Originally published in ACM/IEEE Symposium on Architecture for Networking and Communication Systems (ANCS'15) held May 7-8 at Oakland, CA, USA.

Distributed Control Plane For High Performance Switch-based VXLAN Overlays

Sunay Tripathi
Pluribus Networks, Inc.
2455 Faber St
Palo Alto. CA. 94303
+1-650-618-5490
sunay@pluribusnetworks.com

Roger Chickering
Pluribus Networks, Inc.
2455 Faber St
Palo Alto. CA. 94303
+1-650-618-5490
rogerc@pluribusnetworks.com

Jonathan Gainsley
Pluribus Networks, Inc.
2455 Faber St
Palo Alto. CA. 94303
+1-650-618-5490
jon@pluribusnetworks.com

Abstract

In this paper, we describe a distributed network operating system that runs on merchant silicon based switches. Multiple switches can form a fabric for orchestration and state sharing purposes independent of the network topology. Each switch in a fabric shares its state with other switches over TCP/IP and keeps a global view of virtual ports along with topology to make individual switching, forwarding and encapsulation decisions. This allows the distributed switch OS to track Virtual Machines as they migrate and dynamically orchestrate VXLAN overlays without needing software overlays in servers. Since the Switch OS offloads encapsulation/decapsulation for VXLAN to the switch ASIC, there is no performance penalty.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Network Operating Systems

General Terms

Performance, Design, Experimentation

Keywords

VXLAN, Overlay, Distributed Network OS, Netvisor.

1. Introduction

To overcome the limitations of the layer 2 networking, overlay networks moved networking functionality into server hypervisors. The server initiated overlay introduces the overhead of encapsulating and decapsulating overlay traffic to the server. In this paper, we describe a distributed network operating system for running fabrics of switches based on the latest hardware designs incorporating merchant silicon based switch chips. The programmability and global state sharing allows us to take advantage of the switch hardware tables and offload the overlays to switch TCAM and treat the physical and virtual networks as one logical network.

2. Distributed Switch OS and Open Hardware Platform

The authors have worked with a team to implement Netvisor, a distributed network operating system. Our experience running Netvisor with the current generation of switch chips from Broadcom and Intel and a range of CPU chips from Intel have shown that modern merchant silicon hardware is a good platform for providing server-like programmability and offload capability to switch ASIC via Netvisor's vflow API.

Figure 1 compares the current generation of Open Switch Platforms[2] with a traditional switch where the operating system used to runs on a low powered processor and low speed busses.

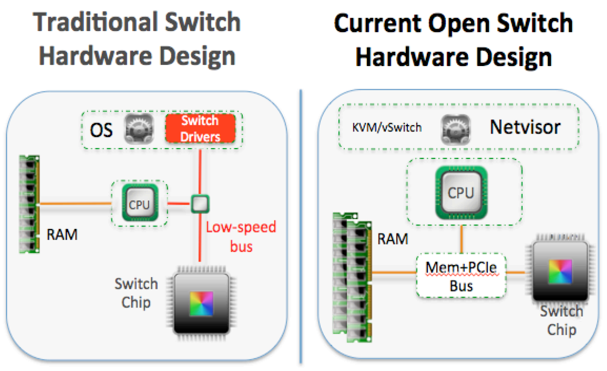

Figure 1: Open Switch Compared to Traditional Switch

The switch chip is integrated into the server operating system over PCI-Express. The switch register space is memory mapped into software, which manages the MAC, IP and flow tables. The design for control plane I/O is fully multithreaded and similar to Crossbow Virtualization Lanes with a dynamic polling[1] architecture. There is no hardware MAC learning on the switch chip and when there is a MAC table miss in hardware the packet is forwarded to software. The server-like CPU and high bandwidth connection to the switch ASIC in modern switch designs enables the powerful software features described in this paper.

2.1 Fabric and Fabric-wide vport Table

A fabric is a collection of switches that share configuration and state information over long standing TCP connections. Switches in a fabric work together to provision the resources allocated by the configuration and to manage state information across the fabric.

In Netvisor a vport is a data structure that represents an endpoint attached to the network. vports are maintained for servers, virtual machines, switches, and any other device connected to a switch port. Each vport contains the switch and physical port the vport is associated with along with the MAC/IP/VXLAN information for the virtual or physical machine. The vport also contains metadata related to state, policy and usage. Vports are stored in shared state tables. Each switch in the fabric stores vports for all the switches in the fabric, and state changes to vports are propagated through the fabric using the fabric protocol. The switch chip's hardware MAC table caches the vports that are active on each switch.

3. Switch Implementation of VXLAN Overlay

Netvisor instances can have a VXLAN tunnel between then over BGP/ECMP layer 3 connectivity. The Netvisor Fabric synchronizes all vport state across all nodes in the fabric and dynamically creates encapsulation rules to enable endpoints to communicate. When the switch chip receives a packet from the network, it expects either a encapsulation/decapsulation flow or a entry in Layer 2/3 table which allows it to forward the packet. If neither is present, the packet is sent to the CPU where the global vport table is consulted to create a new rule in hardware or do endpoint discovery by sending software-encapsulated packets to each switch.

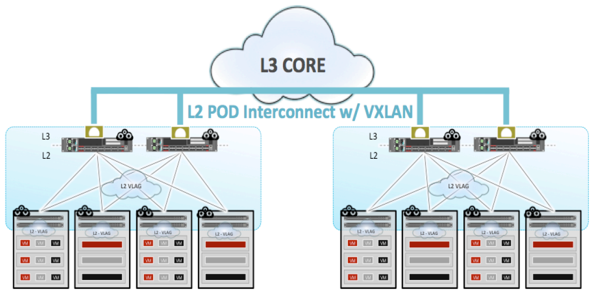

Figure 2: Fabric extends over VXLAN Tunnel

Figure 2 shows an example topology where two Layer 2 pods are connected via a VXLAN tunnel and all the switches form a single Netvisor Fabric enabling L2 connectivity between all connected endpoints.

4. Server Overlay vs. Switch Overlay Performance

Figure 3 compares server-to-server throughput without overlay, server based overlay, and switch based overlay. The OS used was Ubuntu 4.14.

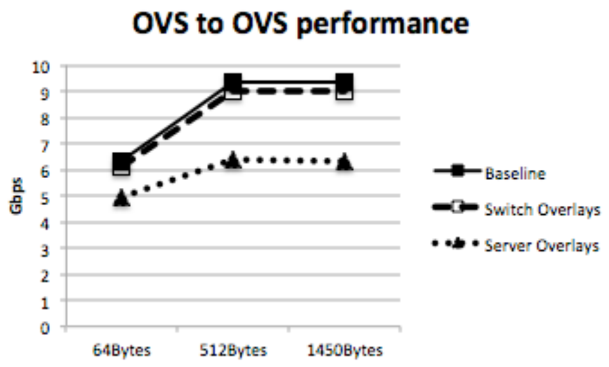

Figure 3: OVS to OVS performance for different packet sizes

As seen from the graph, the switch based overlay, which adds no CPU overhead to the servers, achieves the same performance as the non-overlay experiment while the server based overlay has a high performance penalty. The penalty is especially severe in case of small packet sizes (64 bytes). Koponen[3] showed similar findings with GRE based tunnels where CPU utilization more then doubled and throughput dropped from 9.3Gbps to 2.4Gbps. Figure 4 shows the bandwidth achieved when sending different sized packet streams from a VM to another VM.

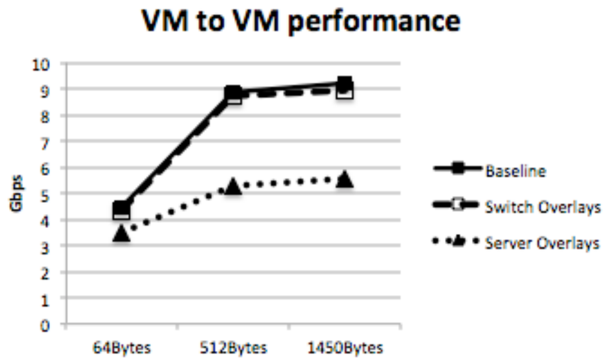

Figure 4: VM to VM Performance for different packet sizes

As evident from the graph, the server overlays added significant performance penalty for throughput.

5. Conclusion

The distributed switch operating system architecture presented in this paper introduces a novel approach to make switches more programmable and realize the vision of Software Defined Networking. While VXLAN overlays may be implemented entirely in servers, an alternative is to implement VXLAN overlays in the switch hardware using a distributed CPU control plane.

6. Acknowledgement

The authors acknowledge and thank the entire Pluribus Engineering team.

7. References

[1] S. Tripathi, N. Droux, T. Srinivasan, K. Belgaied. Crossbow: From H/W virtualized NICs to virtualized networks. Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures. VISA 2009, 53-62.

[2] Open Compute Project Reference Designs. DOI=http://www.opencompute.org/wiki/Networking/SpecsAndDesigns.

[3] T. Koponen, K. Amidon, et. al. Network Virtualization in Multi-tenant Datacenters. 2014. In USENIX NSDI 2014