Ten Things to Look for in an SDN Controller

Ashton, Metzler & Associates

Leverage Technology & Talent for Success

Executive Summary

Over the last six months there has been a great growth in the interest that IT organizations have shown in software defined networks (SDN) as well as in the number of SDN related announcements that vendors have made. While a lot has been written about technologies such as OpenFlow[1] that enable an SDN, much less has been written about SDN controllers and what distinguishes one controller from another. Given that SDN controllers are a new class of product and that relatively little has been written about them, many IT organizations will struggle when it comes to evaluating them. In order to assist IT organizations choose the most appropriate product, this white paper discusses the key criteria that IT organizations should use when evaluating an SDN controller.

The majority of the characteristics that are discussed in this white paper are technical. Some of these technical characteristics will be familiar to the reader as they apply to most if not all network elements. This includes scalability, performance, security, centralized monitoring, visualization and reliability. However, some of the technical characteristics are SDN centric and hence will be new to most readers. This includes OpenFlow support, network virtualization and network programmability. In addition to the technical characteristics, this white paper also discusses the importance of IT organizations evaluating the vendors that provide SDN controllers and the white paper suggests how IT organizations can perform this evaluation.

At the end of this white paper is a checklist of the key criteria that IT organizations should use when evaluating an SDN controller.

Introduction

As is somewhat typical of emerging technologies, there isn't a universally agreed to definition of what is meant by software defined networking (SDN). However, there is a rapidly growing consensus that an SDN involves the:

- Replacement of the control plane on the switch with a programmatic interface
- Migration of the control planes of the switches to a controller
- Provision of programmatic interfaces into the controller

There is also a rapidly growing consensus that an SDN must focus not just on Layer 2 and Layer 3 functionality, but also on Layer 4 through Layer 7 functionality and that is must support both physical and virtual equipment.

Although other techniques are possible, the OpenFlow protocol is the most commonly used technique to program the forwarding behavior of the switches that comprise an SDN. The protocol was developed at Stanford, with v1.0 of the protocol published at the end of 2009 and v1.1 at the beginning of 2011. In March of 2011, the Open Networking Foundation (ONF) was created and the intellectual property rights of OpenFlow were transitioned to it. Part of the ONF charter is to control and commercialize OpenFlow. With that goal in mind, the ONF recently released OpenFlow v1.3

[1] https://www.opennetworking.org/standards/intro-to-openflow

and in 2012 the ONF sponsored two *PlugFest* events[2] the goal of each of which was to drive interoperability between OpenFlow enabled devices.

The vision of the ONF is to make open SDNs the new norm for networks. The typical standards organization is comprised of companies who intend to provide the technology that is being standardized. The ONF organizational model is notably different. While many of the over eighty members of the ONF will indeed provide the technology necessary to implement an SDN, the board members of the ONF are:

- Deutsche Telekom
- Facebook
- Goldman Sachs
- Google
- Microsoft
- NTT Communications
- Verizon
- Yahoo!

The companies listed above run some of the largest data centers in the world. They are driving the development of SDN and OpenFlow because they believe that SDN and OpenFlow will provide them with significant benefits. Their role on the ONF also helps to ensure that OpenFlow will be able to meet the needs of even the largest organizations.

While a lot has been written about OpenFlow[3], much less has been written about controllers, which operate at the heart of every SDN, and what distinguishes one controller from another. In order to fill that void, this white paper discusses the key criteria that IT organizations should use when evaluating an SDN controller.

The SDN Controller

As shown in Figure 1, an SDN controller instructs the switches as to what actions they should take via what is commonly called the southbound API. As previously mentioned, OpenFlow is rapidly becoming the dominant way for an SDN controller to communicate with switches. The switches shown in Figure 1 can either be *pure OpenFlow switches*, whereby the only function that they provide is to forward packets, or they can be *OpenFlow-hybrid switches* that support both OpenFlow enabled flow forwarding and traditional Ethernet switch bridging and routing.

[2] http://www.eetimes.com/electronics-news/4398846/Twenty-companies-attend-OpenFlow-plugfest
[3] https://www.opennetworking.org/standards/intro-to-openflow

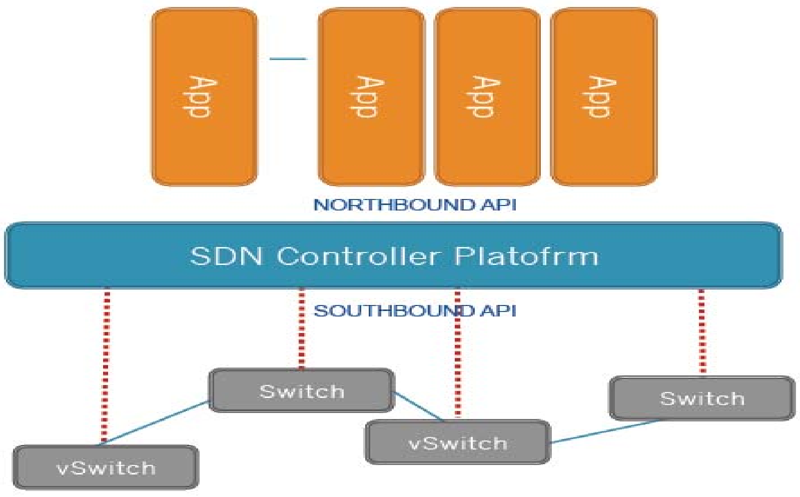

Figure 1: A Layered SDN Architecture

The SDN controller is a new class of data networking product. The ten key characteristics that IT organizations should look for when evaluating an SDN controller are described below.

1. OpenFlow Support

   When a packet arrives at an OpenFlow switch, the header fields are compared to flow table entries. If a match is found, the packet is either forwarded to specified port(s) or dropped depending on one or more actions stored in the flow table. When an OpenFlow switch receives a packet that does not match the flow table entries, it encapsulates the packet and sends it to the controller. The controller then decides how the packet should be handled and notifies the switch to either drop the packet or make a new entry in the flow table to support the new flow.

   Figure 2 shows the 12-tuple of header fields that is used to match flows in the flow table.

Ingress Port	Ether Source	Ether Dest	Ether Type	VLAN ID	VLAN Prior	IP Source	IP Dest	IP Proto	IP TOS	Source Port	Dest Port

Figure 2: The OpenFlow V1.0 Flow Table Fields

   According to the ONF, the majority of its members have implemented v1.0 of OpenFlow. The OpenFlow v1.0 standard contains a number of optional actions such as the ability to change the content of the header fields or to flood the packet along the spanning tree. The ONF has also stated that many of its members are focusing their development efforts on implementing OpenFlow v1.3 that was ratified in June 2012.

   Some of the basic functionality that any OpenFlow controller should be able to support includes:

   - Match on any field in the 12 tuple as shown in figure 2.
   - Support for OpenFlow actions such as header rewriting.
   - Discovering the network using the Link Layer Discovery Protocol (LLDP).

When choosing an OpenFlow controller IT organizations need to understand the OpenFlow functionality that the controller currently supports as well as the vendor's roadmap to implement newer versions of OpenFlow, such as v1.3. One reason for needing to understand this roadmap is that important functionality such as IPv6 support is not part of OpenFlow v1.0 but is part of the OpenFlow v1.3 standard. IT organizations also need to understand the vendor's support for the many optional features in the OpenFlow protocol as well as for vendor extensions to the protocol.

2. Network Virtualization

One of the most important benefits of an SDN is network virtualization. Network virtualization is nothing new. One type of network virtualization that has been in production networks for decades is a virtual LAN (VLAN). VLANs partition an Ethernet network into as many as 4,094 broadcast domains and have been a convenient means of isolating different types of traffic that share the same switched LAN infrastructure. Another type of network virtualization that has been in production networks for a decade is Virtual Routing and Forwarding (VRF) instances. VRF is a form of Layer 3 network virtualization in which a physical router supports multiple virtual router instances, each running its own routing protocol instance and maintaining its own forwarding table.

While virtualization techniques such as VLANs and VRF are helpful, they are limited both in scope and in value. To add maximum value, network virtualization must be end-to-end and it must abstract and pool network resources in a manner similar to how server virtualization abstracts and pools compute resources. These capabilities enable the creation of tenant-specific virtual networks whose topology is decoupled from the topology of the underlying physical network and it also enables IT organizations to dynamically create policy-based virtual networks to meet a wide range of requirements. One of the many advantages of decoupling the virtual networks from the physical networks is that it enables IT organizations to make changes to the physical network, such as scaling out capacity, without impacting the existing flows. In similar fashion, one of the many advantages of having the network virtualization be tenant-specific is that it enables complete isolation between each tenant.

3. Network Functionality

For security purposes, it is common for providers of cloud-based services to want to isolate one tenant from another as mentioned in the preceding paragraph. While this requirement is usually discussed relative to the providers of public cloud-based services, it is also a goal for most enterprise IT organizations. For example, in order to respond to myriad industry and government regulations about data security, IT organizations often need to keep the data generated by one set of users isolated from other users. It is also highly beneficial for IT organizations to be able to allow application developers to run their applications in a production environment without impacting production traffic.

To meet these requirements efficiently, the SDN controller must enable the virtual networks described above to be fully isolated from one another and to be configured centrally and to have those configurations automatically enforced. As shown in Figure 2, there is a 12-tuple of header fields that are used to match flows in the flow table. In order to have the most flexibility in terms of how flows are routed, it is important that the SDN controller can make routing decisions based on multiple header fields. It is also important that the controller can define QoS parameters on a flow-by-flow basis.

Relative to routing traffic, a key networking feature of an SDN controller is its ability to discover multiple paths from the origin of the flow to its destination and to split the traffic for a given flow across multiple links. This capability eliminates the limitations of the spanning tree protocol and increases both the performance and scalability of the solution. This capability also eliminates the need to add to the complexity of networks by adding new protocols such as TRILL (Transparent Interconnection of Lots of Links) or SPB (Shortest Path Bridging).

A traditional LAN is a combination of Layer 2 and Layer 3 networks. The value of having both Layer 2 and Layer 3 networks is not diminished just because of SDN. As such, the SDN controller must support a rich set of constructs that enable the creation of Layer 2 and Layer 3 networks within a tenant-specific virtual network. This includes the ability to create Layer 2 networks that switch the traffic between end points and the ability to create Layer 3 networks that route the traffic that flows between Layer 2 networks. It also includes constructs that are described below that enable operators to implement functionality such as intelligent filtering.

4. Scalability

The traditional LAN is deployed in a multi-tiered architecture in which Layer 3 routing functionality is used to connect multiple Layer 2 networks. These traditional LANs do not scale very well when supporting east-west traffic because at least one Layer 3 device, and most likely multiple Layer 3 devices, are in the end-to-end path.

An SDN enables IT organizations to move to a scale-out model of networking whereby they add networking functionality when needed and the SDN controller enables them to manage all of the networking functionality as if it was one device. A key consideration relative to the scalability of an SDN is the number of switches that an SDC controller can support. In the current environment, IT organizations should expect that the controllers they acquire can support a minimum of 100 switches, but they should also realize that the number of switches that a controller can support will depend on the SDN use cases that are being supported.

IT organizations that are evaluating SDN need to be aware of the fact that network broadcast overhead will limit the scalability of the solutions they implement. As a result, when evaluating SDN controllers, IT organizations need to ensure that the controller can mitigate the impact of network broadcast overhead. Another factor that limits the scalability of an SDN is the proliferation of flow table entries that occurs because without some form of optimization, a hop-by-hop entry is required for each flow. As a result, when evaluating SDN controllers, IT organizations also need to ensure that the controller can minimize the proliferation of flow table entries. One way this can be accomplished is by having the SDN controller use header rewrites in the core of the network. If this technique is implemented, the only unique flow table entry is at the ingress and egress of the network.

Another aspect of scalability is the ability of the SDN controller to create an SDN that can span multiple sites. This capability enables the movement of VMs and virtual storage between sites. To maximize the benefit of this capability, the SDN controller must allow the network policies for routing and forwarding to be automatically applied to the migrated servers and/or storage.

5. Performance

   As previously described, one of the key functions of an SDN controller is to establish flows. As such, two of the key performance metrics associated with an SDN controller are the flow setup time and the number of flows per second that the controller can setup. These performance metrics heavily influence when additional SDN controllers must be deployed. For example, if the switches in a production SDN initiate more flows than can be supported by the existing SDN controller(s), more controllers must be added.

   Flows can be setup in one of two ways: proactively or reactively. Proactive flow setup occurs before the packet arrives at the OpenFlow switch, and so when the first packet arrives at the OpenFlow switch, the switch already knows what to do with the packet. This results in negligible setup delay and no real limit on the number of flows per second that the controller can support. Ideally, the SDN controller pre-populates the flow tables to the maximum degree possible.

   In contrast, reactive flow setup occurs when the OpenFlow switch receives a packet that doesn't match the flow table entries and hence the switch has to send the packet to the controller for processing. Once the controller decides how to process the flow that information is cached on the OpenFlow switch and the SDN controller determines how long to keep the cache alive.

   The time associated with reactive flow setup is the sum of the time it takes to send the packet from the OpenFlow switch to the SDN controller; the processing time in the SDN controller and the time it takes to send the flow modification message back to the OpenFlow switch and to populate the flow table on the switch. The key factors that affect flow setup time include the processing power of the switches that are attached to the controller and processing and I/O performance of the Controller. The processing and I/O performance of the controller are influenced by a number of factors. For example, a controller whose software is written in C will be faster than one whose software is written in Java. In the current environment, IT organizations should expect that SDN controllers should not be a bottleneck in the creation/deletion of flow entries controlled by the switch.

6. Network Programmability

   In the traditional IT environment, configuring the network is done on a device-by-device basis. This approach is time consuming and error prone and it leads to inconsistency. This approach is also static, as the configurations don't change as network conditions change. The static nature of the traditional approach to network programmability leads to a reduction in the potential performance of the network.

   As noted, one of the key characteristics of an SDN is that there are programmatic interfaces into the controller. One example of the type of programmability that IT organizations should look for in an SDN controller is the ability to redirect traffic. For example, for security reasons an IT organization may choose to have traffic that is inbound to a server go through a firewall. However, in order to not consume the resources of the firewall with clean traffic, the IT organization may want to choose to redirect the traffic that is outbound from the server to not go through the firewall. This is difficult to accomplish in a traditional networking environment.

Another example of the type of programmability that IT organizations should look for in an SDN controller is the ability to apply sophisticated filters to packets. These filters can be thought of as dynamic, intelligent ACLs. Simple ACLs can be based on direct packet header matching and can be used to determine whether to drop or pass packets. In contrast, the SDN controller should be able to apply filters that consist of complex combinations of multiple packet header fields. The filters should be able to be deployed dynamically on the virtual networks and it is the role of the SDN controller to push the associated flow table entries down to the switches. An SDN controller can also support programmability by providing templates that enable the creation of scriptable CLIs that, unlike traditional configuration management, allow for the dynamic programming of the network.

Another way that an SDN controller enables programmability is by implementing a northbound API as is shown in Figure 1. The northbound API makes the control information that has been centralized in the controller available to a potentially unbounded set of SDN applications that are capable of dynamically changing the underlying network to perform tasks such as forwarding packets over the least expensive path or changing the QoS settings based on the available bandwidth or other factors. These SDN applications include traditional network services such as firewalls and load balancers as well as orchestration systems such as OpenStack. These applications could also include traffic engineering or applications that gather data that is used to perform tasks such as managing the network or enabling usage sensitive chargeback.

7. Reliability

Part of the value provided by an SDN controller is that intelligence in the controller performs design validation as part of configuring the network and that design validation eliminates manual errors and hence increases network availability. In spite of that capability, one of the criticisms of SDN is that the SDN controller is a single point of failure and hence the controller decreases overall network availability. To counter that criticism, IT organizations that are evaluating SDN controllers need to understand the functionality that the controller offers that increases network reliability. One technique that an SDN controller can use to increase network reliability has already been discussed. That technique is the ability to discover multiple paths from the origin to the destination. If the SDN controller then sets up multiple paths between the origin and the destination, the availability of the solution is not impacted by the outage of a single link. Alternatively, if the SDN controller only sets up a single path from origin to destination, when faced with a link failure, the controller must be able to rapidly reroute traffic onto an active link. This requires that the controller is continually monitoring the network topology.

Relative to the availability of external connections, it is important that the controller support technology and design alternatives, such as the Virtual Router Redundancy Protocol (VRRP) and Multi-Chassis Link Aggregation Group (MC-LAG), which are intended to increase network reliability.

In terms of the availability of the controller itself, it is critical that the controller be built using both hardware and software redundancy features. It is also imperative that the SDN controller enable clustering. For example, clustering two SDN controllers in an active/hot standby mode increases reliability and clustering three or more SDN controllers, with one in hot standby mode, increases the availability, scalability and performance of an SDN controller. However, in order for the clustering to provide for very fast failover times, it is important that the solution is capable of maintaining the synchronization of the memory between the active and standby controllers.

8. Security of the Network

   In order to provide security to the network, an SDN controller must be able to support enterprise class authentication and authorization of the administrators of the network. In addition, network administrators must be able to lock down access to SDN control traffic by following the same type of procedures that they use to lock down other key forms of traffic, such as management traffic.

   Some of the additional security related functionality that an SDN controller should provide was already discussed. One example of that functionality is the ability to apply sophisticated filters to packets, which can alternatively be referred to as the ability to implement dynamic, intelligent ACLs. Another example of that functionality is the capability of the controller to ensure that each tenant that is sharing the infrastructure has complete isolation from all of the other tenants. In addition, because an SDN controller is a candidate for a malicious attack, SDN controllers need the ability to both rate limit the control communications and to be able to alert the network administrators when the network is experiencing a suspected attack.

9. Centralized Monitoring and Visualization

   As highlighted in the 2012 Application and Service Delivery Handbook[4], in the majority of instances in which the performance of an application is degrading, the degradation is noticed first by the end user and not by the IT organization. One of the primary reasons why IT organizations are often unaware of degraded application performance is that in the traditional IT environment, IT organizations lack visibility into the end-to-end network flows. For example, sFlow is a widely accepted technology for monitoring networks. One of the limitations of sFlow is that it only focuses on a subset of the fields in the 12-tuple that is shown in Figure 2; e.g., the IP address of the source. Another limitation is that sFlow uses sampling to achieve scalability. This means that based on a defined sampling rate, an average of 1 out of N packets is randomly sampled, and typically the sampling rate is reduced as the speed of the network is increased. This type of sampling doesn't provide the level of detail that is required for applications such as detailed troubleshooting or usage sensitive chargeback. It also doesn't provide the level of detail that is necessary to enable IT organizations to comply with the large and growing set of government and industry regulations.

   One of the primary advantages of an SDN is that it enables IT organizations to have end-to-end network flow visibility. For example, in contrast to sFlow, OpenFlow can be used to collect counters on any defined flow. An SDN controller should be able to use OpenFlow data to identify problems in the network and automatically change the path that a given flow takes. The controller should also allow the IT organizations to monitor some classes of traffic and not

[4] http://www.webtorials.com/content/2012/08/2012-application-service-delivery-handbook-2.html

monitor other classes. For example, an IT organization may choose to not monitor its replication traffic.

Being able to visualize a traditional network has always been both important and difficult. Both the importance and the difficulty of visualization is increased in an environment in which multiple virtual networks run on top of a physical network. In this case, the SDN controller must be able to discover and present to an IT organization a visualization of the physical links in the network. In addition, the controller must be able to present an IT organization with a visualization of the multiple virtual networks that run on the physical network. The controller must also allow IT organizations to see the flows from both the physical and virtual network perspective and to get more detailed information on each flow.

Another key feature that IT organizations should look for is that it should be possible to monitor the SDN controller using standard protocols and techniques. It should, for example, be possible to monitor the health of the controller and the virtual networks that the controller supports using SNMP. The IT organization should determine if the SDN controller provides support for a wide range of standard MIBs and also provides private MIBs to enable the IT organization to monitor the virtual network elements. Ideally the SDN controller provides access to network information such as connected devices, port status, and link status via a REST API.

10. The SDN Controller Vendor

Given the rapidly growing interest in SDN, numerous vendors have entered the market and many more have announced their intention to enter the market. Because of the volatility of the SDN market in general, and of the SDN controller market in particular, IT organizations that are evaluating SDN controllers need to focus not just on the technical attributes of the controller, but also on a couple of key attributes of the vendors who are providing those controllers.

One key attribute that IT organizations should evaluate is the overall technical competence of the vendor. In particular, IT organizations should evaluate the overall strength of the vendor as well as the depth of their engineering organization. One key litmus test that IT organizations should apply is “Does the vendor have the financial and technical resources to support the ongoing development that will be associated with SDN?” IT organizations cannot afford to have their SDN initiatives impaired by using a vendor that doesn’t keep up with the rapidly changing SDN environment.

Another key attribute that IT organizations should evaluate is the resiliency of the vendor in the SDN marketplace. An IT organization that chooses an SDN controller vendor who exits the market in a year or two does harm to the company that they support and they do harm to the reputation of the IT organization within the company. While choosing an SDN controller vendor who is acquired by a larger company is potentially less harmful, it still causes harm. That follows because when a large company acquires a small company there is a tendency for the development of the small company’s product to be slowed down and there is the potential for the direction of the product to change significantly. One of the reasons for that is because the process of assimilating the two companies consumes key resources in the small company. Another one of the reasons is because the types of IT professionals who are excited by working in a small company are often not satisfied working for a larger company and so they leave.

Checklist of Key SDN Controller Functionality

Below is a checklist of the ten criteria that IT organizations should use to evaluate SDN controllers.

_ OpenFlow Support
IT organizations need to understand the OpenFlow functionality that the controller currently supports, including support for optional features and extensions to the protocol. IT organizations also need to understand the vendor's roadmap to implement new versions of OpenFlow.

_ Network Virtualization
It must be possible to dynamically create policy-based virtual networks to meet a range of requirements. These virtual networks must abstract and pool network resources in a manner similar to how server virtualization abstracts and pools compute resources.

_ Network Functionality
This includes the ability to discover multiple paths from origin to destination and to split the traffic across multiple links. It also includes the ability to utilize a rich set of constructs that enable the creation of L2 and L3 networks within a tenant-specific virtual network.

_ Scalability
An SDN controller should be able to support a minimum of 100 switches. It must also be able to mitigate the impact of network broadcast overhead and the proliferation of flow table entries.

_ Performance
An SDN controller must be able to pre-populate the flow tables to the degree possible and it must have processing and I/O capabilities that ensure that the controller is not a bottleneck in the creation of flow entries.

_ Network Programmability
It must be possible to apply sophisticated filters to packets. The SDN controller should provide templates that enable the creation of scriptable CLIs that allow for the dynamic programming of the network.

_ Reliability
It must be possible to have multiple network paths from origin to destination. The SDN controller should also be built using both hardware and software redundancy features and it must be possible to cluster the controllers.

_ Security of the Network
It must be possible to apply enterprise class authentication and authorization and to completely isolate each virtual network. The SDN controller must be able to rate limit the control communications.

_ Centralized Management and Visualization

An SDN controller should enable the IT organization to choose the classes of traffic that it monitors and it should present to the IT organization a visualization of both the physical network and the multiple virtual networks that run on top of it.

_ The SDN Controller Vendor
The vendor must demonstrate that it has the financial and technical resources to support the ongoing development that will be associated with SDN. The vendor must also demonstrate its long-term position and momentum in the SDN marketplace.