SDN in GENI thoughts (hastily assembled and poorly articulated)

Ilia Baldine ibaldin@renci.org

renci

RESEARCH \ ENGAGEMENT \ INNOVATION

SDN promise

- SDN promises to externalize policies and mechanisms traditionally fixed/hidden inside switching and routing hardware via well-defined remote APIs
  1. Packet forwarding decisions
  2. Packet queue definitions, arbitration and scheduling algorithms
- A lot of interesting things can and are being done with just #1
  - #2 is where things get really interesting
  - An interesting area of research for declarative and verifiable policy specification (for queue arbitration and packet scheduling)
- SDN – 'Active Networks' by another name
  - See 'Active Bridging'

renci

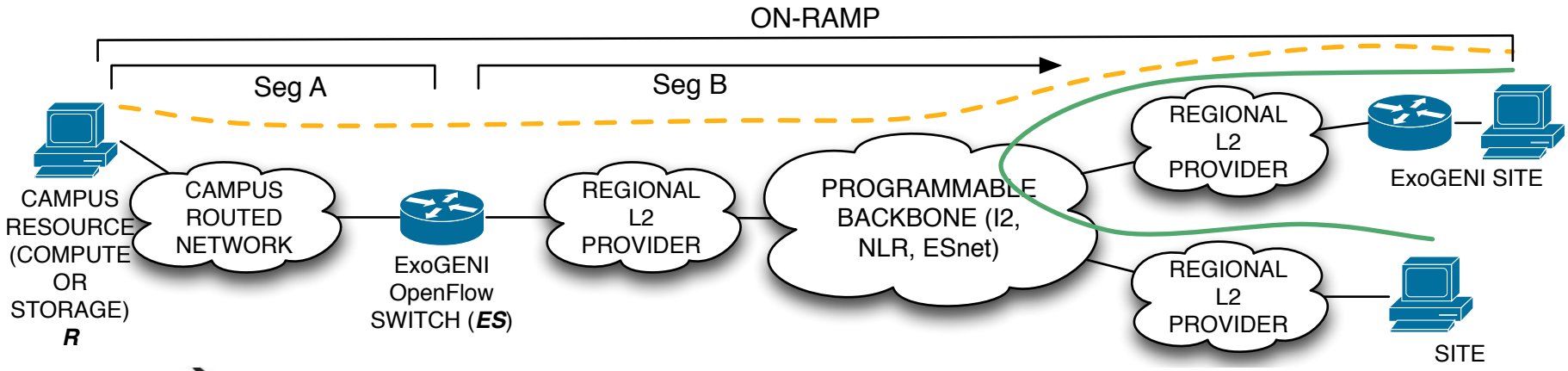
GENI promise

- Deeply programmable, flexible cyber environment for repeatable experimentation
- Compute, storage and networking resources provisioned in concert (as slices) to support repeatable experimental activities
- How/where does SDN fit in?

renci

SDN GENI Uses

- As an experimenter tool:
  - Give the experimenter control over packet forwarding within the slice
- As a slivering mechanism:
  - Provide a flexible mechanism for GENI aggregate operators to create virtual network slices for experimenters
- As an on-ramp:
  - Serve as an on-ramp for traffic in/out of the slice.
  - E.g. permit campus operators to selectively steer their traffic into the slice (and vice versa) with appropriate authorization controls



renci

Where we are today

- OpenFlow as an SDN instantiation used for forwarding control by experimenters today
  - Mesoscale deployment
- Some network providers are starting to implement network virtualization via OpenFlow
- Both require careful attention because:
  - What about performance isolation in the dataplane?
  - What about performance isolation in the API command channel?

renci

SDN in ExoGENI racks

- Each rack has a 48 port 10Gbps OpenFlow enabled switch in every rack
  - Currently working in OF mode
  - Awaiting hybrid support from the vendor
- ExoGENI/ORCA slicing is based on VLANs
- Uses Flowvisor and floodlight to mimic the behavior of a learning switch
- Uses ORCA to create VLAN-based slices with user-specified controllers
- Uses FOAM to support flowspaces based on other packet header fields
  - To e.g. run mesoscale experiments

renci

GENI vs. the rest of the world

- Telco equipment manufacturers don't like open-ended compatibility
  - They typically qualify their equipment for interoperability with specific controllers
- In GENI anyone can build a poorly-behaving controller that can e.g. hog the CPU of the switch, or the flow table, or use the flow table that is not appropriate for this type of flow
- These are resources that until now were hidden and managed by the switch firmware
- SDN opens these resources are to competition
  - This can/does affect not only the experimenter's slice, but other slices on the same datapath(s)
  - How do we give aggregate operator the tools to monitor and stop them?
- In the long run, SDN/OpenFlow products will become more robust, but
  - Considering the small size of the GENI market we should think about helping the manufacturers

renci

Protecting GENI SDN aggregates from experimenters

- One possible solution: harden flowvisor to provide some performance isolation of controllers from one another
  - Have two performance envelope definitions for flowvisor:
    - one for a datapath (provided by the manufacturer)
    - one per slice (provided by the operator)
- Those envelopes define performance limits for controllers over slices within the datapath, like
  - Limit the number of commands the controller can send
  - Limit timeout for flows (no less than X sec)
  - Validate some semantics of the commands sent based on switch architecture
- Saw an encouraging demo of this concept from I2

renci

Using SDN for network virtualization

- Typical approach to connecting racks to national backbones is via a limited pool of static vlans (or otherwise named paths)
  - This limits the number of slices that can operate concurrently
- Using SDN more slices can be multiplexed onto the same paths
  - But you lose performance isolation properties
  - That is if you had any to begin with, since static vlans are typically best effort

renci

Another possible feature

- Allow operator install ‘default’ flow rules into the slice that are always honored, not visible to controller
- Example –
  - OF switch has a slice with tag A going in ports X and tag B going in ports Y.
  - Tag A should be translated into tag B always, without the knowledge of the user’s controller
  - Controller can do anything, as long this translation is respected by the switch
- Can help in stitching as well as policy control over traffic

renci

Questions

- How critical is performance isolation to GENI experimenters?
  - I know it is to those I work with, but it is a small sample
- Lack of performance isolation undercuts repeatability of experiments: GENI *raison d'etre*
- SDN doesn't solve this problem, but makes it more complex
  - Dataplane performance isolation is still needed
  - Adds the need for performance isolation in the SDN API command channel

renci