# *Tutorial Description for SIGCOMM 2002*

# How to Use the "Emulab" Public Network Testbeds

Jay Lepreau Mac Newbold Rob Ricci Chris Alfeld

University of Utah

May 9, 2002

## 1 Overview

This tutorial will focus on teaching attendees how to effectively use a new, public experimental resource for research, education, or development in networking and distributed systems—a resource that is being cloned by others. Emulab (`http://www.emulab.net`) is a time and space-shared reconfigurable "network emulator," comprised of hundreds of PCs, miles of cable, and lots of code. Using a Web interface and a superset of the popular *ns* networking simulation language or a Java GUI, it is simple for remote users to configure, in minutes, any number of these machines, ethernet cables, traffic generators, and their choice of operating systems and disk contents, into a network of arbitrary topology and characteristics, running arbitrary programs. Live demos and—in the full day version—hands-on lab sessions will be fully integrated into the tutorial.

*In new work, Emulab also incorporates transparent ("mix and match") support for physically distributed nodes (starting with MIT's "RON"testbed) and purely simulated nodes.*

Emulab software provides quick and convenient access to large-scale, dynamically reconfigurable testbeds, that serve as platforms for experimentation, testing, validation, prototyping, evaluation, and instruction. Emulab provides zero-penalty for remote use, complete separation between projects, allows full "root" access, allows anyone to run their own custom software, including operating systems, and allows them to observe, instrument, and measure nearly anything that occurs in their experiment. An integrated event system allows both scripted and interactive control of all aspects of an experiment. With trivial changes to their *ns* scripts, users can shift the mapping of many experiments between pure simulation, pure emulation, or a mixture of both.

Over 40 projects and 25 institutions have so far used Utah's Emulab. They include many distinguished external researchers as well as two classes at Kentucky and MIT, in basic and advanced networking. The open-source software underlying Emulab is portable; several other Emulabs are already planned or under construction. The University of Kentucky has already racked their hardware, and additional Emulabs may be running by SIGCOMM in August.

This tutorial will also outline the minimum and optimal resources needed to build your own Emulab, what's needed to convert your generic cluster into an Emulab site, and plans to federate the Emulab sites into a vast, loosely-coupled distributed testbed. We will also outline other plans, including extensions to wireless and mobile nodes.

## 2 Tutorial Organization

### 2.1 Introduction and Motivation

We first outline the current methods for experimentation and validation in networking research, which include simulation, testing on small-scale static testbeds, on special overlay testbeds like RON, and on live networks.

The strengths and weaknesses of each will be briefly discussed.

We then describe, at a high level, the Emulab facility that we have developed over more than two years, which provides a new, complementary approach that provides both complete control and substantial realism. We will outline its key design principles and high-level features.

## 2.2 How to Use It

This is the core of the tutorial, getting the most time. We will start from the premise "So you need a testbed for your research...," and will progress through the process of designing, configuring, and using a simple network on Emulab—all integrated with a live demonstration. This initial example will cover: i) What kinds of experiments are well-suited to Emulab, and what kinds are not. ii) How to use the `emulab.net` Web site to get an Emulab account and to start an experiment. iii) How to formulate the target network description with an *ns*-compatible script, or alternatively, a Java GUI. iv) How to use the configured network. v) How to end the experiment when your work is complete, or "swap" it when you reach a hiatus.

Following this walk-through of the process with a small example, we will discuss how to extend and expand a small network to large-scale or more complex networks and Emulab services. This portion comprises the largest portion of the tutorial. The topics that will be covered (both in this session and, in the full-day version of the tutorial, the hands-on lab session described in Section 2.10) are among those listed in the Appendix.

## 2.3 How Has Emulab Been Used?

This section will be a series of very short case studies of our users' diverse research, focusing on Emulab's role and what it brought to the table—as well as where it worked less well.

## 2.4 Emulab in Education

Emulab has some features especially targeted to educational use, such as an authorization and operational hierarchy that models the prof/TA/student relationship. It has been used in a major way for two networking courses, an advanced course at MIT (Hari Balakrishnan prof, Dave Andersen TA) and a lower-level course at Kentucky (Jim Griffioen). Their experiences and lessons learned will be summarized, either by the Utah instructors or by a guest appearance by Andersen and/or Griffioen.

## 2.5 Under the Hood

A brief overview of Emulab's internal workings, focusing on the more interesting or challenging aspects. Open issues, both research and otherwise, will be mentioned.

## 2.6 Building Your Own

We will outline the issues in building your own Emulab, either low- or high-end, using our software. We'll summarize our experience with the operational load, and the pros and cons of joining the distributed set of federated Emulab sites.

## 2.7 What Might *You* do with Emulab?

This will be an audience-driven conversation of how Emulab could help them simplify, enhance, or expand their work. It will presumably focus on the research, instruction, or facilities in which the audience members are involved or considering. This could also include a discussion of new features the audience would like to see in their current tools or in Emulab.

## 2.8 Ongoing and Future Plans

Those items marked with a * may already be working, somewhat, by August—if so, we will give them time in the "how to use it" sections.

* Federated emulabs: motivation, design, plans, status. Collaboration/merging with the RON overlay network and other distributed testbeds.

- NIMI-like extensible JVM environment on distributed 1-2 node "emulabs" to provide a distributed measurement infrastructure.

* Wireless support (dense mesh of fixed nodes in a building/campus).

- Mobile support (real RF device carried around by passive couriers who travel predictably in time and space: 1) students attending classes, and 2) city buses.

* Checkpointing/swapping, rollback, "single stepping" of a closed distributed system.

* Further simulation/emulation transparency.

* Multiplexing multiple virtual nodes on single physical nodes.

* Support for Windows.

- ...

### 2.9 Conclusion

Lessons learned, including the contrast between our expectations of emulab's value and our discoveries.

### 2.10 Lab Session

In the full-day version of this tutorial, the extra time—in the afternoon—will primarily be allocated to a hands-on "lab session." Attendees will pair up with each other at laptops. We will lead the attendees through some carefully canned projects; they will go through the process of specifying their systems, using the networks they created, and gathering results. A variety of projects will teach the attendees about different aspects of emulab and its features.

Pairs of people seem to attain the right balance in all respects, including interaction with each other and pressure on Emulab and laptop/terminal resources. We can adapt the "people cluster" size as needed.

If time and interest permit, we will undertake one or more attendee-provided projects. To get them thinking, we will have mailed the registrants beforehand.

## 3 Intended Audience

Researchers, teachers, and developers in networking, distributed systems, or simulation: those who are interested in new methods of experimentation, testing, validation, evaluation, or instruction. This would include faculty, students, research staff, and networking professionals from industrial settings—emulab is highly relevant to testing and performance evaluation of any system that involves a network.

Minimum background knowledge: Basic understanding of core networking principles, which everyone at the conference will easily meet. Handy to have, but not required: some familiarity with the *ns* simulator; a "feel" for configuring or programming real networks (as opposed to strictly analytic or textbook understanding of networks); some knowledge of standard network tools like `tcpdump`.

## 4 Half-day and Full-day Offerings

The half-day tutorial, and morning of the full-day version, will contain the main lecture components covering many of the topics outlined above, interspersed with live demonstration of Emulab's capabilities and uses. The afternoon of the full-day version will focus on the hands-on lab exercises, but will also contain additional lecture material giving more detail on the issues mentioned above: "build your own," future plans, etc.

Attendee laptops require only a Java-capable Web browser and an ssh client.

## 5 Detailed List of Topics

The Emulab features and topics covered in the “How to Use It” and “Lab Session” will be drawn from the following list.

- Using the Web interface
- Overview/review of standard *ns* features
- Specifying a topology and configuration using *ns*
- Emulab extensions to *ns*, including:
  - Customizing node disk contents
  - Choosing your OS
  - Choosing your hardware
  - Virtual node and link types; equivalence classes
  - IP addrs
  - Delayed LANs
  - Giving a startup command
- Automatic traffic generation
  - Standard *ns* interface
  - Choosing different implementations
- Specifying a topology and config using the ’netbuilder’ Java applet GUI
- Naming, both virtual and physical
- Overview of the resource allocation report
- Barrier synchronization at startup
- Using Emulab’s *ns*-compatible event system
- Dynamic events
- Batch experiments

- Using physically distributed nodes
- Using purely simulated nodes

- Structure of users’ filesystem layout on the server; protection
- Access to nodes’ console logs and console input
- Rebooting and power control
- Recovering from a scrogged disk
- Routing control
- Generating custom disk images, including kernels
- Using your own non-standard (non-Unix/Linux/Windows) OS
- Experiment swapping
- Security
- Hints on optimizing performance and use of storage
- Standard *ns*-compatible tools: topology generators, visualizers, . . . .
- Hidden directories if things go wrong.

## 6 Instructor Biographies

Unless otherwise noted, all speakers are with the University of Utah, School of Computing. If the tutorial is full-day, most of the instructors will wander the room during the lab exercises, lending individual help.

**Jay Lepreau:** overall coordinator of the tutorial, and speaker for the more general portions. He is a faculty member in the University of Utah’s School of Computing, where he heads the Flux Research Group. He has interests in many areas of software systems, most of them originating in operating systems issues, although many go far afield. Those disparate areas include information and resource security, networking, programming and non-traditional languages, compilers, component-based systems, and even a pinch of software engineering and formal methods. In 1994 he founded the highly successful and prestigious OSDI conference series, one of the two premier OS conferences. He conceived and leads the emulab project.

**Mac Newbold:** Graduate student, doing MS in C.S. He has been working on Emulab for two years; while an undergrad he designed the initial versions of many of its core systems. For three years he has helped teach the “High School Computing Institute:” a summer program for talented high school students, run by the School of Computing, containing a large lab component.

**Rob Ricci** Research Associate. As an undergrad, he prototyped a novel use of active networking ideas, “agile protocols,” for censor-resistant overlay networks, and published it at HotOS’01. Rob has been working on the Emulab software and hardware base for over a year, and knows its internals well. Star of a local TV feature on steganography: gives great interview!

**Chris Alfeld** Ph.D. student in Math at the University of Wisconsin-Madison. As one of the original Emulab developers, he designed much of the *ns*-compatible front end software as well as a core algorithm that performs the NP-hard mapping of the virtual network onto the physical network, and remains involved as needed. For four years he has helped teach the “High School Computing Institute:” a month-long summer program for talented high school students, run by the School of Computing. Chris is anxious to participate if we need him.

**Technical Backup: Leigh Stoller, Mike Hibler:** These two highly skilled and professional research staff will be “on alert” back at Utah, to ensure that should any problems arise in emulab during live demos or lab exercises, they can immediately be fixed. Together they have over 20 years experience working with Lepreau’s group on various projects, from virtual memory to component-based kernels to compilers to simulators. Leigh is the lead developer of much of the Emulab software suite; Mike is knowledgeable on most of it.