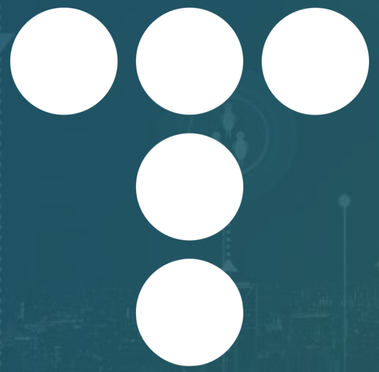


Telefónica

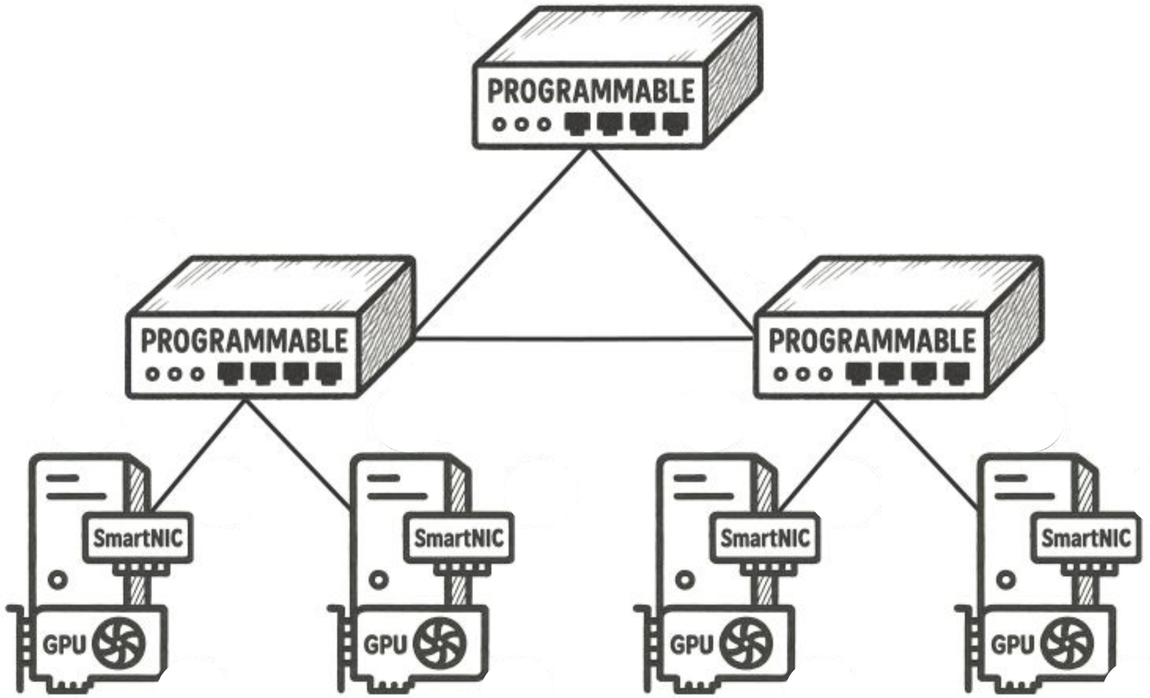
Programmable Hardware Emulation Environment for Realistic and Scalable Network Testing

Fabricio Rodriguez
Telefonica Research

Francisco Germano Vogt
Universidade Estadual de Campinas (UNICAMP)

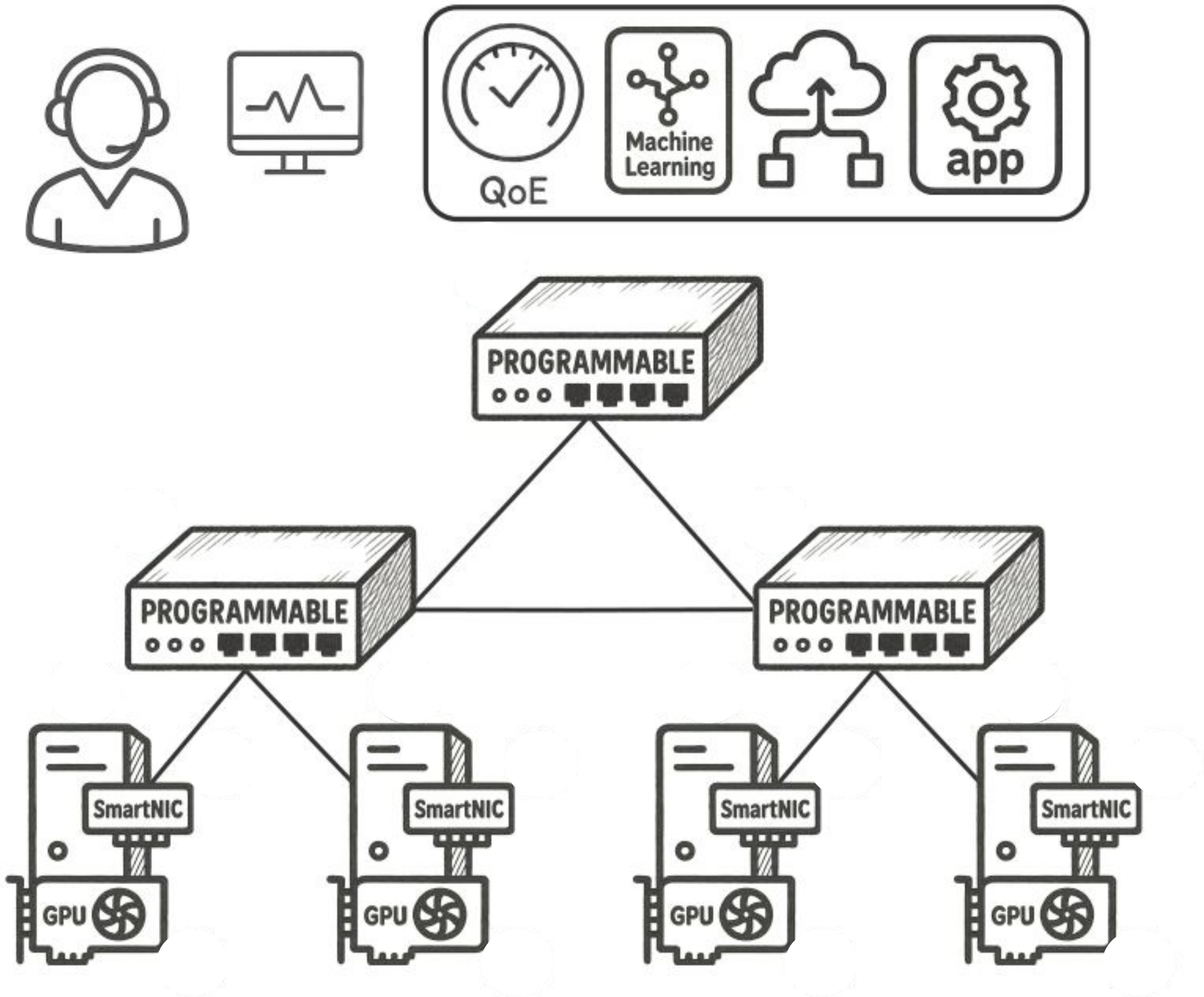
P4 Developer Days 🕒 (10-Dec, 2025)

Context



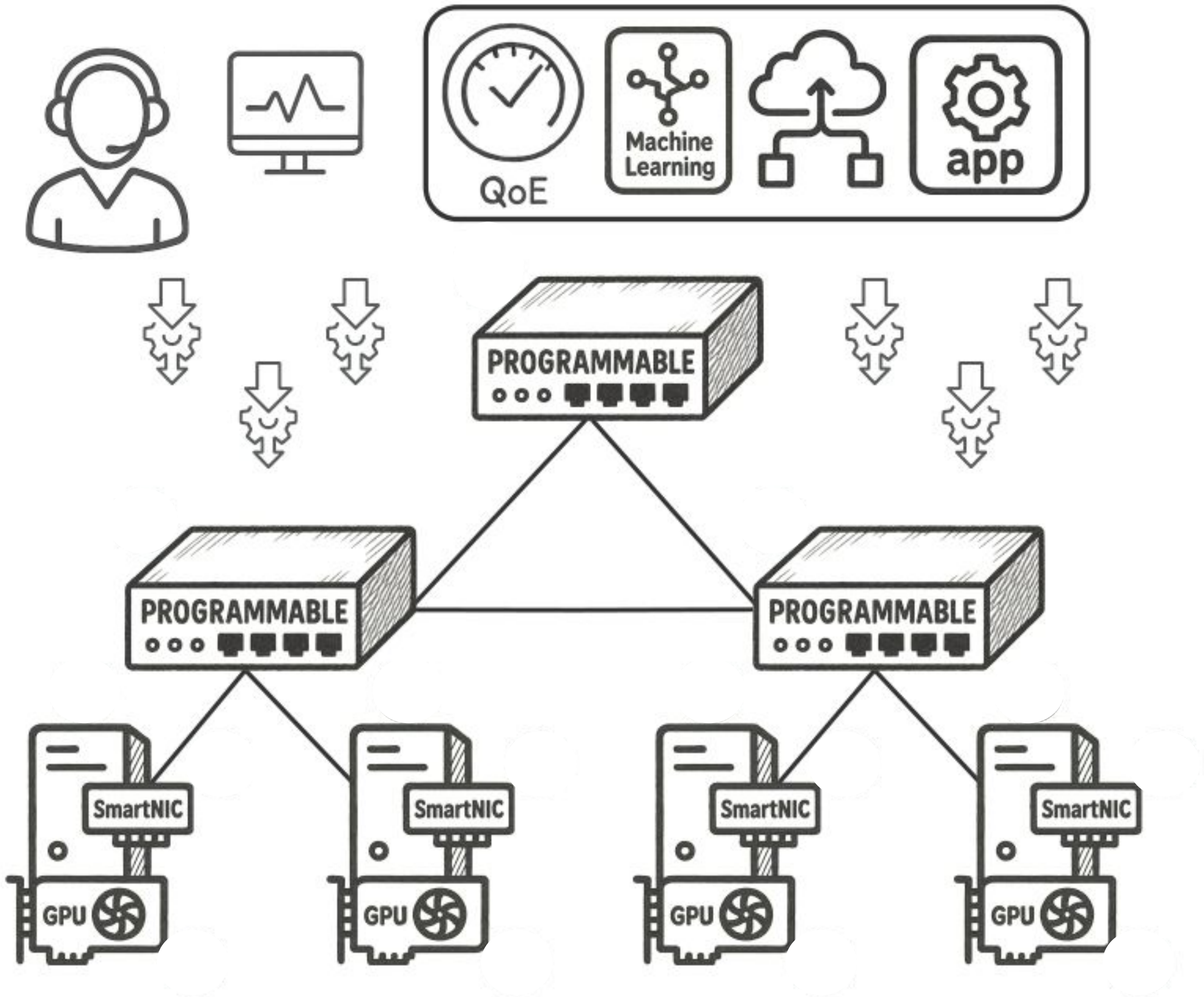
P4 Developer Days 🕒 (10-Dec, 2025)

Context



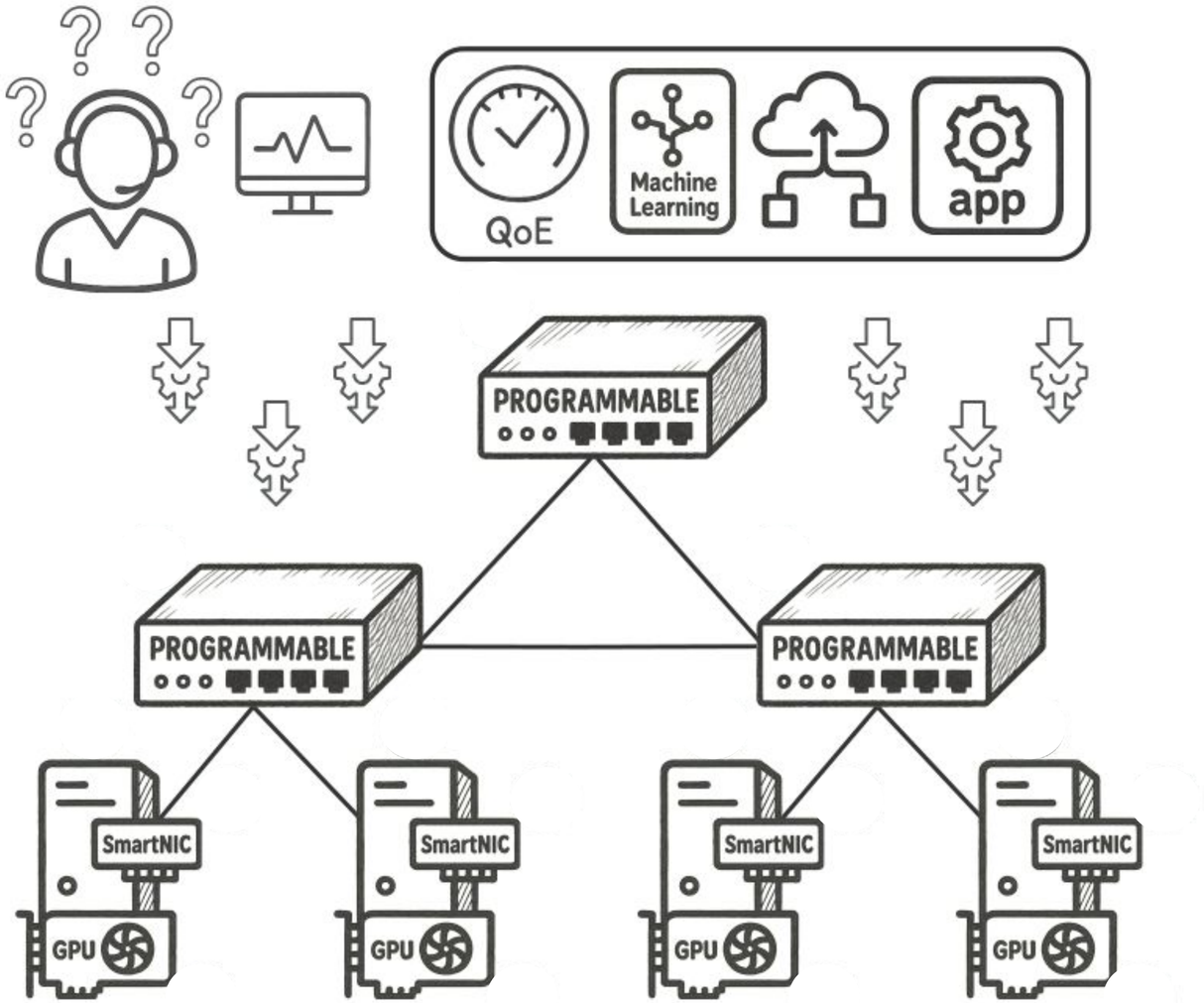
P4 Developer Days 🕒 (10-Dec, 2025)

Context



P4 Developer Days (10-Dec, 2025)

Context



P4 Developer Days 🕒 (10-Dec, 2025)

Context

Fidelity and Scalability: Experiments must combine high fidelity (running on real hardware and devices) with scalability (supporting the size and complexity of real-world topologies, such as data centers).

P4 Developer Days 🕒 (10-Dec, 2025)

Context

Fidelity and Scalability: Experiments must combine high fidelity (running on real hardware and devices) with scalability (supporting the size and complexity of real-world topologies, such as data centers).

Reproducibility: Ensuring that experiments and results can be reproduced by other researchers, while also enabling the reproduction and validation of prior work.

P4 Developer Days 🕒 (10-Dec, 2025)

Context

Fidelity and Scalability: Experiments must combine high fidelity (running on real hardware and devices) with scalability (supporting the size and complexity of real-world topologies, such as data centers).

Reproducibility: Ensuring that experiments and results can be reproduced by other researchers, while also enabling the reproduction and validation of prior work.

Fault and Event Injection: Introducing controlled and reproducible faults or events (e.g., packet loss, delay, congestion) is challenging on real hardware platforms.

P4 Developer Days 🕒 (10-Dec, 2025)

Context

Fidelity and Scalability: Experiments must combine high fidelity (running on real hardware and devices) with scalability (supporting the size and complexity of real-world topologies, such as data centers).

Reproducibility: Ensuring that experiments and results can be reproduced by other researchers, while also enabling the reproduction and validation of prior work.

Fault and Event Injection: Introducing controlled and reproducible faults or events (e.g., packet loss, delay, congestion) is challenging on real hardware platforms.

Scalable Programmable Infrastructure for Network Experimentation

P4 Developer Days 🕒 (10-Dec, 2025)

Context

Fidelity and Scalability: Experiments must combine high fidelity (running on real hardware and devices) with scalability (supporting the size and complexity of real-world topologies, such as data centers).

Reproducibility: Ensuring that experiments and results can be reproduced by other researchers, while also enabling the reproduction and validation of prior work.

Fault and Event Injection: Introducing controlled and reproducible faults or events (e.g., packet loss, delay, congestion) is challenging on real hardware platforms.

Scalable Programmable Infrastructure for Network Experimentation

Can we solve all these challenges?

P4 Developer Days 🕒 (10-Dec, 2025)

Context

Fidelity and Scalability: Experiments must combine high fidelity (running on real hardware and devices) with scalability (supporting the size and complexity of real-world topologies, such as data centers).

- **Up to 15x for switches and 8x for SmartNICs**

Reproducibility: Ensuring that experiments and results can be reproduced by other researchers, while also enabling the reproduction and validation of prior work.

- **Platform able to emulate customized topologies according to the specification**

Fault and Event Injection: Introducing controlled and reproducible faults or events (e.g., packet loss, delay, congestion) is challenging on real hardware platforms.

- **Ability to introduce network metrics like latency/losses in a scripted way**

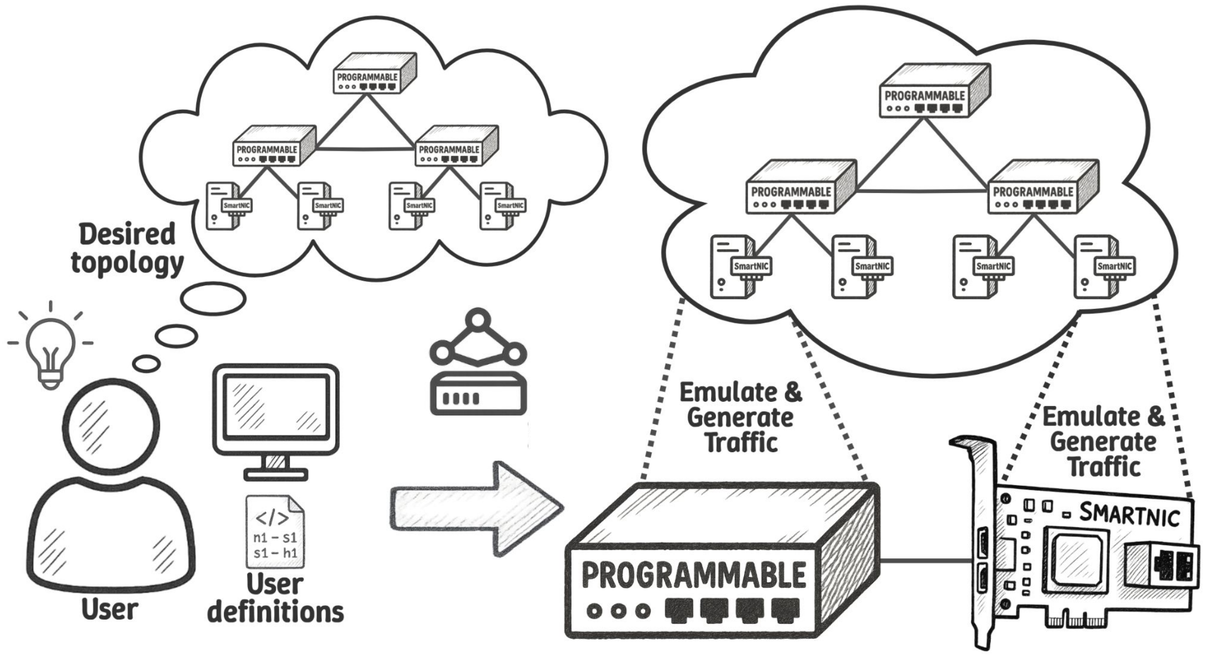
Scalable Programmable Infrastructure for Network Experimentation

Can we solve all these challenges?

No, but it will help advance in that direction!

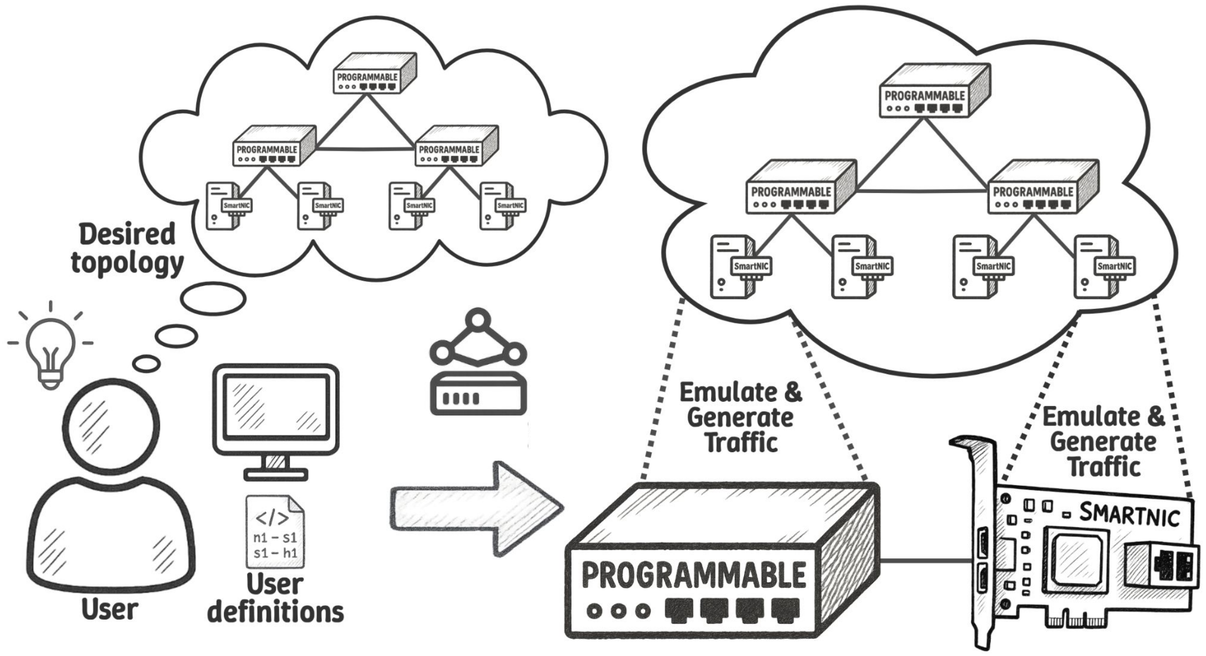
P4 Developer Days 🕒 (10-Dec, 2025)

Overview



P4 Developer Days 🕒 (10-Dec, 2025)

Overview

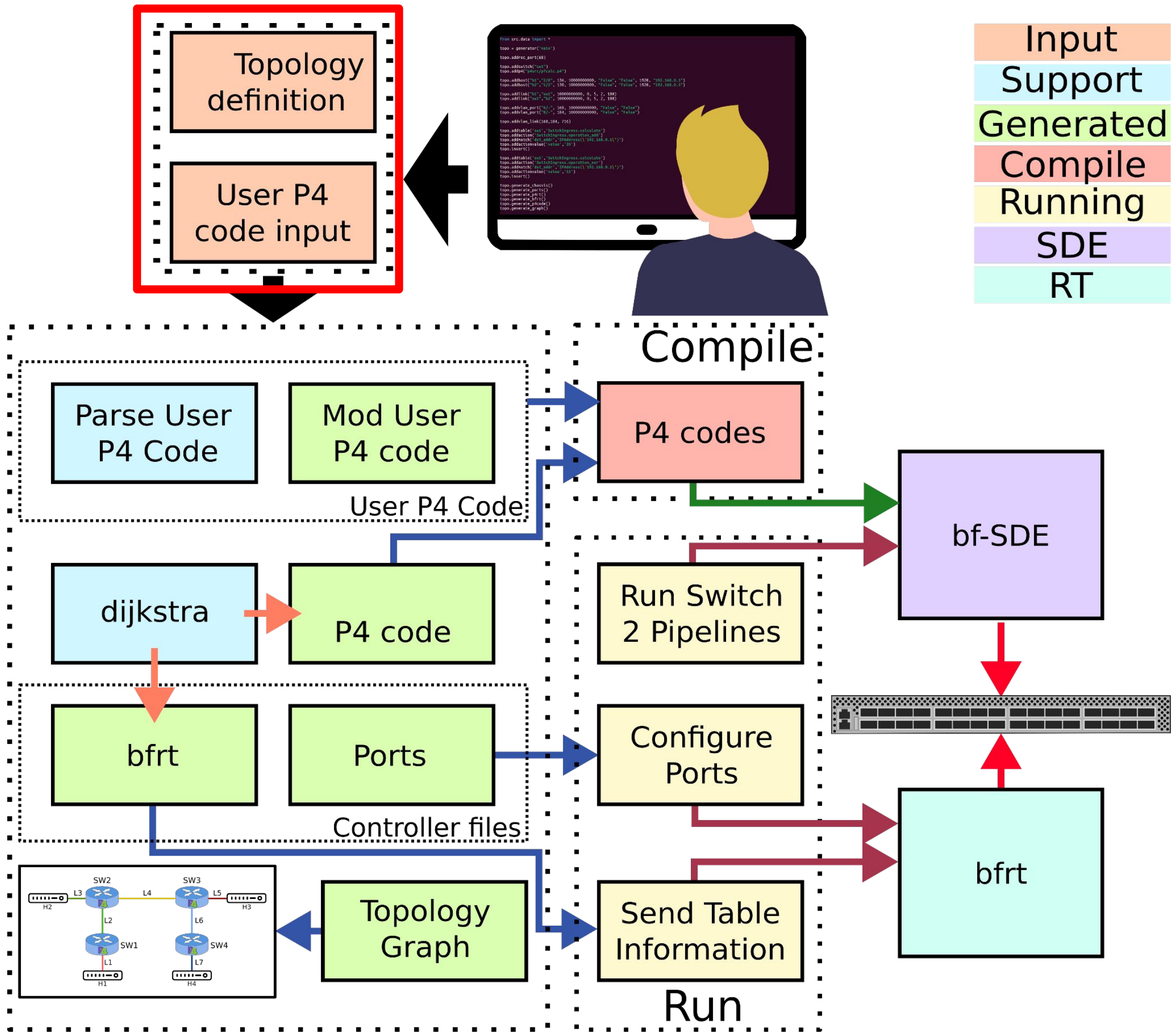


Available features:

- Switch emulation;
  - User-defined P4 Codes
- SmartNIC emulation;
  - User-defined DOCA applications
- Links emulation;
- Link metrics/events:
  - Latency (fixed);
  - Jitter (fixed and model);
  - Packet loss (fixed and model);
  - Bandwidth (fixed);
- Traffic generation:
  - Up to 8 different flows, supporting different protocols and IPGs.

P4 Developer Days (10-Dec, 2025)

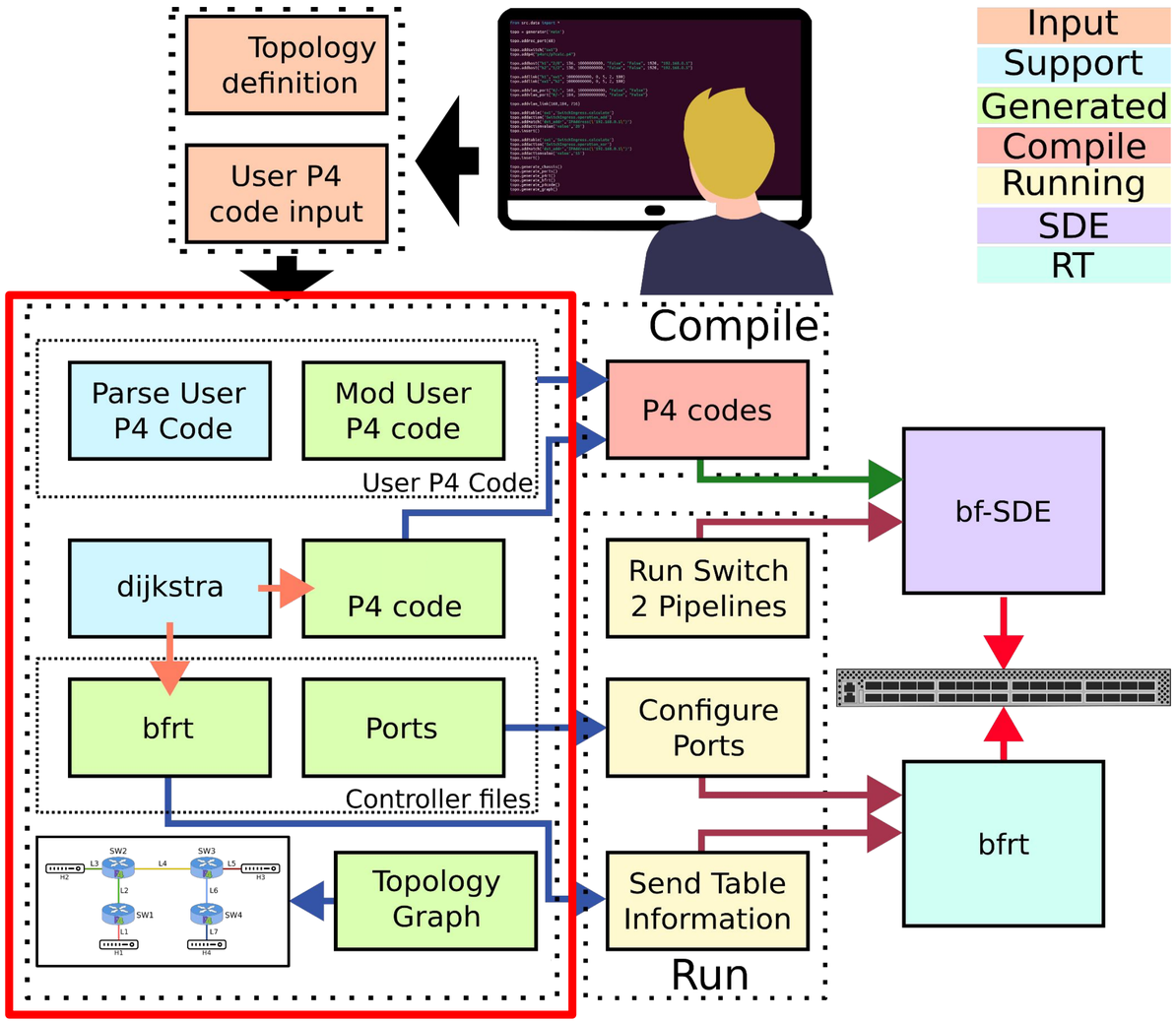
Architecture



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

P4 Developer Days (10-Dec, 2025)

Architecture

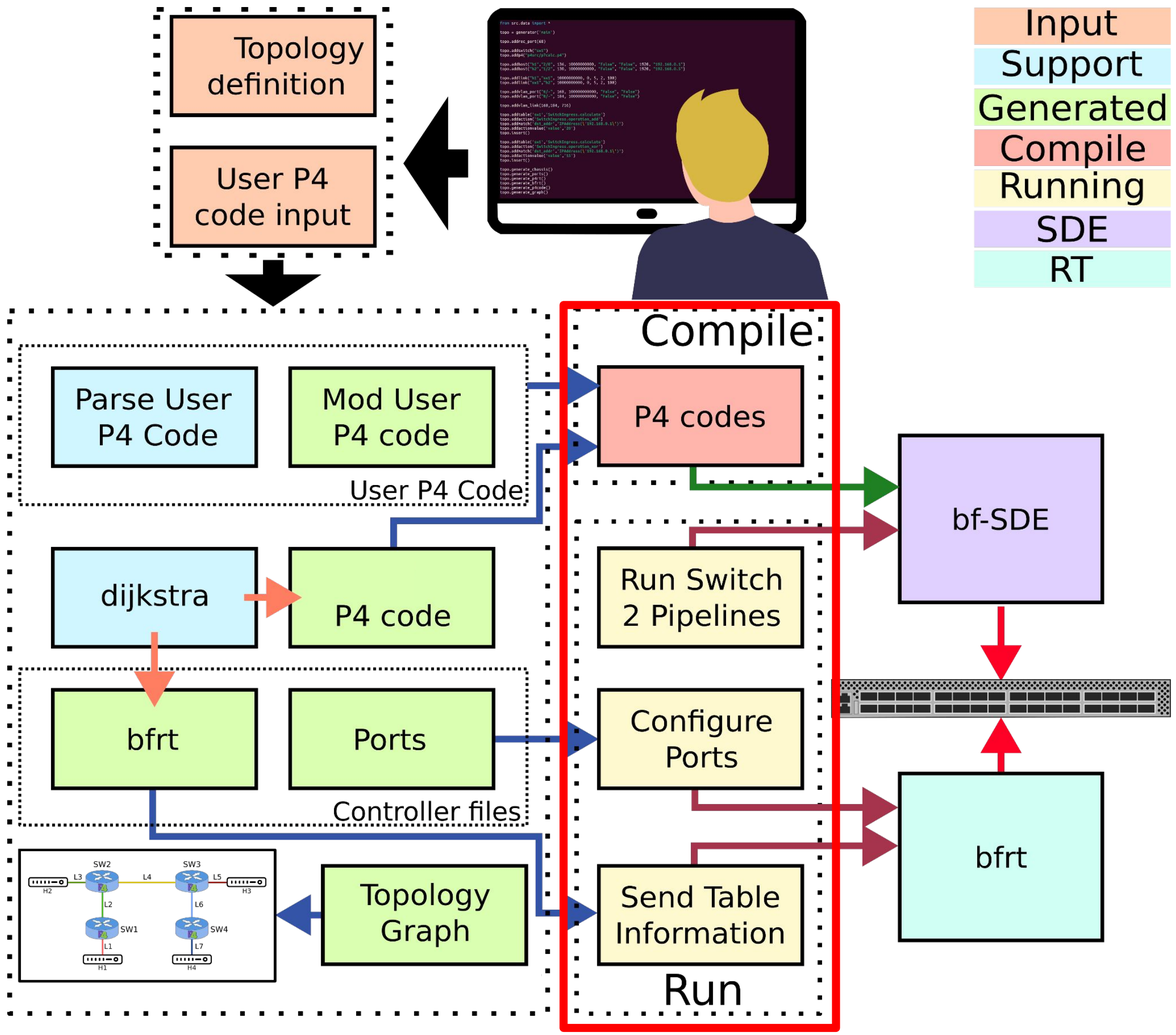


The emulator processes the data and generates the necessary files:

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

P4 Developer Days (10-Dec, 2025)

Architecture



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

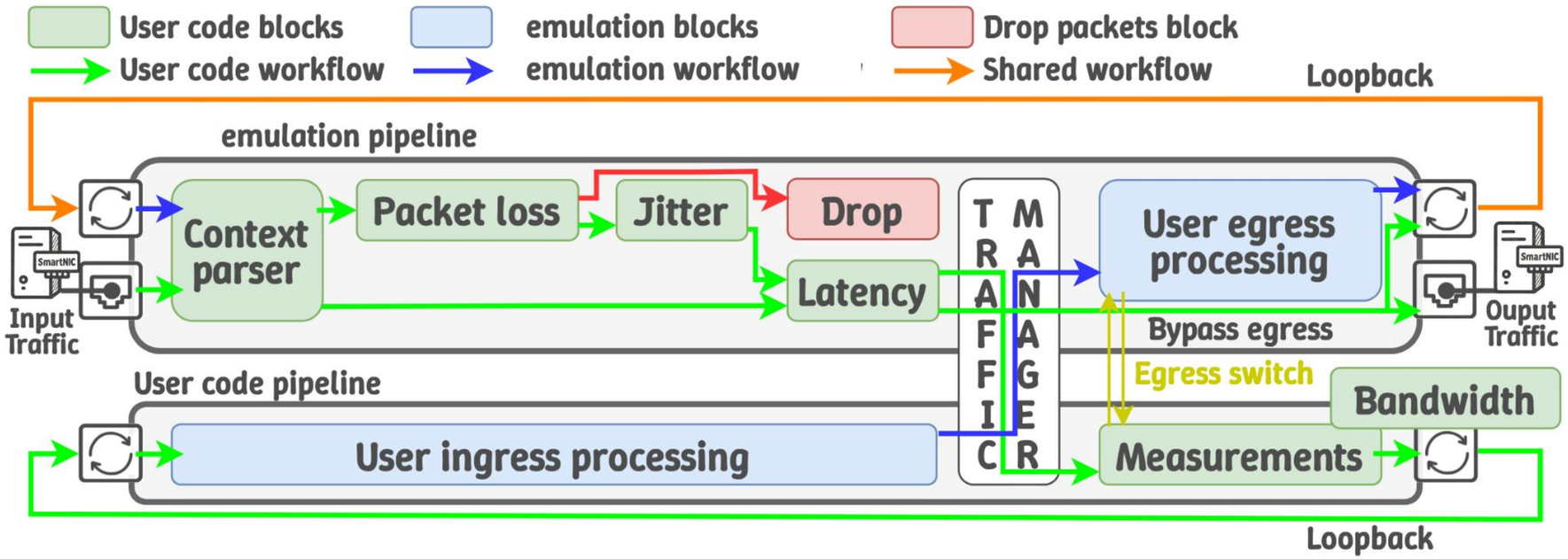
P4 Developer Days 🕒 (10-Dec, 2025)

Switch and links emulation

Telefónica

SMARTNESS

Tofino pipeline:



P4 Developer Days 🕒 (10-Dec, 2025)

Switch and links emulation

Telefónica

SMARTNESS

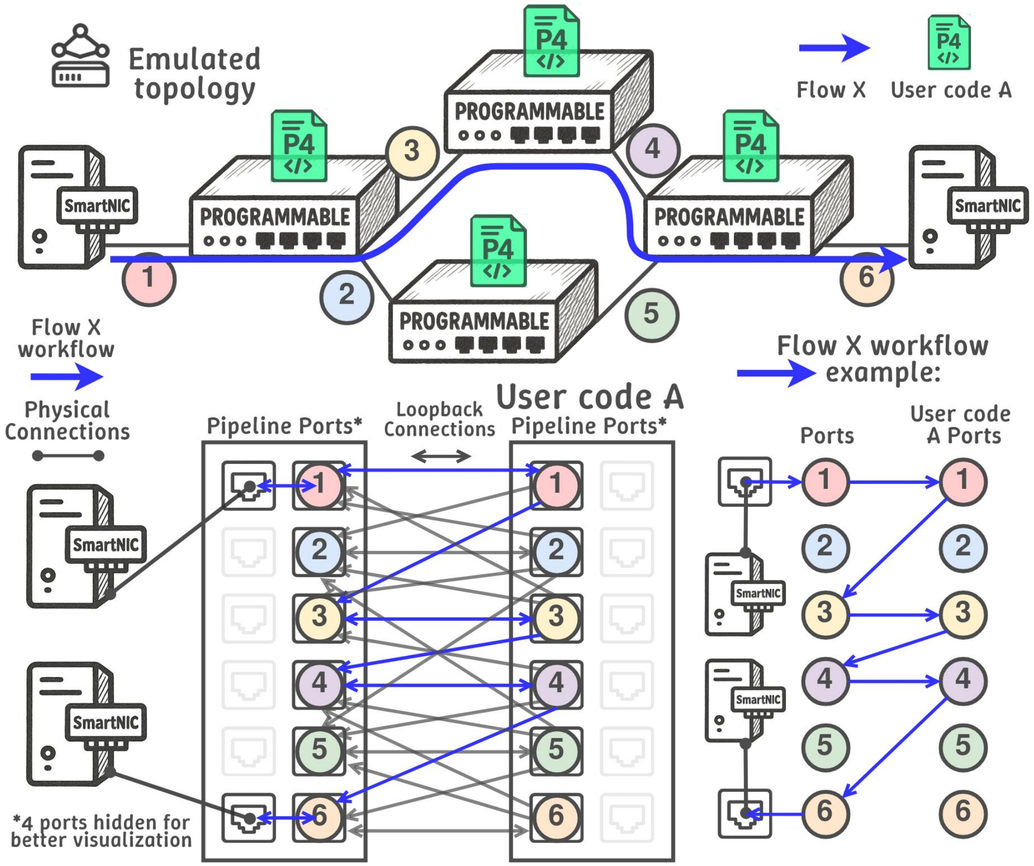
2-Pipes Tofino:

4-Pipes Tofino:

P4 Developer Days (10-Dec, 2025)

Switch and links emulation

2-Pipes Tofino:

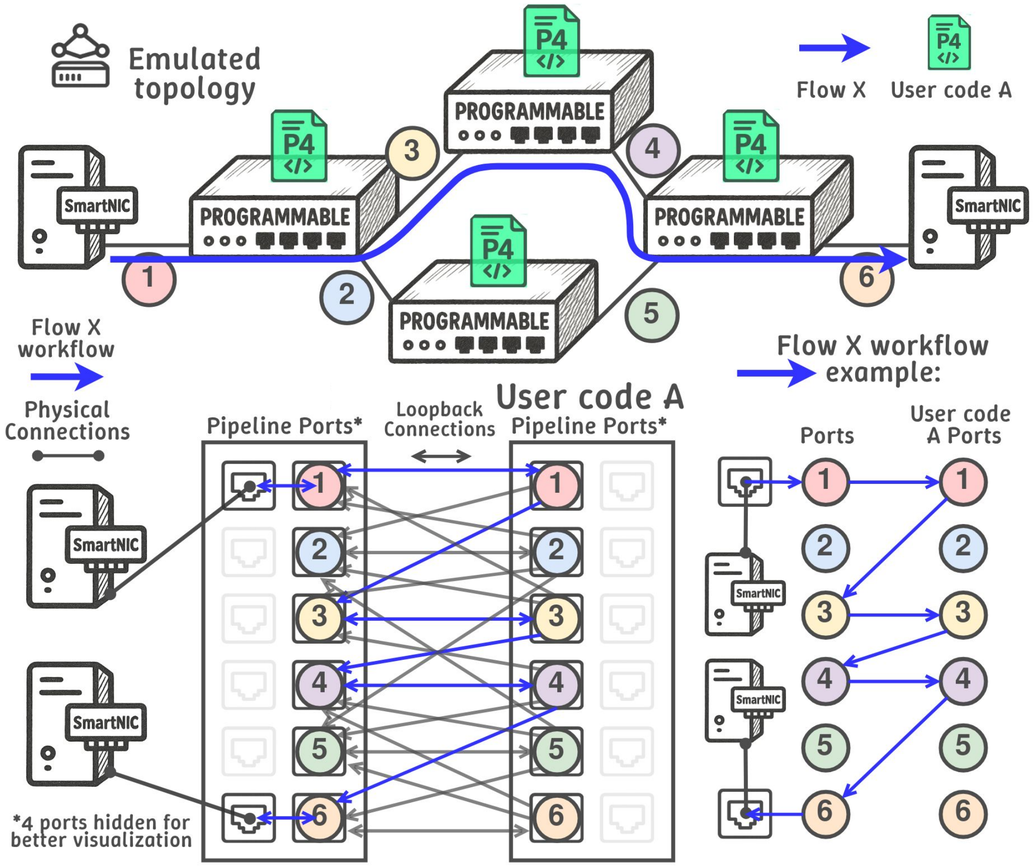


4-Pipes Tofino:

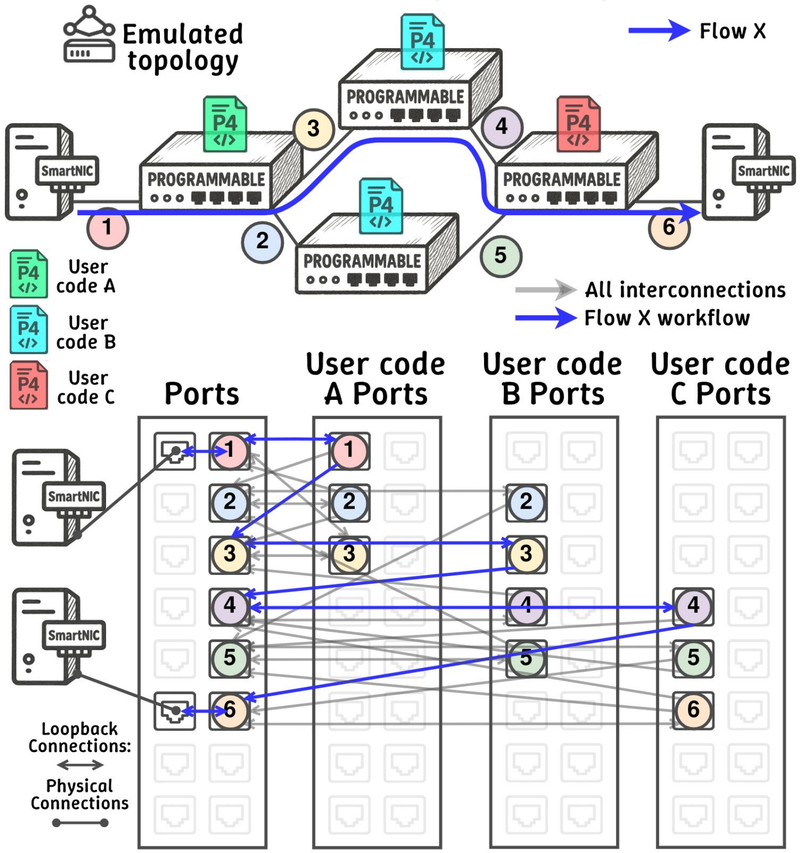
P4 Developer Days (10-Dec, 2025)

Switch and links emulation

2-Pipes Tofino:



4-Pipes Tofino:



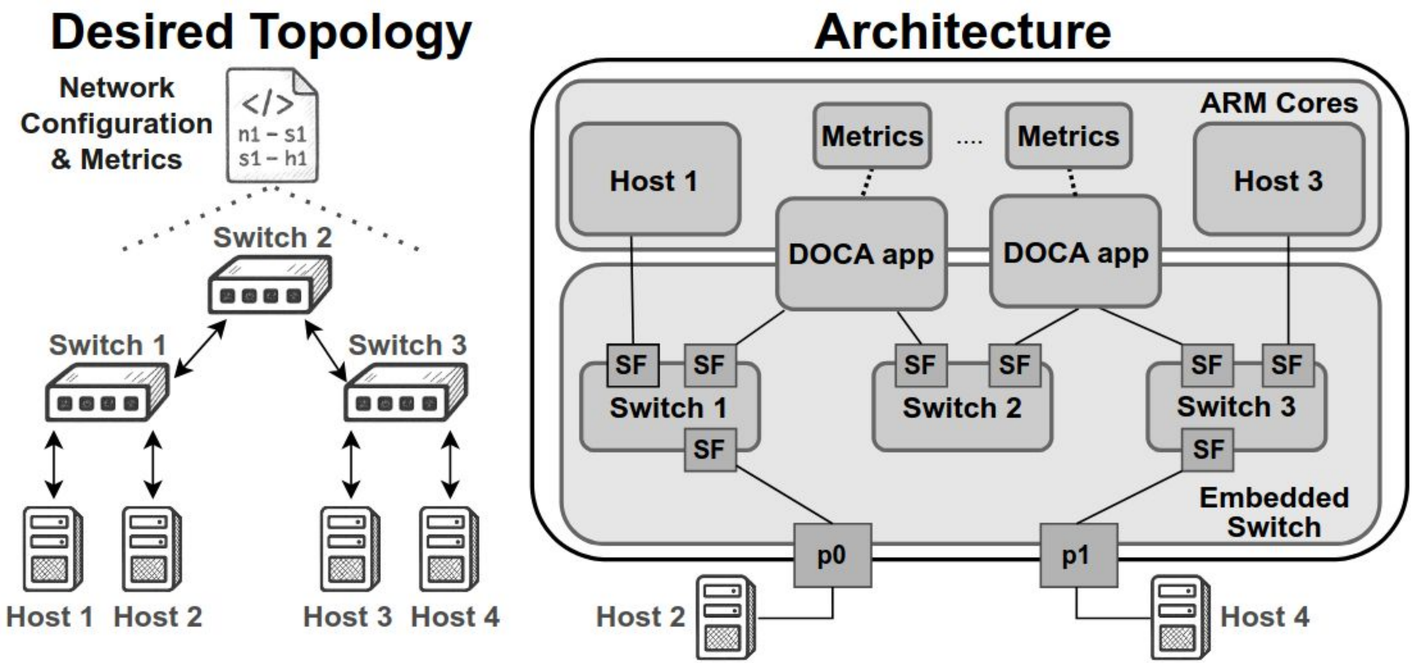
P4 Developer Days (10-Dec, 2025)

SmartNIC emulation

Telefónica

SMARTNESS

Bluefield-2 pipeline:



P4 Developer Days 🕒 (10-Dec, 2025)

Environment definition

```
from src.data import *

topo = generator('main', sys.argv[1:])

topo.addrec_port(196)
topo.addrec_port_user(68)

topo.addports_file("portConfigs.txt")

topo.tofinoVersion(1) #default is 1

#topo.definePipelines("p1", "spine", "p1", "spine")
topo.definePipelines("p1", "spine")

#Register shift
topo.defineRegisterMaxSize(1000)

topo.addp4("p1", "p4src/simple_forward.p4")

topo.routing(2) #type for user defined routing, 0 for default routing
```

P4 Developer Days 🕒 (10-Dec, 2025)

Environment definition

```
22  # addswitch(name, p4code)
23  topo.addswitch("sw1", "p1")
24  topo.addswitch("sw2", "p1")
25  topo.addswitch("sw3", "p1")
26
27  topo.addhost("h1","5/0", 164, 100000000000, "False", "False", 1920, "192.168.0.10")
28  topo.addhost("h2","6/0", 172, 100000000000, "False", "False", 1920, "192.168.0.20")
29
30  topo.addlink("h1","sw1", 100000000000, 0, 0, 0, 100)
31  topo.addlink("sw1","sw2", 100000000000, 0, 0, 0, 100)
32  topo.addlink("sw2","sw3", 100000000000, 0, 0, 0, 100)
33  topo.addlink("sw3","h2", 100000000000, 0, 0, 0, 100)
```

P4 Developer Days 🕒 (10-Dec, 2025)

Environment definition

```
36    # add table entry sw1
37    topo.addtable('sw1','SwitchIngress.forward')
38    topo.addaction('SwitchIngress.send')
39    topo.addmatch('dst_addr','IPAddress(\'192.168.0.20\')')
40    topo.addactionvalue('port','140')
41    topo.addactionvalue('sw', '1')
42    topo.insert()
43
44
45    # add table entry sw1
46    topo.addtable('sw1','SwitchIngress.forward')
47    topo.addaction('SwitchIngress.send')
48    topo.addmatch('dst_addr','IPAddress(\'192.168.0.10\')')
49    topo.addactionvalue('port','132')
50    topo.addactionvalue('sw', '0')
51    topo.insert()
```

P4 Developer Days 🕒 (10-Dec, 2025)

Implementations and limitations

- We implemented the pre-processing in python, and the switch/SmartNIC codes in P4 and DOCA, respectively.

- Our tool auto-generates all codes/configuration files according to the desired topology.

Limitations:

- The user switch P4 code need to be edited, and has a maximum limit of registers/table entries that it can use.
- Latency and jitter can impact significantly in the throughput.
- DOCA applications are limited to simple applications using a single core.

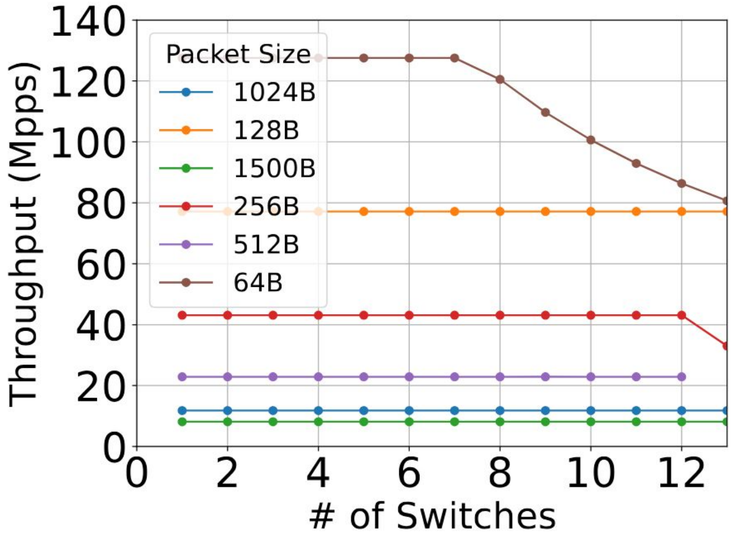
P4 Developer Days 🕒 (10-Dec, 2025)

Scalability and components limits

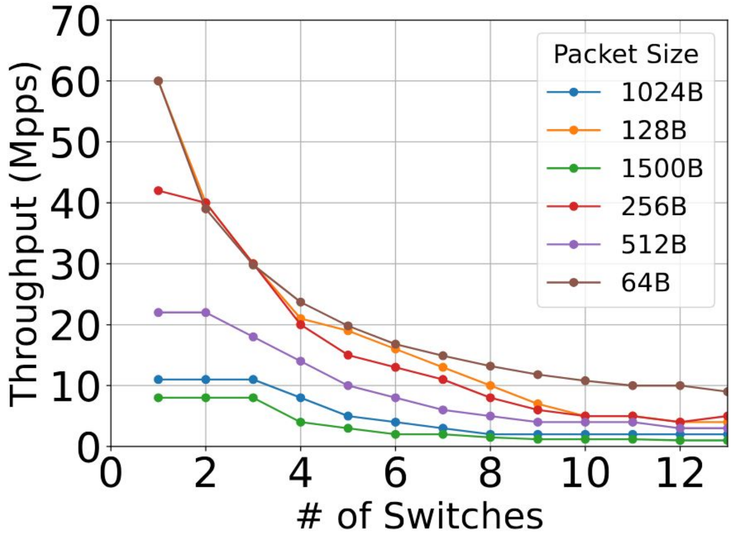
Telefónica

SMARTNESS

Throughput according to the number of switches and packet sizes:



Tofino



Bluefield-2

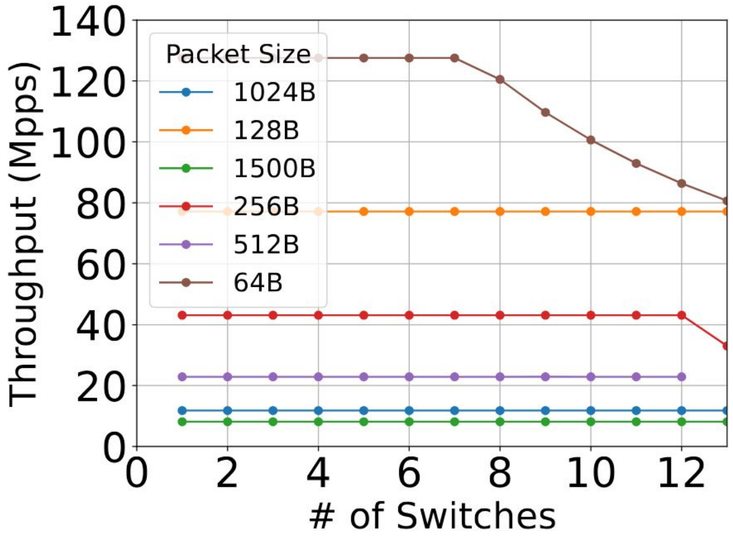
P4 Developer Days 🕒 (10-Dec, 2025)

Scalability and components limits

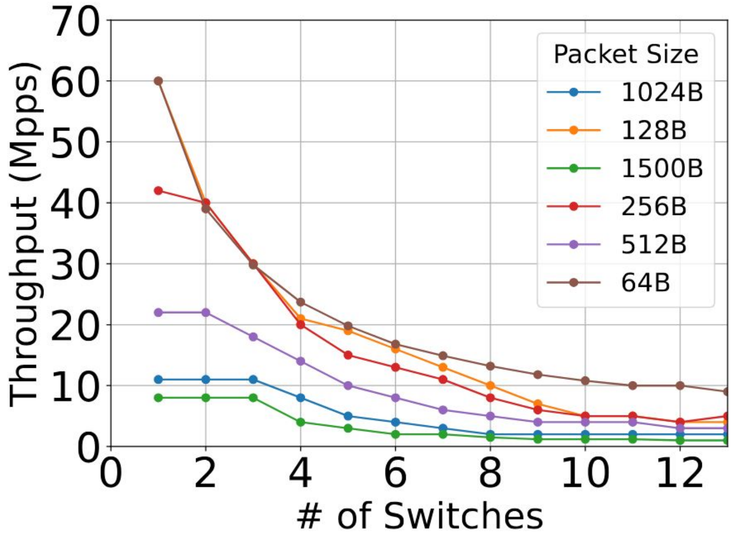
Telefónica

SMARTNESS

Throughput according to the number of switches and packet sizes:

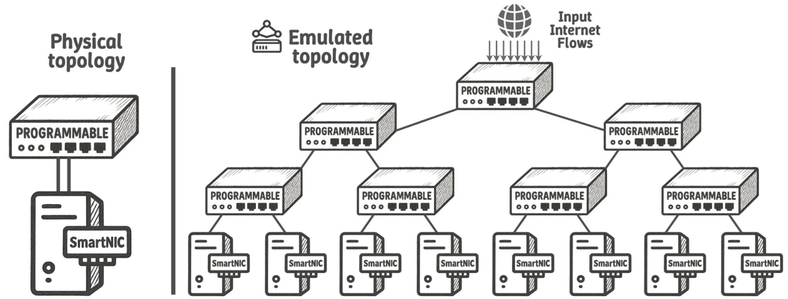


Tofino



Bluefield-2

Topology to be tested:



P4 Developer Days 🕑 (10-Dec, 2025)

Evaluation goals

Scalability and components limits:

Practical applicability:

P4 Developer Days 🕑 (10-Dec, 2025)

Evaluation goals

Scalability and components limits:

- Evaluate the performance limits of each functionality (switch and SmartNIC emulation, links and metrics, etc).
- Validate the maximum size topology in different scenarios.

Practical applicability:

P4 Developer Days 🕒 (10-Dec, 2025)

Evaluation goals

Scalability and components limits:

- Evaluate the performance limits of each functionality (switch and SmartNIC emulation, links and metrics, etc).
- Validate the maximum size topology in different scenarios.

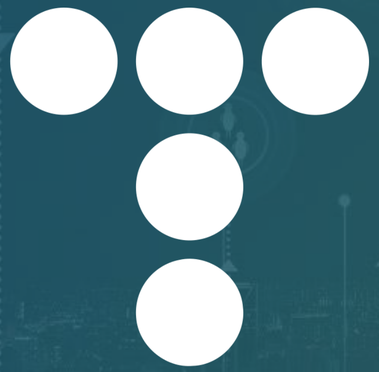
Practical applicability:

- Reproduce common examples (simple switch, in-band network telemetry, datacenter topology, etc).
- Reproduce works from recent high-impact conferences and its results in different scenarios:
  - Works that can benefit from increased experiment scale.
  - Works that can benefit from events/metrics insertion.

P4 Developer Days 🕒 (10-Dec, 2025)

Conclusions

- We introduced a programmable hardware emulation environment that combines Tofino switches and BlueField-2 SmartNICs to emulate realistic, scalable topologies.
- The platform bridges fidelity and scalability, enabling experiments on real hardware while emulating large multi-switch / multi-SmartNIC networks with controllable link metrics (latency, jitter, loss, reordering).
- It auto-generates P4, DOCA and configuration files from a high-level topology description, lowering the barrier to test new in-network applications and data-plane ideas.
- The environment supports reproducible experiments and fault/event injection, which is key to validating prior work and exploring “what-if” scenarios at scale.
- Ongoing work: push larger topologies, richer DOCA apps and more complex traffic/QoE use cases, and open it for collaboration with the P4 community.



Telefónica

Programmable Hardware Emulation Environment for Realistic and Scalable Network Testing

Fabricio Rodriguez
Telefonica Research

Francisco Germano Vogt
Universidade Estadual de Campinas (UNICAMP)

SMARTNESS

Telefónica

Github Repository:

github.com/intrig-unicamp

Contact:

Fabricio:
fabricio.rodriguezcesen@telefonica.com

Francisco:
f234632@dac.unicamp.br