Georgia Institute of Technology

Dr. Nick Feamster
Associate Professor

Software Defined Networking

In this course, you will learn about software defined networking and how it is changing the way communications networks are managed, maintained, and secured.

School of Computer Science

Module 4.2: The Control Plane

- Three Lessons
  - Control Plane Basics (OpenFlow 1.0 and Beyond)
  - **SDN Controllers**
  - Using SDN Controllers to Customize Control
- Programming Assignment (and Quiz)
- Quiz

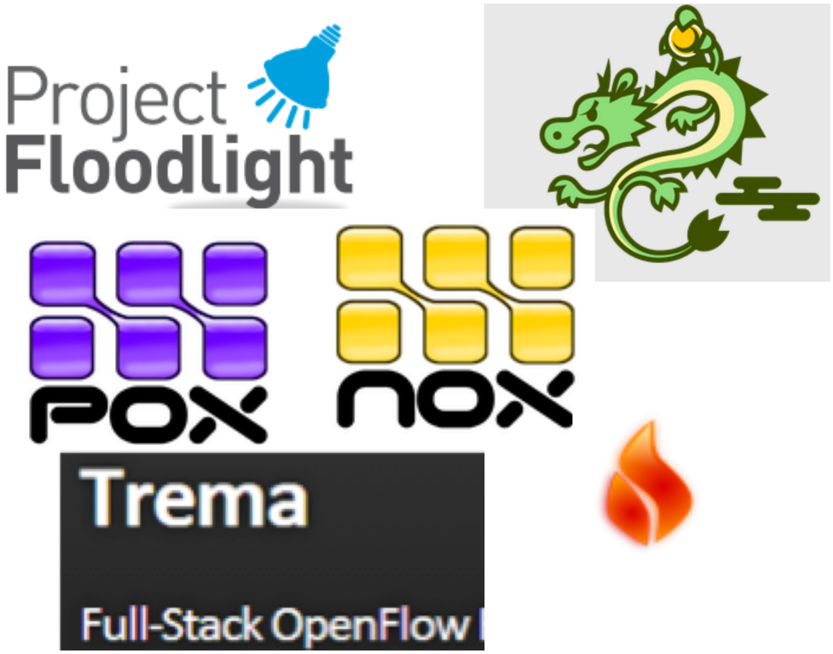
(with help from Hyojoon Kim, Ph.D. student at Georgia Tech)

Lesson Overview

- Overview of different SDN Controllers
- Basic understanding of each controller
  - Concepts
  - Architecture
  - Programming Model
- Pros and cons of each controller
- Ideal situations for each controller

Many Different SDN Controllers

- **NOX/POX**
- **Ryu**
- **Floodlight**
- Pyretic
- Frenetic
- Procera
- RouteFlow
- Trema

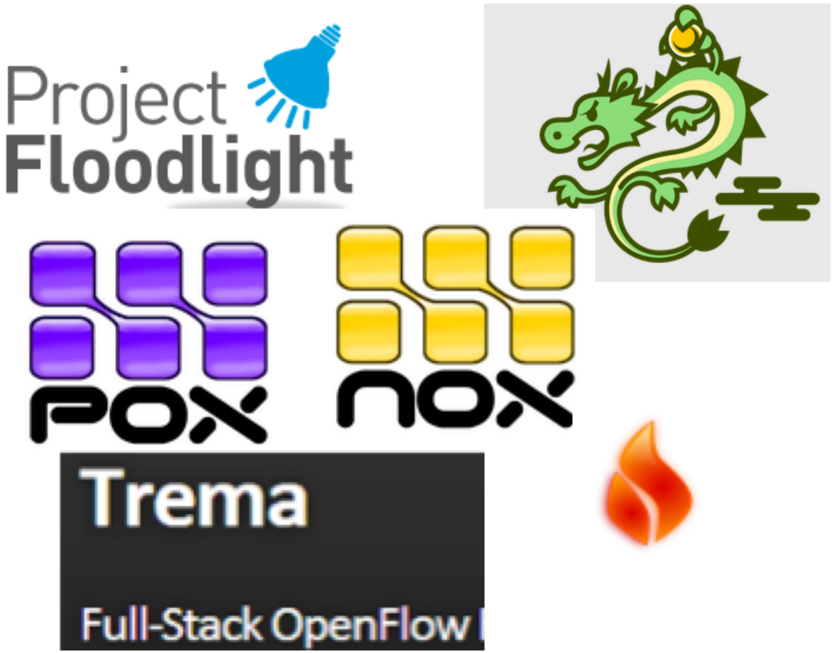


Many Considerations

- Programming Language (can affect performance)
- Learning curve
- User base and community support
- Focus
  - Southbound API
  - Northbound API / “Policy Layer”
  - Support for OpenStack
  - Education, Research, or Production?

Many Different SDN Controllers

- **NOX/POX**
- **Ryu**
- **Floodlight**
- Pyretic
- Frenetic
- Procera
- RouteFlow
- Trema



NOX: Overview

- First-generation OpenFlow controller
  - Open source, stable, widely used
- Two “flavors” of NOX
  - **NOX-Classic:** C++/Python. No longer supported.
  - **NOX (the “new NOX”)**
    - C++ only
    - Fast, clean codebase
    - Well maintained and supported

http://www.noxrepo.org

NOX: Characteristics

- Users implement control in C++
- Supports OpenFlow v.1.0
  - A fork (CPqD) supports 1.1, 1.2, and 1.3
- Programming model
  - Controller registers for events
  - Programmer writes event handler

When to Use NOX

- You know C++
- You are willing to use low-level facilities and semantics of OpenFlow
- You need good performance

POX: Overview

- NOX in Python
  - Supports OpenFlow v. 1.0 only
- **Advantages**
  - Widely used, maintained, supported
  - Relatively easy to read and write code
- **Disadvantages:** Performance

http://www.noxrepo.org/pox/about-pox/
https://openflow.stanford.edu/display/ONL/POX+Wiki

When to Use POX

- If you know (or can learn) Python and are not concerned about controller performance
- Rapid prototyping and experimentation
  - Research, experimentation, demonstrations
  - Learning concepts

Ryu

- Open source Python controller
  - Supports OpenFlow 1.0, 1.2, 1.3, Nicira extensions
  - Works with OpenStack
- Aims to be an “Operating System” for SDN
- **Advantages**
  - OpenStack integration, OpenFlow 1.2 and 1.3
- **Disadvantages:** Performance

http://osrg.github.io/ryu/

Floodlight

- Open-source Java controller
  - Supports OpenFlow v. 1.0
  - Fork from the Beacon Java OpenFlow controller
  - Maintained by Big Switch Networks
- **Advantages**
  - Good documentation
  - Integration with REST API
  - Production-level performance, OpenStack
- **Disadvantages:** Steep learning curve

http://www.projectfloodlight.org/floodlight/

When to Use Floodlight

- You know Java
- You need production-level performance and support
- You will use the REST API to interact with the controller

Summary

	NOX	POX	Ryu	Floodlight
Language	C++	Python	Python	Java
Performance	Fast	Slow	Slow	Fast
OpenFlow	1.0 (CPqD: 1.1, 1.2, 1.3)	1.0	1.0, 1.1, 1.3	1.0
OpenStack	No	No	Yes	Yes
Learning Curve	Moderate	Easy	Moderate	Steep

- Choice of controller depends on needs, language, etc.
- **So far:** Southbound API implementations
  Next week: “Northbound” APIs