



P₄PI: P₄ on Raspberry PI for Networking Education

Sándor Laki, Eötvös Loránd University (HU)

Dávid Kis, Eötvös Loránd University (HU)

Péter Vörös, Eötvös Loránd University (HU)

Robert Soulé, Yale University (US)

Noa Zilberman, Oxford University (UK)

Teaching Computer Networks



- P4 is ideal for teaching computer networks
 - Quick to learn, easy to use
- Hands on experience
 - Increases students' engagement
 - Improves student's understanding
- Software solutions (e.g., BMv2 + Mininet)
 - Can't build a network of students' devices
 - Don't provide real-world experience
 - Not so exciting!

Existing Hardware Platforms



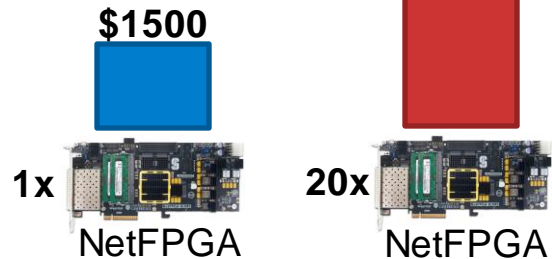
- **High cost**
- **Closed-source (some)**
- **Long learning curve**
- **Availability**

Teaching P4 Using Hardware - Today

- **P4 programmable switch-ASIC**
 - Too expensive for many (>1000's of \$)
 - Closed / partially closed source
- **NetFPGA**
 - Cost ~\$1500
 - Open-source
 - Requires FPGA design knowledge
- **SmartNICs**
 - Cost \$1000-\$2000
 - Closed source
 - Requires micro-architecture knowledge

Equipping a class is expensive

\$\$\$



What Is The Ideal Platform For Teaching?

- **Low cost**
 - Less than \$100
- **Easy to learn**
 - And easy to use
- **High Availability**
 - Worldwide, in-stock
 - Long term support
- **Open-source**
 - Both hardware and software
- **Training resources available**
- **Wireless + Wired connectivity**
 - Students can use their laptops
 - ...or existing lab machines
- **Network performance**
 - "Home" level

P4P



RaspberryPI 4 Model B 4GB

Quad-core ARM64 processor
On-board WiFi, 1GbE RJ-45
\$60-\$80, availability

+



T4P4S P4 Compiler

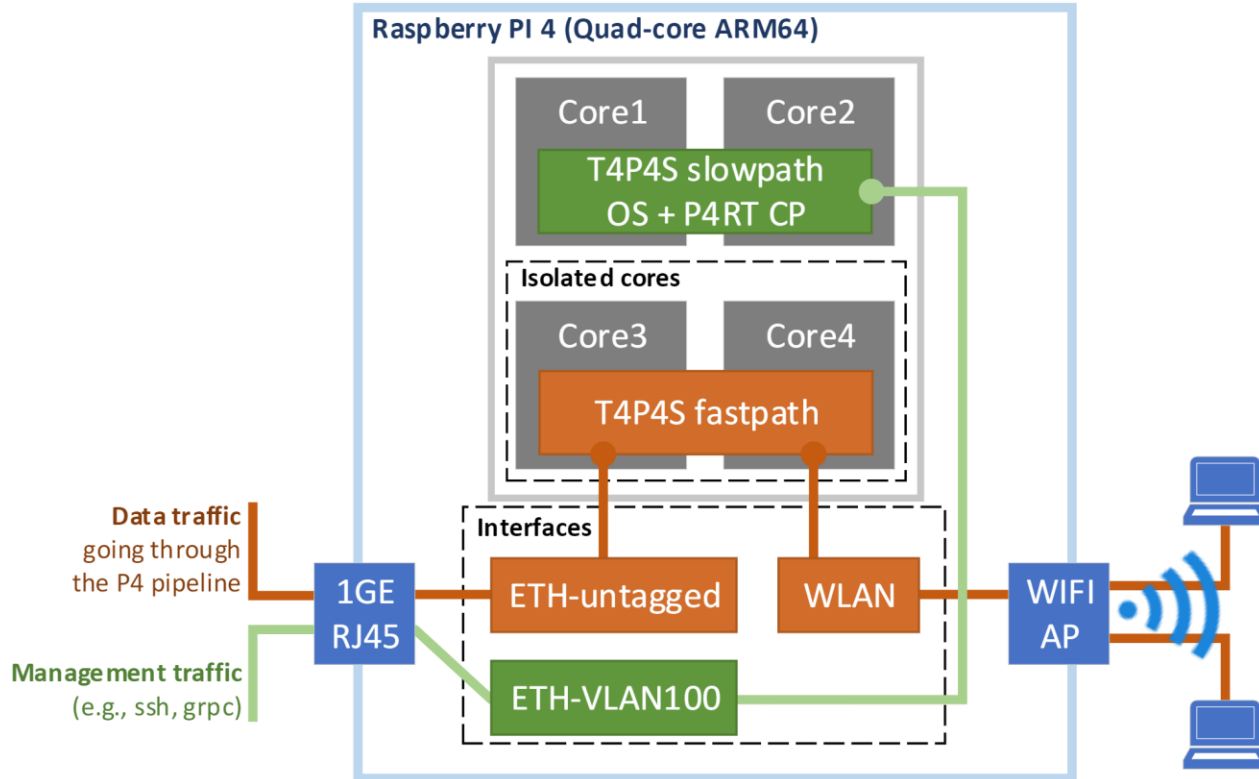
Open source, supports multiple
backends incl. DPDK &
v1 model

P4PI Is More Than a Hardware Platform



- **Training Materials**
 - Tutorials, exercises etc.
- **P4PI Repository**
 - P4PI source code, wiki, tools, reference programs.
- **Community engagement**
- A P4 Education Workgroup project

P4PI Reference Architecture



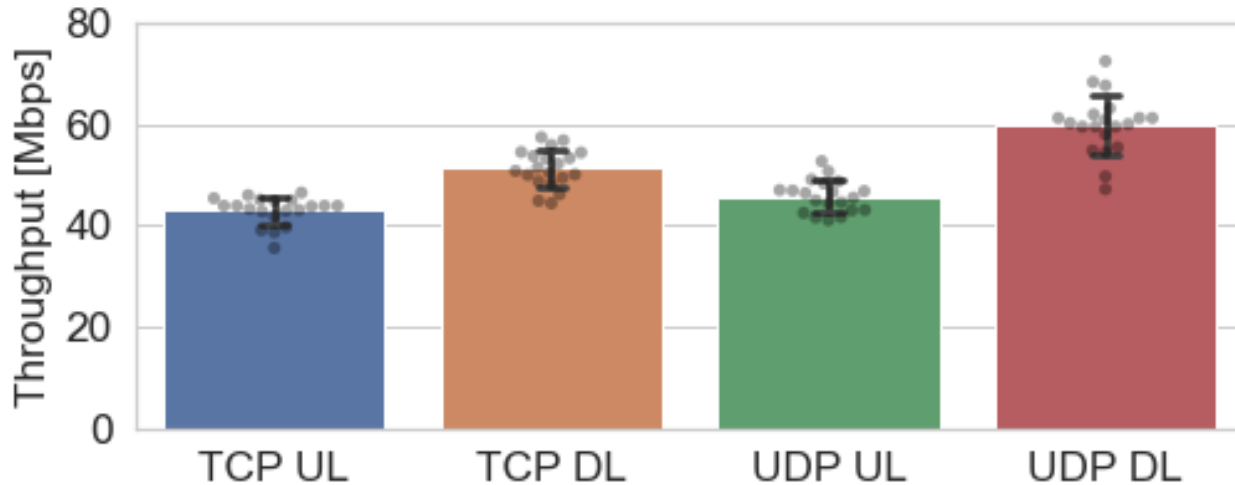
P4PI Reference Architecture

- **T4P4S using DPDK backend**
 - Packet processing on 2 isolated CPU cores
 - Executing the P4 program
 - Using TUN/TAP PMDs
- **WLAN interface in AP mode**
- **Separate management interface**
 - e.g., for SSH
- **Integration with standard networking tools**
 - E.g., DHCP, NAT with iptables, etc.

P4PI Performance Over WiFi (Preliminary)

- **Setup**

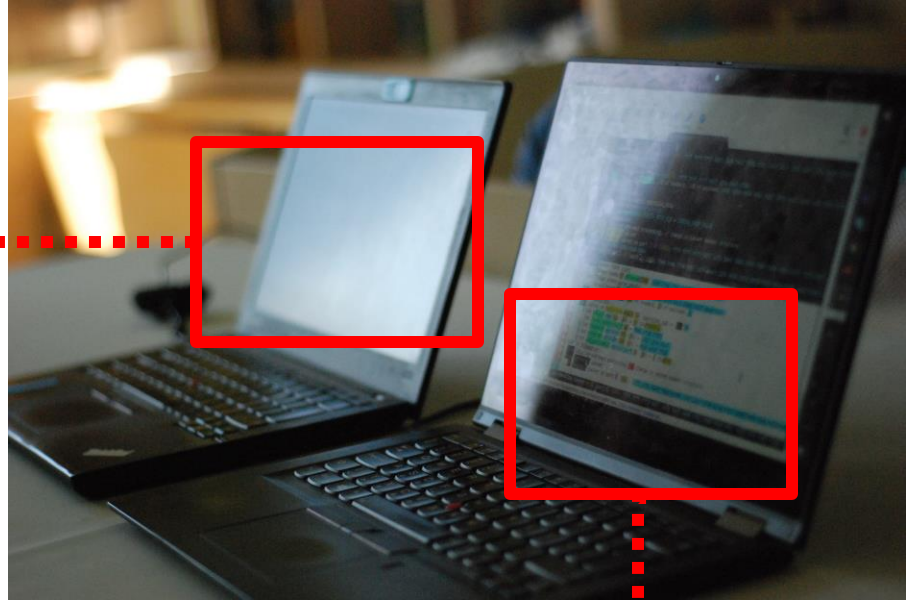
- Laptop + RPi located in different rooms (~5m distance)
- Using iperf3 - upstream & downstream, TCP and UDP
- PortForwarding P4 program using T4P4S and tun/tap pmd



Using P4PI In Class

- **Option 1: A P4PI node per student**
 - Students working independently
 - One laptop per node, using ssh for access and sending test traffic
- **Option 2: A P4PI node per group of students**
 - Still a single network device
 - ... But more complex test scenarios can be implemented
- **Option 3: Multiple P4PI nodes, creating a network**
 - A class of students engaging and collaborating
 - Interoperability between students' projects
 - Implementing a more complex applications

DEMO: P4Calc on P4PI

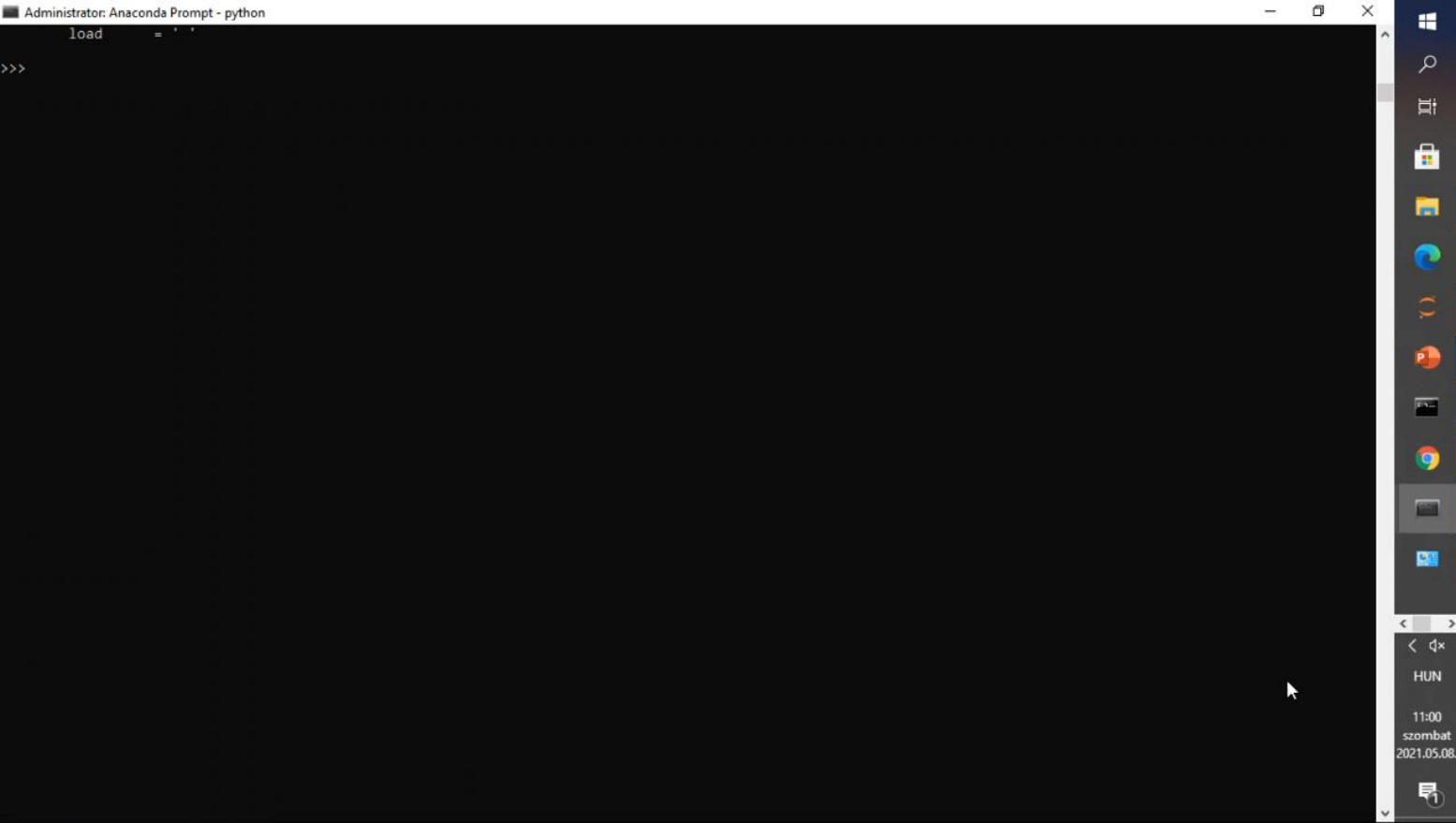


Setup

- P4PI
- Laptop 1 - Tester: Test traffic generator
- Laptop 2 - Ctrl: Control machine on the management network

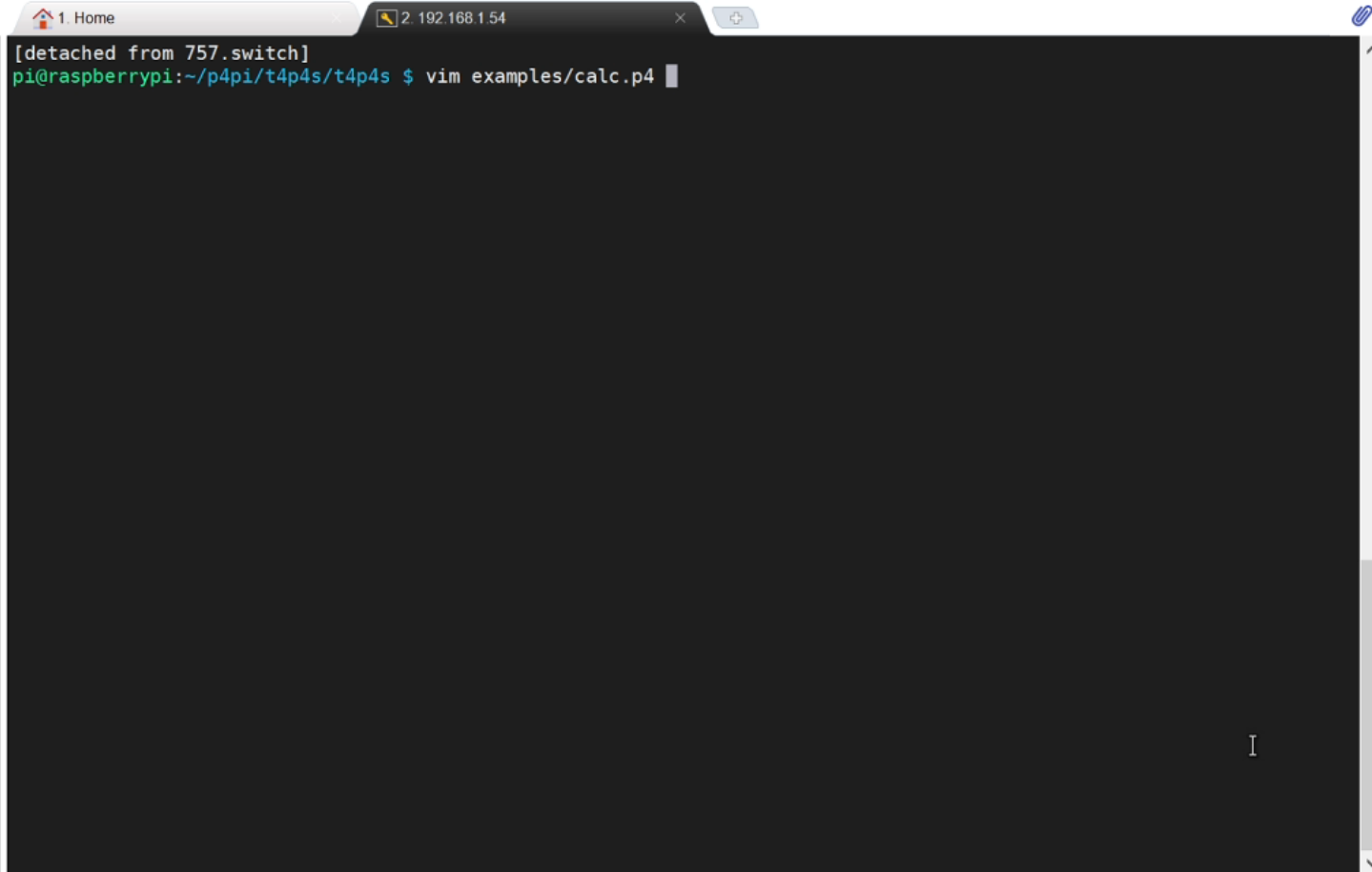
DEMO: P4Calc on P4PI

Step 1 – connecting Tester to P4PI
via WiFi



DEMO: P4Calc on P4PI

Step 2 – executing the P4 program
on P4PI

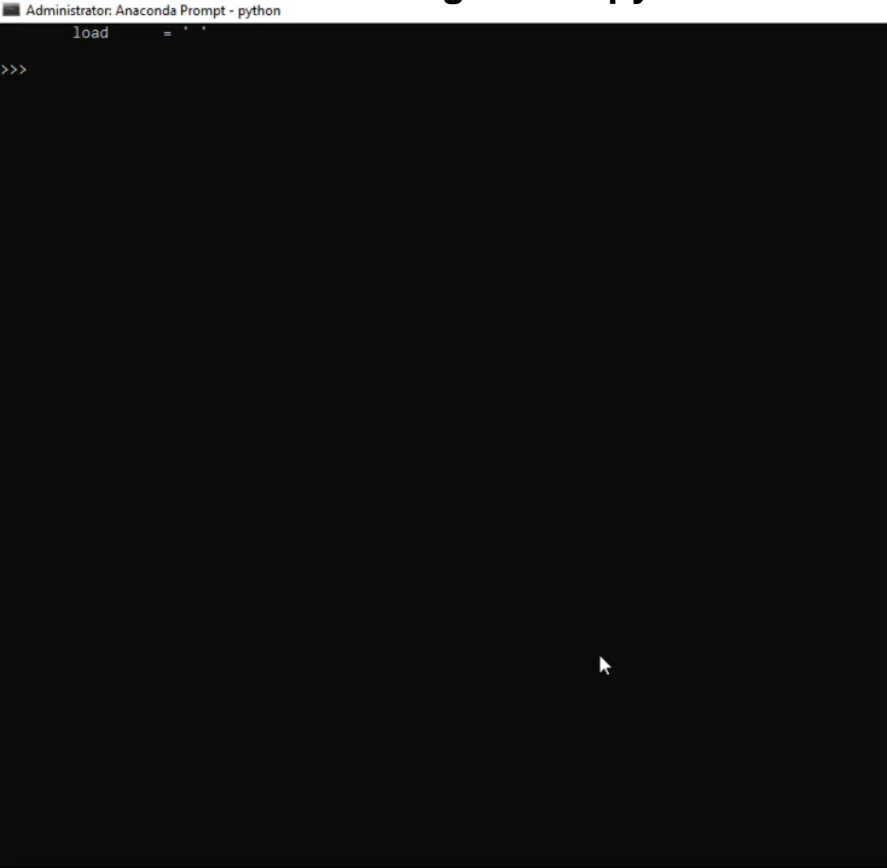


```
[detached from 757.switch]
pi@raspberrypi:~/p4pi/t4p4s/t4p4s $ vim examples/calc.p4
```

SSH session
from Ctrl

DEMO: P4Calc on P4PI

Tester – sends P4Calc messages
to P4PI using the Scapy lib



Step 3 – sending test traffic

Ctrl - Tcpcdump on the wlan0 interface of P4PI
to check the P4Calc messages (ethertype 0x1234)

```
[detached from 757.switch]
pi@raspberrypi:~/p4pi/t4p4s/t4p4s $ ./connect_dtaps.sh
Connecting dtap0
Connecting dtap1
Done.
pi@raspberrypi:~/p4pi/t4p4s/t4p4s $ sudo tcpdump -i wlan0 ether proto 0x1234
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wlan0, link-type EN10MB (Ethernet), capture size 262144 bytes
|
```

DEMO: P4Calc on P4PI

Step 4 – In verbose mode, T4P4S provides details on packet processing

```
e32 3535 2e32 3530 3a31 3930 300d 0a4d 414e ...
2@0 Control MyIngress...
2@0 ~~~~ Action MyIngress.operation_drop
2@0 : Called extern mark_to_drop
2@0 : all_metadata.EGRESS_META_FLD = EGRESS_DROP_VALUE
2@0 Control MyDeparser...
2@0 :::: Skipping pre-emit processing: no change in packet header structure
2@0 <<<< Egressing packet
2@0 << Emitting packet on port 2148 (215B): (showing 80B) 0100 5e7f fffa 6057 187e 8a47 0800 4500 00c9 961c 0000
0111 6e62 c0a8 0403 efff fffa c528 076c 00b5 4502 4d2d 5345 4152 4348 202a 2048 5454 502f 312e 310d 0a48 4f53 543a 20
32 3339 2e32 3535 2e32 3535 ...
2@0 Handling packet #-01 (port 0, 215B): (showing 80B) 0100 5e7f fffa 6057 187e 8a47 0800 4500 00c9 961d 0000 0111 6
e61 c0a8 0403 efff fffa c528 076c 00b5 4502 4d2d 5345 4152 4348 202a 2048 5454 502f 312e 310d 0a48 4f53 543a 2032 333
9 2e32 3535 2e32 3535 ...
2@0 %%% Parser state start
2@0 :: Parsed header#0 ethernet/14B: 0100 5e7f fffa 6057 187e 8a47 0800
2@0 %%% Packet is accepted, 14B of headers, 201B of payload: (showing 80B) 4500 00c9 961d 0000 0111 6e61 c0a8 0403
efff fffa c528 076c 00b5 4502 4d2d 5345 4152 4348 202a 2048 5454 502f 312e 310d 0a48 4f53 543a 2032 3339 2e32 3535 2
e32 3535 2e32 3530 3a31 3930 300d 0a4d 414e ...
2@0 Control MyIngress...
2@0 ~~~~ Action MyIngress.operation_drop
2@0 : Called extern mark_to_drop
2@0 : all_metadata.EGRESS_META_FLD = EGRESS_DROP_VALUE
2@0 Control MyDeparser...
2@0 :::: Skipping pre-emit processing: no change in packet header structure
2@0 <<<< Egressing packet
2@0 << Emitting packet on port 2148 (215B): (showing 80B) 0100 5e7f fffa 6057 187e 8a47 0800 4500 00c9 961d 0000
0111 6e61 c0a8 0403 efff fffa c528 076c 00b5 4502 4d2d 5345 4152 4348 202a 2048 5454 502f 312e 310d 0a48 4f53 543a 20
32 3339 2e32 3535 2e32 3535 ...
2@0 Handling packet #-01 (port 0, 215B): (showing 80B) 0100 5e7f fffa 6057 187e 8a47 0800 4500 00c9 961e 0000 0111 6
e60 c0a8 0403 efff fffa c528 076c 00b5 4502 4d2d 5345 4152 4348 202a 2048 5454 502f 312e 310d 0a48 4f53 543a 2032 333
9 2e32 3535 2e32 3535 ...
2@0 %%% Parser state start
2@0 :: Parsed header#0 ethernet/14B: 0100 5e7f fffa 6057 187e 8a47 0800
2@0 %%% Packet is accepted, 14B of headers, 201B of payload: (showing 80B) 4500 00c9 961e 0000 0111 6e60 c0a8 0403
efff fffa c528 076c 00b5 4502 4d2d 5345 4152 4348 202a 2048 5454 502f 312e 310d 0a48 4f53 543a 2032 3339 2e32 3535 2
```


P4PI - Use Case Examples

- Simple applications (e.g., P4Calc)
- Load balancing
- Tunneling
- Firewall
- Network telemetry
- IoT applications (e.g., sensor data processing)
- **Don't use just one – build a network!**

Summary

- P4PI platform is **currently under development**
 - To be released this summer
 - <https://github.com/p4lang/p4pi>
- Developed **for networking education**
 - Cheap, available and open
 - Hands-on experience
- Making P4 available **for hobbyists**
 - Everyone need a P4 home gateway / AP
 - ... and other crazy ideas



Hackathon at SIGCOMM 2021

Join us!

- August 23rd or 27th (TBD)
- P4 and P4PI tutorials
- **Educators track** - Practicing and developing teaching materials
- **Contributors track** - Porting projects to P4PI, improving tools etc.
- **Hackers track** - Exploring new use cases and cool ideas using P4PI
- Will provide Raspberry PI platforms to registered participants (limited)

MAY 18-20
2021 P4
Workshop
Hosted by ONF



Thank You

Email: lakis@inf.elte.hu

P4PI: coming soon