Project 1 - fun with SDN
(implementing virtual topologies with Mininet and Floodlight)
Project goals

• Becoming familiar with SDN network management paradigms

• Learning to implement network policies in a SDN domain controller (Floodlight in this project)

• Learning to implement virtual network topologies with Mininet

• Side goals:
  • Have fun!
Project tools

- **Mininet**: a powerful network emulator
  - Enables the creation of **virtual software-defined networks** (consisting of **virtual switches**, **virtual hosts**, and **domain controllers**)
  - Runs entirely inside a VM!
  - Most entities in the virtual network **are implemented as VMs** too…
    - …which run inside the Mininet VM
Project tools - II

• **Floodlight**: an OpenFlow domain controller
  
  • Software controller, written in Java

  • Allows implementing network policies as Java code on top of the Floodlight codebase

• Conveniently **packaged in a VM together with Mininet**

  • Mininet instantiates a virtual network

  • Virtual switches “talk” with a running instance of Floodlight to determine the fate of packets
Let’s take a look
Mininet: creating a network

- Mininet comes with various pre-set topologies
  - You will need to implement your own
  - Need to write some limited amount of Python code
Example: topology creation
Using Mininet with Floodlight

- Mininet gives the abstraction of multiple independent hosts and switches (which in practice are just different network namespaces on the same host)

- It does not, by itself, provide a network management infrastructure

- It requires a separate domain controller to “make sense” of packets and install/remove flow rules
Mininet is designed to interact with OpenFlow domain controllers - it is possible to have a Mininet virtual network interact with a controller running on the host VM.

In this project, you will configure Floodlight to manage the Mininet topology that you are going to create.
Example: running Mininet with Floodlight
Implementing policies in Floodlight

• Floodlight can be programmed to implement custom policies

• As part of this project, you will use the Floodlight API to implement SDN management logic for a particular policy

• Refer to project description (http://www.cs.colostate.edu/~cs557/firstproject.html) for policy details
Other aspects

• In order to verify whether your topology and controller works correctly, you will need to **generate synthetic TCP traffic between hosts**

• How can this be accomplished?

  • Each host must run a **TCP server** (necessary to enable TCP connections)

  • Each host must implement a **traffic generation script** (TCP client)
TCP server & client

- Plenty of language frameworks allow you to generate simple TCP servers and clients with little effort.

- E.g.: Python’s `socket` package, Node.js’ `net` package, and so on...
General advice

• In case the project specifications result unclear/underspecified, ask the instructor!

• Talk with classmates to discuss solutions to project challenges

• Do not try to do everything at the last minute

• Do not try to do everything at the last minute

• DO NOT try to do everything at the last minute
Useful resources

• Mininet walkthrough: http://mininet.org/walkthrough/

• Floodlight VM walkthrough: https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/8650780/Floodlight+VM

• Floodlight - how to write a module: https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343513/How+to+Write+a+Module

• Plenty of other material online!
Questions?