GENI OpenFlow Service Abstraction

Tim Upthegrove, GPO

October 22, 2014

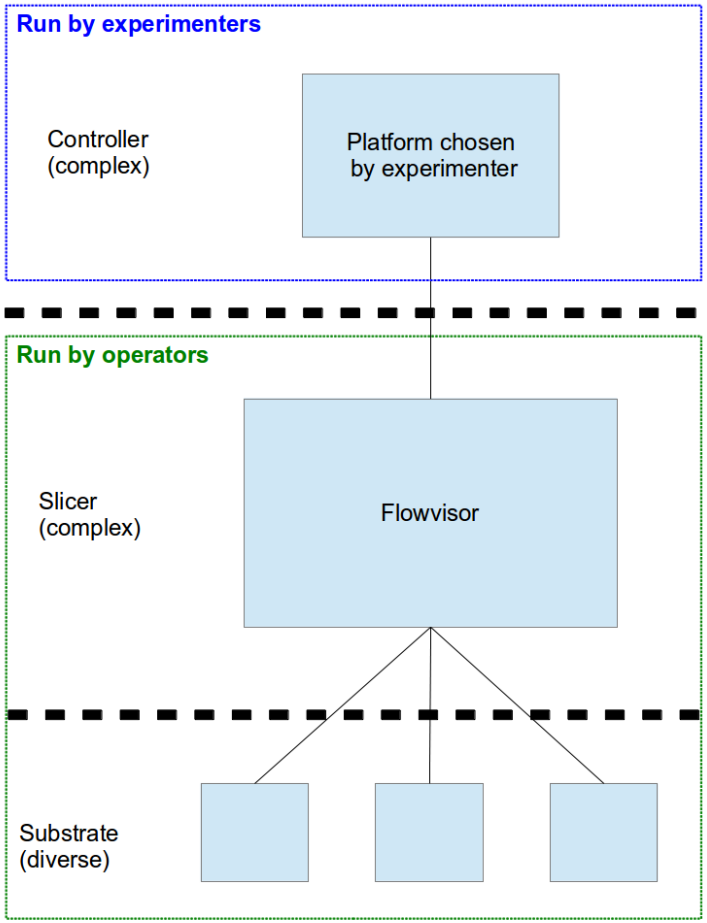
Sponsored by the National Science Foundation

Overview

- Existing solution
- Current problems
- Proposed solution
- Progress

Existing Solution

- Operators run and install slicer (Flowvisor) for OF aggregate
- Flowvisor can do a lot:
  - Can slice by anything
  - Flowmod augmentation
- Operators run a diverse substrate of OpenFlow gear
- Experimenters deal with complexity of substrate and limitations of slicer



Current Problems

- The substrate is intentionally diverse and necessarily complex, but it shouldn't be so difficult to use
- Slicing layer is more complex than necessary, and it is difficult to update GENI-wide
- Researchers are working around the same GENI-specific problems that have nothing to do with their research...
- And aggregate developers with OpenFlow substrates are solving some of those same problems!

Complexity Sausage

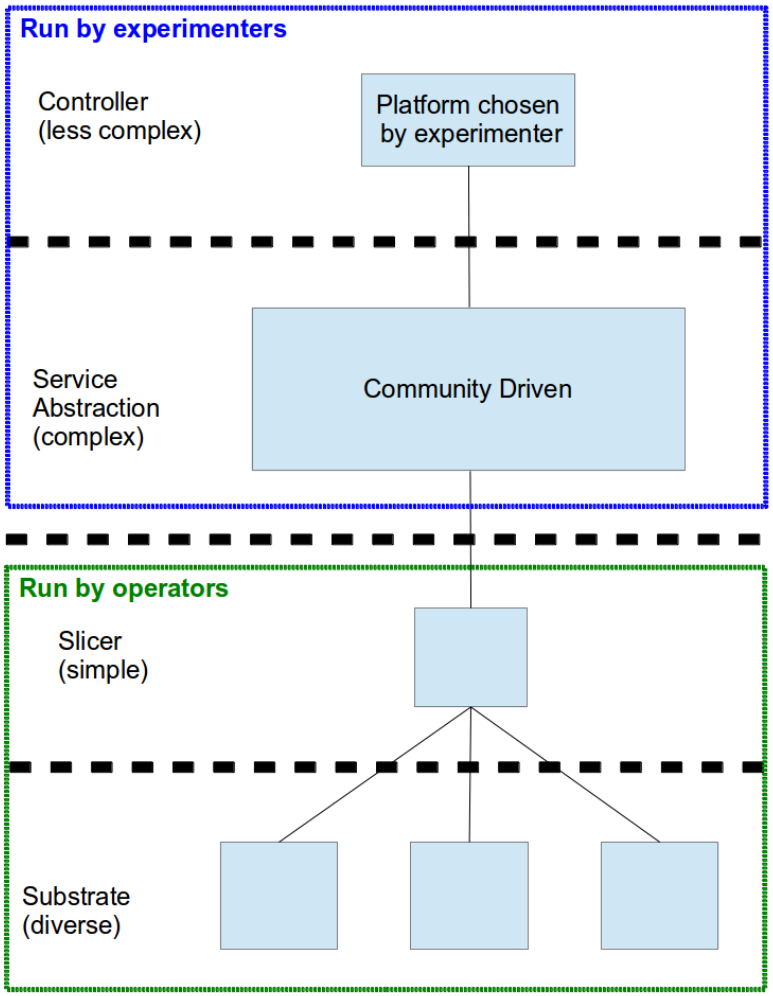
RFC 6670 Section 3.4:

“... simplification in one element of the system introduces an increase (possibly a non-linear one) in complexity elsewhere. This creates the 'squashed sausage' effect, where reduction in complexity at one place leads to significant increase in complexity at a remote location.”

Let's put complexity where stability is not critical and where primary stakeholder has direct control over it.

Proposed Solution

- Operators run and install simpler slicer with fewer requirements and features
- Add community driven service abstraction layer for solving GENI-specific issues
- Operators run a diverse substrate of OpenFlow gear
- Experimenter can **focus on research problem**



Proposed Solution (cont'd)

- **Overarching goals**: Hide complexity where it is not required and increase stability
- Potential features of service abstraction layer
  - Normalize experimenter view of network
  - Layer of indirection for compilers or translators
- Benefits of splitting service abstraction layer from slicer
  - Slicer will require fewer GENI-wide updates
  - Service abstraction layer can develop at its own pace
  - Service abstraction layer can be used to develop GENI services regardless of choice of slicer
- The service abstraction layer is **not required** for experimenters, developers, or operators who want to deal directly with substrate

Slicer Progress

- Requirements
  - http://groups.geni.net/geni/wiki/OpenFlow/Slicer/Requirements
  - Still working on finalizing these before making a formal test plan
  - Allows us to be flexible on the slicer software
  - Defines what the service abstraction can expect
- Inputs welcome!
- Looking at software options compared to requirements to make sure they are realistic
- One big open question from a first round of feedback from operators...

Key Slicer Question

- Should we drop unmatched traffic by default?
  - Similar to how later OF versions work
  - Experimenters can still install rules to forward unmatched traffic to their controller
- Pros:
  - Simple approach that improves stability
  - Simplifies requirements (and software)
- Cons:
  - Different from what we do today
  - Breaks ability to use out-of-the-box controllers

Service Abstraction Progress

- Capturing substrate capabilities
  - We have a lot of this from public manuals
  - There is still a lot missing
  - Need to define a process to make this viable in the long term
    - Likely requires resource owner input
    - Changes occur if substrate updates occur
- Documentation generation
- Some basic services being developed

geni
Exploring Networks of the Future

Questions?