SDN and Network Virtualization

Innocent Obi + Sirui Lu

“The ability to master complexity is not the same as the ability to extract simplicity.’

Scott Shenker

The Road to SDN: An Intellectual History of Programmable History

Nick Feamster, Jennifer Rexford, Ellen Zegura

Overview

1. (Big Bang - 2000): Active Networking
2. (2000 - 2007) : Forces and RCP
3. (2007 - 2010) : Openflow and Network OS
4. Network Virtualization

Active Networking

- Rapid design and slow standards process
- API exposed resources on network nodes + custom func
- Programmability and the use of the **clean state** approach
- Capsule module: executable code carried in-band data packets
- Programmable router/switch mode: executable code established out-of-band
- Active networking became closely associated with capsule module
- “Installation of new data-plane functionality across a network using caching to improve efficiency of code distribution.”

Active Networking

- Technology push: advances in CS and DARPA funding
- Use pull:
  - ISP -> network ossification
  - third parties -> need for fine grained control
  - Researchers -> experimentation
- Unified control of middleboxes:the proliferation of middleboxes, firewalls, proxies,
- Programmable function, main emphasis on data-plane
- Demultiplexing: os -> runtime environments
- M/M: Java (lol) and performance requirements

Separating Control and Data

- Operators got fed up and wanted more control and ‘WE’ only cared because one research was like “yoooo… backbone network are kinda hard to manage and they need solution yesterday’
- Technology push: link speeds grew rapidly -> packet-forwarding in hardware. oops!
- Use pull:
  - ISP -> need for reliability and direct management of routing
- Logically centralized control to the plane: ForCES and RCP: complete removal vs BGP
- Distributed controllers
- Demultiplexing: os -> runtime environments
- M/M: emperor has no clothes (“fate sharing”)
  - Logic -> hardware and ‘scaling techniques”: limited visibility via OSPF areas and BGP route reflector

Separating Control and Data

Separating Control and Data

Data, **D**ata, **D**iscovery,…...

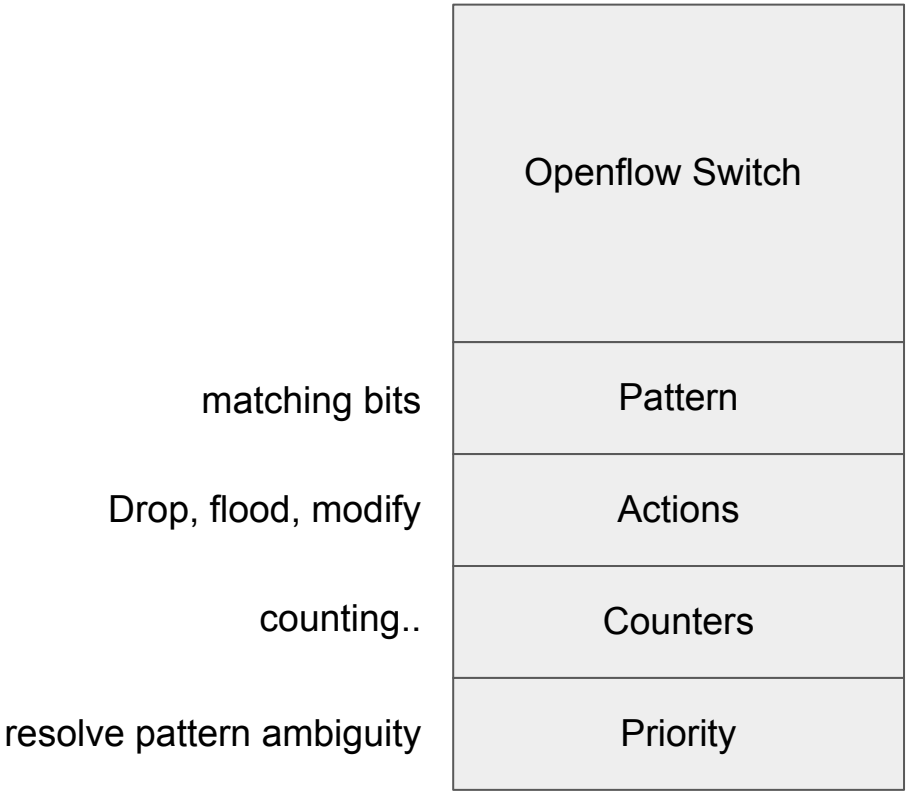
Separating Control and Data

Dissemination……

Openflow and Network OSs

- Operators got fed up and wanted more control and ‘WE’ only cared because one research was like “yoooo… backbone network are kinda hard to manage and they need solution yesterday’
- Technology push: merchant silicon chips (open APIs)
- Use pull:
  - Third party: hire programmers rather than pay for expensive hardware
  - Research: campus networks
- Generalized networking devices + function: not just IP (router, switch, NAT, ..etc)
- networkOS: representation of network topo and other control state

Openflow and Network OSs



Network Virtualization is not SDN

- SDN facilitate Network Virtualization
- “Abstraction of a network decoupled from underlying physical equipment” : VLANs, VPNs to overlays (tunnels + bespoke protocols)
- Technology push: merchant silicon chips (open APIs)
- Use pull:
  - Third party: hire programmers rather than pay for expensive hardware
  - Research: campus networks
- Generalized networking devices + function: not just IP (router, switch, NAT, ..etc)
- networkOS: representation of network topo and other control state

Discussion Responses

1. SDN ostensibly “simplifies” network design. In what ways is that true, and in what ways is that false

- Decoupling Logic / protocol
- Downtime in network routing updates
- Low risk /easy to deploy
- Raising the level of abstraction
- Future-proof (i.e network evolution)

- Complexity apocalypse
- Stability and scalability
- Usability (hard to program the devices)
- Complex dissemination path
- More risk to single-point failure
- Configuration could still be complex due to the complexity of the network environment

Discussion prompt

1. Does SDN smell fishy? What does the move towards SDN say about how the internet operates in practice and how it is described via RFCs?

2. What does SDN say about the practice of network design and engineering? Is it an art or a science? Can you manage complexity with art? Or science?

https://tinyurl.com/cse-550-sdn

Let’s talk about Reachability in Policyland

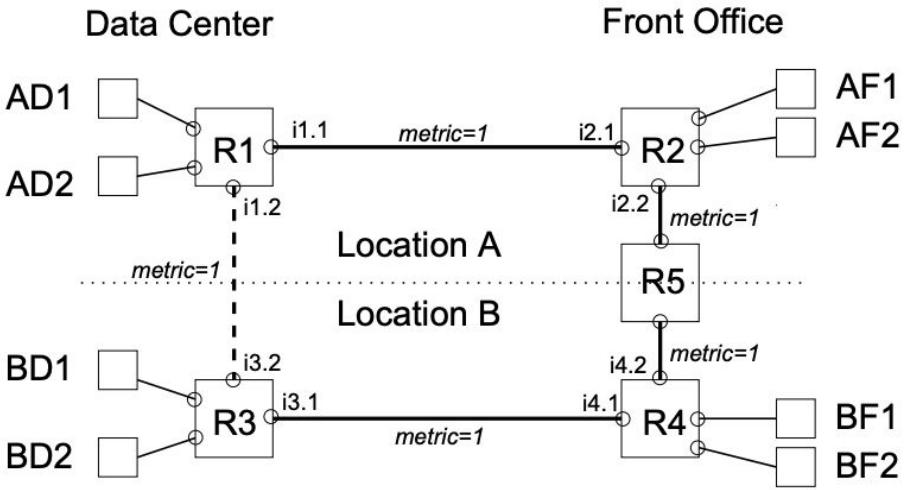


Figure 1: Enterprise network with two locations, each location with a front office and a data-center.

Problems + Challenges

- Data plane: handles data packets
- Control plane: implements routing algorithms
- Management plane: monitors network and data/control configurations

Everyone one is doing too much:

- Data: tunneling, access control, queuing
- Control: bottlenecks, uncoordinated,
- Changes are uncoordinated and breaking
- Requires manual configuration at routers (breeding ground for human error)

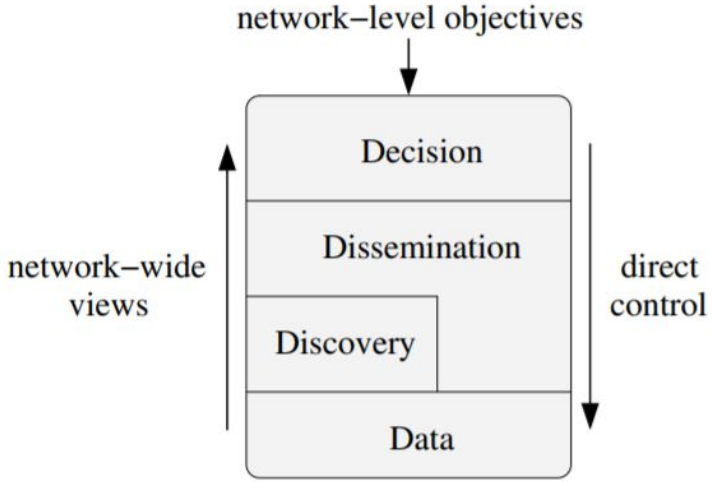
EVERYTHING is unwieldy, brittle, fragile, and duck-tapey.

A Clean Slate 4D Approach to Network Control and Management

Design Principles of 4D

- Network-level objectives
  - Should: do not allow hosts in subnet B to access the servers in subnet A
  - Rather than: the router C should filter some packets with some criteria
- Network-wide views
  - Network-wide views of topology, traffic and events
  - Timely, accurate
- Direct control
  - Have the ability and the sole responsibility for setting all the state in the data plane.
  - The decision logic should be decoupled from the data plane.

4D Network Architecture



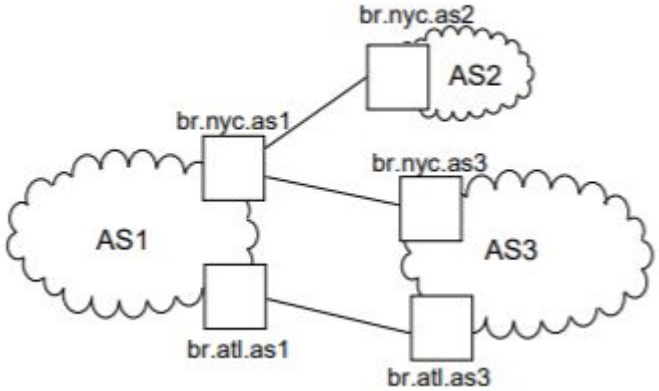
- Decision plane
  - **All** decisions for the network, logically centralized
- Dissemination plane
  - Robust and efficient communication substrate connecting the decision plane and routers/switches
- Discovery plane
  - Discovering the physical components of network, manage the relationships between them.
  - Box-level discovery / neighbor discover / lower-layer link characteristics
- Data plane
  - Handling individual packets based on the output by the decision plane.

Peering policies in transit network example

Network-level objective: Do not allow transit service for d

Network-wide view: br.nyc.as1 & br.atl.as1 are neighboring border router

Direct control: all neighboring border router should have a packet filter.



Advantages and challenges

- Separate networking logic from distributed system issues
- Higher robustness
- Better security
- Accommodating heterogeneity: customized solutions for different networking backbone
- Enabling innovation and network evolution

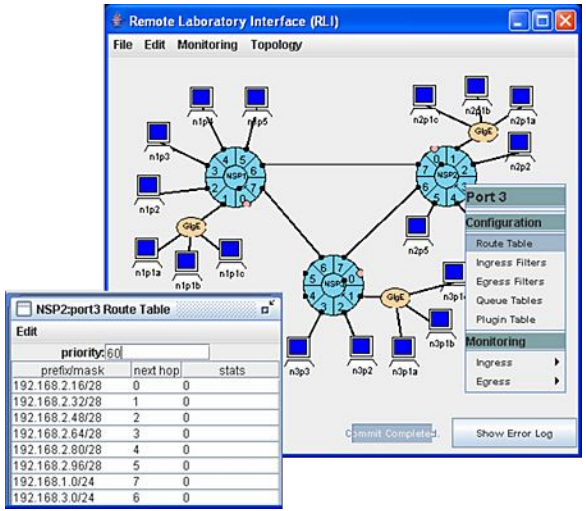
- Complexity apocalypse, stability failures, scalability problems, response time, security vulnerabilities

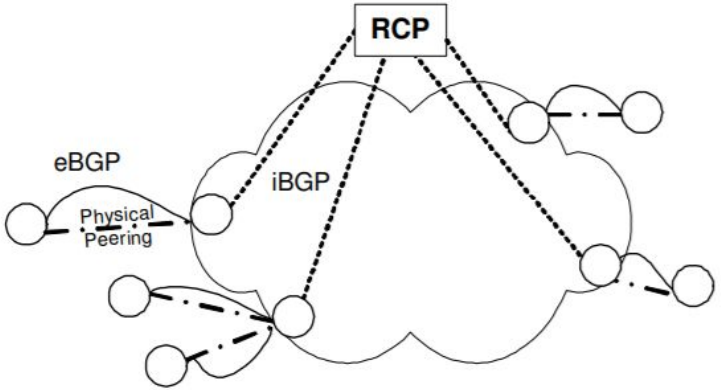
Future works

- Decision plane: algorithms satisfying network-level objectives
- Dissemination plane: connecting decision elements with routers/switches
- Discovery plane: bootstrapping with zero pre-configuration beyond a secure key
- Data plane: advanced data-plane features

Evaluation

Experimental platforms and opportunities for field deployment





Discussion Response

3. Engage some of the challenges for 4D architecture discussed in the paper. Can you propose approaches to address one of the challenges? What other challenges/limitations can you think of?

- Design complexity in the abstraction
- Single lost of failure -> backup behavior (iBGP relay routers)
- Hierarchy of decision plane machines to control different regions of the network.
- Resiliency and Security: risk in both the data plane and control. Formal verification for both the data and control plane
- CAP
- Hybrid approaches: the extreme proposal of 4D is great but impractical / impossible.

Discussion Prompt

1. Design an SDN-like system for managing intra-domain routing (e.g., BGP). Discuss the positives and negatives of your design?
2. Can you think of other systems concepts that could be incorporated as new/different design principles of SDNs?

https://tinyurl.com/cse-550-sdn

View from the inside

ISP

End user

Network Engineer / manufacturers /vendors

Hackers

Regular Computer Scientist

Government