Hands-on Workshop on Open vSwitch and Software-defined Networking

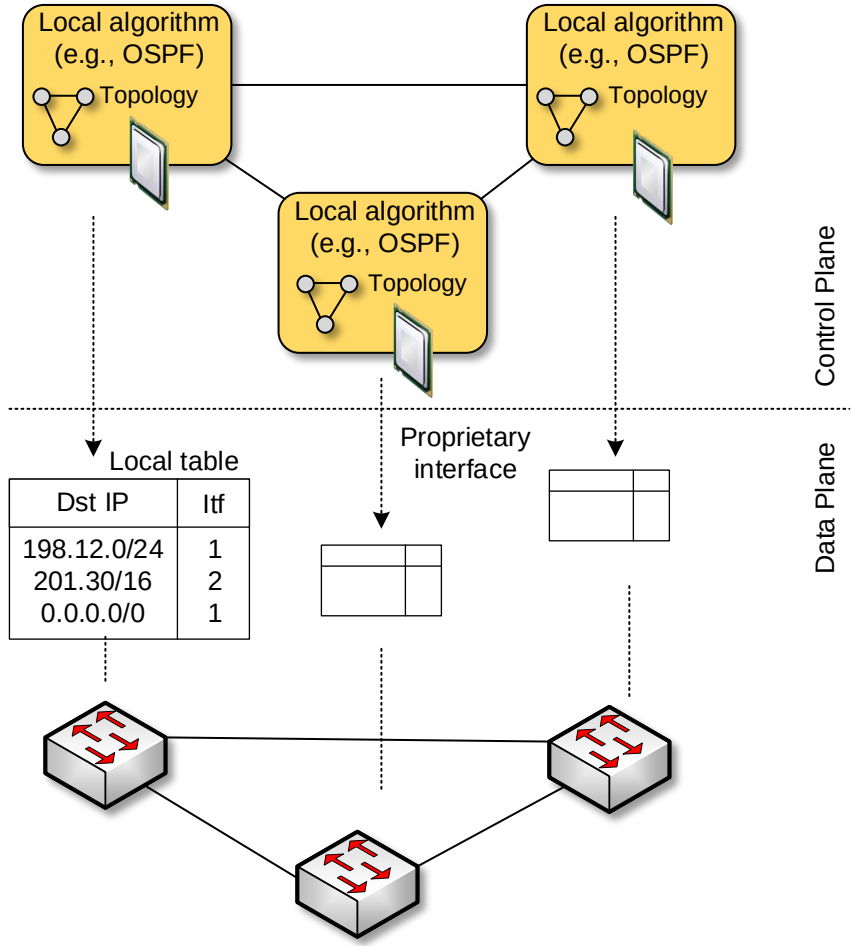
Jorge Crichigno, Elie Kfoury, Shahrin Sharif
University of South Carolina

Western Academy Support and Training Center (WASTC)
June 21-25, 2021

National Science Foundation (NSF), Office of Advanced Cyberinfrastructure (OAC) and Advanced Technological Education (ATE)

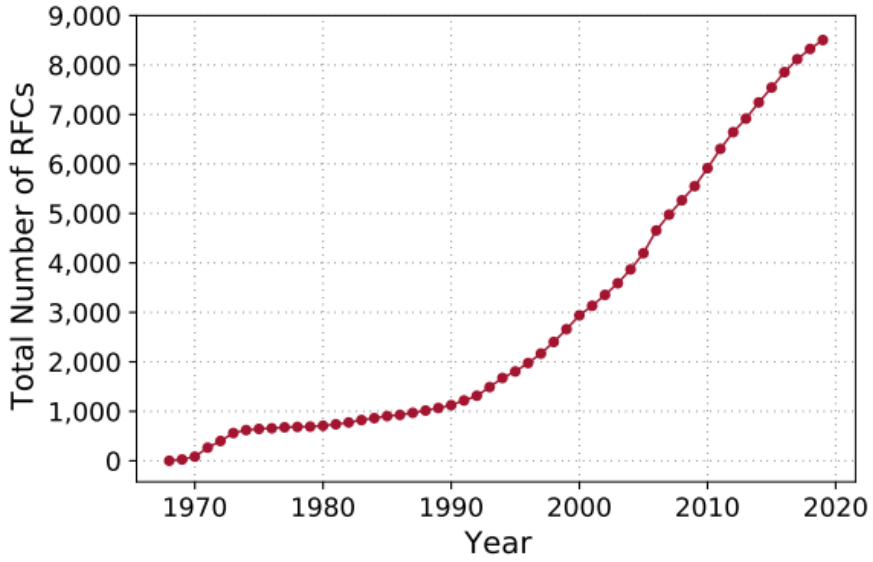
Introduction

- Traditionally, network devices have been designed with fixed functions to forward packets using a small set of protocols
- This closed-design paradigm has limited the capability of the switches to proprietary implementations which are hardcoded by vendors
- The process of updating devices is lengthy, costly, and inflexible process



Introduction

- Over time, thousands of new IETF RFCs and IEEE standards were written
- Router manufacturers needed to serve many customers with one product
- By the 2000s, routers were so complicated that they were based on more than 100 million lines of code (even though individual users only used few features / protocols)
- But time-to-market pressures meant they couldn't start over with a simpler design
- The research community labeled the Internet as "ossified"



Introduction

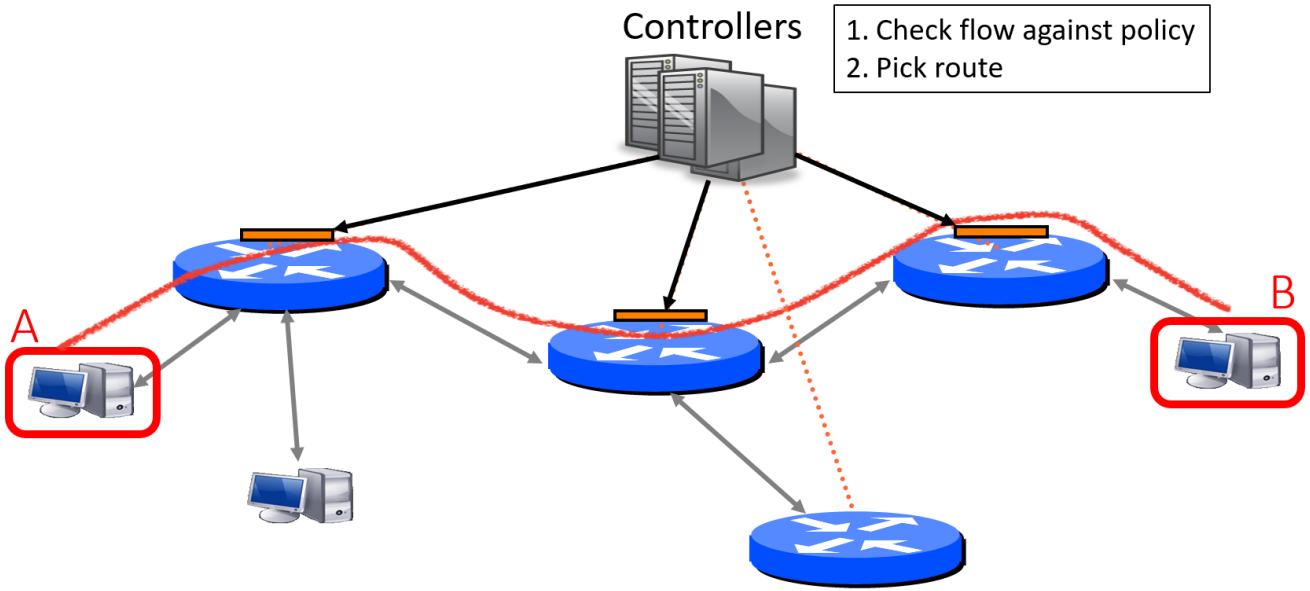
- Research programs at universities (GENI, FIND, Clean Slate, etc.) started to investigate how the Internet might move past this stagnation
- Ethane was the result of one of those project, the Clean Slate at Stanford
- Ethane proposed a different way of building and managing a network; it was tested at Stanford

Introduction

- Stanford network
  - 35,000 users
  - 10,000 new flows/sec
  - 137 network policies
  - 2,000 switches
  - 2,000 switch CPUs
- What if software decides to accept or no each flow, and how to route it?

Introduction

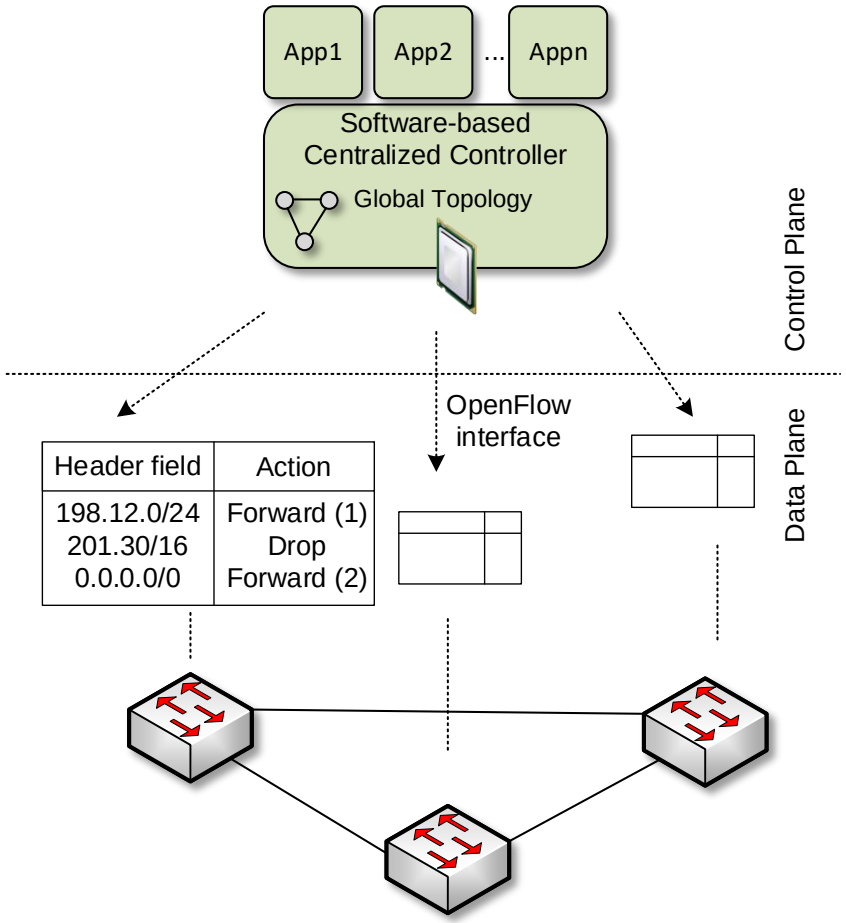
- Ethane
  - Separate the control and data planes
  - Implement the control plane intelligence as a software outside of the switches
  - Designed with FPGA-based switches (prototype in NetFPGA)
  - Flow are intercepted and sent to the centralized controller/s (reactive control)
  - One server suffices to implement controller, make decisions



Martín Casado, Nick McKeown, Scott Shenker, “From ethane to SDN and beyond,” ACM SIGCOMM Computer Communication Review, Vol. 49, Issue 5, 2019.

Introduction

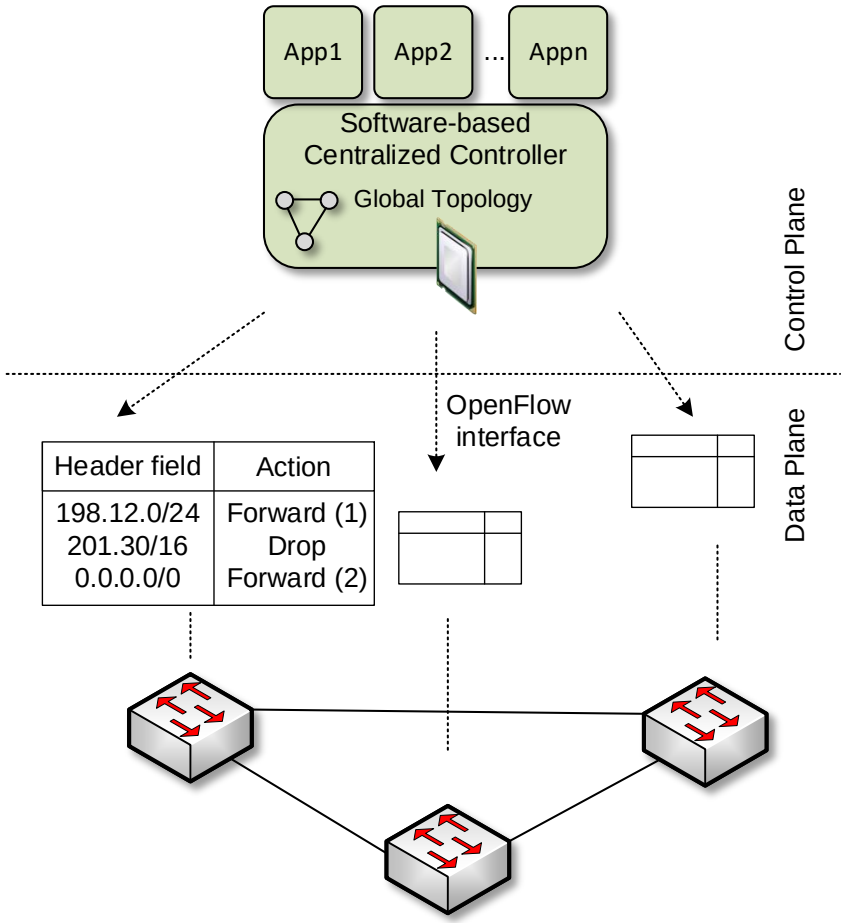
- Ethane reduced operational overhead and complexity
- It demonstrated that a network can be globally programmed from a centralized control plane



Martín Casado, Nick McKeown, Scott Shenker, “From ethane to SDN and beyond,” ACM SIGCOMM Computer Communication Review, Vol. 49, Issue 5, 2019.

Introduction

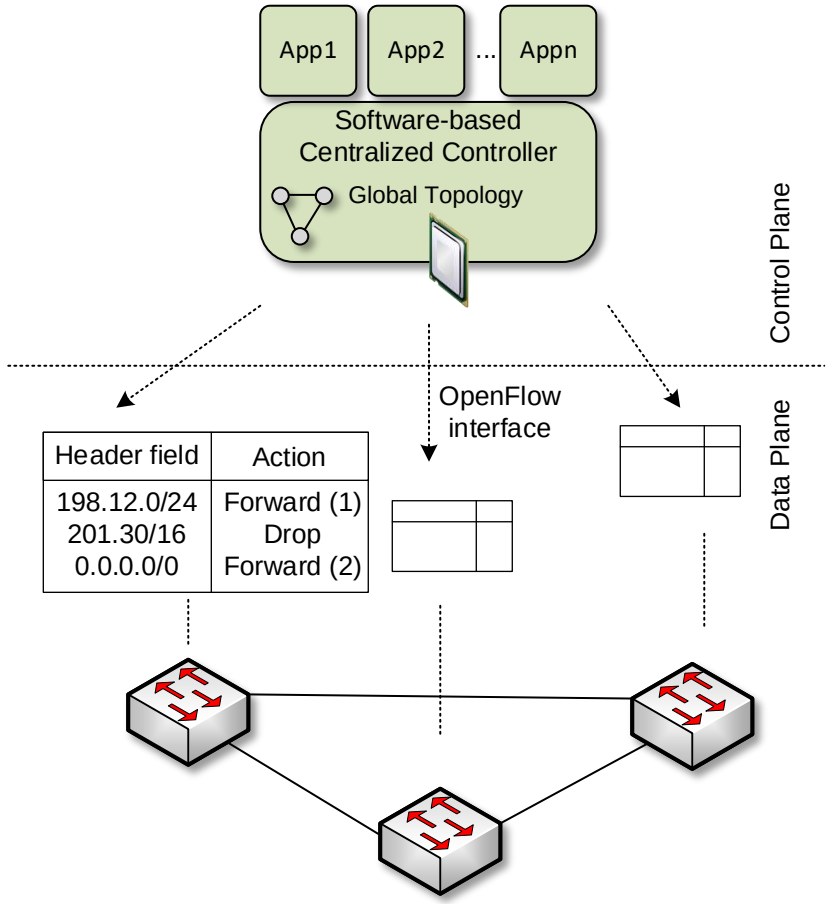
- Researchers from Ethane shared their ideas with software and networking equipment companies
- Software companies were very positive; software makes it easy to develop, test, and deploy new ideas
  - If you put software in the hands of developers, they will tailor it to their needs
- Networking companies were negative; they were threatened by a model that handed over control to the network owner



Martín Casado, Nick McKeown, Scott Shenker, "From ethane to SDN and beyond," ACM SIGCOMM Computer Communication Review, Vol. 49, Issue 5, 2019.

Introduction

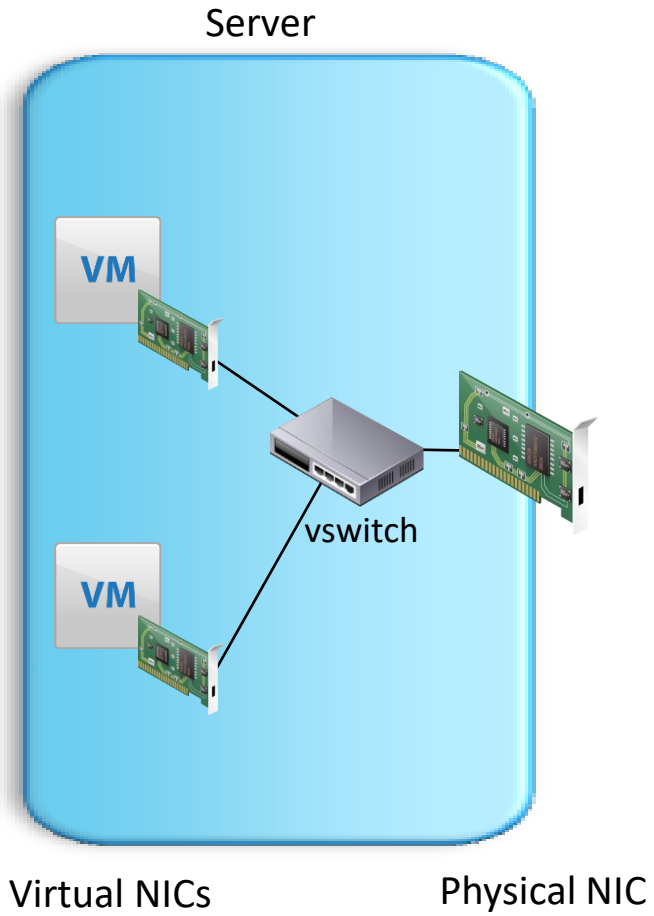
- The Ethane project created an open API (OpenFlow) that allowed the forwarding plane in each router to be externally configured, and
- A general SDN controller (NOX) that would use OpenFlow to control the forwarding plane
- However, networking equipment would be very resistant to this approach



Martín Casado, Nick McKeown, Scott Shenker, “From ethane to SDN and beyond,” ACM SIGCOMM Computer Communication Review, Vol. 49, Issue 5, 2019.

Introduction

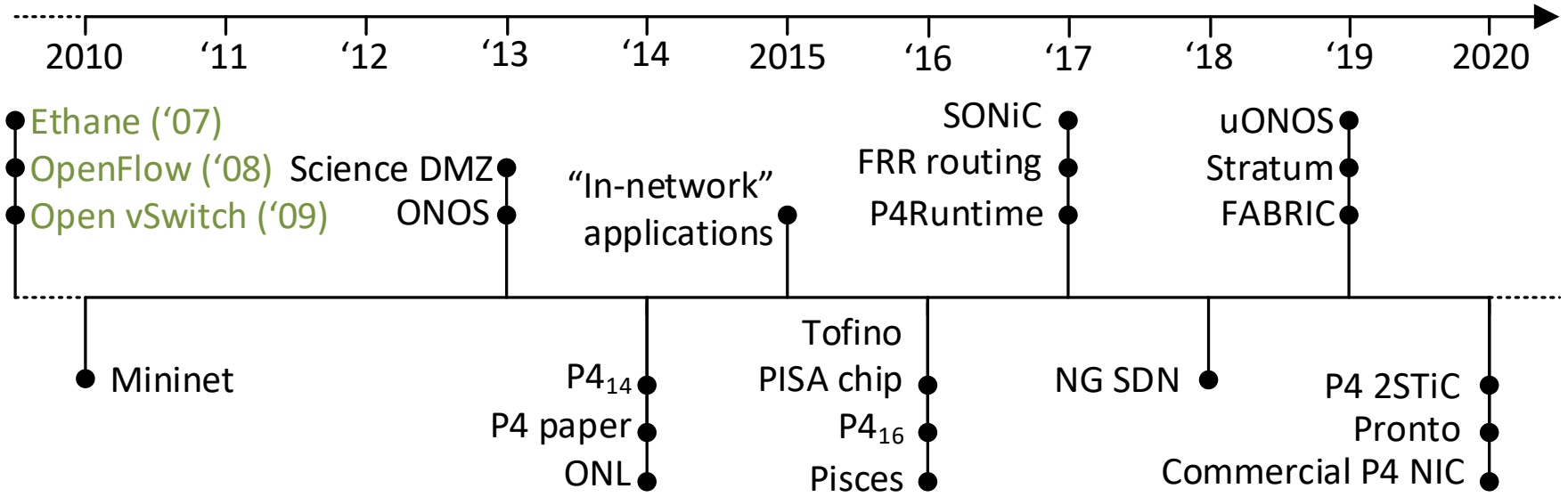
- Fortunately for the researchers, virtualized datacenters were becoming more popular
- In a virtualized datacenter, every server runs a software switch (vswitch) to connect VMs
- Nicira developed Open vSwitch (OVS) to provide an open-source remotely controlled vswitch
  - Viable deployment path: no dependency on network manufacturer, and market need



Martín Casado, Nick McKeown, Scott Shenker, “From ethane to SDN and beyond,” ACM SIGCOMM Computer Communication Review, Vol. 49, Issue 5, 2019.

Introduction

- Since the emergence of SDN and OVS, more open-source projects have been used in production networks



What is OVS?

- OVS is a production quality, multilayer virtual switch licensed under the open-source Apache 2.0 license
- It is designed to enable network automation through programmatic extension, while still supporting standard management interfaces and protocols (e.g., CLI, 802.1Q)
- OVS can operate both as a soft switch running within the hypervisor, and as the control stack for switching silicon
- It has been ported to virtualization platforms and switching chipsets

What is OVS?

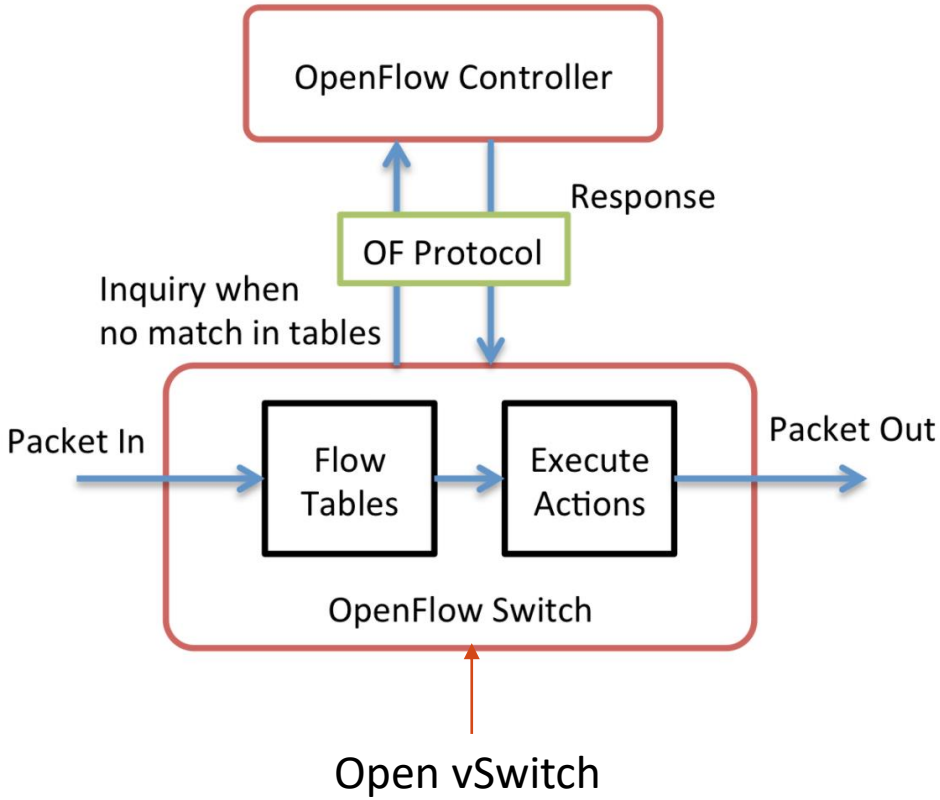
- An OVS switch forwards packets based on flow (rather than based on destination MAC or destination IP)
- A flow can be identified by a tuple (combination of fields)
  - IPv4 or IPv6 source address
  - IPv4 or IPv6 destination address
  - Input port
  - Ethernet frame type `
  - VLAN ID (802.1Q)
  - TCP/UDP source port
  - TCP/UDP destination port
  - Ethernet source address
  - Ethernet destination address IP
  - IP ToS (DSCP field) …

OVS features

- Visibility into inter-VM communication via NetFlow, sFlow, IPFIX
- Standard 802.1Q VLAN model with trunking
- Fine-grained QoS control
- OpenFlow protocol support
- IPv6 support
- Multiple tunneling protocols (GRE, VXLAN, STT, IPsec)
- Supports LACP- Link Aggregation Control Protocol
- Multicast snooping
- NIC bonding with source-MAC load balancing, active backup and L4 hashing
- Kernel and userspace forwarding engine options
- Multi-table forwarding pipeline with flow-caching engine

Open vSwitch and SDN

- Unlike other virtual switches, Open vSwitch supported OpenFlow since its inception
- It can be re-programmed through OpenFlow
- Other virtual switches have fixed packet processing pipelines
- In contrast to closed source virtual switches, Open vSwitch can operate with a user-selected operating system and hypervisor



Supported Platforms

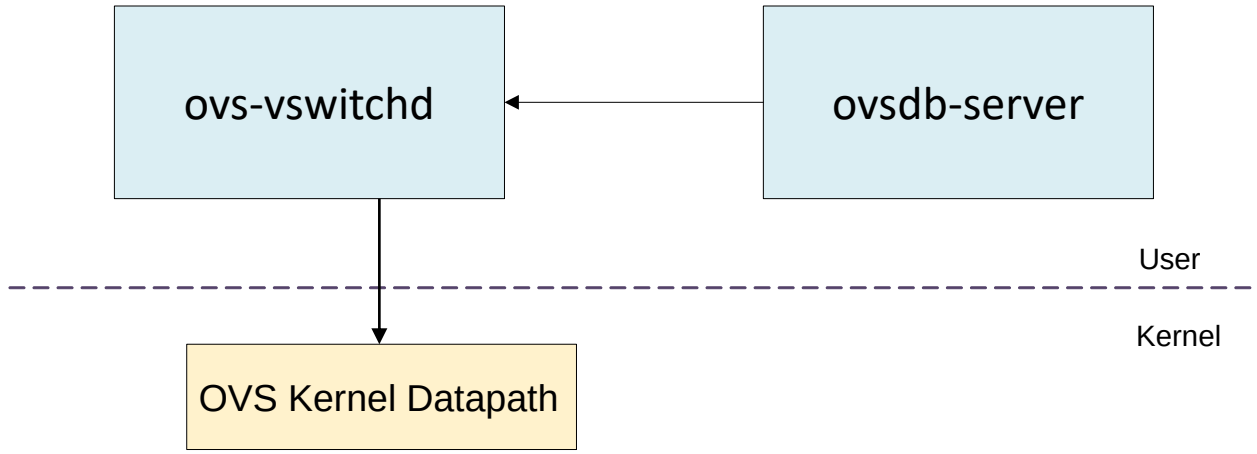
- Default switch in Xen and KVM
- Supported in VMware ESXi, MS Hyper-V
- Integrated in Openstack and vSphere
- Supported on Fedora, Debian, FreeBSD

Sample of Contributors

Open vSwitch Architecture

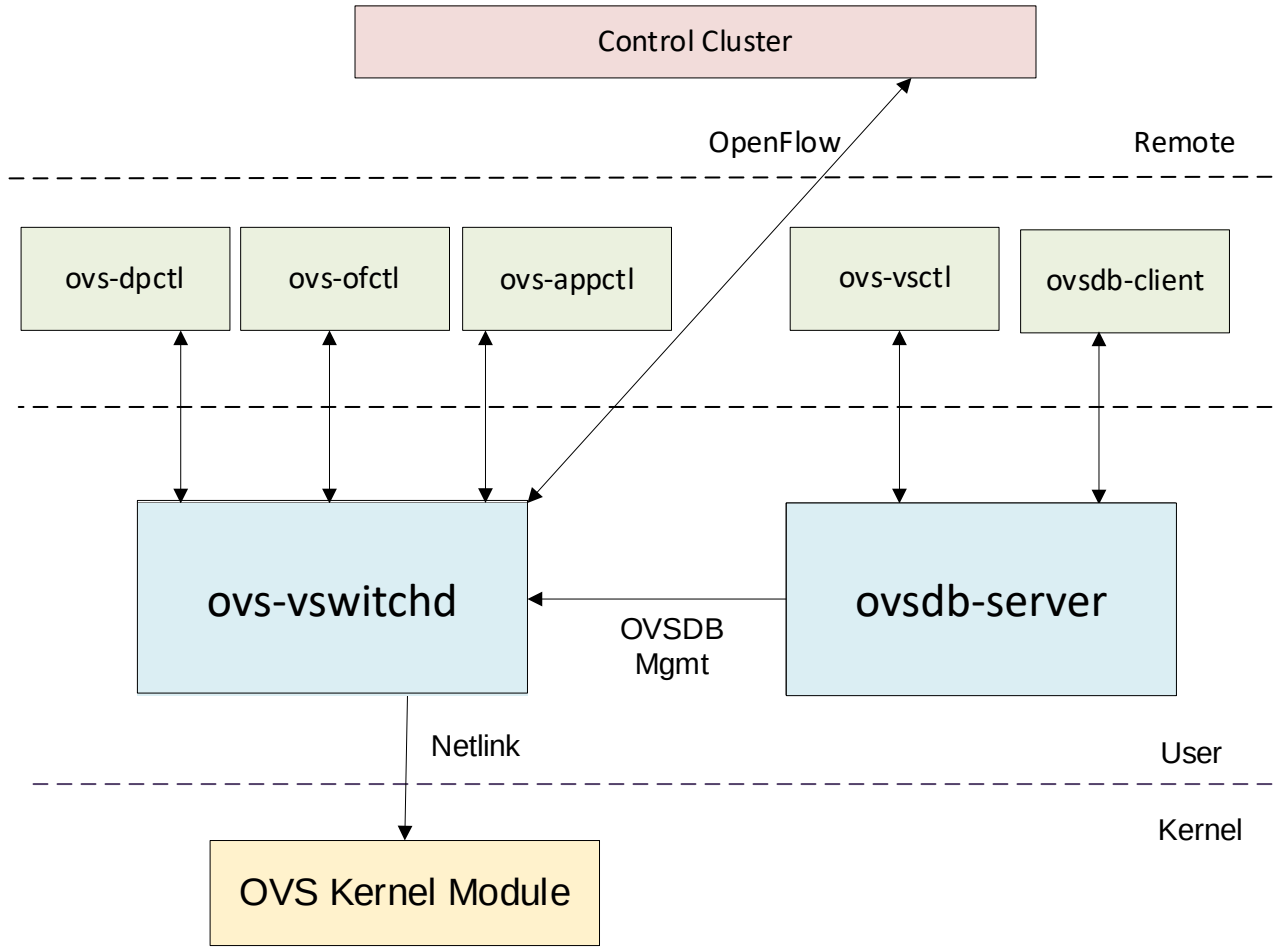
Open vSwitch Components

- Open vSwitch has three main components
- ovs-vswitchd: Open vSwitch daemon running in the userspace
- ovsdb-server: database server of Open vSwitch running in the userspace
- Datapath: Kernel space module, forwards Open vSwitch packets



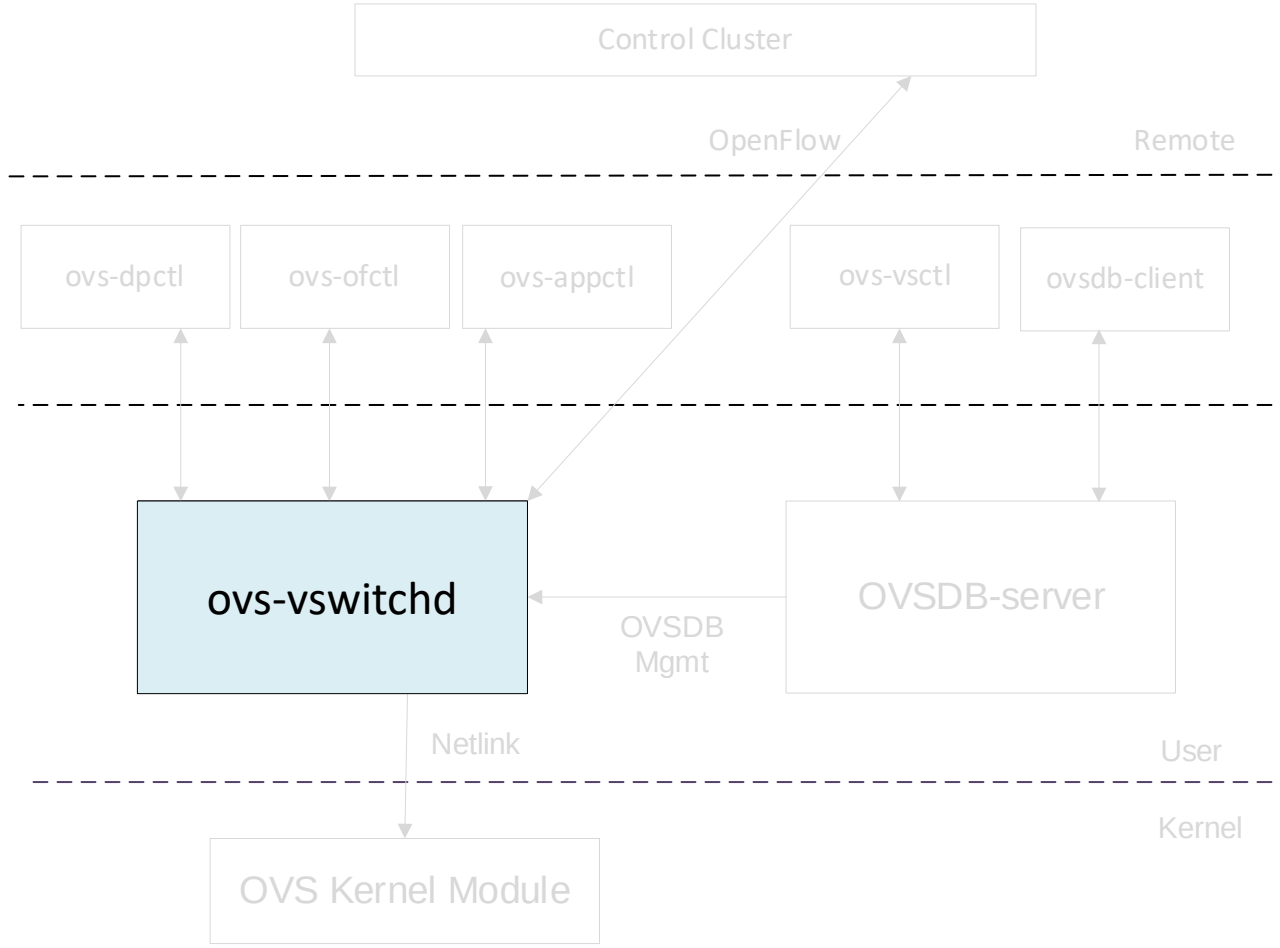
Open vSwitch Architecture

- Various tools are used to interact of the components of Open vSwitch
- External controller is typically used to populate flow table entries



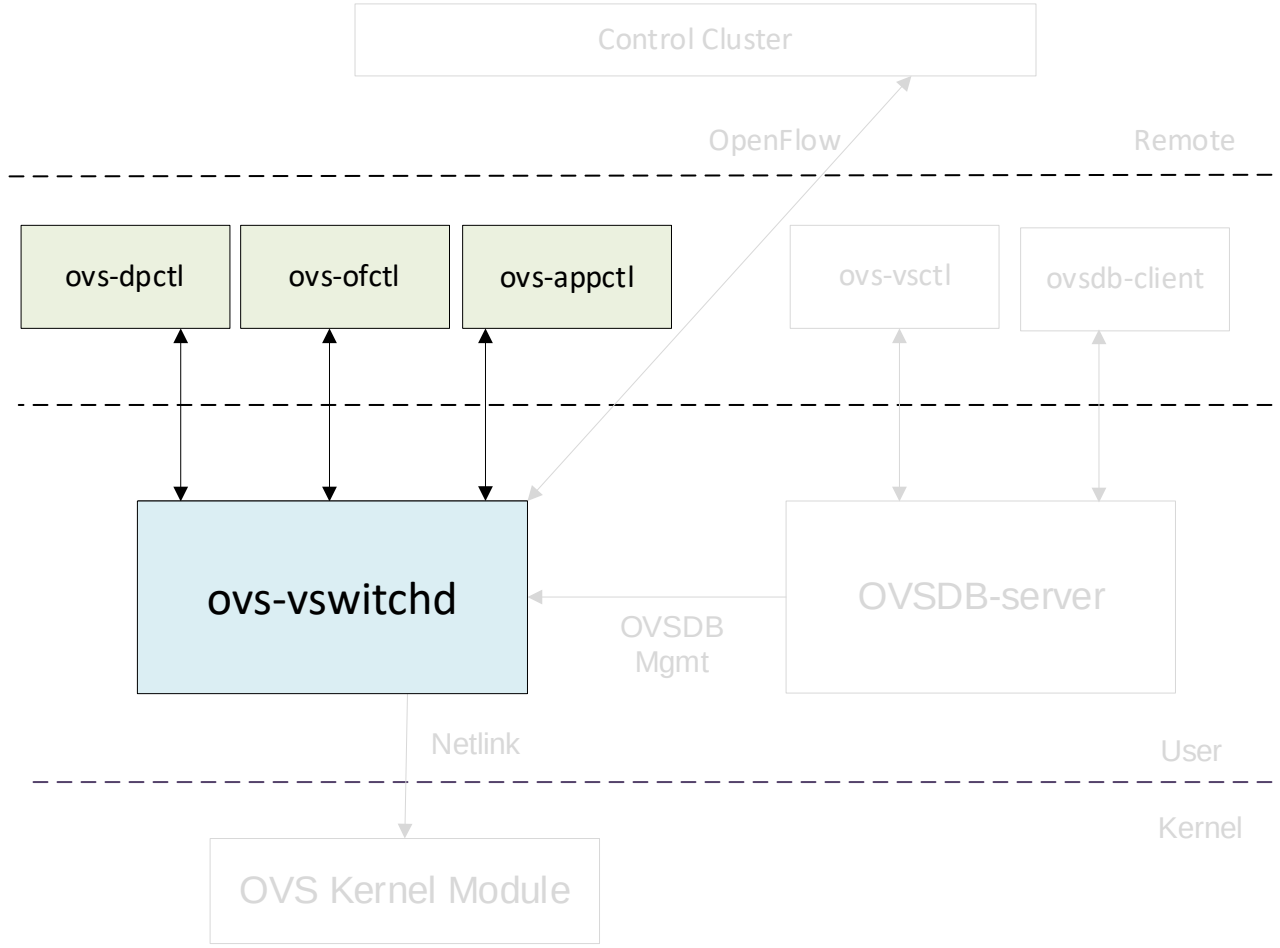
Open vSwitch Architecture

- ovs-vswitchd
  - Implements the switch
  - Communicates with the server through OVSDB management protocol
  - Communicates with the controller using OpenFlow
  - Talks to the Kernel Module via Netlink



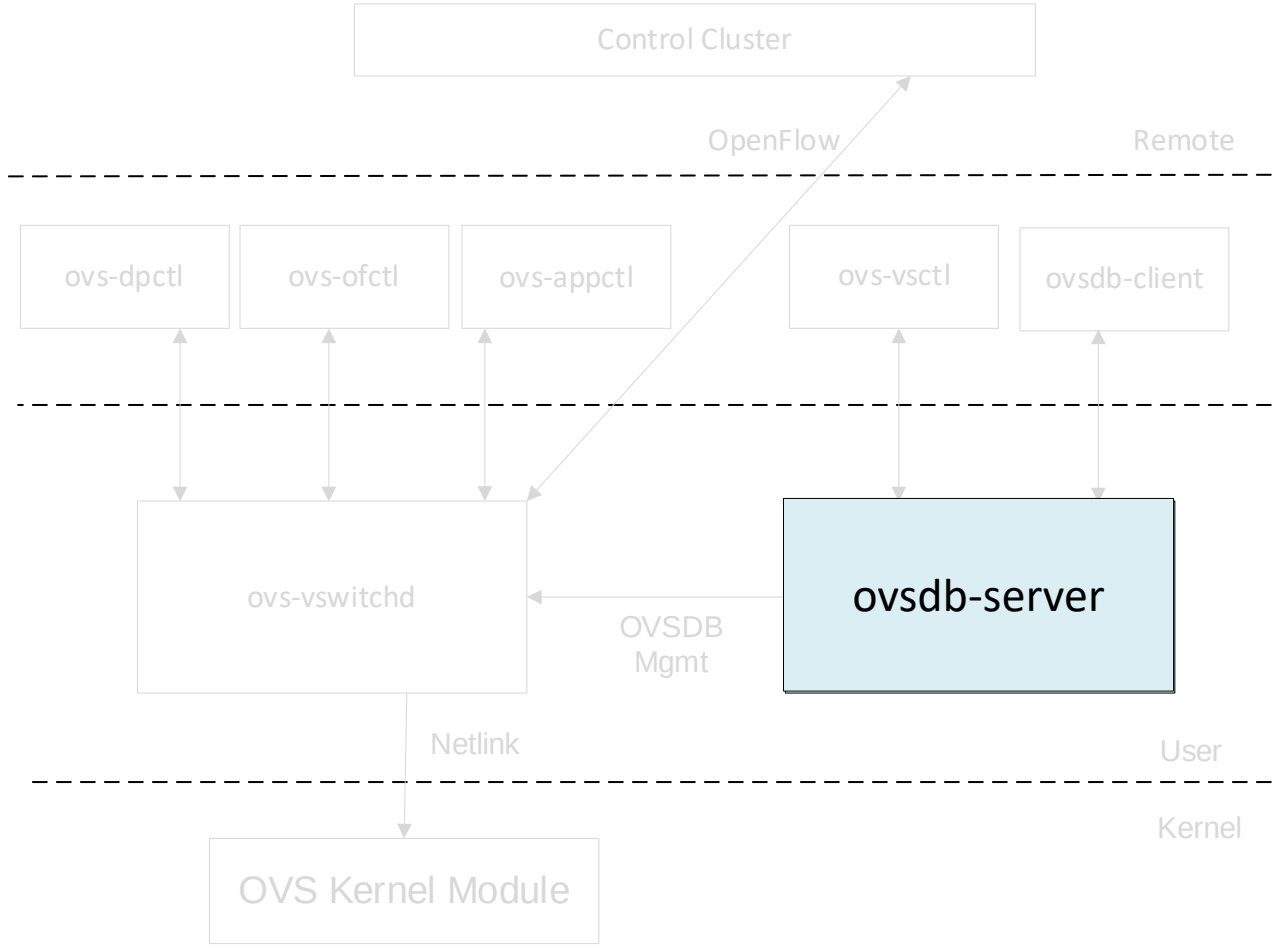
Open vSwitch Architecture

- ovs-dpctl tool
  - A command line tool responsible for creating, modifying and deleting Open vSwitch datapaths
- ovs-ofctl tool
  - A command line tool for monitoring and administering switches
  - Able to show the current state of a switch, features, configuration and table entries
- ovs-appctl tool
  - QoS, MAC, STP, …



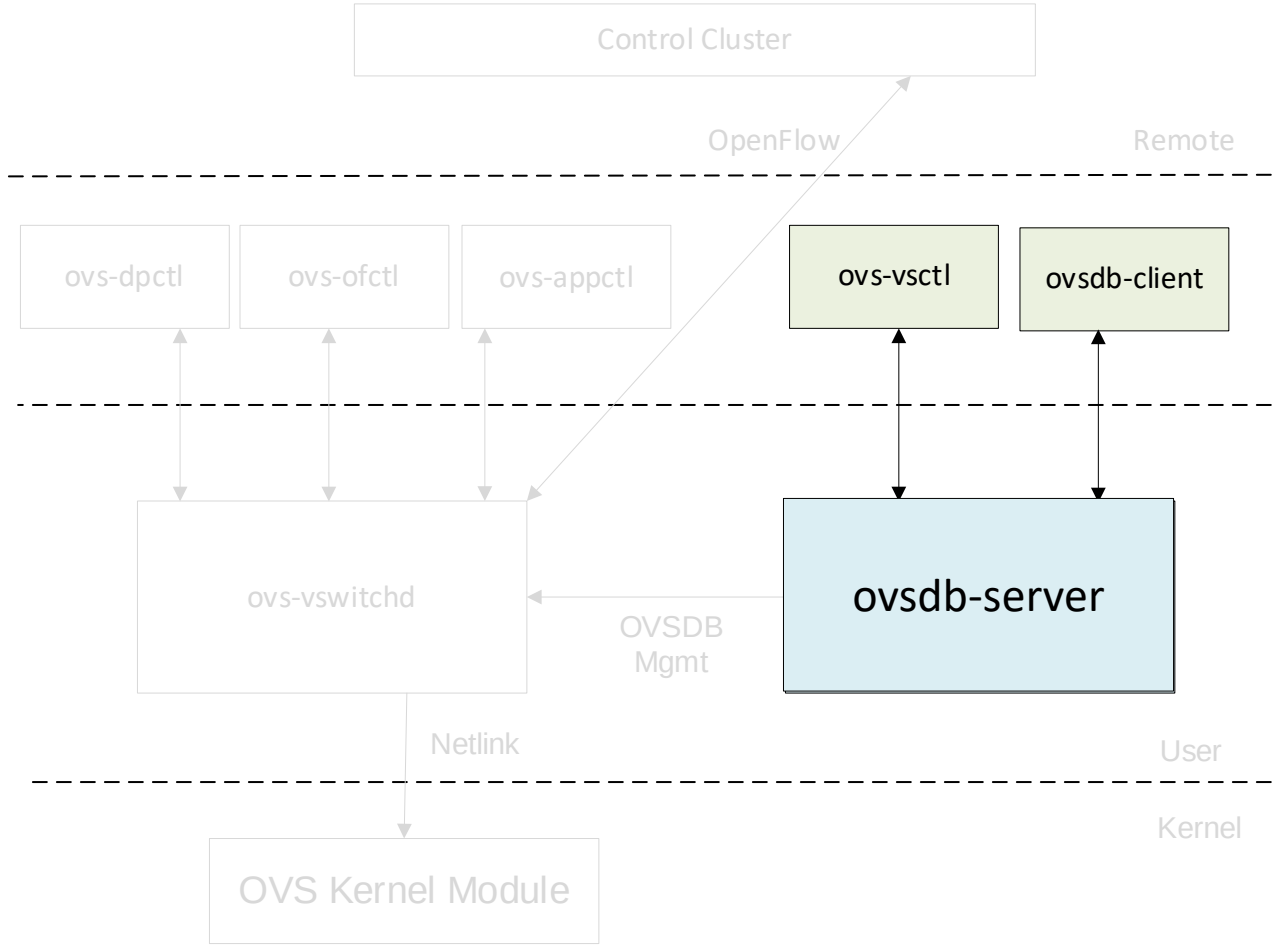
Open vSwitch Architecture

- ovsdb-server
  - Contains switch configuration, keeps track of created and modified interfaces
  - Communicates with ovs-vswitchd using OVSDB management protocol
  - Configuration is stored on persistent storage and survives a reboot



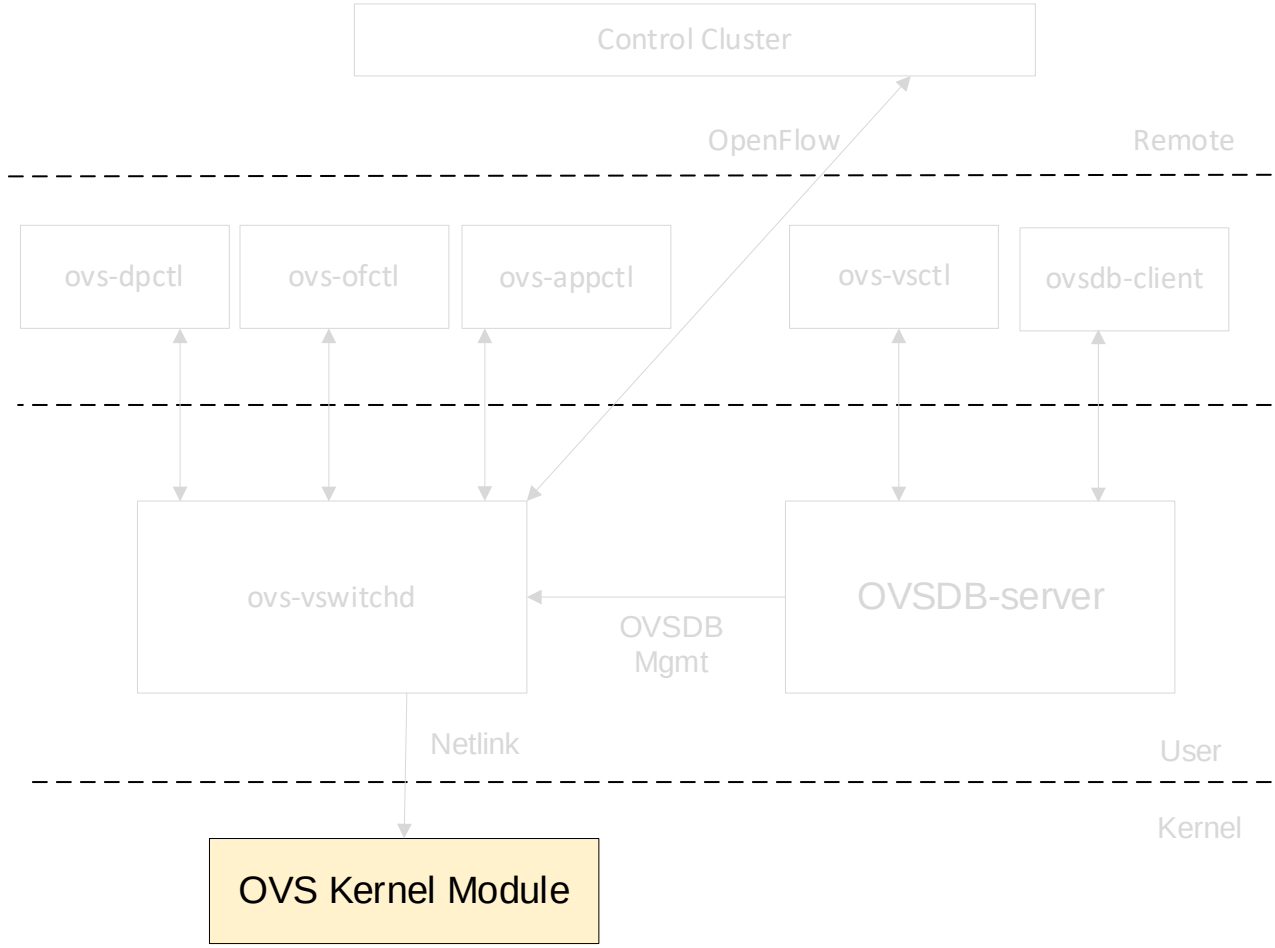
Open vSwitch Architecture

- ovs-vsctl tool
  - Manages the switch through interaction with ovsdb-server
  - Used to configure bridges, ports and tunnels
- ovsdb-client tool
  - A command line client for interacting with ovsdb-server



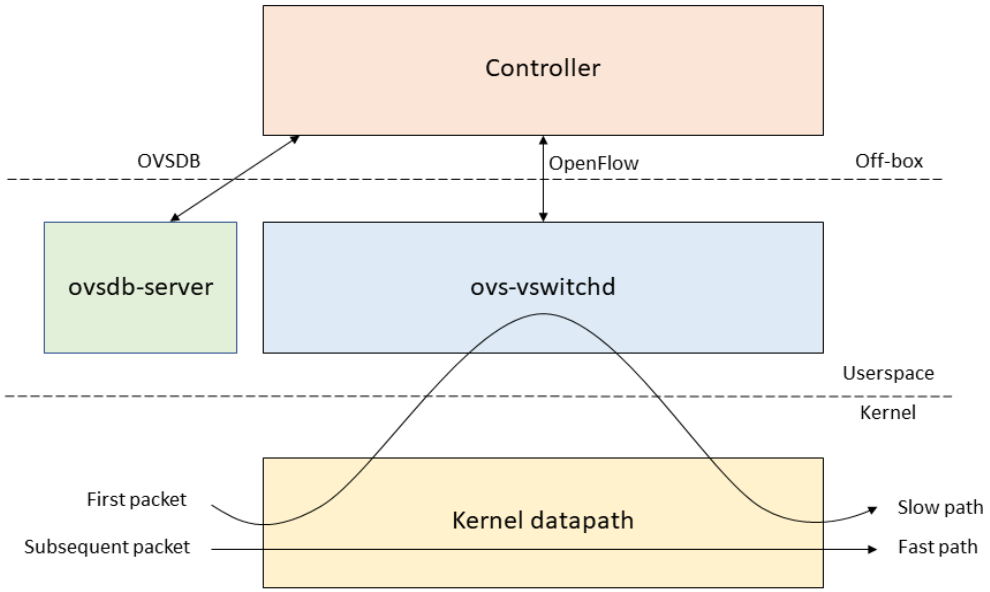
Open vSwitch Architecture

- OVS Kernel Module
  - Designed to be fast and simple
  - Handles switching and tunneling



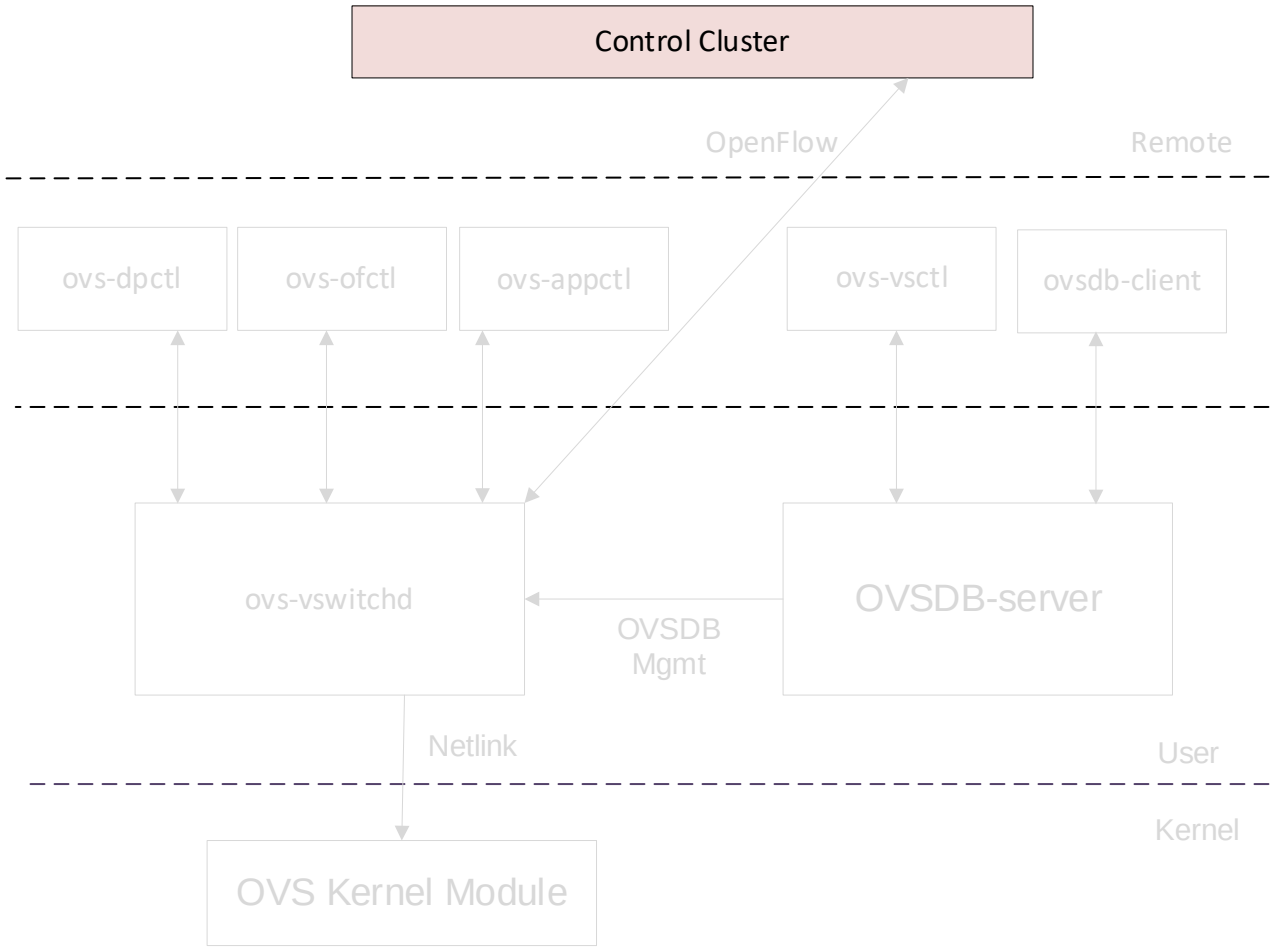
Open vSwitch Workflow

- Kernel receives packets from a physical network interface controller (NIC) or the virtual NIC of a virtual machine (VM)
- Kernel module directs the packet to the userspace. The userspace makes the decisions about the actions to be taken against the packet according to OpenFlow entries (slow path)
- The action entry is stored in the kernel, used to forward subsequent packets, making the forwarding faster (fast path)



Controller Interaction

- Control Cluster
- Manages any number of remote switches over OpenFlow protocol and determine the best path for application traffic

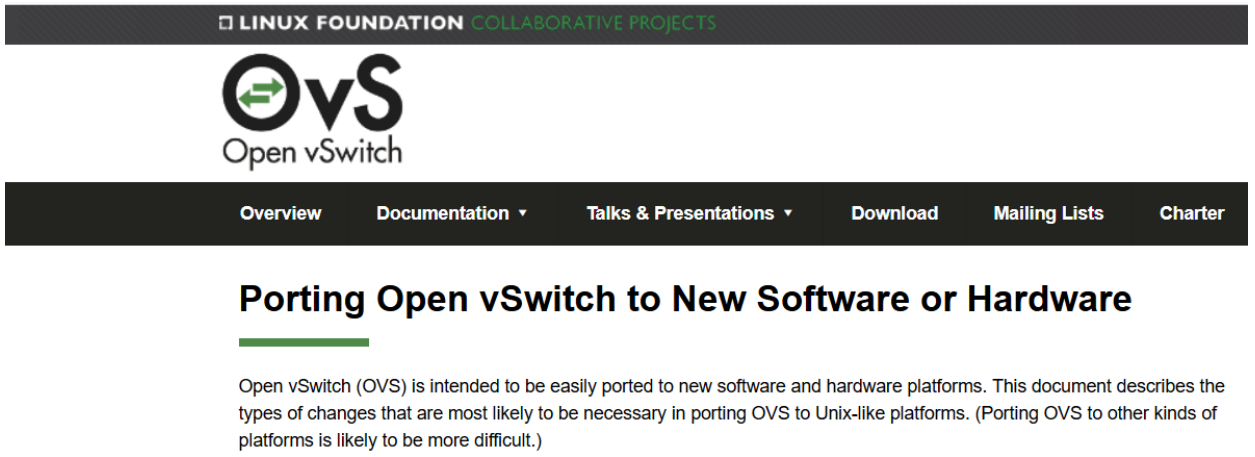


Open vSwitch Fail-modes

- Open vSwitch maintains flow tables that are consulted to determine how to forward traffic
- The flow tables entries are typically populated by a controller
- The controller might be down or not available
- Open vSwitch offers the option to operate in a **standalone** fail-mode
  - Open vSwitch will take over responsibility for setting up flows (regular MAC-learning)
- Alternatively, the switch can operate in **secure** mode
  - Switch will *not* set up flows on its own when the controller connection fails

Open vSwitch Portability

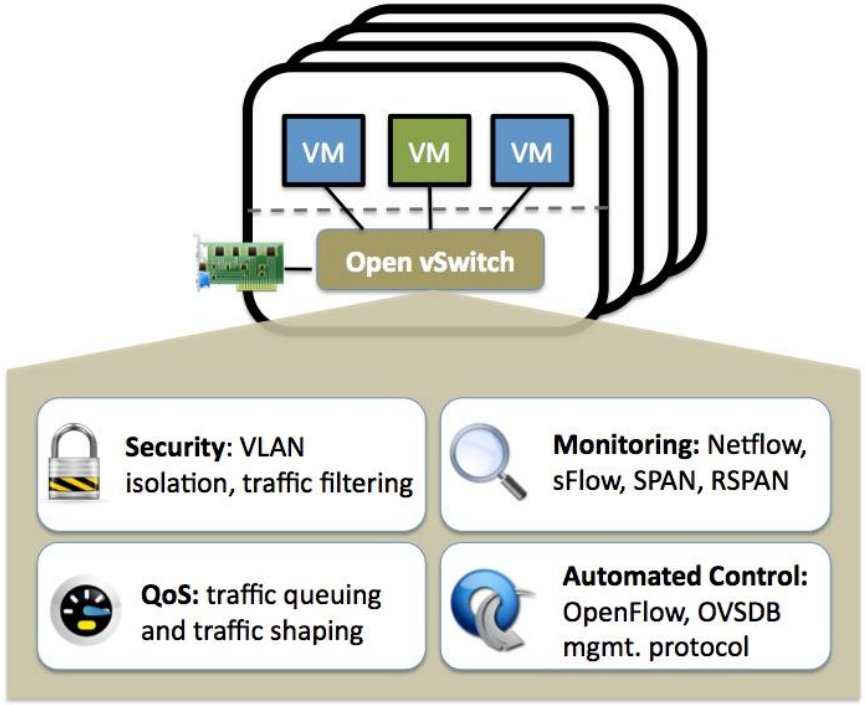
- Open vSwitch (OVS) is intended to be easily ported to new software and hardware platforms
- Datapath in the hardware instead of the kernel




```
+-------------------+
|    ovs-vswitchd   |<-->ovsdb-server
+-------------------+
|      ofproto      |<-->OpenFlow controllers
+--------+-+--------+
| netdev | | ofproto|
+--------+ |provider|
| netdev | +--------+
|provider|
+--------+
```

Open vSwitch Features

- Many of the features provided in standard hardware are provided by Open vSwitch
  - Standard 802.1Q VLAN model with trunking
  - Monitoring: NetFlow, sFlow, IPFIX
  - Spanning Tree Protocol (STP)
  - Quality of Service shaping and policing
  - Port mirroring (SPAN)
  - Tunning: GRE, VXLAN, Geneve
  - IPSec
  - IPv6 support
  - OpenFlow protocol support
  - LACP
  - Stateful/stateless firewalls through conntrack

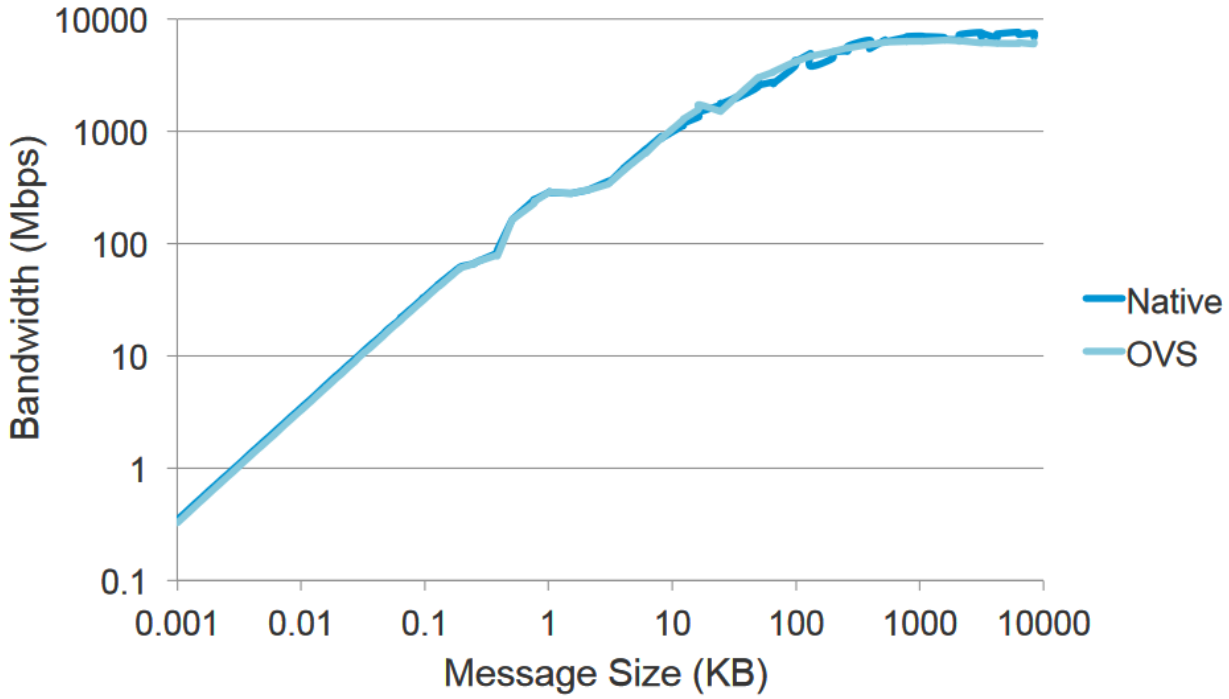


Open vSwitch Programmability

- The flow table in Open vSwitch is nearly a general-purpose processing pipeline
- Supported features:
  - Resubmit
  - Registers
  - Learning
  - Hashing and sampling

Open vSwitch Programmability

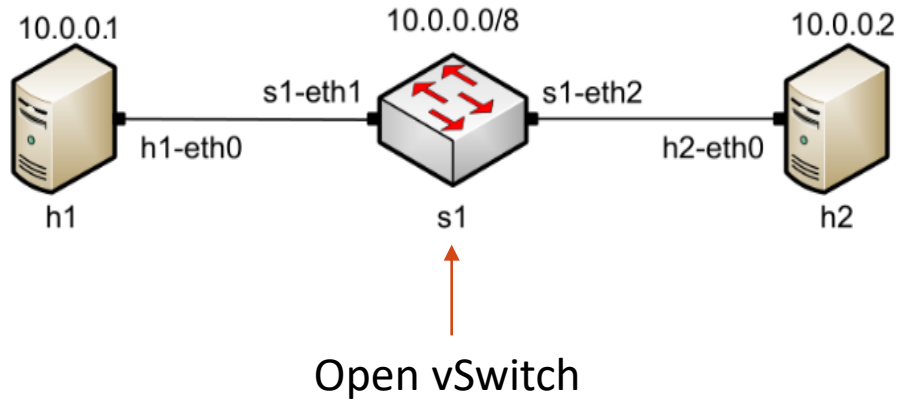
- The flow table in Open vSwitch is nearly a general-purpose processing pipeline
- Supported features:
  - Resubmit
  - Registers
  - Learning
  - Hashing and sampling
- Does programmability impact forwarding rates?[1]
  - Established flows
  - Connection setup
  - Many sustained connections



[1] VMware. LinuxCon Japan. Online: https://events.static.linuxfound.org/sites/events/files/cojp13_gross.pdf

Open vSwitch Performance

- Lab series on high-speed networks using emulation
- Experiments conducted at the University of South Carolina with switching hardware
- Switch is acting as a MAC-learning/layer2 device



"Host: h1"

```
root@admin-pc:~# iperf3 -c 10.0.0.2
Connecting to host 10.0.0.2, port 5201
[ 13] local 10.0.0.1 port 59414 connected to 10.0.0.2 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[ 13]   0.00-1.00   sec  5.18 GBytes  44.5 Gbits/sec    0    843 KBytes
[ 13]   1.00-2.00   sec  5.21 GBytes  44.7 Gbits/sec    0   1.11 MBytes
[ 13]   2.00-3.00   sec  5.20 GBytes  44.7 Gbits/sec    0   1.18 MBytes
[ 13]   3.00-4.00   sec  5.21 GBytes  44.7 Gbits/sec    0   1.24 MBytes
[ 13]   4.00-5.00   sec  5.19 GBytes  44.6 Gbits/sec    0   1.24 MBytes
[ 13]   5.00-6.00   sec  5.22 GBytes  44.8 Gbits/sec    0   1.30 MBytes
[ 13]   6.00-7.00   sec  5.24 GBytes  45.0 Gbits/sec    0   1.44 MBytes
[ 13]   7.00-8.00   sec  5.22 GBytes  44.9 Gbits/sec    0   1.44 MBytes
[ 13]   8.00-9.00   sec  5.21 GBytes  44.8 Gbits/sec    0   1.45 MBytes
[ 13]   9.00-10.00  sec  5.22 GBytes  44.8 Gbits/sec    0   1.52 MBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Retr
[ 13]   0.00-10.00  sec  52.1 GBytes  44.8 Gbits/sec    0             sender
[ 13]   0.00-10.04  sec  52.1 GBytes  44.6 Gbits/sec                  receiver

iperf Done.
root@admin-pc:~#
```

Ongoing and Future Work

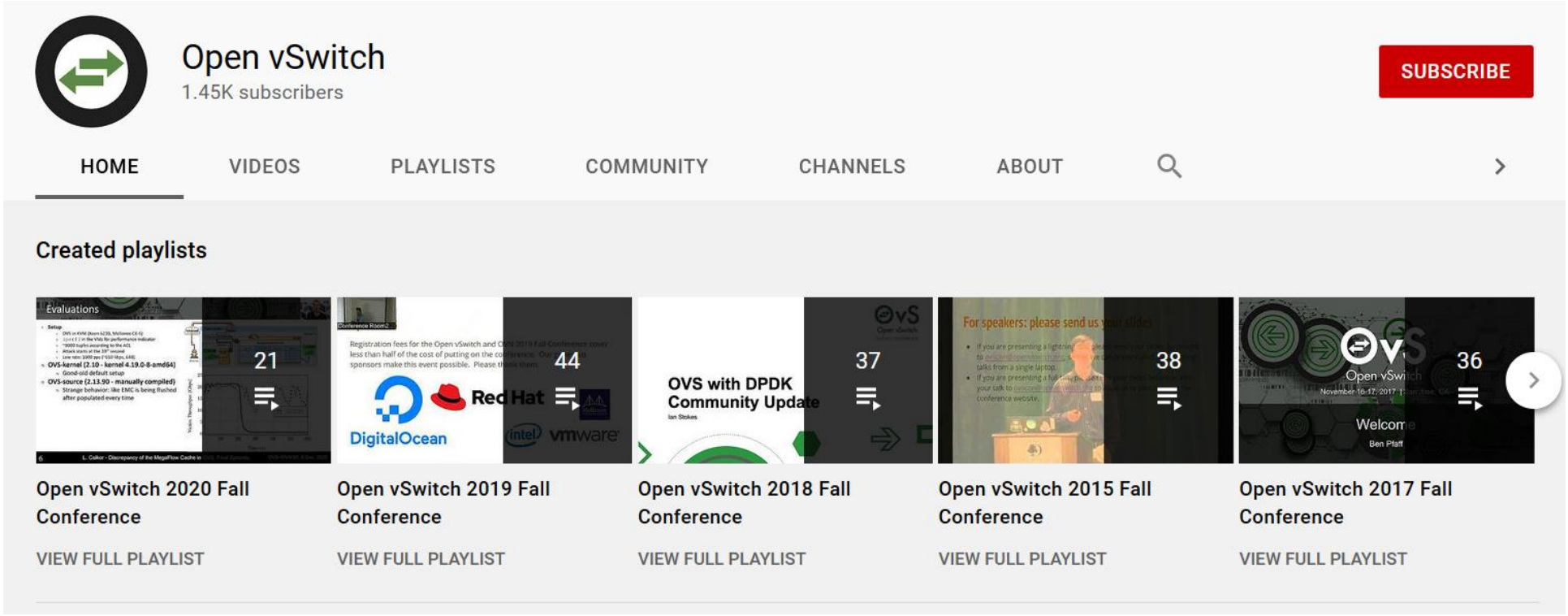
- Most releases of Open vSwitch add support for new fields or protocols
- Every change to Open vSwitch requires building, distributing, and installing the new version
- Every field needs coordination with controller authors (+ONF)
- Define fields and protocols with P4 (de-facto language for data plane programming)
- ~300 lines of P4 for everything in Open vSwitch[1]
- Early efforts: PISCES[2]

[1] Ben Pfaff. Nicira, “P4 and Open vSwitch”. Online: http://www.openvswitch.org//support/slides/p4.pdf

[2] PISCES, “A Programmable, Protocol-Independent Software Switch”. Online: https://p4-vswitch.github.io/

Further Readings

- Open vSwitch official website: https://www.openvswitch.org/
- Pfaff et al. The Design and Implementation of Open vSwitch. USENIX NSDI 15.
- Open vSwitch documentation. https://tinyurl.com/5etbz9ae



Additional Slides

Virtualization

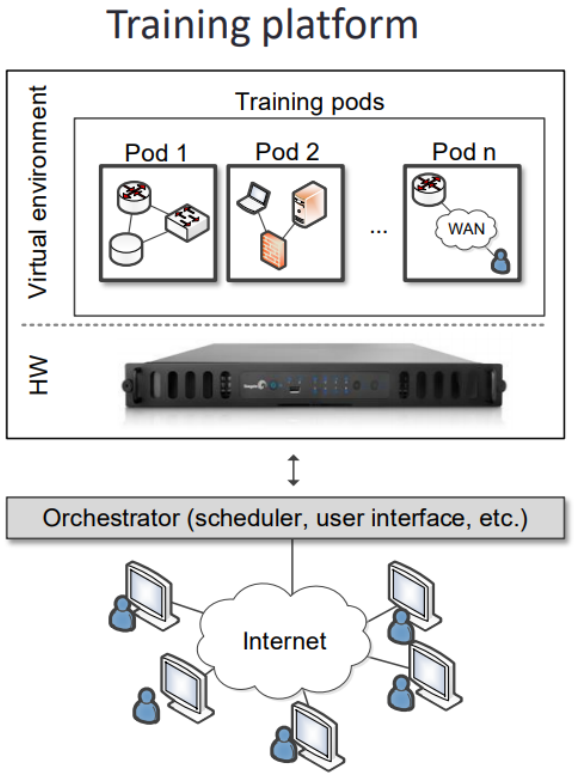
- Virtualization is the ability to run multiple operating systems on a single physical system and share the underlying hardware resources[1]
- According to Cisco, 94% of all workloads will run in some form of cloud environment by the end of 2021[2]

[1] Vmware white paper. Virtualization Overview. Online: https://www.vmware.com/pdf/virtualization.pdf

[2] Cisco. Cisco's Hybrid Cloud Vision meets Black Belt Academy. Online: https://tinyurl.com/3ys4t3dd

Virtualization

- Virtualization is the ability to run multiple operating systems on a single physical system and share the underlying hardware resources[1]
- According to Cisco, 94% of all workloads will run in some form of cloud environment by the end of 2021[2]
- The training platform we are using today is all virtualized

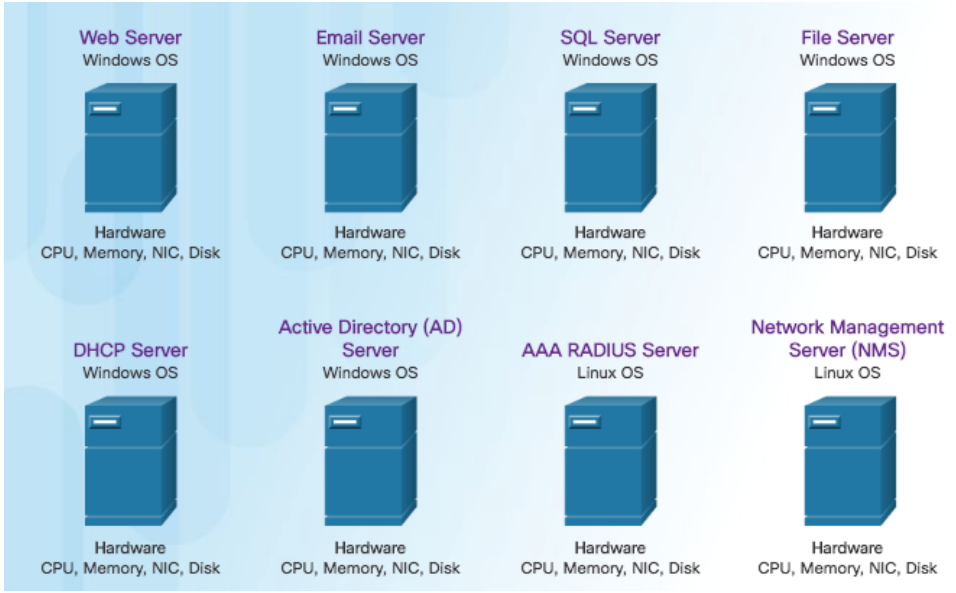


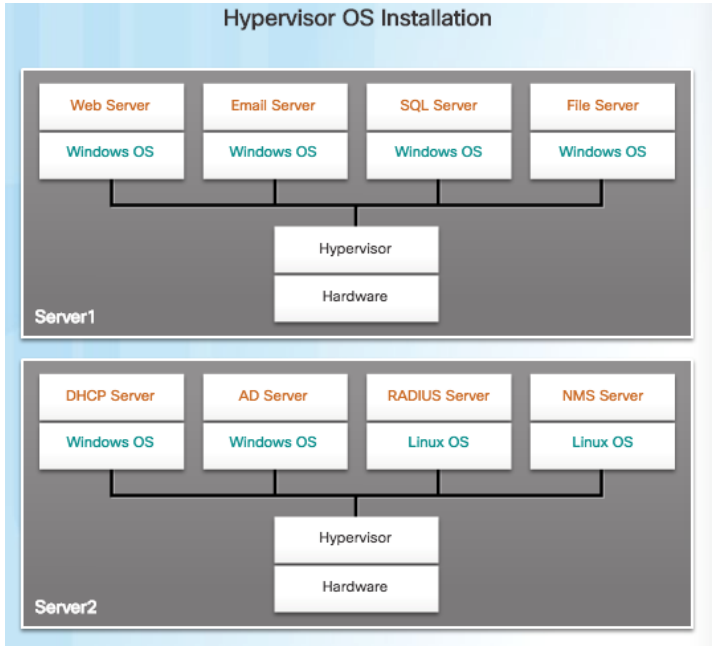
[1] VMware white paper. Virtualization Overview. Online: https://www.vmware.com/pdf/virtualization.pdf

[2] Cisco. Cisco's Hybrid Cloud Vision meets Black Belt Academy. Online: https://tinyurl.com/3ys4t3dd

Dedicated vs Virtualized Servers

- Dedicated servers versus server virtualization[1]





[1] Cisco. Chapter 7: Network Evolution, CCNA Routing and Switching.

Network Virtualization

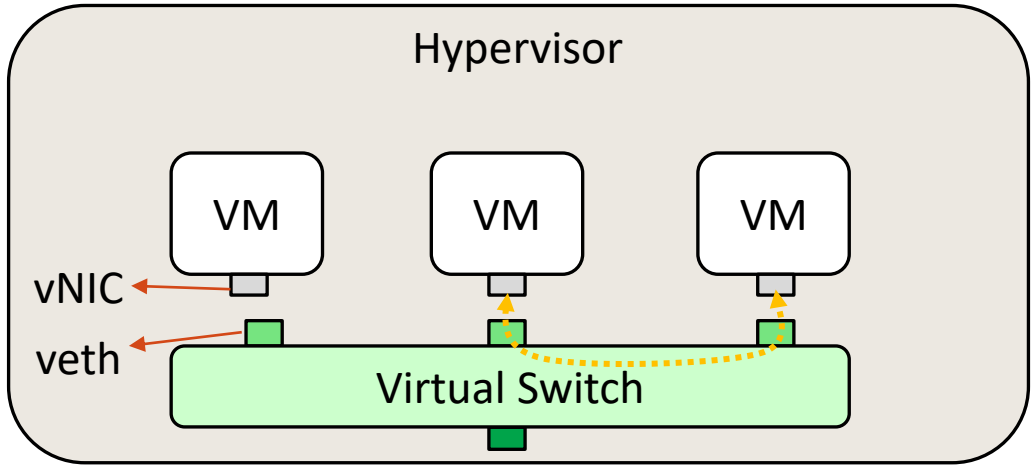
- Virtual machines on the same server and virtual machines on different servers often need to communicate
- Networking between VMs is not straightforward
- Scalability: 10K VMs or much more
- Isolation: many switches can operate in different locations
- Mobility: network state associated with network entities should be migratable

[1] Vmware white paper. Virtualization Overview. Online: https://www.vmware.com/pdf/virtualization.pdf

[2] Cisco. Cisco's Hybrid Cloud Vision meets Black Belt Academy. Online: https://tinyurl.com/3ys4t3dd

Virtual Switch

- A virtual switch is essential to interconnect VMs
- VMs can reside on a single server



Virtual Switch

- A virtual switch is essential to interconnect VMs
- VMs can reside on a single
- VMs can reside on different servers