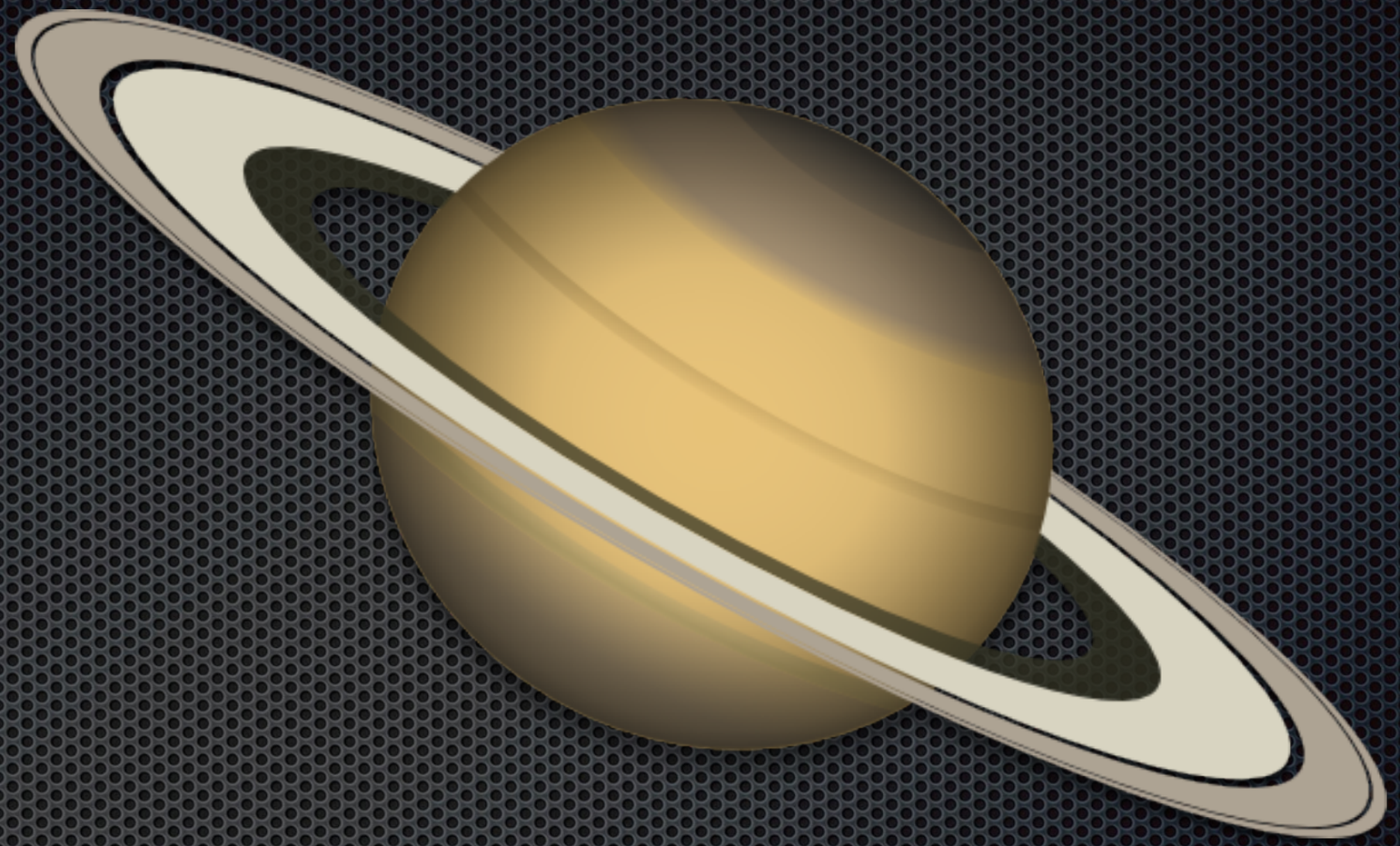
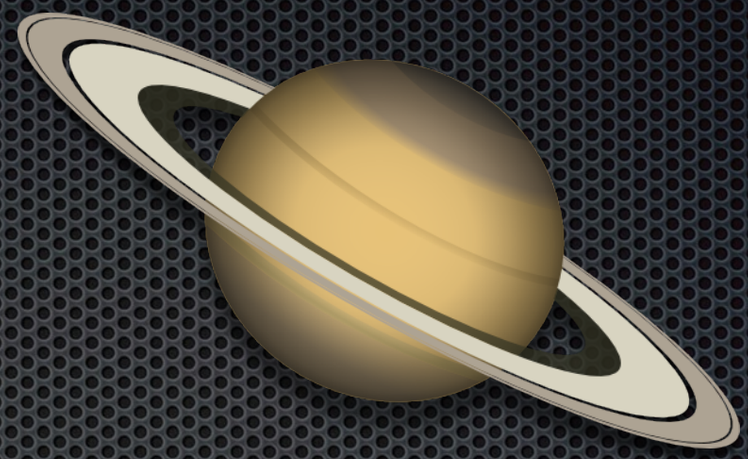


Cassini

SDN Network Edge Manager

Chris Small
University of Washington

Cassini



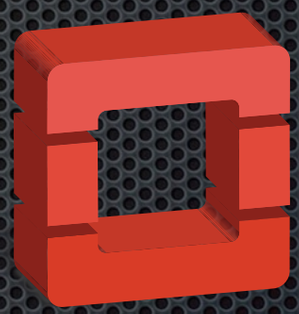
- Network Edge Manager
- Goal: Manage all networks like Wireless
  - Airwave
  - Drop switch with only config to connect to controller
- Open-Source, Vendor Neutral
- Applications can be added as needed
- SDN not OpenFlow

Cassini technologies

- Constructed like a Scaleable Web App
  - Can scale based on number of connections
  - Virtualized infrastructure - not reliant on hardware
  - Could be deployed in Private or Public Cloud
  - REST based interfaces to integrate with other components
    - e.g. Vulnerability Scanner can send messages to block ports automatically

Cassini components

- Automation layer
  - “Openstack for edge networks”
- Higher level abstraction
  - Python based framework
  - REST interfaces
  - “Drivers” to lower level implementations
    - Not dependent on SDN Controller
  - Web interface tied to high level model
    - Extensions

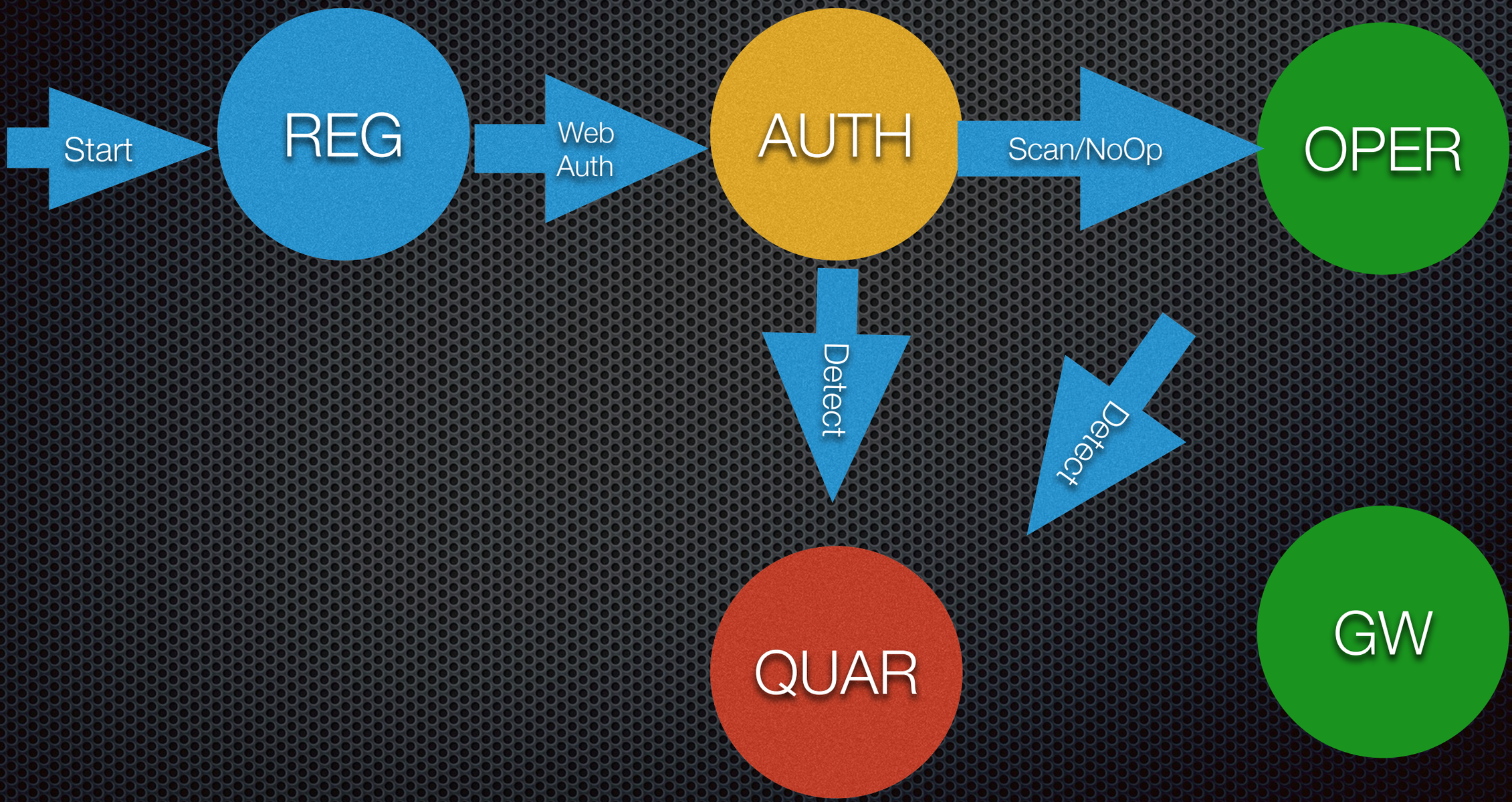


openstack™
CLOUD SOFTWARE

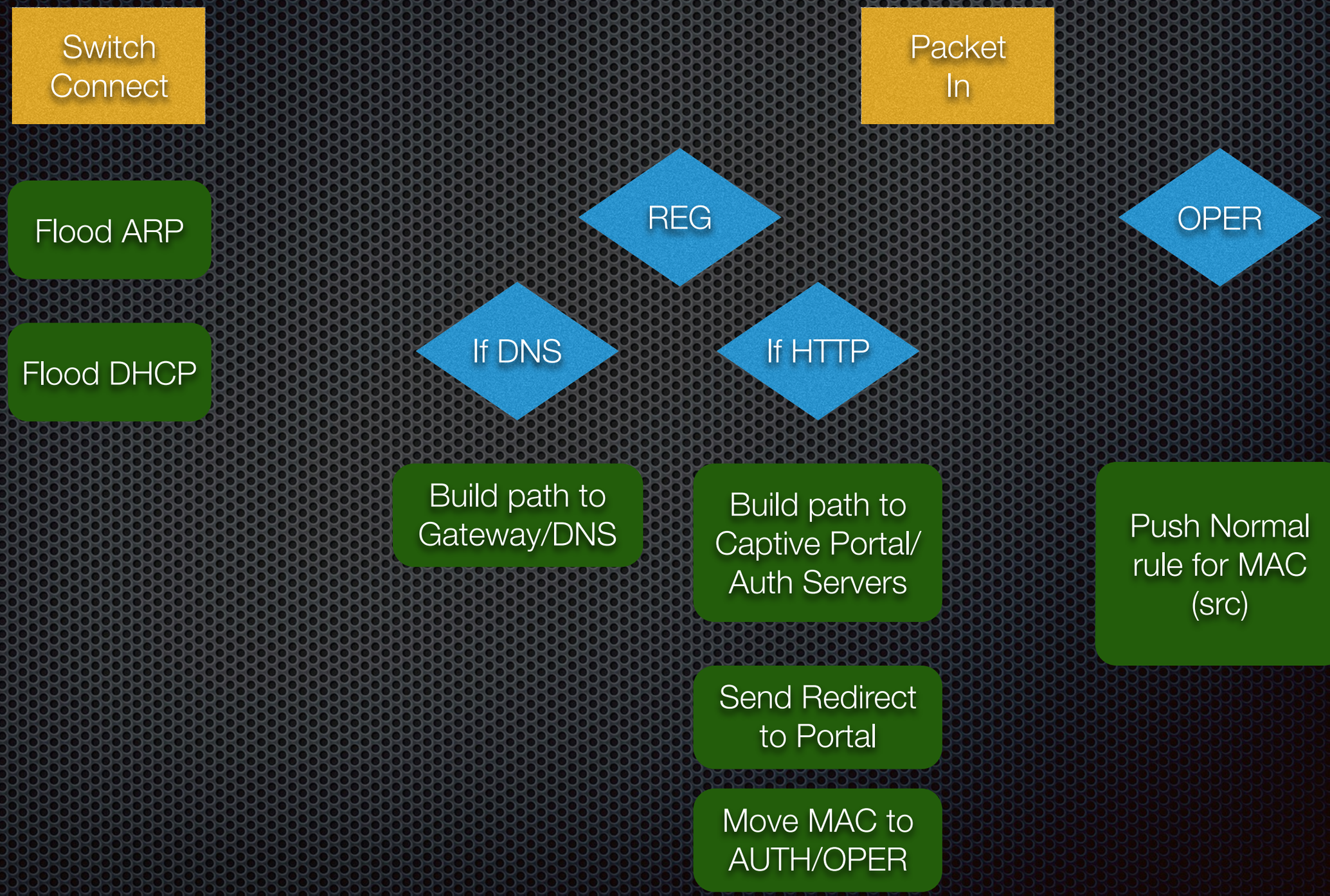
Initial Application - NAC

- Network Access Control
  - Enable authentication for MAC address
  - Not requiring 802.1x but work with it if available
  - Captive Portal
  - Do not require re-DHCP to change state
  - Single sandbox
  - Wired and Wireless
  - Inspired by Resonance - Georgia Tech

NAC State Diagram



Cassini Flowchart



NAC Controller

- OpenFlow Based
- ARP/DHCP/DNS Whitelisted
- Shibboleth authentication
  - Lots of IPs need to be whitelisted - DNS loadbalancer
- Topology, Status and other info

New Service models

- Multi-Tenancy and Clearinghouse
  - Allow Network Application as a Service
    - Divide network applications across the same organization
    - Allows applications too be provided by other groups an organizations

Web Interface Technologies

- Django
- AJAX
- Bootstrap
- D3JS - w/ ESNet Libs
- DataTables
- JSON-RPC
- Web APIs - (more functionality in future)

Controller

- PoX based - Python
- SQLAlchemy DB libraries
- Independent of Web interface models
- Controller independence for different applications
- OpenDayLight, Floodlight, etc….

Testing Methodology

- Mininet
  - Can create arbitrary topology
  - Scripts can simulate user experience
  - Can do load testing at campus scale
  - Uses NAT to allow real connections

Sustainable Development

- GitHub
  - https://github.com/uw-tm/cassini
- Broader Impact/Outreach
  - External instance on AWS
    - Anyone can create an account and a project (with limits)
    - Point switch to controller(s) associated with a Cassini project
  - Mailing Lists and Documentation

Cassini Live Demo

Todos

- CSS and Javascript cleanup
- Still some admin tasks not through “nice” UI
  - Creating new project - Spin up controller
  - User account management
  - Status and measurement
  - Whitelist addresses for Auth
- Load Testing

Next Applications

- Bonjour/AirPlay Manager
- Dynamic Circuit Services
  - OSCARS, OESS, etc..
- Automated Patch Panel
- Datacenter to Edge integration — Network Virtualization
  - OpenStack, VMWare
- Network Function Virtualization
- Multiple Controllers