Deploying P4 Applications in Server-Based Networks

Abhijeet Prabhune, Bapi Vinnakota (Netronome)
Tu Dang, Fernando Pedone, Robert Soule (USI)
• New complex use cases drive server-based networking requirements
  • Virtualization, NFV, Security, SDN, Analytics, Streaming

• P4 used in diverse applications for server-based networking
  • P4 applications on Netronome SmartNICs:
    • Connection authentication, stateful security appliance, flow aggregation, telemetry, consensus as a service, and others in development

• Modern data center workloads are elastic
  • Networking requirements change as rapidly as the workloads

• Need to easily manage and operate P4 applications

To Scale: Need to manage server-based networking P4 apps like server software apps
### P4 Binary Management: Application-Specific

<table>
<thead>
<tr>
<th>Step</th>
<th>Netronome</th>
<th>Xilinx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Develop P4 App</td>
<td>Programmers’ Studio IDE</td>
<td>Vivado Design Suite</td>
</tr>
<tr>
<td>Compile into Binary</td>
<td>SDK 6.0 Compiler</td>
<td>Xilinx Vivado + SDNet</td>
</tr>
<tr>
<td>Load Executable</td>
<td>SDK Linker/Loader</td>
<td>Xilinx Vivado + SDNet</td>
</tr>
<tr>
<td><strong>Configure Application</strong></td>
<td>Rule updates through server</td>
<td>Application-specific</td>
</tr>
<tr>
<td><strong>Launch Application</strong></td>
<td>Control application through server</td>
<td></td>
</tr>
</tbody>
</table>

Experience with Paxos on Netronome SmartnNIC and Xilinx NetFPGA

P4-based SmartNIC applications can be managed like other server applications.
A “Bare-Metal” SmartNIC model can simplify function management
Create Data Plane Function Packages

- A standard binary package format
- Combine Function app and Function binary components
- Should only depend on Bare metal driver and Bare metal binary
- Common installation and configuration locations

Leverage package management infrastructure

- Package downloads
- Install process
ICONICS: Simplifying SmartNIC Management

Call For Action: Join to define and refine ICONICS

- Draft specification submitted to the Open Compute Project
  - Accessible @ https://iconics.io

Repository of Data Plane Function Packages

Host

Data Plane Function Package (DPFP)

Bare Metal SmartNIC FW Loader

Function Driver

Bare Metal SmartNIC Driver

SmartNIC Hardware

Bare Metal Binary

Functional Binary

Programmable cores
**ICONICS Beta for P4-based Applications**

- SmartNIC hardware installed on server
- P4 binary + app installed
- P4 Data Plane Function Package (DPFP) installed

**Component** | **Binary Functionality** | **Server App**
--- | --- | ---
Bare metal SmartNIC | Host loopback | Open-source kernel driver
Functional SmartNIC for P4 binaries | Simple wire pass through. Host interface Network interface | P4 firmware loader Real-time Thrift interface for match-action tables
P4 Dataplane Function Package | Application-specific binary (vendor provides tools) | Application-specific creation and operation (developer specifies)

Simple P4 application management
P4 Tool Chain Components

SDK-6 - Integrated Development Environment (IDE)

- Editor with language highlight and breakpoint support
  - Simple_Router.p4
  - Packet_filter.c
- P4 front end compiler
  - Simple_Router.IR
- P4 back end compiler
  - Simple_Router.C
- Assembler (NFAS)
- Compiler
- Linker
- C scripting
- Loader
- Hardware Debugger
  - Connect through Host
- Simulator
P4 DPFP Build Step 1: Follow P4 Development Work Flow

Native code compiler

Sandbox C

Stateful Filtering
Filter packets of fixed IP addresses
Filter the IP address with TCP ports, add VLAN tag

Stateful Statistics
Count a flow — with a fixed IP address
IPv4/6 statistics

Run time API generated by P4 compiler

Tabledata.JSON  Runtime I/F
Prototype:
- Combine firmware with table configuration
- Separate DPFP package per SmartNIC type
- Supported only on CentOS
- Operated through server management interface
- In-field configuration, outside scope

P4 DPFP builder prototype available. Contact Open-NFP for access.
Example: P4 Telemetry

P4 Telemetry DPFP
- **Binary**: Per hop telemetry functions
- **Host App**: Interface to collector
- **In-field**: Configuration to support target flow

**System Flow**
- In-Band Network Telemetry (INT) instruments every packet
- VXLAN GPE Encapsulation used to carry the metadata (other encapsulations are possible)
- Each hop adds:
  - Hop ID
  - Ingress Timestamp
  - Egress Timestamp

Telemetry Data Visualization (Track Path & Latency)
Example: P4/C Stateful Firewall

**P4/C Firewall DPFP**
- **Binary**: P4/C Match-action flow
- **Host App**: Firewall controller
- **In-field**: System configuration

**System Flow**
- Stateful Firewall + NAT
- P4 Tables, Actions and Rules
  - Lookup State Table
    - Match - Ingress Port
    - Action - Apply hash function for state lookup
  - Update State
    - Match - State, Egress Spec
    - Action - Update State
  - Controller Packet Table
    - Match - Ingress Port
    - Action - Clear Ports Timeouts

See the demo!
Example: PAXOS

Paxos DPFP
- **Binary**: Paxos data path
- **Host App**: Replicated key-value store
- **In-field**: Routing configuration identifying coordinators and acceptors

System flow
- **Coordinator**
  - Add a monotonically increasing sequence number of every packet
- **Acceptor**
  - Store history of messages
  - compare instance number in arriving proposals to instance number in history, route forward or drop based on the result
The Open-NFP P4 App Store

- [www.open-nfp.org](http://www.open-nfp.org)
  - Portal for research in data plane acceleration, 40+ organizations,
  - PS/SDK, Lots of P4 code at [https://github.com/open-nfpsw](https://github.com/open-nfpsw)

- Creating ICONICS P4 Beta repository
  - Netronome Agilio 40GbE SmartNIC, CentOS, Live soon
  - Bare metal SmartNIC installed as a tarball
  - Functional SmartNIC for P4, Telemetry, Stateful firewall, Load balancer, Consensus as a service
  - Aim to grow repository to include all apps on [https://github.com/open-nfpsw](https://github.com/open-nfpsw)
  - Support cross-platform packaging, naming convention and documentation guidelines

- Big Thank you
  - Pietro Bressana (USI), David George, Helgard van Rensburg, Rando Wiesemann (Netronome)

**Call For Action:** Use repository, create and/or contribute P4 binary packages.
Thank You