An Application-Centric Approach to Emulating Internet Paths

Jonathon Duerig Robert Ricci Junxing Zhang Sneha Kasera Jay Lepreau
University of Utah www.flux.utah.edu

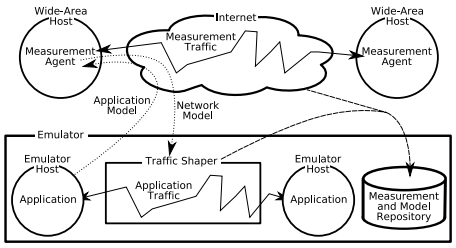

Figure 1: A diagram of the application control flow.

Emulation testbeds are valuable tools for testing and evaluating research prototypes of networked systems. They give users great control over the host and network environments and offer easy reproducibility. But emulation testbeds have a shortcoming: their network conditions are artificial and thus do not exhibit some features of real networks. Users have turned to overlay testbeds for real network conditions, but these systems have their own drawbacks. First, due to the large degree of sharing, host resources such as CPU, memory, and I/O bandwidth are frequently inadequate and can cause misleading results. Second, while it may be possible to isolate an experiment from other users of the testbed, it is impossible (by design) to isolate it from the Internet's varying conditions, making it tedious to gather statistically significant results. Third, in today's overlay testbeds users cannot perform many privileged operations, including choosing the OS or modifying the kernel.

Our solution is to use network conditions measured using a wide-area testbed, in this case, PlanetLab, to control an emulator, Emulab. There are several possible approaches. One is to measure the paths between all pairs of PlanetLab nodes frequently, and build a model from which Emulab would emulate those paths. This approach entails many of the unsolved challenges of realistically modeling the Internet.

An alternative centers on defining an application model for emulation. We pair each node in the emulated network with a peer in the Internet (Figure 1). The real application runs on the emulation testbed. We use the application's traffic to generate a straightforward traffic model that is sent to the peer node in the wide-area testbed. "Stubs" on the peers in the Internet then generate traffic based on these parameters. We analyze the conditions seen by the traffic generated on the wide-area network, and derive a model which is sent back to the emulated network to shape the application's traffic.

The application modeling approach has several important advantages. The traffic is shaped based on the network conditions perceived by the traffic itself, removing artifacts introduced by special measurement traffic. Another key benefit is that the amount of traffic used by the measurement infrastructure is bounded by the amount of traffic actually sent by the application, rather than the number of paths (which would be $O(n^2)$ in the number of nodes).

Students: Duerig is undergrad, Ricci MS, Zhang PhD. Corresponding author: duerig@cs.utah.edu.

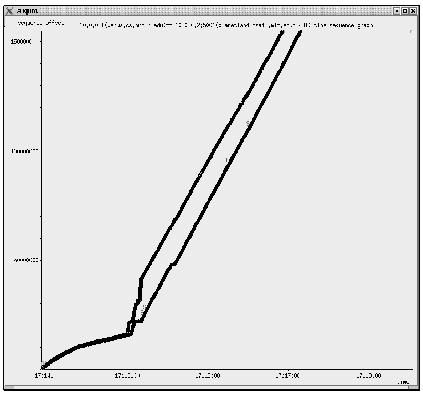

Figure 2: Sequence number (y-axis) vs. time (x-axis) of an Internet connection and its emulated counterpart.

Application modeling does have some limitations. Because the measurement traffic is based on the application traffic, there is no measurement data available for paths that are yet to be used. We address this issue by taking periodic background measurements using tools that consume less network bandwidth, but are therefore less accurate. So, while we have reasonable initial conditions with which to bootstrap the emulator, some artifacts will remain. Another issue is that because the feedback loop is over a wide-area Internet connection, the latency of the loop may be high. This would cause measurement artifacts in applications that are reactive to Internet conditions on a small time scale.

Figure 2 shows the offset in TCP sequence numbers over time. The upper line is from the real application (on the emulated network). The lower line is from the measurement agent (on the real network), measuring traffic between PlanetLab nodes at Brown University and MIT. The slopes of the lines correspond to throughput. During periods that the throughputs differ, the two lines diverge. There is inevitably some gap due to control latency between the emulated and wide-area nodes. Whenever a change in bandwidth or latency occurs, there is a delay before that change is reflected in the traffic shaper. A measure of the accuracy of our emulation is how much the lines diverge for a given set of bandwidth changes. Figure 2 shows that when available bandwidth is relatively stable, the bandwidth experienced by the application quickly converges to the correct value. We are currently exploring rate-based mechanisms to better track drastic changes in available bandwidth.