Securing the Internet and 5G Infrastructure
The network as a programmable platform

Nick McKeown
Stanford University

With: Nate Foster, Jen Rexford, Guru Parulkar, Larry Peterson, Oğuz Sunay
and the whole Pronto team
1: A change of who is in control

Closed & proprietary
Bloated & power hungry
Little incentive to improve
“Internet ossification”

Networks hard to manage
Unreliable, hard to secure
Hard to scale

Homegrown
Merchant silicon
Linux based
Open source

Easier to fix
Add new features:
• Traffic engineering
• Faster recovery
• Intrusion detection
Faster innovation

Phase 1
Network owners take control of their software

2020

2005

2020

Compute mainframes
Network mainframes
Whitebox Switch
Server running control plane

1: A change of who is in control

Phase 1
Network owners take control of their software

Phase 2
Network owners take control of packet processing too
2: Taking control of packet processing, too

Fixed function switches and NICs

Programmable switches and smartNICs
Switch with fixed function pipeline
OSPF  BGP  etc.

Switch OS

New
Domain Specific Processors

Computers
- Java Compiler
- OpenCL Compiler
- Matlab Compiler
- TensorFlow Compiler

Graphics
- CPU
- GPU
- DSP
- TPU

Signal Processing
- Matlab

Machine Learning
- TensorFlow Compiler

Networking
- P4 Compiler
- PISA
“Programmable switches run 10x slower, consume more power and cost more.”

Conventional wisdom in 2010

Not true any more
3: The rise of open-source networking software

Phase 1
Network owners take control of their software

2005

Phase 2
Network owners take control of packet processing too

2020

Open source Networking Software
Open source has re-emerged as a legitimate and trustworthy way to control networks
e.g. OVS, SONiC, FBOSS, FRR, ONOS ...

• Network code that was opaque and closed has become transparent and open
**ONF: Open Networking Foundation**
- *Linux Foundation*
- *TIP: Telecom Infrastructure Project*

**Rise of Merchant Switch Chips**

1994
- IBM, Compaq, Dell run Linux

1996
- Linux v1

1998
- Linux v2

2008
- 1st DC with switch chip + Linux

2010
- 1st WAN with Switch chip + Linux

2011
- # Virtual Ethernet ports > # Physical ports

2012
- NEC + HP
- OCP Server
- OVS

2013
- OCP Switch
- ONIE
- ODL
- FBOSS

2015
- OVN
- OCP Wedge
- ONL
- ONOS
- OpenNFV

2016
- ONAP
- HP Open Switch
- ONAP
- SONIC
- P4 Runtime
- Danos
- Stratum
- Trellis
- ...

2018

**Key Events:**
- 1st DC with switch chip + Linux
- 1st WAN with switch chip + Linux
- # Virtual Ethernet ports > # Physical ports

**Projects and Technologies:**
- NEC + HP
- OCP Switch
- ONIE
- ODL
- FBOSS
- OVN
- OCP Wedge
- ONL
- ONOS
- OpenNFV
- ONAP
- HP Open Switch
- SONIC
- P4 Runtime
- Danos
- Stratum
- Trellis
- ...
So what happens next?
PROGRAMMABLE END-TO-END & TOP-TO-BOTTOM

- BGP
- Traffic Engineering
- Load Balancing
- Overlay Virtualization

SDN Controller

- DPDK
- XDP/eBPF

- User space
- Kernel stack

- NIC
- P4

- SDN Controller
- P4
- NIC
- P4
- SDN Controller
- P4
- NIC
- P4

- BGP
- Traffic Engineering
- Load Balancing
- Overlay Virtualization

- DPDK
- XDP/eBPF

- User space
- Kernel stack
We will think of a network as a programmable platform.

We will describe the desired behavior at the top, then partitioned, compiled and run across elements.

Cloud, ISP and 5G networks will be programmed and tailored locally.
Who better to improve the reliability and security of networks than their owners?

Fine-grain per-packet measurement will monitor function and performance.

Software engineering techniques will be used routinely: formal verification and on-the-fly checking of correctness.
Future networks will be programmed by many.
And operated by few.
Networks, for the first time, will be:
- Programmable end-to-end
- Specified top-to-bottom
- Defined entirely by software.

This creates new possibilities:
- To verify networks are “correct by construction”.
- To measure and validate, in real-time against the network specification.
- To correct bugs through closed-loop control.

Because of open-source, we (the research community) get to take part.
Networks, for the first time, will be
• Programmable end-to-end
• Specified top-to-bottom
• Defined entirely by software.

This creates new possibilities
• To verify networks are “correct by construction”.
• To measure and validate, in real-time against the network specification.
• To correct bugs through closed-loop control.

Because of open-source, we (the research community) get to take part.

5G is being redefined by software, too!
How it will happen

1. **Fine-grain telemetry**: Instrumenting *every* packet
2. **Verification**: Is every packet (and all state) “correct”?  
3. **Control**: If needed, update state and programs
Specified Behavior

Compile

Partition

Compile

Compile

Compile

SDN Controller

Kernel stack

User space

DPDK

XDP/eBPF

Overlay Virtualization

Traffic Engineering

Load Balancing

BGP

Compile

Compile

Compile

Specified Behavior
Initial Aether Deployments

Aether Platform + Closed-Loop Control

Fine-grain per-packet telemetry
How to find out more and get involved

**Research**
prontoproject.org

**Watch...**
1. Jen Rexford’s talk about closed loop control, and
2. Nate Foster’s talk about verification

**Aether Platform**
aetherproject.org

**Watch...**
Oguz Sunay and Larry’s Peterson’s talk (next)

**Join us...**
1. Campus: Deploy Aether edge and use it for research
2. Enterprise: Plan a trial for 5G connected edge cloud
3. Supply-chain: Join Aether open-source project
Thank you!