Protocol-Oblivious Forwarding
Towards OpenFlow 2.0
OpenFlow 1.X

A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks (Figure 2)
The Evolution of OpenFlow

- New Specifications at high rate
- Reactive Evolution
- Increased matching fields from 12 to 41
- Need for more matching fields for specific applications (e.g. Datacentre)

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Header Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>OF 1.0</td>
<td>Dec 2009</td>
<td>12 fields (Ethernet, TCP/IPv4)</td>
</tr>
<tr>
<td>OF 1.1</td>
<td>Feb 2011</td>
<td>15 fields (MPLS, inter-table metadata)</td>
</tr>
<tr>
<td>OF 1.2</td>
<td>Dec 2011</td>
<td>36 fields (ARP, ICMP, IPv6, etc.)</td>
</tr>
<tr>
<td>OF 1.3</td>
<td>Jun 2012</td>
<td>40 fields</td>
</tr>
<tr>
<td>OF 1.4</td>
<td>Oct 2013</td>
<td>41 fields</td>
</tr>
</tbody>
</table>

P4: Programming Protocol-Independent Packet Processors (Table 1)
The Problem with OpenFlow 1.X

- New OF Specifications at high rate
- Hardware development cycle rate much lower
- Most OF Switches still run OF v1.0
- OF table type pattern

New Service Required

OF Support?

Wait for new revision of OF Specification

Wait for device support

Deploy the new service
Towards OpenFlow 2.0

• New Services/Applications
  – Demand for new Matching Fields
  – Increasing Complexity of OF Instruction Set

• Trend to Programmable Switches
  – CPU/Software
  – Network Processors
  – FPGA
  – Flexible Match+Action ASICs
Protocol Independent Forwarding

- Protocol Independence
- Target Independence
- Reconfigurability
- Language Independence

“We believe that future generations of OpenFlow should allow the controller to tell the switch how to operate rather than be constrained by a fixed switch design” – Nick McKeown
PIF - Initiatives

- P4 – Programming Protocol-Independent Packet Processors

- POF – Protocol-Oblivious Forwarding

- ONF PIF – OpenSource Project
  [https://www.opennetworking.org/protocol-independent-forwarding](https://www.opennetworking.org/protocol-independent-forwarding)
Protocol-Oblivious Forwarding

Network Device Programming Models

- **Closed and proprietary design that meets the predefined service requirements**
  - Device Vendor-driven Programmability
  - "Black Box" Data Plane

- **Open interface using existing technology building blocks for limited programmability**
  - Device Vendor/Service Provider Split Programmability
  - "Gray Box" Data Plane

- **Open interface using generic network programming model, enabling full programmability**
  - Service Provider-driven Programmability
  - "White Box" Data Plane

Management -> Control Plane -> OpenFlow 1.x

Control Plane

Program -> OpenFlow+

Future-Proof Forwarding Plane – SIGCOMM 2013 HotSDN Presentation
PC Analogy

- Standard instruction set
- Intermediate level abstracts the hardware

Protocol Oblivious Forwarding: Unleash the Power of SDN through a Future-Proof Forwarding Plane (Figure 1)
POF Framework

- Flow Instruction Set
  - Parsing, Editing, Forwarding
- Flow Instructions Blocks
  - Flow Handling
- Several Northbound Interfaces possible

Forwarding in Protocol-Oblivious Instruction Set (Figure 1)
POF – Flow Instruction Set

- Concise set of instructions
- Optimized for networking applications
- Platform independent
- Protocol agnostic

<table>
<thead>
<tr>
<th>Category</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDITING</td>
<td>SET_FIELD, ADD_FIELD, DEL_FIELD, ALG, CALCULATE_CHECKSUM,</td>
</tr>
<tr>
<td></td>
<td>SET_FIELD_UPDATE_CHECKSUM, INC_FIELD, DEC_FIELD, OR_FIELD, SRL_FIELD,</td>
</tr>
<tr>
<td></td>
<td>SLL_FIELD, AND_FIELD, XOR_FIELD, NOR_FIELD, NOT_FIELD</td>
</tr>
<tr>
<td>FORWARDING</td>
<td>GOTO_TABLE, COUNTER, OUTPUT, GROUP, MOVE_PACKET_OFFSET, SET_PACKET_OFFSET</td>
</tr>
<tr>
<td>ENTRY</td>
<td>SET_TABLE_ENTRY, ADD_TABLE_ENTRY, DEL_TABLE_ENTRY</td>
</tr>
<tr>
<td>JUMP</td>
<td>BRANCH, COMPARE, JUMP</td>
</tr>
<tr>
<td>FLOW</td>
<td>SET_FLOW_METADATA, GET_FLOW_METADATA, ORDER_ENFORCE</td>
</tr>
</tbody>
</table>

Forwarding in Protocol-Oblivious Instruction Set (Table 1)
POF – Flow Instruction Set

• Flow metadata
• Flow table manipulation by data-path
• Reuse of instruction blocks
• Statistic counters are shared resources
• Categorization of lookup tables
POF – Instruction Set

Is this an **IP** packet?

- Use **destination IP** to search FIB
- Process **TTL**
- Update **Checksum**

Conventional FE Forwarding Flow

Data **@{12B, 2B} = 0x0800?**

- Extract Data **@{30B, 4B};**
  - Use it as key to search LPM table x
- Decrement Data **@{22B, 1B};**
  - If result is 0, drop the packet
- Use algorithm y to calculate checksum over Data **@{14B, 20B};**
  - Write result **@{24B, 2B}**

POF Forwarding
Characteristics of POF-FIS

- Flexibility
- Independence
  - Northbound Interface
  - Service and Application
- Completeness
  - Same functionality as OpenFlow
    Instead of hardcoded protocol fields, search key \{offset, length\}
Realization

• Controller based on Floodlight
• Two POF FE prototypes
  – Huawei NE5000 core router platform
  – Software-based FE
• Northbound-Interface
  – CLI
  – GUI
  – High-Level Language Description
• Forwarding throughput reduction of up to 30%
Successful Transition

• OpenFlow Compatibility
  – Incremental Deployment

• Forwarding Element Implementation
  – New chips support POF natively
  – Make existing chip support POF
“As long as the performance is satisfactory, service providers will never need to deal with vendors again after devices are deployed.”

“For instance, EDIT includes all packet editing related instructions we believe ever needed, and FORWARDING and JUMP cover all the common packet processing related instructions.”

“If one has sufficient understanding (...), he or she is able to design the whole forwarding process by directly assembling the POF flow instructions (...) without any compiler or interpreter.”