Open vSwitch:
A Whirlwind Tour

Justin Pettit

March 3, 2011

Overview

- Visibility (NetFlow, sFlow, SPAN/RSPAN)
- Fine-grained ACLs and QoS policies
- Centralized control through OpenFlow
- Port bonding, LACP, tunneling
- Works on Linux-based hypervisors: Xen, XenServer, KVM, VirtualBox
- Open source, commercial-friendly Apache 2 license
- Multiple ports to physical switches

Visibility

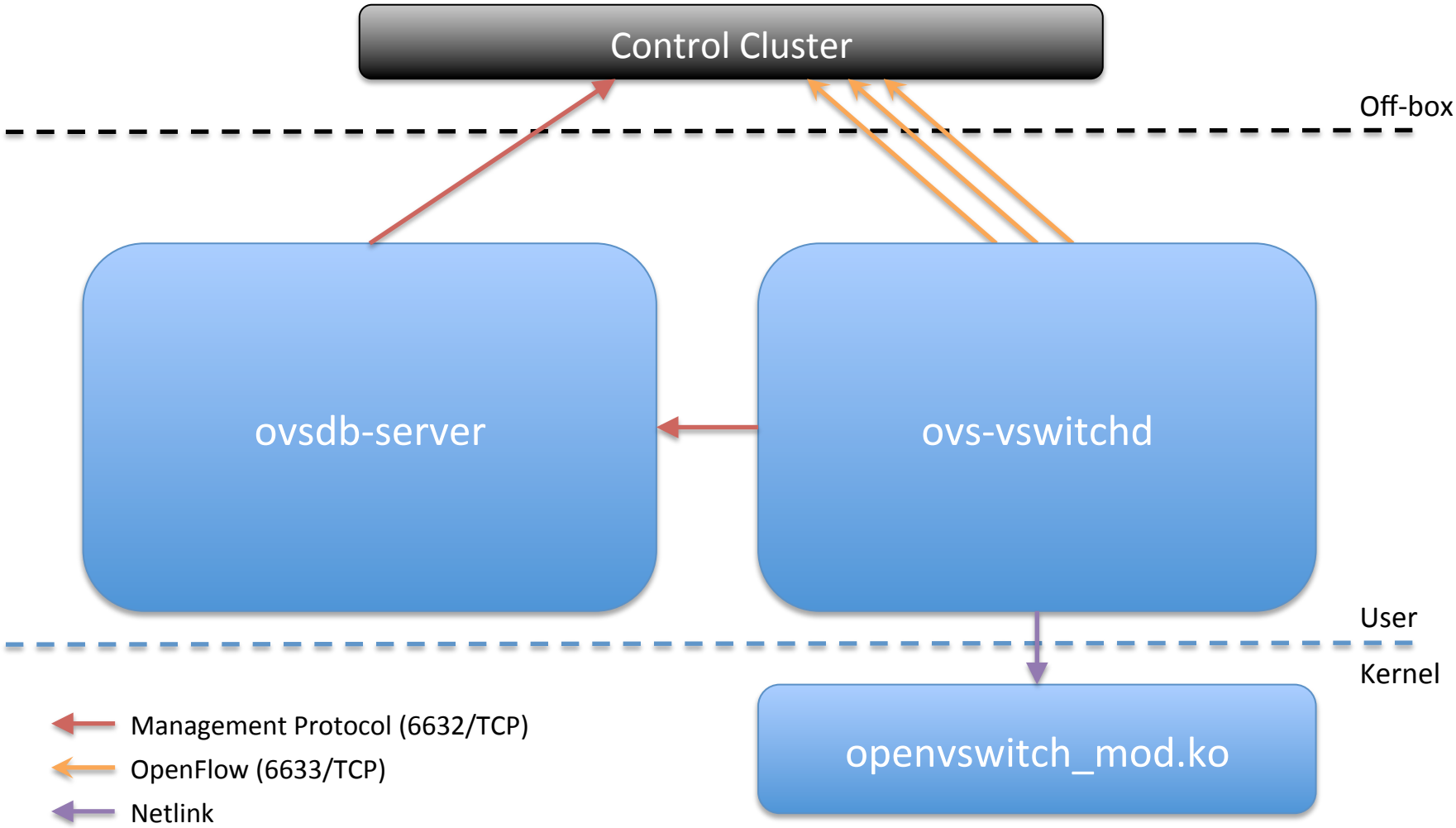
- Number of subscribers to mailing lists:
  - discuss: 309
  - announce: 195
  - dev: 161
  - git: 48
- openvswitch.org gets about 4900 unique visitors per month

(Partial) List of Contributors

External-facing Development

- Work underway to upstream kernel module
- Fix VLAN handling in kernel
- Default networking stack for Xen Cloud Platform (XCP) and next XenServer release
- Distribution packaging
  - Debian
  - Ubuntu
  - SUSE
  - Red Hat

Main Components



ovsdb-server

- Database that holds switch-level configuration
- Custom database with nice properties:
  - Value constraints
  - Weak references
  - Garbage collection
- Log-based (awesome for debugging!)
- Speaks management protocol (JSON-RPC) to manager and ovs-vswitchd

Tools: ovs-vsctl, ovsdb-tool, ovsdb-client, ovs-appctl

ovs-vswitchd

- Core component in the system:
  - Communicates with outside world using OpenFlow
  - Communicates with ovsdb-server using management protocol
  - Communicates with kernel module over netlink
  - Communicates with the system through netdev abstract interface
- Supports multiple independent datapaths (bridges)
- Packet classifier supports efficient flow lookup with wildcards and "explodes" these (possibly) wildcard rules for fast processing by the datapath
- Implements mirroring, bonding, and VLANs through modifications of the same flow table exposed through OpenFlow
- Checks datapath flow counters to handle flow expiration and stats requests

Tools: ovs-ofctl, ovs-appctl

openvswitch_mod.ko

- Kernel module that handles switching and tunneling
- Exact-match cache of flows
- Designed to be fast and simple
  - Packet comes in, if found, associated actions executed and counters updated. Otherwise, sent to userspace
  - Does no flow expiration
  - Knows nothing of OpenFlow
- Implements tunnels

Tools: ovs-dpctl

Types of Channels

- One OpenFlow connection per datapath
  - Flow table configuration
- One management channel per system
  - Switch-level configuration
  - Resources
  - Counters

OpenFlow

- Idealized view of a switch's datapath
- Centralized controller configures flow table
  - Lookup based on L2-L4
  - Supports full wildcarding and priorities
  - Flows associated with actions: forward, drop, modify
  - Missed flows go to controller
- Remote visibility
  - Description of switch (supported actions, flow tables' sizes, etc.)
  - Statistics (flows, tables, ports)

Nicira Extensions to OpenFlow

- Resubmit
- NXM (Extensible Match)
  - Tunnels
  - Registers
  - IPv6
  - Labels used by new actions
- Flexible tunnel tagging
- Multiple controllers
- Separate setting a QoS queue from transmitting
- Multipathing

Management Channel

- Built around configuration database
  - Simple type system, batching, key/value, triggers, referential integrity
- Benefits:
  - No global lock
  - Granular updates
  - Allows multiple front-ends (OpenFlow management, SNMP, CLI)
- In addition to configuration, it is also used to retrieve stats

Tunneling

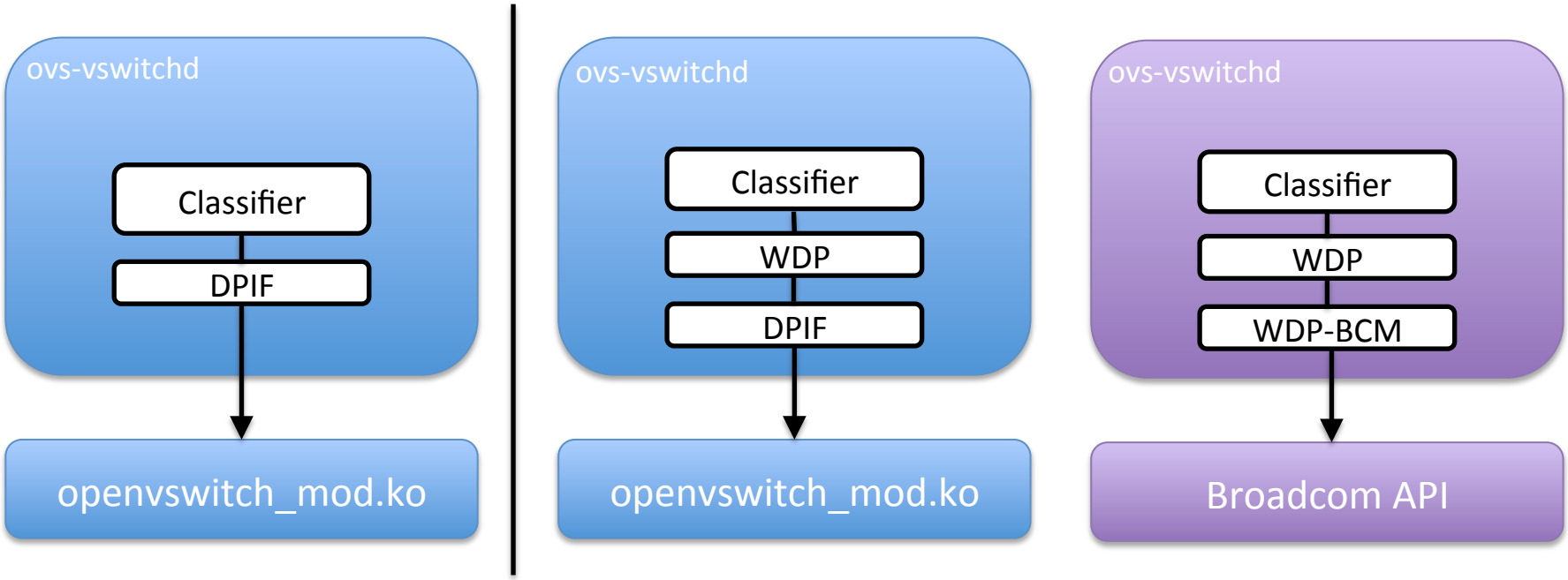
- Required to provide “true” virtual networks
- Focus on performance
  - Header caching
  - Hardware offloading
- Supported tunneling modes
  - GRE
  - GRE-over-IPsec
  - CAPWAP

Bringing OVS to Hardware

- Hardware switches have slow CPUs but fast specialized hardware
- Exact match flows are the wrong approach for TCAMs*
- netdev abstraction
- WDP (wildcard datapath) abstraction
  - Currently a branch, in the process of reimplementing in master

*Expensive and power-hungry

WDP Architecture



Standard CS Response: Introduce layer of indirection!