University Of Miami
Senior Capstone Project 2014-15

SDN Controller

Project Description

Table of contents

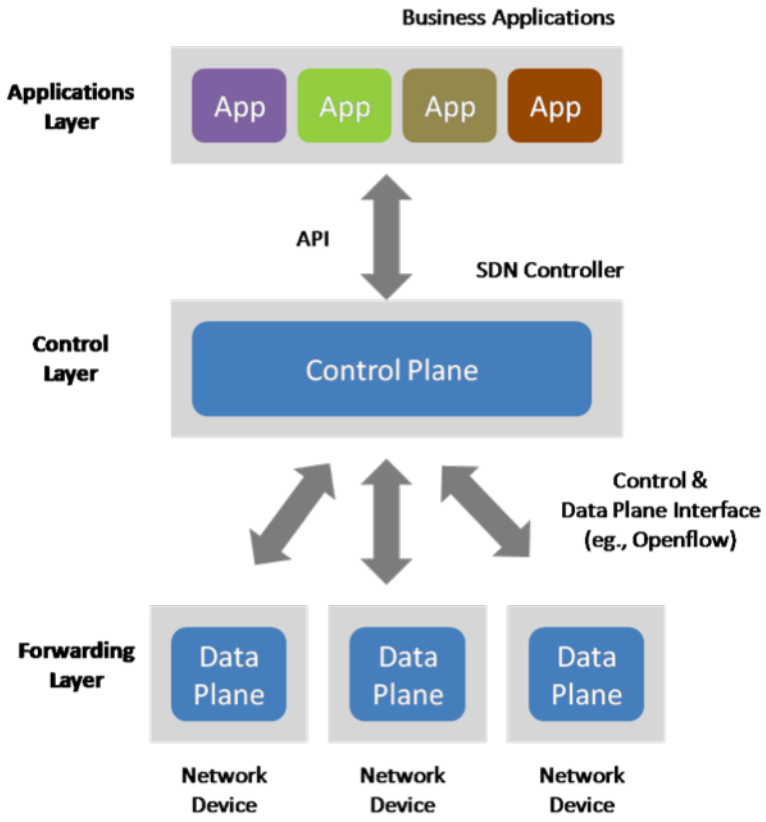
Change Log

Revision	Date	Change Description
1	Sep-15-2014	Initial Document

Introduction

For our Senior Capstone Project proposal, Fortinet is asking the University of Miami students to develop a practical network controller in support of a software defined network (SDN) flow-programmable switch. The goal of this project is to provide students with practical experience in developing solutions within an emerging network technology.

SDN is an emerging networking technology that will change how networks are designed, particularly for data center architectures. In its simplest form, SDN performs an abstraction of networking functions by separating the network device into three layers:



- The Forwarding Layer represents the inline traffic forwarding plane, either a hardware device such as a network switch, or a software device such as a vSwitch or VM
- The Control Layer is the SDN Controller, which is the target of this project. The SDN Controller can be very basic, or can support very complex functions such as route aggregation and reflecting.
- The Application Layer provides APIs into external applications

Prerequisites

A successful participant in this project will have a basic understanding of:

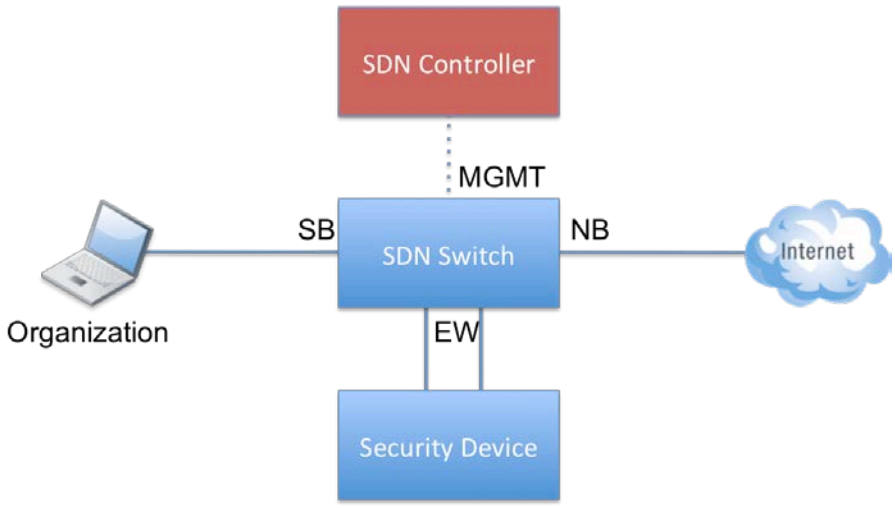
- IP addressing and networking concepts
- An understanding of Ethernet switching, including the use of VLAN tags
- The ability to work with Linux-based OS environments, and associated tools required to establish the project environment
- As the project deliverables require the use of a virtual machine compatible with a Linux KVM hypervisor, a working knowledge of KVM will be required
- As the project is expected to use a web-based interface, the ability to build a web-server and web-based applications is required
- Depending on the methodology, either in using an existing OpenFlow-compatible controller software, or developing your own, an appropriate programming knowledge will be required

A participant will require resources necessary to create a functional environment, to include:

- A device to function as an SDN flow programmable switch. This is most likely a Linux device supporting Open vSwitch (http://openvswitch.org) or LINC/LINCX (http://www.flowforwarding.org/software). Note that the target commercial switch will make use of Open vSwitch as an SDN abstraction layer. However LINC is supported by low-cost Raspberry Pi units.
- A development environment to create the project goal SDN controller
- A firewall or other security device though which the SDN flow-programable switch will forward traffic-of-interest to. A number of virtual machines exist that can operate in this role.

Environment

The following diagram describes the project environment:



Each of these elements is discussed in a subsection below.

SDN Switch

The SDN Switch is an OpenFlow programmable switch supporting OpenFlow 1.3 or better. It will have a minimum of four interfaces, and a maximum of forty-eight interfaces:

- A pair of interfaces will operate as Northbound(NB)/Southbound(SB) interfaces. The SB interface will link to the Organization, while the NB interface will link to the Internet. These interfaces will not have an IP address. The principle operation of the SDN Switch is to forward traffic between the NB & SB interfaces. The NB/SB interfaces also support VLAN tags.
- One interface will act as a management (MGMT) interface. This interface will have an IP address that is reachable by the SDN Controller. The purpose of this interface is to pass OpenFlow statements between the SDN Controller and SDN Switch.
- The remaining interfaces will operate as East/West (EW) interfaces. The purpose of these interfaces are to send traffic to/from Security Devices. There will be a minimum of one EW interface. EW interfaces will not have IP addresses, but will support VLAN tags.

The SDN Switch is intended to detect and send traffic-of-interest to/from an Organization through a defined Security Device. All other traffic should flow directly between the Organization and the Internet.

The commercial SDN switch to be used in testing the project in the second semester will:

- Support OpenFlow 1.3 statements
- Will have four 40G (fiber) interfaces as NB/SB interfaces
- Will have thirty-two 10G (fiber) interfaces as EW interfaces
- Will have two 1G (10/100/1000 copper) interfaces as MGMT interfaces
- Support up to 128K programmed flows

- Will be capable of switching traffic between interfaces at line-rate performance
- Will support a virtual machine (VM) that supports a local SDN controller

Security Device

The security device is any device operating as a Layer 2 switching device performing security functions. This is likely a firewall or intrusion protection system (IPS). There are a number of open source devices and virtual machines available, as well as potential resources at the University's Cybersecurity Lab.

Relative to the SDN Switch, a Security Device can be defined as:

- An EW interface on the SDN Switch, to which the Security Device is attached
- A set of VLANs on an EW interface. For example, if a Security Device uses one EW interface, it may use two VLANs as logical interfaces to traverse traffic.
- An IP protocol and port number that is used to identify the application(s) that are supported by the Security Device

Organization

This is a loosely defined concept that represents a grouping on edge devices. Relative to an SDN Switch, an Organization can be defined as:

- The SB interface
- A VLAN on the SB interface. Multiple Organizations can be segregated by different VLANs
- A set of IP subnets, as defined by an IP address and subnet mask. This can be a single subnet, or a group of dis-contiguous subnets

SDN Controller

This is the goal of the project. The SDN Controller programs the SDN Switch in performing its function. OpenFlow is the protocol used to transfer flow statements from the SDN Controller to the SDN Switch. Likewise, the SDN Switch reports to the SDN Controller its capabilities, as well as statistics on flow utilization.

A superflow or policy, represents a bi-directional set of traffic flows between two entities, in this case between the Organization and the Internet. In the process of inspecting traffic-of-interest, a superflow can be subdivided into four individual flows:

- Assuming the Organization is the originator, originating packet from Organization>Security Device
- Originating packet from Security Device>Internet
- Assuming the Internet is the replier, reply packet from Internet>Security Device
- Reply packet from Security Device>Originator

A key element of the project is in the concept of orchestration. It would be very difficult for an administrator to program flow statements individually. It would make more sense for the SDN Controller to break down a policy statement into the required individual flow statements. For example, a policy statement could be in the form:

"from <Organization> to <Internet> must be inspected by <Security Device>"

The SDN Controller would be responsible for orchestration, by generating the required individual OpenFlow statements from the general policy.

Internet

It is important to understand that the Internet represents a generic target of 0.0.0.0/0 (for IPv4) or ::0/0 (for IPv6). However, the Internet could represent more IP addressable targets. For example, traffic towards a specific IP protocol/port combination (broadly representing an application) may require specific inspection.

Deliverables

A successful project will create an SDN Controller to program the target commercial SDN Switch. It must be delivered in the form of a KVM-compatible VM. It must be able to orchestrate general policy statements into programmable flows delivered via OpenFlow statements. Its capabilities must meet the following requirements as defined by these subsections.

KVM Characteristics

The KVM environment should require no more than 4GB of RAM, and no more than 16GB of storage. It can be supported with up to 2 CPU cores. It will support a single IP addressable network interface

Policy Objects

The SDN Controller will be capable of converting policy statements into individual programmable flow statements. The syntax for the policy statements will be:

From:<Source>, To:<Destination>, Inspector:<Device>

Where:

- Source is the source of the originating packet. This can be either the Organization or the Internet
- Destination if the source of a reply packet. If the Source is the Organization, the Destination is the Internet, and vice-versa
- Device is the Security Device that traffic-of-interest must be inspected by

Relative to these objects, the SDN controller must support up to 2K Organization objects, with a maximum of eight subnets per organization. When using subnet definitions, both IPv4 and IPv6 addresses must be supported. It is important to note that the Internet is also represented using Organization notation

The SDN Controller must support up to 128 Security Devices. It is important to note as described in the Project Environment section, and IP protocol/port designations are assigned to Security Devices rather than Organizations. Additionally, only single device chaining is required to be supported. That is, that only a single Security Device will be supported within a given policy.

From an Orchestration standpoint, the number of generated flows from a policy can be calculated as follows:

(# of subnets in Source) * (# of subnets in Destination)*4 = (# of individual flows)

It is an important consideration that IPv4 Source subnets do not forward to IPv6 Destination subnets, and vice-versa. In generating the individual flows, the SDN Controller must segregate IPv4 and IPv6 flows

GUI Support

The SDN Controller must support a graphical user interface (GUI) that is web-based. The GUI should support:

- Administrator logon
    - Support for at least 10 administrators
- Organization and Security Device object definition and database
- Policy Definition
- SDN Switch Definition
- Flow Visibility & Statistics

CLI Support

The project requires the creation of a CLI shell to support all of the functions as described by the GUI

OpenFlow Support

The SDN Controller will support the OpenFlow 1.3.4 specific, which is available at:

https://www.opennetworking.org/en/sdn-resources/onf-specifications/openflow

Additional Resources

As noted, there are a number of SDN Controller software suites available for use in completing this project, although there is always the option of creating your own based on OpenFlow 1.3.4. The most popular are:

Open Daylight – Provides a GUI-based controller. Available at http://www.opendaylight.org

RYU – An SDN controller framework that is programmable via Python. Available at http://osrg.github.io/ryu/

You should also visit SDN Central at https://www.sdncentral.com/, for additional resources and ideas