Lab 5. API Misuse Attacks in SDN

Experiment Task Design

Problem Definition: Networking functionalities can be implemented in software as applications on top of the control plane in SDN. Each application has its own distinct functional requirements for accessing the controller. Fallacious network applications that misuse APIs in the controller can cause serious security threats to network resources, services, and functions through the control plane due to lack of authentication and authorization for applications and lack of standard open APIs. Students will explore how these unprivileged applications can crash the controller and launch memory leakage attacks.

Submission

Students must take screenshots of all the steps involved and explain in one or two paragraphs. Describe why the attack can be performed.

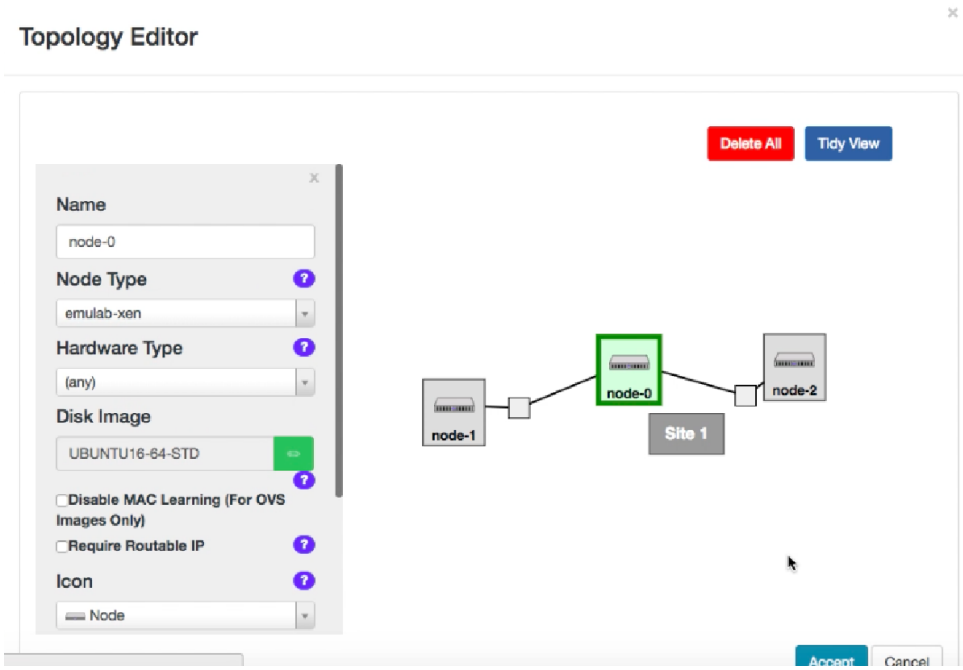
Students can refer the link (http://docs.cloudlab.us/cloudlab-tutorial.html) for more details about creating profiles on CloudLab.

Contents

Conducting the lab

Step 1: Create a Profile

Create a profile with 3 nodes. Use **Ubuntu 16 STD** as the OS and set the node type as **XEN VM**. Click '**Accept**' the topology and then click '**Create**'. Give an appropriate description and then instantiate the experiment. Students will be prompted to select a cluster. Select a cluster with available resources and click finish. Once the experiment is online, proceed to installing Floodlight.



Step 2: Floodlight Installation in middle node.

1) Open a new terminal

Note: Use Floodlight version 1.2

2) Follow the steps below to install floodlight

Get sudo user privileges

sudo su

Update APT repo

apt-get update

Install Java 8

apt-get install -y default-jdk

Install dependencies

apt-get install -y build-essential ant maven python-dev

Install Floodlight:

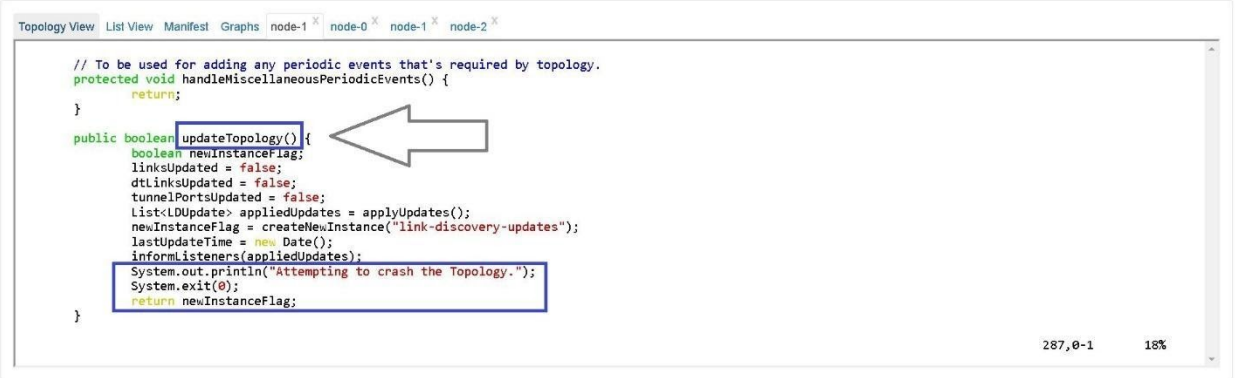
git clone git://github.com/floodlight/floodlight.git -b v1.2
cd floodlight
git submodule init
git submodule update
ant
sudo mkdir /var/lib/floodlight
sudo chmod 777 /var/lib/floodlight

At this stage, students are required to make some changes in one of the modules of Floodlight. Once the change is done, build again using "**ant**" and then run the controller.

3) cd to **/floodlight/src/main/java/net/floodlightcontroller/topology/**

4) Open the **TopologyManager.java** in the desired editor of choice

5) Locate the **updateTopology()** method and before the return statement add a statement "**System.exit(0)**". This function can be found at line 293 in the file.



To verify that the attack works, users can put in a message before **System.exit(0) such as " System.out.println("Attempting to crash the Topology") "** as shown in the screenshot above so that they know that the attack was conducted successfully.

6) cd into the **/floodlight** directory

Run "**ant**" to build the project
Note: To save time for the installation process, students can download an installation script to automate the installation of Floodlight. To use the script, students must do the following:

a) Run "**wget https://people.cs.clemson.edu/~hongdal/set_floodlight.sh**" to get the installation script.
b) Run "sudo chmod +x " to set the installation script as an executable file.
c) Run "**sudo ./set_floodlight.sh**" to install all the dependencies along with Floodlight

7)
8) Run **java -jar target/floodlight.jar**

Step 3: Install OpenVSwitch and setup bridge on switch node

1) Open a new terminal on the middle node.

2) Run '**sudo apt-get update**' to update the repositories

3) Run "**sudo apt-get install openvswitch-switch**" to install OpenVSwitch

4) Setup a bridge on **either node** and connect it to the SDN controller using the following commands listed below:

```
sudo su
ovs-vsctl add-br ovs-lan1
ovs-vsctl add-port ovs-lan1 eth1
ifconfig eth1 0
ovs-vsctl set-controller ovs-lan1 tcp:127.0.0.1:6653
ifconfig ovs-lan1 10.10.10.1 netmask 255.255.255.0 up
```

Note: On the last two steps of the commands above, the user needs to ensure to add the interface containing **the 10.10.*.*** address to ensure that the bridge is detected by the controller. **DO NOT** change the eth0 address as changing the eth0 address will cause the SSH connection to the terminal to go down.

Note: Ensure that the '**set-controller**' command is updated with the appropriate address containing the right subnet. If the command does not have the right address, Floodlight won't

be able to discover the bridge that is trying to communicate to it, thus rendering the attack useless.

Example:

On **Host 0** the address can be seen in the following screenshot:

```
eth1      Link encap:Ethernet  HWaddr 02:29:29:9a:f7:01
          inet addr:10.10.1.1  Bcast:10.10.1.255  Mask:255.255.255.0
          inet6 addr: fe80::29:29ff:fe9a:f701/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:47 errors:0 dropped:0 overruns:0 frame:0
          TX packets:18 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3272 (3.2 KB)  TX bytes:1980 (1.9 KB)
```

To set the appropriate address for '**set-controller**' we need to find the interface on the Controller that has the corresponding right subnet. The following screenshot can be seen below:

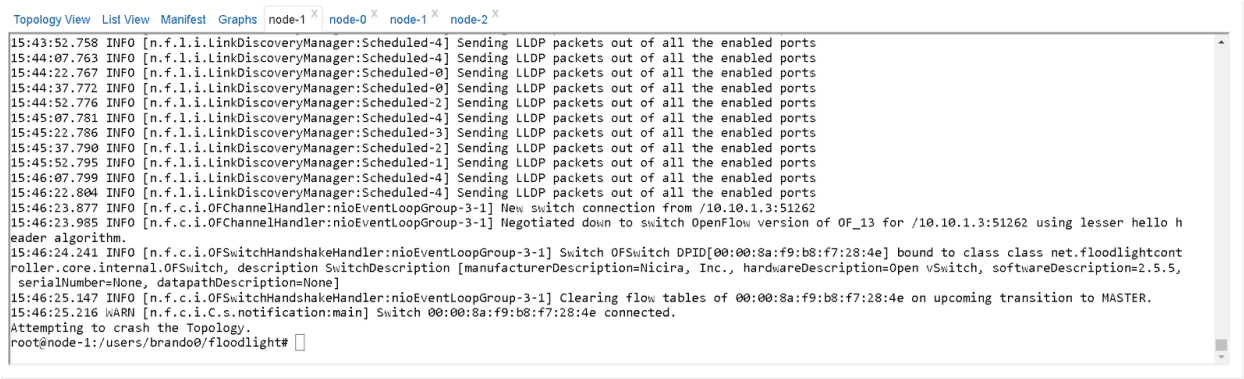
```
brando0@node-1:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 02:2a:7a:99:99:79
          inet addr:172.17.67.3  Bcast:172.31.255.255  Mask:255.240.0.0
          inet6 addr: fe80::2a:7aff:fe99:9979/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:80407 errors:0 dropped:0 overruns:0 frame:0
          TX packets:34471 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:538353295 (538.3 MB)  TX bytes:2803303 (2.8 MB)

eth1      Link encap:Ethernet  HWaddr 02:14:57:d6:21:4f
          inet addr:10.10.1.2  Bcast:10.10.1.255  Mask:255.255.255.0
          inet6 addr: fe80::14:57ff:fed6:214f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:29 errors:0 dropped:0 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1808 (1.8 KB)  TX bytes:1568 (1.5 KB)
```

ovs-vsctl set-controller ovs-lan1 tcp:**10.10.1.2**:6653
ifconfig ovs-lan1 **10.10.1.1** netmask 255.255.255.0 up

Result

As soon as the newly created bridge tries to connect with the controller, the controller crashes and exits.



Explanation:

The Floodlight Controller does not use any robust verification/access control check to verify the use of API that are critical in terms of Floodlight operation or data breach. An attacker can create a malicious module application and try to either crash the controller or access sensitive information using the un-protected APIs. In our case we edited **TopologyManager** which is responsible for recording the state of the network. Everytime a new switch gets added the "**updateTopology()**" method is called. In our malicious code we use **System.exit(0)** function to close the floodlight application. This can also be performed by any other user module. This is possible because floodlight, before using the System library did not verify the authenticity/privilege of the caller.

References

G. Pickett “Abusing Software Defined Networks” [Online], Blackhat.com, Available: “https://www.blackhat.com/docs/eu-14/materials/eu-14-Pickett-Abusing-Software-Defined-Networks-wp.pdf”

Hitesh Padekar, Younghee Park*, Hongxin Hu, Sang-Yoon Chang, “Enabling Dynamic Access Control for Controller Applications in Software-Defined Networks,” the 21st ACM Symposium on Access Control Models and Technologies (SACMAT), June, 2016. (* is the corresponding author.)

S. Shin, Y. Song, T. Lee, S. Lee, J. Chung, P. Porras, V. Yegneswaran, J. Noh, and B. B. Kang, “Rosemary: A robust, secure, and high-performance network operating system,” in Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security . ACM, 2014, pp. 78–89.