PISCES: A Programmable, Protocol-Independent Software Switch

Muhammad Shahbaz, Sean Choi, Ben Pfaff, Changhoon Kim, Nick Feamster, Nick McKeown, and Jennifer Rexford

Slide credit to Muhammad Shahbaz
Also appears at SIGCOMM 2016!

http://goo.gl/wmBmTu
Importance of Software Switches

OVS
Hypervisor

VM
VM

OVS
Hypervisor

VM
VM

OVS
Hypervisor

VM
VM

OVS
Hypervisor

VM
VM

ToR

Core
Importance of Software Switches

- OVS
- Hypervisor
- VM
- VM
- ToR
- Core
- OVS
- Hypervisor
- VM
- VM
- ToR
- Core
- OVS
- Hypervisor
- VM
- VM
- ToR
- Core
- OVS
- Hypervisor
- VM
- VM
- ToR
- Core
Ease of Customization?

Enable **Rapid Development and Deployment of Network Features!**

Is it REALLY the case?
Ease of Customization?

For example, OVS supports following tunneling protocols:

- **VXLAN**: Virtual Extensible LAN
- **STT**: Stateless Transport Tunneling
- **NVGRE**: Network Virtualization Generic Routing Encapsulation

What about adding new protocols?
Rapid Development & Deployment?
Rapid Development & Deployment?

Requires domain expertise in:

- Network protocol design
- Software development

Can take 3-6 months to release a new feature

Can even be harder to maintain

Arcane APIs
Rapid Development & Deployment?
Rapid Development & Deployment?

OVS

Parser  Match-Action Pipeline  DPDK
Rapid Development & Deployment?

Domain Specific Language (P4)

Parser  Match-Action Pipeline

Compile

OVS

Native OVS

14,535 lines of code

341 lines of code
Rapid Development & Deployment?

Domain Specific Language (P4)

Parser  Match-Action Pipeline

Compile

OVS

Parser  Match-Action Pipeline

DPDK

What is the performance overhead?
What is the **cost of programmability** on Performance?
PISCES: A Protocol-Independent Software Switch
PISCES: A Protocol-Independent Software Switch

- P4
- Compiler
  - parse
  - match
  - action
- OVS
- Runtime Flow Rules
- Flow Rule Checker
- Executable
- **Performance overhead** of a naïve mapping from P4 to OVS.

- **PISCES compiler optimizations** to reduce the performance overhead.
Naïve Mapping from P4 to OVS

A naïve compilation of L2L3-ACL benchmark application

Performance overhead of ~ 40%
Causes of Performance Degradation

Ingress → Packet Parser → Match-Action Pipeline → Packet Deparser → Egress

CPU Cycles per Packet
Causes of Performance Degradation

- Factors affecting CPU cycles:
  - Fully-specified checksum calculation
  - Redundant parsing of header fields
  - Many more ...
Causes of Performance Degradation

Factor #1: **Fully-Specified Checksums**

- **Checksum** (version, ihl, diffserv, totalLen, identification, flags, fragOffset, ttl, protocol, hdrChecksum, srcAddr, dstAddr)

- **Incremental-Checksum** (ttl)

Diagram:

- **Ingress** → **Packet Parser** → **decrement(ttl)** → **Packet Deparser** → **Egress**
Causes of Performance Degradation

Factor #2: Redundant parsing of headers
Optimized Mapping from P4 to OVS

All optimizations together

Performance overhead of < 2%
Another Cause for Performance Degradation

Packet Parser

Match-Action Cache

Packet Deparser

Optimizations
- Cached field modifications
- Stage assignment

Ingress

Egress

Cache Misses
Next Steps

- Support for **stateful memories** and **In-band Network Telemetry (INT)**

- Integration with the **mainline OVS**
Summary

With appropriate compiler optimizations ...

P4 + OVS == Fast Forwarding!
Questions?

Learn more and try PISCES here:
https://github.com/P4-vSwitch