Refactoring OpenFlow Solutions to P4Runtime

Jesper Eriksson
Co-Founder and VP Product Management
NoviFlow

Agenda

Refactoring OpenFlow Solutions to P4Runtime

Thursday, September 12 • 4:30pm - 5:00pm:

This presentation explores the evolution of SDN from OpenFlow to P4 and P4Runtime, outlining the key differences and advantages of each. Special focus will be placed on the value of the programmable match-action pipeline, the programmable parser and the OpenFlow and P4Runtime protocols, including a look at specific commercial implementations of each, and the refactoring of a specific commercial OpenFlow solution into a P4Runtime solution. Key benefits in this transition will then be examined such as increased flexibility responding to new customer requirements and shorter time to market for implementing new features.

NoviFlow

FOUNDED
2012

FOCUS
Optimizing the Network/Cloud Edge

PRODUCTS
Networking software for WAN and Cybersecurity

Business Model
Software Licensing
Systems Sales

MEF NoviFlow 2017 AWARDS FINALIST

RED HERRING 100 WINNER GLOBAL

Production deployments worldwide
by global network operators,
hyperscale datacenters,
enterprises and government agencies

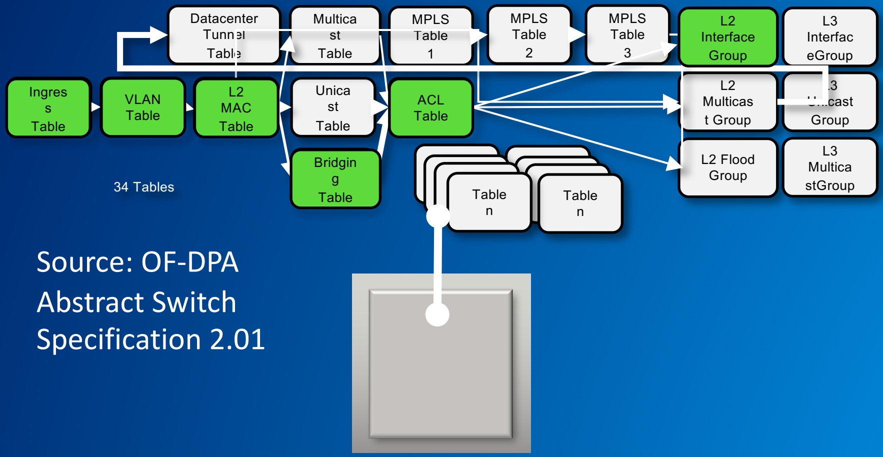
50 Technology Fast 50 2015 CANADA Deloitte.

LAZARIDIS INSTITUTE

One of 10 Canadian Scale-Up companies in 2016

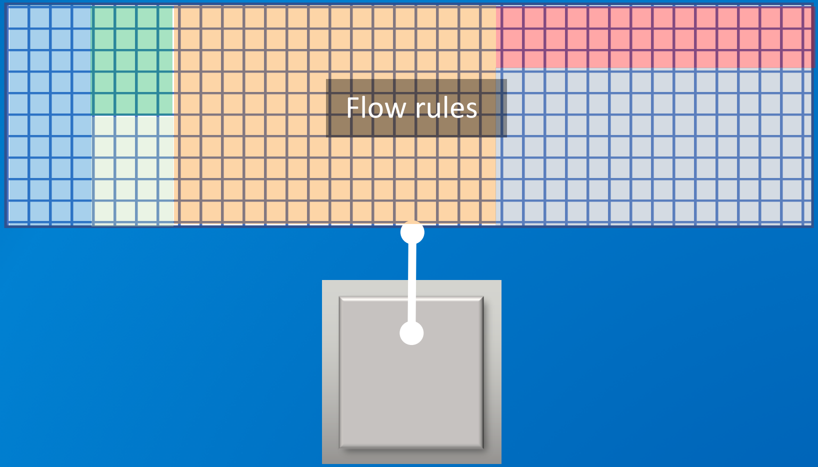
Match-Action Pipeline Processing

Fixed ASIC



Source: OF-DPA
Abstract Switch
Specification 2.01

Programmable Silicon



- A fixed set of match-action tables are defined in the silicon:
  - Fixed number of tables, table sizes with predefined match fields and actions to be used in each table
- The application programmer tries to map the application into this fixed match-action pipeline
- **Bottoms up programming paradigm**

- No prior set of match-action tables defined in the silicon
- The application programmer creates the match-action pipeline to specifically meet the needs of the application
  - Assigns the number of tables and the size, the match fields and the actions to be used in each table
- **Top down programming paradigm**

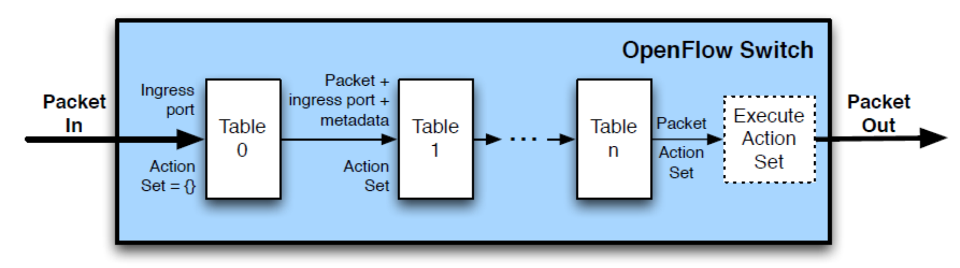
Why Programmable Match-Action Pipelines?

- Faster introduction of new networking functionality and protocols
  - VxLAN, INT, SRv6, …
  - Fast prototyping of new capabilities
- Features are defined in software and not in hardware
  - No forced obsolesces of networking equipment
  - Repurposing of networking equipment
- Disaggregation of networking hardware and software
  - Software not locked into networking hardware
  - Control and Data Planes scale independently

OpenFlow Match-Action Pipeline

- OpenFlow 1.4 Sample Implementation
- Provides the application programmer with a programmable match-action pipeline
- The supported match fields, instructions/actions are defined in the various OpenFlow specifications:
  - OpenFlow 1.0, 1.1, 1.2, 1.3, 1.4, 1.5… (Protocol dependent)
  - The Experimenter Extensions allows companies to innovate:
    - Experimenter match fields, actions and more

OpenFlow Match-Action Pipeline



Source: OpenFlow 1.3 Specification

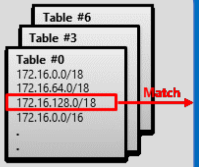
Fully programmable OpenFlow pipeline:

- Supports all OF 1.4 match fields (41), instructions (6) and actions (56)
- Any match field(s), action(s) and instruction(s) in any table
- Table type individually configurable:
  - **Wildcard match** (TCAM) (OpenFlow standard)
    or
  - **Exact Match** (DRAM)
- Each table's width and depth is individually configurable through CLI

Wild Card Matching Use Cases:

- OpenFlow standard (17/41)
- L3 Forwarding, e.g. LPM
- ACLs/Firewalls e.g. subnet matching
- ….

TCAM

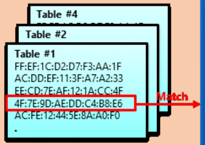


Up to **1M** flow entries with wild cards in up to **60** tables

Exact Matching Use Cases:

- OpenFlow standard (24/41)
- L2, MPLS Forwarding
- Segment Routing
- Service chaining
- ACLs, e.g. MAC address
- …

DRAM



Up to **6M** exact match flow entries in up to **60** tables

OpenFlow Experimenters

- Facility within the OpenFlow protocol for defining additional match fields, actions and more
  - Extensively used by NoviFlow to plug gaps in the OpenFlow specification identified by our customers
- Experimenters propagate through to the OpenFlow protocol:

Flow entries can be defined with the NOVI_ACTION_SEND_REPORT action and be loaded in the switch using the OFPT_FLOW_MOD message including the following ofp_action as one of the actions:

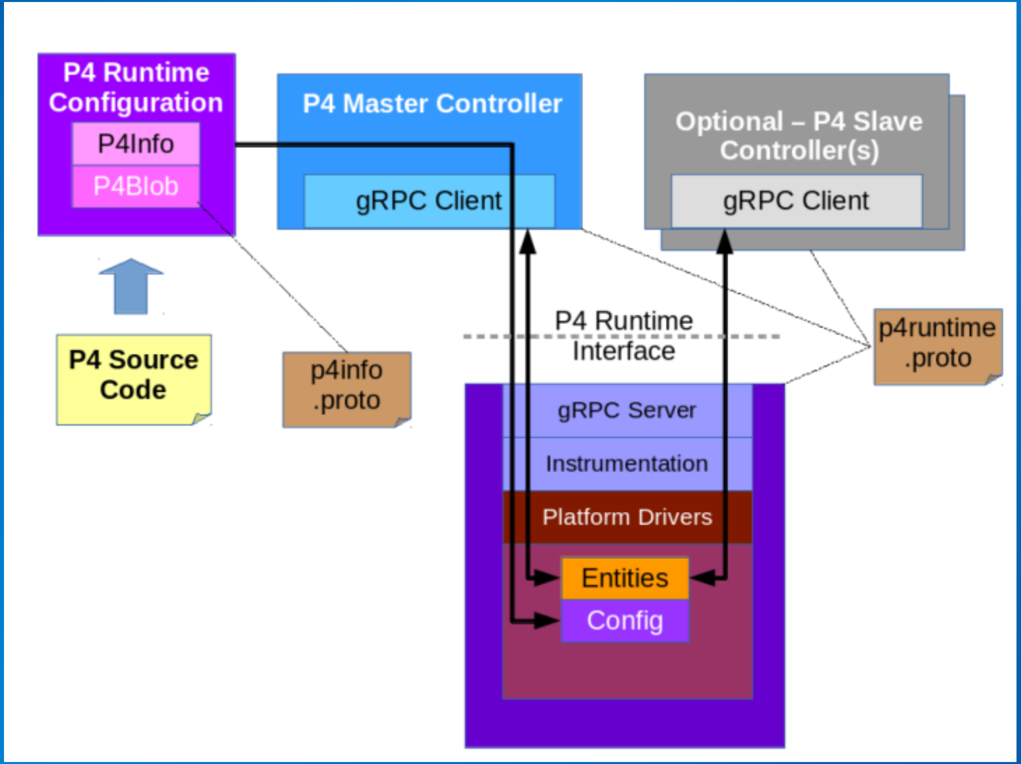
```
struct novi_action {
  uint16_t type;               /* ofp_action_type OFPAT_EXPERIMENTER 0xffff */
  uint16_t len;                /* Length of action, including the header and any
                               padding to make it 64-bit aligned */
  uint32_t experimenter;       /* NoviFlow experimenter ID 0xff000002 */
  uint8_t customer;            /* Customer ID 0xff*/
  uint8_t reserved;            /* Reserved for future use 0x00 */
  uint16_t novi_action_type;   /* NoviFlow action type
                               NOVI_ACTION_SEND_REPORT 0x000F*/
  uint8_t pad[4];              /* 4 bytes of all zeros */
};
```

OpenFlow Experimenters	Use cases
Ethernet Payload Matching	MPLS Forwarding (e.g. MPLS Segment Routing) and Cybersecurity
MPLS Payload Matching	MPLS Forwarding (e.g. MPLS Segment Routing) and Cybersecurity
L2MPLS Push/Pop	MPLS SER (MPLS L2 tunnel end points)
IP Payload Matching and Set Field	L4-L7 Forwarding (e.g. TCP flags, GTP TEID) and Cybersecurity
BFD Link Monitoring	Liveness mechanism
UPD Payload Matching and Set Field	L7 Forwarding, Load Balancing (e.g. GTP TEID) and Cybersecurity
VxLAN Push/Pop	DC TOR and Cybersecurity
L2GRE Push/Pop	DC TOR and Cybersecurity
STT Push/Pop	DC TOR and Cybersecurity
Hashing on a List of Fields	Load balancing and Cybersecurity
Symetric Hashing on a List of Fields	Load balancing and Cybersecurity
Swap Values between Fields	Load balancing and Cybersecurity
Hardware Generated Time stamp	Network Performance Monitoring
Software Generated Time stamp	Network Performance Monitoring
Packet trimming	Network Performance Monitoring and Cybersecurity
INT Telemetry Report	Network and VNF (incl. Cybersecurity tools) Performance Monitoring
Postcard Telemetry Report	Network Monitoring and Traffic Classification
GTP Push/pop	4G/5G User Plane Function (UPF)
Set H-QoS Class	Broadband access (BNG) and 4G/5G User Plane Function (UPF)
PPPoE Header Matching and Handling	Broadband access (BNG)
L2TP Header Matching and Handling	Broadband access (BNG)

P4 and P4Runtime

- P4 is a programming language used to define how a switch silicon processes packets
  - Programmable parser (match fields)
  - Programmable actions
  - Programmable match-action pipeline

- P4Runtime is an interface between a P4 Controller and a P4 programmable switch:
  - Similar role to OpenFlow
  - Load a compiled P4 program into the switch silicon
  - Add/delete flow entries in the match-action tables
  - Collect statistics from the switch



Source: P4Runtime Specification

P4 Language

- Header types: defines the headers
- Parsers: defines how to parse a packet
  - User defines the protocol
- Actions: describes how a packet is manipulated
- Metadata: data structures associated with each packet as it traverses the pipeline
- Tables: what to match on and what the actions are
- Extern objects: Architecture specific constructs with well defined APIs
  - Checksum calculation
  - Registers
  - Counters
  - Meters

```
header Ethernet_h {
      bit<48> dstAddr;
      bit<48> srcAddr;
      bit<16> etherType
}
```

```
header IPv4_h {
      bit<4> version;
      bit<4> ihl;
      bit<8> diffserv;
      bit<16> totalLen;
      bit<16> identification;
      bit<3> flags;
      bit<13> fragOffset;
      bit<8> ttl;
      bit<8> protocol;
      bit<16> hdrChecksum;
      bit<32> srcAddr;
      bit<32> dstAddr;
      varbit<320> options;
}
```

Source: P4.org

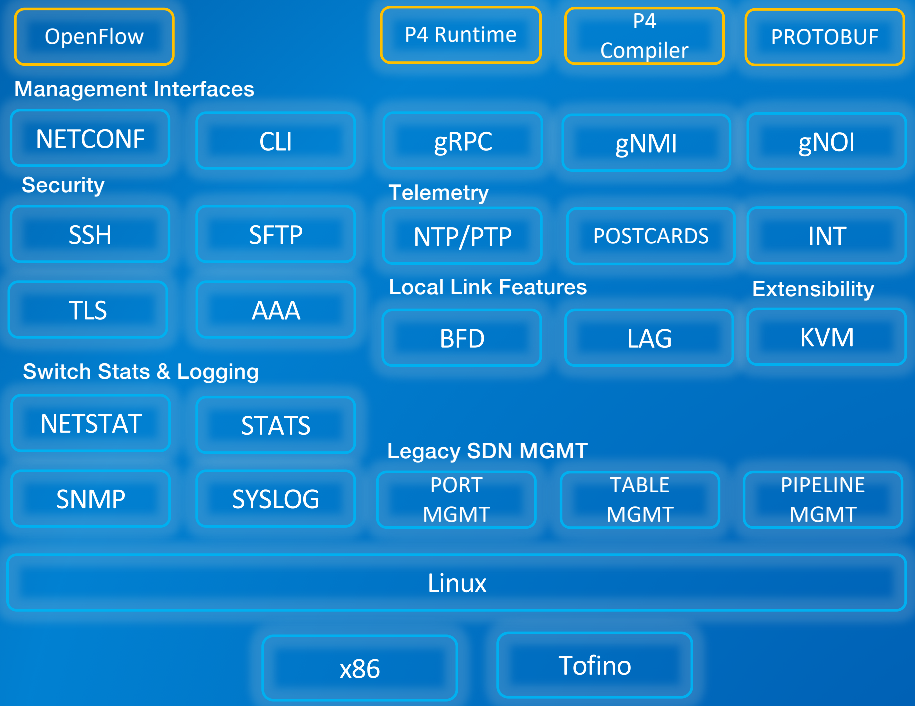
```
table ipv4_lpm {
      key = {
         hdr.ipv4.dstAddr: lpm;
      }
      actions = {
         ipv4_forward;
         drop;
         noAction;
       }
      size = 1024;
      default_action =
NoAction();
}
```

Comparisons between OpenFlow and P4/P4Runtime

- Both OpenFlow and P4/P4Runtime provides the application programmer with a programmable match-action pipeline
- Additionally, P4/P4Runtime allows the application programmer to program the parser
  - P4 is protocol independent
  - In OpenFlow, the match fields are predefined (Note: see Experimenter Extensions below)
- Additionally, P4/P4Runtime allows the application programmer to define the actions
  - In OpenFlow, the instructions/actions are predefined (Note: see Experimenter Extensions below)
- OpenFlow supports Experimenter Extensions where the developer can define new match fields, actions and more
  - Used by many companies, including NoviFlow

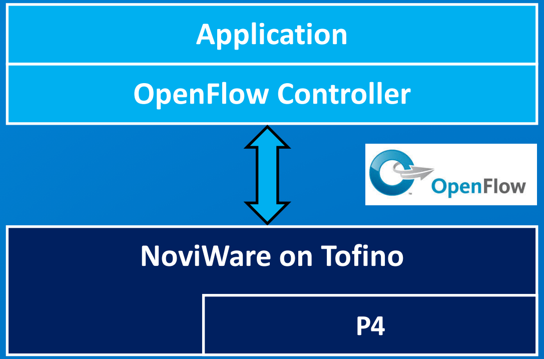
Components of a Deployable SDN NOS

- The OpenFlow/P4Runtime interface is only one component of what is needed in an SDN NOS
  - Configuration Management
  - Operations Management
  - Security Management
  - Extensibility
  - Telemetry



NoviFlow’s Phased Approach Towards P4/P4Runtime

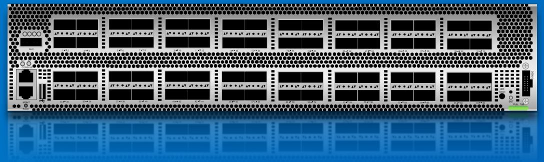
- Phase 1: OpenFlow over P4
  - Released in March 2018
  - Wraps OpenFlow around P4
  - OpenFlow 1.4 support
    - Match fields: 40 out of 41 (8 out of 13 Novi_Exp)
    - Instructions: 5 out of 6 (Clear actions missing)
    - Actions: 67 (including Novi_Exp)
  - Allows customers to create their own OpenFlow pipelines
    - Up to 9 tables
    - Any match field, instruction or action in any table
    - Up to 300k flows
    - 3.2/6.4Tbps Tofino white boxes
- Customers can integrate Tofino white boxes to existing OpenFlow applications


Application

OpenFlow Controller

NoviWare on Tofino

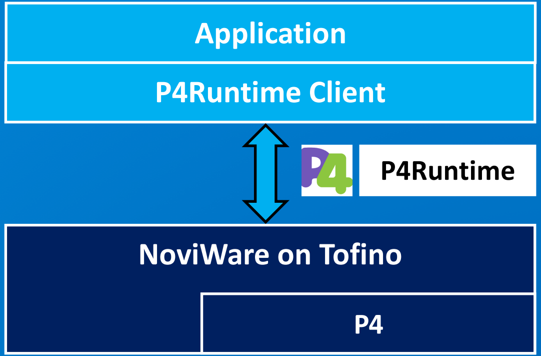
P4

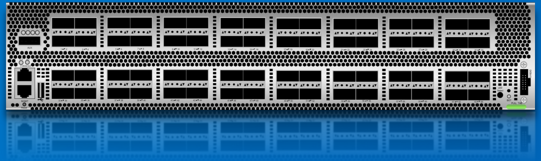


Speed Programmability Cost

NoviFlow's Phased Approach Towards P4/P4Runtime

- Phase 2: Native P4 in NoviWare
  - Released in June 2019
  - Uses P4Runtime between controller and switch
    - Ingest P4 program
    - Configure pipeline
    - Add/modify/delete flows
    - Get counters
    - Support externs
  - OpenFlow or P4Runtime mode can be set at start-up of the white box switch
  - Maintains control and data plane separation and allows customers to migrate existing applications on OpenFlow controllers and then migrate to P4Runtime





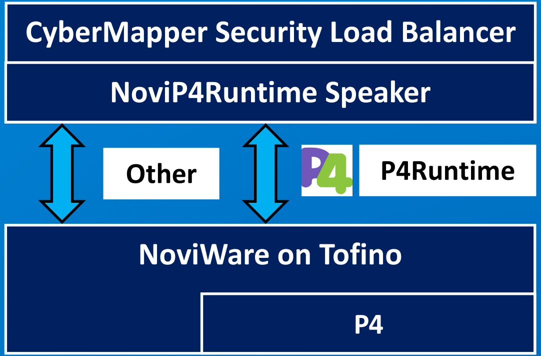
Speed

Programmability

Cost

NoviFlow’s Phased Approach Towards P4/P4Runtime

- Phase 3: CyberMapper on P4Runtime
  - Planned for September 2019
  - NoviP4Runtime Speaker
    - Push compiled P4 applications
    - Configure pipeline
    - Set/get flow entries (add/delete/modify)
    - Get flow stats
  - CyberMapper Security Load Balancer
    - Refactoring an existing OpenFlow application



CyberMapper Security Load Balancer

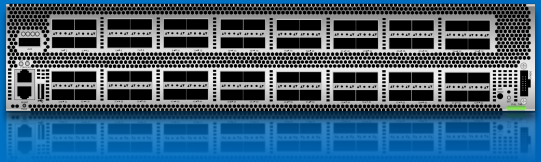
NoviP4Runtime Speaker

Other

P4Runtime

NoviWare on Tofino

P4



Speed

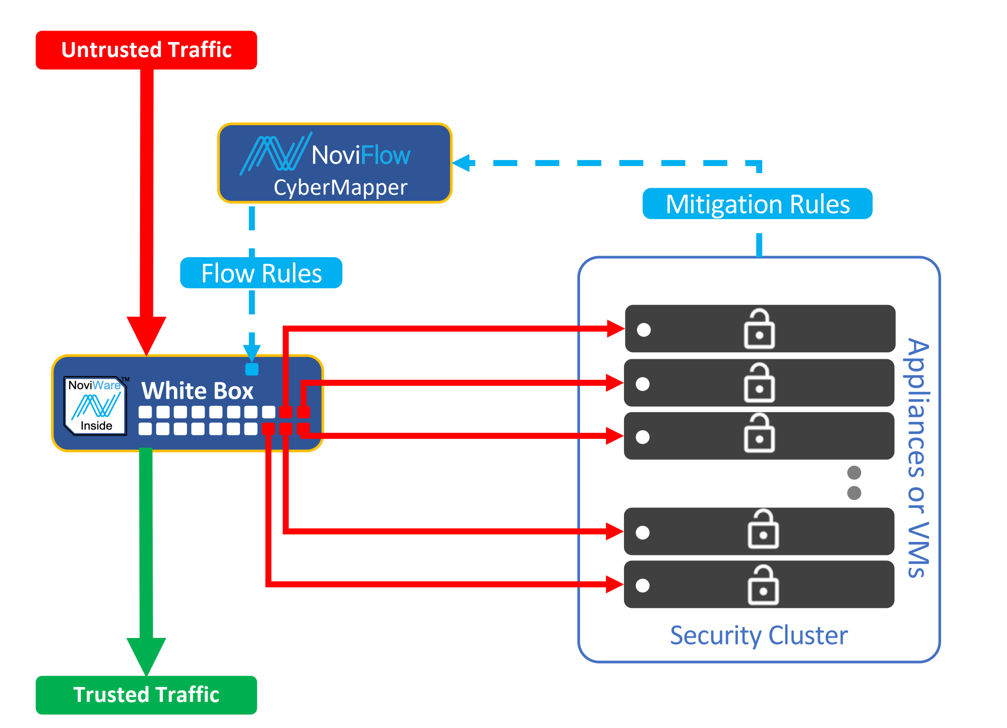
Programmability

Cost

NoviFlow *CyberMapper™* Security Load Balancer

Features and Benefits

- 6.4Tbps load balancer for virtual and physical security services
  - Scales security services into Tbps range
- Dynamic security pool scaling
  - Add/remove members of the pool with minimal impact on state
- Tofino white box switches
  - Low cost hardware
- Accelerates the performance of security services by off-loading whitelists and blacklists
  - Less expensive to do things in Tofino silicon than on x86
- Latency and Throughput Visibility
  - Monitor health of security cluster



CyberMapper™
Analytics and Management

NoviFlow
CyberMapper

- Latency measurements of virtual appliances
- Management of the switch
- Visualize network statistics
- Syslog events from switches

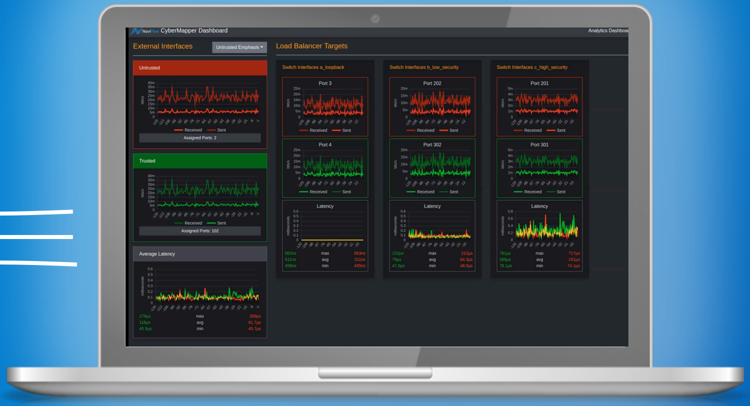
Networking

- Remotely enable/disable/restart subsystems
- Correlate SEL log entries with multiple events

Management

- Collect Hardware sensor information
- Collect syslog events

Compute



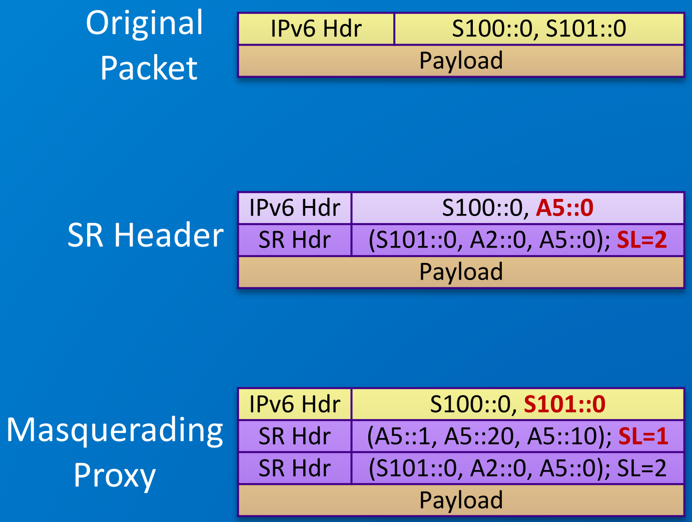
CyberMapper™

Refactored from OpenFlow to P4Runtime

- **More efficient use of switch silicon resources**
  - The P4 program defining the CyberMapper match-action pipeline is optimized for the functionalities needed for the CyberMapper application
    - Reduced computing overhead compared to the Phase 1 OpenFlow implementation which supports full OpenFlow functional specifications

- **Faster time to market for new features**
  - Not slowed down by having to reprogram the entire OpenFlow stack for new functionality outside the OpenFlow specification
    - Example: Implementing SRv6 for service chaining or 5G SRGW Gateway in OpenFlow would require new experimenter match fields and actions

- **More freedom in defining new match-action pipeline functionality**
  - Not constrained by the OpenFlow-only paradigm
    - Example: L2 learning, ARP learning



Original Packet

IPv6 Hdr	S100::0, S101::0
Payload	

SR Header

IPv6 Hdr	S100::0, **A5::0**
SR Hdr	(S101::0, A2::0, A5::0); **SL=2**
Payload	

Masquerading Proxy

IPv6 Hdr	S100::0, **S101::0**
SR Hdr	(A5::1, A5::20, A5::10); **SL=1**
SR Hdr	(S101::0, A2::0, A5::0); SL=2
Payload	

P4 and P4Runtime defines the next generation of Match-Action Pipelines

ONF CONNECT

Thank You

Follow Up Links:

NoviFlow.com