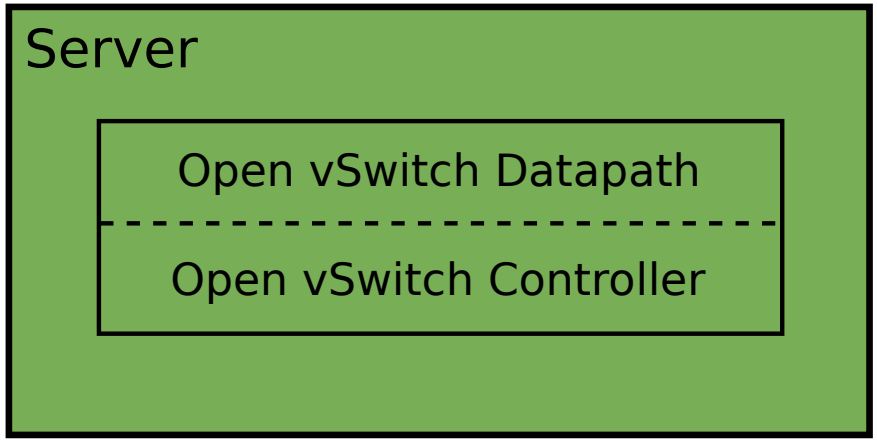
An Introduction to Open vSwitch

Netfilter Workshop, Seville, Spain

Simon Horman <simon@horms.net>
Horms Solutions Ltd., Tokyo

October 2010

Open vSwitch



- Flexibility for Networking in Virtualised Environments
- Flexible Controller in User-Space
- Fast Datapath in Kernel

Open vSwitch Availability

- Available from openvswitch.org
- Development code is available in git
- Announce, discussion and development mailing lists
- User-space (controller and tools) is under the Apache license
- Kernel (datapath) is under the GPLv2
- Shared headers are dual-licensed

Open vSwitch Concepts

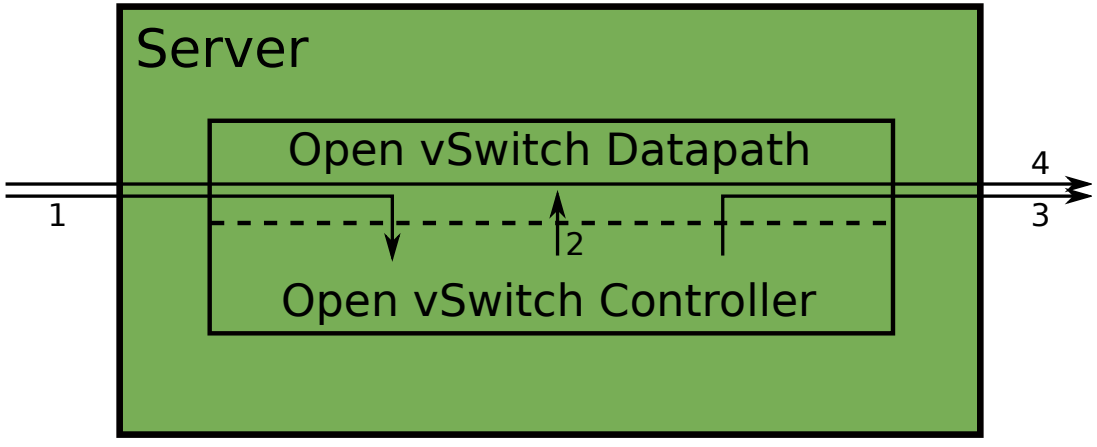
- A switch contains ports
- A port may have one or more interfaces
  - Bonding allows more than once interface per port
- Packets are forward by flow

Packets are Managed as Flows

- A flow may be identified by any combination of
  - Input port
  - VLAN ID (802.1Q)
  - Ethernet Source MAC address
  - Ethernet Destination MAC address
  - IP Source MAC address
  - IP Destination MAC address
  - TCP/UDP/... Source Port
  - TCP/UDP/... Destination Port

Packets are Managed as Flows

1. The first packet of a flow is sent to the controller
2. The controller programs the datapath's actions for a flow
   - Usually one, but may be a list
   - Actions include:
     - Forward to a port or ports, mirror
     - Encapsulate and forward to controller
     - Drop
3. And returns the packet to the datapath
4. Subsequent packets are handled directly by the datapath



Network Scalability Problems in Virtualised Environments

- Migration
- VLANs
- QoS
- Management

Migration

- KVM and Xen provide Live Migration
- With bridging, IP address migration must occur with in the same L2 network
- Open vSwitch avoids this problem using GRE tunnels

VLANs

- Per-Customer VLANs are desirable for security reasons
- But there is a limit of 4094 VLANs

More VLANs

Two, apparently competing, approaches

1. IETF / Cisco
   - RFC5517 — Private VLANs
2. IEEE
   - 802.1ad — Provider Bridges (Q-in-Q)
   - 802.1ah — Provider Backbone Brides (MAC-in-MAC)

RFC5517 — Private VLANs

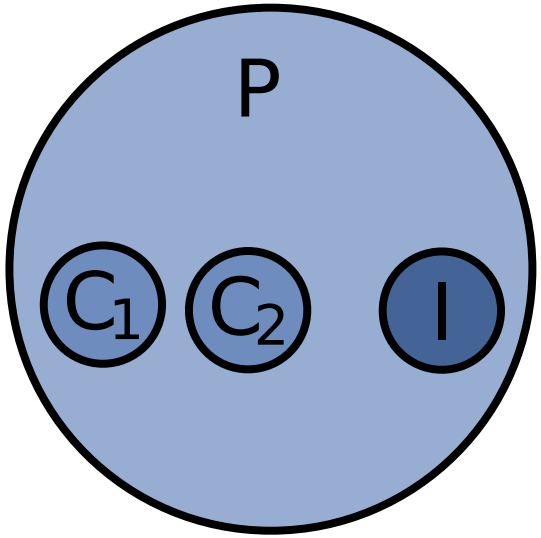
- Uses existing 802.1Q framing
  - Simple to implement (in software/firmware)
- Makes use of pairs of VIDs
  - Requires all switches to support of Private VLANs otherwise switch tables may not merge
- Provides L2 broadcast isolation
  - Forwarding may occur at L3
  - Requires the router to perform proxy ARP
- Currently not supported by Open vSwitch

RFC5517 — Private VLANs

Three VLAN classifications

- Promiscuous
  - May communicate with endpoints on any port
  - e.g.: Gateway, Management Host
- Community
  - May only communicate with endpoints on promiscuous ports or ports belonging to the same comunity
  - e.g.: Different hosts belonging to the same customer
- Isolated
  - May only communicate with endpoints on promiscuous ports
  - e.g.: Hosts that only require access to the gateway

Private VLANs — Domain View



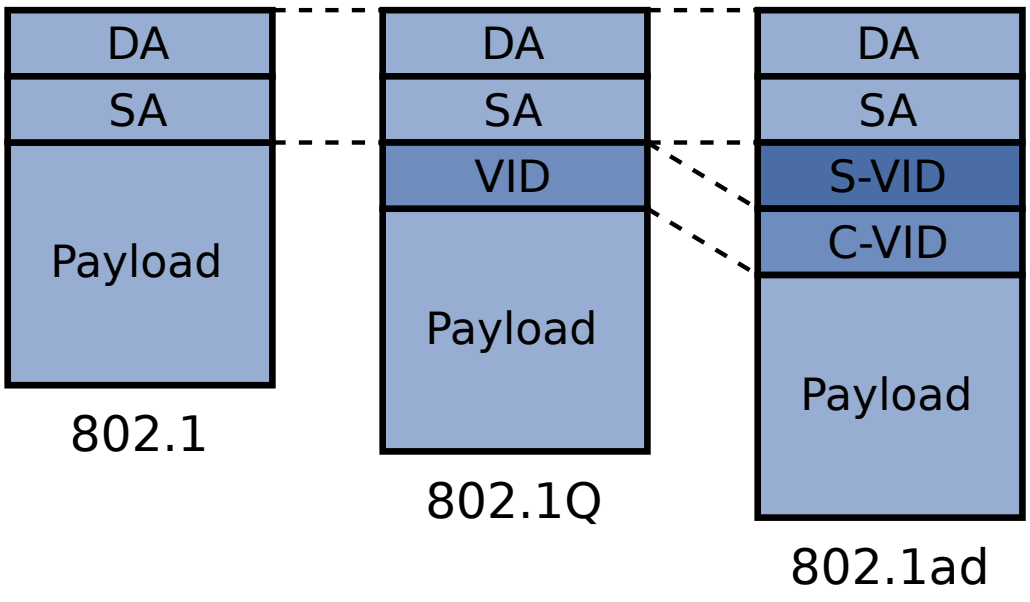
- Promiscous domain (P)
  - May communicate with endpoints in the same domain and sub-domains
- Two community sub-domains (C_1, C_2)
  - May communicate with endpoints in the same domain and parent-domain
- Isolated sub-domain (I)
  - May communicate with endpoints in the parent domain
  - May *not* communicate with endpoints in the same domain

802.1ad — Provider Bridges (Q-in-Q)

- Current standard is 802.1ad-2005, Approved December 2005
- Builds on 802.1Q
- New Framing
  - C-VID (inner)
    - Renamed 802.1Q VID
    - There may be more than one C-VID (inner-inner, ...)
  - S-VID (outer)
    - Different ether-type to C-VID
    - May be translated
- Currently not supported by Linux Kernel / Open vSwitch

802.1ad Framing — Provider Bridges

DA	Destination MAC address
SA	Source MAC addresss
S-VID	Service VLAN ID
C-VID	Customer VLAN ID
VID	VLAN ID

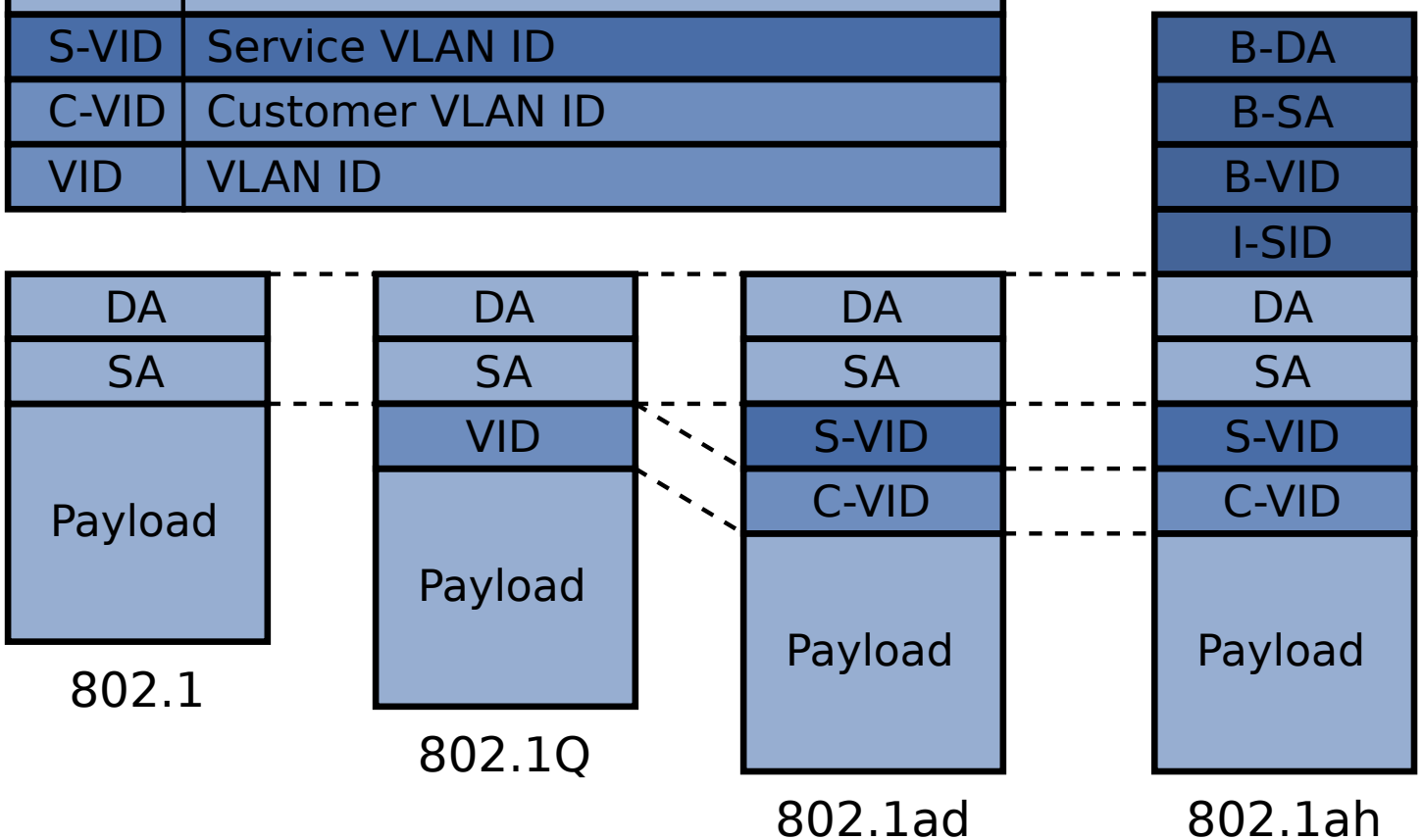


802.1ah — Provider Backbone Bridges (MAC-in-MAC)

- Current standard is 802.1ah-2008, Approved August 2008
- Builds on 802.1ad
- New Framing
  - MAC encapsulation provides full Client VLAN isolation
    - Inner MAC is unknown outside of its scope
  - I-SID: Up to $2^{24} \approx 16$million backbone services
  - I-VID semantics are the same as the S-VLAN
    - Only edge switches need to be Provider Backbone Bridge aware
    - Core switches need only be Provider Bridge (802.1ad) aware
- Currently not supported by Linux Kernel / Open vSwitch

802.1ah Framing — Provider Backbone Bridges

B-DA	Backbone Destination MAC address
B-SA	Backbone Source MAC address
B-VID	Backbone VLAN ID
I-SID	Service ID
DA	Destination MAC address
SA	Source MAC addresss
S-VID	Service VLAN ID
C-VID	Customer VLAN ID
VID	VLAN ID



QoS

Open vSwitch QoS capabilities

1. Interface rate limiting
2. Port QoS policy

QoS: Interface rate limiting

- A rate and burst can be assigned to an Interface
- Conceptually similar to Xen's netback credit scheduler

```
# ovs-vsctl set Interface tap0 ingress_policing_rate=100000
# ovs-vsctl set Interface tap0 ingress_policing_burst=10000
```

- Simple
- Appears to work as expected

QoS: No interface rate limiting example

```
# netperf -4 -t UDP_STREAM -H 172.17.50.253 -- -m 8972
UDP UNIDIRECTIONAL SEND TEST from 0.0.0.0 (0.0.0.0) port 0 AF_
to
+172.17.50.253 (172.17.50.253) port 0 AF_INET
Socket  Message  Elapsed      Messages
Size    Size     Time         Okay Errors   Throughput
bytes   bytes    secs            #      #   10^6bits/sec

120832    8972   10.01       146797      0     1052.60
109568           10.01       146620            1051.33
```

- tap networking used
- jumbo frames required to reach line speed (≈210Mbits/s with 1500 byte frames)
- virtio should do better?

QoS: Interface rate limiting example

```
# netperf -4 -t UDP_STREAM -H 172.17.50.253
UDP UNIDIRECTIONAL SEND TEST from 0.0.0.0 (0.0.0.0) port 0 AF_
to 172.17.50.253 (172.17.50.253) port 0 AF_INET
Socket  Message  Elapsed      Messages
Size    Size     Time         Okay Errors   Throughput
bytes   bytes    secs            #      #   10^6bits/sec

120832    8972   10.01       149735      0    1073.66
109568           10.01        14684           105.29
```

- Difference in sent and received packets indicates a flow control problem.
- virtio should do better?

QoS: Port QoS policy

- A port may be assigned one ore more QoS policy
- Each QoS policy consists of a class and a qdisc
  - Classes and qdisc use the Linux kernel's tc implementation
  - Only HTB classes are supported at this time
  - Each class has a single qdisc associated with it
  - The class of a flow is chosen by the controller
- The QoS policy (i.e. class) of a flow is chosen by the controller

QoS: Port QoS policy example

Programming the Datapath

```
1:# ovs-vsctl set port eth1 qos=@newqos \
2:  -- --id=@newqos create qos type=linux-htb \
3:       other-config:max-rate=200000000 queues=0=@q0,1=@q1 \
4:  -- --id=@q0 create queue \
5:       other-config:min-rate=100000000 \
6:       other-config:max-rate=100000000 \
7:  -- --id=@q1 create queue \
8:       other-config:min-rate=50000000 \
9:       other-config:max-rate=50000000
```

QoS: Port QoS policy example

Hard-coding the controller

```
# ovs-ofctl add-flow br0 "in_port=2 ip nw_dst=172.17.50.253 \
    idle_timeout=0 actions=enqueue:1:0"
# ovs-ofctl add-flow br0 "in_port=3 ip nw_dst=172.17.50.253 \
    idle_timeout=0 actions=enqueue:1:1"
```

- Only suitable for testing

QoS: Port QoS policy example

```
Guest 0:
# netperf -4 -t TCP_STREAM -H 172.17.50.253 -l 30 -- -m 8972
TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to
172.17.50.253 (172.17.50.253) port 0 AF_INET
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec
87380  16384   8972    30.01      99.12

Guest 1:
# netperf -4 -t TCP_STREAM -H 172.17.50.253 -l 30 -- -m 8972
...
87380  16384   8972    30.14      49.56
```

QoS: Port QoS policy controller improvements

- Add a default queue to the Port table
- Add enqueue to the FLOOD and NORMAL ports
- or use NOX (a different controller)

Conclusion

- Open vSwitch is aimed at addressing short-comings in using bridging in virtualised environments
- It is a young project and there is much scope to contribute
  - Extended VLAN support
    - Private VLANs
    - 802.1ad
    - 802.1ah
  - Improved QoS
    - Add a default queue to the Port table
    - Add enqueue to the FLOOD and NORMAL ports
    - or use NOX (a different controller)
  - High-Level Management