Powerful Properties
of
Packet Processing
with
P4

Aaron A. Glenn
Internetworking Curmudgeon

@aag@bsd.network
@networkservice

DENOG 10
GERMAN NETWORK OPERATORS GROUP

This talk is RFC1925 compliant

This talk is about SDN

Defining Software Defined Networking

Defining Software Defined Networking

Separate control plane from data plane

Defining Software Defined Networking

Separate control plane from data plane

Open & consistent API directly to data plane

Defining Software Defined Networking

Separate control plane from data plane

Open & consistent API directly to data plane

Defining Software Defined Networking

✓ Separate control plane from data plane

Open & consistent API directly to data plane

Defining Software Defined Networking

✓ Separate control plane from data plane

✗ Open & consistent API directly to data plane

OpenFlow
CONFORMANT

Defining Software Defined Networking

✓ Separate control plane from data plane

✗ Open & consistent API directly to data plane

Programming

Protocol-Independent

Packet

Processors

Dispelling unfortunate common misconceptions P4 is not...

...a general purpose language

computation and memory is bounded

...only for Barefoot Tofino switches

represent any packet forwarding device

...OpenFlow 2.0

völlig falsch. gänzlich.

P4 is a domain specific language for programming protocol-independent packet processors

P4 specifies packet forwarding behavior

enabling *reconfiguration* of parsing and processing

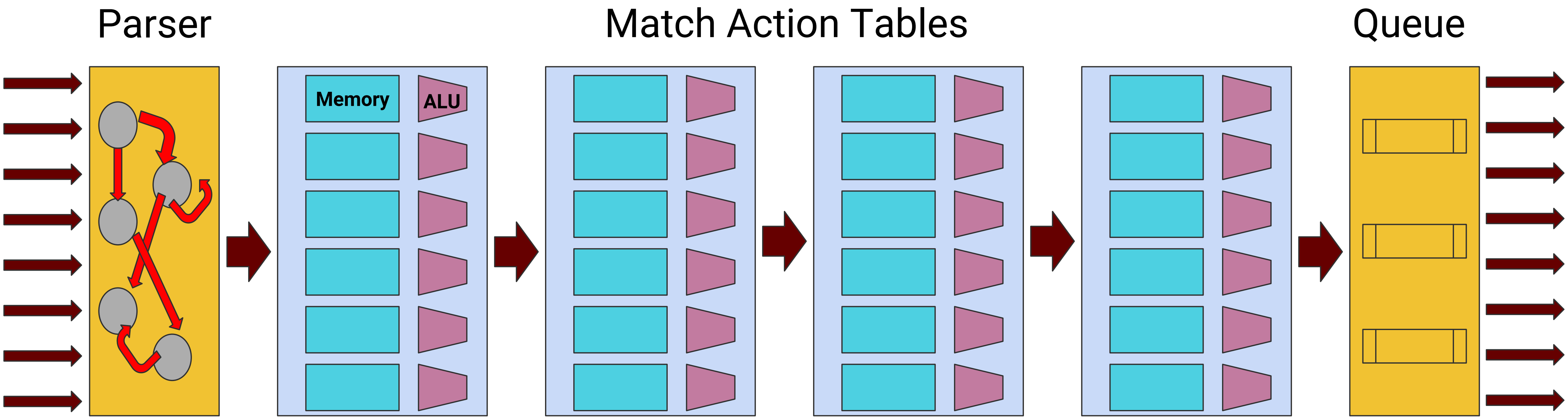
P4 is protocol independent

program defines packet headers and processing logic

P4 is target independent

describe forwarding behavior regardless of underlying hardware

P4 is a behavioral description language to unambiguously define the forwarding plane



Introducing a programming language
in 10 minutes or less

Defining the $P4_{16}$ approach

Target

a specific hardware or software implementation

Architecture

specific set of programmable components available

Platform

Architecture implemented on a Target

Elements of the P4_{16} language

Parsers
finite state machine mapping packets into headers + metadata

Controls
define tables, actions, and control flow

Expressions
< > * % ? << >> && != ==

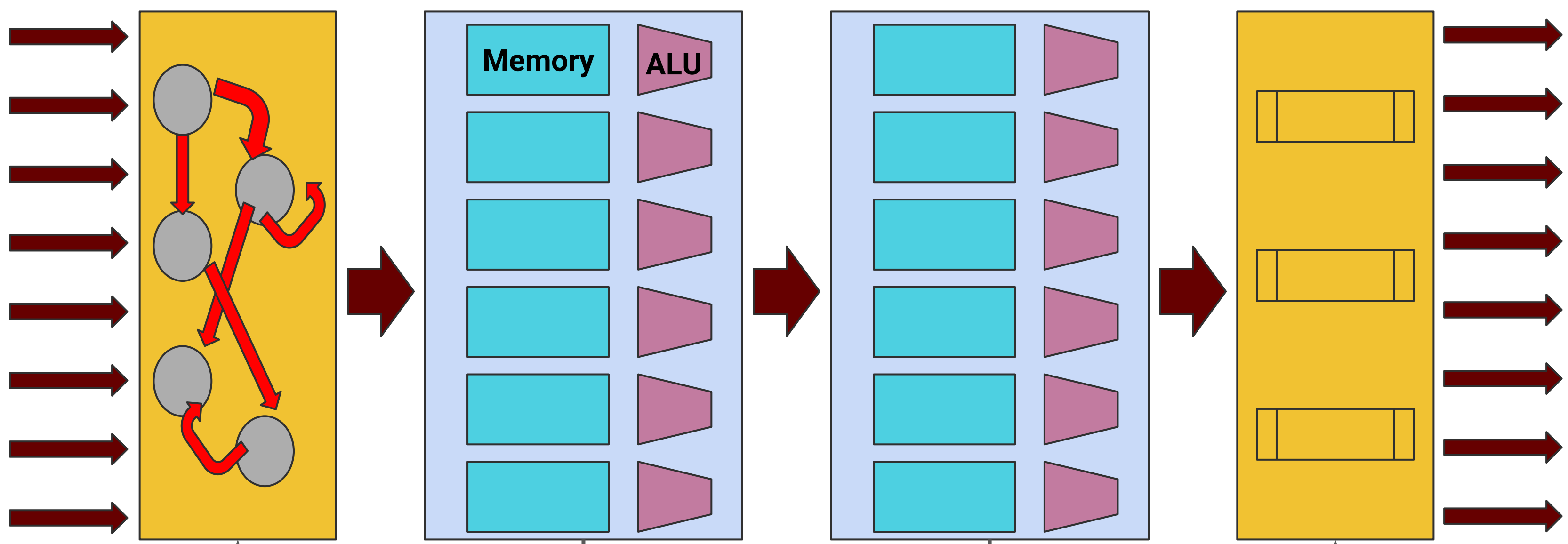
Elements of the $P4_{16}$ language (continued)

Data types
statically typed language with limited range of type casting

Architecture description
definition of interfaces to programmable blocks

External interfaces
architecture specific objects and functions defined by description

Main components of a P4 program




```
V1Switch(
 MyParser(),
 MyVerifyChecksum(),
 MyIngress(),
 MyEgress(),
 MyComputeChecksum(),
 MyDeparser()
) main;
```

```
#include <core.p4>
#include <v1model.p4>
```

Libraries

```
const bit<16> TYPE_IPV4 = 0x800;
typedef bit<32> ip4Addr_t;
header ipv4_t {...}
struct headers {...}
```

Declarations

```
parser MyParser(...) {
  state start {...}
  state parse_ethernet {...}
  state parse_ipv4 {...}
}
```

Parse packet headers

```
control MyIngress(...) {
  action ipv4_forward(...) {...}
  table ipv4_lpm {...}
  apply {
        if (...) {...}
  }
}
```

Control flow to modify packet

```
control MyDeparser(...) {...}
```

Assemble modified packet

```
V1Switch(
 MyParser(),
 MyVerifyChecksum(),
 MyIngress(),
 MyEgress(),
 MyComputeChecksum(),
 MyDeparser()
) main;
```

“main()”

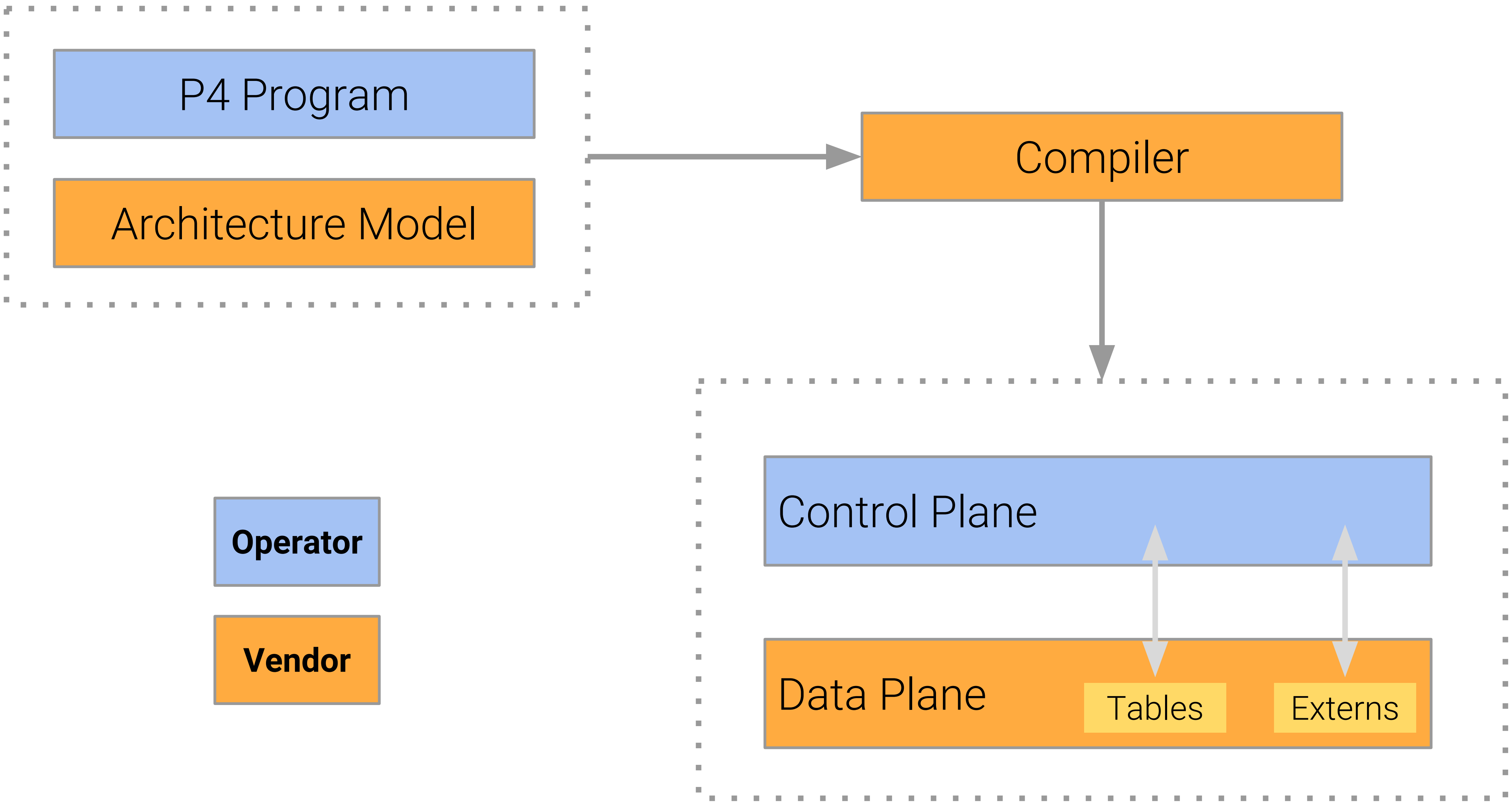
From slide 105 of *https://adv-net.ethz.ch/pdfs/01_intro.pdf*

Pipeline	Functionality	Language constructs	Source code
Ingress	Virtual LANs (VLANs)	Action profiles, header lists	port_vlan_mapping.p4
Ingress	Spanning Tree Protocol	Exact match	spanning_tree.p4
Ingress	Common logic to handle routing for NVGRE [12], VXLAN [24], and ERSPAN [5] tunnels	Exact match	outer_rmac.p4 ipv4_dest_vtep.p4 ipv4_src_vtep.p4 tunnel.p4
Ingress	Packet validation	Exact match	validate_packet.p4
Ingress	ECMP	Action profiles, Action selectors, Field lists	ecmp_group.p4
Ingress	IP forwarding	Longest-prefix match	ip_fib.p4
Ingress	Link Aggregation (LAG)	Action profiles, Action selectors, Field lists	lag_group.p4
Ingress	MAC and IP Access Control Lists	Counters	mac_ip_acl.p4
Ingress	Packet Mirroring	Clone packet	mirror_acl.p4
Ingress	MAC learning	Digest Generation	learn_notify.p4
Egress	Tunnel decapsulation for NVGRE, VXLAN, ERSPAN	Add headers	tunnel_decap.p4
Egress	Tunnel encapsulation for NVGRE, VXLAN, ERSPAN	Remove headers modify_field_with_hash	tunnel_rewrite.p4 tunnel_src_rewrite.p4 tunnel_dest_rewrite.p4
Egress	VLAN tag add/removal	Header lists	egress_vlan_xlate.p4
Egress	MTU Check	Ternary match on mtu_check_fail field	egress_system_acl.p4
Egress	Process packets to/from switch CPU	Add/remove header	cpu_rewrite.p4

Table 1: Feature list of DC.p4 along with language constructs and corresponding source files

Sivaraman, Anirudh, et al. "DC.p4: Programming the forwarding plane of a data-center switch." *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*. ACM, 2015

Elements of a P4 environment



Incomplete list of P4 targets

eBPF

eXpress Data Path

Vector Packet Processing

Netcope VHDL FPGA

Xilinx PX FPGA

Barefoot Tofino ASIC

P4GPU (CUDA)

P4FPGA (Verilog)

PISCES (OpenVSwitch)

T4P4S (DPDK)

MACSAD (ODP+DPDK)

Netronome SmartNIC (NPU)

	LoC	Methods	Method Size
OVS	14,535	106	137.13
PISCES	341	40	8.53

Table 2: *Native OVS compared to equivalent baseline functionality implemented in PISCES.*

		Files Changed	Lines Changed
Connection Label:	OVS [70, 71]	36	633
	PISCES	1	5
Tunnel OAM Flag:	OVS [27, 28]	21	199
	PISCES	1	6
TCP Flags:	OVS [61]	20	370
	PISCES	1	4

Table 3: *The number of files and lines we needed to change to implement various functionality in P4, compiled with PISCES, compared to adding the same functionality to native OVS.*

Muhammad Shahbaz, Sean Choi, Ben Pfaff, Changhoon Kim, Nick Feamster, Nick McKeown, and Jennifer Rexford. *PISCES: A Programmable, Protocol-Independent Software Switch*. In ACM SIGCOMM, Florianópolis, Brazil, August 2016.

P4Runtime, the control plane interface to P4

protobuf based API definition

the standard is a collection of protobuf files

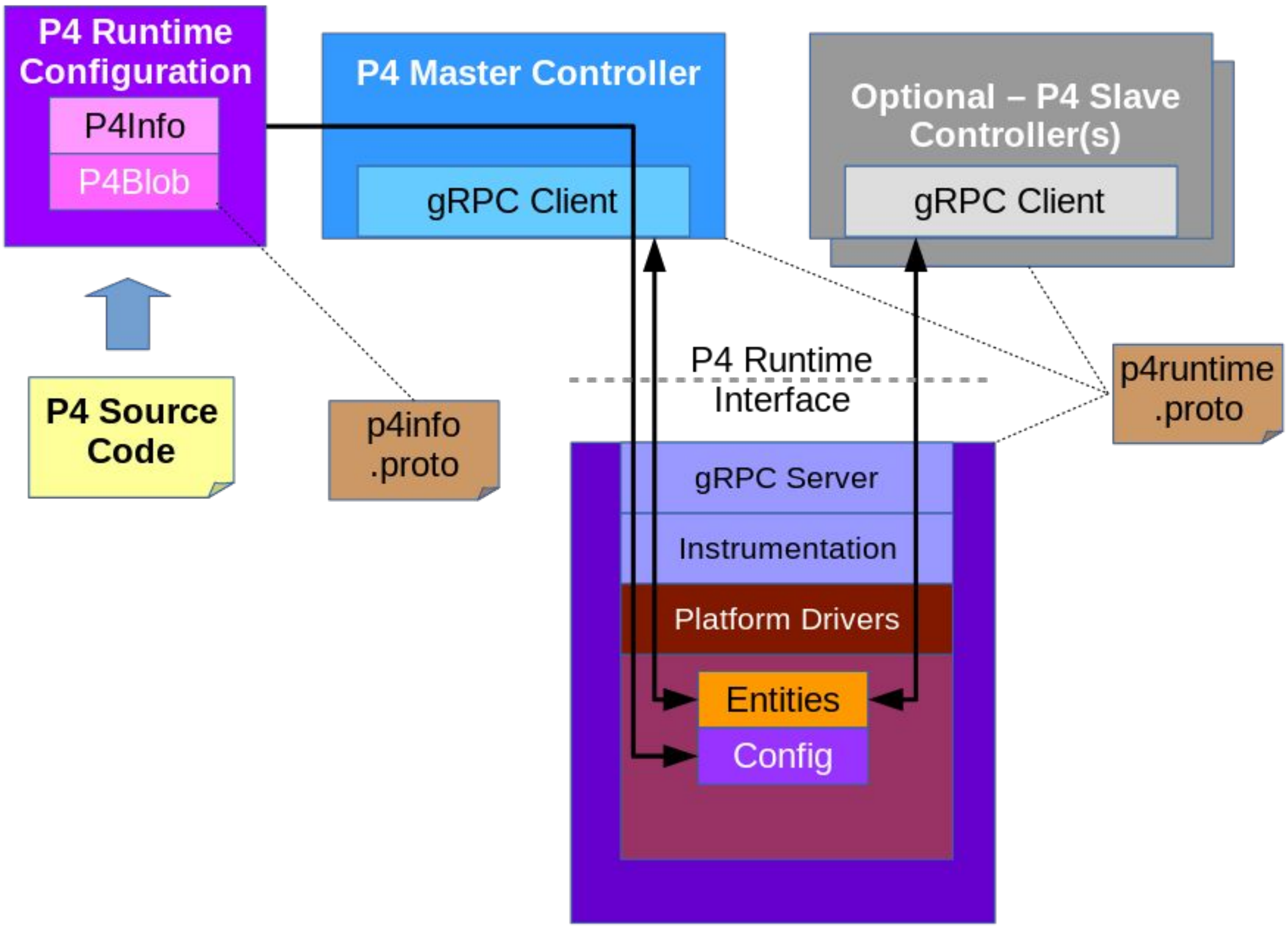
P4 program independent

API does not change when P4 program has changed

reconfigurability in the field

push a new P4 program without recompiling everything

P4Runtime reference architecture



P4Runtime for fixed function devices

No P4 Source Available, P4Info Available

presumably how well-known vendors “support P4”

“it is not necessary to have a P4 source program to begin with, since the controller does not use it. From the standpoint of controller (not pipeline) implementers, the P4 source code is just helpful documentation. Some parties may wish to keep their P4 source code private. [...] As long as the target supports the operations implied by the P4Info file, the underlying implementation is moot.”

https://s3-us-west-2.amazonaws.com/p4runtime/docs/v1.0.0-rc3/P4Runtime-Spec.html

P4 + P4Runtime is a powerful *building block*

scheduling algorithms

expressing data plane *algorithms* is challenging

higher level abstractions

NetKAT, Domino, various formal verification techniques

In-Network Computing

“a dumb idea whose time has come”

In-network Neural Networks

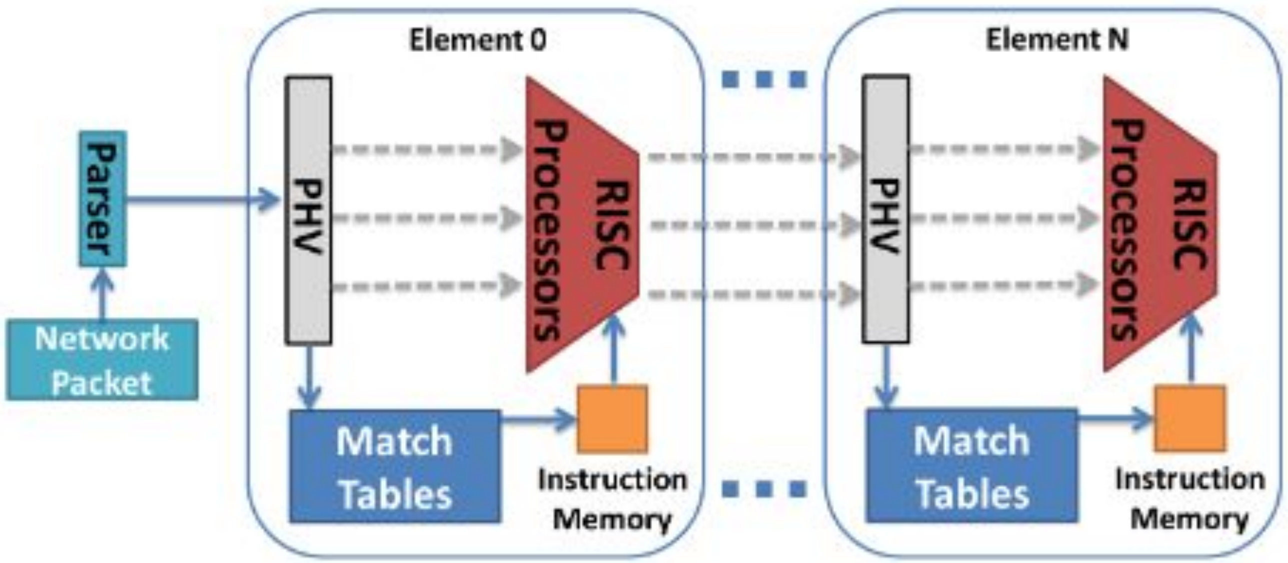


Figure 1: A schematic view of a switching chip's pipeline.

Siracusano, Giuseppe, and Roberto Bifulco. "In-network Neural Networks." *arXiv preprint arXiv:1801.05731* (2018).

Complex Event Processing

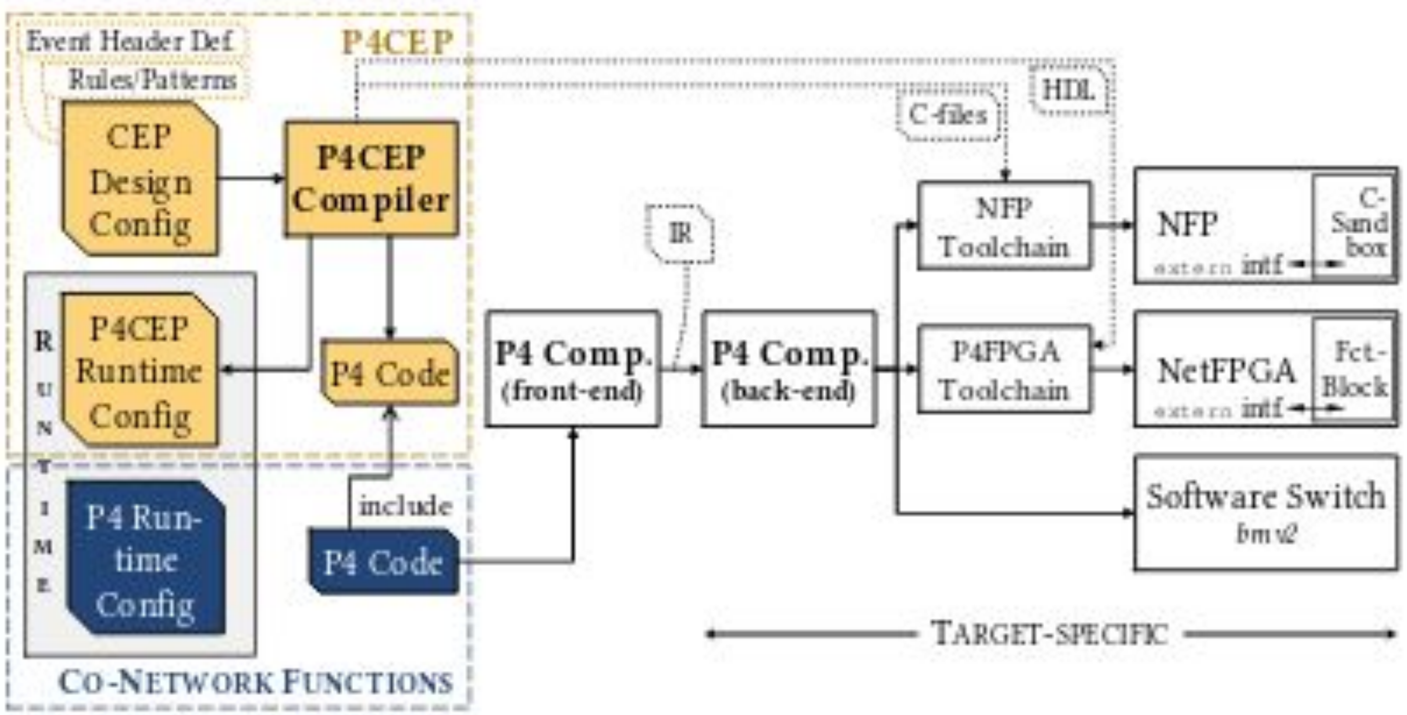
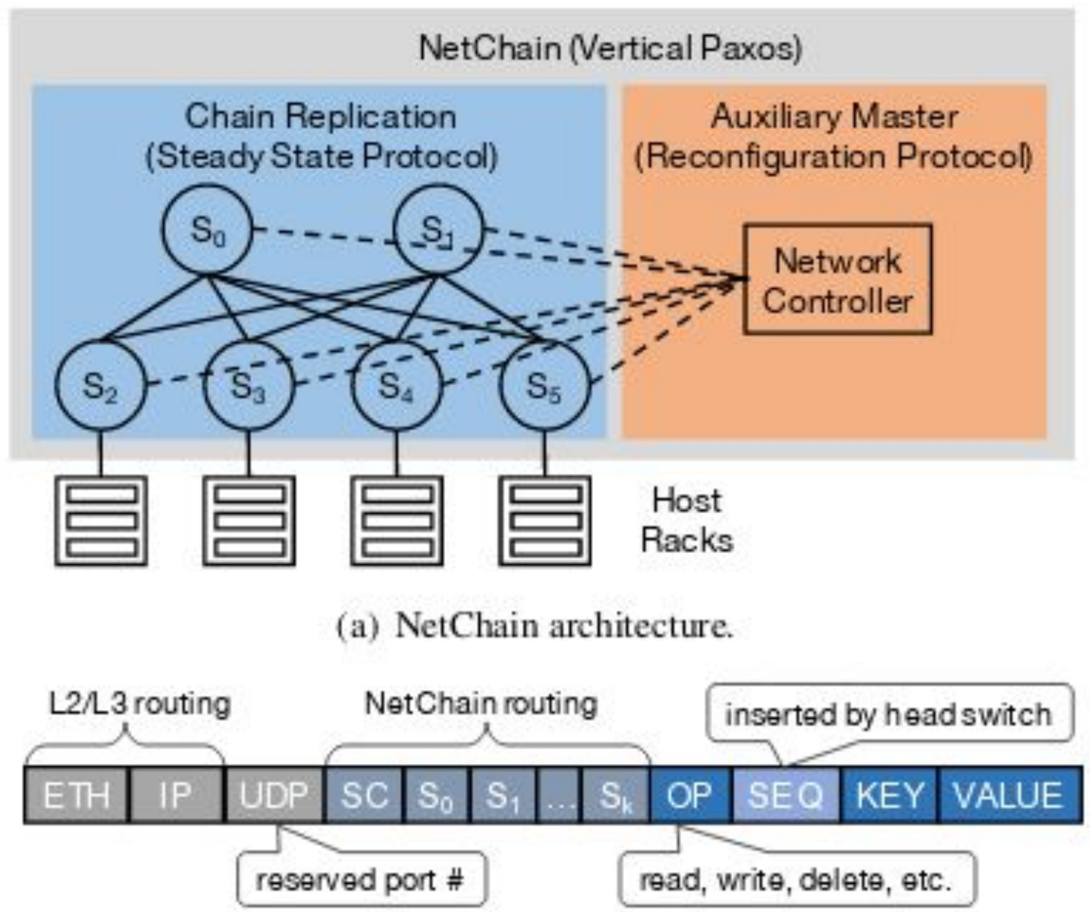


Figure 2: P4CEP workflow: design-time components and source files involved in building P4CEP for different targets.

Kohler, Thomas, et al. "P4CEP: Towards In-Network Complex Event Processing." *arXiv preprint arXiv:1806.04385* (2018).

In-Network Coordination & Consensus



(b) NetChain packet format.

Figure 2: NetChain overview.

Jin, Xin, et al. "NetChain: Scale-Free Sub-RTT Coordination." *15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18)*. USENIX} Association}, 2018.

Benefits of Data Plane Programmability

C
- Control and Customization. Make the device behave exactly as you want

R
- Reliability. Reduce the risk by removing unused features

E
- Efficiency. Reduce energy consumption and expand scale by doing only what you need

A
- Add new features on your schedule

T
- Telemetry. Be able to see inside the Data Plane

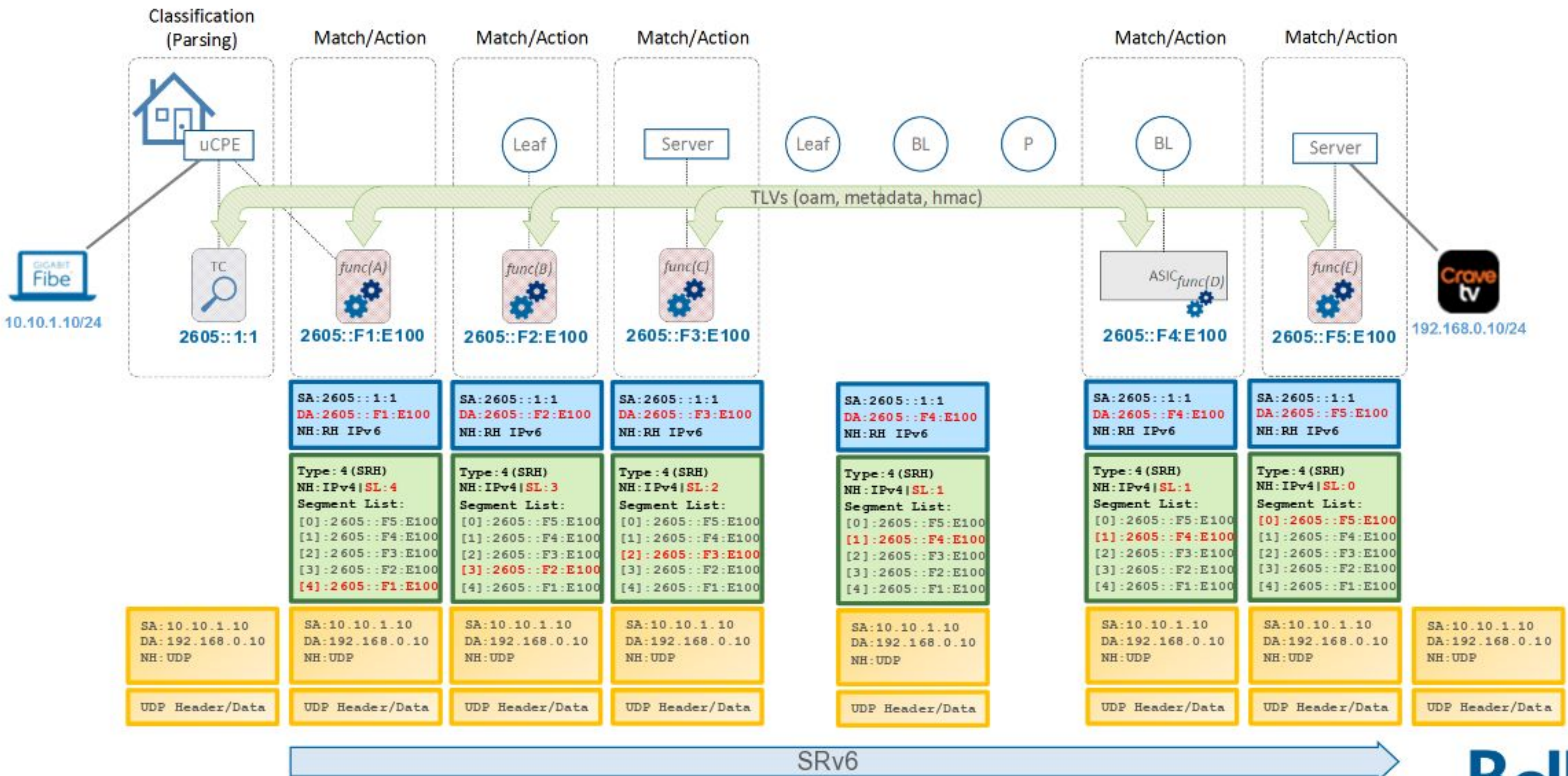
E
- Exclusivity and Differentiation. No need to share your IP with the chip vendor

More resources on P4

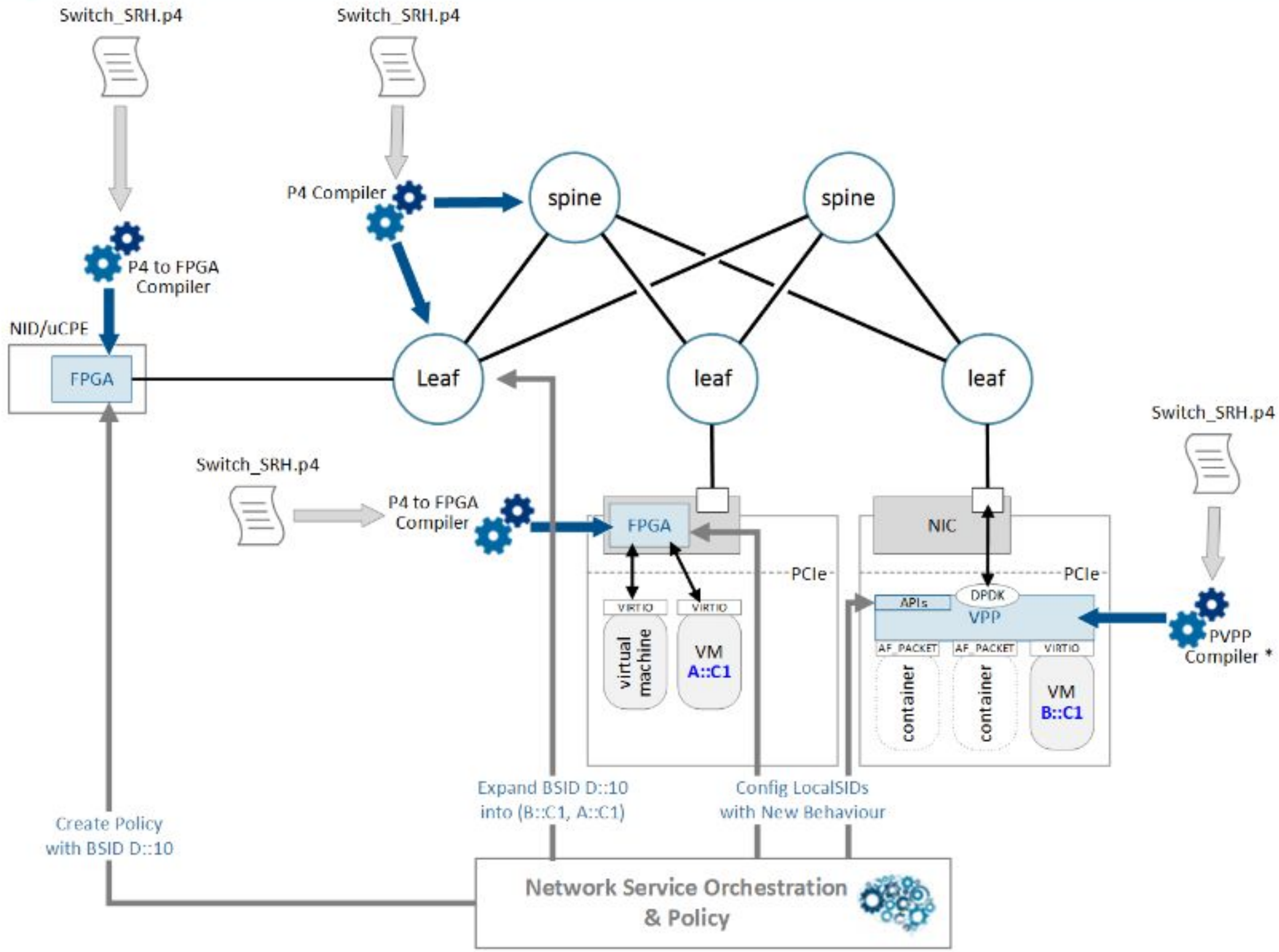
Andy Fingerhut's P4 Guide	github.com/jafingerhut/p4-guide
P4 Association specifications	p4.org/specs
Tutorials repository (WIP)	github.com/p4lang/tutorials
@p4works	p4.works
email me, please	hallo@p4.works

The “*Network-as-an-ASIC*”

- Traffic classification at the edge of the network → e.g. parsing.
- Simplified *Match/Action* primitive looking at the function Identifier.
- Contextual metadata carried through TLVs
- Programming at All Layers
  - P4 to define the END and TRANSIT behaviors in data plane.
  - SRv6 to define the “*end to end network behavior*”



Extending the Network With a New Behavior



- Program the new data plane behavior in P4.
- Compile to multiple targets.
- Program the control plane and/or northbound APIs required.
- Configure the new function using the behavior (e.g. attach a LocalSID to the behavior).

Continuously Extending the Network … With no new Protocol

Bell