The ProtoGENI RSpec

Robert Ricci and Jonathon Duerig

RSpec Workshop

June 25, 2009

Where We Are

- Working prototype RSpec
- Supports components, interfaces, links
- Used to create slivers:
  - Raw PCs, VMs, VLANs, tunnels
- Expressed as an XML schema
  - http://www.protogeni.net/trac/protogeni/wiki/RSpec
- Slice Embedding Service that understands it
- Under development
  - Extensions using NVDL
  - Cross-aggregate RSpecs

Lifecycle

- Progressive annotation
- User creates **request** (bound or unbound)
- Passes to a Slice Embedding Service
  - Annotates with physical resources selected
  - Maybe more than one
- Gives to CM
  - CM signs (**ticket**)
- **Manifest** returned by CM
  - Adds details like access method, MACs, etc.

Four Types

- Similar, but not identical
- Advertisement: “Catalog” or “Classifieds”
  - Published by Component Manager
- Request: “Purchase Order”
  - Constructed by user (maybe from advertisement)
- Ticket: “Receipt”
  - Signed (special type of credential)
- Manifest: “Packing Slip”
  - Returned by CreateSliver()

RSpec Design Principles

- Descriptive data structure
  - Mapping between requested sliver and physical resources
- Contains information
  - About what CM provides
    - Describe the pen, not the novel
  - That the client needs to select resources
    - Additionally, how to use components in manifest
- Progressively annotated

Important Ideas: Identifiers

- Advertisement must have component IDs
- Request must have virtual IDs
- A *bound* request has both, creating a mapping
- Identifiers are URNs (GMOC proposal)
- A sliver is uniquely identified by slice ID + virtual ID + CM ID

Paths to Constructing a Request

- Direct
  - Cut-n-paste from advertisement
  - Add virtual IDs
- Indirect
  - Start with desired topology
  - Get help filling in component IDs
- Combinations of these styles possible

Important Ideas: Mapping

- Each requested node is mapped to a single physical node
- A requested link may be mapped to multiple physical links
  - Path or graph

Important Ideas

- Separate Schemas
  - Ads << Requests < Tickets << Manifests
- Extensions (in development)
  - Each extension lives in its own namespace
  - Use NVDL to provide modular verification
  - Extensions are **ignorable** (this may change)

Hard Problems

- Allocation of components the user didn't explicitly ask for
  - Measurement devices
  - Sub-components (eg. NetFPGA inside PC)
  - Firewalls
  - Traffic Shaping
- Multi-level Hierarchies
  - Network layers
  - Virtual nodes

Hard Problems (2)

- Splitting/Combining RSpecs
  - Robust semantics
- Special Nodes
  - The Internet
  - Wireless Networks
  - NAT/Firewall
- Coordination across aggregates

Coordination Across Aggregates: The Problem

- Both (or many) ends may need to share information
  - Eg. tunnel endpoints
- Ordering/timing may be important
- Negotiation may be necessary
  - Eg. session key establishment
- Some are transitive problems
  - Eg. VLAN #s (unless translation possible)
- Assumption: Cross-aggregate links are established by endpoints within each aggregate

Coordination Across Aggregates: Design Space

- A) Client negotiates with each CM
  - RSpec is the medium
- B) CMs coordinate among themselves
  - Using a new standardized control plane API
  - RSpec *could be* medium
- C) Untrusted intermediary negotiates for client
  - Intermediary has no privs. that client does not
- D) Trusted intermediary negotiates for client
  - Pre-established trust betw. intermediary and CMs

Coordination Across Aggregates: Our Plan

- Hybrid of B and D
- CMs negotiate two-party arrangements directly
  - Eg. Tunnels
- Trusted intermediary negotiates multi-party
  - Eg. VLANs
  - Trusted authority picks VLAN #
- Client is oblivious
  - Only CMs talk to intermediary
  - All knowledge about necessary negotiation lives in CM

http://protogeni.net

Manifest

- Contains information not necessary for selection, but needed for use
- Might contain some info not existing at time of request
  - Virtual machine identifier
  - MAC address of virtual interfaces
  - VLAN tags
- CM may let experimenter modify