RNP
ORGANIZAÇÃO SOCIAL DO MCTI

Open RAN BRASIL

Adicionando emulação realista de condições de redes no Testbed OpenRAN Brasil

Prof. Christian Rothenberg (Unicamp)

4-Apr. 2023

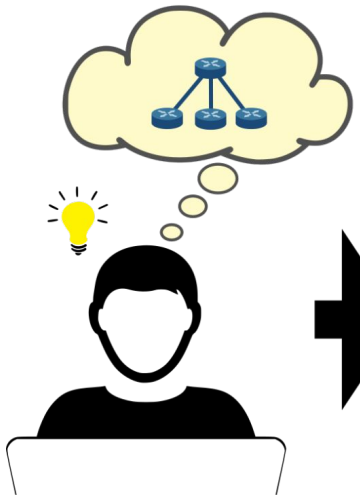
INTRIG
INFORMATION & NETWORKING TECHNOLOGIES RESEARCH & INNOVATION GROUP

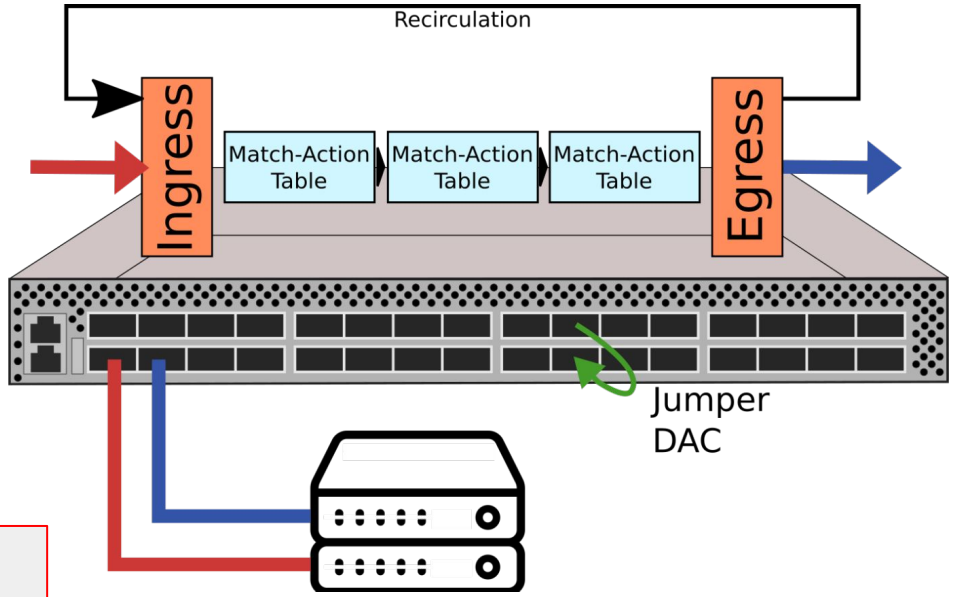
UNICAMP

Agenda

- P7 Emulator
  - Overview
  - Link emulation characteristics
  - Multiple pipelines approach
  - Architecture updates
- Demo
- Future of P7

P7





P7 is a high-fidelity 100G traffic network emulation, including various link characteristics such as latency, jitter, packet loss, and bandwidth, as well as the option to customize network topologies.

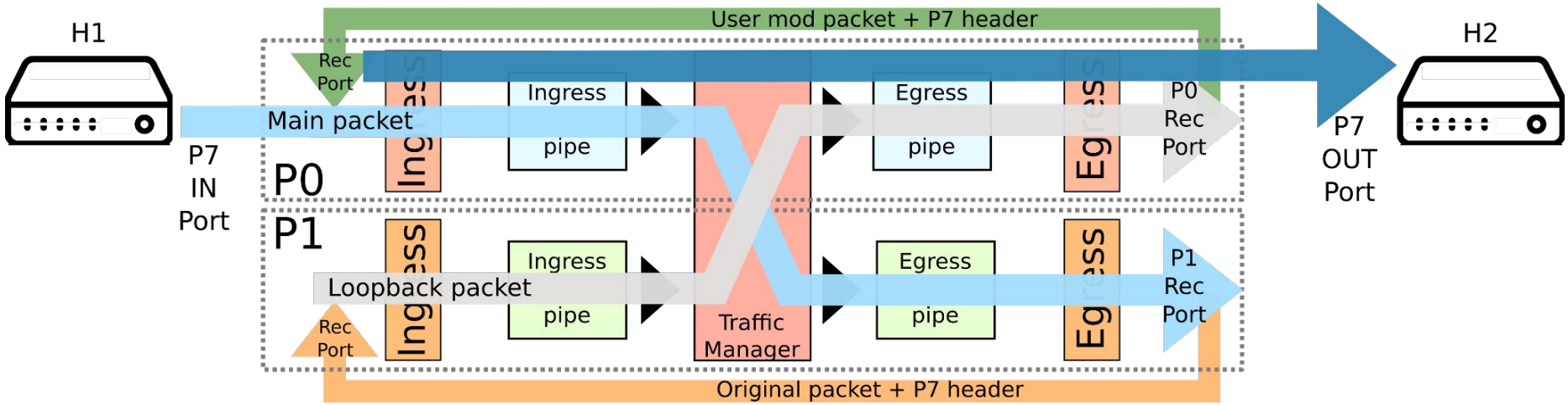
Everything implemented in a single P4 switch

Link characteristics and P4/TNA implementation approaches

Link Connectivity	Jumper cabling with internal Tag Intern Recirculation + internal Tag
Latency [ms]	Internal timer + recirculation TM + Pipelines recirculation
Jitter [ms]	Hash to determine recirculation times Lookup table with mathematical functions
Packet loss [%]	Random function to determine the probability to discard packets Realistic packet loss model
Bandwidth	Rate limit TNA TM feature Port configuration and shaping

This are the available link metrics and how were implemented

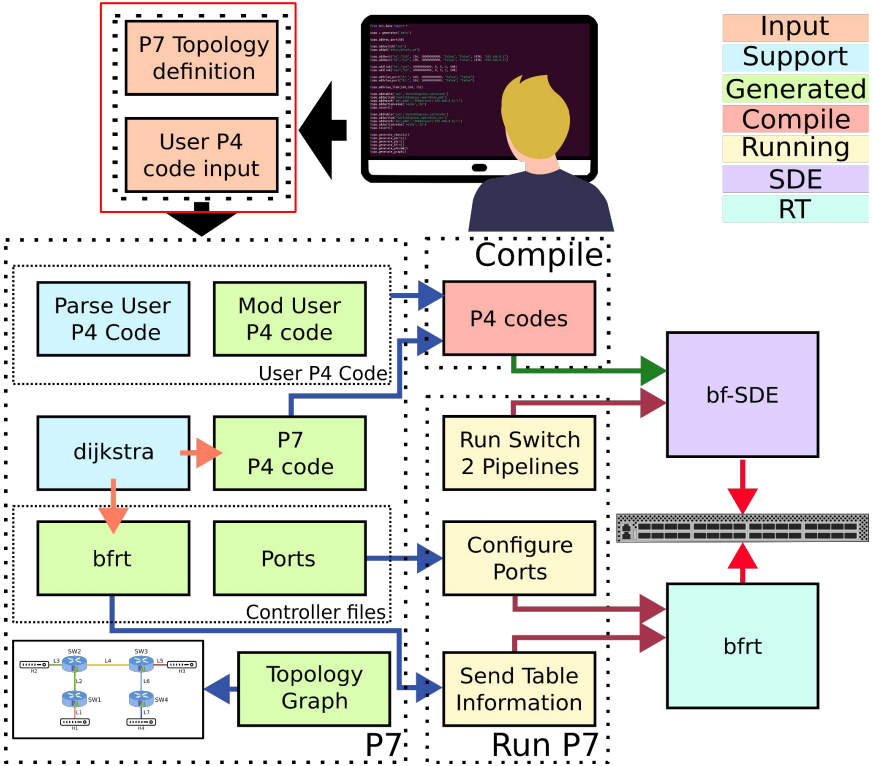
P7 multiple pipelines approach



We propose a solution where a dedicated pipe runs the P7 P4 code, and a separate pipe runs the user-defined P4 code

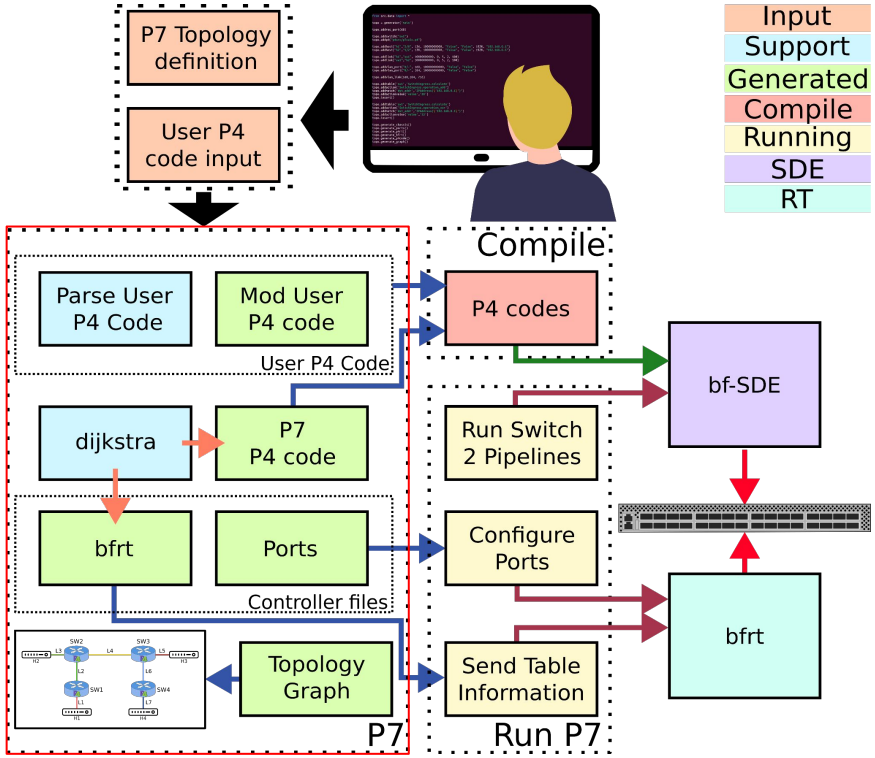
We send the packet in the P7 pipe (P0) to the pipe where the user-defined P4 code is running (P1) using recirculation

P7 Architecture & User workflow



The user defines the topology and sets a custom P4 code.

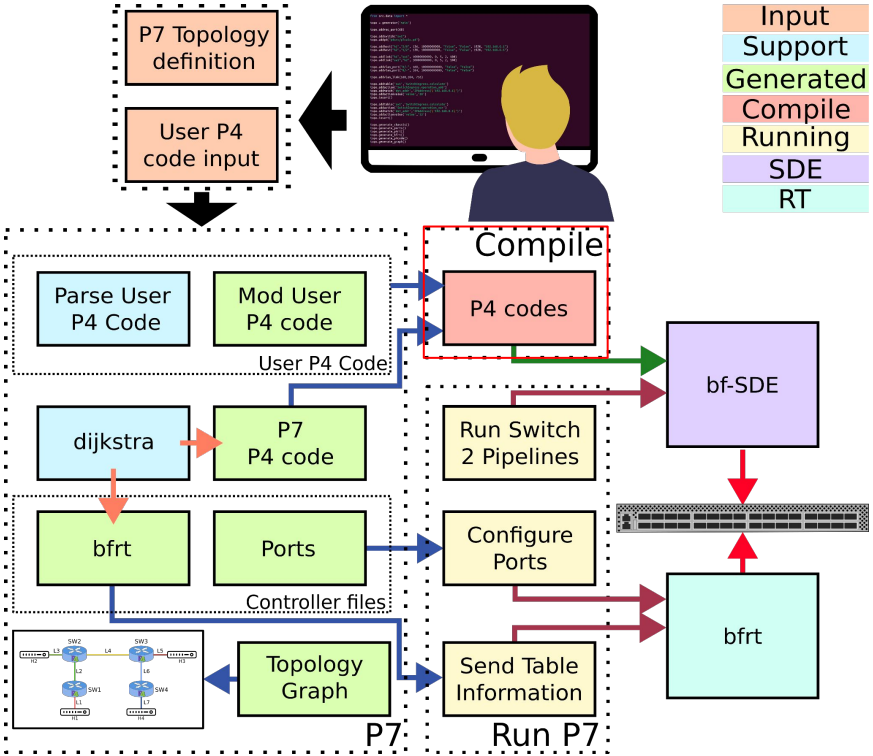
P7 Architecture & User workflow



P7 processes the data and generates the necessary files:

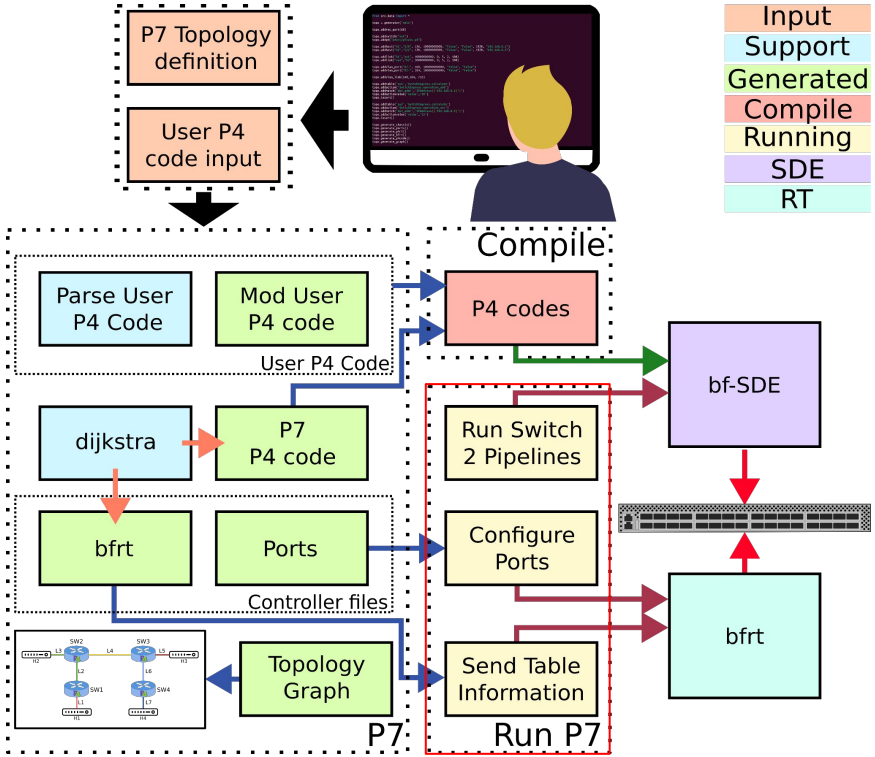
- P7 P4 code
- User P4 code
- Tables information
- Ports configuration

P7 Architecture & User workflow



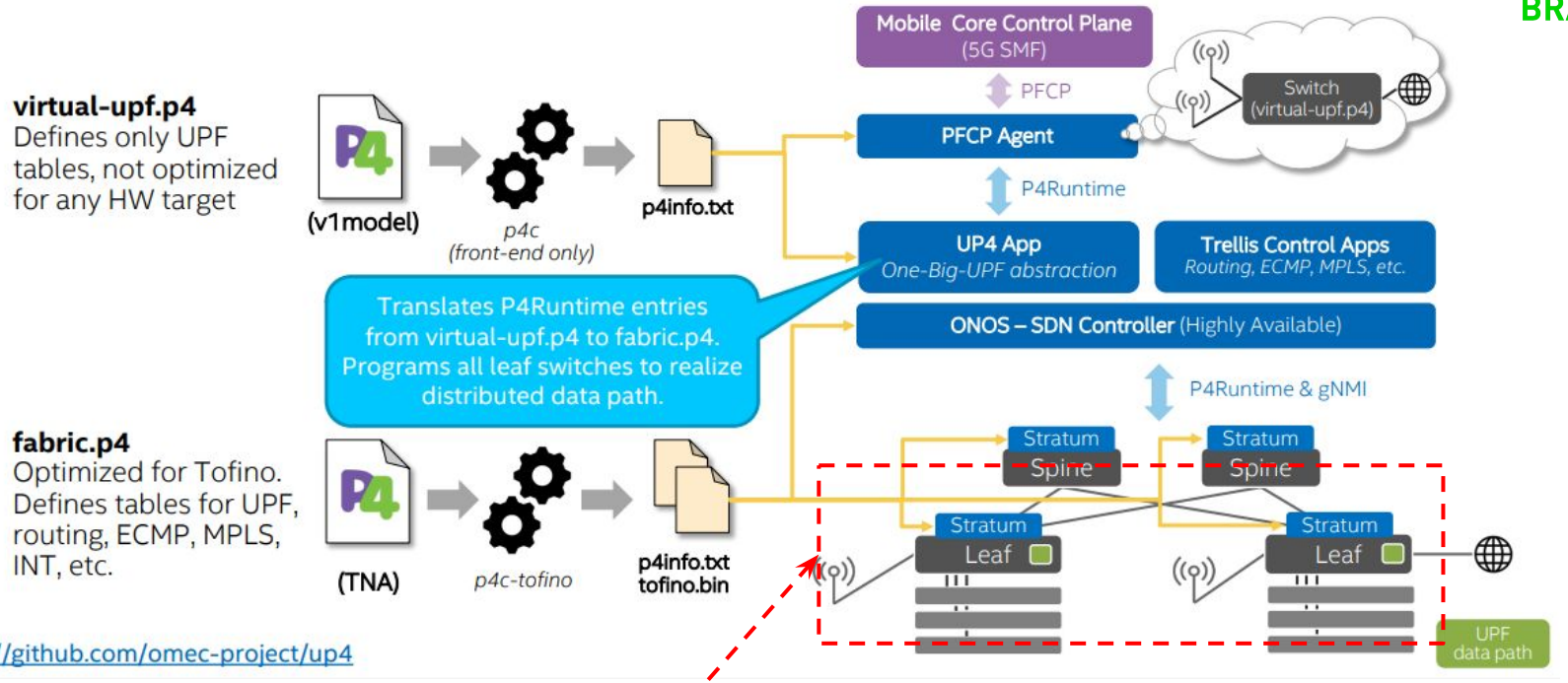
The user needs to compile both P4 codes

P7 Architecture & User workflow



Finally, the user can run the switch with both P4 codes and send the tables and ports configuration using the bfrt.

P4 + P7 @ Open RAN Brasil



https://github.com/omec-project/up4

We can leverage P7 to emulate UPF functionalities and link characteristics in different scenarios

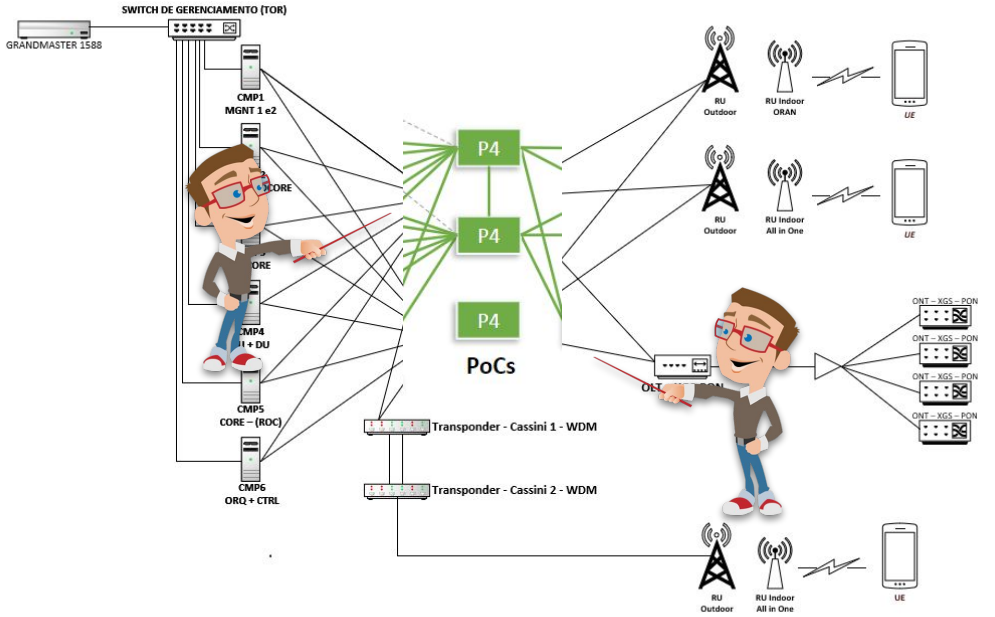
P4 + P7 @ Open RAN Brasil

Physical Testbed architectural options

- Bump-in-the-wire P7 emulation
  - Allocate one physical Tofino switch into the topology
- Shared Tofino
  - 1 pipe for P7
  - 1 pipe for UPF

Further considerations

- INT support
- Barefoot Runtime dynamic configuration of tables and ports
- 100G workloads



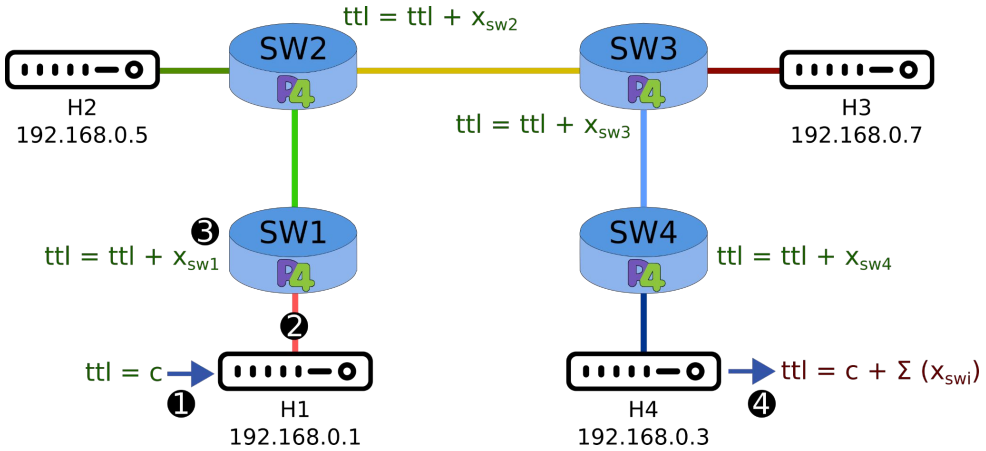
RNP
ORGANIZAÇÃO SOCIAL DO MCTI

DEMO

INTRIG
INFORMATION & NETWORKING
TECHNOLOGIES RESEARCH &
INNOVATION GROUP

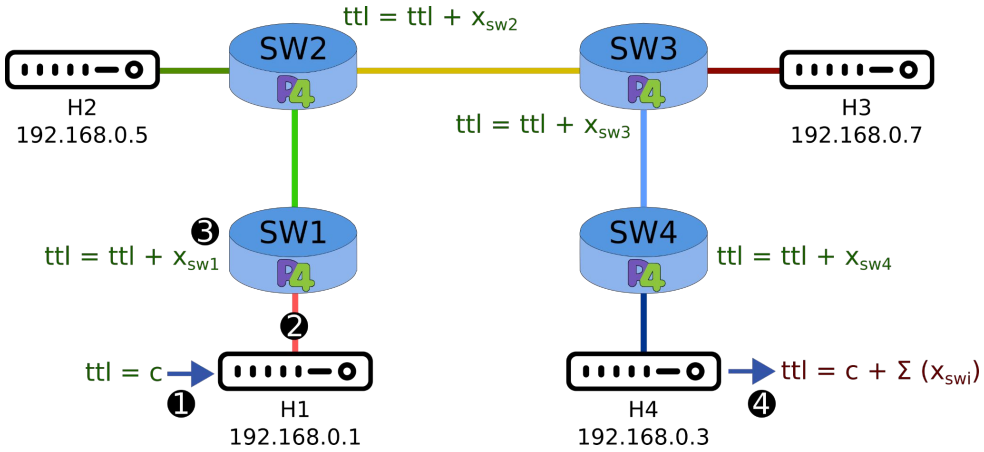
UNICAMP

Network topology

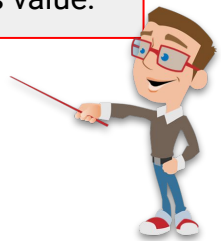


P4 code that contains different mathematical operations that are applied to the IP field ttl.

Network topology

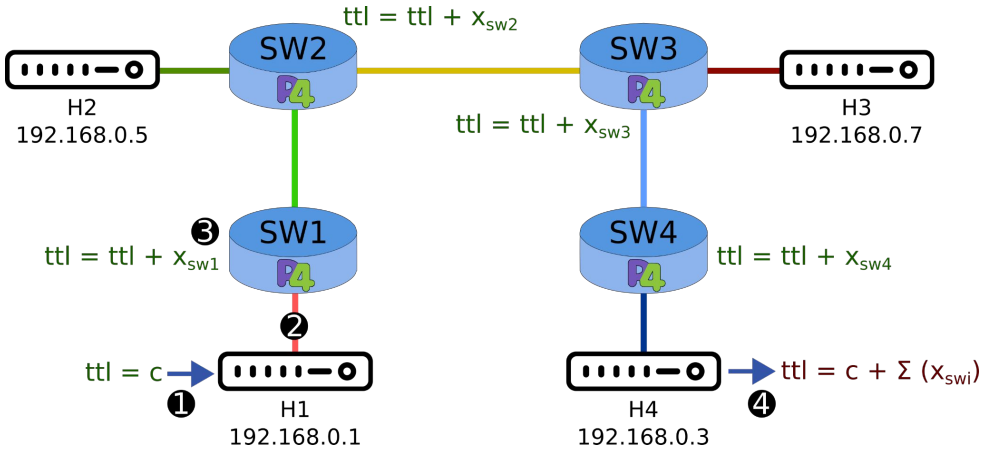


These operations are defined by a P4 table that contains the operation and its value.



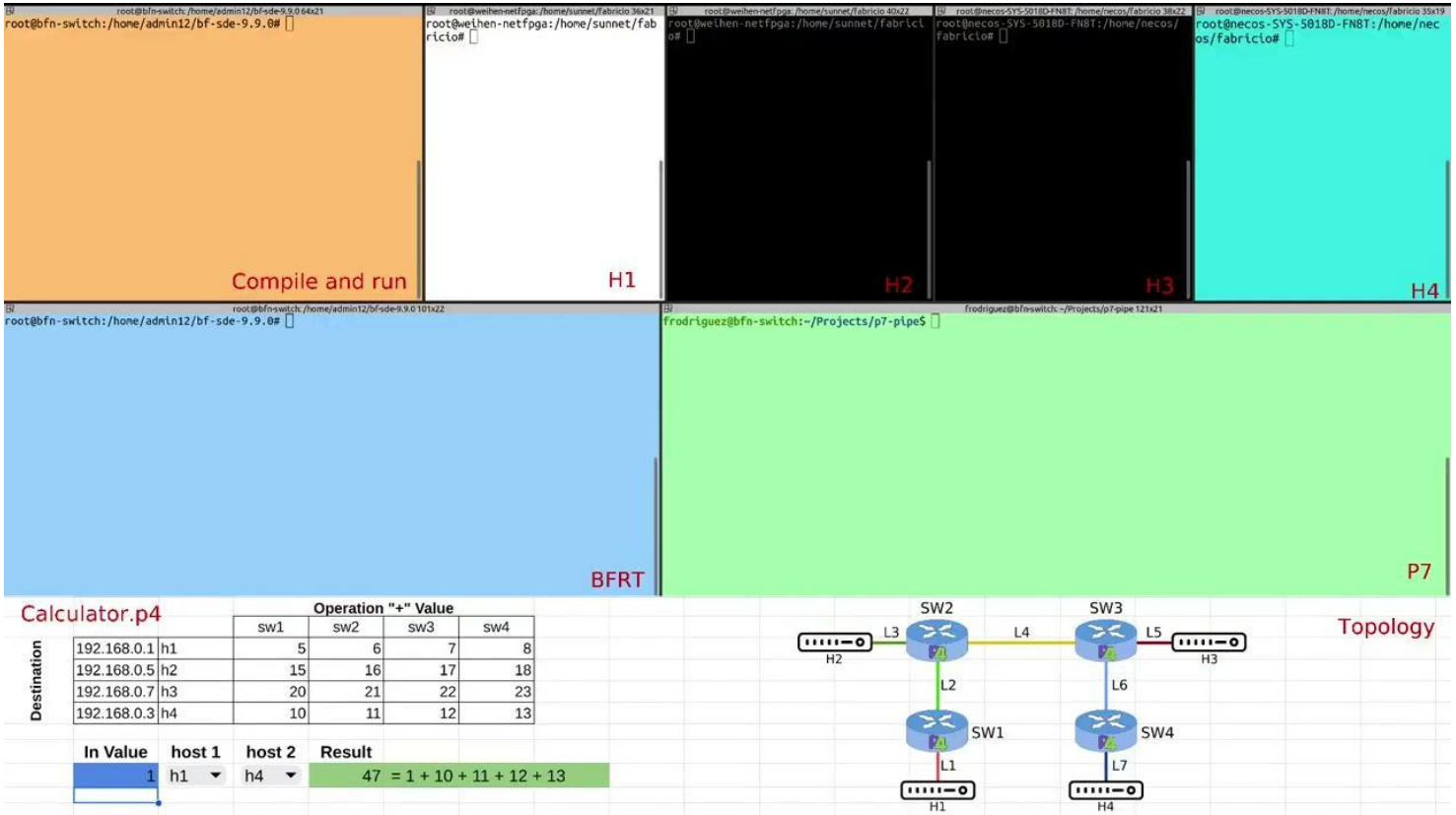
```
61  # add table entry sw1
62  topo.addtable('sw1','SwitchIngress.calculate')
63  topo.addaction('SwitchIngress.operation_add')
64  topo.addmatch('dst_addr','IPAddress(\'192.168.0.1\')')
65  topo.addactionvalue('value','5')
66  topo.insert()
```

Network topology



The P4 code will perform a specific operation based on the destination IP of the packet and the information filled in a table for each switch.

```
61  # add table entry sw1
62  topo.addtable('sw1','SwitchIngress.calculate')
63  topo.addaction('SwitchIngress.operation_add')
64  topo.addmatch('dst_addr','IPAddress(\'192.168.0.1\')')
65  topo.addactionvalue('value','5')
66  topo.insert()
```



Demo running P7 with the calculator P4 code

https://drive.google.com/file/d/1EYjpcuwoSguL9yO2cnliIYH7vW-mRFMQ/view?usp=share_link

P7 repository

Available on:

https://github.com/intrig-unicamp/p7

Future of P7

- Address scalability challenges
  - Topology Size
  - Buffers consumption
- New features
  - In-band Network Telemetry (INT)
  - Dynamic link behaviors
  - Trace base link characteristics
- Embed into disaggregated network testbed initiatives
  - e.g Open RAN Brasil
  - Facilitate reproducible experiments based on use case scenarios (e.g. congestion, heavy-hittters, DDoS, bufferbloat, slicing, etc.)

RNP
ORGANIZAÇÃO SOCIAL DO MCTI

Thank You

https://intrig.dca.fee.unicamp.br

INTRIG
INFORMATION & NETWORKING TECHNOLOGIES RESEARCH & INNOVATION GROUP

UNICAMP

BACKUP

Required IT infrastructure

- Laptop (provided by us)
- Remote P4 Tofino hardware switch (provided by us)
- 4x Remote Servers (provided by us)
- Power.
- Internet connectivity.
- 1x Monitor.

During the demo

We will present different use cases to attendees, including topologies with link metrics and custom P4 codes.

We will run P7 remotely in physical Tofino Hardware connected with different physical servers.

We will present the calculator use case with different configurations of operations and values.

During the demo

We will show the calculator use case with various operation values and different source and destination hosts. Also, the results can be confirmed by a local script (link below) that calculates the operations accordingly to the topology and table information.

https://docs.google.com/spreadsheets/d/1C3yPnvxPETJxFhxvbey-Ho8AhBPjeWPYsiKpb5twTgY/edit?usp=sharing

During the demo

Attendees will be asked to define their topologies with different link metrics (e.g., latency, jitter, packet loss, background traffic, bandwidth) and a custom P4 code.

They can validate the defined topology and see P7 in action, including the auto-generation of files and the complete environment running.

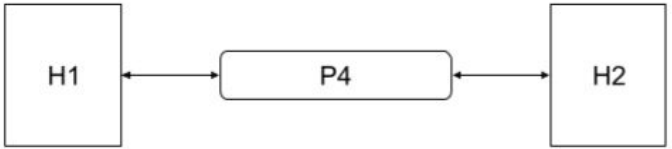
During the demo

Real-time visualization of network traffic will contribute to validating on-the-fly performance of the link metric and the emulation capabilities.

We will run traffic generators in different serves to validate the link metrics. In addition, the servers can send custom traffic to test user-defined P4 codes.

Link characteristics

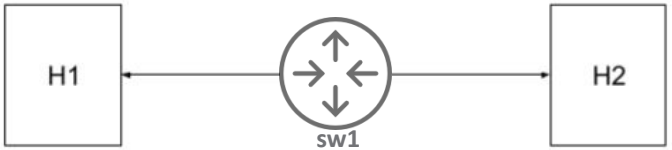
Different types of topologies and how are represented in the emulation



Physical Topology

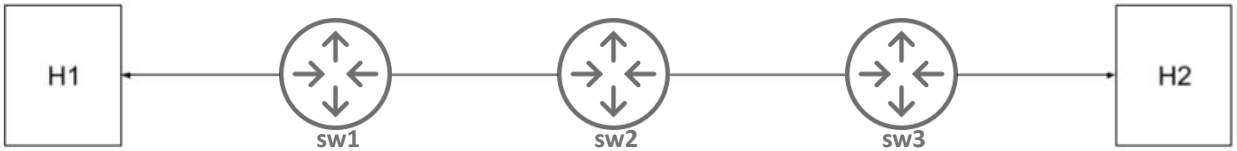
Emulated Topology

Direct link



Emulated Topology

Device in the middle

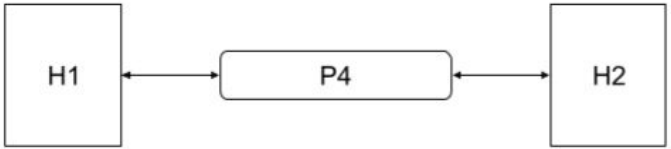


Emulated Topology

Various devices in the middle

Link characteristics

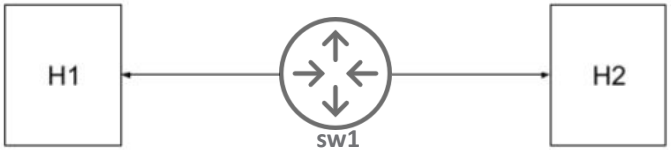
It is also possible to set an specific vlan to not pass over the P7 processing



Physical Topology

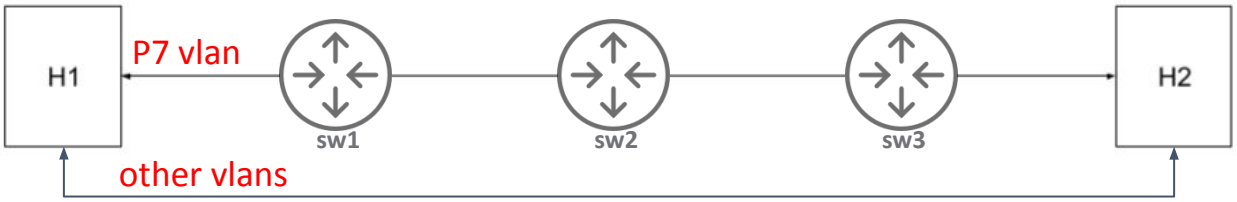
Emulated Topology

Direct link



Emulated Topology

Device in the middle



Emulated Topology

Various devices in the middle

Direct link

```
1  from src.data import *
2
3  topo = generator('main')
4
5  # Stratum ip:port
6  topo.addstratum("192.168.110.238:9559")
7
8  # Recirculation port default 68
9  topo.addrec_port(68)
10
11 # addhost(name,port,D_P_speed_bps,AU_FEC,vlan)
12 topo.addhost("h1",19, 20, 100000000000, "False", "False", 1920)
13 topo.addhost("h2",20, 28, 100000000000, "False", "False", 1920)
14
15 # addlink(node1, node2, bw, pkt_loss, latency)
16 topo.addlink("h1","h2", 100000000000, 0, 5)
17
18 topo.generate_chassis()
19 topo.generate_p4rt()
20 topo.generate_p4code()
21
22 topo.generate_graph()
```

Host name

Port number

Port speed

Port config

P7 vlan

The user can define physical servers and define where are connected in the Tofino

Direct link

```
1  from src.data import *
2
3  topo = generator('main')
4
5  # Stratum ip:port
6  topo.addstratum("192.168.110.238:9559")
7
8  # Recirculation port default 68
9  topo.addrec_port(68)
10
11 # addhost(name,port,D_P,speed_bps,AU,FEC,vlan)
12 topo.addhost("h1",19, 20, 100000000000, "False", "False", 1920)
13 topo.addhost("h2",20, 28, 100000000000, "False", "False", 1920)
14
15 # addlink(node1, node2, bw, pkt_loss, latency)
16 topo.addlink("h1", "h2", 100000000000, 0, 5)
17
18 topo.generate_chassis()
19 topo.generate_p4rt()
20 topo.generate_p4code()
21
22 topo.generate_graph()
```

node 1

node 2

link bw

latency

pkt loss

The user can define the characteristics of each emulated link

Device in the middle

```
1  from src.data import *
2
3  topo = generator('main')
4
5  # Stratum ip:port
6  topo.addstratum("192.168.110.238:9559")
7
8  # Recirculation port default 68
9  topo.addrec_port(68)
10
11 # addswitch(name)
12 topo.addswitch("sw1")
13
14 # addhost(name,port,D_P,speed_bps,AU,FEC,vlan)
15 # include the link configuration
16 topo.addhost("h1",19, 20, 100000000000, "False", "False", 1920)
17 topo.addhost("h2",20, 28, 100000000000, "False", "False", 1920)
18
19 # addlink(node1, node2, bw, pkt_loss, latency)
20 topo.addlink("h1","sw1", 100000000000, 0, 5)
21 topo.addlink("sw1","h2", 100000000000, 0, 5)
22
23 topo.generate_chassis()
24 topo.generate_p4rt()
25 topo.generate_p4code()
26
27 topo.generate_graph()
```

switch name

links

The user can define physical servers and define where are connected in the Tofino

Custom P4 code

```
24  # Recirculation port default 68
25  topo.addrec_port(196)
26  topo.addrec_port_user(68)
27
28  # addswitch(name)
29  topo.addswitch("sw1")
30  topo.addswitch("sw2")
31  topo.addswitch("sw3")
32  topo.addswitch("sw4")
33  topo.addp4("p4src/p7calc.p4")
34
35  # addhost(name,port,D_P,speed_bps,AU,FEC,vlan)
36  # include the link configuration
37  topo.addhost("h1","2/0", 136, 10000000000, "False", "False", 1920, "192.168.0.1")
38  topo.addhost("h2","1/0", 128, 10000000000, "False", "False", 1920, "192.168.0.7")
39  topo.addhost("h3","2/1", 137, 10000000000, "False", "False", 1920, "192.168.0.5")
40  topo.addhost("h4","1/2", 130, 10000000000, "False", "False", 1920, "192.168.0.3")
41
42
43  # addlink(node1, node2, bw, pkt_loss, latency, jitter, percentage)
44  # bw is considered just for the first defined link
45  topo.addlink("h1","sw1", 10000000000, 0, 0, 0, 100)
46  topo.addlink("h2","sw2", 10000000000, 0, 0, 0, 100)
47  topo.addlink("h3","sw3", 10000000000, 0, 0, 0, 100)
48  topo.addlink("h4","sw4", 10000000000, 0, 0, 0, 100)
49  topo.addlink("sw1","sw2", 10000000000, 0, 0, 0, 100)
50  topo.addlink("sw2","sw3", 10000000000, 0, 0, 0, 100)
51  topo.addlink("sw3","sw4", 10000000000, 0, 0, 0, 100)
```

switch name

custom P4 code

links

The user defines the number of switches and the custom P4 code

P4 table information

```
61  # add table entry sw1
62  topo.addtable('sw1','SwitchIngress.calculate')
63  topo.addaction('SwitchIngress.operation_add')
64  topo.addmatch('dst_addr','IPAddress(\'192.168.0.1\')')
65  topo.addactionvalue('value','5')
66  topo.insert()
67
68  topo.addtable('sw1','SwitchIngress.calculate')
69  topo.addaction('SwitchIngress.operation_add')
70  topo.addmatch('dst_addr','IPAddress(\'192.168.0.3\')')
71  topo.addactionvalue('value','10')
72  topo.insert()
73
74  topo.addtable('sw1','SwitchIngress.calculate')
75  topo.addaction('SwitchIngress.operation_add')
76  topo.addmatch('dst_addr','IPAddress(\'192.168.0.5\')')
77  topo.addactionvalue('value','15')
78  topo.insert()
79
80  topo.addtable('sw1','SwitchIngress.calculate')
81  topo.addaction('SwitchIngress.operation_add')
82  topo.addmatch('dst_addr','IPAddress(\'192.168.0.7\')')
83  topo.addactionvalue('value','20')
84  topo.insert()
```

switch name

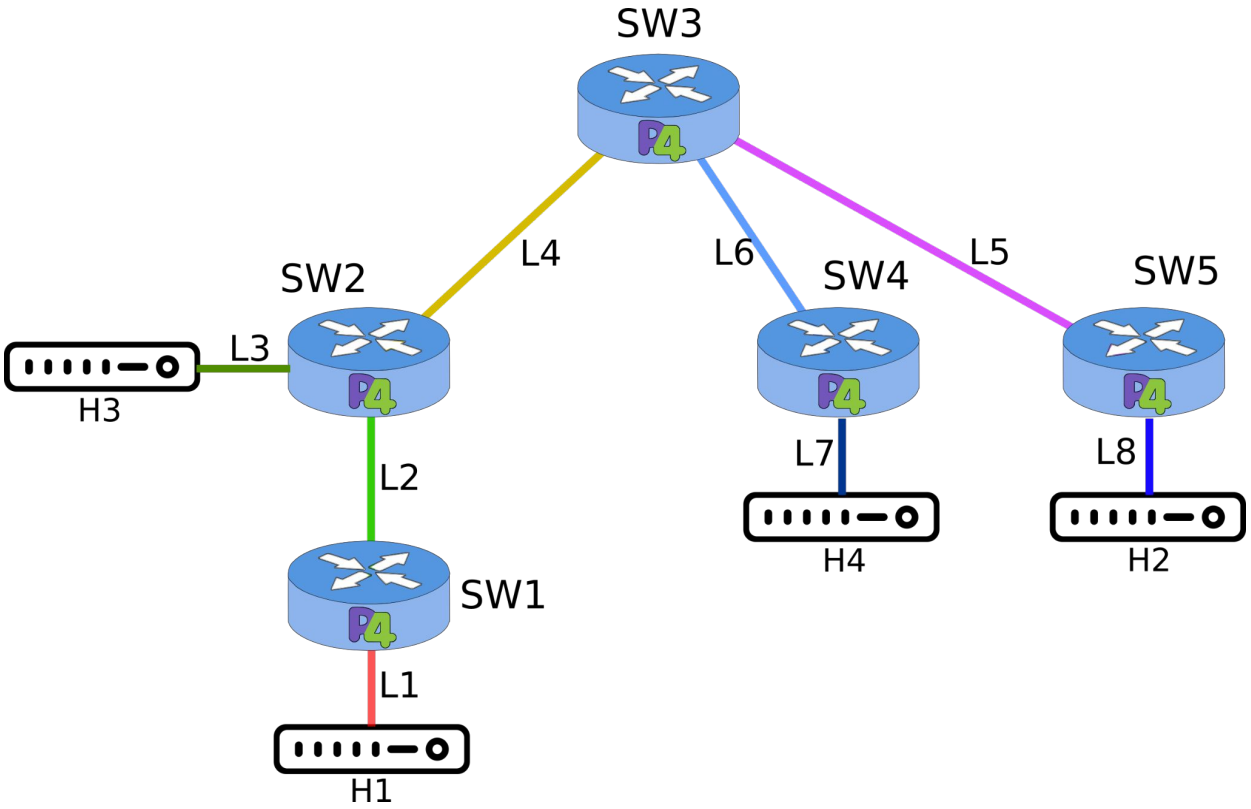
match action

table keys

action value

The user defines the table information for each emulated switch

Network topology



P7 network topology taxonomy