Emulab Federation Preliminary Design

Robert Ricci
with Jay Lepreau, Leigh Stoller, Mike Hibler

University of Utah

USC/ISI Federation Workshop
December 11, 2006

"Many Masters" Model

- No global master
- Emulabs make peering agreements
- "Home" testbed for
  - Users
  - Projects
  - Experiments

Identifying Users

- Users identified by UUID (X.667) rather than login name
  - Implementation begun
- Login names *may* be unique per project
- Web login now done with email address
  - Already done on Utah Emulab
- We may do this for projects too

Starting an Experiment

- User submits NS file to one testbed
  - This will be the experiment "master"
  - Does this have to be project or user master? Hope not, but that poses unsolved challenges.
- Master testbed does most setup work
- Each remote testbed boots its part, then reports in
- Experiment controlled through the master testbed

Approving Remote Experiments

- Experiments come with "certifications"
  - Really: Signatures
- Each Emulab has a list of certification authorities it trusts
- Experiment accepted if signed by an appropriate authority
  - In the future, more complex policies
- Allows for a large set of acceptance policies

Some Possible Certifiers

- Testbed-wide
  - "Allow all experiments started at DETER"
- Project-wide
  - "Allow all experiments in project tbres from Utah's Emulab"
- Experimenter
  - "Allow Steve Schwab to run experiments"

More Possible Certifiers

- Vetting committee
  - "Allow experiments that have been OKed by the DETER board"
- Other criteria
  - "Allow projects that Rob has verified as being classes"

Federation API

- Done with XML-RPC
- Generally, exchange "virtual to physical" mappings
  - Send reserved table, not virt_nodes
  - Send vlans table, not virt_lans
- Precedents in elab-in-elab support

Dealing With Version Skew

- General model
  - Node boot managed by local testbed
  - Everything else managed by master
- Leaves a somewhat narrow waist between virtual and physical
  - Hope: This is less susceptible to skew
- Version every interface
  - Allows testbeds to innovate

Proxied Services

- tmcd (virtual nodes)
- Event system (PlanetLab portal)
- Console (capture)
- VLAN creation (elab-in-elab)
- Power cycling (elab-in-elab)

- Frisbee done by pre-staging disk image, then running locally

New Policy Knobs Necessary

- Prefer local users to remote
- Limit resource use by remote users
- Policies based on "threat level"
- These may also apply to single Emulabs
- Add more policy knobs as we go

Pre-emptive Model

- Allow high priority experiments to force swapout of lower-priority experiments
- assign has some features to help
- Stateful swapout will make this more painless

“Library” Model

- You “check out” lease nodes for some period of time
- You can renew as long as no one has asked for those nodes
- Probably coupled with a reservation system

Control Network

- Maintain: All nodes in an experiment can directly communicate
- Impediments:
  - Unroutable control networks
  - Duplicated private IP addresses
- Solution: Tunnel
- Nodes get an extra control net IP alias

Filesystem

- First stage: NFS mounted from project master (TCP)
- Ideal: Use a real distributed filesystem
- But, we have to make sure all clients can mount it, or we can proxy them
- Possibly add special support for single writer, multi reader data

Local Administrators

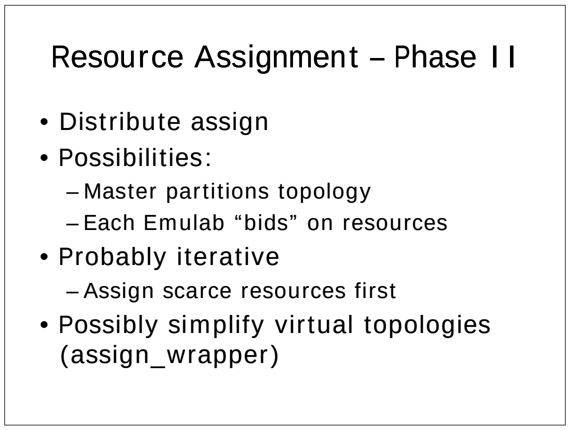
- Can see remote experiments they are hosting
- Can force swapout of these experiments
- Can still easily find out about who is responsible for an experiment

Resource Assignment – Phase I

- Experiment master runs assign
- Each federated testbed needs physical topologies of others
- Loosely consistent reservation state
  - We already support optimistic allocation
- assign should scale for a while
  - Scales with number of pclasses
  - Utah: 66
  - Six testbeds (1120 nodes): 133

New assign Features

- “Typing” for cross-emulab links
- Support for links with non-zero latency
- Generalize switch notion to handle VPN boxes
- Fuzzy latency/bw allocation
- New XML file format (some simplified version of rspec)



Resource Assignment – Phase II

- Distribute assign
- Possibilities:
  - Master partitions topology
  - Each Emulab “bids” on resources
- Probably iterative
  - Assign scarce resources first
- Possibly simplify virtual topologies (assign_wrapper)