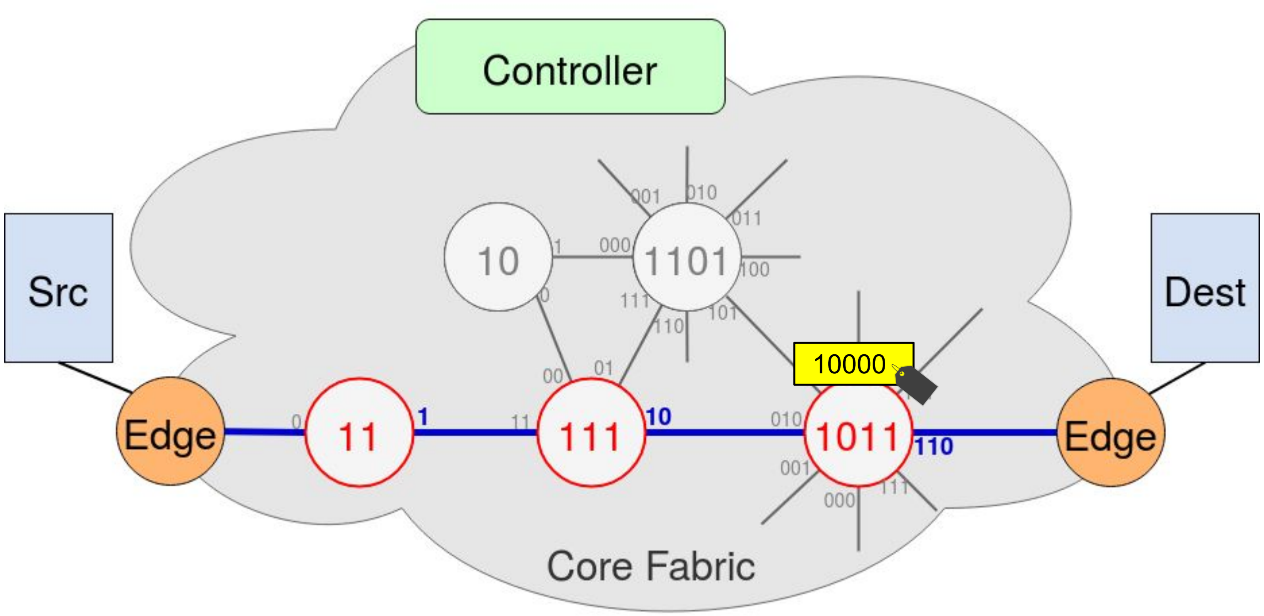
Packet forwarding at the core node

- Forwarding using **mod** operation: $\langle 10000 \rangle_{0111}$ = 10 ⟶ output port
- No *routeID* rewrite! Tableless routing at stateless core !



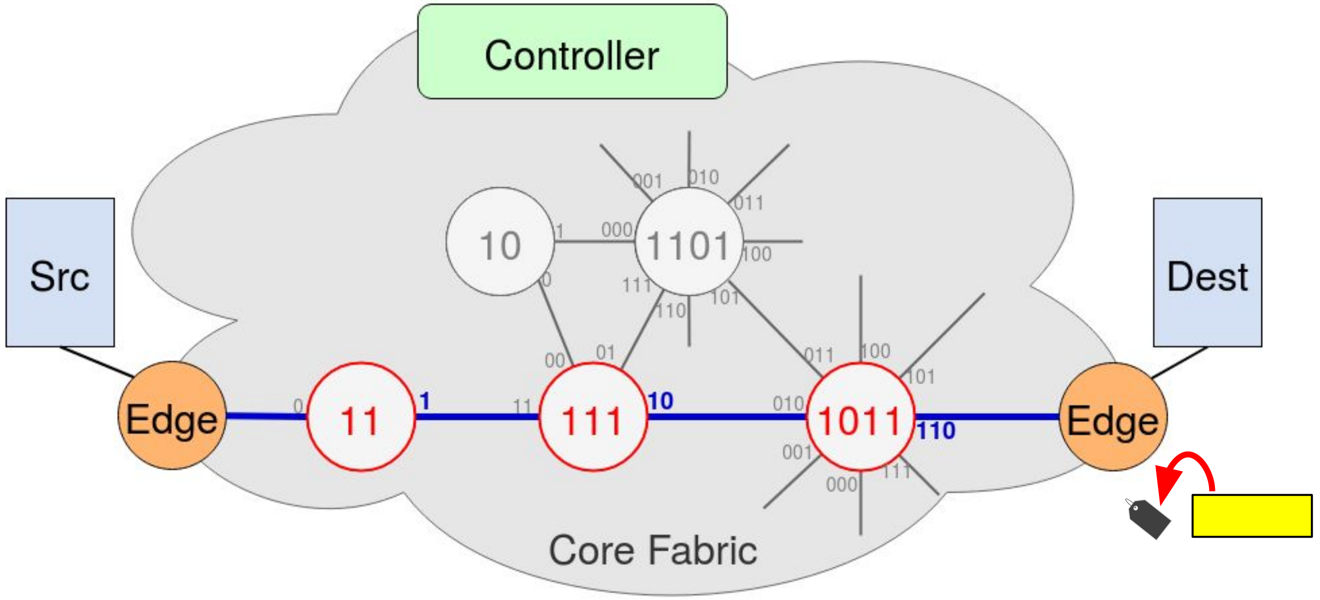
Packet forwarding at the core node

- Forwarding using **mod** operation: $\langle 10000 \rangle_{1011} = 110 \rightarrow$ output port
- No *routeID* rewrite! No tables!



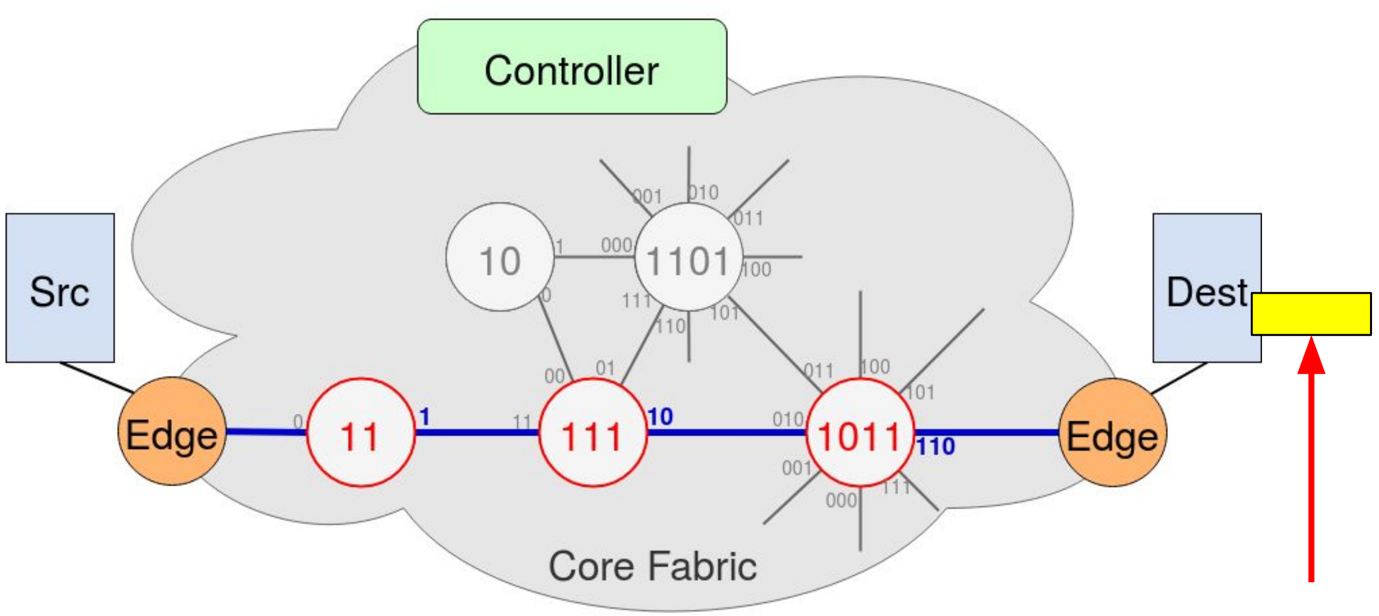
Egress edge removes the label

- Finally, an action at edge egress node removes *routeID*.



PolKA is agnostic of legacy protocols

- Packet is delivered to the application in a transparent manner.



How to implement PolKA's in high speed line rate?

- **P4 language does not natively support the mod operation.**

- **CRC hardware** (Cyclic Redundancy Check) offers polynomial mod.
  - The Tofino Native Architecture (**TNA**) supports **custom** CRC polynomials.
  - Line rate MOD computation = 2 SHIFTs + 1 CRC + 2 XORs

Agenda

- Motivation
- Proposal
- Design
- **Deployment**
- Demonstration
- Conclusions

Timeline

PolKA received the 2021 Google Research Scholar Award

M-PolKA received the Intel Connectivity Research Grant (Fast Forward Initiative)

2020 | 2021 | 2022 | 2023

2020
PolKA paper IEEE NetSoft

Routing proposal based on RNS and reuse of CRC hardware

Emulated prototype in Mininet

2021
ONDM paper Deploy @RARE

Hardware prototype in Tofino

Integration with RARE+FreeRtr

PolKA data & control plane implementation + integration

Emulated prototype in FreeRtr &

Hardware prototype in Tofino w/ FreeRtr control plane

2022
M-PolKA paper IEEE TNSM

Extension to multipath SR for reliable communications

PolKA@pangr IETF 113

Lightning Talk Path Aware Networking

PolKA@Global P4 Lab

Deployment @Caltech SDN Lab

Talk at LHC-ONE

PolKA Demo at SC-22

M-PolKA talk at ONF

2023
Proof-of-Transit paper IEEE NetSoft

PolKA Demo at SC-23

Innovative apps:

- Resilient routing with security compliance
- Inband Network Telemetry
- Optimal load balancing by G2

Innovations to be demonstrated

- Data plane
  - Source Routing with Stateless Core
  - Forwarding at line rate by **reusing CRC in P4** programmable switches

- Control plane
  - Easy to configure tunnels
  - **Integrated** in the FreeRtr platform

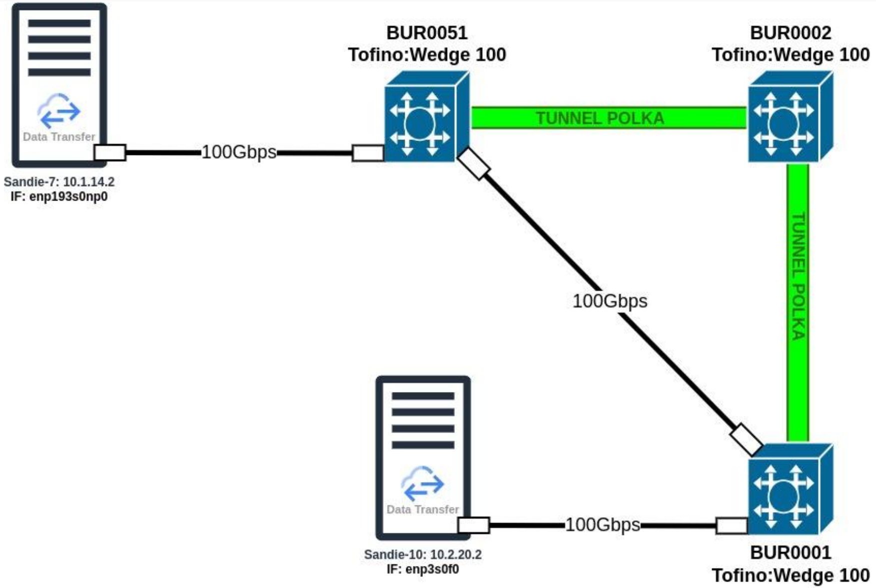
- Potential to support:
  - Transfer of big data streams with aggregation of multiple flows
  - Dynamic traffic steering configured at the edge

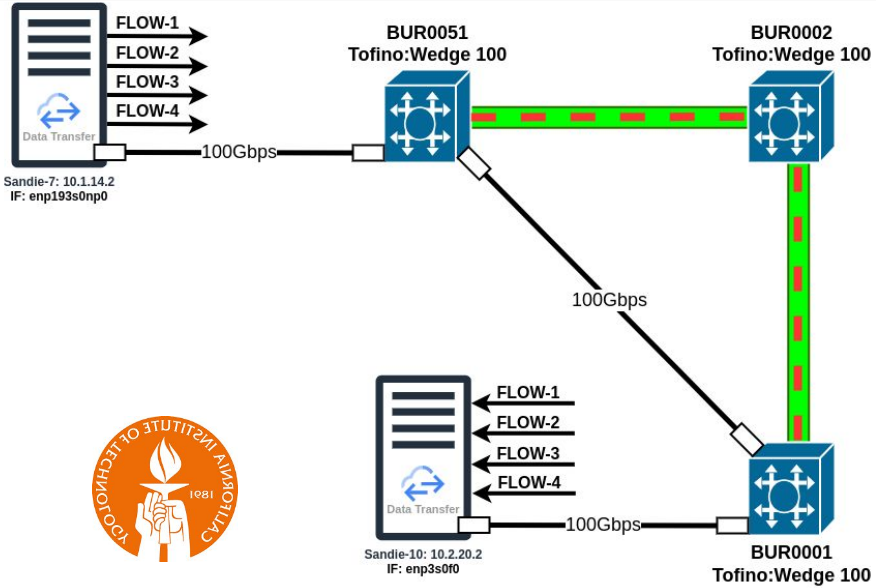
Demonstrations

- **Big data streams at 100 Gbps**
  - PolKA@ Caltech P4 lab testbed
  - Multiple aggregated TCP flows steered to pre-configured tunnels
    - A route label represents paths in the underlay network

- **Multiple big data streams to achieve more than 100 Gbps**
  - PolKA@ Caltech P4 lab testbed at the SC 23
  - Multiple aggregated TCP flows from different computers steering traffic to pre-configured tunnels
    - A route label represents paths in the underlay network

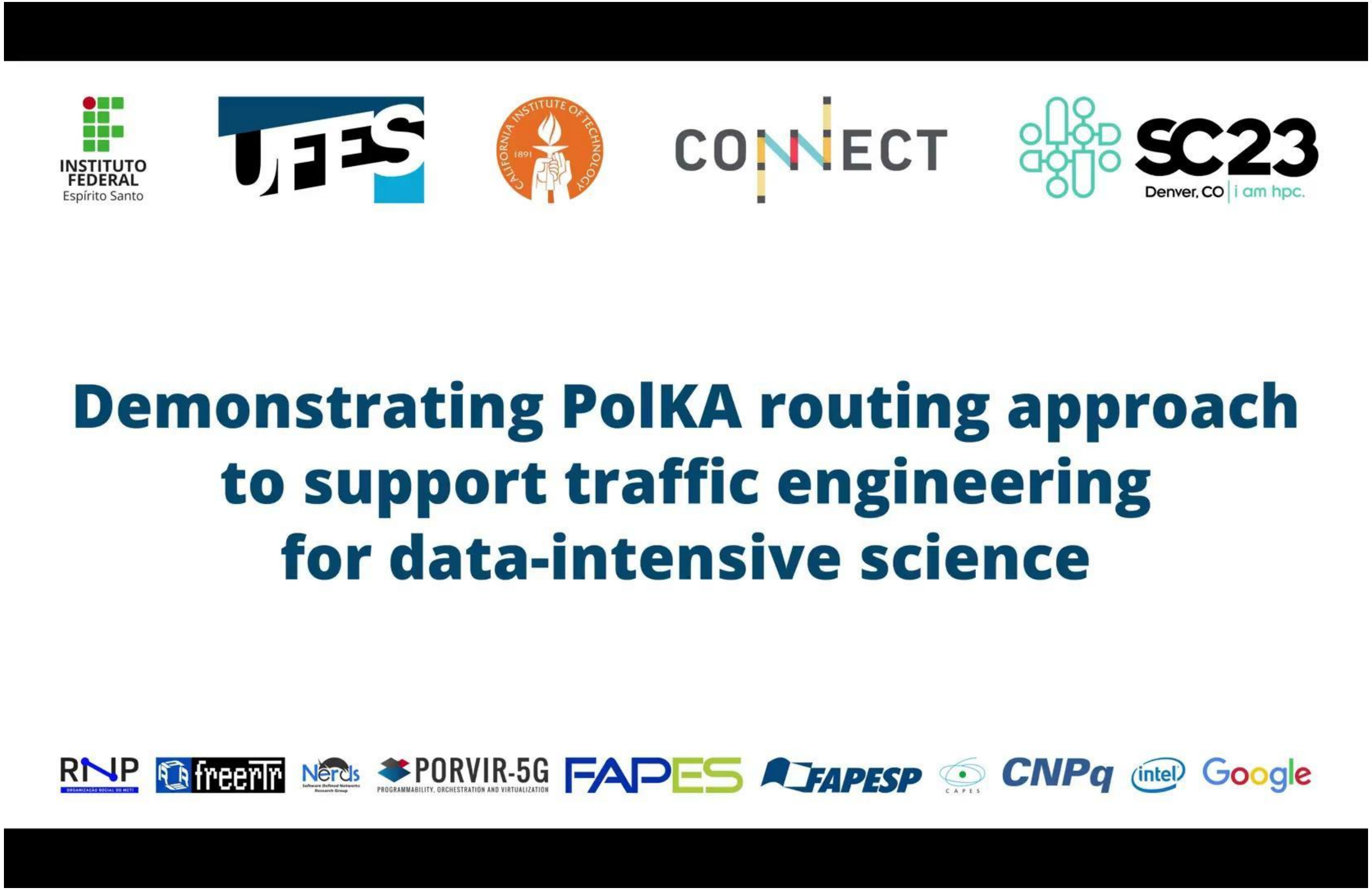
Big Data streams over PolKA tunnels at 100 Gbps in Caltech



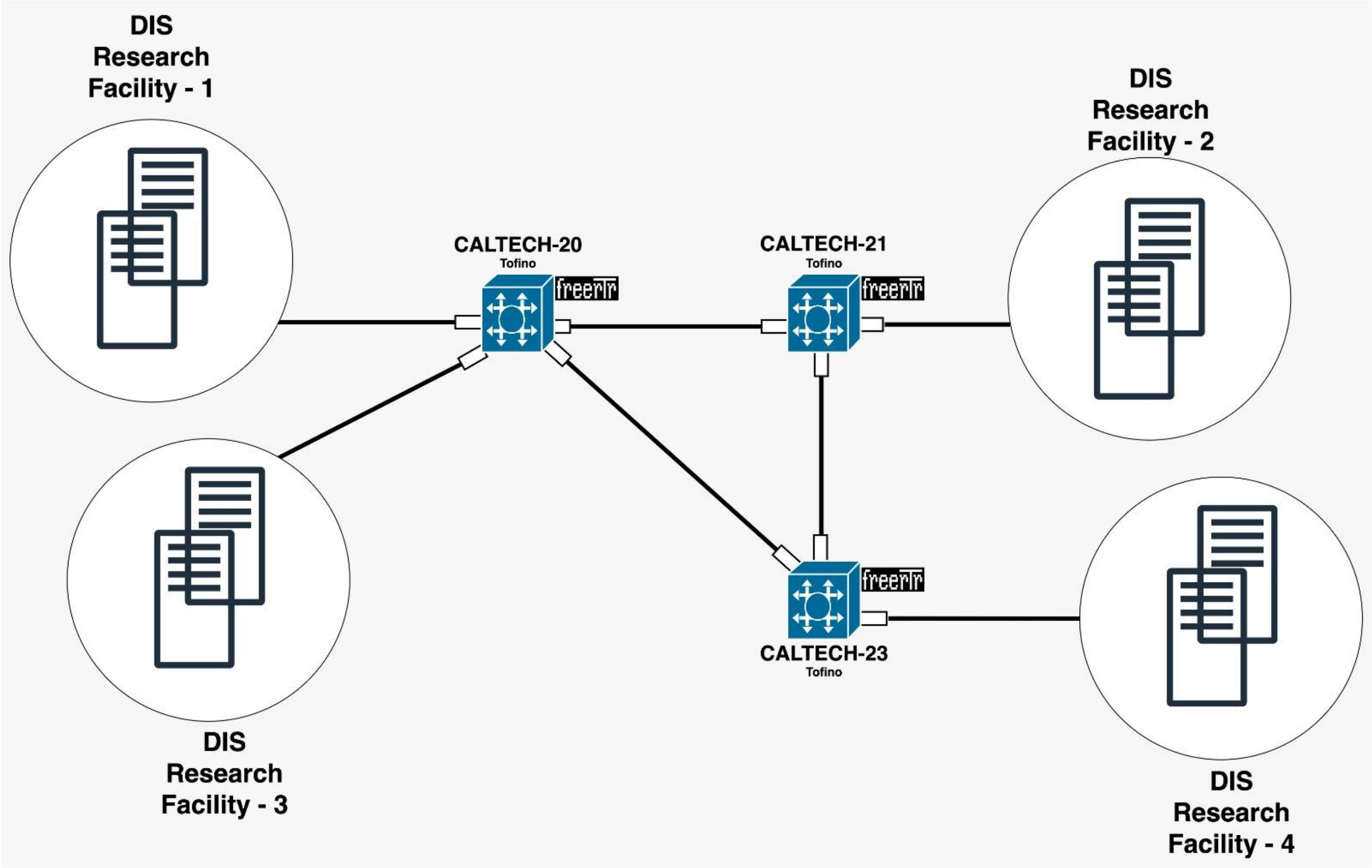


[Big data streams over PolKA tunnels at 100 Gbps]

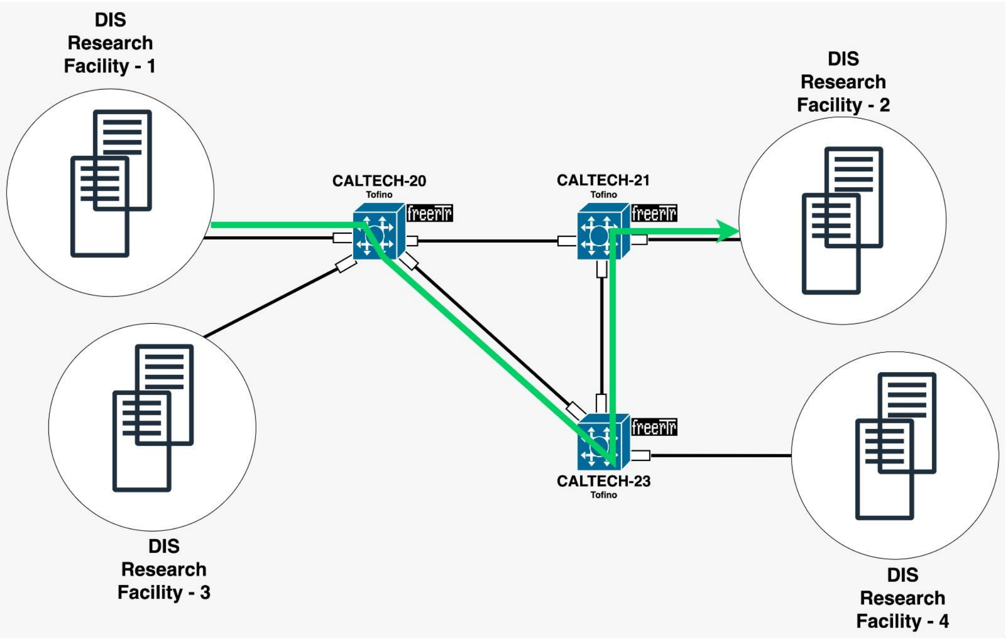
Big Data streams over PolKA tunnels at 100 Gbps in Caltech

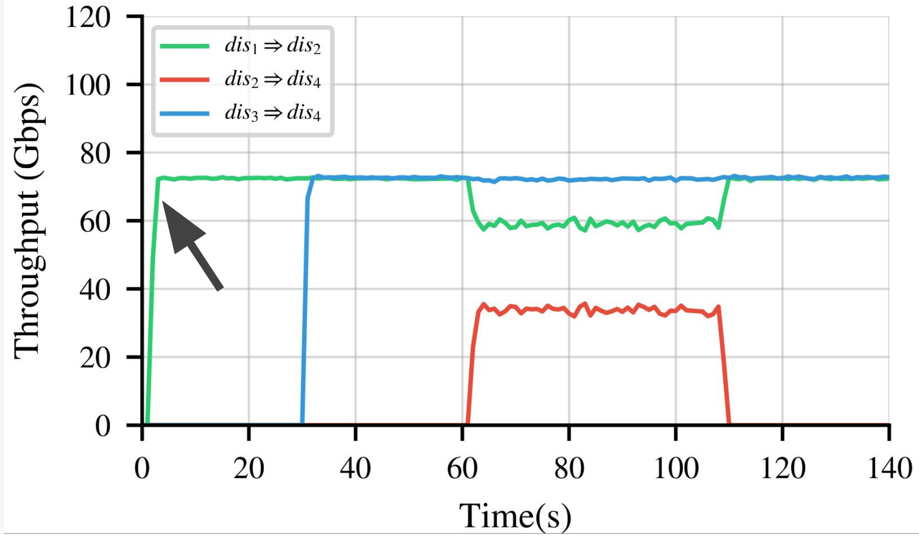


Big Data streams over PolKA tunnels at Caltech Booth

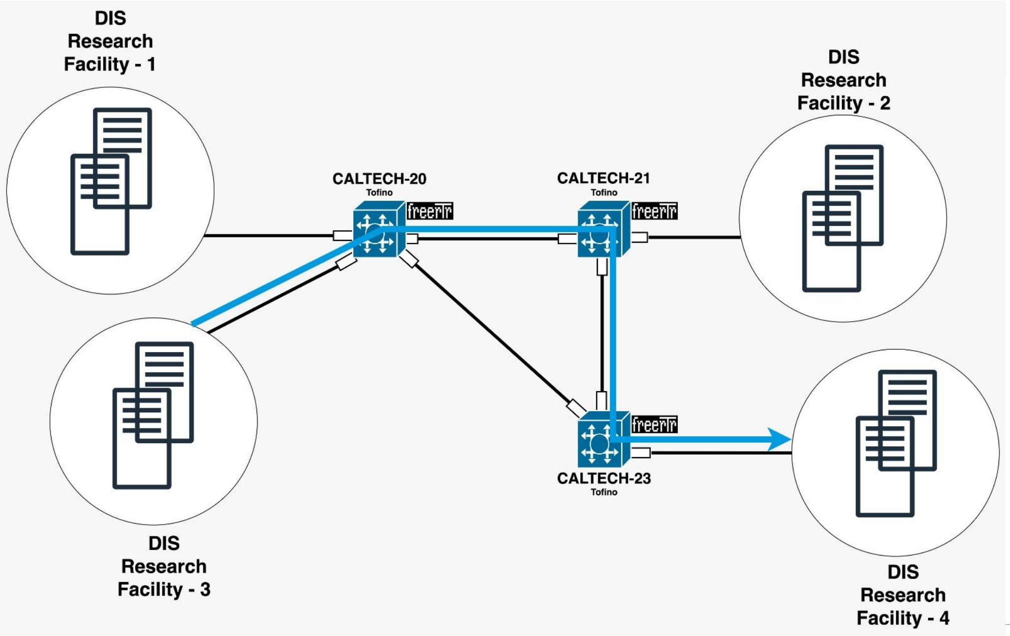


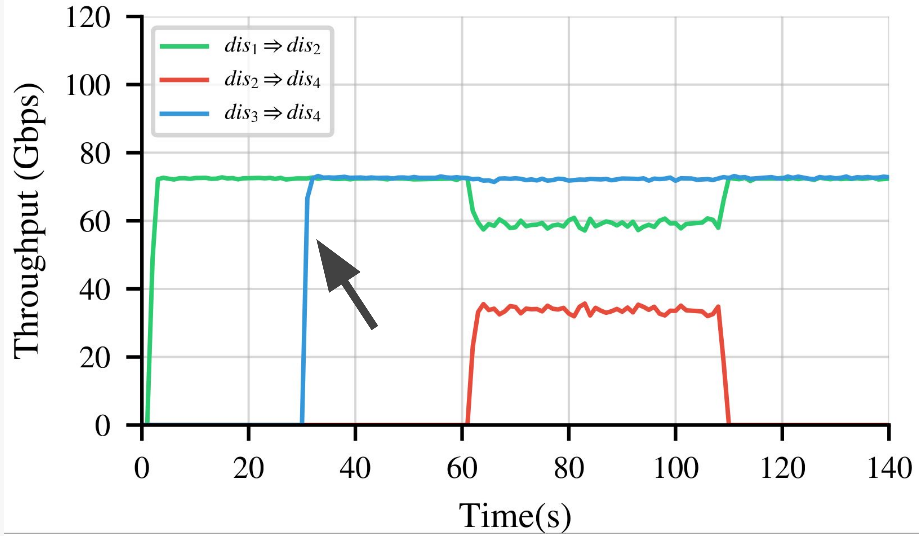
Big Data streams over PolKA tunnels at Caltech Booth



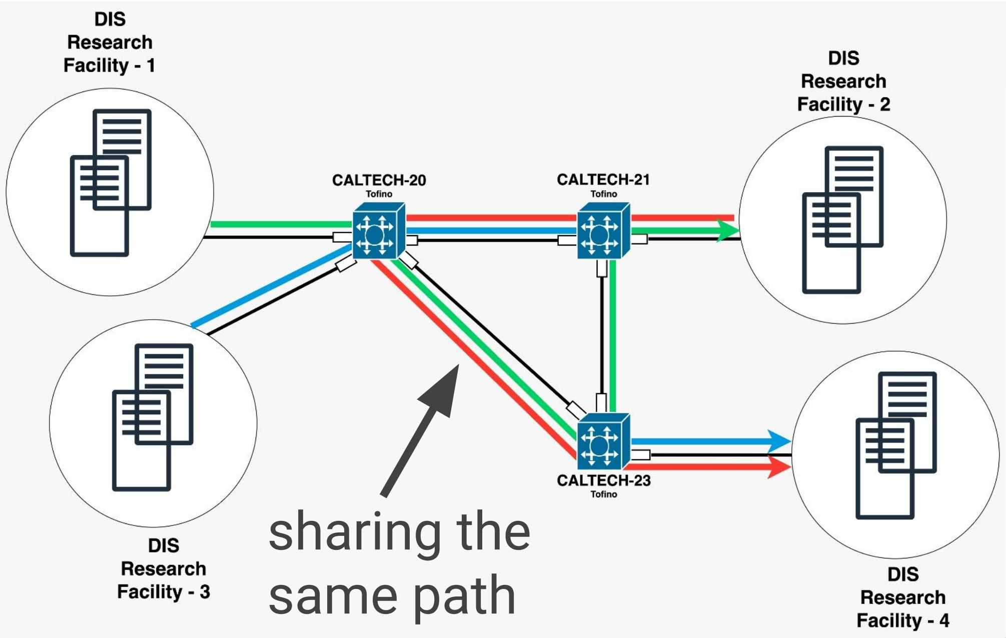


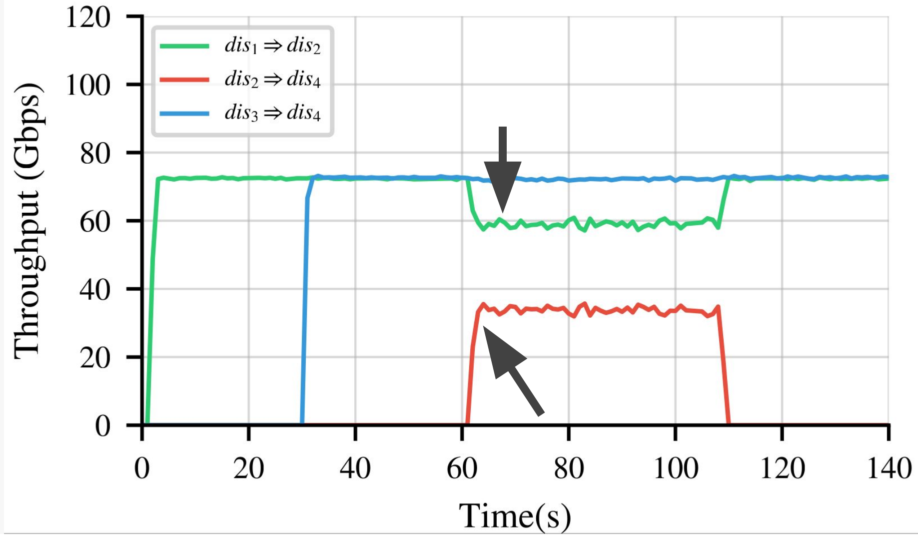
Big Data streams over PolKA tunnels at Caltech Booth



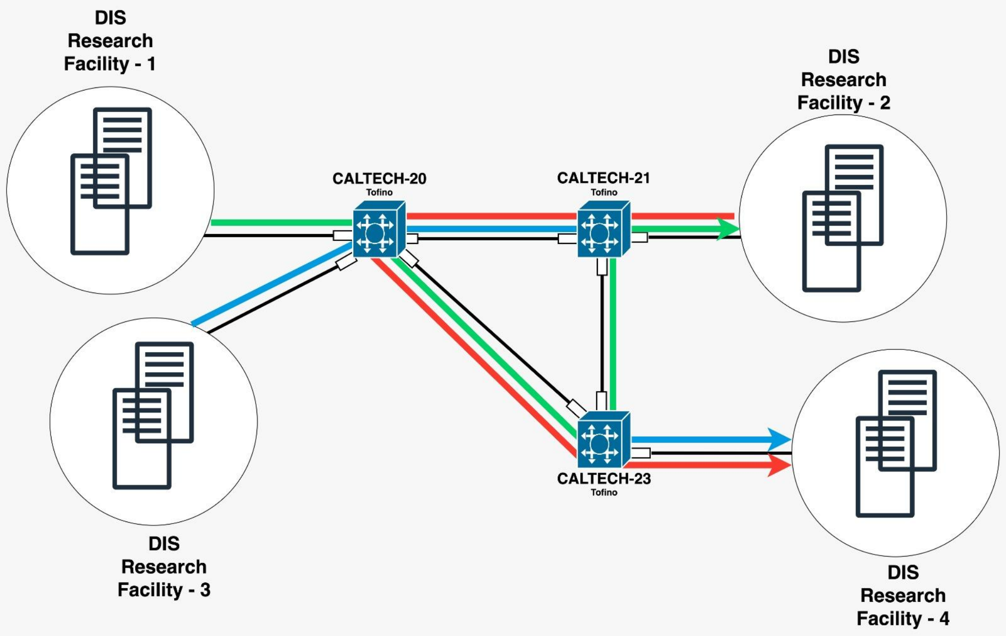


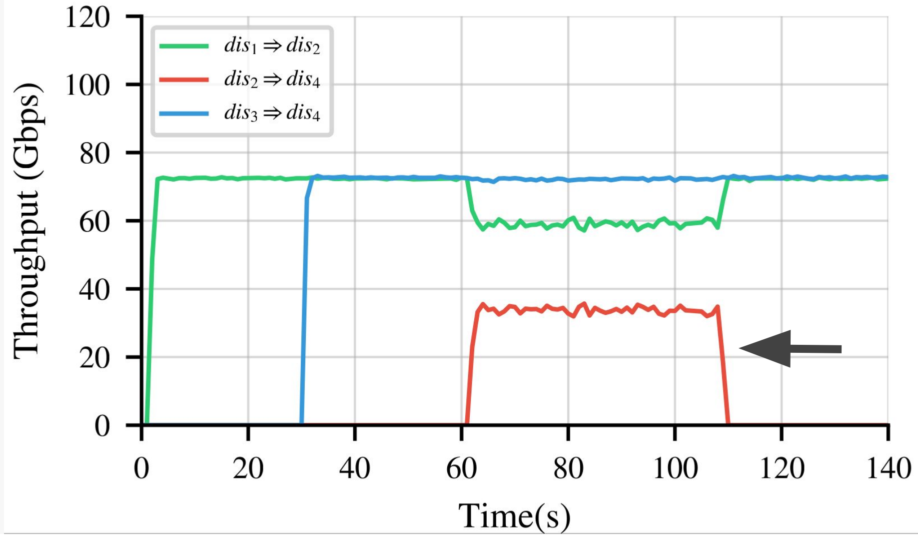
Big Data streams over PolKA tunnels at Caltech Booth



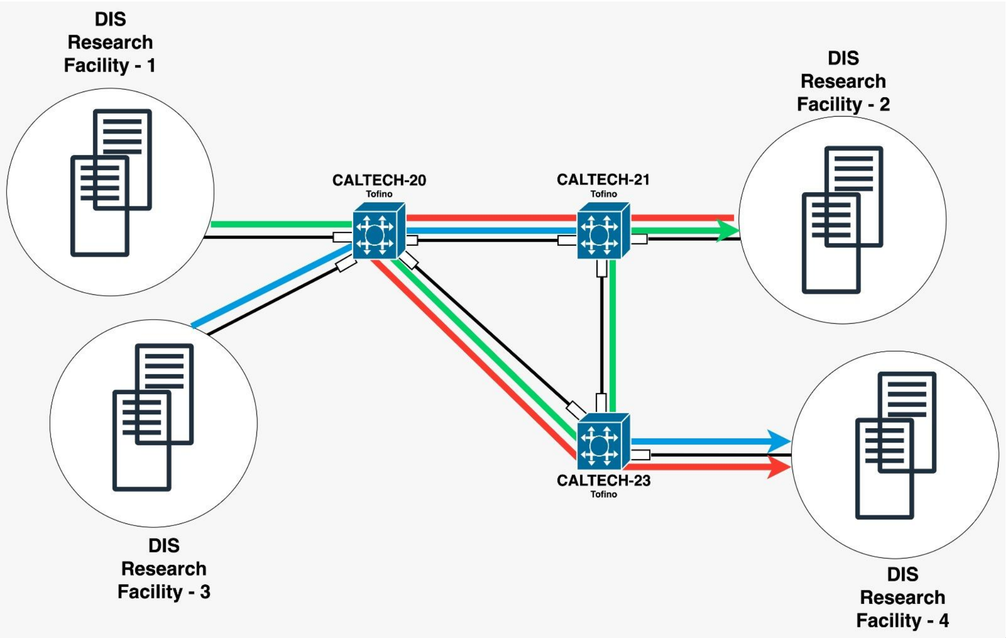


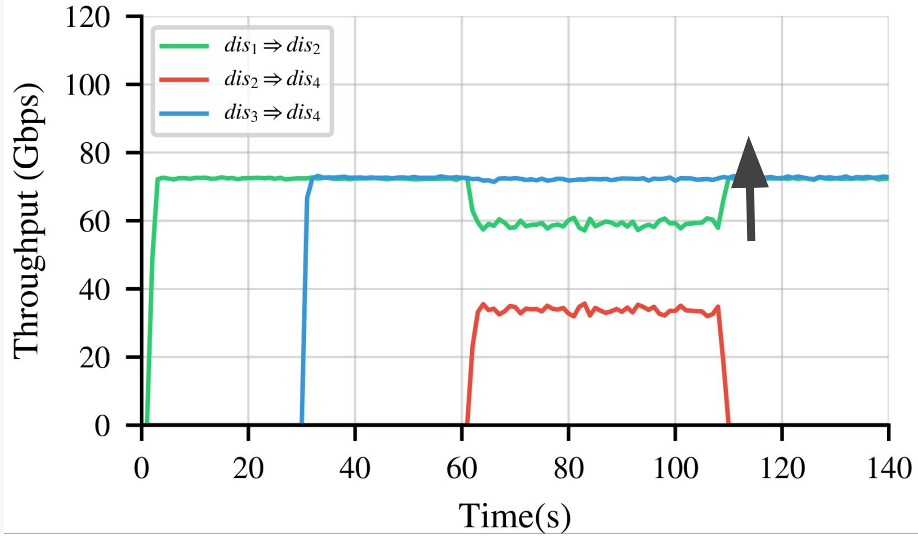
Big Data streams over PolKA tunnels at Caltech Booth





Big Data streams over PolKA tunnels at Caltech Booth





Take away messages

- It is **feasible to deploy PolKA in high-performance programmable network equipment** by reusing CRC hardware.
    - PolKA deployment in Caltech P4 lab testbed demonstrated its performance achieving transfer rate > 100 Gbps to multiple aggregated flows (TCP)

- **Easy to configure PolKA tunnels** with a common standard (CLI) or REST API

- Potential to support
    - Big pipes/tunnels configured in a underlay network
    - Massive data transfer with aggregation of a large number of large flows
    - Flow Steering exploring PolKA properties (e.g. stateless core nodes)
        - Explicit path and TE both **at the edge**

Acknowledgments

- The framework **provided by the GNA-G** and the whole ecosystem of **NRENs (Global P4 Lab)** enabled the PolKA routing to be tested thanks to:
  - GNA-G Data Intensive Science WG
  - GNA-G AutoGOLE / SENSE WG
  - GEANT RARE Project
  - … And all it’s collaborating institutions and teams
- Collaboration **with Caltech was crucial** hosting us at the booth
- Provides all the resources (e.g. Tofinos + servers +...) to deploy it in a near production allowing us to demonstrate PolKA in the best conditions

PolKA Community, Partners and Collaborations

- **Caltech :** Professor Harvey Newman and Raimondas Širvinskas
- **RARE GÉANT:** Frédéric Loui, Csaba Mate, Eoin Kenny
- **RNP:** Marcos Schwarz
- **Qualcomm**: Jordi Ros Giralt
- **Trinity College Dublin (Connect)**: Marco Ruffini and Frank Slyne
- **CNPq and FAPES (Brazilian research funding agencies)**
- **2021 Google Research Scholar Award**
- **2022 Intel Connectivity Research Grant (Fast Forward Initiative)**
- **University of Waikato (NZ)**

Selection of Our Recent Publications

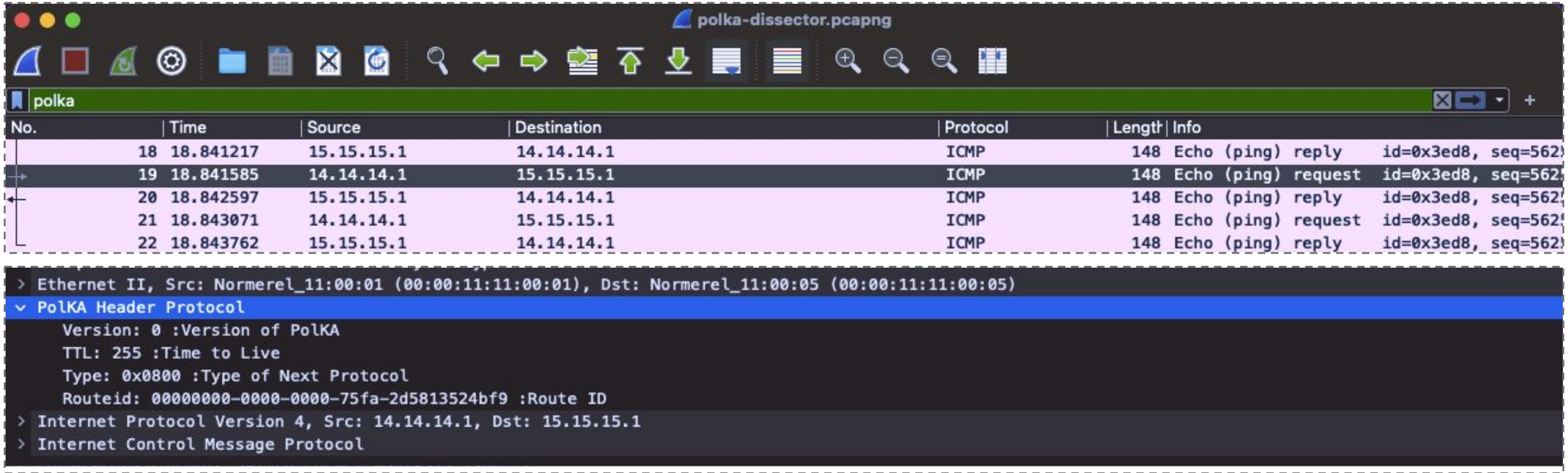
- In-situ Proof-of-Transit for Path-Aware Programmable Networks (IEEE NetSoft, 2023)
- M-PolKA: Multipath Polynomial Key-based Source Routing for Reliable Communications (IEEE TNSM, 2022)
- Chaining-Box: A Transparent Service Function Chaining Architecture Leveraging BPF (IEEE TNSM, 2021)
- Programmable Switches for in-Networking Classification (IEEE INFOCOM, 2021)
- Deploying PolKA Source Routing in P4 Switches (ONDM, 2021)
- PolKA: Polynomial Key-based Architecture for Source Routing in Network Fabrics (IEEE NetSoft, 2020)
- PlaFFE: A Place-as-you-go In-network Framework for Flexible Embedding of VNFs (IEEE ICC, 2020)
- ProgLab: Programmable labels for QoS provisioning on software defined networks (Computer Comm, 2020)
- KeySFC: Traffic steering using strict source routing for dynamic and efficient network orchestration (Computer Networks, 2020)
- FUTEBOL Control Framework: Enabling Experimentation in Convergent Optical, Wireless, and Cloud Infrastructures (IEEE COMMUNICATIONS MAGAZINE, 2019)
- RDNA: Residue-defined networking architecture enabling ultra-reliable low-latency datacenters (IEEE TNSM, 2018)

Additional references

1. Global Network Advancement Group: Towards a Next Generation System for Data Intensive Sciences
2. Documentation: Let's enable PolKA in freeRtr
3. PolKA presentation at Google Research Scholar Award
4. Multipath PolKA presentation at ONF 2022
5. PolKA github
6. RARE website
7. FreeRouter website
8. LabNERDS Videos
9. PolKA NetSoft 2020 conference paper
10. V. Shoup, A computational introduction to number theory and algebra, 2008.

PolKA: Github

- https://nerds-ufes.github.io/polka/
  - References
  - Tutorials (Mininet and FreeRouter)
  - Wireshark dissector
  - More to come…



Thank you for attention !

Rafael Guimarães

rafaelg@ifes.edu.br

** This work was a recipient of the 2021 Google Research Scholar and the 2022 Intel Connectivity Research Grant (Fast Forward Initiative) Awards, and Received funds from CAPES (Finance Code 001), CNPq, FAPESP, FAPES, CTIC, and RNP.*

INSTITUTO FEDERAL
Espírito Santo

UFES

Ok… Why should I use PolKA?

- One good reason…

  … It is easy to setup paths/tunnels!

- It has some interesting properties that enable innovative applications.
  - Ex: multicast communication model support, multipath routing, failure protection, Proof of Transit, telemetry…
- Open source implementation in software and in hardware
  - RARE/FreeRtr