Proactive Overlay versus Reactive Hop-by-Hop

Juniper’s motivations for the Contrail Architecture explained

Table of Contents

Executive Summary

The network is a strategic asset and can be used in new and interesting ways by service provider and enterprise users. But legacy networks are too complex to manage, too inflexible to adapt to the needs of dynamic applications, and too closed to allow innovation. Software-defined networking (SDN) enables businesses to have more dynamic, fine-grained control over the behavior and performance of their networking Infrastructure.

This white paper describes the architectural underpinning of the Juniper Networks® Contrail family of products, namely *proactive overlay* virtual networks. We explain why we chose this architecture and compare it to the *reactive end-to-end architecture*. We also describe the control plane protocols and the data plane protocols used by Contrail and our motivation for choosing those specific protocols.

Introduction

Operators and enterprises are looking at software-defined networking (SDN) to provide more simplicity, manageability, flexibility, and agility for their networks.

SDN simplifies the following use cases in the data center:

- Multi-tenancy – also known as network slicing
- Gateways – the ability to connect a virtual network to a physical network (such as the Internet or an enterprise VPN) and/or to non-virtualized servers
- Service chaining (also known as Network Function Virtualization)—the ability to flexibly deploy virtualized services such as firewalls, load balancers, or caches in virtual networks
- Traffic engineering and bandwidth calendaring in the WAN—for optimized Data Center Interconnect (DCI)

Earlier this year, Juniper **announced its high-level SDN strategy**, and we described the six principles of SDN as follows:

1. Separate networking into four planes—forwarding, control, services, and management
2. Centralize management, control, and services
3. Use the cloud for elastic scale and flexible deployment, enabling usage-based pricing
4. Use common platforms for network and security applications and management integration
5. Use standard protocols for interoperability across vendors
6. Broadly apply SDN across the networking domains

We also described four concrete steps towards implementing SDN:

1. Centralize management
2. Extract services
3. Centralize the controller
4. Optimize the hardware

Since announcing our overall SDN strategy, Juniper mapped some of these steps to concrete products:

- Juniper Networks Junos® Space for centralizing management
- Juniper Networks JunosV App Engine as well as a broad suite of virtualized services such as JunosV Firefly for extracting services
- Juniper Networks EX9200 Ethernet Switch as the most programmable SDN switch

SDN Controller and OpenFlow

With respect to centralizing the controller, Juniper is following a two-pronged approach.

Our Contrail acquisition will bring a range of technologies including an SDN controller. While Juniper publicly announced the Contrail family of products quite recently, we did already publish **IETF drafts** which describe the overlay technology on which the product will be based.

In parallel, we are also implementing OpenFlow across our router and switch product line. Our OpenFlow implementation is interoperable with the major open source SDN controllers, including the Open Daylight controller. Juniper has joined the Open Daylight consortium as a platinum founding member and is contributing Contrail technology to the project.

When we describe our two-pronged overlay and OpenFlow approach to customers, we are often asked: Which is the right approach, overlays or OpenFlow? It turns out that this is a false dichotomy, as overlays and OpenFlow are not mutually exclusive.

OpenFlow

There are multiple ways in which the OpenFlow protocol can be used. Some approaches are reasonable, while others are not.

The original approach to using OpenFlow is what we refer to as reactive hop-by-hop OpenFlow. Here, every first packet of every flow is punted to a centralized controller, which decides the hop-by-hop path for the flow and programs the flow into every switch on the path. This is not a scalable approach.

Another more recent approach to using OpenFlow is to use it in combination with overlays. In this case, OF-Config is used to create overlay tunnels and OpenFlow is used to proactively install flows to steer packets into the tunnels. This is a reasonable way of using OpenFlow, although there are other approaches as well (e.g., XMPP) which are arguably even better because they work at a higher level of abstraction.

Rather than asking which of the two is better—overlays or OpenFlow—a better way to frame the problem is to ask two separate but related questions. First, in the data plane do we use overlays on top of the IP fabric or do we program end-to-end flows into the IP fabric to virtualize the network? Second, which control plane protocol do we use to virtualize the network?

Network Virtualization

Let's start with the first question. In the data plane, how is the network virtualized; in other words, how is the network sliced into multiple virtual networks? Do we use "hop-by-hop" virtualization or do we use an overlay?

VLANs are an example of hop-by-hop virtualization. In this approach, there is one VLAN per tenant and each switch in the network must be aware of the VLANs and the media access control (MAC) addresses of multiple tenants. The problems with this approach are numerous and well known: explosion of the forwarding state on the aggregation switches; the necessity to reconfigure physical switches every time a tenant or virtual machine is added; the complexity and slowness of VLAN configuration; the complexity of VLAN pruning to avoid unnecessary flooding, stability, and convergence problems of Layer 2 protocols, etc.

Hop-by-hop OpenFlow is another example of hop-by-hop virtualization. This is the approach where a centralized controller creates a flow path from one edge of the network to the other edge of the network using OpenFlow to install a flow *on each switch in the path*, including the aggregation or core switches. Typically, this approach uses a "reactive controller" where the first packet of each new flow is sent to a centralized SDN controller that applies policy, computes the path, and uses OpenFlow to install a flow into each switch on the path.

Unfortunately, the hop-by-hop OpenFlow approach suffers from most of the same problems as VLANs and introduces some new ones as well:

- It creates an explosion of forwarding state in the flow tables on the physical switches, possibly even worse than the VLAN approach if fine-grained flow matching is used.
- It is necessity for the SDN controller to "touch" each switch in the path every time a new flow needs to be programmed, i.e., each time a tenant or virtual machine is added to the network.
- The reactive model introduces extra latency and new failure modes.
- Since OpenFlow runs over TCP, an out-of-band control network is needed. OpenFlow cannot be used to bring up the control network because that would create a chicken-and-egg problem.

There are very few companies that still support the reactive hop-by-hop OpenFlow model. The model is not scalable, and it harkens back to the old days of process switching, which is not used by any modern switch anymore.

Overlays

In the early days of SDN, there were some companies using the reactive hop-by-hop approach but most of them have since **evolved their position**. They now also support tunnels to create overlays, support the proactive model in addition to the reactive model, and use a combination of fine-grained flows in the virtual edge with coarse-grained flows in the physical core to avoid scaling limitations.

The overlay approach, by contrast, uses overlay tunnels to virtualize or "slice" the network. These tunnels generally terminate in virtual switches or virtual routers in hypervisors, but they can also terminate in physical routers or switches for "gateway" use cases.

The overlay approach was pioneered by software vendors such as VMware (Nicera) and Microsoft. But it is now widely supported by major networking vendors as well.

The overlay approach has many advantages, including:

- It greatly reduces the size of the forwarding or flow tables in the physical underlay switches. They only need to contain addresses of physical servers and not addresses of virtual machines.
- Adding a new tenant, adding a new virtual machine, or applying a new policy only involves touching the edge switches, i.e., the virtual switches or virtual routers in the hypervisors. It does not involve touching the physical switches in the underlay in any way. This makes the physical network more stable—if you don't touch it, you don't break it.
- It provides a seamless migration path for introducing SDN into existing networks without disrupting existing services provided on those networks. No forklift upgrades are needed.

Reactive hop-by-hop networks and proactive overlay networks are two extremes of a continuous spectrum. Between these two extremes, there are intermediate steps. The "tricks" to make hop-by-hop reactive networks more scalable (using coarser grained flows in the aggregation switches, using a proactive model, and using tunnels) are simply a gradual transition from the reactive hop-by-hop model to the proactive overlay model.

Of course, there is no free lunch—overlays also introduce some new complications. Because every packet is encapsulated into a tunnel, it is more difficult to troubleshoot the network and to provide per-tenant quality of service (QoS) in the underlay network.

The Juniper Solution

Trio ASICs (used in the Juniper Networks MX Series 3D Universal Edge Routers) and One ASICs (used in the newly announced EX9200 Ethernet aggregation switch) are uniquely positioned to solve these challenges because of their flexible and programmable microcode architecture. Their ability to look deep into the encapsulated packets and extract the virtual network identifier allows them to maintain per-tenant statistics which aid in troubleshooting and debugging. Their ability to do fine-grained queuing allows them to provide per-tenant QoS, which helps to isolate tenants from each other, if needed. The micro programmable architecture also allows Juniper to support new data plane protocols without respinning the ASICs, which provides future proofing in the still developing area of SDN.

Furthermore, Juniper is building the virtual overlay (including service chaining) and the physical underlay in such a way that 1 + 1 will add up to more than 2. The virtual overlay will be aware of the physical underlay and vice versa. Some examples of the integration between virtual and physical world include:

- Flow-through provisioning of the gateway functions, for example in EX Series Switches, QFabric™ Family of Products, and MX Series routers where the virtual network meets the physical network
- Flow-through provisioning of service chaining, including the steering of traffic into the right service chains on the virtual and physical service appliances
- Tenant awareness in the underlay for troubleshooting and QoS
- Efficient and scalable solutions for dealing with broadcast and multicast traffic in the overlay without requiring multicast in the underlay

There are multiple data plane protocols which can be used to create overlay tunnels, including MPLS over generic routing encapsulation (GRE), VXLAN, STT, NV-GRE, and others. Juniper has chosen to support both MPLS over GRE and VXLAN, with the latter chosen because it is evolving as the de facto interoperability standard for overlay networking in the data center and because of its support for multipathing. MPLS over GRE is also supported because it supports L3 overlays as well as L2 overlays, and because it makes interworking with existing service provider networks very easy.

Control Plane Protocols

We now get to the second question which is: Assuming we use overlays, which control plane protocol do we use to program the networking devices at the edge of the overlay (which may be physical or virtual routers or switches)?

In the early days of VXLAN overlay networking, a "flood-and-learn" approach was proposed, which is now widely discredited. Most vendors are moving towards an approach where the forwarding state is installed on the virtual routers and switches using a "real" control plane protocol.

Generally speaking, there are two approaches for overlay control plane protocols—using OpenFlow or using some flavor of a control plane protocol loosely similar to MPLS VPNs.

Note that here we are using the *OpenFlow protocol*, but not the *hop-by-hop* OpenFlow model discussed previously. Here we are speaking of using the OpenFlow protocol at the edges of the network, using OF-Config to create overlay tunnels, and using OpenFlow proper to steer flows of traffic into the tunnels. OpenFlow does not touch the core or aggregation switches in this case.

OpenFlow as a Control Plane Protocol for proactive overlay networks

OpenFlow is a perfectly feasible protocol for proactive overlay networks, but in our opinion it is not the ideal protocol. The problem with using OpenFlow as a control plane protocol is its low level of abstraction—it essentially provides a way to program ternary content addressable memory (TCAM) entries directly into the hardware. The more advanced features of OpenFlow (multi-table lookups, groups, etc.) impose specific restrictions on how the hardware must work. This makes it very difficult to implement the complete OpenFlow protocol in hardware. And this, in turn, leads to different vendors implementing different subsets of OpenFlow, making it quite challenging to implement a multivendor OpenFlow controller.

MPLS VPN-Like Control Plane Protocols for proactive overlay networks

In the other approach, control plane protocols conceptually similar to the ones used in MPLS VPNs are used to create tunnels and to install forwarding state. Either MP-BGP is used directly, or some other protocol such as XMPP or OVS-DB is used to perform functions similar to BGP (such as exchanging routes and exchanging VPN membership information), but also more (such as creating routing instances, collecting analytics, etc.) These control plane protocols deal with objects at a much higher level of abstraction such as routing instances and routes (as opposed to OpenFlow, which deals with very low levels of abstraction, namely flows).

Using control plane protocols at a higher level of abstraction such as XMPP or OVS-DB, either of which is conceptually similar to MPLS VPNs, has numerous advantages, particularly in service provider environments:

1. MPLS VPN is a mature, proven, scalable, and stable technology already deployed in countless very large service provider and enterprise networks.
2. MPLS VPN-based overlays can interoperate seamlessly and trivially with physical networks such as the Internet and L3VPNs.
3. MPLS VPNs support both L2 and L3 overlays.
4. This approach provides a seamless migration path for introducing SDN into existing networks without disrupting the existing services provided on those networks.

Whereas some vendors (such as Nuage/Alcatel-Lucent) have chosen to use OpenFlow as the control plane protocol for overlays, other vendors have chosen protocols at a higher level of abstraction. VMware/Nicera uses OVS-DB, while Juniper uses XMPP.

Both camps generally agree that BGP is the right protocol for federation between controllers and as the control plane protocol for gateway routers.

Even Martin Casado, one of the founding fathers of OpenFlow, has stated that using OpenFlow to control physical switches is the wrong approach and that using overlays in general (and MPLS VPNs in particular) is the right approach.

Conclusion

In summary, we at Juniper believe that proactive overlays are the right approach. Our JunosV Contrail family of products are designed around a proactive overlay architecture.

We believe that the reactive hop-by-hop approach to network virtualization (e.g. using VLANs or hop-by-hop OpenFlow) is not a feasible solution for large-scale production networks.

For our own overlay solution based on the Contrail technology, we have chosen XMPP as the control plane protocol mainly because it provides a higher level of abstraction which allows us to abstract the details of our various hardware platforms (MX Series 3D Universal Edge Routers, PTX Series Packet Transport Switches, M Series Multiservice Edge Routers, T Series Core Routers, EX Series Ethernet Switches, QFabric family of products, SRX Series Services Gateways, etc.) away from the controller.

That said, we recognize that OpenFlow is also a feasible control plane protocol for creating an overlay. Our JunosV Contrail controller supports multiple south-bound protocols, and there is no architectural reason why we would not be able to add OpenFlow as an additional south-bound protocol if the need arises.

In fact, the Contrail "SDN as a compiler" architecture uses data models and transformation engines to provide north-bound interfaces at a high level of abstraction. In that architecture, it is easy to change the south-bound protocol without affecting the north-bound interface in any way and without any repercussions on the applications.

About Juniper Networks

Juniper Networks is in the business of network innovation. From devices to data centers, from consumers to cloud providers, Juniper Networks delivers the software, silicon and systems that transform the experience and economics of networking. The company serves customers and partners worldwide. Additional information can be found at www.juniper.net.

Corporate and Sales Headquarters
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089 USA
Phone: 888.JUNIPER (888.586.4737)
or +1.408.745.2000
Fax: +1.408.745.2100
www.juniper.net

APAC and EMEA Headquarters
Juniper Networks International B.V.
Boeing Avenue 240
1119 PZ Schiphol-Rijk
Amsterdam, The Netherlands
Phone: +31.0.207.125.700
Fax: +31.0.207.125.701