# PISCES: A Programmable, Protocol-Independent Software Switch

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



**Fixed Set of Protocols**

TCP

Ethernet

IPv4

UDP

HTTP

IPv6

BGP

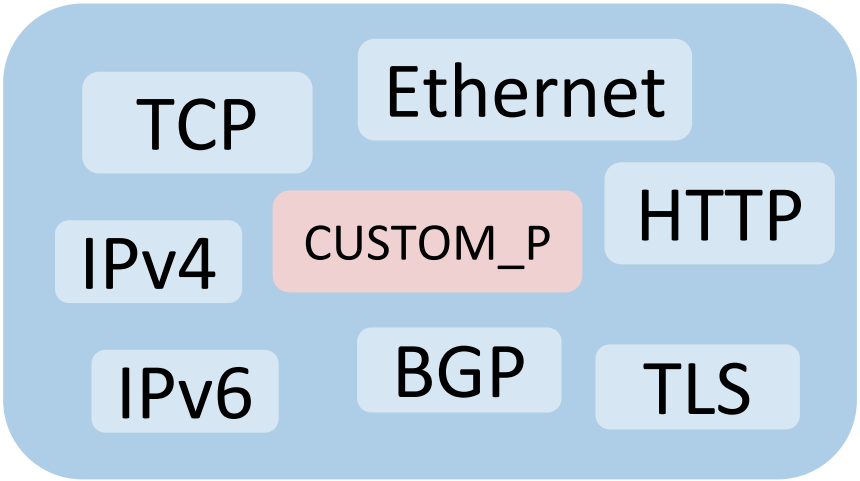
TLS

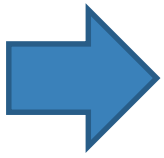
**Fixed-Function Switch Chip**

- Ease of **Adding** new protocols
- Ease of **Removing** unused protocols
- Gain greater **Visibility** into the network
- **Fold** network functions into the switch

**Custom Protocols**

**Programmable Switching Chip**

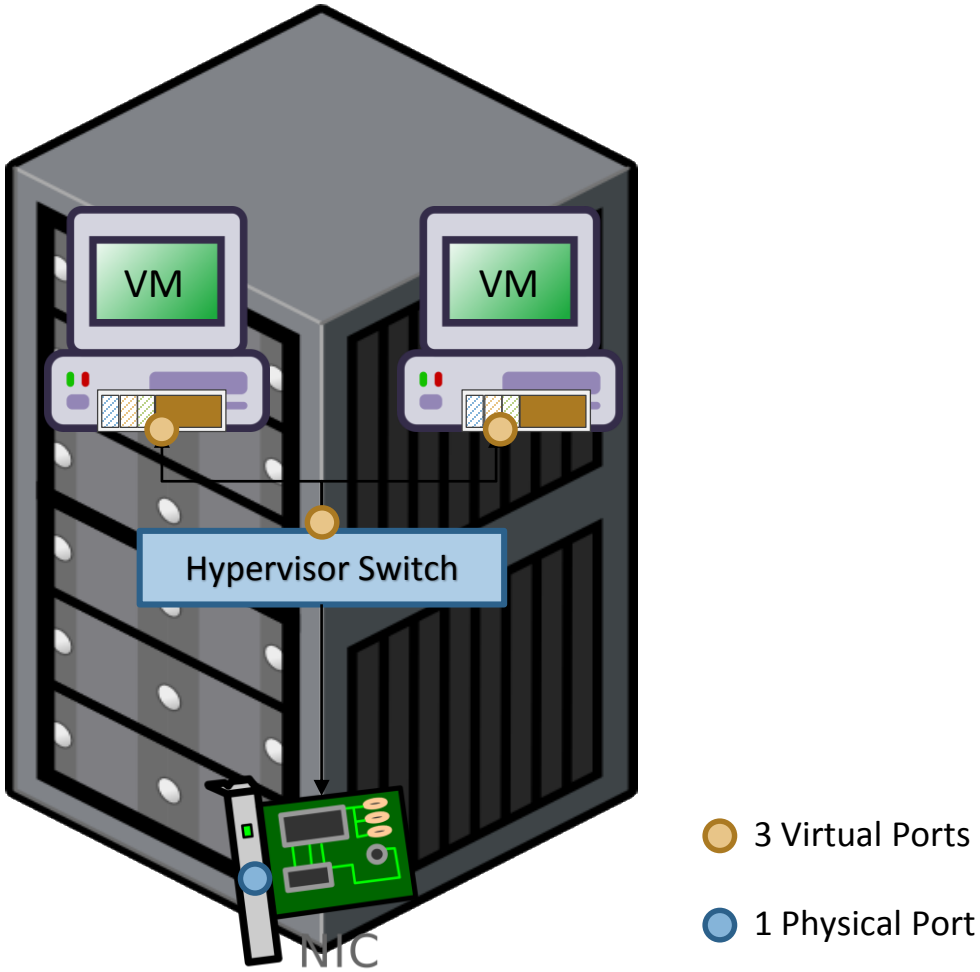


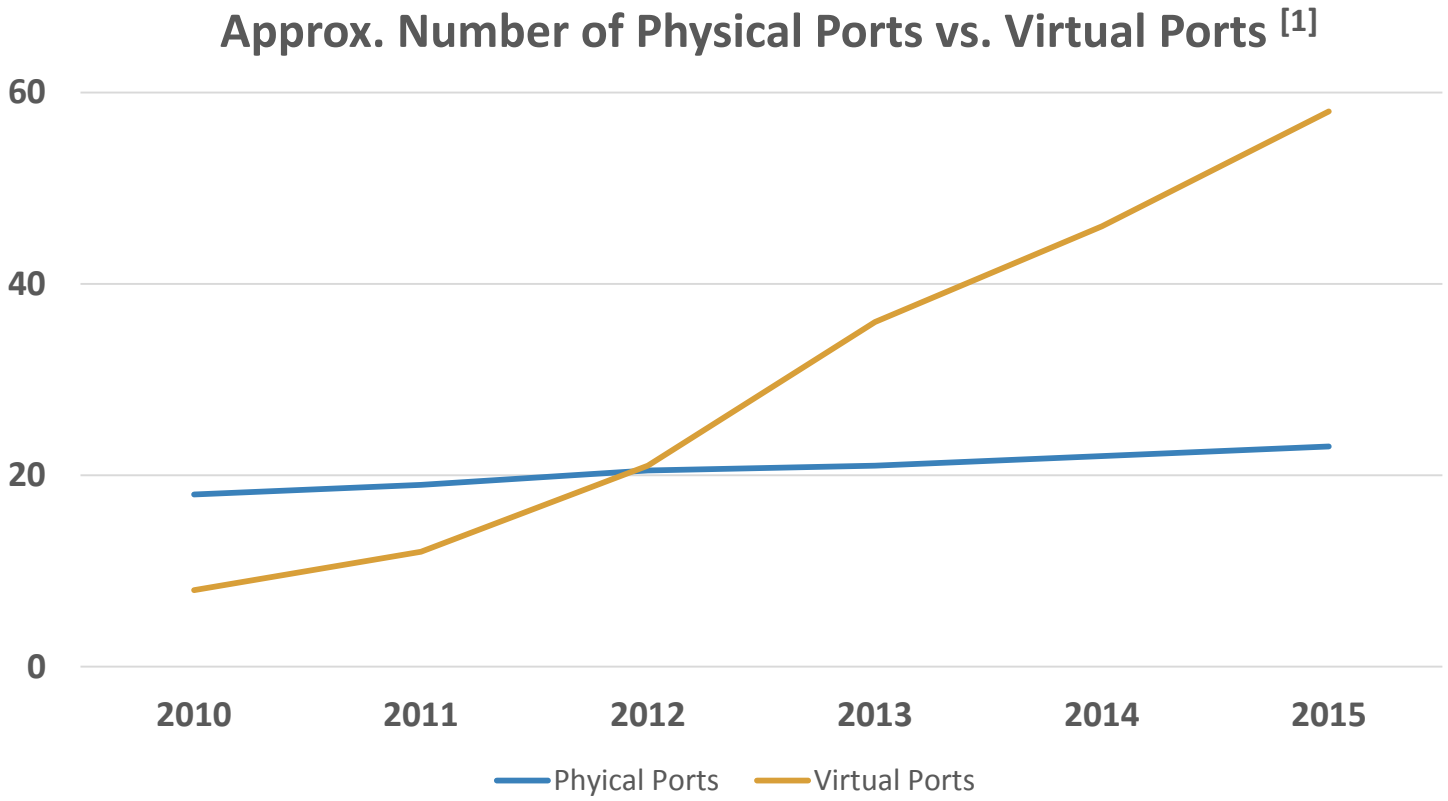



intel®

Xpliant

BAREFOOT NETWORKS

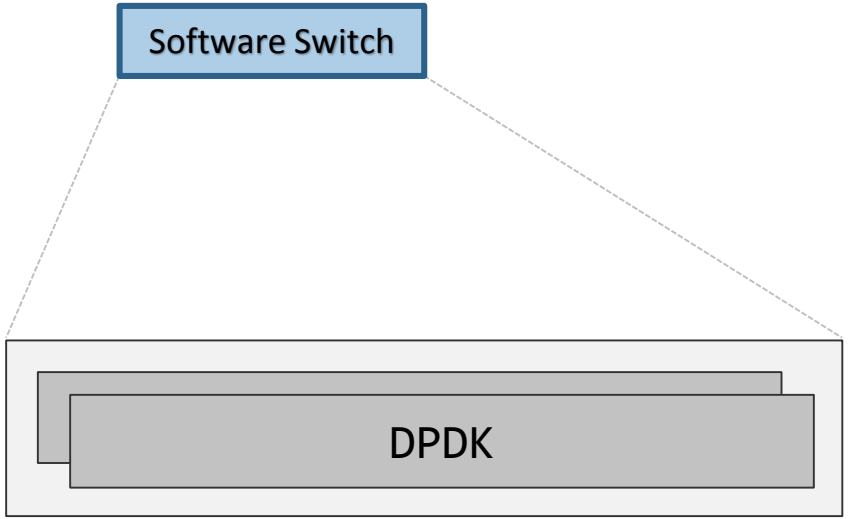

VM
VM
Hypervisor Switch
NIC
3 Virtual Ports
1 Physical Port

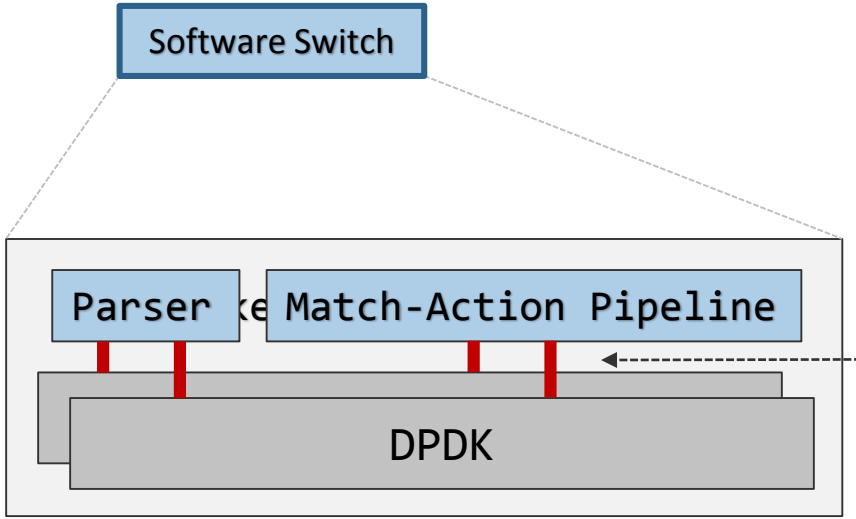



[1] Martin Casado, VMWorld 2013

It should be **EASY** to **program** software switches!

**Not really…**


Software Switch
DPDK

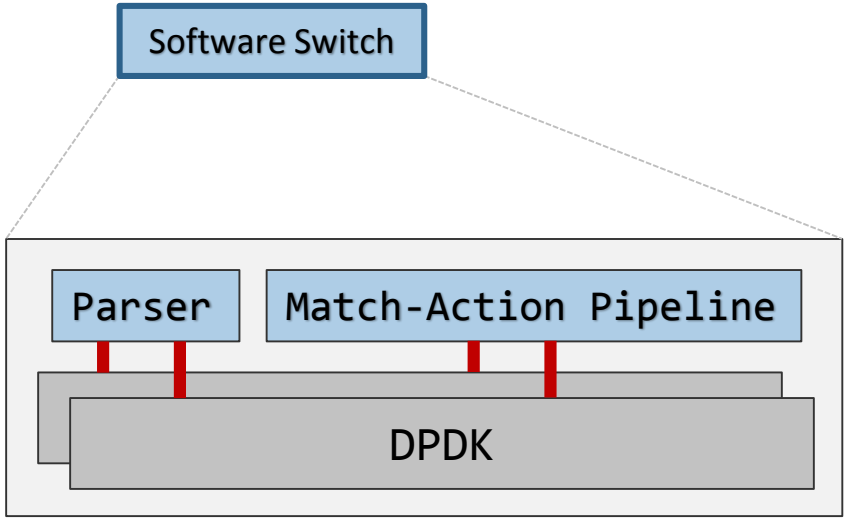



**Requires domain expertise in:**

- Network protocol design
- Kernel development

**Slow to release changes**

**Specialized APIs**




Adding **TCP Flag** in Open vSwitch required changes in…

**20 Files** and **370 Lines of Code!**[1]

**Weeks** of Development and Test

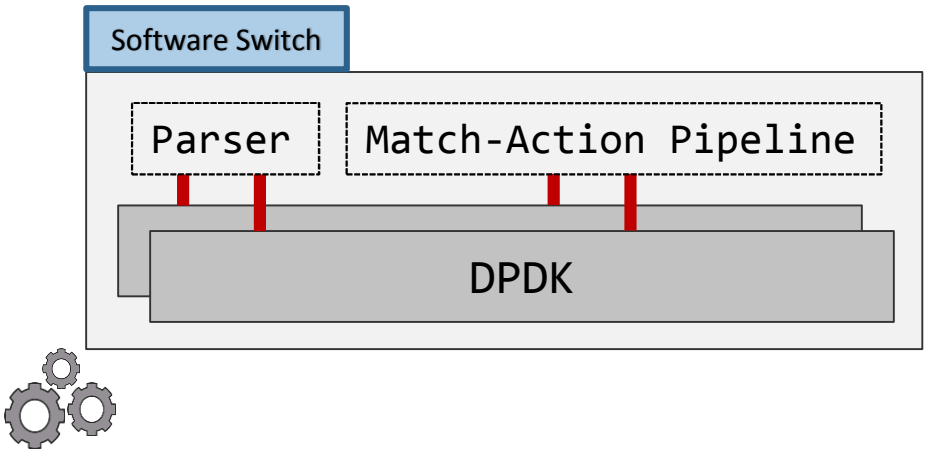
[1] https://github.com/openvswitch/ovs/commit/dc235f7fbcff

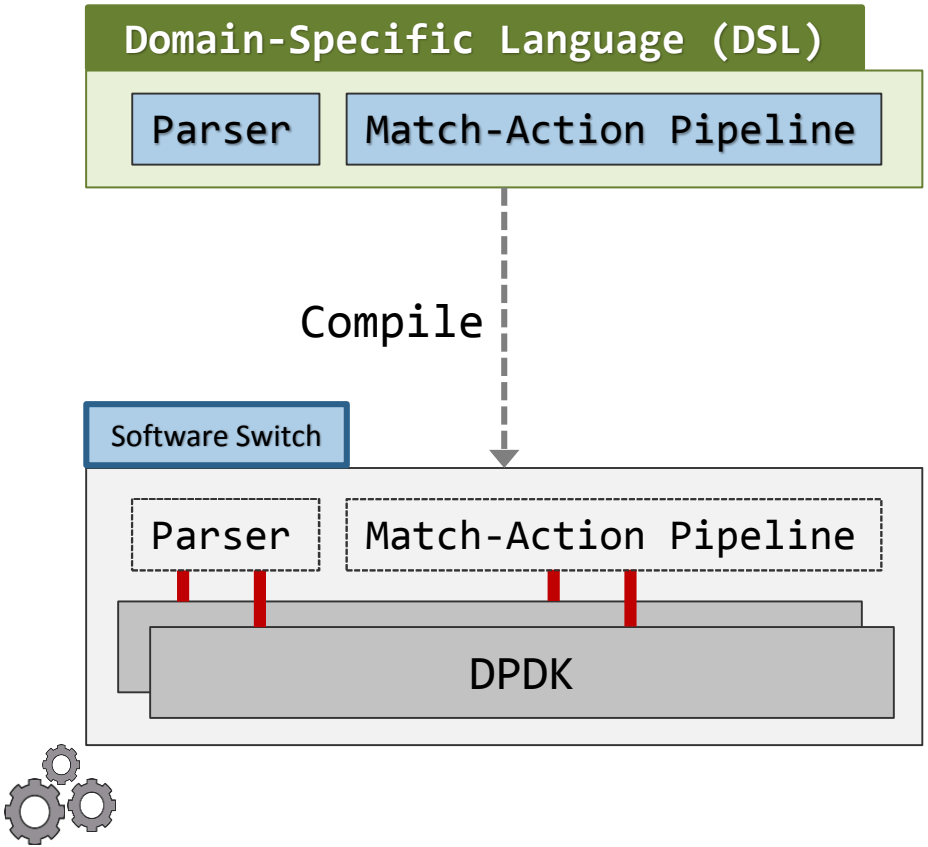
# We can do this in **4 lines** and **within minutes** with **PISCES!**

```
header_type tcpv2_t {
  fields {
    srcPort : 16;
    dstPort : 16;
    seqNo : 32;
    ackNo : 32;
    dataOffset : 4;
    res : 4;
    tcp_flags : 12;
    window : 16;
    checksum : 16;
    urgentPtr : 16;
  }
}
```

```
parser tcpv2 {
  extract(tcpv2);
  set_metadata(flow.tcp_flags,
               tcpv2.tcp_flags);
  return ingress;
}
```

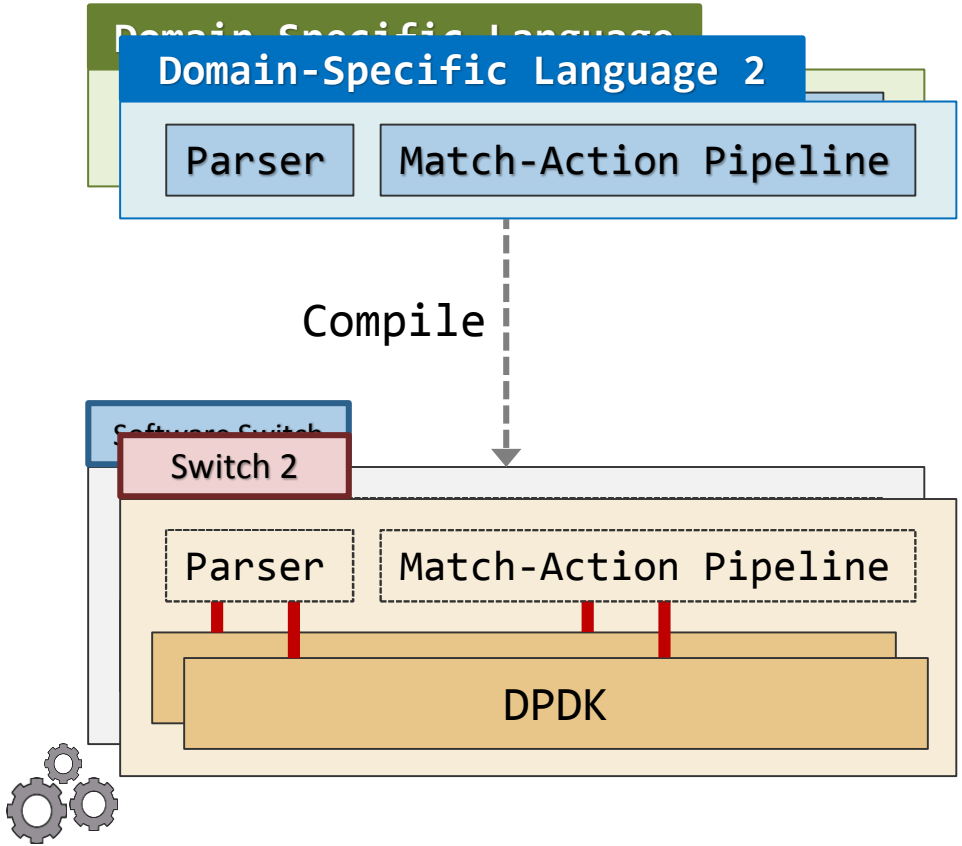
```
header_type flow_t {
  fields {
    ...
    tcp_flags_pad : 4;
    tcp_flags : 12;
    ...
  }
}
```


Software Switch
Parser
Match-Action Pipeline
DPDK



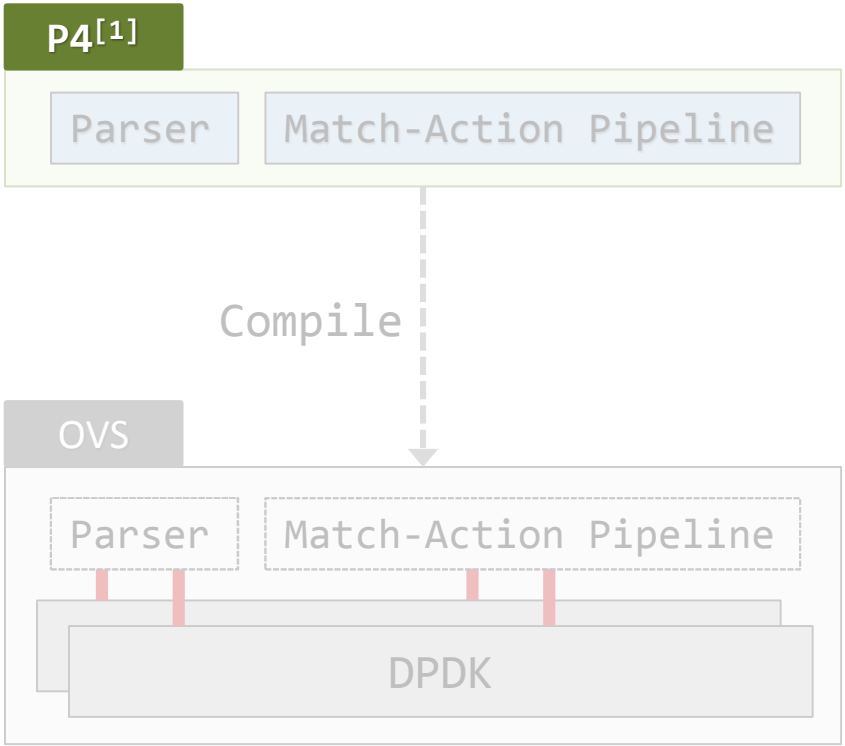

**TCP Header**

```
header_type tcp_t {
  fields {
    srcPort : 16;
    dstPort : 16;
    seqNo : 32;
    ackNo : 32;
    dataOffset : 4;
    res : 4;
    window : 16;
    checksum : 16;
    urgentPtr : 16;
  }
}
parser tcp {
  extract(tcp);
  return ingress;
}
...
```




**PISCES** is an **implementation** with a

- Specific Domain-Specific Language
- Specific Software Switch Target

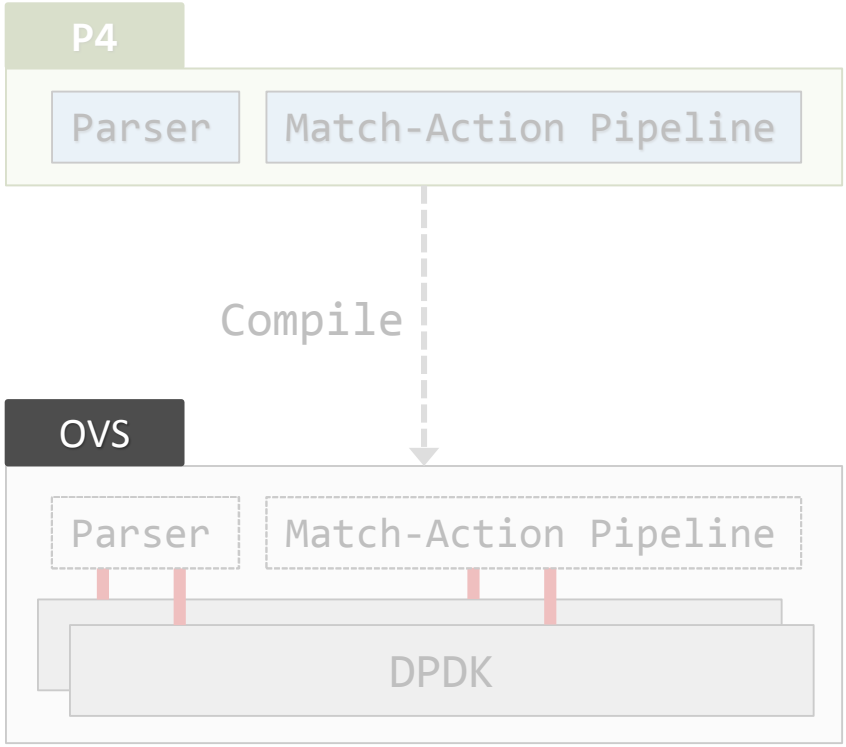



P4 is an **open-source language**.[1]

Easy to express different aspects of a packet processor:

- **Packet headers and fields**
- **Parser**
- **Actions**
- **Match-Action Tables**

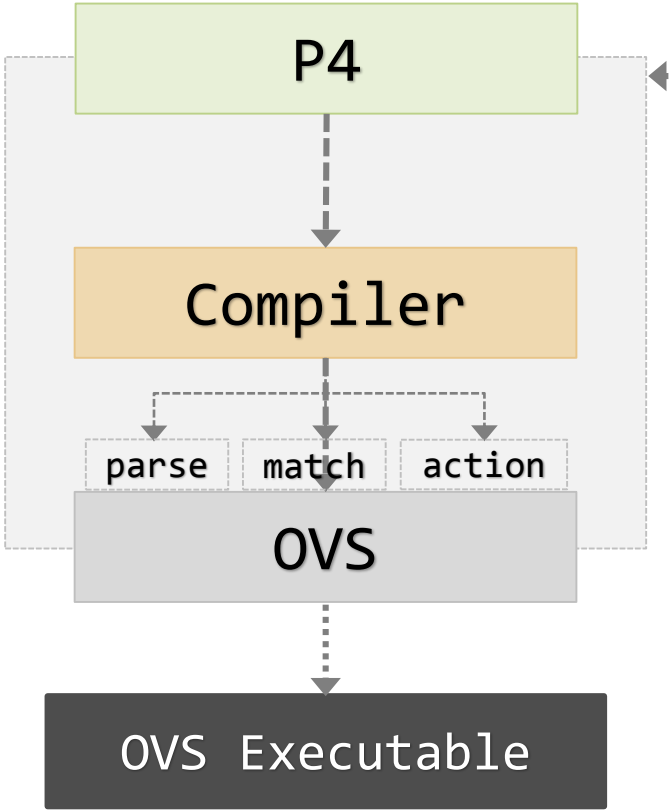
[1] http://p4.org



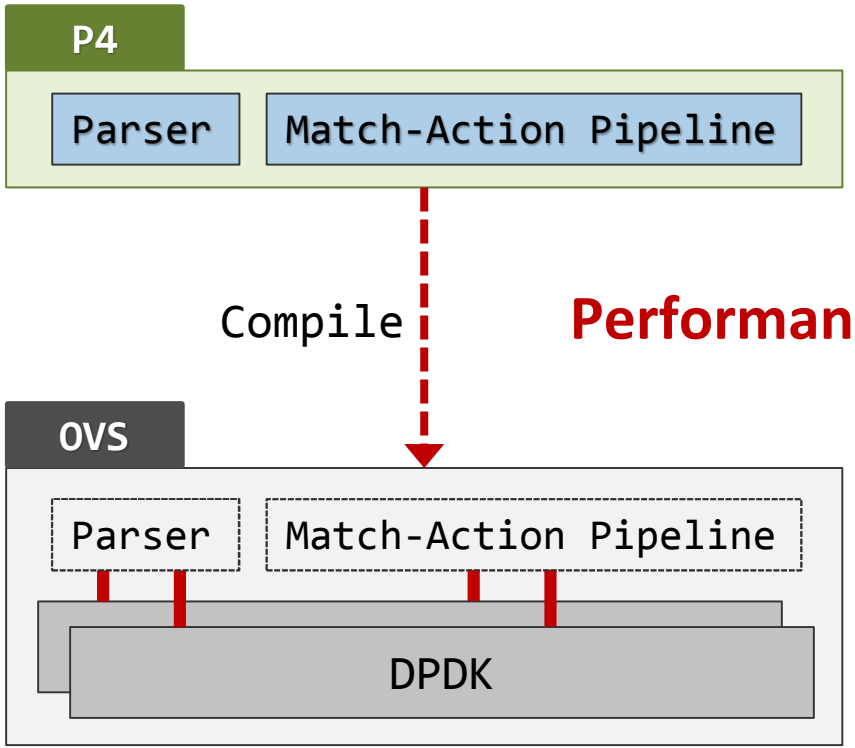

**341** lines of code

Native OVS
Packet Processing Logic

**14,535** lines of code

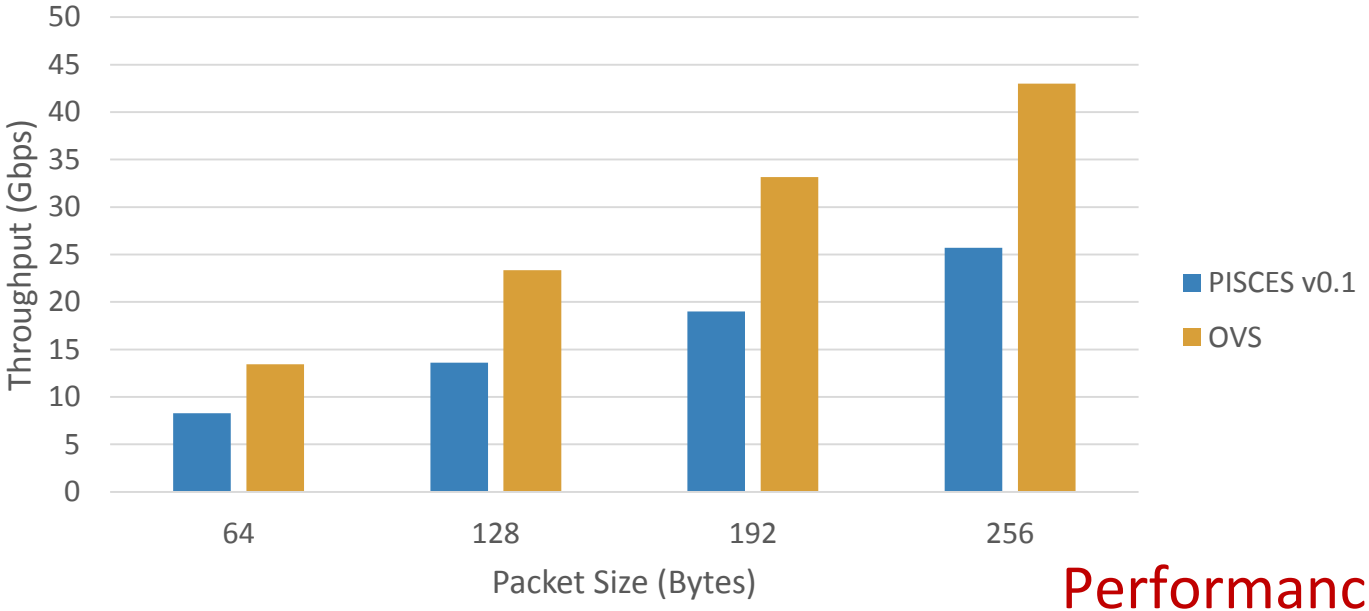



```
header_type tcp_t {
  header_type tcpv2_t {
    fields {
      srcPort : 16;
      dstPort : 16;
      seqNo : 32;
      ackNo : 32;
      dataOffset : 4;
      res : 4;
      tcp_flags : 8;
      window : 16;
      checksum : 16;
      urgentPtr : 16;
    }
  }
  parser tcpv2 {
    extract(tcpv2);
    set_metadata(flow.tcp_flags,
                 tcpv2.tcp_flags);
    return ingress;
  }
  ...
```




**Performance Overhead?**

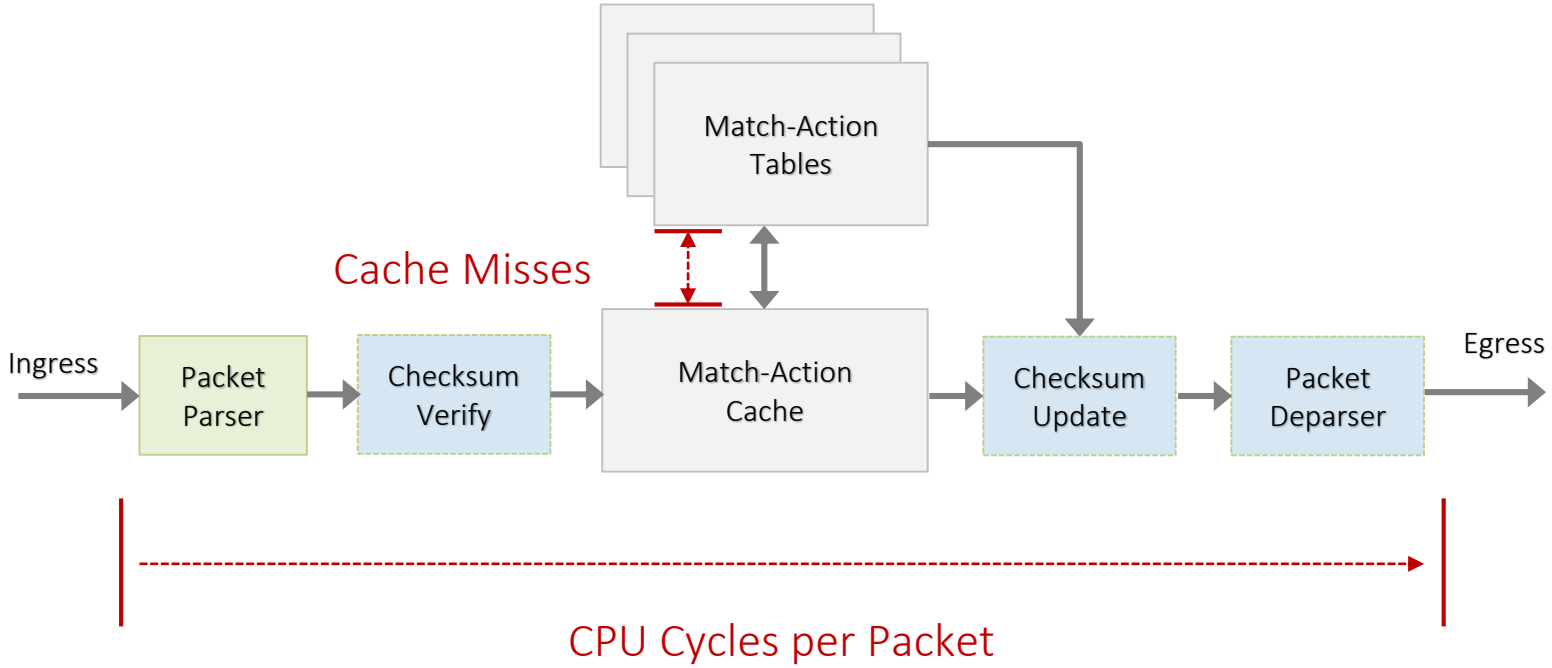
Throughput on **L2L3-ACL** benchmark application




Performance overhead of

**~40%**

# Causes for the Cost on Performance

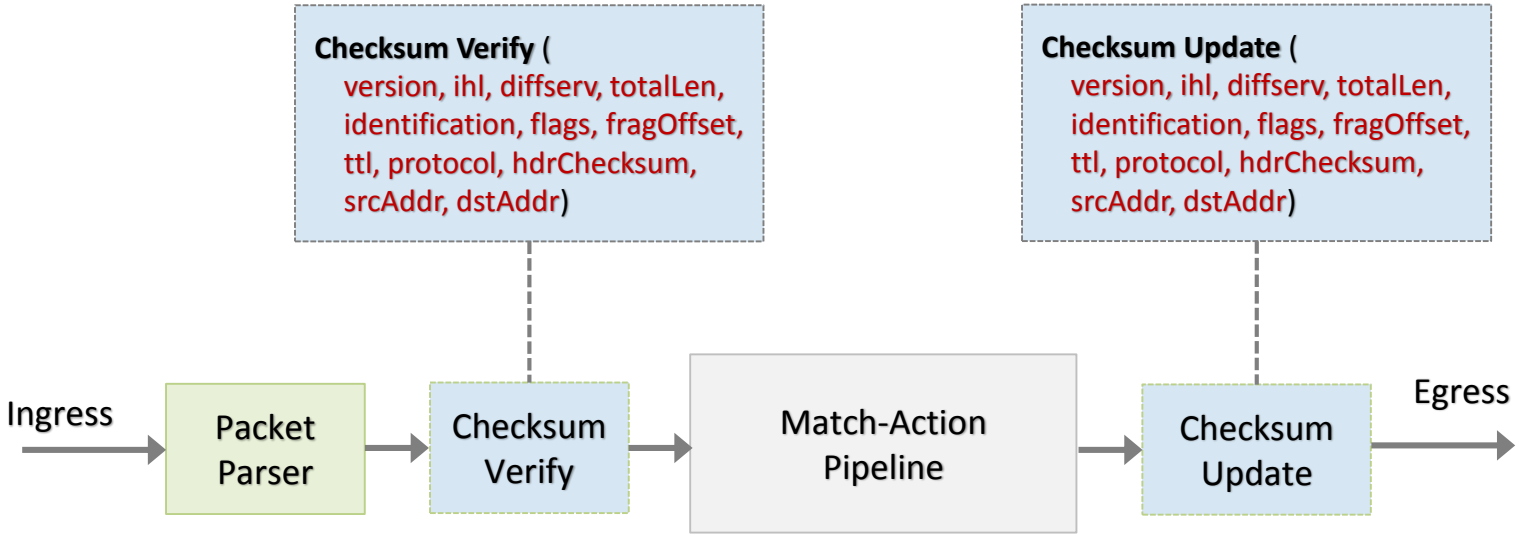


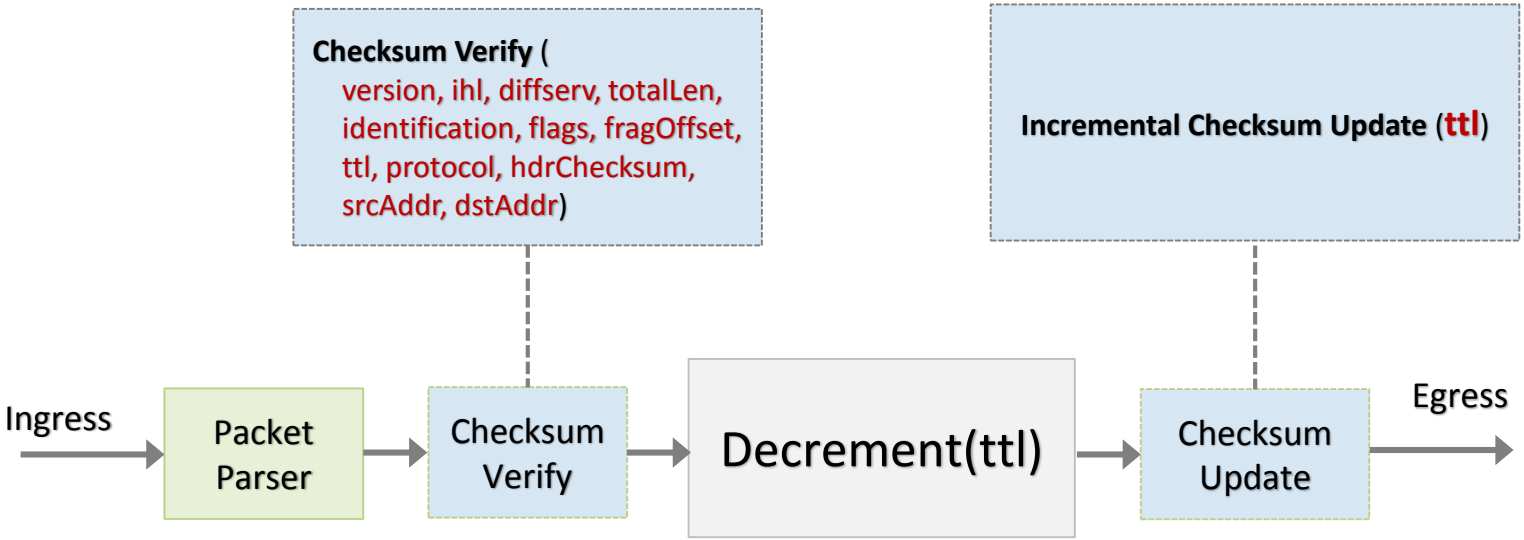
# Factors affecting CPU Cycles per Packet

a. Fully-specified Checksum

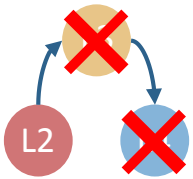
b. Parsing unused header fields

and more …

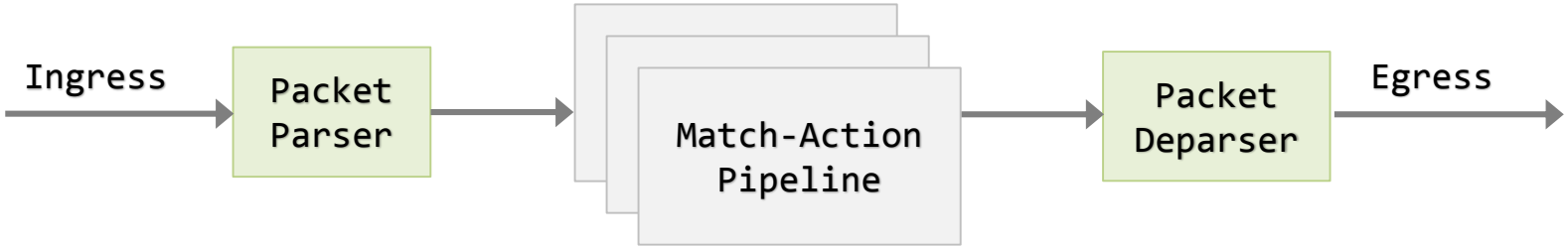

Checksum Verify (
version, ihl, diffserv, totalLen,
identification, flags, fragOffset,
ttl, protocol, hdrChecksum,
srcAddr, dstAddr)
Checksum Update (
version, ihl, diffserv, totalLen,
identification, flags, fragOffset,
ttl, protocol, hdrChecksum,
srcAddr, dstAddr)
Ingress
Packet Parser
Checksum Verify
Match-Action Pipeline
Checksum Update
Egress


Checksum Verify (
version, ihl, diffserv, totalLen,
identification, flags, fragOffset,
ttl, protocol, hdrChecksum,
srcAddr, dstAddr)
Incremental Checksum Update (ttl)
Ingress
Packet Parser
Checksum Verify
Decrement(ttl)
Checksum Update
Egress

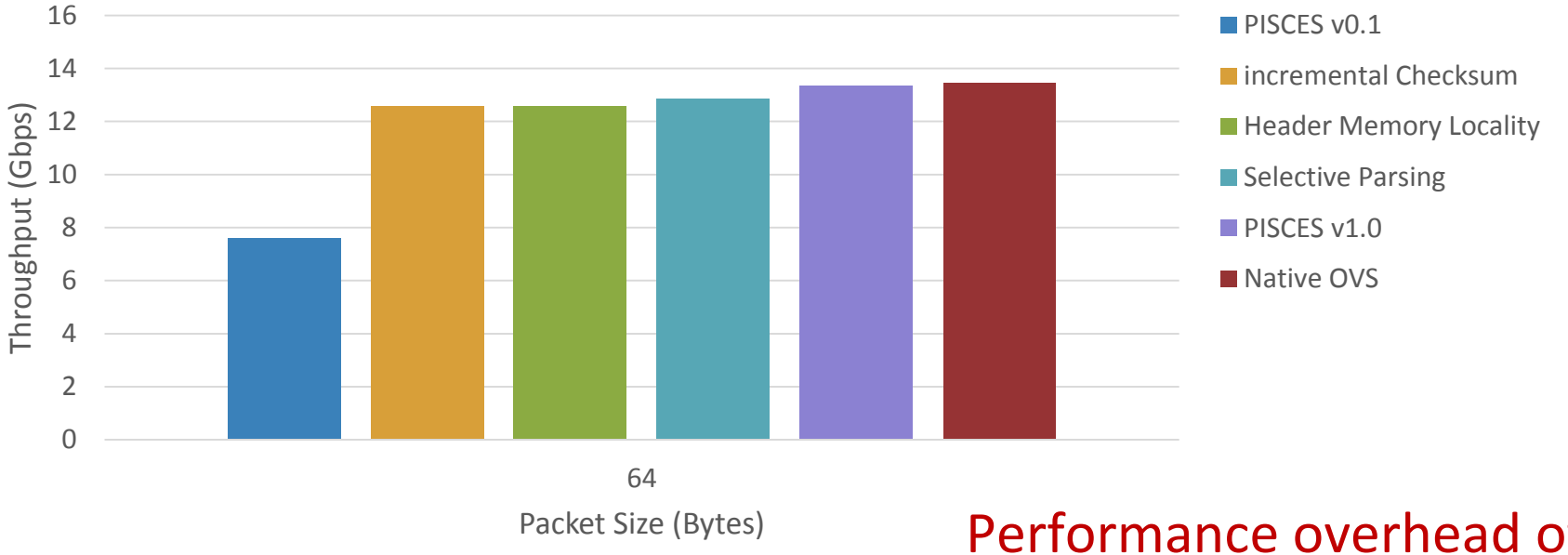
# Selective Parsing




L2



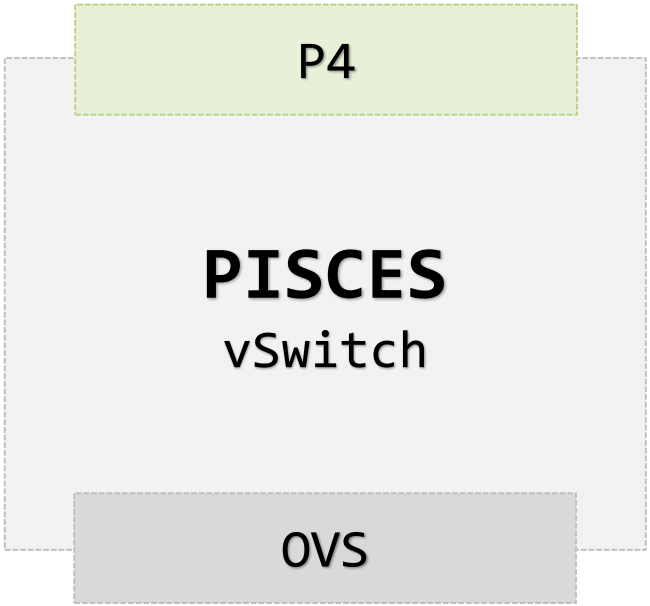
# Throughput on **L2L3-ACL** benchmark application




Performance overhead of

**< 2%**

# Summary




- **Quickly develop and deploy** new packet header format.
- With hardly **any performance cost!**

# Questions?

Learn more and **Try** PISCES here:

**http://pisces.cs.princeton.edu**