Hard Problems in SDN

Rob Sherwood
SDNRG – Atlanta, November 2012
Outline

- Level Setting and Framework
- **Sexy problems** (that I’m not going to talk about)
- **My (not so sexy) Problems**
  - Forwarding Memory Abstractions
  - Mixed OpenFlow + Non-OpenFlow Networks
  - Controller/Application (“north bound”) API distractions
- Food for Thought
Level Setting

• Who am I?
  • Day job: Principal Architect for Big Switch Networks
  • Side job: Chair of Open Networking Foundation “archWG”
  • Background: Research, campus networking/sysadmin

• (Not so hidden) Agenda
  • Try to motivate researchers to focus on the right problems
    • So we can all work together, increase impact
  • (re)Socialize working code for problem solving in IETF/IRTF
    • The real purpose of SDN is that anyone can write code again
  • Intentionally inflammatory statements – fodder for panel!
Problems Framework

Difficult Problems

Mixed OF and Non-OF Networks

Forwarding Memory Abstractions

Policy Transactions

Consistent Scalable/State Distribution

Precise/Scalable Controller Testing

Provably Correct Policy Languages

Standardized Controller/Application API

Rob’s Guess on Time to Impact

- Short
- Medium
- Long

Intellectually Attractive (Sexy!) Problems

Observation: my problems are anti-correlated with Sexy 😊

Short Term Distraction!
Cliff Notes Version of Framework
All of these are great ideas... that are hard to use right now

- Policy Transactions:
  - Enforce policy only at first hop/ingress
  - Transactions are trivial if only one place to update

- Provably Correct Policy Languages
  - Need to rewrite a lot of code – not practical
  - If non-networking is an example, domain specific wins

- Consistent/Scalable State Distribution
  - Very hard from engineering perspective
  - Largely solved from research perspective – e.g., routing protocols, distributed agreement protocols, etc.
Not So Sexy Problems
Potential for short-term impact

Technical:

• Forwarding Memory Abstractions
• Mixed OF and Non-OF Networks
• Testing: lots of good work being done, I won’t add to it here

Non-Technical:

• Convincing people that the controller/application API is a short-term distraction
Forwarding Memory Abstractions

- Lots of forwarding memory types
  - CAM, TCAM, LPM, FPGA, network processors, CPUs, tree mem

- Every type of memory has different trade-offs
  - Latency, capacity, width, programmability

- Lots of potential for caching and memory hierarchies
  - Lots of research shows packets of Zipf-like lookups

- “Real” boxes use complex combinations of forwarding mem

- Hard question: how do we abstract these memories and their capabilities?

- How does a controller reason about how to best use them?
Mixing OpenFlow and Non-OF
Lots of work and lots of room for optimizations

- Interoperability is easier
  - Needed for incremental deployments: meet the bar
  - But still – litany of protocols to re-implement

- But optimizations are possible!
  - Needed for incremental improvements: exceed the bar
  - Big potential for bootstrapping new protocols
  - Need to understand and “out fox” Non-OF control logic

- Examples:
  - Use multiple paths between STP and OpenFlow networks
  - Re-advertise Weird SDN routing protocols via BGP
This is (currently) a distraction: solve other issues first
- Please avoid and we’ll discuss in more detail in $N$ years

Technical: How do you standardize an API before the apps?
- New apps are coming all of the time in SDN – that’s the point!
- Let’s build some running code together: platform + apps
- Open source + de facto standards == path to experience
- Still need to understand what works and what does not

Non-Technical: App portability doesn’t exist anywhere else!
- No standard exists for PC vs. Linux vs. Mac Apps
- No standard exists for Android vs. iPhone vs. XXX Apps
- Can you name one? Where the standard was created before a de facto standard emerged?
Conclusions and Food For Thought

None of this is my unique insight.

• Research is really good at solving problems ~2 years out

• Industry seems tunnel-focused at ~6 months out

• Technology moves faster than 2 years, but slower than 6 months
  • How do we best bridge this gap?

• “Skate to the puck”
  • But how far out?
  • What is an acceptable miss rate?

• SDN is an unprecedented way for researchers and operators to become directly involved in solving real problems now
Notes

- Policy transactions
- Provably correct languages
- Testing
- SDN state distribution