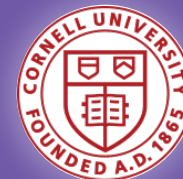




From Programmable Switches to Programmable Networks

Nate Foster
Cornell University





Please join the conversation!

Join the [#p4-2021-workshop](#) Slack channel on the ONF Community Slack workspace, to interact with workshop speakers and the broader community.

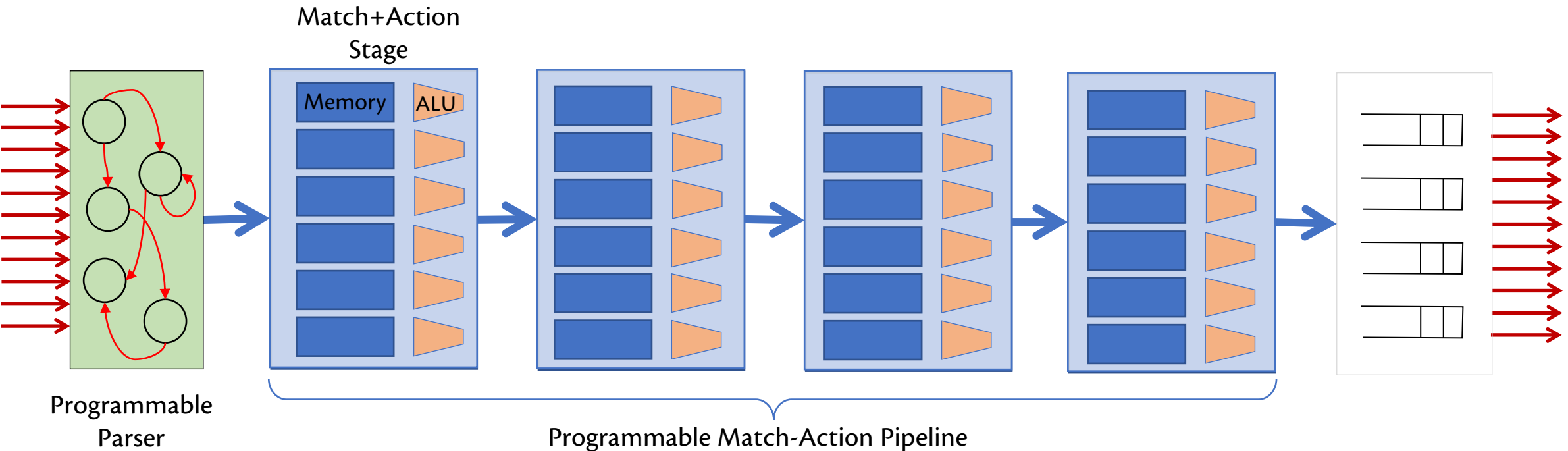
If you are not a member of the ONF Community Slack, you can sign up at:

<https://onf-community.slack.com/>

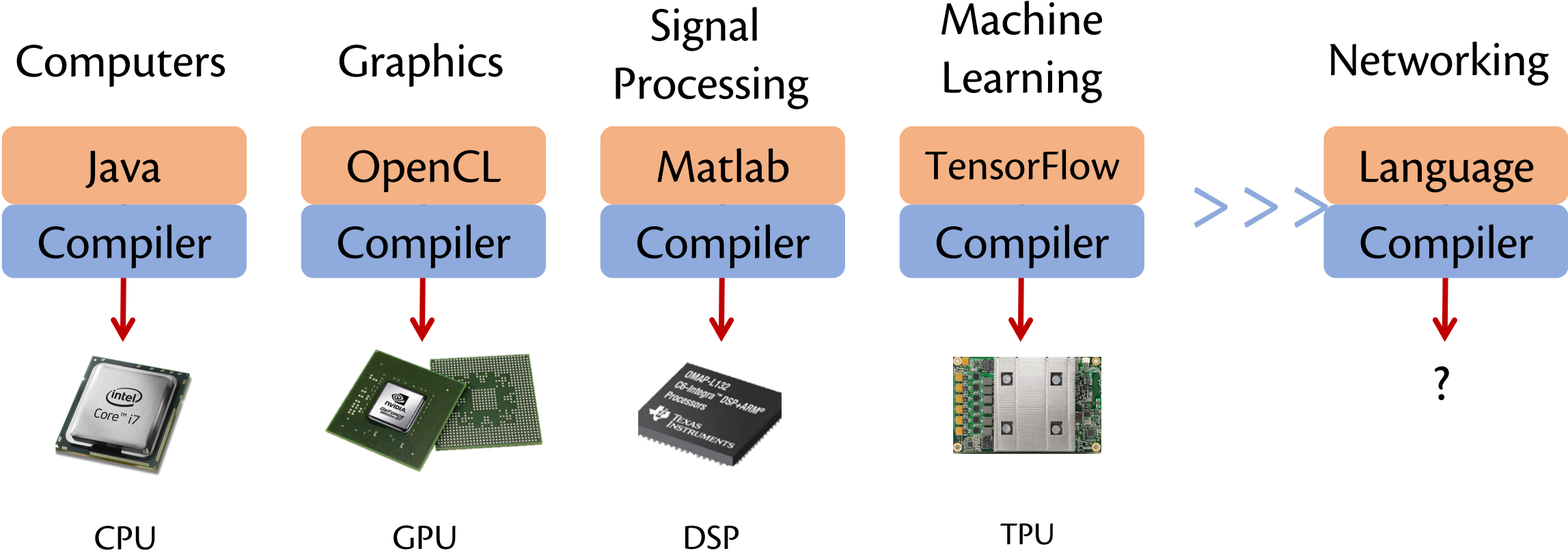
Vote for your favorite novel use of P4 on the 2021 P4 Workshop website!



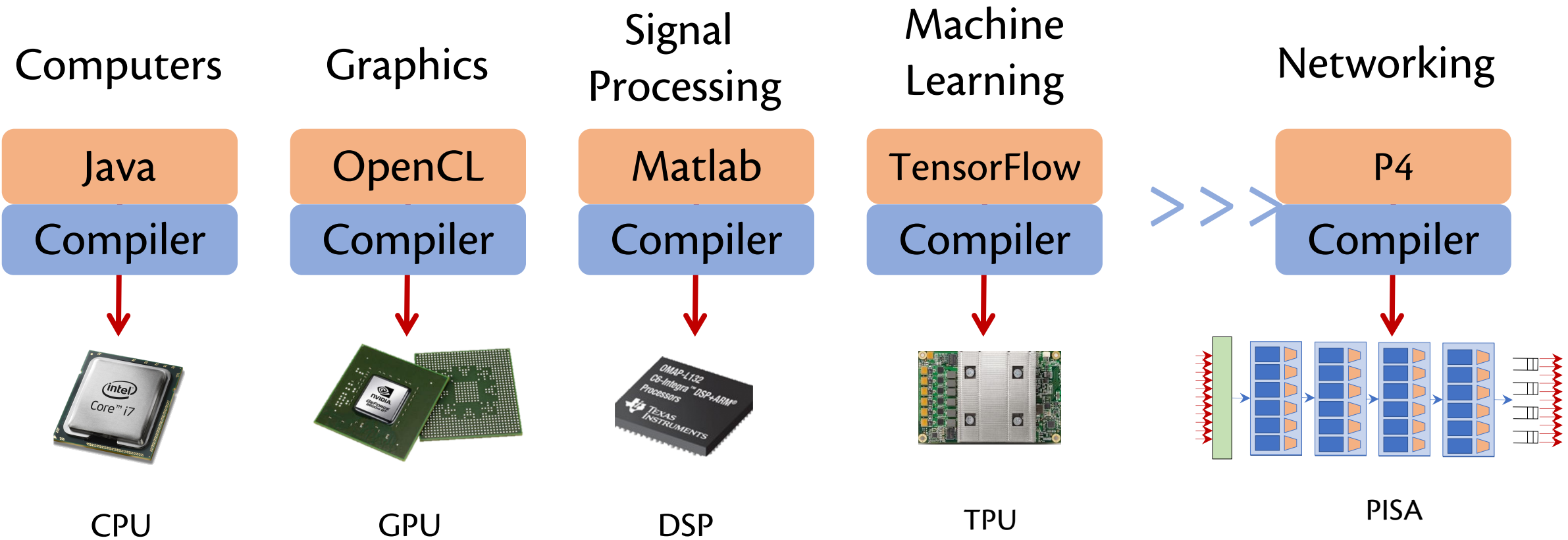
PISA: Protocol Independent Switch Architecture



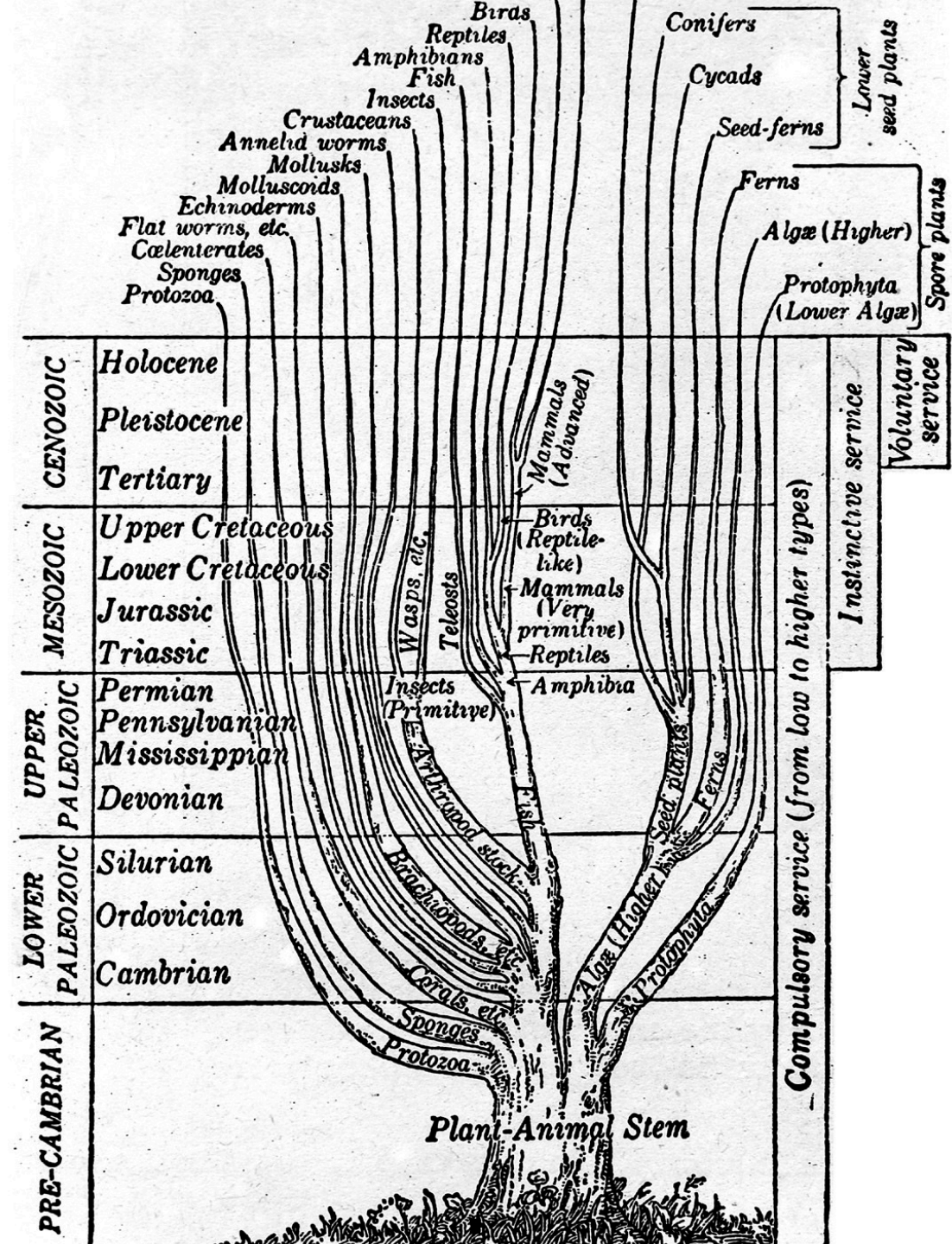
Domain Specific Processors



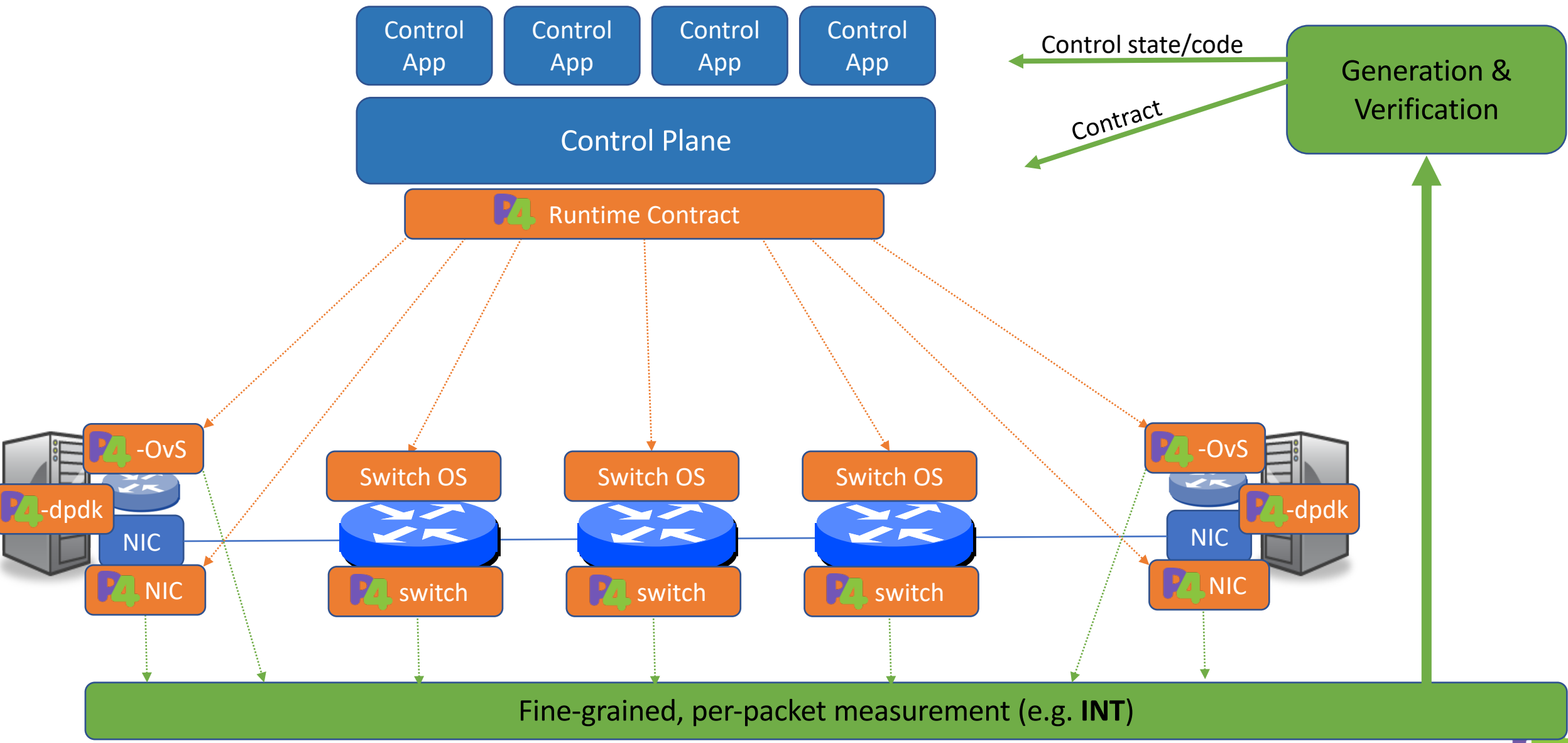
Domain Specific Processors



We are on the cusp of a Cambrian explosion!



Deep Programmability



State of P4 in 2021

New Features

- Continued evolution of P4₁₆ Language, P4Runtime, and P4 architectures (PSA, PNA, etc.)
- Open-source developers contributing to a growing set of software targets and tools

New Targets

- User-space (e.g., p4-dpdk)
- Kernel networking (e.g., P4-OvS)
- FPGAs and SmartNICs (multiple vendors)

New Applications

- Hardware offloads
- Congestion control
- Security



P4₁₆ Language Specification v1.2.2 (May 2021)

- Added new features to parsers, tables, expressions, statements, etc.
- Generalized P4's type system to make it more flexible and expressive
- Revised several aspects of the specification to clarify the intended meaning
- Changes are increasingly being guided by formal specifications of the P4 language

P4₁₆ Language Specification

version 1.2.2

The P4 Language Consortium

2021-05-17

Abstract

P4 is a language for programming the data plane of network devices. This document provides a precise definition of the P4₁₆ language, which is the 2016 revision of the P4 language (<http://p4.org>). The target audience for this document includes developers who want to write compilers, simulators, IDEs, and debuggers for P4 programs. This document may also be of interest to P4 programmers who are interested in understanding the syntax and semantics of the language at a deeper level.

Contents

1. Scope	5
2. Terms, definitions, and symbols	6
3. Overview	6
3.1. Benefits of P4	9
3.2. P4 language evolution: comparison to previous versions (P4 v1.0/v1.1)	9
4. Architecture Model	10
4.1. Standard architectures	12
4.2. Data plane interfaces	12
4.3. Extern objects and functions	12
5. Example: A very simple switch	13
5.1. Very Simple Switch Architecture	14
5.2. Very Simple Switch Architecture Description	16

P4Runtime v1.3.0 (December 2020)

- Obtained an IANA assigned TCP port 9559
- Added detailed treatment of security considerations for P4Runtime servers
- Adopted inclusive terminology (e.g., primary/backup)
- Clarified role of annotations in P4Info files

P4Runtime Specification

version 1.3.0

The P4.org API Working Group

2020-12-01

Abstract

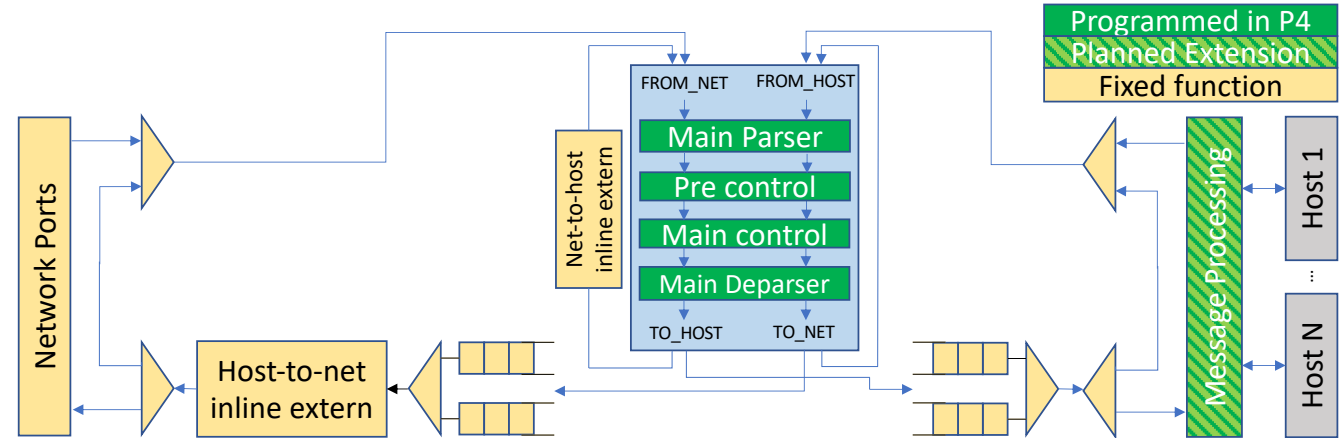
P4 is a language for programming the data plane of network devices. The P4Runtime API is a control plane specification for controlling the data plane elements of a device defined or described by a P4 program. This document provides a precise definition of the P4Runtime API. The target audience for this document includes developers who want to write controller applications for P4 devices or switches.

Contents

1. Introduction and Scope	5
1.1. P4 Language Version Applicability	5
1.2. In Scope	5
1.3. Not In Scope	5
2. Terms and Definitions	6
3. Reference Architecture	7
3.1. P4Runtime Service Implementation	8
3.1.1. Security concerns	9
3.2. Idealized Workflow	9
3.3. P4 as a Behavioral Description Language	9
3.4. Alternative Workflows	10

Portable NIC Architecture v0.5 (May 2021)

- An initial design intended to provide enough features to implement a virtual switch
- Main features:
 - Pipeline structure
 - Inline externs
 - Connection tracking
- Planned extensions to support message processing in the future
- See Andy Fingerhut's on-demand talk for more!



P4 Portable NIC Architecture (PNA)

(working draft)

The P4 Language Consortium

2021-05-18

Abstract

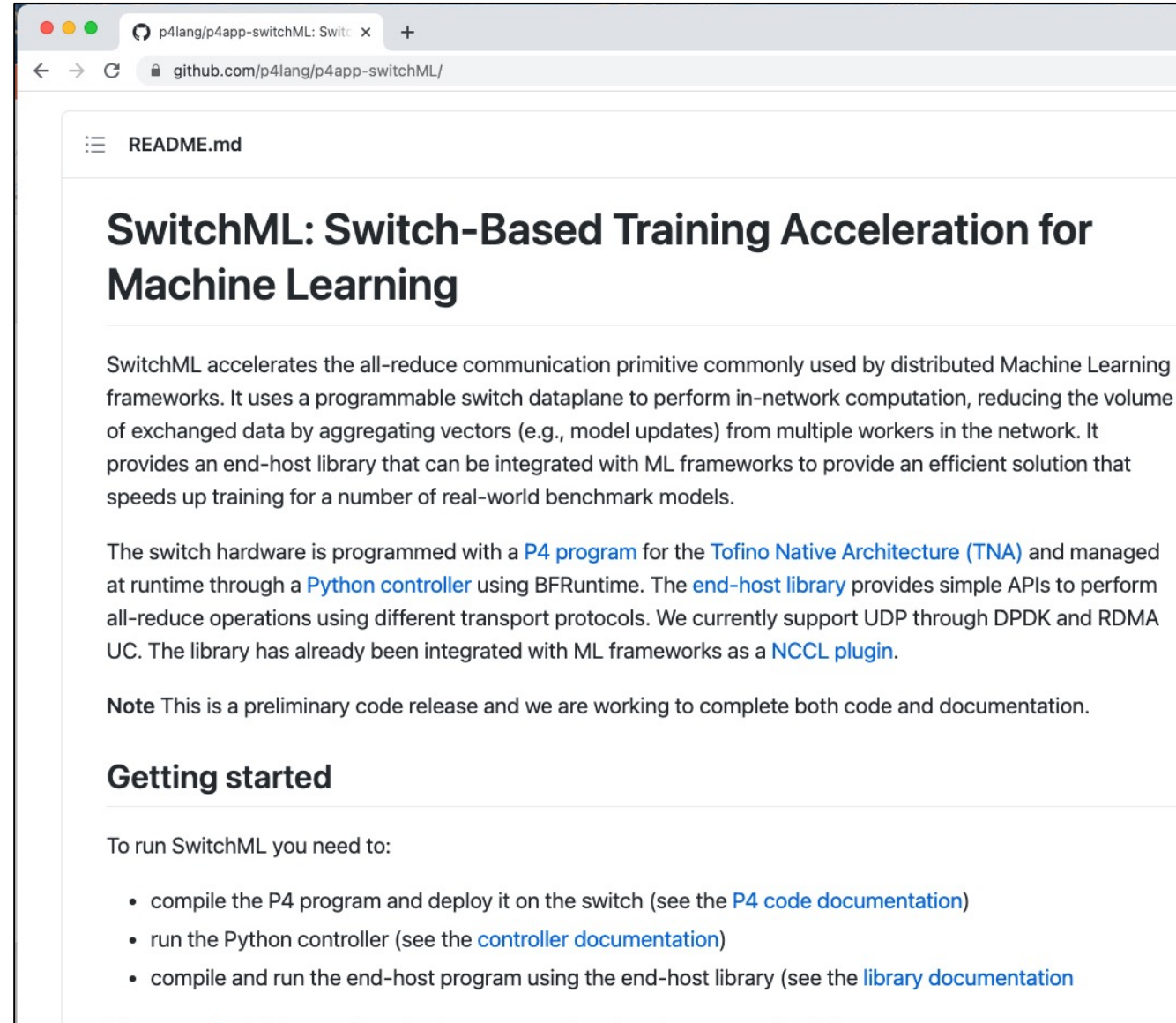
P4 is a domain-specific language for describing how packets are processed by a network data plane. A P4 program comprises an architecture, which describes the structure and capabilities of the pipeline, and a user program, which specifies the functionality of the programmable blocks within that pipeline. The Portable NIC Architecture (PNA) is an architecture that describes the structure and common capabilities of network interface controller (NIC) devices that process packets going between one or more interfaces and a host system.

P4 Apps Repository (April 2021)

A P4 code repository on GitHub

Two categories of contributions:

- P4 Libraries
 - Goal: community access
 - Example: reference implementations of standard protocols in P4
- P4 Systems
 - Goal: community contributions
 - Example: switchML (by MSR, KAUST, Intel)
- Intel's OpenTofino is a key catalyst for sharing open-source code



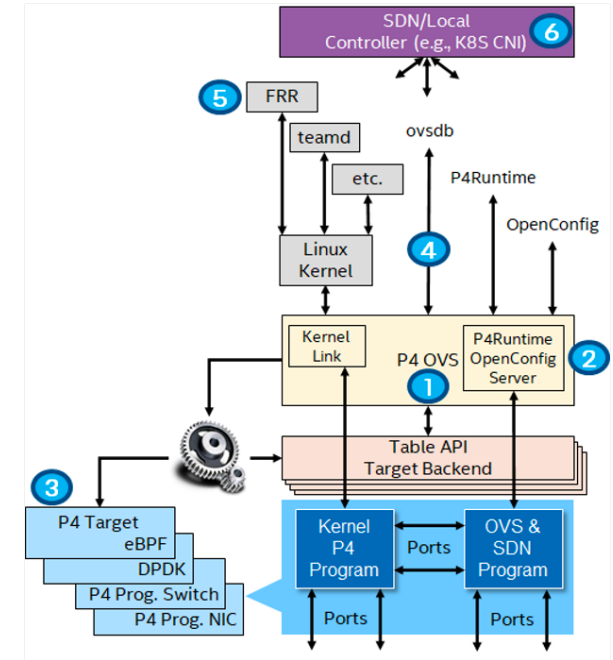
P4 Software Targets

P4-OvS

- A framework for executing P4 code in the Linux kernel
- Integrates with the standard networking stack as well as SDN controllers
- See on demand tutorial, presented by ONF, Orange, and Intel for more!

p4-DPDK

- A new compiler backend based on DPDK
- Facilitates fast user-space processing on CPUs
- See on-demand talk by Han Wang and Christian Dumitrescu for more!



P4 Semantics

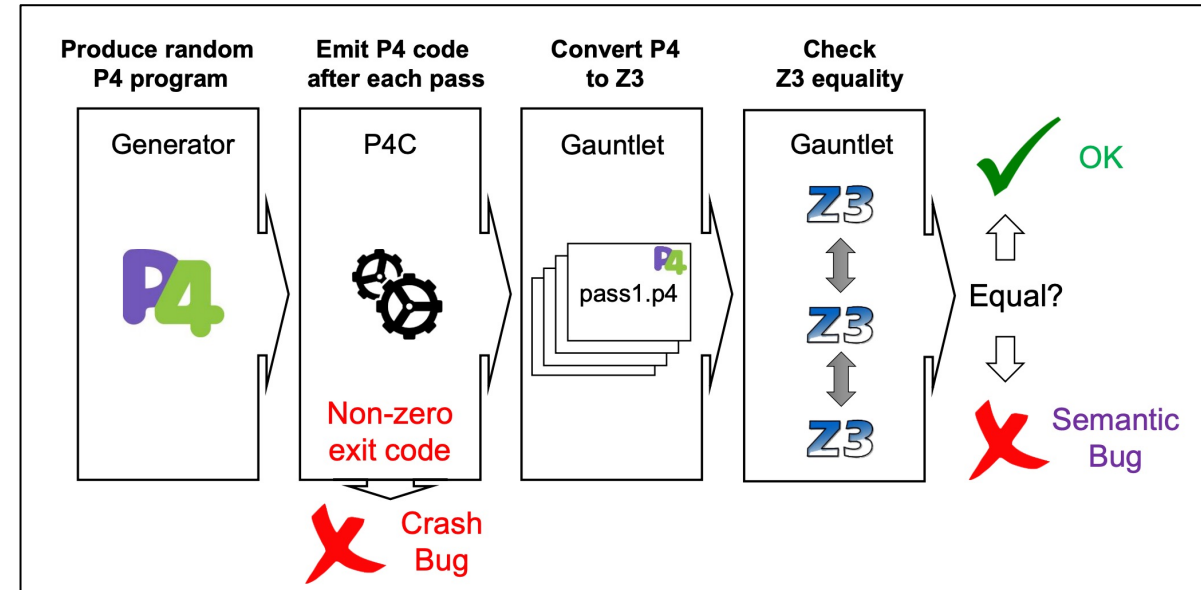
Some academics have started to build tools for reasoning about P4 code

Gauntlet [OSDI '20]

- Fuzzing and translation validation
- Used to validate every commit to the open-source P4 compiler

Petr4 [POPL '21]

- Formalization of P4₁₆'s type system and operational semantics
- Semantics is mechanized in Coq
- Being used by Princeton in DARPA's OPS-5G program



The screenshot shows a web browser displaying a Coq code file from a GitHub repository. The code defines inductive big-step semantics for P4 expressions. The visible code is as follows:

```
584
585 (** Expression big-step semantics. *)
586 Inductive expr_big_step {tags_t : Type} (ε : epsilon) : E.e tags_t -> V.v -> Prop :=
587 (* Literals. *)
588 | ebs_bool (b : bool) (i : tags_t) :
589   ( ε, BOOL b @ i ) ⇓ VB00L b
590 | ebs_bit (w : positive) (n : Z) (i : tags_t) :
591   ( ε, w W n @ i ) ⇓ w VW n
592 | ebs_int (w : positive) (z : Z) (i : tags_t) :
593   ( ε, w S z @ i ) ⇓ w VS z
594 | ebs_var (x : string) (τ : E.t) (i : tags_t) (v : V.v) :
595   ε x = Some v ->
596   ( ε, Var x:τ @ i ) ⇓ v
597 | ebs_slice (e : E.e tags_t) (τ : E.t) (hi lo : positive)
598   (i : tags_t) (v' v : V.v) :
599   eval_slice hi lo v = Some v' ->
```

On the right side of the screenshot, there is a logo consisting of a vertical chain of four links with a padlock at the bottom, next to the text "Verified Network Toolchain".

Future Directions

Language Design

Enhance flexibility, support modular programming, interface with other languages

Architecture

Develop PSA, PNA, and P4-StdLib

APIs

Integration with industry-standard network operating systems

Applications

Cultivate standard implementations and nurture emerging use cases

Education

Develop P4-based teaching materials for users around the world



P4 Community Updates



Distinguished Service Award

Distinguished Service Award



Noa Zilberman
Oxford University

“For dedicated service to the P4 community as an evangelist and promoter of the P4 language, designer of the P4 -> NetFPGA workflow, and co-chair of the P4 Education Working Group.”

p4.org

What: A new online home for the P4 community

Features

- Specifications
- Community
- Ecosystem
- Blog
- Learning resources
- Forms for contributing
- Fun videos



The screenshot shows the p4.org website homepage. At the top left is the P4 logo. The navigation menu includes 'Ecosystem', 'Specifications', 'Learn', 'Blog', 'Events', and 'Community'. The main heading is 'P4 Open Source Programming Language'. Below this is a paragraph explaining that P4 is a domain-specific language for network devices. A code block shows a routing table configuration. Another paragraph discusses how P4 gives application developers and network engineers more control over network behavior. A video player shows a man speaking. A 'Why P4?' section features a video thumbnail and a 'WATCH VIDEO 03:16' button. The page is decorated with a polar bear mascot and abstract shapes.

P4

Ecosystem Specifications Learn Blog Events Community

P4 Open Source Programming Language

Programming Protocol-independent Packet Processors (P4) is a domain-specific language for network devices, specifying how data plane devices (switches, NICs, routers, filters, etc.) process packets.

Before P4, vendors had total control over the functionality supported in the network. And since networking silicon determined much of the possible behavior, silicon vendors controlled the rollout of new features (e.g., VXLAN), and rollouts took years.

P4 turns the traditional model on its head. Application developers and network engineers can now use P4 to implement specific behavior in the network, and changes can be made in minutes instead of years.

```
table routing {
  key = { ipv4.dstAddr : 1pm; }
  actions = { drop; route; }
  size : 204 8;
}
control ingress() {
  apply {
    routing.apply();
  }
}
```

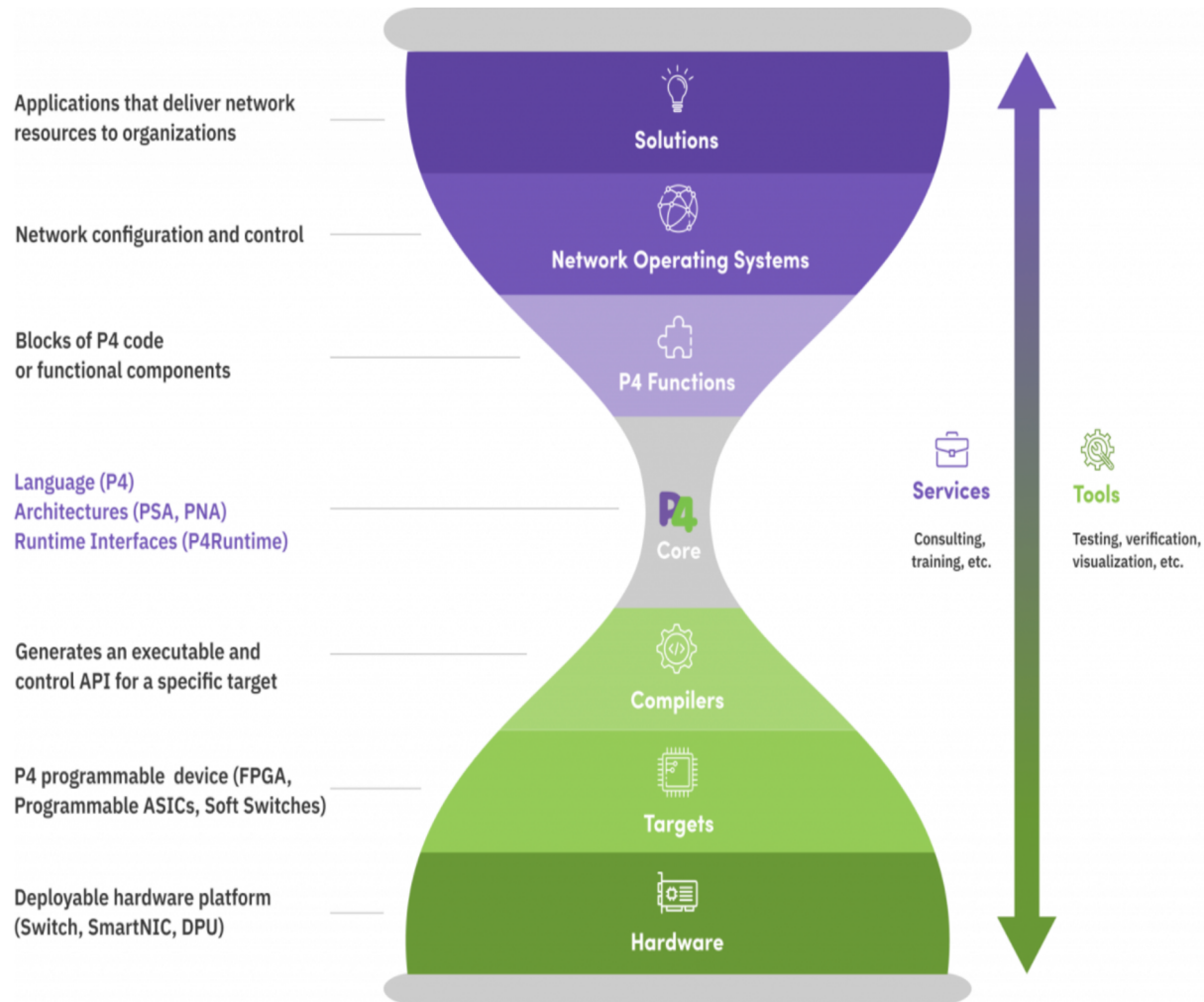
Why P4?
Industry luminaries share their perspectives

WATCH VIDEO 03:16



P4 Ecosystem

- **What:** A one-stop shop for P4 projects, products, and services
- Submit your data to be included!



If you have a product / project / service that utilizes P4 and would like to be featured as part of the P4 ecosystem we invite you to submit your information.

[SUBMIT DATA](#)

ECOSYSTEM CATEGORIES

- Compilers
- Hardware
- Network Operating Systems
- P4 Core
- P4 Functions
- Services
- Solutions
- Targets
- Tools

P4 FUNCTIONS

- L2 switching
- L3 routing
- L4-7 application processing
- Security / Filtering / ACL
- BNG
- INT
- UPF

PRODUCT / PROJECT TYPES

- Consulting
- Hardware
- Open Source
- Research
- Services

[See 3 more](#)

PETRA4 [MORE INFO](#)

Category: Tools

Product / Project Types: Tools

Overview: The Petr4 project is developing the formal semantics of the P4 Language backed by an independent reference implementation. Petr4 consists of a clean-slate definitional interpreter and a...

INTEL P4 STUDIO [MORE INFO](#)

Category: Compilers

Product / Project Types: Compilers

Overview: Intel® P4 Studio is a complete set of tools to develop, debug, and optimize P4 applications. For more information, visit the P4 Studio website

INTEL P4 COMPILER FOR DPDK [MORE INFO](#)

Category: Compilers

Product / Project Types: Compilers

Overview: p4c-dpdk is a compiler for P4 programs to execute on multi-core CPUs. Visit the p4c-dpdk repository.

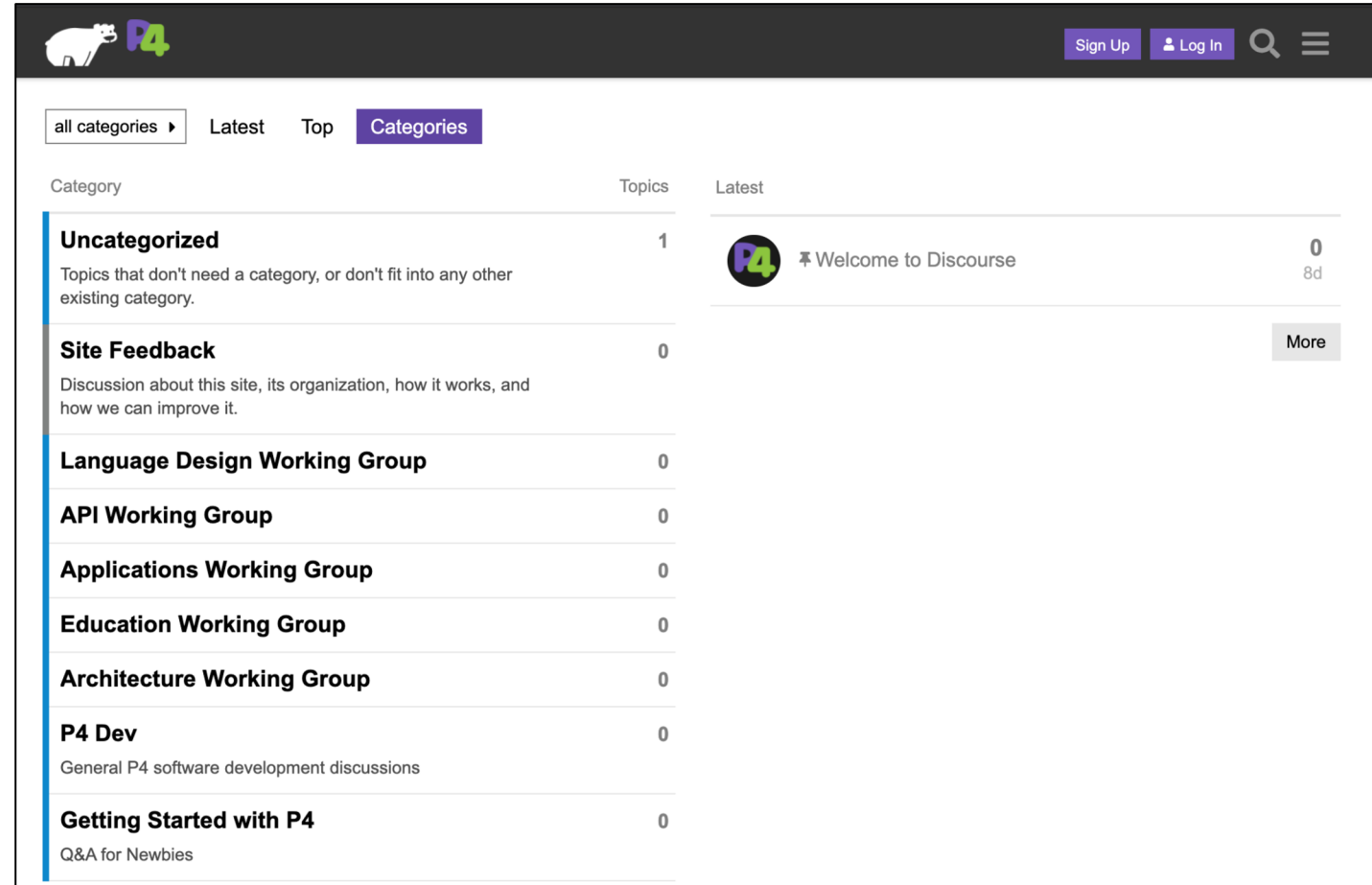


forum.p4.org

What: A new discussion forum for the P4 community

Features: Q&A, search, badges, permanent storage

Getting started: Create an ONF account, log in, and start posting!



The screenshot shows the forum.p4.org website interface. At the top, there is a navigation bar with the P4 logo (a purple 'P' and a green '4') and a cow icon. To the right of the logo are links for 'Sign Up' and 'Log In', along with search and menu icons. Below the navigation bar, there are tabs for 'all categories', 'Latest', 'Top', and 'Categories'. The main content area is divided into two columns. The left column is a list of categories with their respective topic counts:

Category	Topics
Uncategorized Topics that don't need a category, or don't fit into any other existing category.	1
Site Feedback Discussion about this site, its organization, how it works, and how we can improve it.	0
Language Design Working Group	0
API Working Group	0
Applications Working Group	0
Education Working Group	0
Architecture Working Group	0
P4 Dev General P4 software development discussions	0
Getting Started with P4 Q&A for Newbies	0

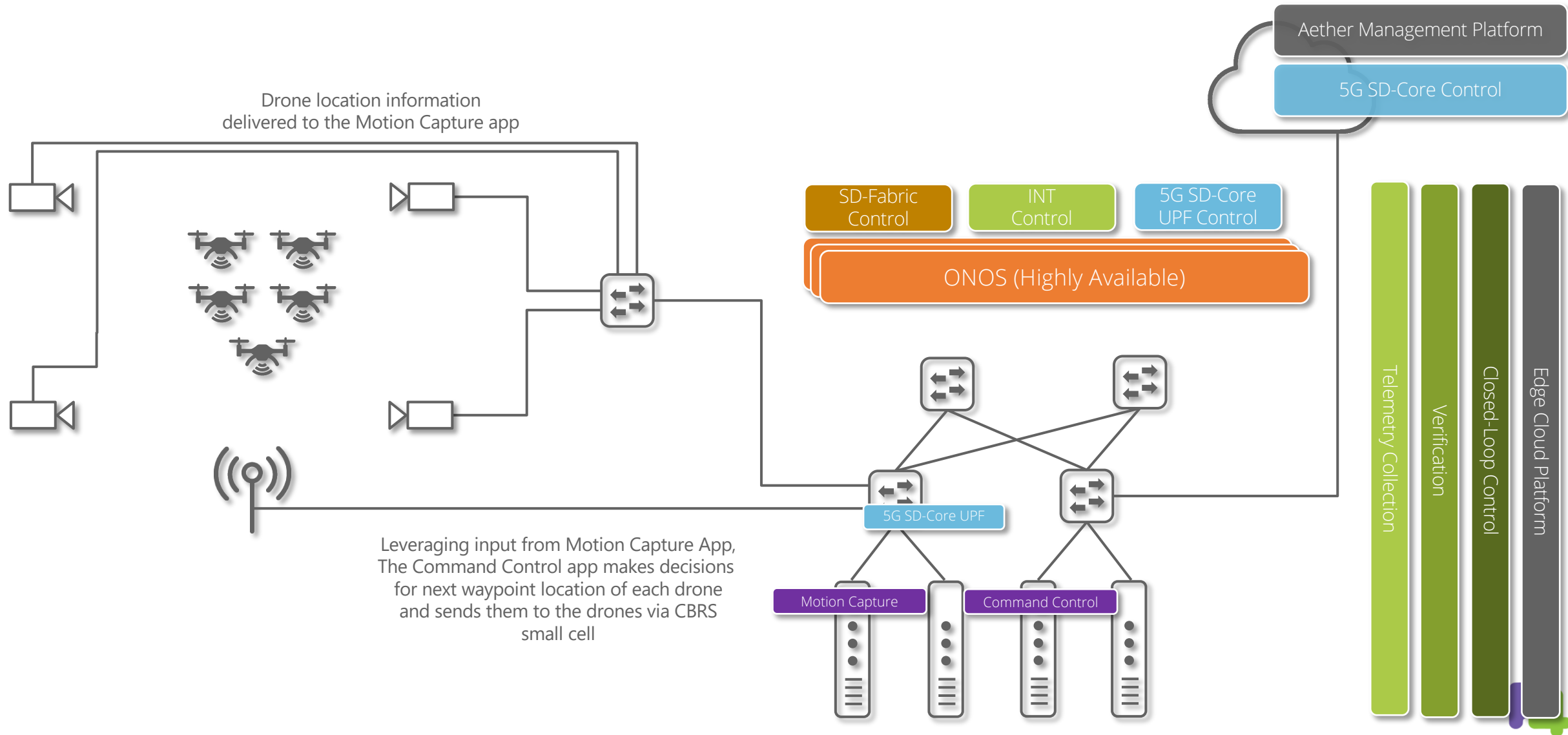
The right column shows the 'Latest' posts. The first post is a welcome message from a user with a P4 profile picture: 'Welcome to Discourse' with 0 replies and posted 8 days ago. A 'More' button is visible below the post.



A Neat Use Case...

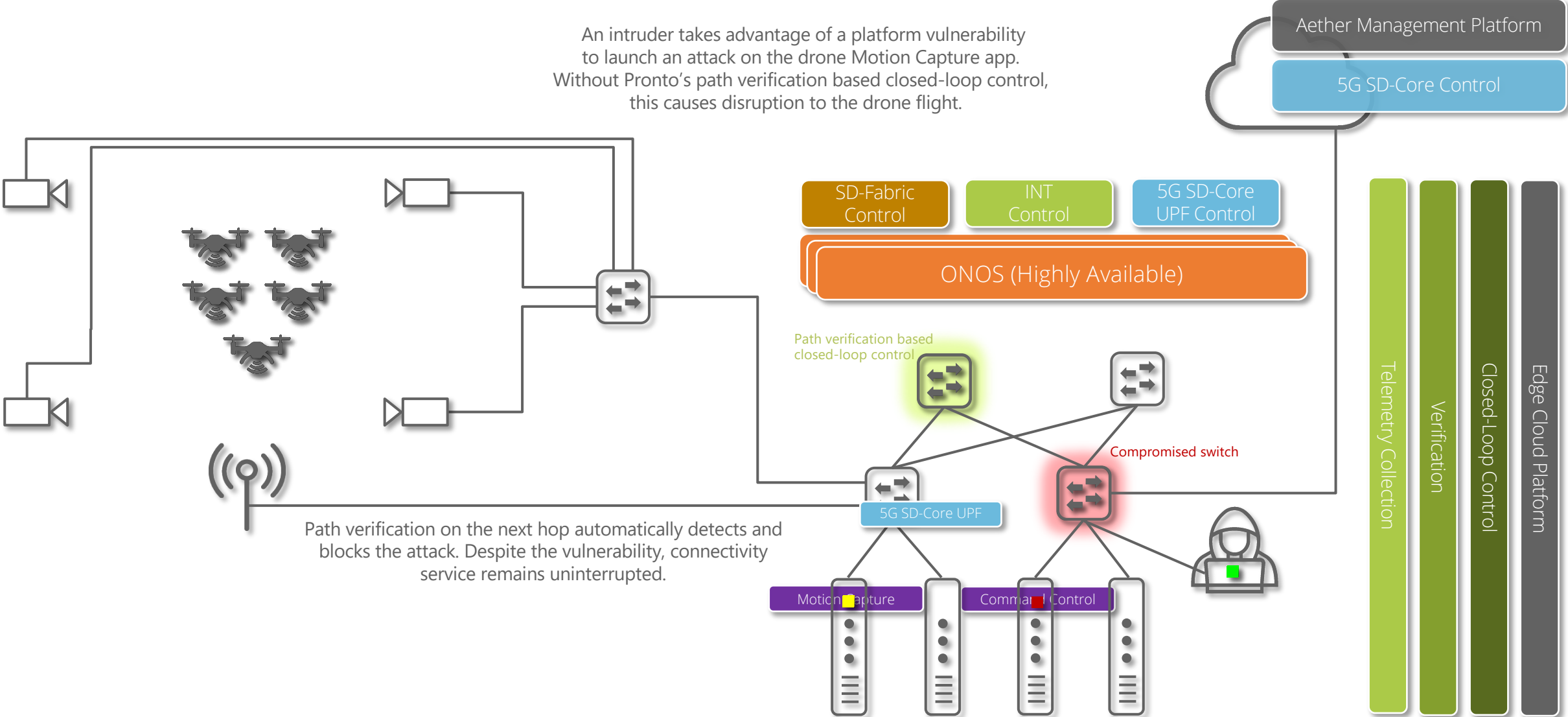
with Sundararajan Renganathan,
Bruce Sprang,
Nick McKeown,
the entire Pronto Team

Pronto Deployment at Stanford Flight Lab



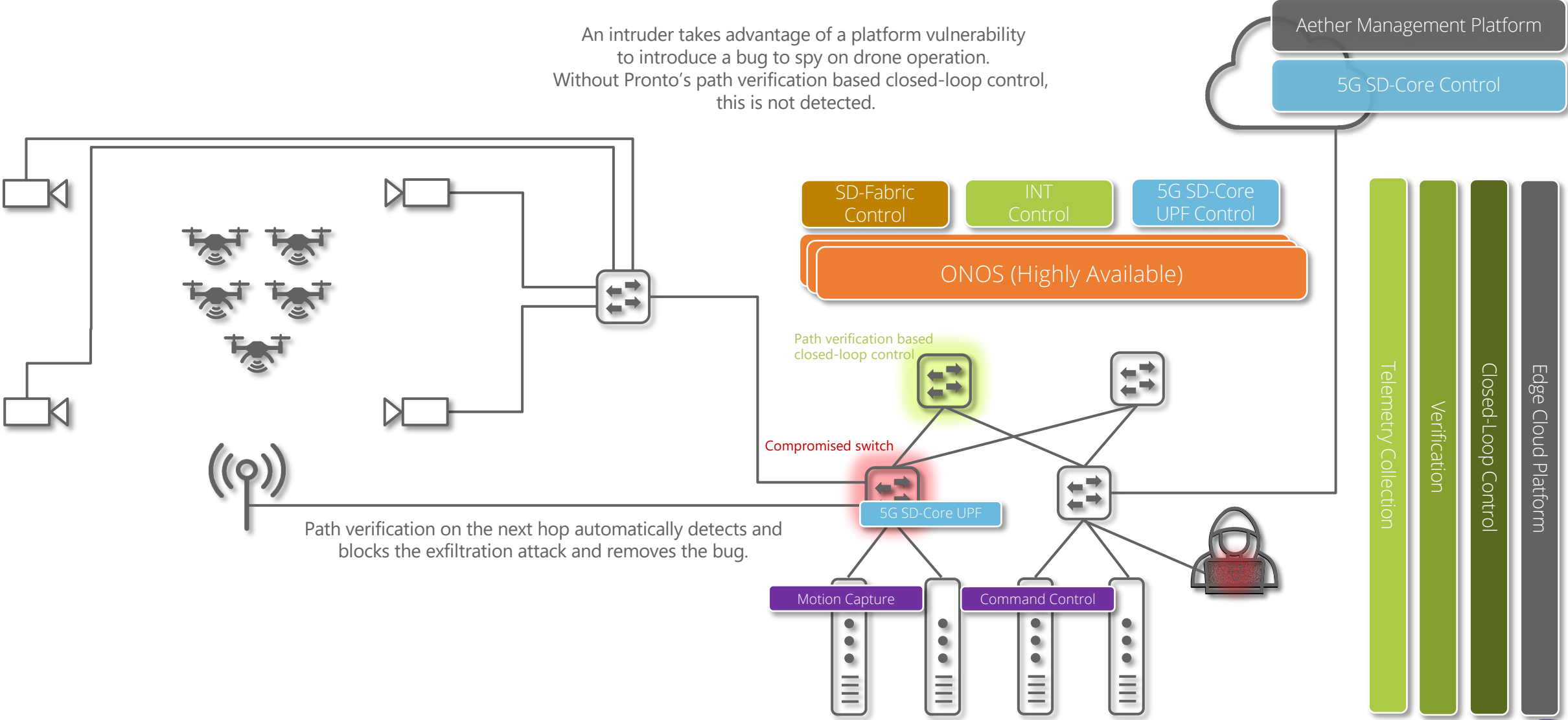
Runtime Path Verification: DDoS Attack

An intruder takes advantage of a platform vulnerability to launch an attack on the drone Motion Capture app. Without Pronto's path verification based closed-loop control, this causes disruption to the drone flight.



Runtime Path Verification: Exfiltration

An intruder takes advantage of a platform vulnerability to introduce a bug to spy on drone operation. Without Pronto's path verification based closed-loop control, this is not detected.



Path verification on the next hop automatically detects and blocks the exfiltration attack and removes the bug.



PRONTO

Wrapping Up...

P4's Guiding Principles

Open Community

- Open to anyone who wants to participate
- Decisions based on technical merit (not business or politics)

Strategic Goals

- Make P4 *the* de facto standard for packet processing, whether in hardware or software
- Find synergies with related frameworks (e.g., eBPF, XDP, DPDK, etc.)

Core Philosophy

- Declarative features with clear semantics
- Domain-specific constructs familiar to practitioners
- Predictable resource utilization and performance

Get Involved

- **Join the P4 Project!**
 - No fee to participate
 - Lightweight legal agreement based on Apache2 License
 - Possible to become an ONF Collaborator or Member
- **Participate in Working Groups**
 - Anyone with a good idea can help shape the future of P4
 - Open governance model with code of conduct
 - Decisions made by consensus on technical merits
- **Contribute to P4 Software**
 - Compiler (p4c)
 - Software targets (bmv2, OvS, DPDK)
 - Control-plane APIs (P4Runtime)
 - Tutorials & Documentation
 - Applications (INT, SwitchML, etc.)

Thank You

P4 Advisory Board

- Nate Foster (Cornell)
- Nick McKeown (Stanford)
- Guru Parulkar (ONF)
- Jennifer Rexford (Princeton)
- Amin Vahdat (Google)

P4 Technical Steering Team

- Nate Foster (Cornell, chair)
- Robert Soulé (Yale)
- Amin Vahdat (Google)
- Larry Peterson (ONF)
- Stefan Heule (Google)
- Andy Fingerhut (Intel)

Working Group Co-Chairs

- **Language:** Mihai Budiu, Nate Foster
- **APIs:** Antonin Bas, Waqar Mohsin
- **Architecture:** Andy Fingerhut, Gordon Brebner
- **Applications:** Mukesh Hira, Jeongkeun Lee
- **Education:** Robert Soulé, Noa Zilberman

P4 Workshop Program Committee

- Mina Tahmasbi Arashloo (Cornell)
- Mario Baldi (Pensando)
- Dan Daly (Intel)
- Nate Foster (Cornell)
- Carmelo Cascone (ONF)
- Minlan Yu (Harvard)
- Noa Zilberman (Oxford)

P4 Workshop Organizers

- Denise Barton (ONF)
- Rachel Everman (Intel)
- Brian O'Connor (ONF)
- Guru Parulkar (ONF)
- Larry Peterson (ONF)
- Michelle Roth (ONF)
- Timon Sloane (ONF)

P4 Workshop Sponsors

- Alibaba
- APS
- AT&T
- Google
- Intel
- NoviFlow
- Open Compute Project
- Vector Data

MAY 18-20
2021 P4
Workshop
Hosted by **ONF**

Enjoy!