emulab: A Public Emulation Platform for Research in Distributed Systems and Networks

Jay Lepreau, Chris Alfeld, Shashi Guruprasad, Mike Hibler, Mac Newbold, Rob Ricci, Leigh Stoller, Brian White

University of Utah

Until now there have been three environments in which to perform experimental research in distributed systems and networking: simulators, small-scale static testbeds, and live networks—each with obvious strengths and weaknesses. Over the last two years we've built a new type, a scriptable "Internet Emulator," which complements and helps validate existing environments. Today's instrument, a time- and space-shared 328-node facility, is used by many institutions.

The major leverage emulab provides is ease of use while providing extreme configurability of the network topology, the link characteristics, and the node software. The trick is to accomplish both at the same time. We manage to provide fast tools in a simple or invisible manner for the common cases, while still allowing escape to the raw hardware. emulab also offers extremely precise and detailed measurement, but so far this has not been our users' emphasis. We expect that once we develop tools that make it simple to gather detailed network and host data, emulab's measurement aspect will become more important.

ns compatibility: ns simulator scripts drive emulab's configuration and high-level execution. This language provides familiarity, allows a graceful transition from simulating to emulating, and leverages existing ns tools, e.g., visualization and topology generation.

Virtualized and consistent names for everything (nodes, network interfaces, IP addresses) are one key to ease of use. Doing so also allows entire "experiments" to be "swapped out"—and back in to different physical resources. A randomized algorithm underlies all emulab resource allocation, solving the NP-hard problem of mapping virtual resources to physical ones.

The "programmable patch panel" is a set of ethernet switches with the commonly available "Virtual LAN" support. VLANs allow arbitrary topologies and isolate simultaneous emulab users.

Future and ongoing work: Building emulab poses numerous challenges, both research and engineering. In the research realm, for example, we have plans to extend it to wireless and even mobile nodes. Another challenge arises from our goal of making the boundary between simulation and emulation a flexible one. We are extending the types of experiments and network models that can "transparently" move between a simulated and an emulated environment. In other words, the same input script would cause a simulation run when fed to ns; if instead it's fed to our tools, it is compiled into equivalent hardware and software state on the testbed and executed. Currently, we can do this for network topology, link characteristics, and CBR traffic generation.

Portability and distribution: The underlying software (which is open source) is portable to other switches, and four other universities as well as a large corporate are deploying their own testbeds using our software. emulab is also an educational tool: external users are already using ours for mainstream class projects, and the new "emulab"s are primarily targeted at instruction. All that is needed to allow a generic PC lab also to function as a small, modest emulab is a separate low-end ethernet interface on each PC, and a reserved PC to function as the master server. Finally, plans are afoot to extend the emulab software to handle a distributed set of testbeds with flexible local/global resource allocation. Some local testbeds may have unique hardware resources, such as special routers or programmable switches.

emulab as a computational grid: While emulab has been primarily used for research in networks and distributed systems until now, it is certainly an excellent facility for studying
traditional computational grid applications namely, i) distributed supercomputing, in which many grid resources are used to solve very large problems, ii) high-throughput, in which grid resources are used to solve large numbers of small tasks, iii) on-demand, in which grids are used to meet peak needs for computational resources, iv) collaborative, in which grids are used to connect people across administrative domains, and v) data-intensive, in which the focus is on coupling distributed data resources. The emulab could be used both as a large LAN cluster or an emulated WAN cluster to study such applications. emulab provides a controlled environment to study the impact of network topology and characteristics on grid applications. Another important use of such a facility is to be able to test any grid middleware implementations in the OS over a reasonably large-scale emulated, yet isolated network with an Internet-like topology before deploying it in the public Internet. This mitigates the risk of “implementation bugs” causing runaway behaviour in the Internet. An example study could be “How frequently should checkpointing be done when the grid application runs on a LAN cluster versus a WAN cluster with varying network topology and characteristics?” The facility is therefore a grid enabler.

The formal verification research group at the University of Utah recently used emulab as a computational grid to perform parallel model checking. Model checkers are inherently highly memory and compute intensive and thus suitable computational grid applications. They ported the parallel “Active Messages” based Murph model checker to use MPI. Using a simple 100Mbps LAN cluster configuration, the experimenters were able to achieve superlinear speedup to verify a restricted model of the Itanium processor shared memory consistency protocol. The above model has a very large state space and is not practical to verify on a sequential model checker. The ability to schedule and run batch experiments was one of the important features of emulab that this user found extremely useful.

Use: If you’ve got legitimate use for it, sign up at http://www.emulab.net/ — it’s free. We can also help you build one of your own.
How To Use It ...

- Submit ns TCL script via web form
- Relax while emulab ...
  - Generates config from script & stores in DB
  - Maps specified virtual topology to physical nodes
  - Provides user accounts for node access
  - Assigns IP addresses and host names
  - Configures VLANs
  - Loads disks, reboots nodes, configures OSes
  - Yet more odds and ends ...
  - Runs experiment
  - Reports results
- Takes ~3 min to set up 25 nodes

Who's Using It?

- 22 projects have used it (18 external)
  - Plus several class projects
- Two OSDI’00 and three SOSP’01 papers
  - 20% SOSP general acceptance rate
  - 75% SOSP acceptance rate for emulab users!
- More emulabs:
  - Planned or in progress: Kentucky, CMU, Duke, Cornell, HPLabs
  - Others considering
- Available today at www.emulab.net
emulab: A Public Emulation Platform for Research in Distributed Systems and Networks

Jay Lepreau, Chris Alfeld, Shashi Guruprasad*, Mike Hibler, Mac Newbold, Rob Ricci, Leigh Stoller, Brian White

University of Utah

http://www.emulab.net/

(*) = here at HPC

Why?

- "We evaluated our system on five nodes."
- "Job talk from university with 300-node cluster"
- "We evaluated our Web proxy design with 10 clients on 100Mbit ethernet."
- "Simulation results indicate ..."
- "Memory and CPU demands on the individual nodes were not measured, but we believe will be modest."
- "The authors ignore interrupt handling overhead in their evaluation, which likely dominates all other costs."
- "You have to know the right people to use the cluster."
- "The cluster is hard to use."
- "<Experimental network X> runs FreeBSD 2.2.x."
- "February's schedule for <Testbed Y> is..."
- "<Network Z> is tunneled through the Internet"
What?

- An instrument for experimental CS research
- A completely configurable “Internet emulator” in a room
  - At its core, it’s bare hardware, with...
  - ... complete remote access and control
- But, also simple to use
  - Lots of fast tools for common case
    - Automatic topology, node and link configuration
    - Automatic traffic generation
  - Universally available
    - Universities, research labs, companies
  - Zero-penalty for remote research
computational grids & emulab

- Study computational grid applications, middleware:
  - over an Isolated facility ➫ Risk-Free and Artifact-Free
  - on a simple LAN cluster to complex WAN configuration complete with cross-traffic
  - in batch experiment mode which makes it even more suitable for such use
  - for effects on changing network topology and characteristics
  - e.g.: parallel model checking in the formal verification group of U of Utah (refer overview text)
Key Design Aspects

- Allow experimenter complete control
  - Configurable link bandwidth, latency, and loss rates, via transparently interposed “traffic shaping” nodes that provide WAN emulation
- ... but provide fast tools for common cases
  - OS’s, state mgmt tools, IP, batch, ...
  - Disk loading - 6GB disk image FreeBSD+Linux
    - Unicast tool: 88 seconds to load
    - Multicast tool: 40 nodes simultaneously in < 5 minutes
- Virtualization
  - of all experimenter-visible resources
  - node names, network interface names, network addr
  - Allows swapin/swapout, easily scriptable

Key Design Aspects (cont’d)

- Flexible, extensible, powerful allocation algorithm
  - Matches desired “virtual” topology to currently available physical resources
- Persistent state maintenance:
  - none on nodes, all in database
  - work from known state at boot time
- Familiar, powerful, extensible configuration language: ns
- Separate, isolated control network
Some Issues and Challenges

- Network management of unknown and untrusted entities
- Security (users get root on nodes!)
- Scheduling of experiments
- Calibration, validation, and scaling
- Artifact detection and control
- NP-hard virtual --> physical mapping problem
- Providing a reasonable user interface
- ....
Automatic mapping of desired topologies and characteristics to physical resources

- NP-hard problem: graph to graph mapping
- Algorithm goals:
  - Minimize likelihood of experimental artifacts (bottlenecks)
  - "Optimal" packing of many simultaneous experiments
  - Extensible for heterogeneous hardware, software, new features
- Randomized heuristic algorithm: simulated annealing
- Typically completes in < 1 second

Ongoing and Future Work

- Wireless nodes, mobile nodes
- IXP1200 nodes, tools, code fragments
  - Routers, high-capacity shapers
- Simulation/emulation transparency
- Event system
- Scheduling system
- Federated Emulab (analogous to Computational Grids)
- Data capture, logging, and visualization tools
- Microsoft OSs, high speed links, more nodes!
### Virtual Topology

![Virtual Topology Diagram]

### Constraints on Mapping

- Must map each virtual node and link to available physical resources
- Multiple switches usually have limited interconnect bandwidth
- In our example, there are four switches, each with 400 Mbps interconnect
- More than 4 links mapped onto a given interconnect would produce an artifact because of the bottleneck
- In solution, node color indicates its switch
Mapping into Physical Topology

Complementary to Other Experimental Environments

- **Simulation**
  - Fast prototyping, easy to use, but less realistic
- **Small static testbeds**
  - Real hardware and software, but hard to configure and maintain, and lacks scale
- **Live networks**
  - Realistic, but hard to control, measure, or reproduce results

Emulab complements and also helps validate these environments