Open vSwitch and Software Defined Networking

Jesse Gross
Nicira

LinuxCon Japan June 6, 2012

Open vSwitch: What is it?

- Open source switching stack for virtualization.
- The new entry point into the network is the hypervisor.
- Two ways to view OVS:
  - Gaining back visibility and control that usually comes from the features of a hardware switch
  - An opportunity to exploit the flexibility that comes from software and virtualization

Open vSwitch: Why do I care?

- Data centers are both larger and more dynamic than before.
- Virtualization provides flexibility to computation resources but the network is now the bottleneck.
- This requires:
  - a programmatic interface
  - access to hypervisor state
  - data plane constructs to build a distributed system

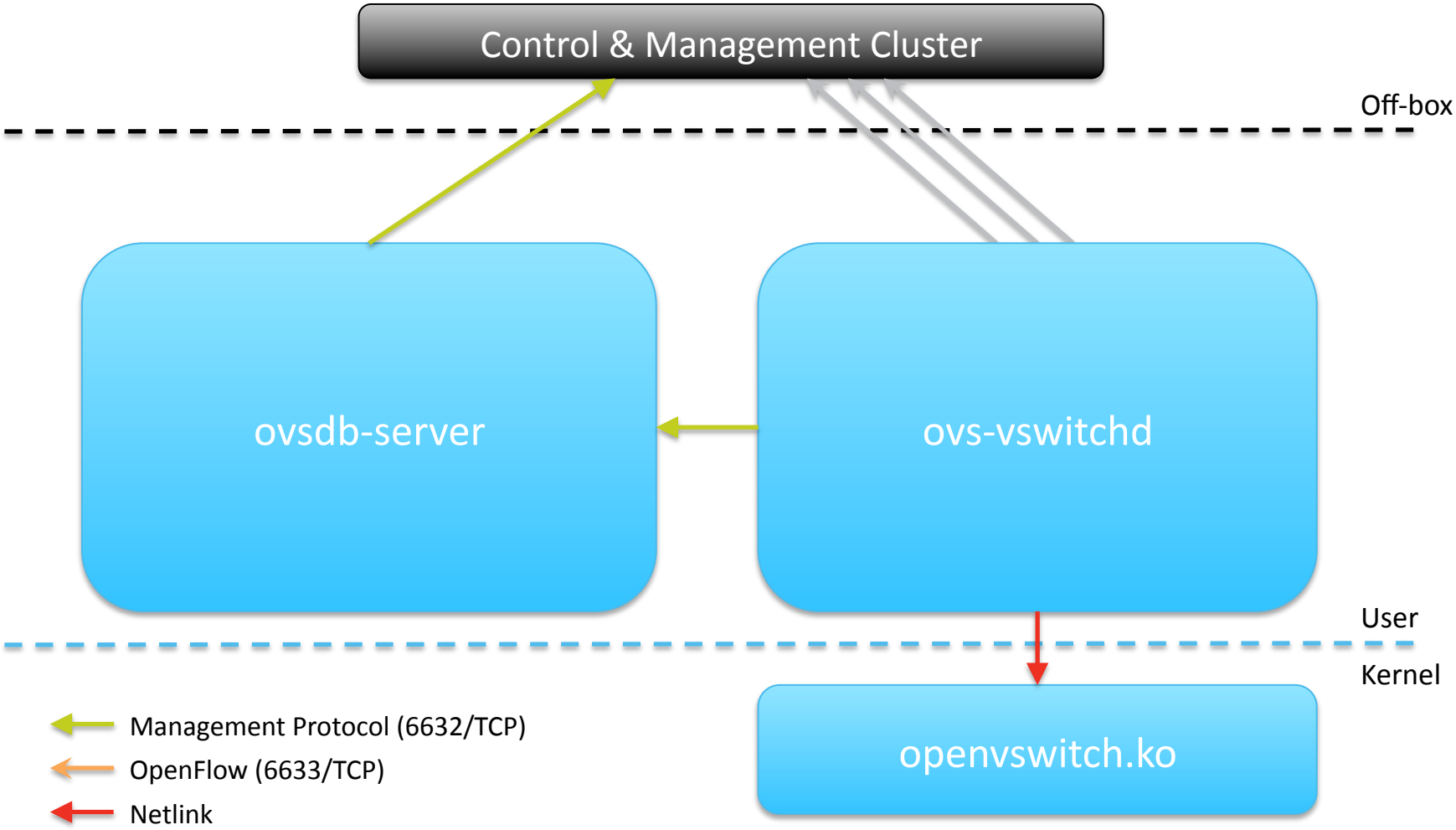
(Partial) List of Contributors

Project

- http://openvswitch.org
- Mailing Lists
  - Announcements: announce@openvswitch.org
  - User-level discussion: discuss@openvswitch.org
  - Dev (code review, etc): dev@openvswitch.org
  - Archives available
- Userspace is Apache licensed
- Kernel is GPLv2
- Source repository:

  git clone git://openvswitch.org/openvswitch

OVS Main Components



Centralized Control

- One OpenFlow connection per datapath
  - Exports idealized view of switch's datapath
    - Lookup based on L2-L4
    - Full wildcarding and priorities
    - Actions: forward, drop, modify, and queue
    - Missed flows go to central controller
- One management channel per system
  - Switch-level configuration
  - Resources
  - Counters

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

openvswitch.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
- ~170 KLOC total in Open vSwitch
- ~12 KLOC in kernel

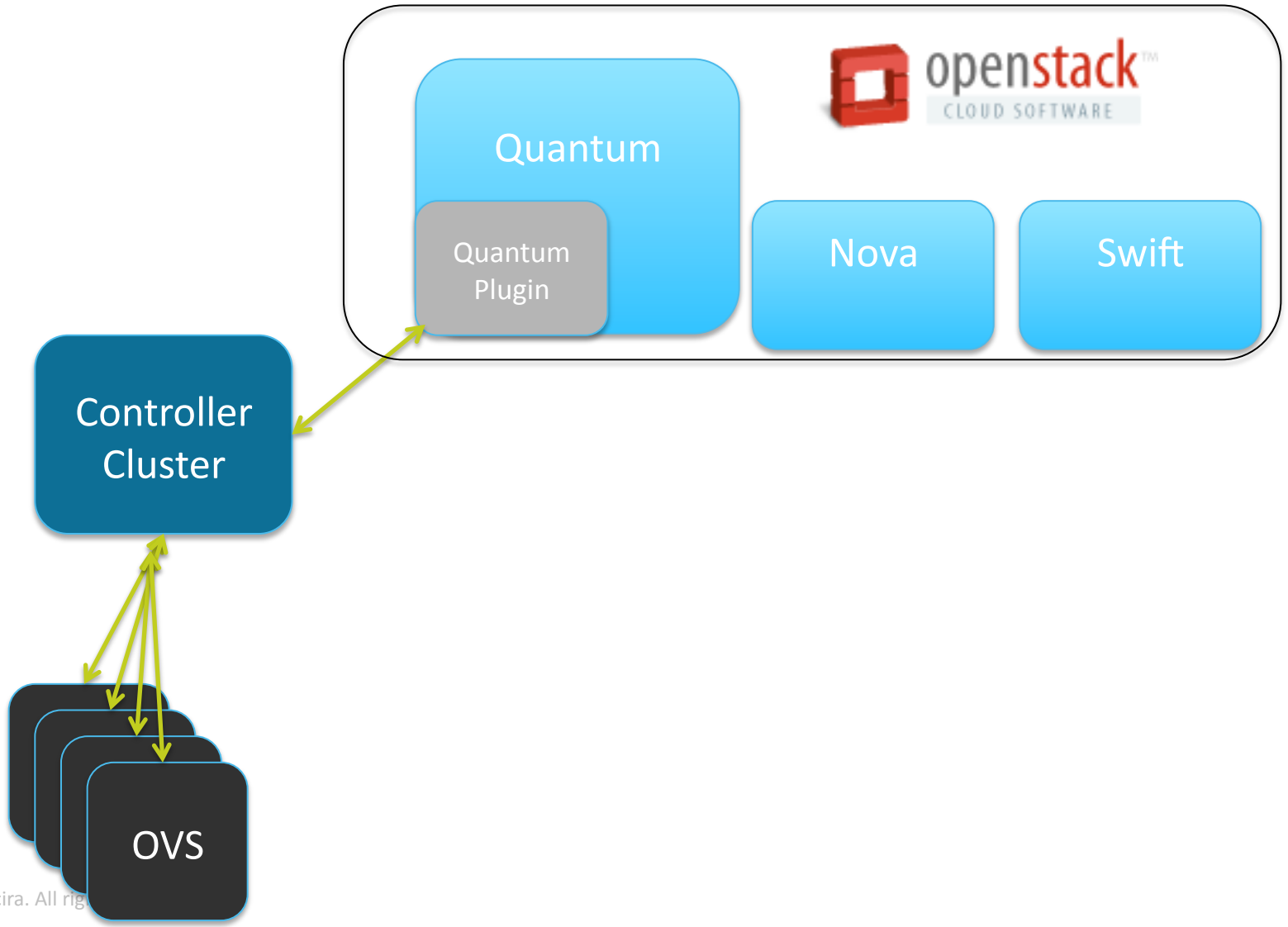
ovsdb-server

- 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

Packaging

- Default networking stack for open source Xen Cloud Platform (XCP)
- Default networking stack for Citrix XenServer and basis for their Distributed Virtual Switch (DVS)
- Distribution packaging
  - Debian
  - Ubuntu
  - SUSE
  - Fedora
  - Red Hat
- Kernel module part of Linux 3.3
- Natively supported by libvirt 0.9.11

The Bigger Picture



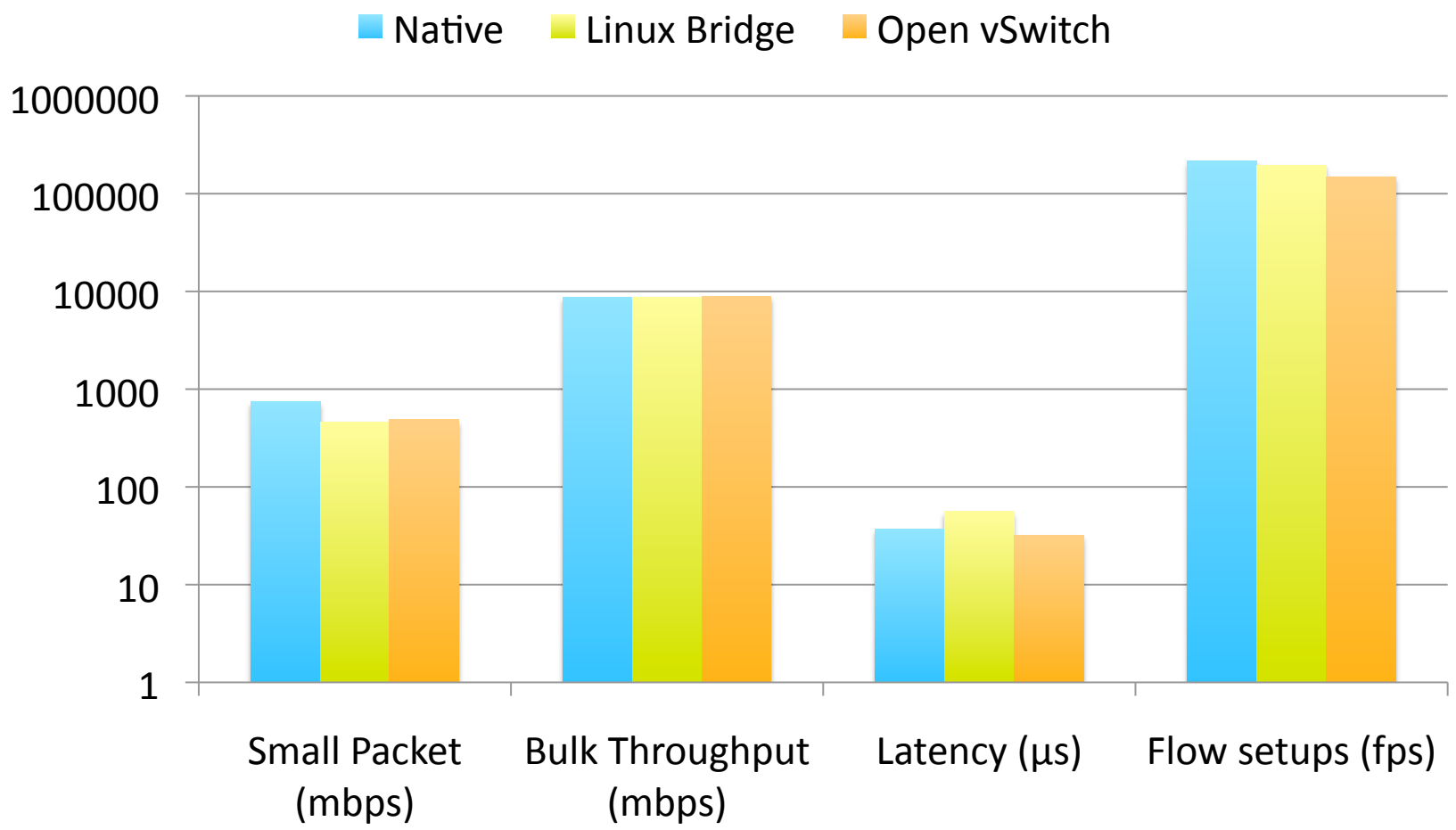
OVS and OpenFlow

- OVS has been the de facto reference implementation of OpenFlow for some time
- OVS has full OpenFlow 1.0 Support, plus
  - Nicira Extensible Match (basis of OXM in 1.2)
  - Resubmit
  - Support for multiple controllers (basis for 1.2 design)
  - Other extensions
- Support for later versions planned
  - http://openvswitch.org/development/openflow-1-x-plan/

Demo

- Simple L2 learning switch using OpenFlow
- All switches (including OVS) can do this autonomously
- Programmability makes it a building block

Performance



Tunneling

- Networking is a distributed system
- Different components of the system need to interconnect and share state
- IP has won as the fast and cheap backbone of choice
- Tunneling is the mechanism to give the power of Open vSwitch to the entire network
- Currently working on expanding the capabilities of OVS tunnels and integrating with upstream

Speaking of tunnels…

- You may have heard of STT
  - draft-davie-stt-01.txt
- Motivation:
  - Hypervisor-originated tunnels needed
  - Software performance of tunnels typically lags native I/O performance due to loss of NIC support for TCP Segmentation Offload (TSO) and other offloads
  - Remember the order of magnitude difference between large and small packets?
  - STT retains TSO by faking out the NIC

Opportunities

- Keeping OVS up with OpenFlow specs (1.1, 1.2 and beyond)
- Your favorite networking functionality
  - MPLS, private VLANs, etc.
- Port to your favorite hardware
  - This has been done for some HW
  - Can leverage HW forwarding paths more sophisticated than the kernel module (e.g. TCAM)


nicira