Software-Defined Network Management

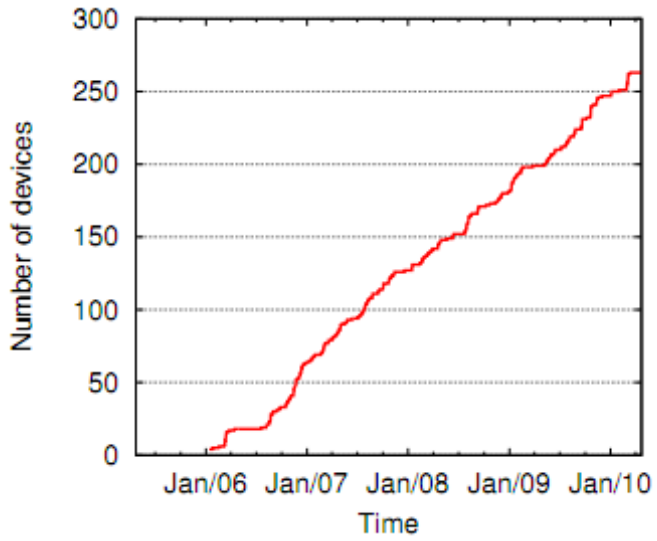
Nick Feamster
Georgia Tech



(with Joon Kim, Marshini Chetty, Srikanth Sundaresan)

Network Management is Hard!

- Manual, error-prone, complex
- Network configurations change continually
  - Provisioning of new users and devices
  - Adjustments to access control
  - Response to incidents
- Changes result in errors



Morning Reports

Georgia Tech

OIT Office of Information Technology

Friday 10/14/2011

Network

-Fri 0439-0612: Network equipment in O'Keefe down, possibly due to a power outage. Switches came back up on their own.

Systems

-Thu 1305-1320: Network monitoring appliance in BCDC/811 Marietta rebooted, causing users of applications served from BCDC to experience problems. This included MyGatech email, buzzport, HR Psoft, Degreeworks. OIT technicians resolved the problem and all services became available again around 1320.

Software Defined Network Management

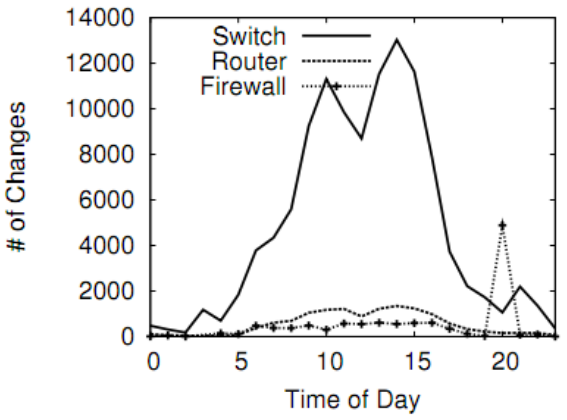
- Software defined networking makes it easier for network operators to evolve network capabilities

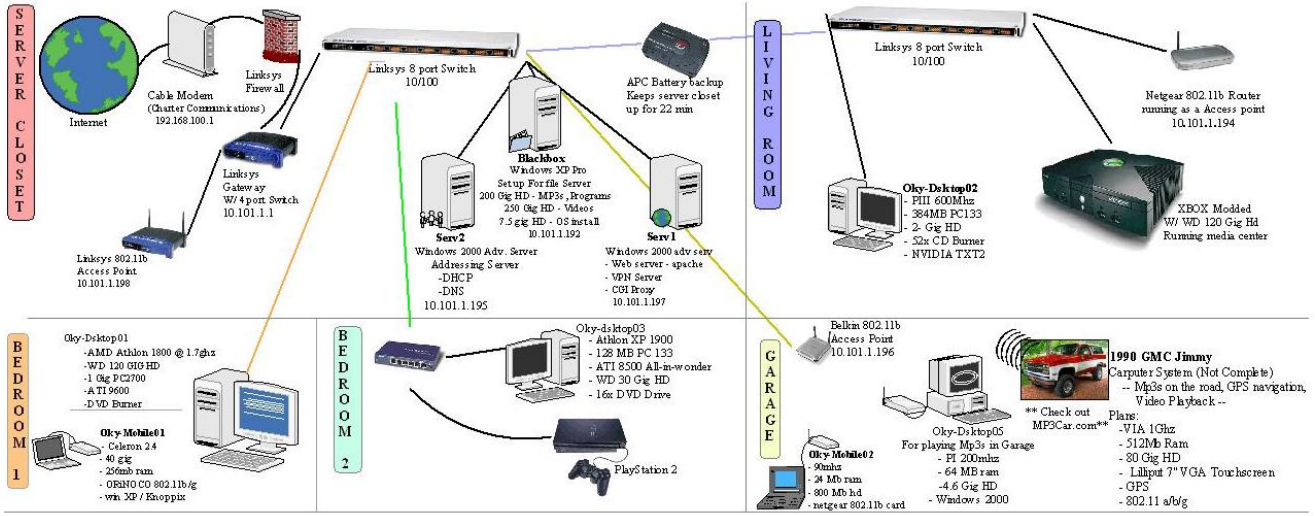
- Can SDN also help network operators manage their networks, once they are deployed?
  - Campus/Enterprise networks
  - Home networks

Why is network management so hard today?

Configuration is Complex, Low-Level

- A campus network may have
  - More than one million lines of configuration
  - Thousands of devices
  - Hundreds of thousands of changes every year
- Home networks can be complex, too





Network State is Dynamic

- Enterprise and campus networks are **dynamic**
  - Hosts continually coming and leaving
  - Hosts may become infected
- Today, configuration is **static**, and poorly integrated with the network
- **Instead:** Dynamic network configuration
  - Track state of each host on the network
  - Update forwarding state of switches per host as these states change

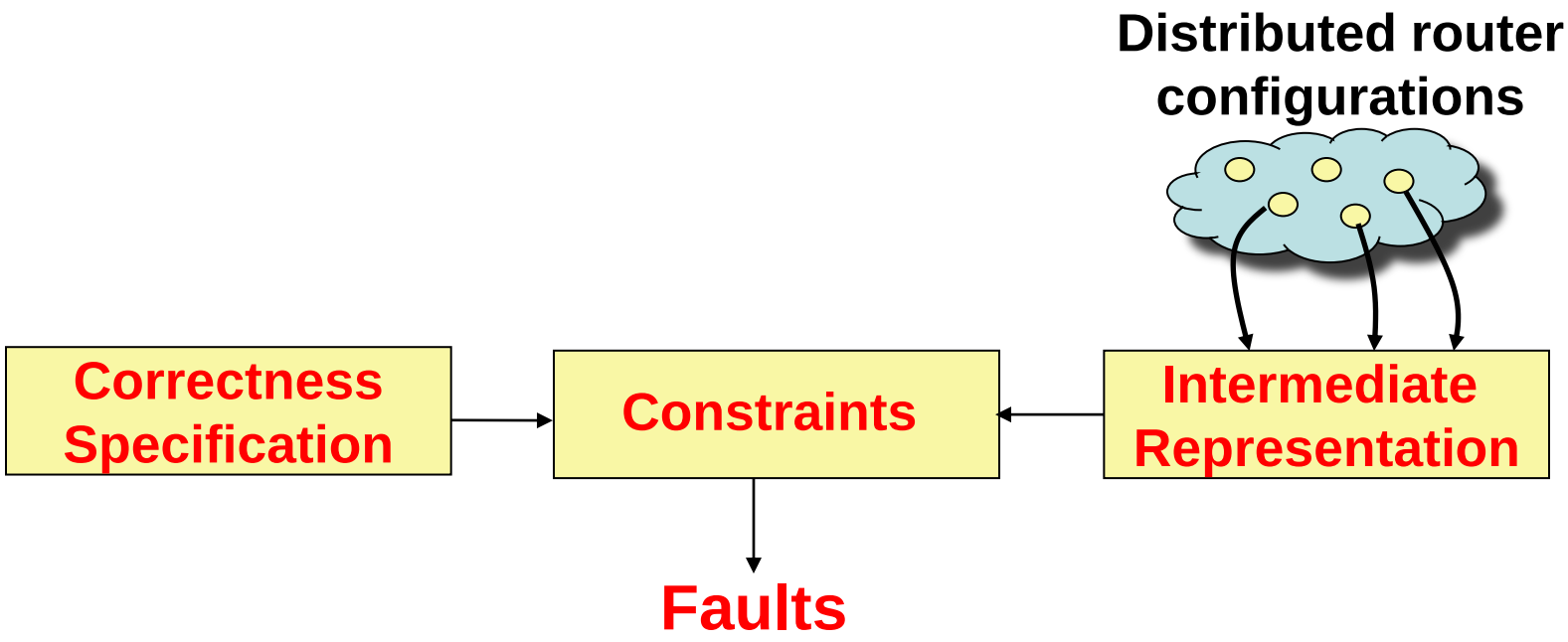
Too Much Complexity is Exposed

Network Devices are Heterogeneous

- Many components “bolted on” after the fact
  - **Campus:** Firewalls, VLANs, Web authentication portal, vulnerability scanner
  - **Home:** Set-top boxes, cameras, laptops, desktops, phones

- Separate (and competing) devices for performing different functions
  - Registration (based on MAC addresses)
  - Vulnerability scanning
  - Filtering
  - Rate limiting

Retrofit: Configuration Checkers



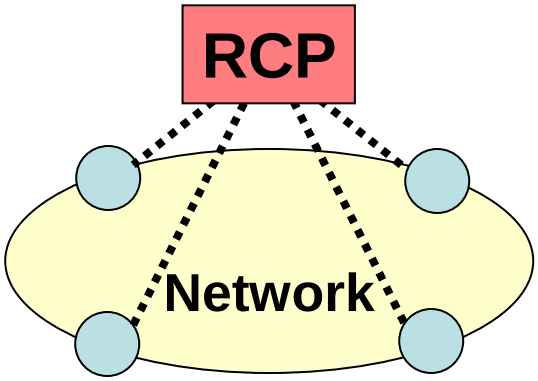
- Downloaded and used by hundreds of ISPs
- Configuration faults found in every network

Feamster and Balakrishnan, Detecting BGP Configuration Faults with Static Analysis. *Proc NSDI*, 2005. Best Paper Award

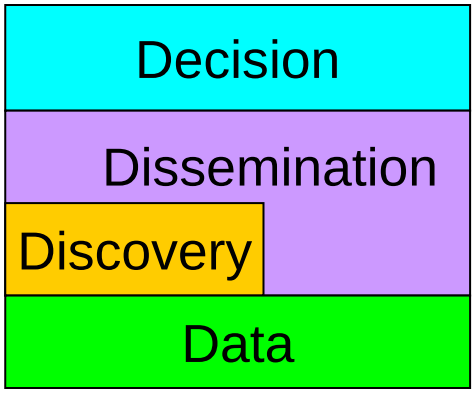
Better: Software-Defined Networking

- Distributed configuration is a bad idea
- **Instead:** *Control* the network from a logically centralized system
- Policies become high-level programs

2004



2005



2008

Feamster *et al*. The Case for Separating Routing from Routers. *Proc. SIGCOMM FDNA*, 2004

Caesar *et al*. Design and implementation of a Routing Control Platform. *Proc NSDI*, 2005

Resonance: Don't Configure the Network, Program It!

- **Today:** Configuring networks with low-level, distributed, vendor-specific configuration

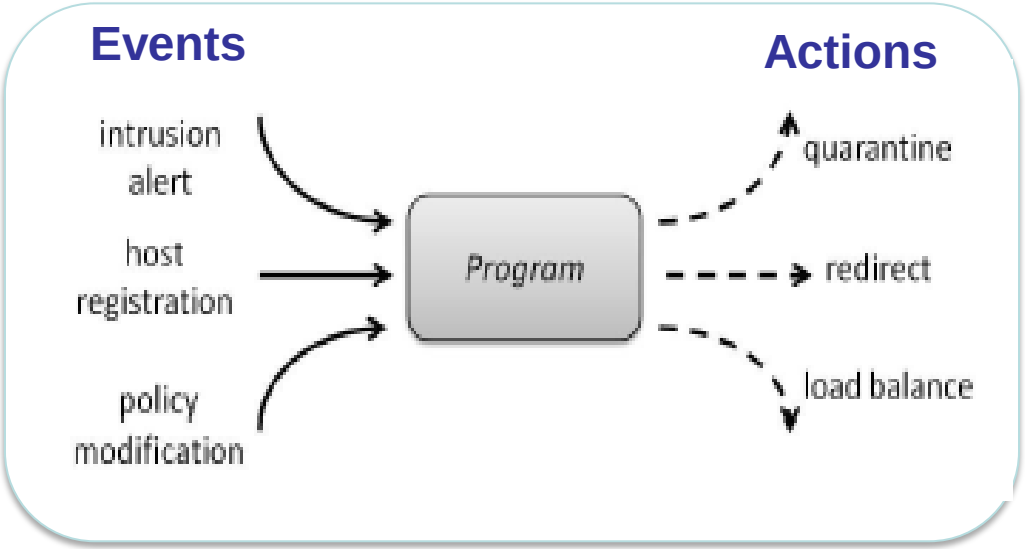
- **With SDN:** Writing network policies and protocols as programs
  - More expressive
  - More predictable
  - More evolvable
  - More usable

Resonance: Approach

Challenge	Approach
Dynamic State	Event Listener w/State Machine
Low-Level Configuration	High-Level Policy Language
Exposed Complexity	Refactoring Functions
Heterogeneity	Standard Control Protocols

Processing Dynamic Events

- **Idea:** Express network policies as event-based programs.



- Policies can be expressed as centralized programs

State Machines

- Step 1: Associate each host with generic states and security classes
- Step 2: Specify a state machine for moving machines from one state to the other
- Step 3: Control forwarding state in switches based on the current state of each machine
  - Actions from other network elements, and distributed inference, can affect network state

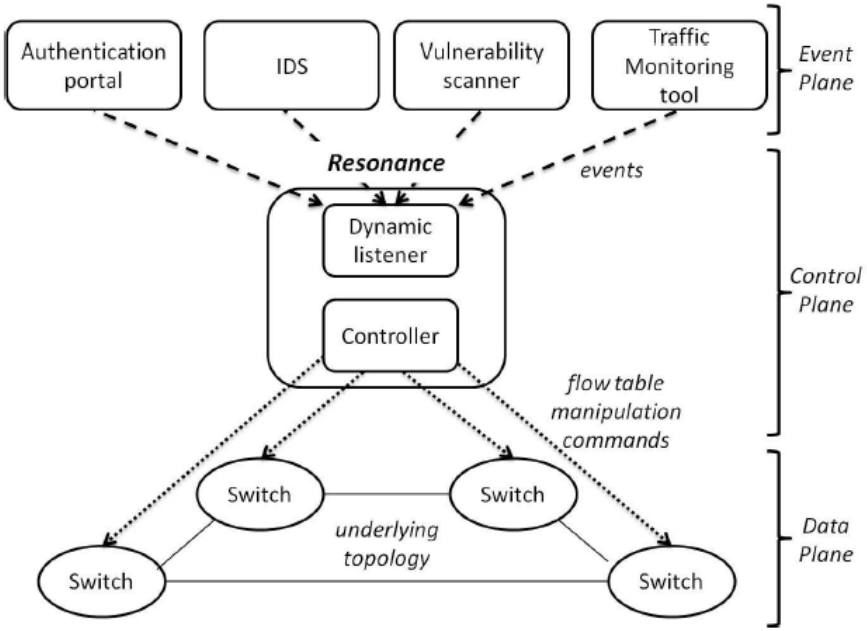
High-Level Policy Language

- Defines states, actions, transitions
- High-level, logically centralized
  - Easier testing and analysis
  - Less complex
- Design is still in-progress

```
if packet-in event occurs:
 - lookup the table by src Ethernet address
 - determine state and security class
switch(state)
   case Registration:
      redirect to web portal: HTTP traffic(to port 80,8080,443)
   case Operation:
      switch(security class)
         case guest:
            if (time is between 12am to 6am)
               block: all
            else
               block: to netws machines
               allow: HTTP traffic
         case gtuser:
            block: to netws machines
            allow: all
         case gtnet:
         case netws:
            allow: all
   case Quarantined:
         block: all
```

Standard Controls

- **Events:** Heterogeneous devices generate standard events that a dynamic listener processes
- **Actions:** OpenFlow channel between controller and switches controls behavior



Refactoring Functions

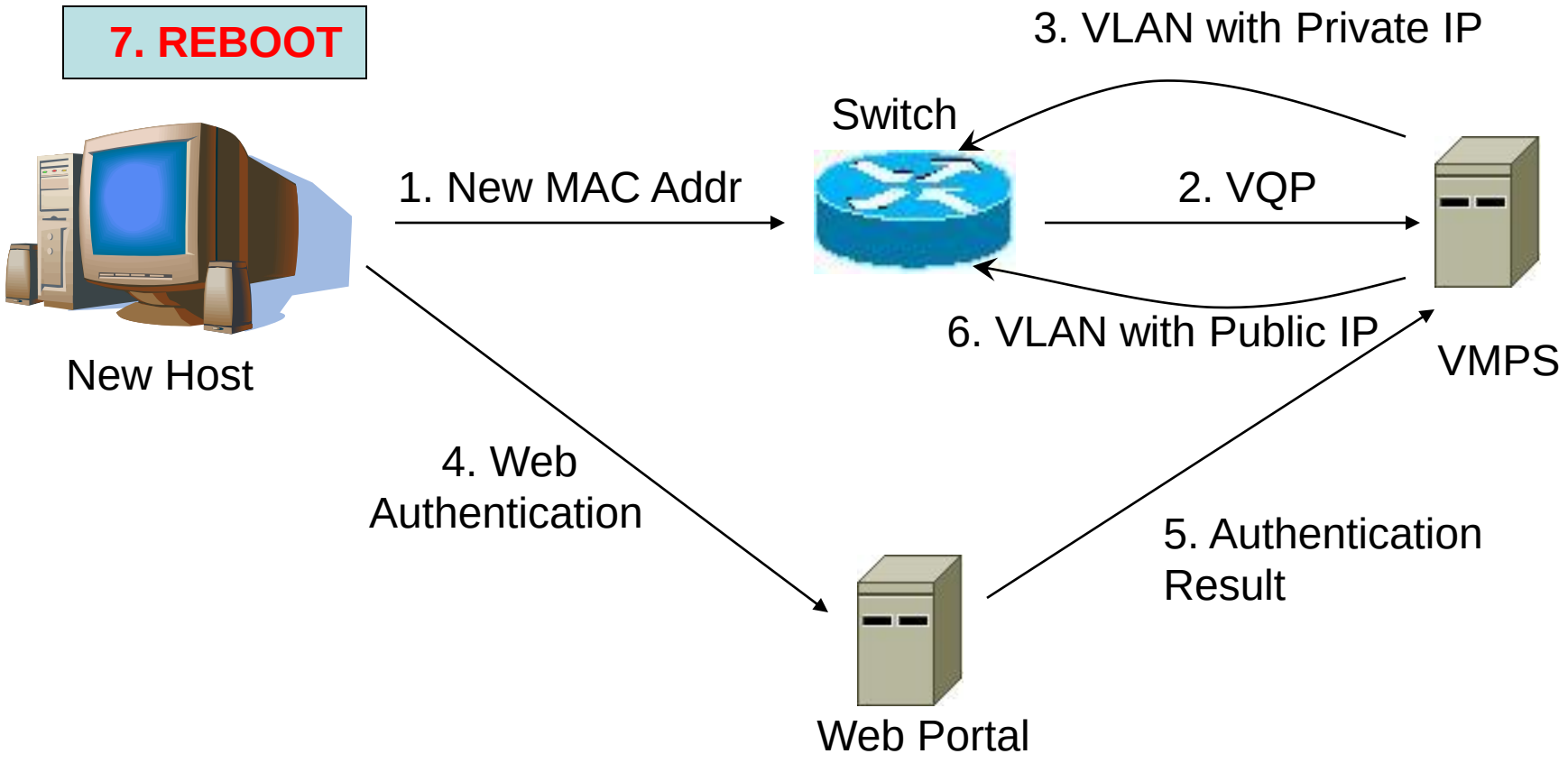
- **Current interfaces:** Decisions only about whether to hide or display complexity

- **Instead:** Changing where function is placed in the system can make the system more usable
  - Only expose information if it
    - Improves situational awareness
    - Is actionable

Two Case Studies

- **Access control in enterprise networks**
  - Re-implementation of access control on the Georgia Tech campus network
  - **Today:** Complicated, low-level
  - **With SDN:** Simpler, more flexible
- **Usage control in home networks**
  - Implementation of user controls (e.g., usage cap management, parental controls) in home networks
  - **Today:** Not possible
  - **With SDN:** Intuitive, simple

Case Study:
Enterprise Access Control

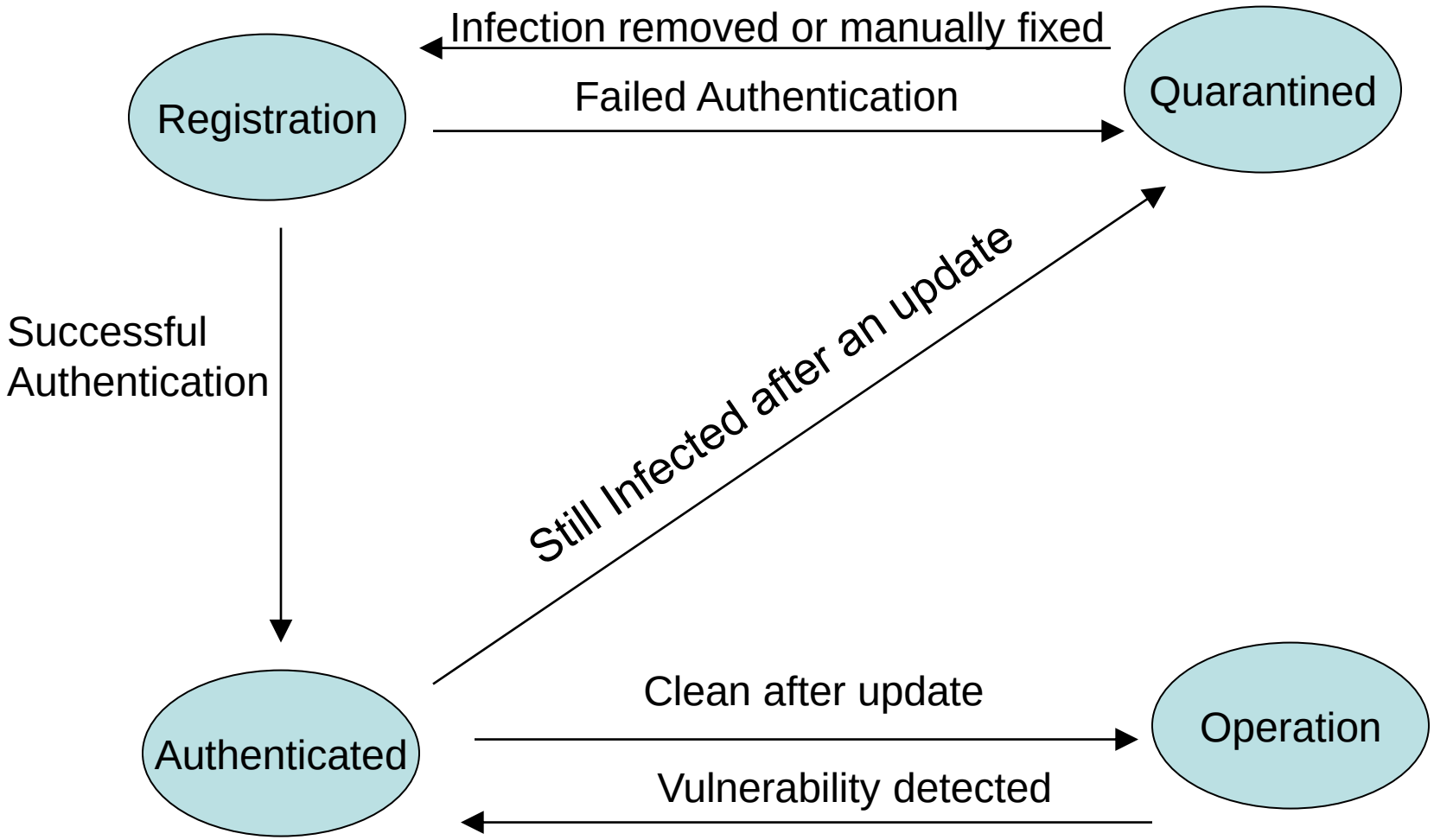


Problems with Current Architecture

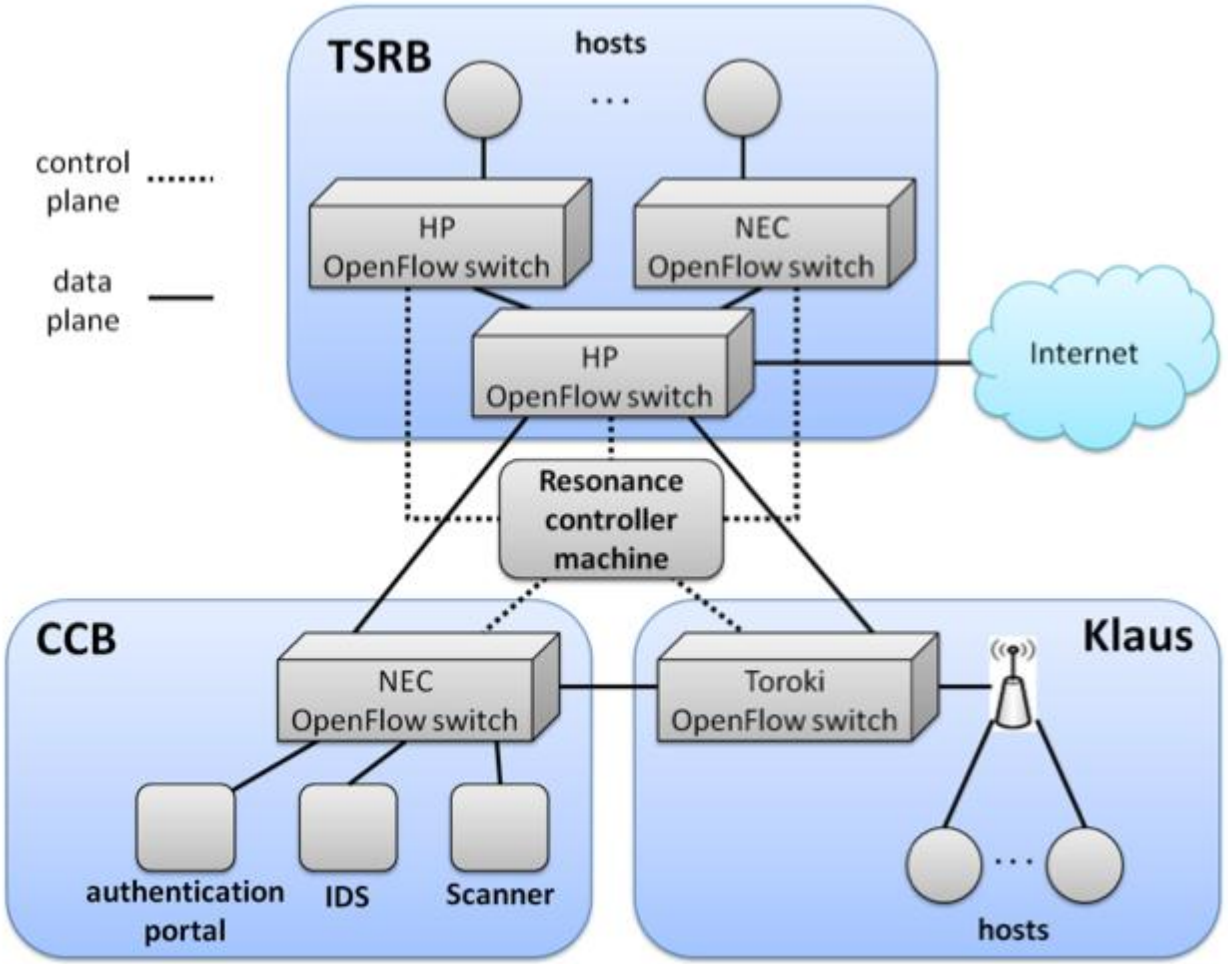
- Access Control is **too coarse-grained**
  - Static, inflexible and prone to misconfigurations
  - Need to rely on VLANs to isolate infected machines
- **Cannot dynamically remap** hosts to different portions of the network
  - Needs a DHCP request which for a windows user would mean a reboot
- **Monitoring is not continuous**

Express policies that incorporate network dynamics.

Handling Dynamics: State Machine



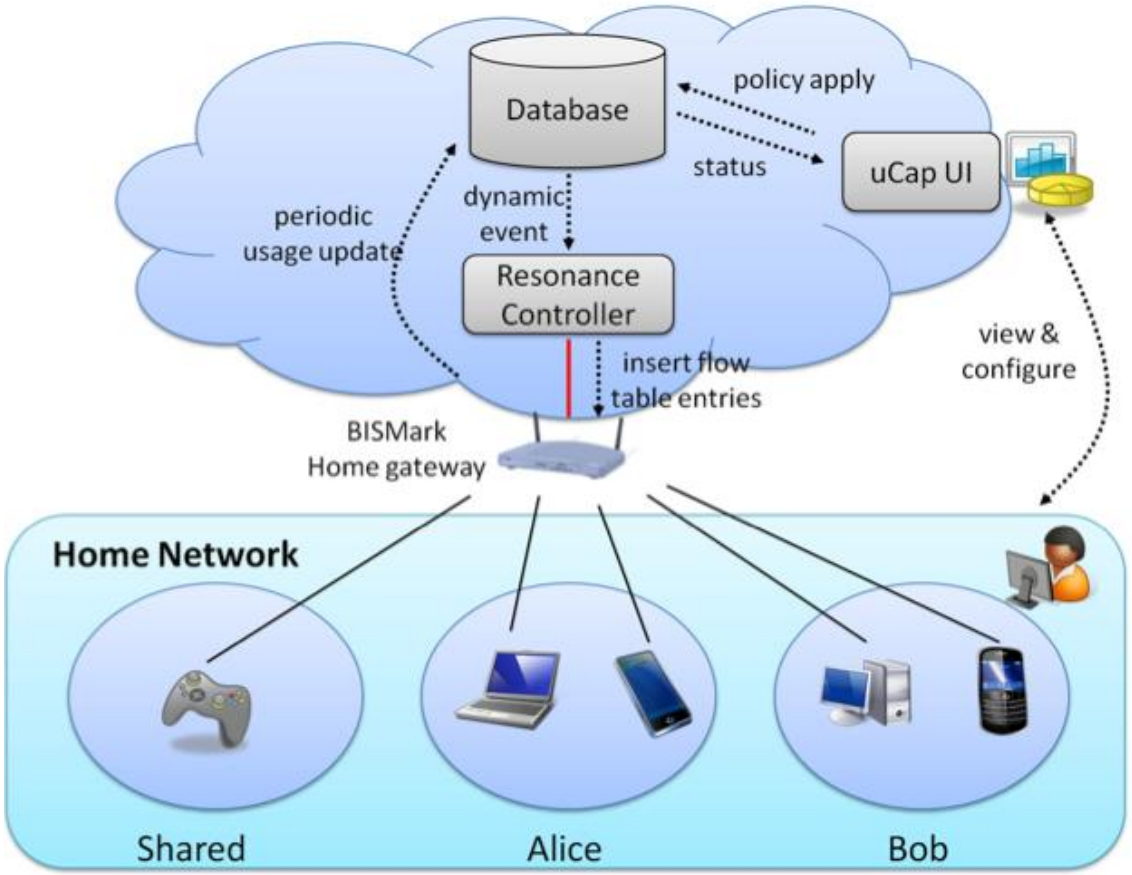
Georgia Tech Campus Deployment



Case Study: Usage Controls in Home Networks

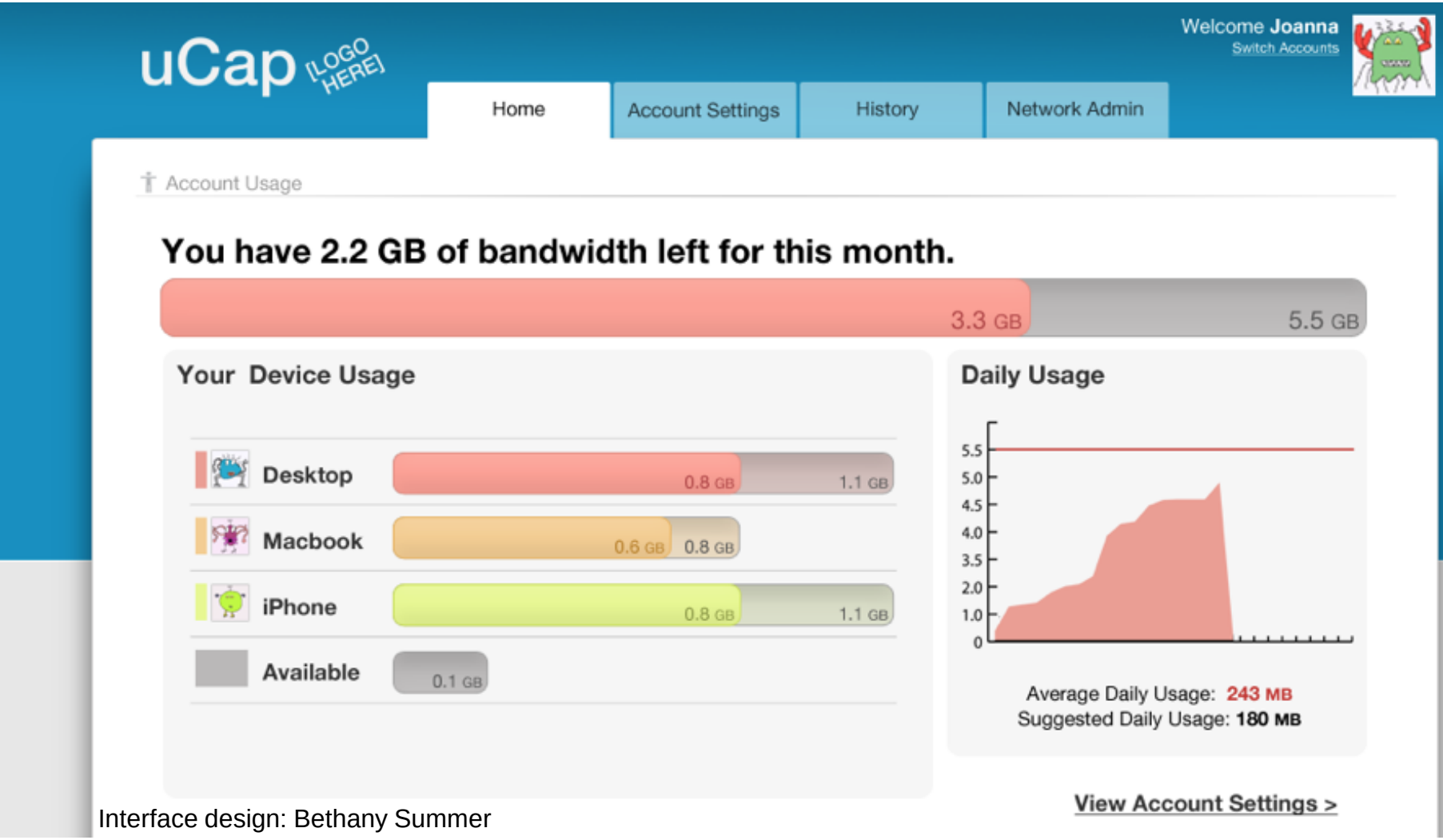
- Network management in homes is challenging
- One aspect of management: usage control
  - Usage cap management
  - Parental control
  - Bandwidth management
- **Idea:** Outsource network management/control
  - Home router runs OpenFlow switch
  - Usage reported to off-site controller
  - Controller adjust behavior of traffic flows

Usage Cap Management in Homes: Design and Implementation



- User monitors behavior and sets policies with UI
- Resonance controller (OpenFlow) manages policies and router behavior

Intuitive Usage and Policies



Interface design: Bethany Summer

Conclusion: Software-Defined Network Management

- Many problems result from the fact that configuration is **low-level** and **distributed**
- **Resonance:** Program the network from a logically centralized control point.
  - Higher-level configuration language
  - Handling of dynamic events, heterogeneity
  - Enables refactoring of function
- **Two case studies**
  - **Access control in enterprise networks**
  - **Usage control in home networks**