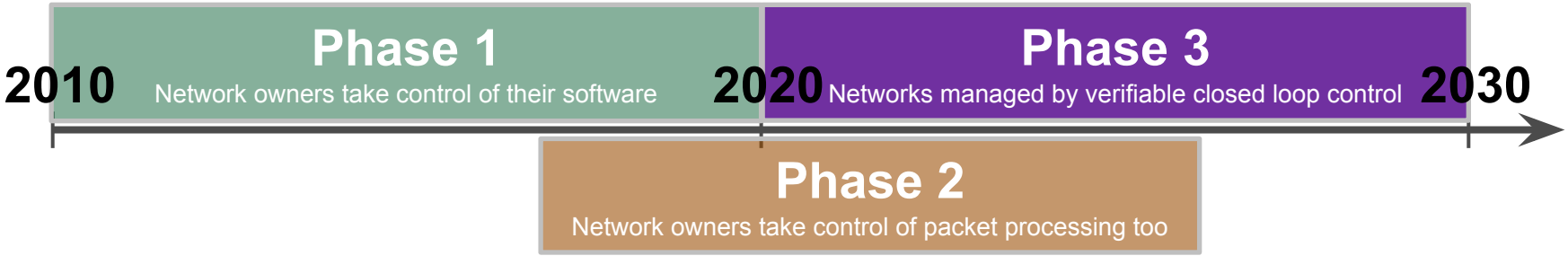
Next-Gen SDN

Moderator: Timon Sloane, ONF

Nick's Phases of SDN
from Day 1 of ONF Connect 2019



Journey to Next-Gen SDN

Bold SDN vision established
- Separation of Ctrl & Dataplane
- Centralized Control
- OpenFlow

Enabling
- Simpler forwarding devices
- Faster innovation
- Reduced capex/opex

Google
& others
SDN implementations

New Technologies
Programmable Silicon
P4, P4Runtime
Cloud Dev Models

Lessons Learned

Leap forward in capabilities
- Top-Down Programmability
- Hardware Independence
- Cloud-like lifecycle
- Verifiability

Next-Gen SDN

Enabling
- Top-Down Operator Control
- Rapid Network Innovation
- Zero Touch Operation
- Robust Hardware Ecosystem

SDN

Open Source SDN
Enters Production

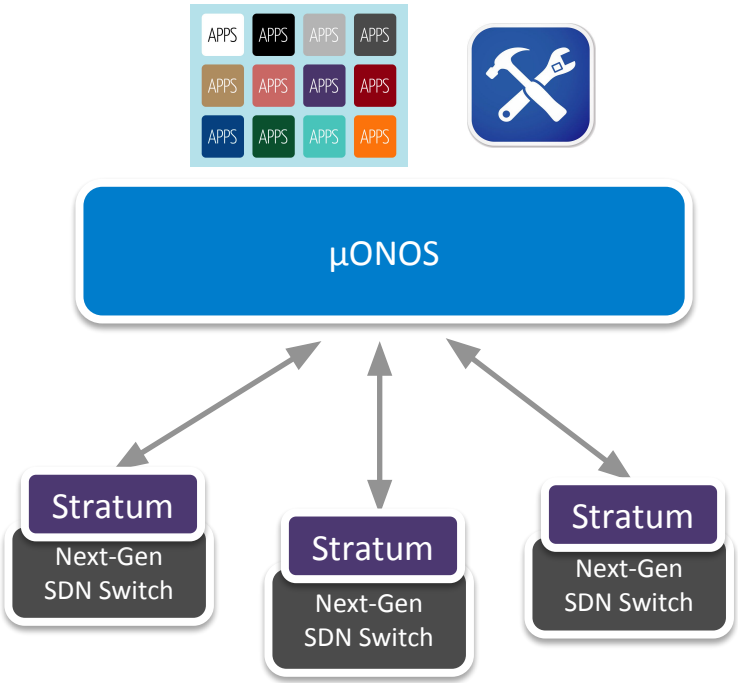
SDN--

Vendors responded with software configuration tools:
- Traditional switching & routing using conventional embedded control protocols

Value

Time

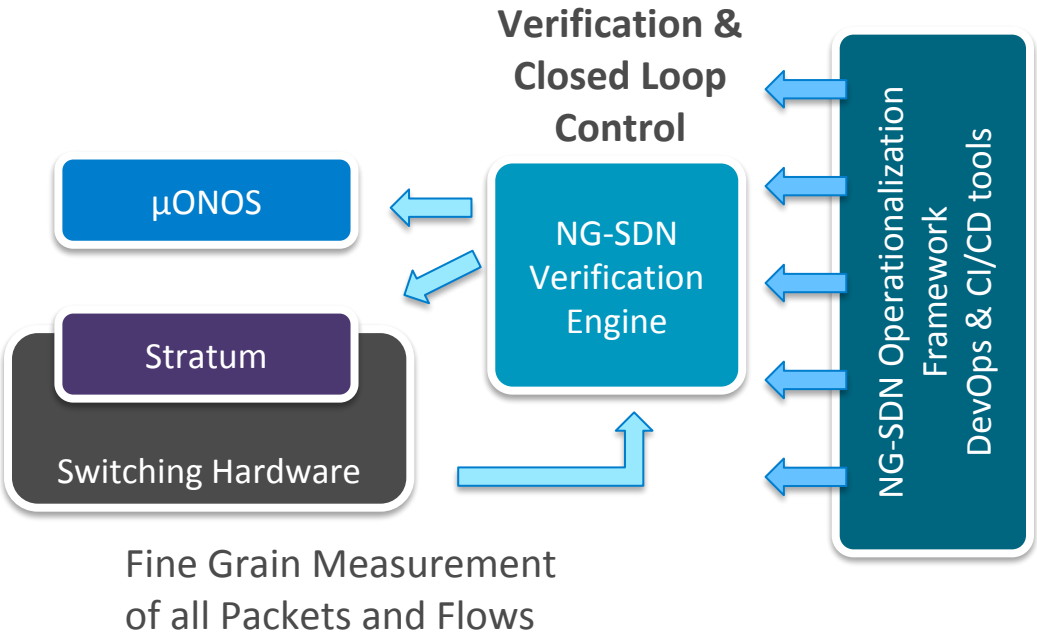
Next-Gen Software Stack Components



- µONOS
  - Supports Next-Gen SDN interfaces (P4Runtime, gNMI, gNOI, gRIBI)
  - New config and management subsystem (OpenConfig)
  - Cloud-native: microservices, Kubernetes, gRPC, etc.
  - Enable apps to take advantage of the new capabilities
- Stratum
  - Thin switch OS
  - Supports Next-Gen SDN interfaces (P4Runtime, gNMI, gNOI)
  - Supports OpenConfig models
- Forwarding devices
  - Supports programmable forwarding (P4)
  - Also supported fixed function and partially programmable devices
    - Enables smooth migration and diversity of silicon options

NG-SDN Stack

Builds on SDN – Adding Verification & Cloud-like Operationalization



Enables Cloud-like Feature Velocity

Features can be rapidly rolled out (and rolled back)

Functionality can be verified

Networks become tamper-proof

Overview: Next-Gen SDN Stack

Nate Foster
Associate Professor
of Computer Science,
Cornell University

Timon Sloane
VP, Marketing &
Ecosystem, ONF

Brian O'Connor
MTS, ONF

Thomas Vachuska
Chief ONOS Architect, ONF

STRATUM

The Data Plane for NG-SDN

Brian O'Connor
brian@opennetworking.org

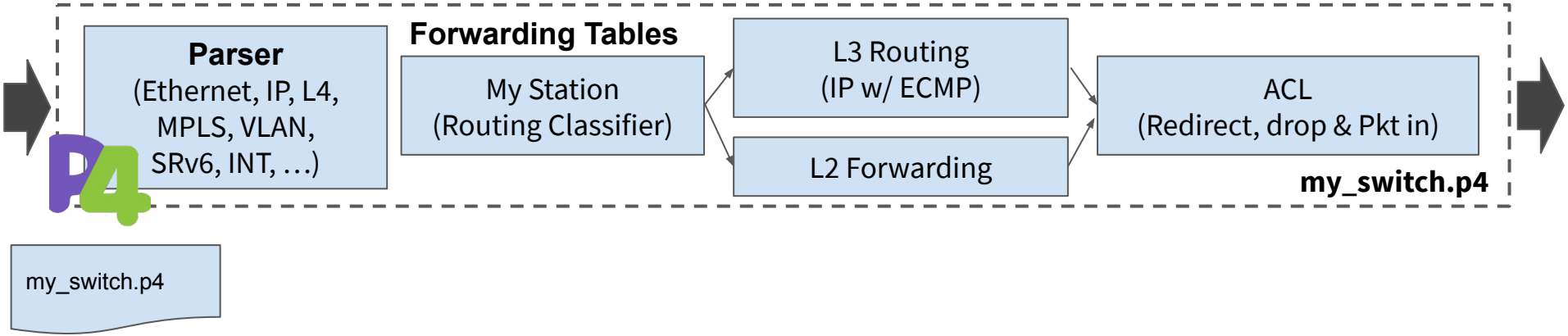
ONF Connect
9/12/2019

Why Stratum for NG-SDN?

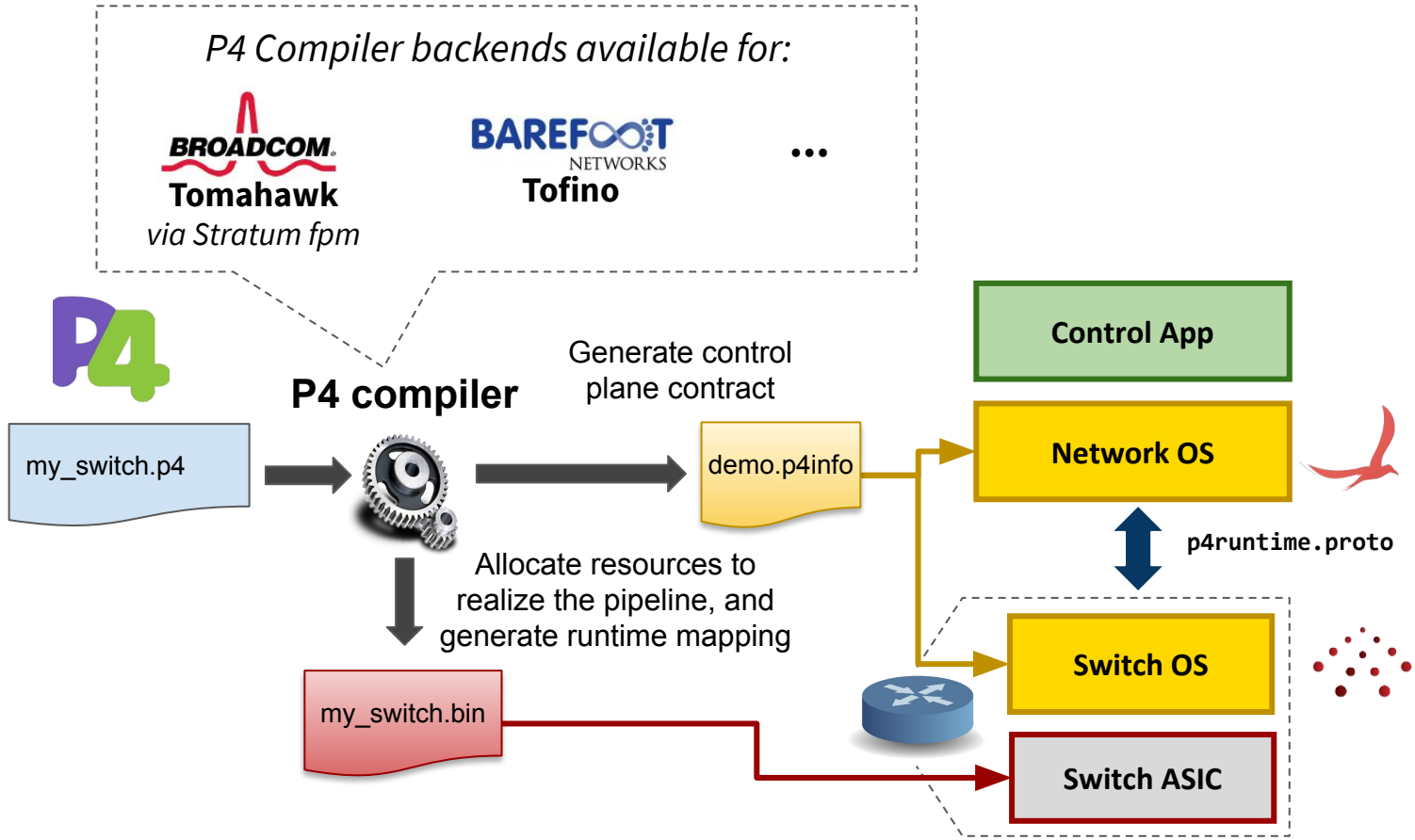
1. **Stratum Interfaces, Models, and Pipeline Definition**
   - ***Built from today's SDNs & Optimized for NG-SDN***
   - **Enables Top Down Programming**
     Functionality defined in formal, programmatic languages (Protobuf, YANG, P4)
   - **Enables Hardware Independence / Vendor Optionality**
     Same interfaces, same models, same programs, different targets
2. **Stratum Packaging and Operational Interfaces**
   - **Enables Automated Full-Lifecycle Operation**
     Telemetry provides necessary feedback for automated upgrade/rollback
     Machine-optimized operational interfaces (vs. SSH and `expect`)
     Seamless deployment with Docker

Defining the Data Plane

All network behavior, down to the packet header definitions and forwarding rules, is unambiguously specified in software via programmable interfaces.



Defining the Data Plane



P4Runtime

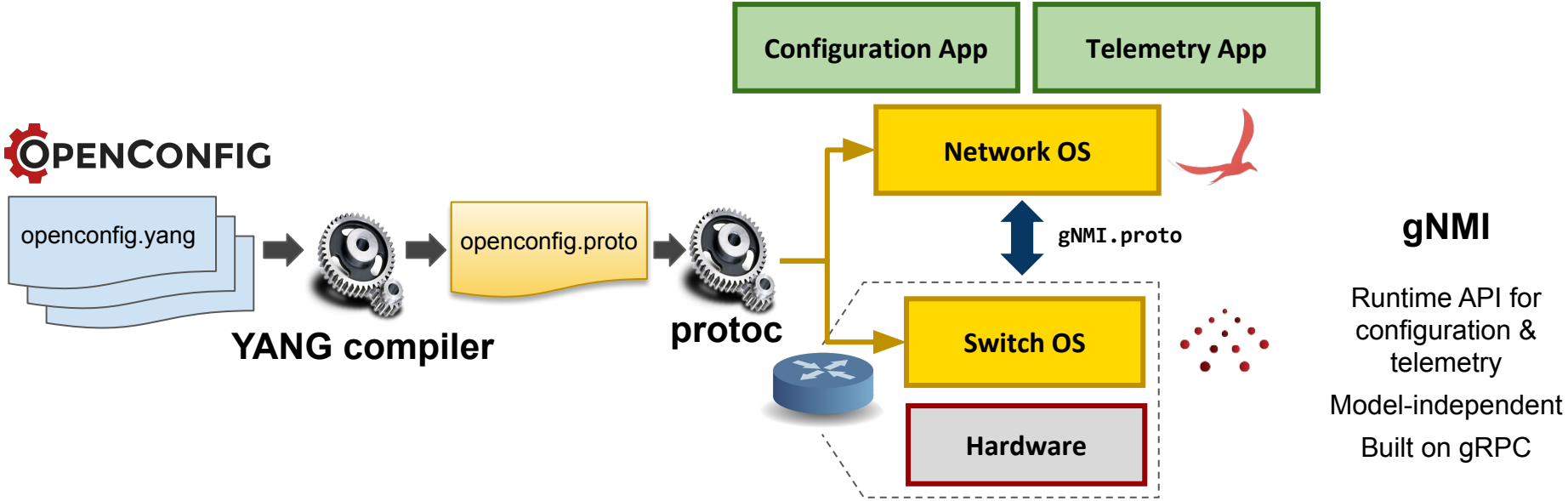
Runtime API for pipeline control

Program-independent

Built on gRPC

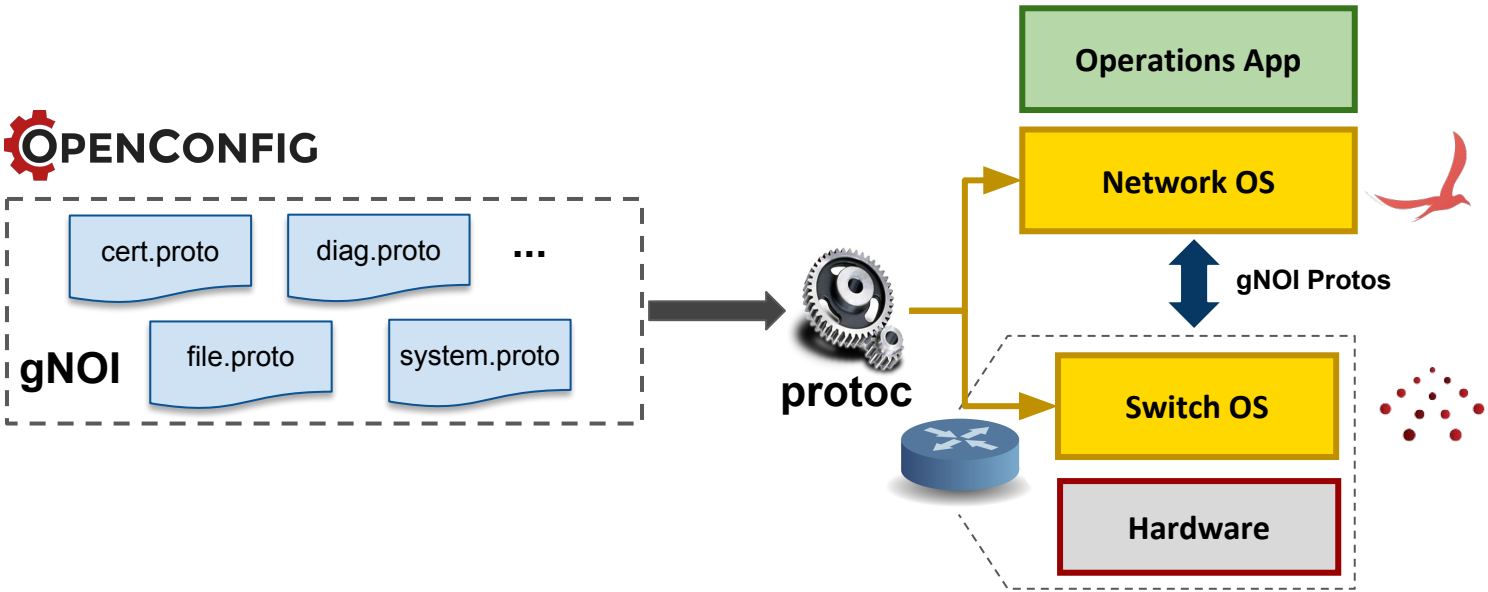
Defining Configuration & Telemetry

- All network configuration and telemetry state information is modeled using programmatic definitions
- Use of the same set of industry-driven models across all Stratum switches ensures vendor interoperability



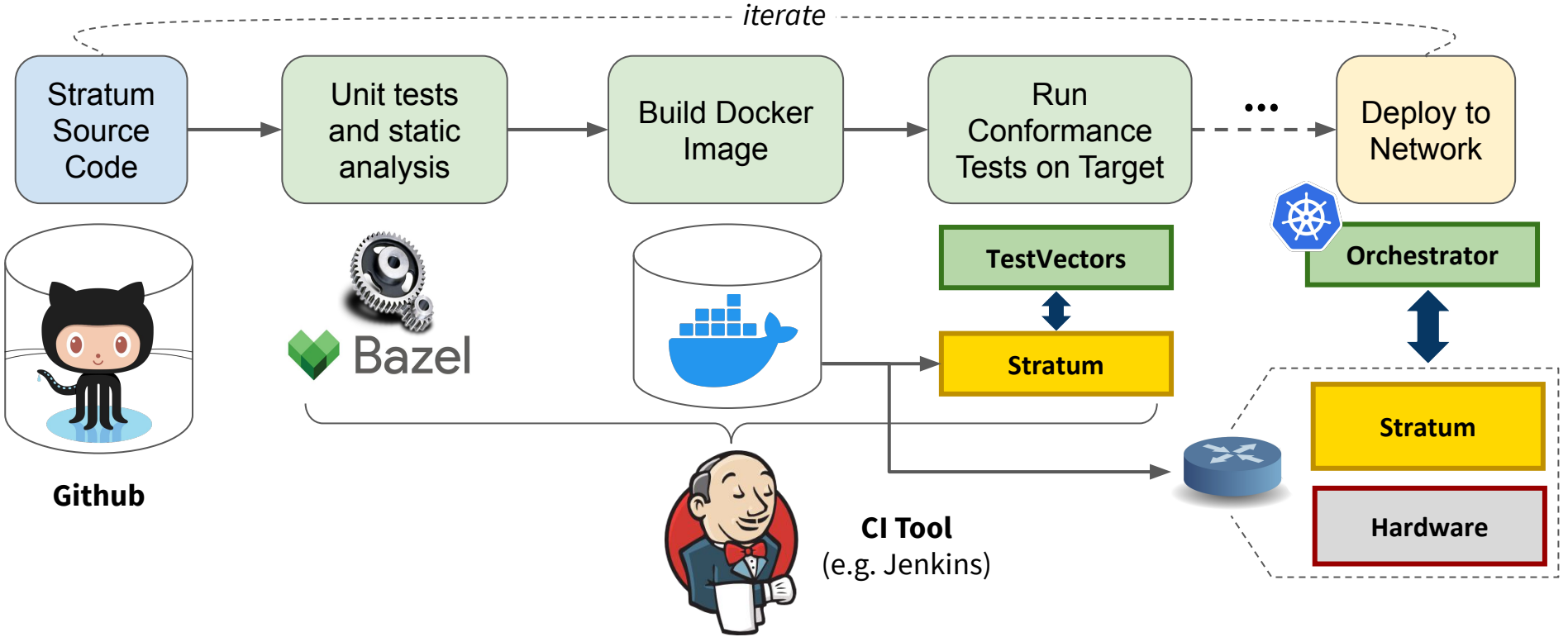
Defining Operations

- Network operations services and RPCs are modeled using programmatic definitions
- Clear definitions and semantics are perfect for machine-driven operations and enable automation



Supporting Cloud-Style Agility

- Stratum deployment and testing framework enable best practices in CI/CD
- Telemetry and end-to-end verification allow for failure detection and rollback



What to Expect in Open Source Stratum Today

Open Source Launch is an Alpha Release

- Software Architecture is in place
- Support for fixed-function and programmable targets
- Some features not yet implemented

What can I do with Stratum today?

- Demos (ONF has done several over the past year)
- User Experimentation; Porting to New Platforms

Apache 2.0 license means Stratum can now be incorporated into vendors' products

Stratum Switch Support Today

Switch Vendor / Switching ASIC	DELL	DELTA	Edge-corE NETWORKS	Inventec	QCT	STORDIS
BAREFOOT NETWORKS **Tofino** *Up to 6.5 Tbps*		**AG9064v1** *64 x 100 Gbps*	**Wedge100BF-32X** *32 x 100 Gbps* **Wedge100BF-65X** *65 x 100 Gbps*	**D5054** *32 x 100 Gbps + 48 x 25 Gbps*		**BF6064X** *64 x 100 Gbps*
BROADCOM **Tomahawk** *Up to 3.2 Tbps*	**Z9100** *32 x 100 Gbps*		**AS7712** *32 x 100 Gbps*	**D7032** *32 x 100 Gbps*	**T7032-IX1** *32 x 100 Gbps*	

\+ 2 software switches: **bmv2** (functional software switch) & **dummy switch** (used for API testing)

Near-term future platforms:

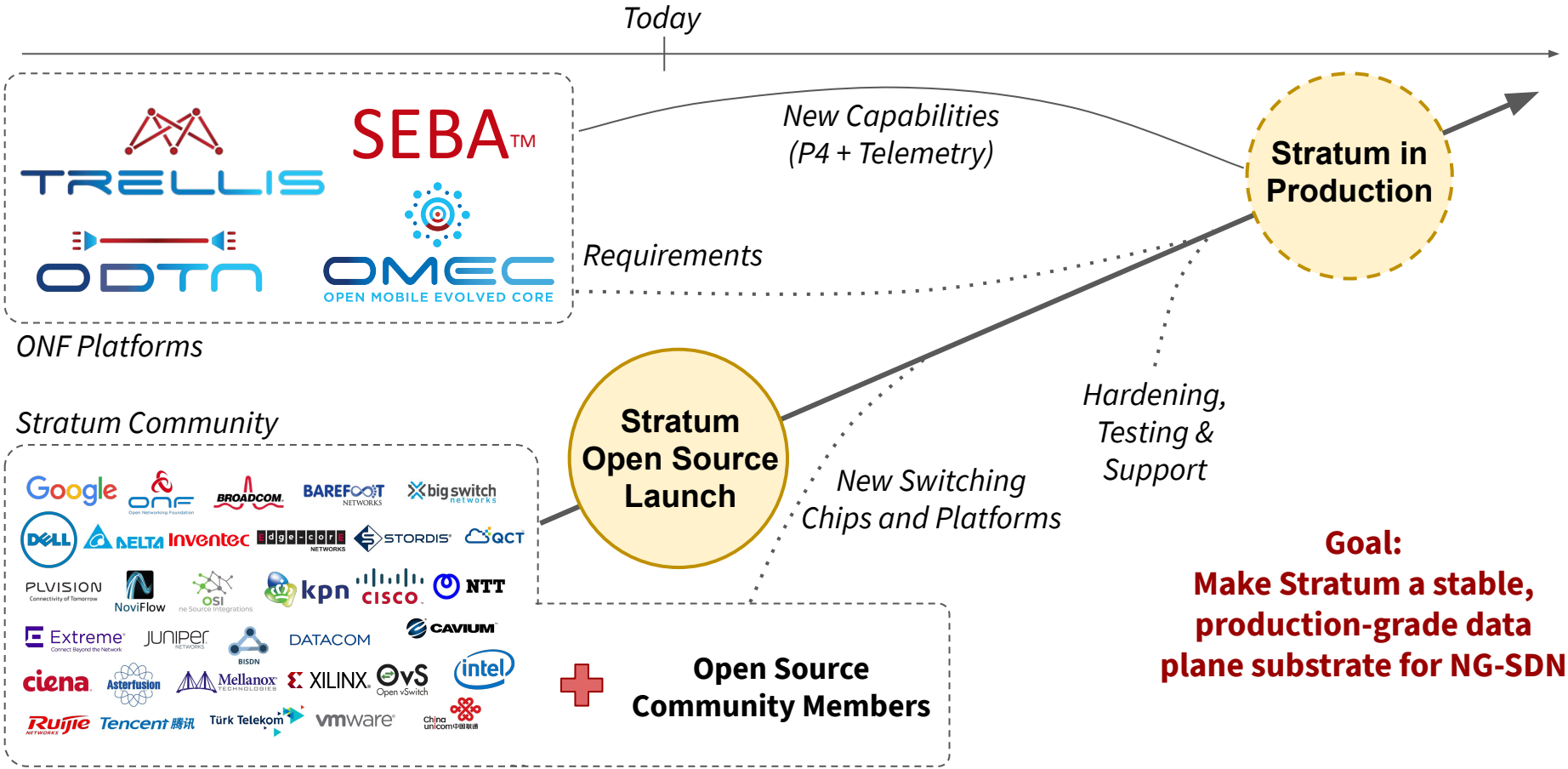
- Additional platforms for existing targets
  - Existing vendors + Asterfusion, ...
- Mellanox SN2700 (Spectrum)
- Datacom platforms (PowerPC-based)

Asterfusion

Mellanox TECHNOLOGIES

DATACOM

Stratum Project Roadmap



Demos @ ONF Connect ‘19

1. Stratum Interoperability
   - Community Launch Demo with 2 ASIC and 6 platform vendors
   - 8 Stratum switches in an IP routed, leaf-spine fabric
   - Highlighting **hardware independence** and **automatic full-lifecycle operation**
2. µONOS
   - 3 Stratum switches
   - Highlighting **zero-touch provisioning** and device configuration
3. SEBA
   - 1 Stratum switch
   - Highlighting **top-down data plane programmability** (offloading the BNG)

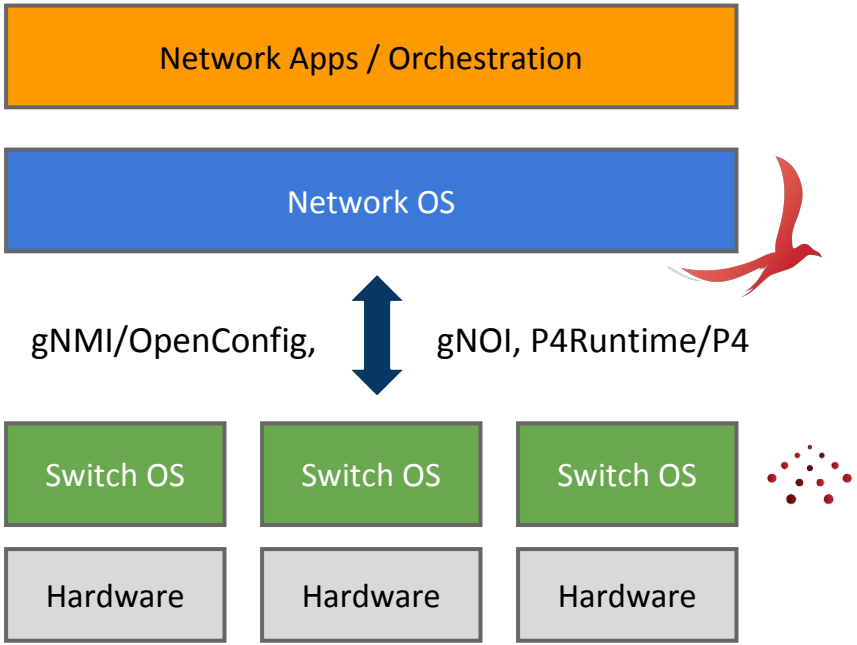
µONOS

Next-Gen SDN control plane

Thomas Vachuska

ONF Connect
2019-09-12

NG-SDN Objectives

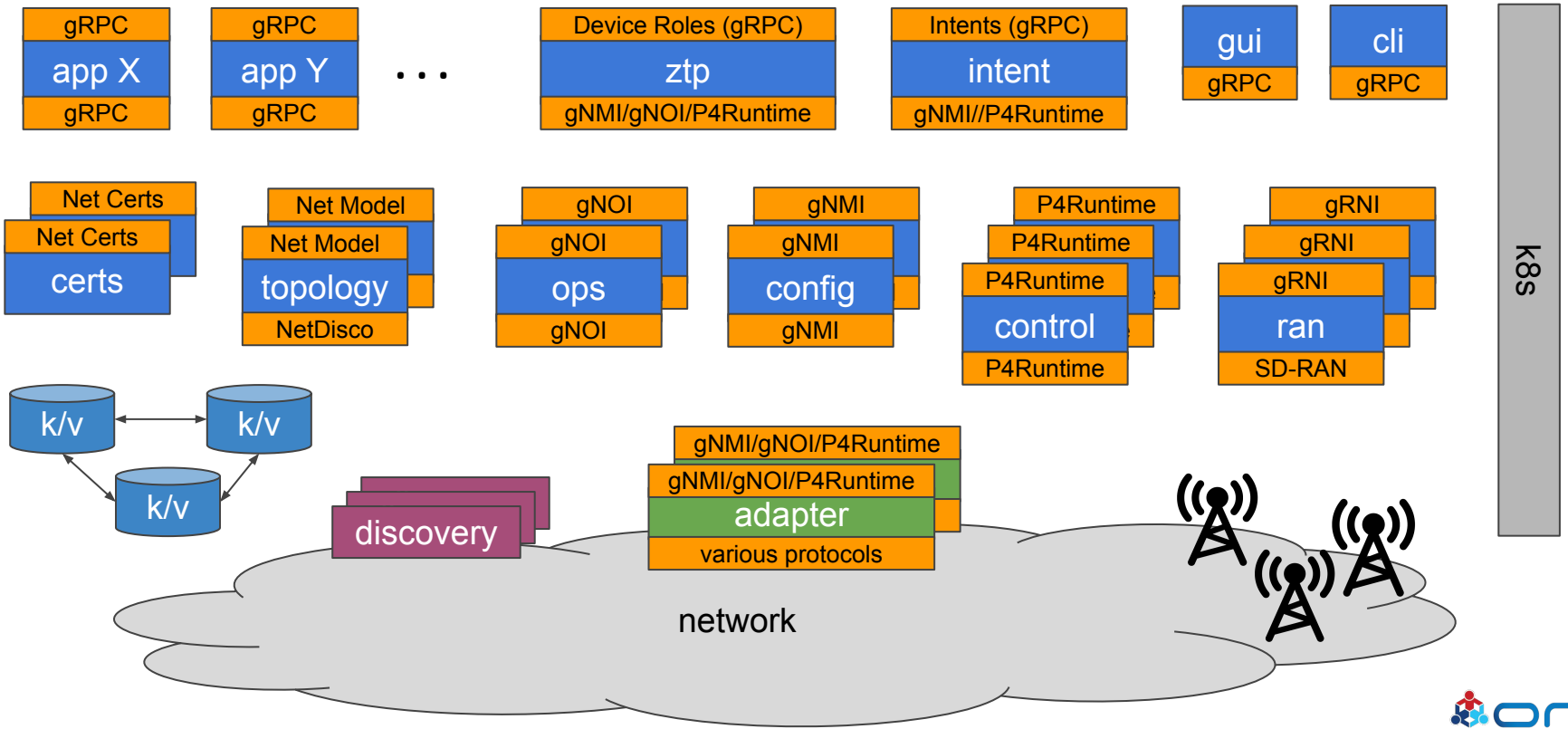


- SDN Apps and Solutions
- CI/CD Orchestration
- Zero-touch operation
- Network verifiability
- Top-down programmability
- Fine-grained control
- Fine-grained measurement
- Hardware independence

µONOS

- µONOS is the next generation architecture of ONOS
  - aims to provide smooth transition from current architecture
- Comprehensive platform for operations
  - configuration, control, monitoring, verification, live update, diagnostics
- Native support for next-gen SDN interfaces and models
  - e.g. gNMI, gNOI, P4Runtime, gRIBI, OpenConfig, etc.
- First-class support for 5G RAN edge
  - performance, scale and strict latency guarantees
- Cloud-native deployment - aimed at edge-cloud
  - built as set of µ-services, with gRPC interfaces, orchestrated by k8s

µONOS Deployment



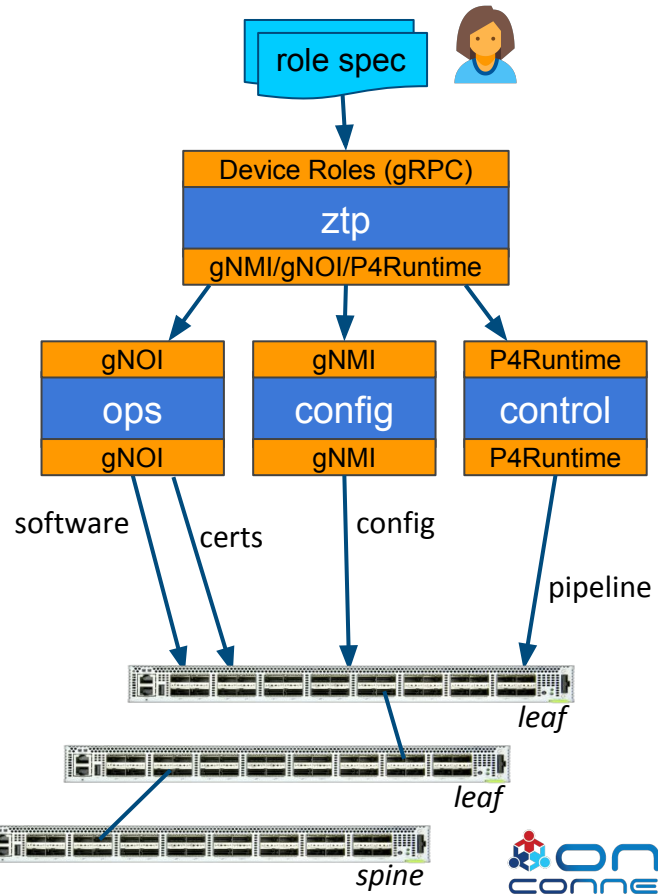
Common Network Operations

- Provisioning new devices and adding them to the network
- Adding new features to the existing data-plane
- Isolating faulty network devices and re-directing traffic
- *… and many others*

- All of the above require:
  - broad top-down control
  - intelligent and cooperative data-plane
- NG-SDN interfaces and software stack makes these possible

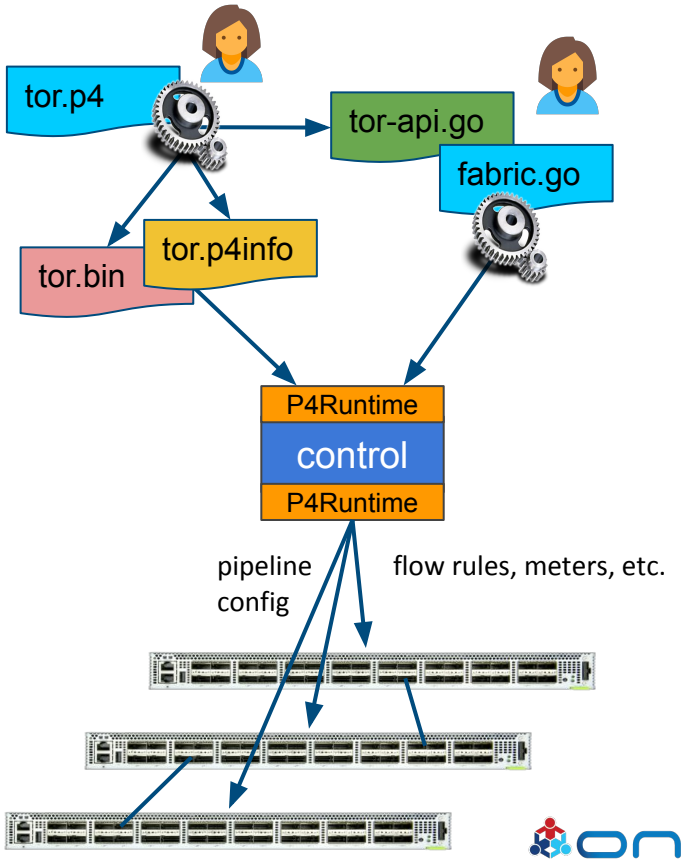
Example: Add New Device to Network

- Operator defines various device roles - a priori
  - sw version, chassis configuration, pipeline configuration, etc.
- ONOS given the address and role of a new device
- Based on the role and the type of the device, ONOS
  - downloads the desired version of the switch software via gNOI
  - may install or rotate certificates via gNOI
  - sets desired chassis configuration via gNMI
  - applies prescribed pipeline configuration via P4Runtime
  - enables ports via gNMI and marks device as available
- ONOS discovers or verifies new links
- Apps proceed to program the new device
  - using P4Runtime and pipeline-specific constructs



Example: Add New Data-Plane Feature

- Create or modify P4 program
  - to introduce new data-plane feature(s)
- Compile the P4 program using the P4 toolchain
  - as P4 info & binary for download to the switch
  - as pipeline-specific API for app development
  - assembled as an ONOS app for operational deployment
- Developer uses APIs to create control app(s)
  - APIs assist IDE creation of pipeline-specific apps
- ONOS downloads the P4 info and binary to switch
  - via P4Runtime
- Application programs the device pipeline
  - via P4Runtime and using pipeline-specific constructs



µONOS Status Update

- *onos-config*
  - implemented gNMI NB and SB APIs
  - multi-device configuration transactions
  - model driven, multi version support
  - rollback to previous points in time
  - device updates through subscription
  - flat storage of configuration data in K/V store
  - configuration validation against YANG models
- *onos-gui*
  - interactive configuration views
  - uses same framework as GUI2 in ONOS 2.2
- *onos-topo*
  - device inventory and topology APIs
  - API design still work-in-progress
- *onos-ztp*
  - basic role-based configuration via gNMI to *onos-config*
  - basic table pipeline setup of devices via REST API to ONOS 2.2
- *atomix-go*
  - Go APIs for distributed primitives
- *onit*
  - integration tests suite and deployment

NG-SDN Control Plane

- ONOS has been and continues to support NG-SDN capabilities
  - enabling broad top-down data-plane programming, monitoring
- µONOS is the next step in the evolution of the platform
  - adding more breadth to top-down programmability via config and ops support
  - adding zero-touch provisioning
  - allows orchestration via Cloud-native technologies
  - advances the high standard for high-availability, performance and scale
- µONOS carries forward the NG-SDN vision

Engage!

Thank You

Verifiable Networks

Nate Foster
Cornell University

A Bit of History…

The Design Philosophy of the DARPA Internet Protocols

David D. Clark*
Massachusetts Institute of Technology
Laboratory for Computer Science
Cambridge, MA. 02139

(Originally published in Proc. SIGCOMM '88, Computer Communication Review Vol. 18, No. 4, August 1988, pp. 106–114)

Abstract

The Internet protocol suite, TCP/IP, was first proposed fifteen years ago. It was developed by the Defense Advanced Research Projects Agency (DARPA), and has been used widely in military and commercial systems. While there have been papers and specifications that describe how the protocols work, it is sometimes difficult to deduce from these why the protocol is as it is. For example, the Internet protocol is based on a connectionless or datagram mode of service. The motivation for this has been greatly misunderstood. This paper attempts to capture some of the early reasoning which shaped the Internet protocols.

1. Introduction

For the last 15 years[1], the Advanced Research Projects Agency of the U.S. Department of Defense has been developing a suite of protocols for packet switched networking. These protocols, which include the Internet Protocol (IP), and the Transmission Control Protocol (TCP), are now U.S. Department of Defense standards

architecture into the IP and TCP layers. This seems basic to the design, but was also not a part of the original proposal. These changes in the Internet design arose through the repeated pattern of implementation and testing that occurred before the standards were set.

The Internet architecture is still evolving. Sometimes a new extension challenges one of the design principles, but in any case an understanding of the history of the design provides a necessary context for current design extensions. The connectionless configuration of ISO protocols has also been colored by the history of the Internet suite, so an understanding of the Internet design philosophy may be helpful to those working with ISO.

This paper catalogs one view of the original objectives of the Internet architecture, and discusses the relation between these goals and the important features of the protocols.

2. Fundamental Goal

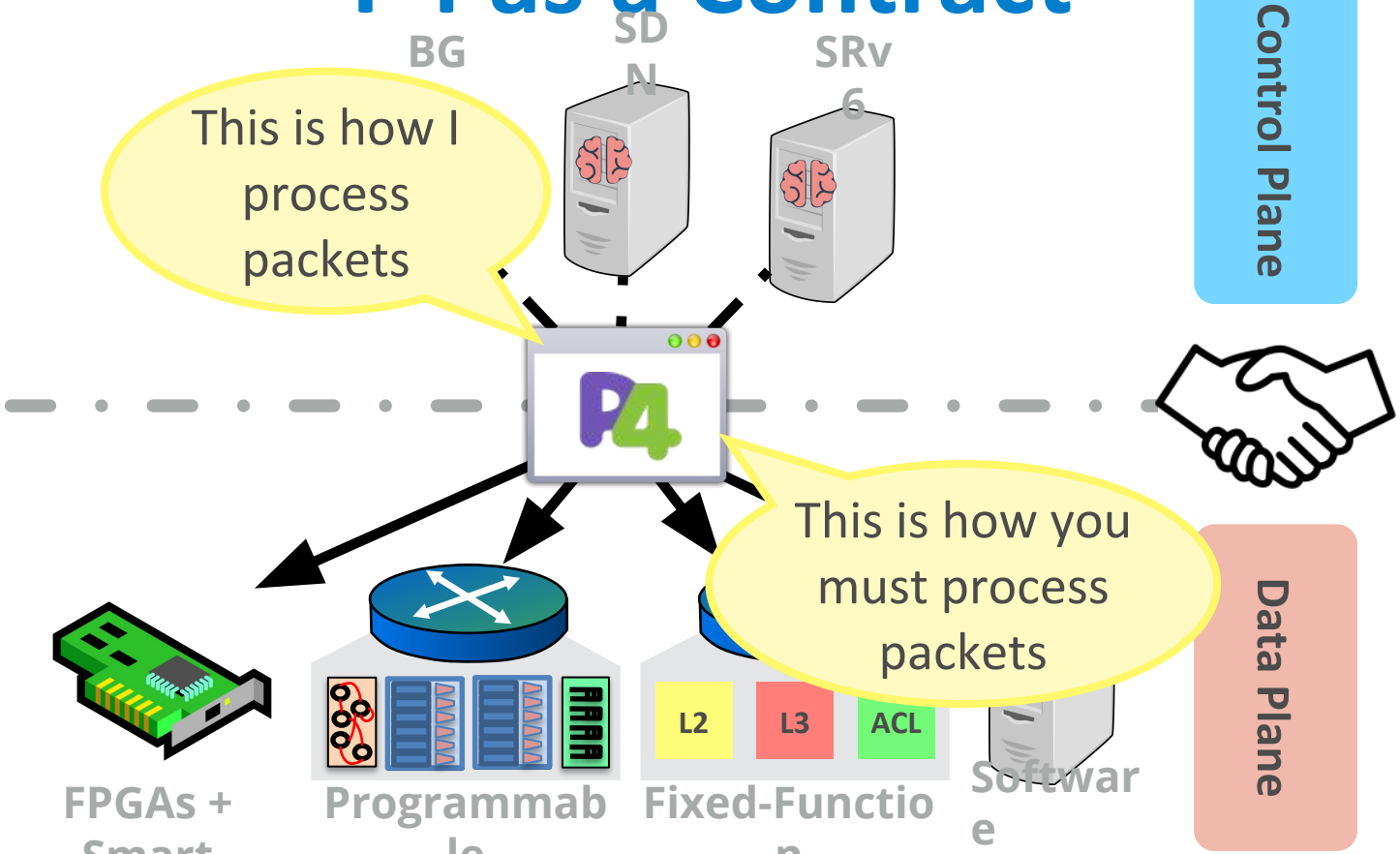
The top level goal for the DARPA Internet Architecture

"While tools to verify logical correctness are useful, both at the specification and implementation stage, they do not help with the severe problems that often arise related to performance."

What's Different?

- Increased scale and complexity
  - Modern networks *much* larger than 1980s networks
  - Operators demanding richer behaviors
- New networking architectures
  - Shift toward SDN and disaggregated control
  - Precise models of key components (e.g., P4)
- Formal methods advances
  - Techniques for modeling complex systems
  - Scalable automated solvers (e.g., SAT/SMT)
- Property enforcement using programmability
  - Programmable telemetry and run-time verification
  - (Eventually) top-down design and closed-loop control

P4 as a Contract



Three Methods to Gain Assurance

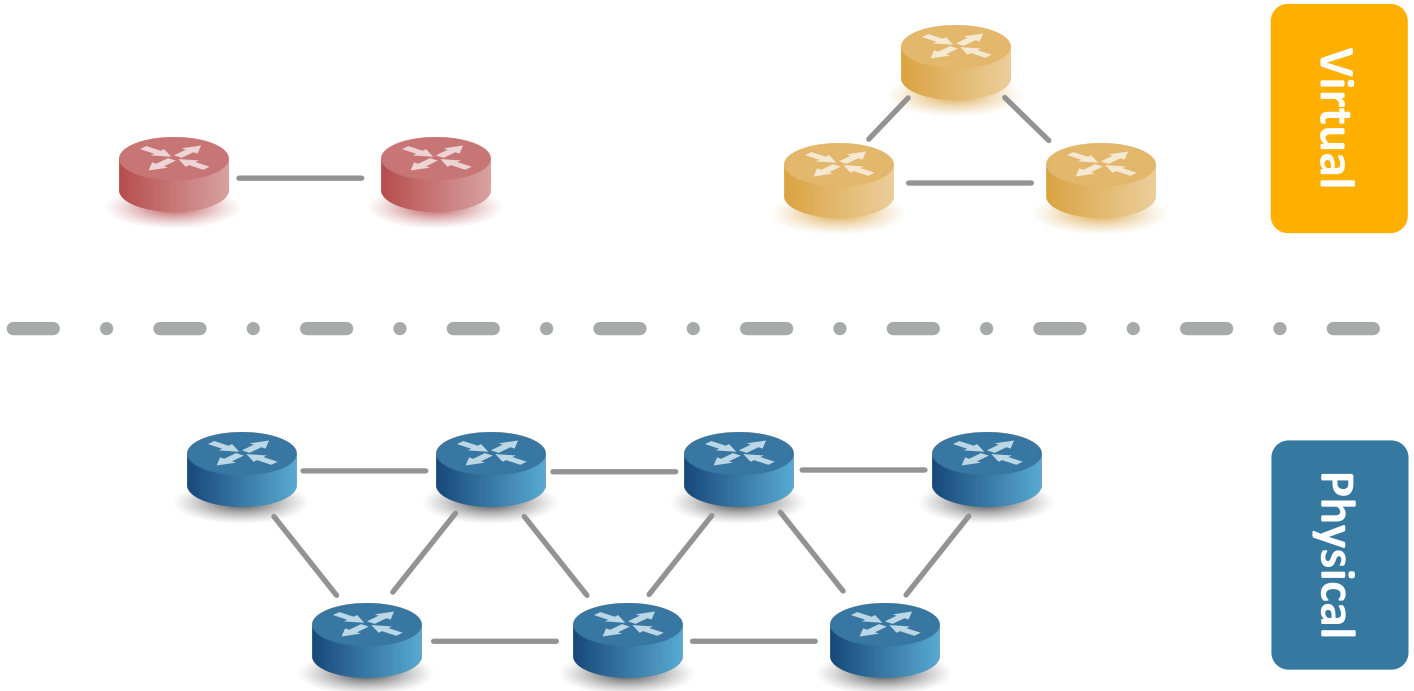
By Construction

By Analysis

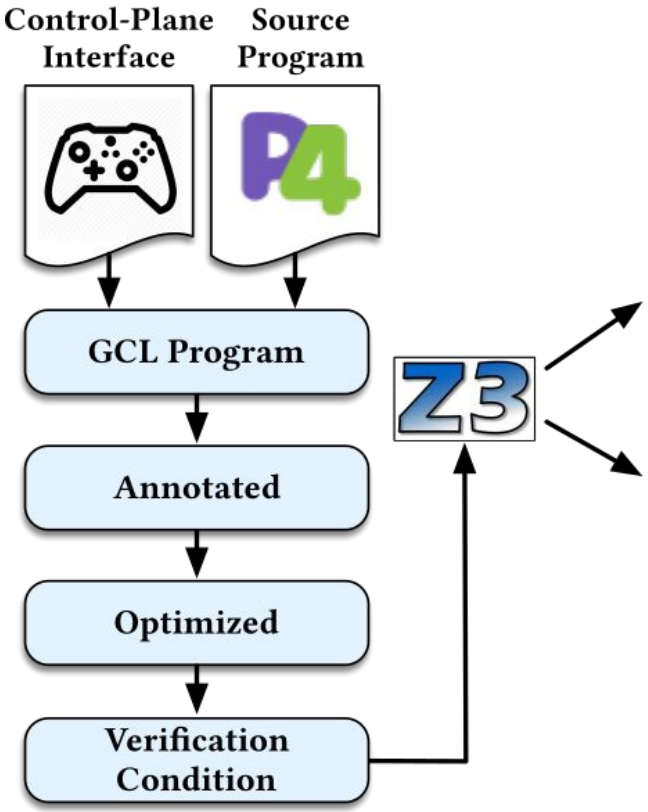
By Synthesis

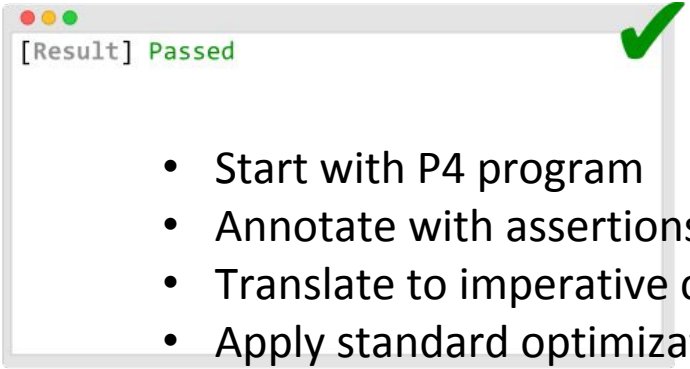
Logical Attestation [SOSP ‘11]

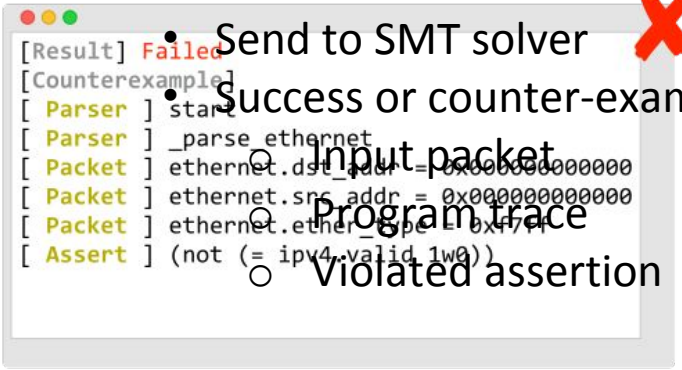
Construction: Virtualization



Analysis: Static Verification

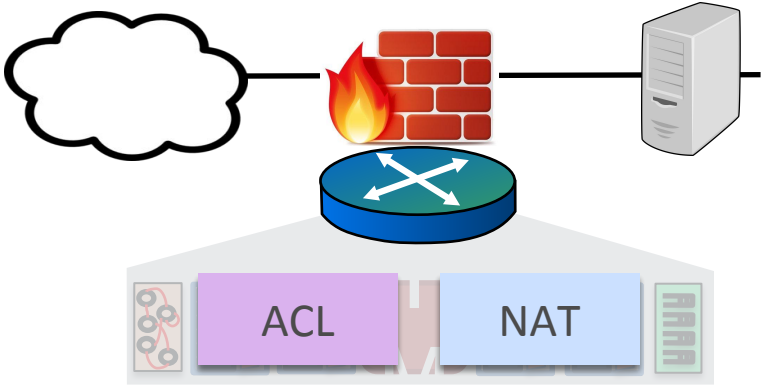


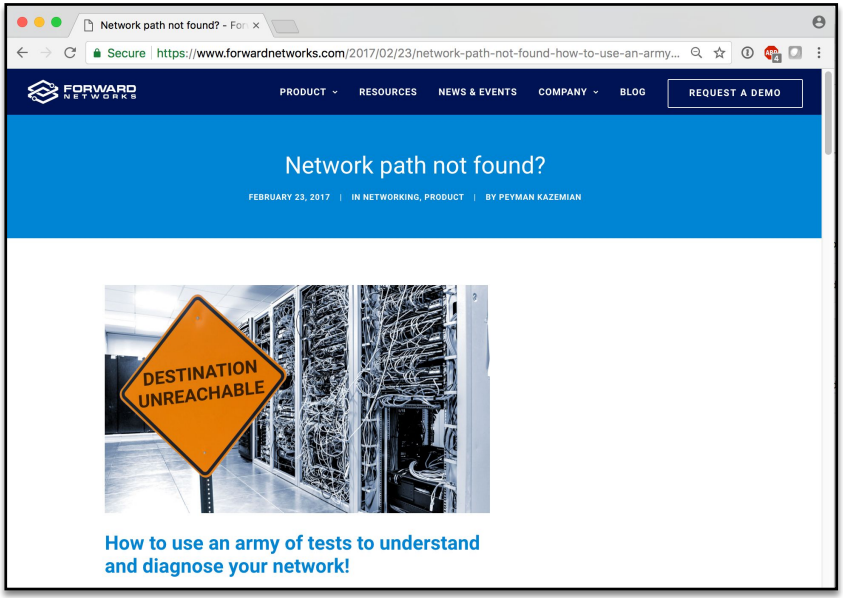




- Start with P4 program
- Annotate with assertions
- Translate to imperative commands
- Apply standard optimizations
- Gennerate first-order formula
- Send to SMT solver
- Success or counter-example
  - Input packet
  - Program trace
  - Violated assertion

Example: NAT + ACL





demo.p4

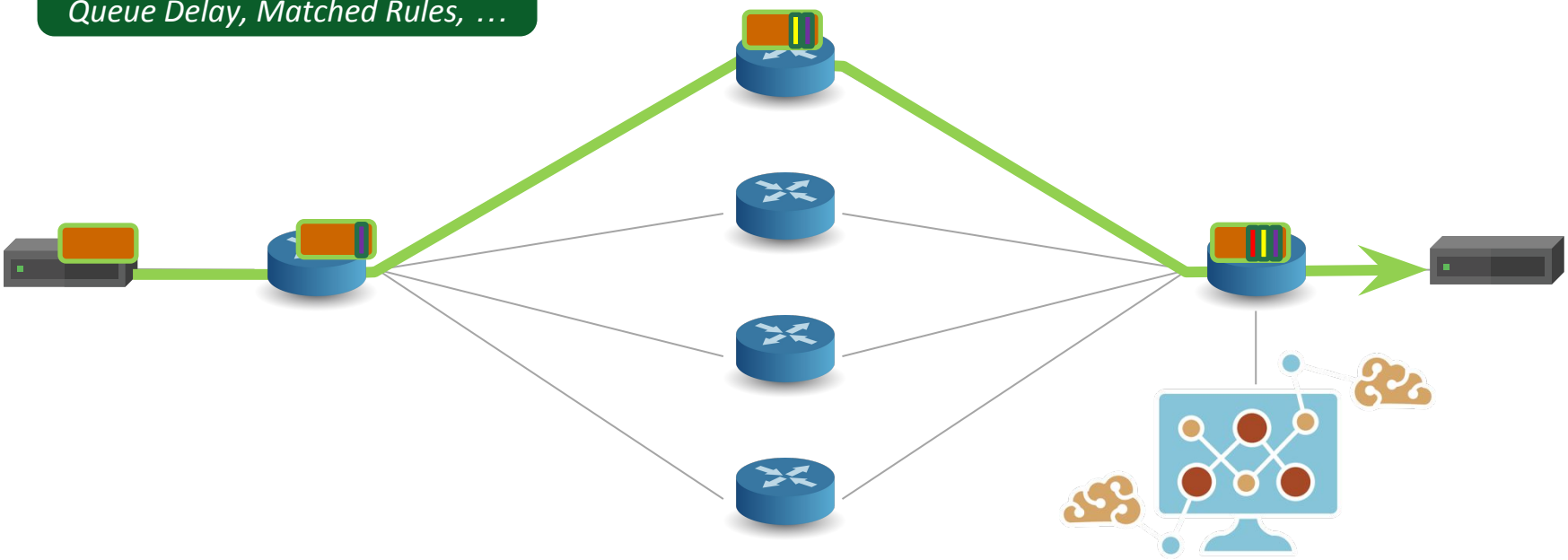
```
/* Headers and Instances */
header_type ethernet_t {
 fields {
  dst_addr:48;
  src_addr:48;
  ether_type:16;
 }
}
header_type ipv4_t {
 fields {
  pre_ttl:64;
  ttl:8;
  protocol:8;
  checksum:16;
  src_addr:32;
  dst_addr:32;
 }
}
header ethernet_t ethernet;
header ipv4_t ipv4;

/* Parsers */
parser start {
  extract(ethernet);
  return select(ethernet.ether_type) {
    0x800: parse_ipv4;
    default: ingress;
  }
}
```

-:--- demo.p4 Top L1 (P4_16 +2)

Quit

Synthesis: Run-Time Verification

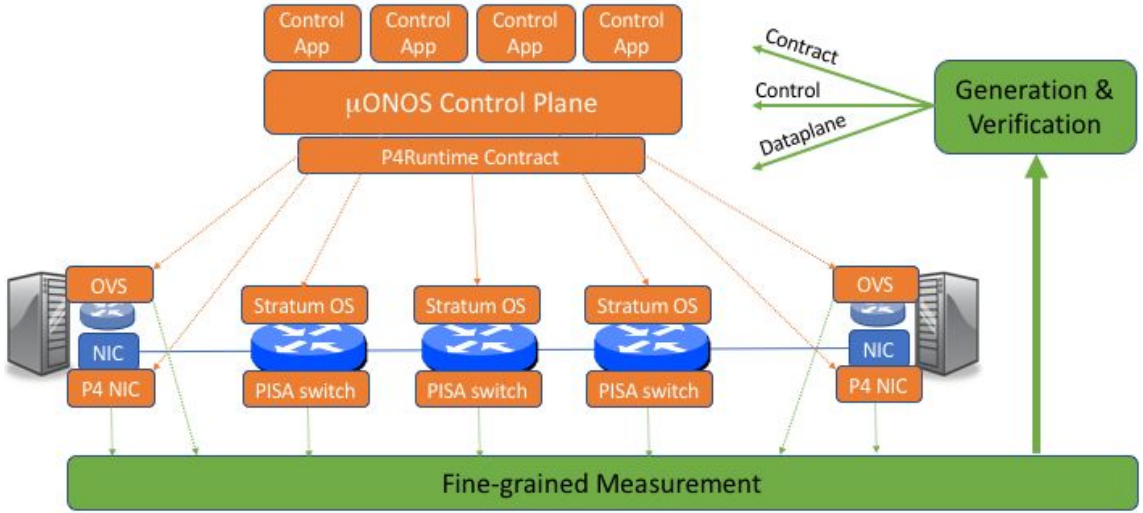
SwitchID, Arrival Time,
Queue Delay, Matched Rules, …



Log, Analyze
Replay and Visualize

SONATA [Sigcomm ‘18], Sketches [Sigcomm ‘12] …

Conclusion: Verifying Next-Gen SDN



Challenges:

- Cross-cutting mechanisms for enforcing application properties
- Quantitative reasoning about performance properties
- Scaling verification up to large networks

NG-SDN as Part of ONF's Big Picture

Timon Sloane

NG-SDN: Advancing the SDN Substrate for Networking

NG-SDN

SEBA - Broadband

ODTN - Optical

COMAC - Mobile

TRELLIS - NFV Fabric

Platforms

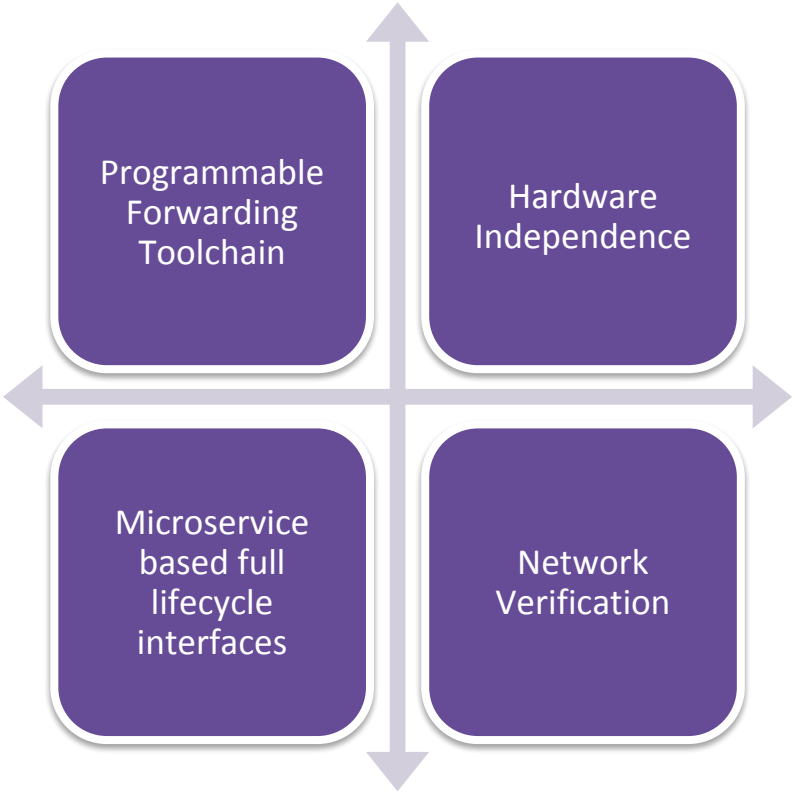
Substrate

µONOS

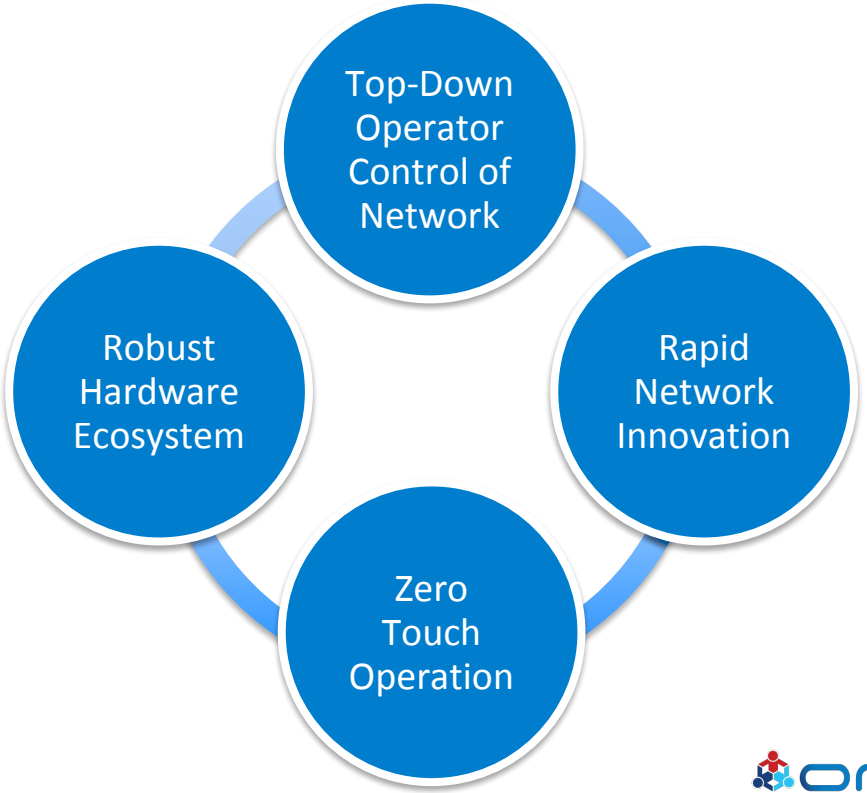
Verification Engine

Stratum

Revolutionary New Capabilities



Transformational Benefits



One More Thing I Learned This Week

Apparently, after we deliver all of this we'll all be drinking lots of piña coladas at the beach *

* For context, see Nick McKeown's ONF Connect 2019 Talk: https://vimeo.com/359434741

Overview: Next-Gen SDN Stack

Nate Foster
Associate Professor
of Computer Science,
Cornell University

Timon Sloane
VP, Marketing &
Ecosystem, ONF

Brian O'Connor
MTS, ONF

Thomas Vachuska
Chief ONOS Architect, ONF

Thank You

Follow Up Links:

www.opennetworking.org/ng-sdn/

Nick's talk on future of SDN: vimeo.com/359434741