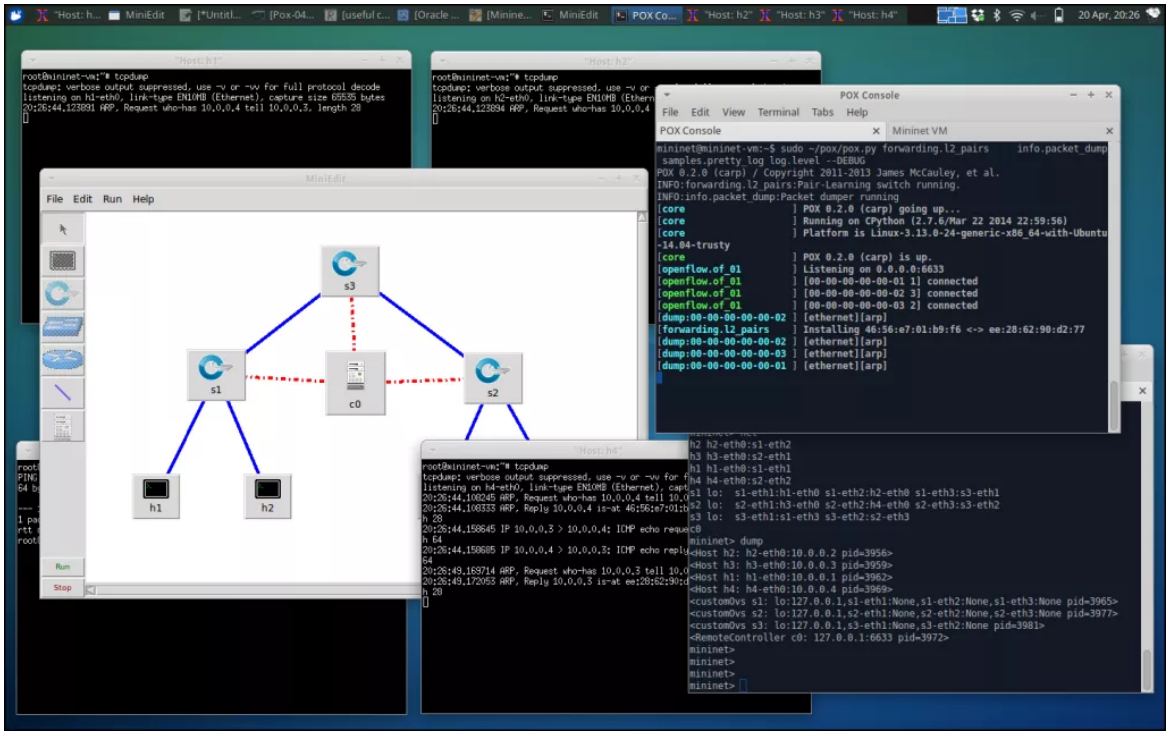
Open-Source Routing and Network Simulation

Using the POX SDN controller

April 20, 2015

In this tutorial, we demonstrate basic software-defined networking (SDN) concepts using the POX SDN controller, POX components, and the Mininet network simulator.



We will show how to use the POX SDN controller to update flow tables on the SDN switches in a simulated network so every host on the network can forward packets to another host. We will use the Mininet network simulator to create the network of emulated SDN switches and hosts that are controlled by the POX SDN controller.

About Mininet

Mininet is an open-source network simulator designed to support research and education in the topic of software defined networks. If you are not already familiar with Mininet, you should review the following posts before starting this tutorial:

- Set up the Mininet network simulator VM

- Using the Mininet network simulator
- Using MiniEdit, the Mininet GUI

More information about Mininet is available at the Mininet web site.

About POX

POX provides a framework for communicating with SDN switches using either the OpenFlow or OVSDB protocol. Developers can use POX to create an SDN controller using the Python programming language. It is a popular tool for teaching about and researching software defined networks and network applications programming.[1]

POX can be immediately used as a basic SDN controller by using the stock components that come bundled with it. This is the scenario we will cover in this tutorial.

Developers may create a more complex SDN controller by creating new POX components. Or, developers may write network applications that address POX's API. In this tutorial, we only briefly discuss programming for POX.

See the POX documentation to learn more about POX.

POX components

POX components are additional Python programs that can be invoked when POX is started from the command line. These components implement the network functionality in the software defined network. POX comes with some stock components already available.

The POX stock components are documented in the POX Wiki and the code for each component can be found in the *~/pox/pox* directory on the *Mininet 2.2* VM image.

For example, the *forwarding.l2_learning* component is in the *~/pox/pox/forwarding* directory, as seen below:

```
$ cd pox/pox
$ ls
boot.py datapaths info log proto tk.py
boot.pyc forwarding __init__.py messenger py.py topology
core.py help.py __init__.pyc misc py.pyc web
core.pyc host_tracker lib openflow samples
$ cd forwarding
$ ls
hub.py l2_flowvisor.py l2_nx.py l3_learning.py
__init__.py l2_learning.py l2_nx_self_learning.py l3_learning.pyc
__init__.pyc l2_multi.py l2_pairs.py topo_proactive.py
$
```

Programming for POX

The general purpose of all SDN controllers, including POX, is to allow users to write their own applications that use the controller as an intermediary — or abstraction layer — between network applications and the network equipment.

To learn how to write applications for POX, developers may study the stock POX components as examples that show how to write their own components or they may review the POX API documentation to learn how to write networking applications that use the POX Python API.

Installing POX

POX comes already installed on the Mininet 2.2 VM image. In this tutorial, we will use the VM image. See my previous post about setting up the Mininet 2.2 VM.

If you wish to install Mininet and POX on your own Linux system — either hardware or a virtual machine — you may use the Mininet install script, which also installs the POX controller when it installs Mininet.

If you wish to install POX by itself, follow the POX installation instructions from the POX documentation.

Running POX

Start POX by running the *pox.py* program, and specifying the POX components to use. For example, to run POX so it makes the switches it controls emulate the behavior of Ethernet learning switches, run the command:

```
mininet@mininet-vm:~$ sudo ~/pox/pox.py forwarding.l2_learning
```

The POX Console

The *POX console* is the terminal session from which we run the POX controller. After POX starts, it will display log information and may optionally display an interactive Python command line, if POX is started with the *py* component.

Quit POX

To quit POX, use the *control-C* key combination in the POX console.

Create a network of switches and hosts

In this tutorial we assume you have already set up a Mininet VM in VirtualBox. Start VirtualBox and then start the VM.

Log into the VM using SSH with X forwarding enabled. From a terminal window on your computer, enter the following command (use *PuTTY* if you use Microsoft Windows):

```
brian@t400:~$ ssh -X mininet@192.168.56.102
```

Where 192.168.56.102 is the IP address of the VM's connection to the *host-only network*. (The address may be different in your case)

Start MiniEdit

We will use MiniEdit, the Mininet graphical user interface, to set up an emulated network made up of OpenFlow switches and Linux hosts.

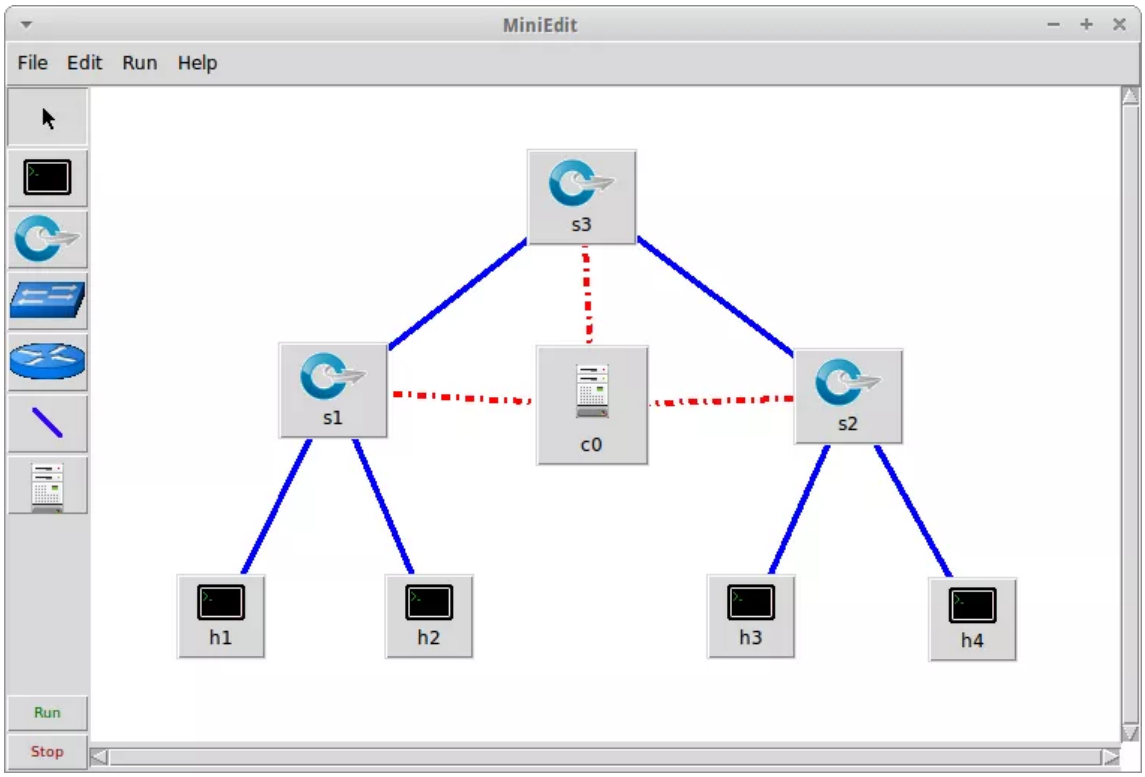
To start Mininet, run the following command on a terminal window connected to the Mininet VM:

```
mininet@mininet-vm:~$ sudo ~/mininet/examples/miniedit.py
```

Now the Mininet window will appear on your computer's desktop. (If you see display errors, first check that you have X forwarding enabled on the SSH connection to the Mininet VM)

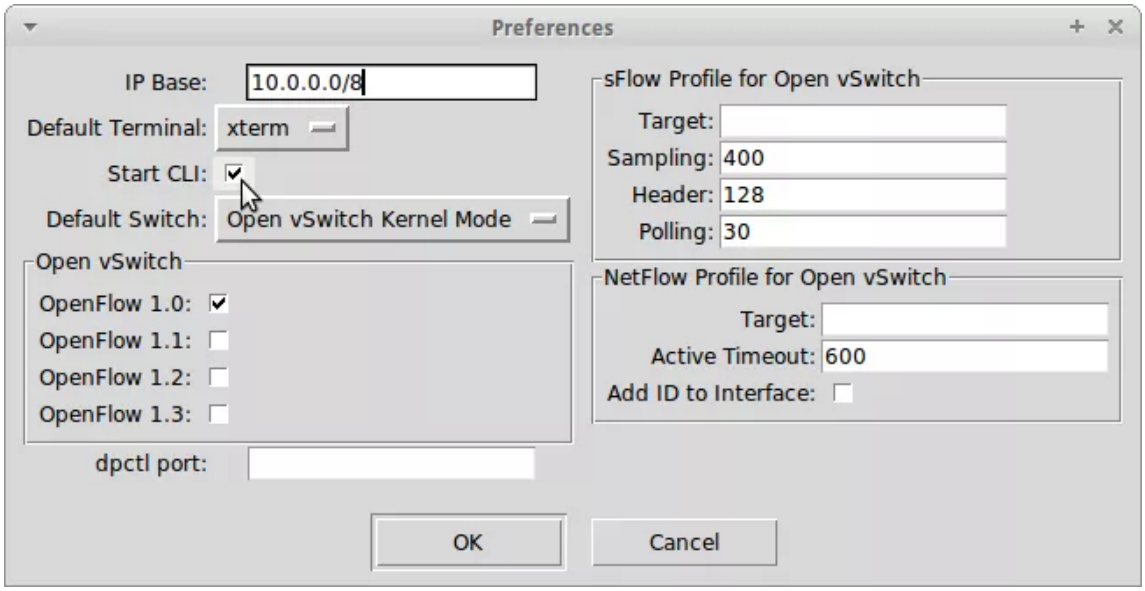
Build the network and use a remote controller

Build the network consisting of a tree to switches with a central core switch connected to two other switches that are connected to two hosts, each. Connect a controller to all the switches.



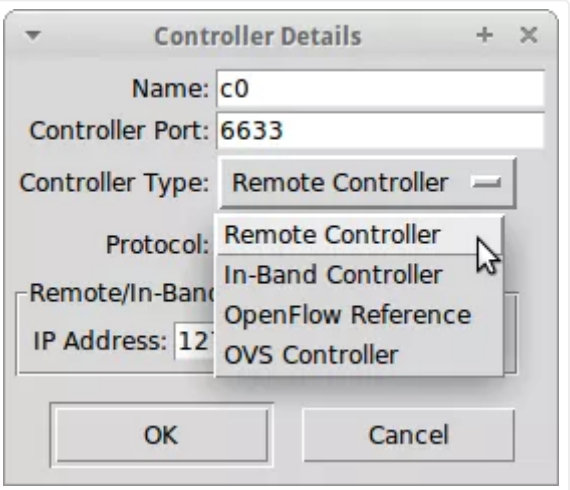
Simple tree with three switches and four hosts

Ensure that the MiniEdit preferences are set so that we can use the MiniEdit command line after starting the simulation. Click on the *Start CLI* box in the MiniEdit preferences window.



Enable CLI in the MiniEdit preferences window

Set up the controller as a remote controller. Right click on the controller and select *Properties* from the menu that appears. Then select *Remote Controller* in the controller properties window.



Configure the controller as a remote controller

In this tutorial, the POX controller is running on the same virtual machine that all the emulated switches and hosts created by Mininet are running on.

When default settings are used, MiniEdit configures OpenFlow switches to try to communicate with a remote POX controller using the host system's loopback IP address and the default OpenFlow port number. So, in this tutorial, all switches created by MiniEdit or Mininet are looking for a remote controller with IP address and port number 127.0.0.0:6633.

Start the MiniEdit simulation

First, save the MiniEdit topology for future use.

Then start the simulation by clicking on the *Run* icon in the MiniEdit tool bar.

The MiniEdit console window will show information about the simulation starting and then will display the Mininet CLI prompt.

At this point, there is no controller connected to the OpenFlow switches so they will no be able to pass any traffic.

Alternative method: Mininet command line

As an alternative to using MiniEdit, the same network can be set up using the Mininet topology commands. I suggest using MiniEdit to create the network topology because MiniEdit provides a visual representation of the network. While MiniEdit is a great tool for creating topologies for the Mininet network simulator, in this tutorial we created a topology that can also be created using standard Mininet commands.

```
mininet@mininet-vm:~$ sudo mn --topo tree,2,2 --controller remote
```

Start the POX controller

Before we start the POX SDN controller, we need to determine which components we want to run when we start the controller.

Select the POX components to run

To select the correct stock components, determine what behavior we want the network of switches to exhibit. Then we will select the stock POX component that provides that functionality.

In this tutorial, we will use components that make POX work like a Layer 2 learning switch, and that dump copies of packets received by the controller to the controller's log file (so we can see what packets the controller sees), and that list controller events to the console log screen in an easy-to-read format. According to the POX documentation, the stock components that do these tasks are: *forwarding.l2_pairs*, *info.packet_dump*, *samples.pretty_log*, and *log.level*.

See the POX documentation for information about other POX components.

Start POX

To start the POX controller with the selected stock components, enter the following command on a terminal session connected to the Mininet VM.

```
mininet-vm:~$ sudo ~/pox/pox.py forwarding.l2_pairs \
info.packet_dump samples.pretty_log log.level --DEBUG
```

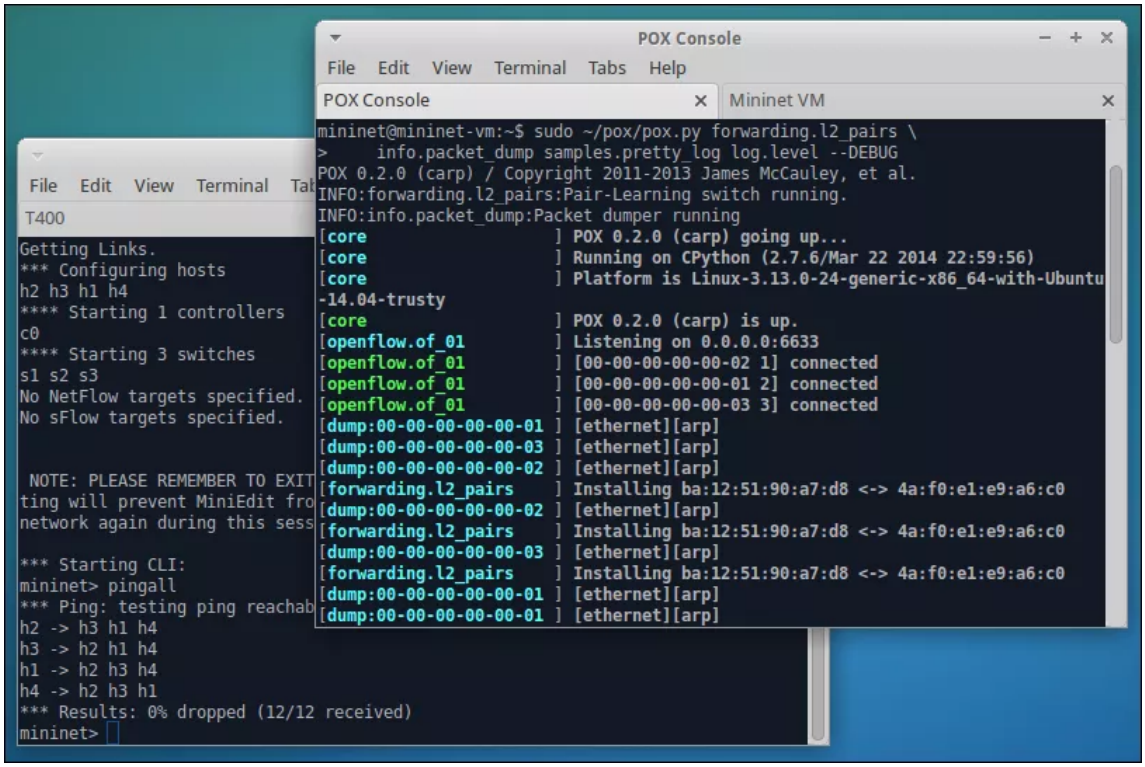
In the POX console, see that the log shows the controller starts and connects to the switches previously set up by the Mininet network simulator:

```
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:forwarding.l2_pairs:Pair-Learning switch running.
INFO:info.packet_dump:Packet dumper running
[core ] POX 0.2.0 (carp) is up.
[openflow.of_01 ] [00-00-00-00-00-03 1] connected
[openflow.of_01 ] [00-00-00-00-00-02 2] connected
[openflow.of_01 ] [00-00-00-00-00-01 3] connected
```

Test the controller

The *forwarding.l2_pairs* component is a very simple application that just matches MAC addresses so it creates a simple scenario to study.

Generate some test traffic between hosts to see how POX builds flows in the network. Run the Mininet `pingall` command, which runs ping tests between each host in the emulated network. This generates traffic to the controller every time a a switch receives a packet that has a destination MAC address that is not already in its flow table.



Mininet *pingall* command and results seen in the POX console

You can see in the POX console window the log messages showing what is happening. When the POX controller running the *forwarding.l2_pairs* component receives a packet from a switch, it tells the switch to flood the ARP packet out its other ports to other switches or hosts. One host eventually responds to the ARP request and then the *forwarding.l2_pairs* component sends OpenFlow messages to each switch to load the required flows into the switch flow tables.

Checking flow tables

To see the contents of the flow tables on all switches, execute the Mininet command:

```
mininet> dpctl dump-flows
```

To check ARP tables on each host, execute the Mininet *arp* command. For example, to show the ARP table for host *h1*, enter the following command:

```
mininet> h1 arp
```

To clear all flow tables on all switches, enter the Mininet command:

```
mininet> dpctl del-flows
```

Next steps

As a next step, we can explore OpenFlow control messages and how flow tables are updated on the switches. In future posts, I will explore more about *OpenFlow* and *Open vSwitch* and the tools used to modify flow tables and view OpenFlow messages.

We will also explore how the other stock POX components work individually and in combination with other components or applications. For example, I'd like to see if we can get the *Gephi* graphing tool working with the

misc.gephi_topo component.

Conclusion

We showed how to set up and run the POX SDN controller, how to choose and run stock POX components, and how to test the operation of the controller by using the Mininet SDN network simulator.

1. See Examples of SDN Controllers from Nick Feamster's SDN course

Share this:

63

Related Posts:

1. **Using the OpenDaylight SDN Controller with the Mininet Network Emulator**
2. **Using POX components to create a software defined networking application**
3. **How to map OpenFlow switches to TCP ports in Mininet SDN simulations**
4. **Visualizing software defined network topologies using POX and Gephi**
5. **How to use *MiniEdit*, Mininet's graphical user interface**

In Learning IP, Open Source Networking mininet, POX, SDN

How to use *MiniEdit*,... Install *Gephi* on the...

31 responses to *Using the POX SDN controller*

Deep Gajjar *June 27, 2015 at 8:33 pm*

Awesome post!

amit bisht *September 19, 2015 at 2:43 pm*

awesome , great site to explore ur knowledge ……

Ahmad *January 10, 2016 at 8:33 pm*

Brian,
Thank you for this post.

Is there some sort of a web UI that let you work with POX controller instead of using CLI?

Brian Linkletter *January 11, 2016 at 2:08 pm*

Hi Ahmad,
No, POX does not have a GUI. I wrote about using Gephi to build a graphical representation of a network managed by a POX controller. You may find that post helpful.
Brian

Douglas *January 26, 2017 at 3:43 pm*

Hi,
I finded this solution in https://github.com/MurphyMc/poxdesk

Moin *February 25, 2016 at 9:52 am*

Load Balancing in pox

Pls help me. where do i got load balancing module in pox

Brian Linkletter *February 25, 2016 at 9:57 am*

POX offers an example of a simple load balancer. See the component misc.ip_loadbalancer.

Moin *February 25, 2016 at 10:20 am*

thank you Mr. Brain for your reply
how to measure performance in mininet like throughput, delay, packet loss etc.

Brian Linkletter *February 25, 2016 at 3:07 pm*

Mininet does not offer the ability to measure traffic performance. You may install software such as iperf on Mininet hosts and use that to generate and evaluate traffic properties. You may also us the tc utilities to manipulate the properties of ports on Mininet host nodes (but not on Mininet switches).

Moin *February 26, 2016 at 2:34 am*

how to install iperf on mininet pls guide me through

Moin *February 27, 2016 at 10:20 am*

pls reply. i am not getting how to measure the performance

Brian Linkletter *February 29, 2016 at 8:45 am*

Hi Moin,
You can find the information you need at https://iperf.fr/. I will add *iperf* to my list

of topics to cover in my blog in the future. Thanks for reading my blog.
Brian

Moin *March 2, 2016 at 3:19 am*

Thank u

Moin *February 25, 2016 at 10:19 am*

thank you Mr. Brain for your reply
how to measure performance in mininet like throughput, delay, packet loss etc.

Aiman *April 8, 2016 at 7:50 am*

Thanks a lot , great job. i am new to SDN and i have a few questions. Q!: Can Mininet be used to make an SDN experiment for a research paper ?. Q2: I read in some research papers that Pox is a controller and Openflow is a controller , how they can be used at the same time if they both are controllers ? the last Q which the most important : How can i modify and extend OpenFlow in Mininet, do know any resource to read as a beginner ? . Again thanks a lot for the great job you have done.

Jojo *April 13, 2016 at 9:05 am*

To Aiman:

Q1:
Yes it can.
i have view this in legit and currently doing my research in this.
Q2:
You have better to straighten up your fact back. The OpenFlow is isn't controller. The controller is are among Ryu, Trema, Pox, Nox, Floodlight, Griffon, OpenDaylight and many more. The agent that realize the controller itself is OpenFlow. A Protocol who gave in for the controller to work.
Q3:
About you can check it out the command in OpenFlow itself in the development log and also in Mininet.

Normally when you construct a topology in mininet you will using this.

*******#,protocols=OpenFlow**(this one is determine version) the rest is more up to OpenFlow script to work.

To Brian:

i have come across your blog early of 2015. It seem you have help to understand and conduct my experiment. But i have meet in dead end. Do you have script for load balance in using 3 host communicate at least?

Keep the good work buddy. I like most of your article.

Regards.

comment devenir riche sur internet *April 9, 2016 at 9:56 am*

I am really inspired together with your writing skills and also
with the format oon your weblog. Is thjis a paid theme or did
you customize it your self? Anyway stay up the excellent high quality writing, it is rare to peer
a great weblog like this one nowadays..

Adnan Anwar *April 23, 2016 at 10:10 pm*

Hi Brian…
when i used the sudo ~/pox/pox.py forwarding.l2_pairs \
info.packet_dump samples.pretty_log log.level –DEBUG
so it says that
Module not found: info.packet_dump

please help…

Jojo *May 2, 2016 at 2:48 pm*

Hi, What was the error, besides that? normally it didn't give error like that.

teguh *May 2, 2016 at 8:37 am*

can i change default listening port pox controller from 6633 to 6634 , cause when i use multiple controller, the mininet can't run the topology.

Brian Linkletter *May 9, 2016 at 12:03 pm*

Yes you can change the port used by the controller so you may run multiple controllers in the same network scenario. See my post about using MiniEdit, where I show an example of using multiple controllers in the same network. http://www.brianlinkletter.com/how-to-use-miniedit-mininets-graphical-user-interface/

SANCHIT *November 19, 2016 at 3:30 am*

Hi Brain,
Thank you for providing this document.
I am trying to implement fat tree topology in mininet using pox controller by creating topology using mini edit . How Can I configure the flow table in the switches for this fat tree topology ?

abha *December 26, 2016 at 7:19 am*

Hi Brian,

I want to write some own algorithm for traffic control for pox controller in mininet. What i have do for it. and also i want to change bandwidth to all link during remote controller. when i changed it i get some problem during run-time.

Thank You

jitendra bhatia *January 16, 2017 at 5:48 am*

Hi

Can I create vehicular adhoc network scenario in mininet like ns2? one more que which controller is good for traffic control in network?

Brian Linkletter *January 16, 2017 at 6:26 am*

I do not know. I suggest you ask this question on the Mininet-WiFi mailing list.

Arijit Panigrahy *March 10, 2017 at 8:58 am*

Hi Brian,

I want to attach two different controllers (pox) to the same network at a time. How should i do it.

For ex, h1———-s1————–s2————–h2

s1 is connected to c1 and s2 is connected to c2.

How should I do it?

mitra *March 12, 2017 at 3:35 pm*

hi Brian
I want set a quality of service module in pox controller and I do not have any information about that.
please guide me
thank you

mitra *March 12, 2017 at 3:43 pm*

hi Brian
I want set a quality of service module in pox controller and I do not have any information about it.Please guide me
thank you

Rashid Amin *April 4, 2017 at 10:45 am*

Hi Brian,

```
h1 --- s1 ------- s2---- h3
\ \
h2 h4
```

I have above topology, where h1 and h2 hosts are connected to switch s1 and h3, h4 hosts are connected to switch s2. I want to block host h1 or h2 traffic at ingress port of s2 or at egress ports of s2, so that it can not communicate to h3 and h4. Please guide me how i can modify my l2_learning file in pox to implement these restrictions at both points (ingress or egress).

kumar *April 4, 2017 at 5:02 pm*

I need a sample code for DDOs attack in mininet or mininet wifi using RYU controller connected to two acces point and two hosts in wireless mode. Can anyone help me with this?

Trackbacks and Pingbacks:

1. How to map OpenFlow switches to TCP ports in Mininet SDN simulations | Open-Source Routing and Network Simulation - September 21, 2015
   […] How to use the POX SDN controller. […]

Follow me on Twitter

Join my mailing list

Please type your e-mail address

Subscribe

Network Simulators

Open-Source Network Simulators

Cloonix network emulator

CORE network emulator

GNS3 network emulator

IMUNES network emulator

Mininet SDN network emulator

Netkit network emulator

NS-3 network simulator

OpenStack all-in-one (DevStack, etc.)

Shadow network simulator

UNetLab and EVE.NG network emulators

VNX and VNUML network emulators

Search this site

Search...

Translate this page

Select Language

Powered by Google Translate

Top Posts

How to emulate a network using VirtualBox

Open-Source Network Simulators

Using the OpenDaylight SDN Controller with the Mininet Network Emulator

How to set up the UNetLab or EVE-NG network emulator on a Linux system

Installing Debian Linux in a VirtualBox Virtual Machine

Recent Posts

Enable nested virtualization on Google Cloud

Install and run the Cloonix network emulator on Packet.net

Set up a dedicated virtualization server on Packet.net

Install the CORE Network Emulator from source code

Netdev 2.1 conference report

Build a custom Linux Router image for UNetLab and EVE-NG network emulators

How to set up the UNetLab or EVE-NG network emulator on a Linux system

DNS and BIND demonstration using the Cloonix network emulator

OFNet SDN network emulator

Psimulator2 forked, updated

How to emulate a network using VirtualBox

How to build a network of Linux routers using quagga

How To Install dCore Linux in a virtual machine

Mininet-WiFi: SDN emulator supports WiFi networks

OpenStack all-in-one: test cloud services in one laptop

Using the OpenDaylight SDN Controller with the Mininet Network Emulator

Saving a Cloonix network topology

Lenovo Thinkpad T420: Another excellent, inexpensive Linux laptop

Use ImageMagick to quickly and easily process images for your blog

Cloonix Network Simulator updated to v28

KVM Performance Limits for virtual CPU cores

Capture data on open-source router interfaces in GNS3

Using VirtualBox linked clones in the GNS3 network simulator

Using POX components to create a software defined networking application

How to map OpenFlow switches to TCP ports in Mininet SDN simulations

IMUNES on Linux

GNS3 Version 1.3: What's new for Open-Source Routers

Book review: The Book of GNS3

Install the GNS3 network simulator version 1.x

Build your own network simulator using open-source DevOps tools

CORE Network Emulator updated to 4.8

Cloonix network simulator updated to version 26

Visualizing software defined network topologies using POX and Gephi

Install *Gephi* on the Mininet VM

Using the POX SDN controller

How to use *MiniEdit*, Mininet's graphical user interface

Install Mininet on an Amazon EC2 server

Control an Amazon EC2 server from an Apple iPad using SSH and VNC

Install the CORE Network Emulator on Amazon AWS

How to run GUI applications on an Amazon AWS cloud server

Archive

Archive

Select Month