Learn how to build secure infrastructure with these three tricks!

Nate Foster
Cornell University
Lord make me secure
...but not yet
Security Landscape

Image credits: Monzo & Istio
Security Landscape

Image credits: Monzo & Istio
Edge Cloud Architecture

Enterprise Datacenter

Edge IoT application may leverage cloud-based ML services

Legacy IoT Devices

Wi-Fi
Bluetooth LE
Wired

Gateway

CBRS Small Cell

Cellular IoT Devices

Communication Devices

Enterprise Datacenter

Edge IoT App

4G/5G Local Breakout

Aether Connected Edge (ACE)

Aether Management Platform

IoT & ML Services

Cloud Services

Aether Connectivity Platform

Control traffic for managed Edge-Cloud-as-a-Service

Network control traffic for managed Connectivity-as-a-Service

Image credit: ONF
Three Tricks

Verified Network Devices
Three Tricks

Verified Network Devices

Proof-Carrying Authorization
Three Tricks

- Verified Network Devices
- Proof-Carrying Authorization
- Timing-Safe Information Flow
**Verified Data Planes** [SIGCOMM ’18]

**Goal:** automatically verify behavioral properties for network devices

**Credits:** Bill Hallahan, Robert Soulé, many colleagues at Barefoot
Formal Foundations for P4

\[
\begin{align*}
\langle C, \Delta, \sigma, \epsilon, \text{exp} \rangle & \downarrow \langle \sigma', \text{val} \rangle \\
\langle C, x, \text{val} : x \rangle & \downarrow_{\text{match}} x(\text{exp}) \\
\langle C, \Delta, \sigma, \epsilon, \text{stmt} \rangle & \downarrow \langle \sigma', \epsilon', \text{sig} \rangle \\
\langle C, \Delta, \sigma, \epsilon, \text{decl} \rangle & \downarrow \langle \Delta', \sigma', \epsilon', \text{sig} \rangle
\end{align*}
\]

Expression evaluation
Match-action evaluation
Statement evaluation
Declaration evaluation

Petr4: Formal Foundations for P4 Data Planes

RYAN DOENGES, Cornell University, USA
MINA TAHMASBI ARASHLOO, Cornell University, USA
ALEXANDER CHANG, Cornell University, USA
NEWTON NI, Cornell University, USA
SAMWISE PARKINSON, Cornell University, USA
RUDY PETERSON, Cornell University, USA
ALAIA SOLKO-BRESLIN, Cornell University, USA
AMANDA XU, Cornell University, USA
NATE FOSTER, Cornell University, USA

P4 is a domain-specific language for specifying the behavior of packet-processing systems. It is based on an elegant design with high-level abstractions, such as parsers and match-action pipelines, which can be compiled to efficient implementations in hardware or software. Unfortunately, like many industrial languages, P4 lacks a formal foundation. The P4 specification is a 160-page document with a mixture of informal prose, graphical diagrams, and pseudocode. The reference compiler is complex, running to over 40KLoC of C++ code. Clearly neither of these artifacts is suitable for formal reasoning.

This paper presents a new framework, called Petr4, that puts P4 on a solid foundation. Petr4 uses standard elements of the semantics engineering toolkit, namely type systems and operational semantics, to build a compositional semantics that assigns an unambiguous meaning to every P4 program. Petr4 is implemented as an OCaml prototype that has been validated against a suite of over 750 tests from the reference implementation. While developing Petr4, we discovered dozens of bugs in the language specification and the reference implementation.
HyperFlow [CCS ’18]

Goal: timing-safe information flow security with expressive policies and strong assurance

- **Software**
  - DIFC policies: confidentiality, integrity → Mutually distrusting yet communicating parties

- **ISA**
  - New HW-SW contract for timing-safe IFC → Encode expressive security policies in hardware

- **MicroArch**
  - Tagged architecture for enforcement → Remove timing channels

- **HDL**
  - Secure HDL for information flow security → Timing-sensitive non-interference

**Credits:** Ed Suh and Andrew Myers
Goal: specify and enforce fine-grained network policies with distributed authorization

Credits: Christian Skalka, David Darais, Minseok Kwon
Takeaways…

verified network devices

proof-carrying authorization

timing-safe information flow