Experience with an Evolving Overlay Network Testbed

David G. Andersen, Hari Balakrishnan, M. Frans Kaashoek, Robert Morris
MIT Laboratory for Computer Science
{dga,hari,kaashoek,rtm}@lcs.mit.edu
http://nms.lcs.mit.edu/ron/

ABSTRACT

The MIT RON testbed consists of 36 Internet-connected nodes at 31 different sites. It has been in operation for two years. This paper presents an overview of the testbed, summarizes some of the research for which it has proved useful, and presents the lessons we learned during its development. The testbed has been useful both for our own research and for that of external researchers because of its heterogeneous, diverse network connections; its homogenous hardware and software platform; its incremental, bottom-up development; and its easy configuration and management tools.

1. INTRODUCTION

In the fall of 1999, we embarked on a project to improve the resilience of distributed Internet services to faults and problems in the underlying IP routing substrate. Our approach was based on the idea of a *Resilient Overlay Network (RON)* [3, 1]. A group of collaborating nodes interested in communicating with each other would form a RON, routing packets via one another in an application-layer overlay network if the direct Internet path between two nodes experiences an outage, severe congestion, or poor performance for other reasons. Nodes in a RON would use a combination of active probing and passive observations of on-going traffic to obtain information about alternate paths in the overlay and disseminate this path information using a link-state routing protocol. Our second idea was to integrate overlay routing more tightly with the distributed application, to give applications better choice in path selection.

The attractiveness of the RON approach is its conceptual simplicity. We had numerous hypotheses about its real-world behavior: for instance, that (1) despite the failure of the Internet path between nodes A and B, RON could find a set of intermediate nodes in the overlay that connected them; (2) it was mostly sufficient to consider exactly one intermediate node C for such a path; (3) outages could be detected and bypassed in a timely manner; and, (4) in practice, the latency-, loss-, and throughput-optimized paths between two nodes in the overlay were not always the same.

This research was sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Space and Naval Warfare Systems Center, San Diego, under contract N66001-00-1-8933.

To test these hypotheses, it became imperative that we deploy an overlay network for our experiments. We have since been deploying and evolving the *RON testbed* to answer questions like those posed by our initial hypotheses, and many others that have arisen in the course of research conducted by us and others on the testbed. The RON testbed now consists of 36 operational machines at 31 different sites in 8 countries, with nodes obtaining Internet service from at least 12 different providers (ISPs) (see Table 1). It has been our experience that at any given time about 90% of these machines are alive and well.

In addition to confirming our hypothesis that an overlay can yield dramatic improvements in communication availability and often leads to better performance, the RON testbed was unexpectedly useful to several other researchers over the past two years. Numerous projects have benefited from the testbed, including overlay routing, distributed hash tables (DHTs) and DHT-based peer-to-peer file systems, network analysis experiments, and experiments to correlate end-to-end path failures with BGP routing messages.

This paper gives an overview of the RON testbed, summarizes the diverse set of research projects being conducted on it, and describes its key positive features and problematic limitations. Based on our experience and discussions with other testbed users, we have concluded that the most important strength of the testbed is in its *heterogeneous network connectivity*. This was an important factor in our choice of sites for the original RON research, since a group of nodes hosted at university sites all on Internet2 lacks the performance diversity of the commercial Internet. We have strived to deploy nodes at companies, homes, ISP co-location facilities, and international sites, and have been moderately successful on these fronts. Currently, only 6 of the 31 sites are at US universities.

The second reason for the success of the testbed in facilitating interesting research is the *homogeneity of the hardware and software platforms*. All our sites run identical versions of FreeBSD; this homogeneity reduces the hassles of deploying software on the testbed.

Because our original motivation was RON research and not testbed development, the RON testbed grew incrementally in *bottom-up fashion*. This turned out to be a strong advantage, because it allowed us to debug deployment problems more easily than if had been required to deal with a large number of problematic machines all at once.

Finally, a major operational win in the testbed arises from the *ease of testbed configuration and management*. We learned the importance of this from painful experience. Through collaboration with the Netbed researchers, we now have a set of simple tools that make it possible for a single person to manage the entire testbed: it is currently managed by one graduate student as a "side" activity to his actual PhD research.

2. THE RON TESTBED

Over the years, many researchers have deployed and used testbeds for network research. These testbeds have ranged from complete parallel networks like the Internet2 [11], CAIRN [5], and DARTNet [26] to software-only distributions like NIMI [19] that run on shared machines.

Separate network testbeds are useful for low-level network protocol research because they provide an environment in which everything above the hardware is mutable. This flexibility, however, comes at significant cost. Dedicated network connections are much more expensive than commodity Internet connections, and maintaining these networks requires a large investment of time and expertise.

At the other end of the spectrum, projects such as NIMI [19] and the ABone [2] used no dedicated hardware, relying on shared UNIX accounts donated by host sites. The cost of the monetary savings is that these projects run in a heterogenous environment, and must strictly isolate hosted experiments from the host machine. The heterogeneity creates a larger burden on software developers, and the stringent security concerns restrict the types of experiments that these testbeds can host.

The RON testbed takes a middle approach to testbed development, running on relatively homogenous, centrally managed hardware that resides in a heterogenous, donated Internet environment. This approach has predictable (and relatively low) up-front costs of about US $2,000 per node, and low on-going costs. It is much easier to develop software in this homogenous environment, and having the testbed run on machines we manage makes us less subject to the whim of the old, out-of-date, and ill-managed machines that people are likely to donate to a testbed.

2.1 Hardware and Software

It became apparently early in the deployment of the testbed that hardware and platform homogeneity, to the extent it could be achieved, would enhance our ability to maintain the system. Testbed node hosts may choose between 1U rack-mount machines and mini-tower boxes. The rack-mounted units are vastly more popular, accounting for all but four of the deployed nodes. Because we have deployed the machines over three years, the processor, disk, and memory capacity has increased over time. The type of hardware used, however, has been identical (Pentium III and 4 chips, the same series of Seagate SCSI drives, and Intel network interfaces), so the driver configuration on the machines has remained the same.

About half of the testbed nodes are equipped with CDMA (cellular) time receivers. These timers[1] provide approximately $10\mu s$ time resolution, similar to GPS, but can operate inside most machine rooms without an external antenna. They provide time synchronization and facilitate one-way path delay estimates.

The machines run a recent version of FreeBSD, which we upgrade about every six months. The testbed interface is a non-root account on all of these UNIX-like machines. Select trusted users have root access on some nodes, and we install setuid binaries for users when necessary.

Standard accounts on the testbed are not virtualized—users see the base OS and the real network. Users are granted access to all nodes in the testbed, and must manually coordinate experiments and cooperatively share resources with each other. To date, the only resource abuses we've encountered were accidental and short-lived, but this is mostly a result of having a small and well-behaved user community. Low-overhead experiments execute concurrently and continuously, but we ask users with higher bandwidth requirements to check in with us before running large experiments, providing some human-mediated resource allocation.

Software homogeneity precludes some kinds of research (*e.g.*, experimenting with different TCP implementations, etc.), but has significant benefits. We note that Paxson moved his early measurement infrastructure from a multi-platform effort [18] to FreeBSD and NetBSD only, noting that he and his colleagues experienced problems with site configuration differences even with a small number of distinct platforms [19]. In our testbed, two sites donated machines with an OS already installed, and despite upgrading the machines to our standard software, we have encountered problems from root partition sizes and old configuration files. With our adoption of the Netbed boot CD (Section 4.5), we can address these problems more aggressively.

2.2 Deployment

We deployed the first testbed nodes in February, 2001, and continue to ship about one new node per month. We currently have 36 active nodes at 31 sites[2], listed in Table 1. These nodes reside in a variety of locations, from private residences with DSL connections, to large, well-connected companies and schools. Table 2 shows the breakdown of sites by the type of institution hosting the node, and Table 3 shows the types of network connections.

Six testbed sites provide the RON nodes full BGP feeds from their border routers. An additional site provides customer routes, and the four Global Crossing sites are bringing feeds online soon.

2.3 Management

[1]The Praecis Ct from EndRun Technologies.

[2]PSG and sg have firewall issues that make them only partly available to researchers.

Name	Location	Upstreams
Aros	Salt Lake City, UT	UUNET, ELI
AT&T	Florham Park, NJ	AT&T
CA-DSL	Foster City, CA	Pacific Bell
CCI	Salt Lake City, UT	ViaWest/L3
* CMU	Pittsburgh, PA	Many via PSC
Coloco	Laurel, MD	Many
* Cornell	Ithaca, NY	Applied Theory
Cybermesa	Santa Fe, NM	Espire
Gblx-ams	Amsterdam, NL	Global Crossing
Gblx-chi	Chicago, IL	Global Crossing
Gblx-jfk	New York, NY	Global Crossing
Gblx-lon	London, UK	Global Crossing
Greece	Athens, Greece	GEANT
Intel	Palo Alto, CA	AT&T+?
Korea	KAIST, Korea	
Lulea	Lulea, Sweden	
MA-Cable	Cambridge, MA	AT&T
Mazu	Boston, MA	Genuity
* **MIT**	Cambridge, MA	Genuity
NC-Cable	Durham, NC	
Nortel	Toronto, Canada	AT&T Canada
* NYU	New York, NY	Applied Theory
PDI	Palo Alto, CA	Qwest
PSG	Bainbridge Island, WA	Genuity, Verio
PWH	Sunnyvale, CA	Many
Sightpath	Near Boston, MA	UUNET+
sg	Singapore	
* UCSD	San Diego, CA	Many
* Utah	Salt Lake City, UT	Qwest
Vineyard	Cambridge, MA	Savvis, Qwest
VU-NL	Amsterdam, Netherlands	

Table 1: Testbed sites. Asterisks indicate U.S. universities on the Internet2 backbone. Bold indicates a BGP feed. If known, we list the site's upstream ISPs.

Site type	Number
US Private Residence	3
US Company	13
US University	6
International Company	3
International School	5

Table 2: RON Testbed site breakdown.

Connection type	Number
DSL	1
Cable Modem	2
T1	2
$\leq$ T3	3
100Mbps *	8
OC3+	6
International	8

Table 3: Connectivity breakdown. Some sites listed as 100Mbps may have T3s or fractional T3s. All International links are high speed to a local backbone, but connection speeds to the rest of the world vary.

The nodes are managed through the Netbed [27] testbed. Users apply for accounts at Netbed, and request access to the MIT RON testbed nodes. If granted, the users get accounts on all of the testbed nodes. Software updates and node installation are handled through a Netbed-based software update and CD-ROM based boot system.

Access to the testbed is granted only to "mostly trusted" researchers. The testbed is a research tool only; we have not invested time in sophisticated isolation techniques for untrusted users. We aim to keep the population of testbed users large enough to ensure that it's useful, but small enough that a well-defined Acceptable Use Policy (AUP) and social pressure can ensure good behavior.

2.4 Data Collection

As a basic service on the RON nodes, we collect periodic latency and loss samples, topological information using `traceroute`, and, at some sites, BGP routing data. Unlike the RON system itself, this general measurement service on the testbed does not yet use passive data collection or permit users to insert their own measurements.

Roughly once per second (the exact period is randomized), every node sends a small UDP packet to another randomly selected node, which echos the packet back. These samples provide latency and loss information. If our probing daemon observes multiple losses to a particular host, it initiates a traceroute to record the path along which the failure occurred. Once per day, we traceroute from every node to every other node to provide a topology snapshot. We have established a BGP peering session with six of our hosts border routers, and we record and archive all BGP routing updates received at the hosts for later analysis. This data is freely available upon request.[3]

3. PROJECTS USING THE TESTBED

3.1 Overlay Networks and DHTs

We initially deployed our testbed to determine how effectively overlay networks can route around network problems [3]. We followed this research with a study examining the effectiveness of packet duplication to avoid losses [4].

Researchers from both DePaul and HP used the testbed to collect relative one-way delay measurements of packets streamed at various constant bit rates. The HP researchers examined delay/loss correlation of two parallel flows from diversely located sources to examine the possible benefits of path diversity for video streaming systems.

Ziskind and Krishnamurthy used the testbed to perform experiments with overlay multicast, and Freedman used the testbed to measure the latency to hundreds of Gnutella hosts from a variety of locations within the Internet [10].

Dabek et al. used the RON testbed to evaluate the Chord distributed hash table (DHT) in real-world conditions [22] and to develop a novel congestion control protocol for multiple-sender file transfers. They then used RON nodes as clients

[3]Our BGP website, `http://bgp.lcs.mit.edu/` provides an interface to obtain snapshots of the data.

and servers to validated the CFS wide-area cooperative filesystem [7]. The Ivy project ran storage servers on RON nodes [17]. Using a real Internet testbed pointed out the impact of slow hosts on the filesystem, and the need for new congestion control mechanisms.

Researchers from Rice have used the testbed to conduct similar follow-up measurements of their own DHT, Pastry [21].

As the RON research showed, Internet failures are often quite complex; the Internet certainly does not provide fail-stop semantics. These complex failures played a part in the testing of Chord, and also in Keidar et al.'s development of group membership protocols. The latter used the RON testbed extensively to validate and develop their wide-area membership protocols. In a similar vein, Jacobsen et al. used stored latency and loss traces from the testbed to evaluate their group membership protocol [12].

3.2 Network Analysis

Katabi and Blake used packet dumps of TCP transfers between testbed nodes to evaluate passive methods for inferring shared congestion[14]. The experiments benefited from the ability to take raw packet traces and from the diversity of network connections in the testbed. In similar link analysis work, both Strauss [23] and Dovrolis et al. [13] evaluated methods for bandwidth estimation using TCP transfers and packet-pair-like techniques. Tang and Crovella [24] used datasets collected from the RON testbed as part of their evaluation of a new Internet coordinate scheme.

3.3 BGP Routing

Our ongoing work on the RON project combines end-to-end probing data from the machines with the BGP feeds we established at 6 of our host sites to explore the interaction between end-to-end reachability and BGP routing [9].

Mao et al. are using BGP feeds from the testbed to observe and analyze BGP beacons, unused prefixes that have well-defined schedules for announcement and withdrawal [15]. BGP data from the RON testbed provides topologically and geographically distributed observation points for these beacons, allowing researchers to monitor the propagation of BGP updates.

3.4 Feedback from Users

Diverse network connectivity. By a wide margin, the most popular feature of our testbed was the ability to run real code on a real network with a diverse set of network paths and properties. Despite this heterogeneity, one researcher commented that the overall loss rate between nodes still seemed low. While the overall loss rate appears in 2003 to be as low as 0.42% [4], we still observe numerous failures between the RON measurement hosts. Figure 1 shows the number of failures with which each host was involved between February and October, 2002. Two researchers indicated that the testbed was still too small to perform some studies they wanted to do.

Time synchronization. Two researchers used the time synchronization to obtain one-way delays. We had hoped this facility would be more useful and more popular, but discovered that it still requires considerable care to obtain valid one-way delay measurements. One unexpected benefit of having stratum-1 time servers was that it provided an incentive for sites to host us—5 of our sites responded to an offer for a managed stratum-1 time source.

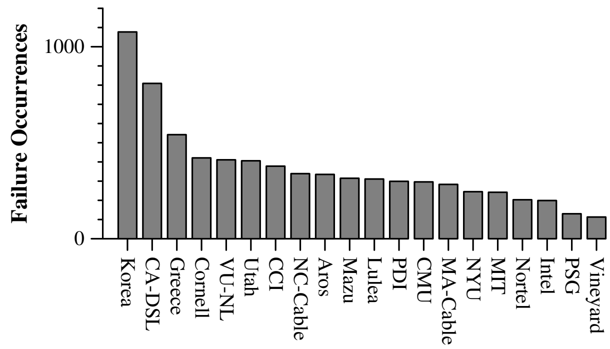

Figure 1: Failure distribution by measurement host from 02/2002 - 10/2002. The rightmost four hosts were online for less time than the others.

Root and other features. Users who had root access to the nodes commented that it was a great feature, reducing the amount of work they had to do to get their code to work. Several users made extensive use of the SFS [16] remote filesystem, finding it convenient for distributing software to the nodes.

Node problems. One user experienced persistent authentication problems during the transition to Netbed account management. We believe this was mostly due to our failure to clearly document the login procedure, a problem we observed with other users, though none commented.

Three users noted problems with hosts that were firewalled and behaved in strange ways, and two users experienced recurring problems on a non-standard node. One user noted problems with DNS and `hostname` values not matching up.

Information dissemination. Most of our users liked the status pages that are available, but at some point, nearly all requested further information about the geographic location and network connectivity of the site hosts. It seems clear that we should enhance our existing user interfaces.

4. LESSONS LEARNED

4.1 Security

Our testbed has had one major internal security incident, and a handful of external incidents. One testbed node that was running on a donated, shared machine was compromised by a host OS exploit. We had corrected this exploit in the pool of MIT-managed machines, but missed it in the external node. This incident increased our determination to keep the testbed homogenous, and we subsequently took over management of this machine.

Our external security problems were all unauthorized probes from testbed machines. Several researchers attempted to "map the Internet" with far-flung pings and traceroutes. While such research is legitimate, it generates complaints from firewalls and intrusion detection systems [6]. The targets of such probing complain to our host sites, not to us, giving us no chance to intervene. We therefore clarified our AUP to prohibit general network probing. We instead invite researchers to use a machine at MIT, `mit-network-monitor.lcs.mit.edu`, that we set up for wide-area probing.

Testbed traffic caused false alerts on three host sites' Intrusion Detection Systems. In each case, we found that the sites' rules were erroneous and submitted corrections to the maintainers of the Snort IDS system, but identifying the errors was extremely time consuming. We recommend that hosts place RON testbed machines outside of any firewalls and intrusion detection systems, though that is not always an option.

4.2 Resource Control

We have encountered eight resource-related incidents, which highlight the importance of having at least a small degree of resource control. Four incidents required machine reboots and two generated complaints of excessive bandwidth consumption.

In two cases, accidental file descriptor leaks in users' programs rendered the machines useless. Existing processes could run, but we could not login to the machines to kill the offending processes. We partially solved this problem by contributing a patch to FreeBSD to reserve some file descriptors for the root user; we are still searching for a better solution.[4]

One memory leak was sufficiently severe to require a machine reboot. This was a simple and unfortunate oversight on our part: We left the per-process memory limits too high. Finally, one user's program spin-waited while running at maximally elevated priority. Unlike the accidental problems, we addressed this one only through policy, requiring that our users not spin wait for more than 2 milliseconds, and requiring that their programs not run at elevated priority without prior permission.

We had two bandwidth consumption problems on each of the cable modem sites. One incident was accidental—probes left running when they should have been killed. The remainder were legitimate TCP transfers that used so much bandwidth that the site hosts complained about their response time. At present, we have only solved bandwidth consumption problems with policy. We believe the right solution is one being investigated by the Planetlab team: providing the site hosts with a "maximum bandwidth" control for the node. Our experiments show that this is easily implemented with Dummynet [8], but we have shunned bandwidth caps thus far to avoid possible interference with fine-grained measurements.

Many of the solutions we rely upon would be insufficient to deal with malicious, or simply greedy, users. Because the testbed is only available to a (relatively) small and trusted group of users, social pressure is more effective at preventing resource contention than it would be in a more public testbed.

4.3 Reliability

The two major reliability problems we've encountered so far have been hardware misconfigurations (our errors) and dependencies on external services. Perhaps surprisingly, we have only had two failures due to software upgrades, and both of these were on machines that were donated with an existing operating system. We have had no hardware failures yet.

To understand actual machine shutdowns that required contacting site hosts, we tracked six months of machine failures from May, 2002 until December, 2002. During this time, we had 18 failures of various durations. In five cases, the nodes failed to reboot after a power failure due to an incorrectly configured BIOS. Three failures arose from host site IP address changes. Two failures came from software configuration errors when we upgraded machines. These configuration errors occurred on our Korea and Lulea nodes, where we were given root access to an existing machine, and we failed to update an out-of-date SSH configuration file.

4.3.1 DNS Configuration

At one point, external DNS dependencies invalidated three months of measurements by causing false probe failures. This seems analogous to Paxson's problems with centralized probe triggering [18]. We now run a local caching-only nameserver on all RON nodes to reduce these effects, and also re-architected our probe system to eliminate DNS dependencies.

Because the BIND nameserver has historically been a common source of security holes, we configured it to listen only to the loopback interface. This reduces the security risk, and permits researchers to bind the external nameserver port for research projects.

4.3.2 NTP Configuration

Only half of the testbed nodes are equipped with hardware time receivers, and even those must be properly configured with backup time servers in case the CDMA receiver fails. To achieve the best time synchronization possible without hardware, each node is configured with

1. CDMA hardware, if attached
2. The MIT and Utah RON nodes
3. 2 nearby RON nodes
4. 1+ nearby external stratum-1 time servers

The default of using an east coast and (roughly) west coast stratum-1 timeserver ensures that US nodes have a timeserver within 50ms without extra configuration. We then manually configure nodes to peer with two other RON nodes and attempt to locate one nearby stratum-1 timeserver. The mean delay to the server for US nodes without a receiver is 22ms.

Manually configuring a known good nameserver provided resilience in the case of the two hosts that filter NTP packets. Nodes NTP configurations are fetched from the Netbed central database.

4.4 Account Management

One account. The initial nodes had a single, shared account to which we would add the SSH public keys of valid users. Redistributing a new `authorized_keys` file was much easier than dynamically creating accounts on many machines, and

[4] The Linux kernel provides the same reservation solution.

the mechanism sufficed while the testbed was only used by about 5 trusted researchers. We avoided most common shared login systems such as NIS and RADIUS because they introduced a dependency on the network, something obviously unsuited to a testbed initially intended to measure network failures.

Netbed-based account creation. Once we opened the testbed to select external researchers, the Netbed developers helped us transition to their account management software. Each machine periodically checks in with Netbed and checks the list of users it should support. These users are then added to (or removed from) the local password file, so that the only effect of network failures is to postpone account creation or deletion. This shift was the single most time-saving change we made to our initial system.

4.5 Maintenance and Deployment

Initial attempt. Our first nodes were deployed with FreeBSD installed on the single hard drive, with no CD-ROM drive. We installed the initial system image by using `dd` to copy the hard disk image, and then rebooted and performed per-machine configuration. This process was moderately time-intensive, taking about 30 minutes of active user time to completely install a node.

We performed system upgrades manually from the source tree using `''make world''`, and upgraded components using FreeBSD's package system. While effective (we never lost a node during an upgrade) and parallelizable (because nodes are homogenous), this method became impractical. As the number of nodes grew, more and more nodes were offline at any given time, so a single upgrade couldn't be parallelized to all nodes concurrently.

The `sup` solution. The first part of automating upgrades to the machines was to run Netbed's wrappers for `sup`, the software update protocol, on all of the nodes and have them automatically pull upgrades from the server. While pull-based, the `sup` upgrade can also be triggered via SSH to avoid the pull delay. The nodes now `sup` themselves from the Netbed control machine. Each node performs *two* `sup` attempts, the first from a global tree for all RON testbed nodes, and the second from a per-node tree. This allows us to customize individual nodes where necessary, without sacrificing automation.

CD-ROM boot system. `sup` alone did not solve all of the problems with the upgrade process. Kernel upgrades and major system upgrades were still risky, and a major error could shut the machine down until we had considerable assistance from the host. To combat this, and to speed the initial installation process, we moved to a CD-based boot system that we designed with Netbed. We now burn a customized CD-ROM for each node, and perform a brief initial burn-in. Node configuration now takes about 10 minutes of active user time.

With the CD, testbed nodes can survive the loss of the entire contents of their hard drive. When the CD boots, it validates the initial hard disk contents, and checks with Netbed to see if it should reload the drive from scratch. As long as we can reboot a corrupted node, we can install a fresh image. We considered placing two nodes at each site to allow them to control each other, but we believe the CD solution is nearly as good, and it is easier and cheaper to manage.

To reduce administrative involvement, the CD-ROM prompts users to see if they wish to change the IP address information for the node during start-up, so that the site hosts can reconfigure the machine with no administrative privilege.

4.6 Rebooting

To reduce cost and complexity, we did not ship power controllers with the RON testbed nodes. Of the 18 critical failures we've experienced, 10 of them only required that the machine be power cycled. 5 of these 10, however, were due to the aforementioned BIOS problems, so power control would have been ineffective. In 3 of the remaining cases, the kernel was still functional. We therefore decided that an interrupt-level kernel reboot mechanism would suffice: the Internet Ping of Death. While the machine is functional, it securely establishes a secret key that can trigger a reboot. If the machine receives an ICMP message with the proper secret key, it will reboot. While not applicable to all crashes, this mechanism is capable of resetting the machine under several resource exhaustion scenarios that have in the past crippled our machines.

4.7 Network Connectivity

After machine differences, the next most common problem we encountered was the presence of network firewalls. Many sites' filtering policies have interfered with experiments on the RON nodes. Utah filtered NFS traffic in and out of the CS department, which prevented the Ivy experiments from working there. Several sites filter NTP traffic to some degree, which prevented both NTP measurement experiments *and* time synchronization. One site, for unknown reasons, blocks port 7777, which the Netbed control daemon uses. Finally, many sites filter low ports like 4, used by the SFS filesystem.

As a result of these numerous headaches, our policy is that RON testbed nodes must be deployed as far outside the firewall as possible. To avoid innumerable problems, we do not place RON nodes behind NAT boxes.

4.8 User File Distribution

One area in which we could have provided better support for our users was in tools for distributing files to the nodes. We've found that most of our users have independently created scripts that `rsync` or `scp` data to and from all of the testbed nodes and execute commands on the remote nodes.

Providing the SFS [16] remote filesystem on all of the nodes provided a good interface for *interactive* testbed use, with users executing quick tests of their software directly from the remote filesystem. Long-running experiments, however, need a more reliable distribution mechanism. We used the system-wide *sup* capability to install specific globally-useful binaries on the nodes if users requested them, but this leaves a large middle ground uncovered.

5. CLOSELY RELATED TESTBEDS

Emulab [27], the Netbed local cluster, provides a "network in a bottle" via a centralized, reconfigurable cluster of nodes. Researchers can create networks of up to 100 nodes with arbitrary bandwidth and delays between the nodes. By using the same TCL-based configuration as the popular network simulator *ns* [25], it simplifies the transition from simulated experiments to running real code in an emulated environment. Emulab users can wipe disks, install operating systems, and test non-IP protocols. The price of this power is that researchers cannot test their protocols in a real wide-area environment. We now use the Netbed facilities to manage the RON testbed, providing a complete path from simulation to emulation to wide-area testing.

Planetlab [20] is a “global overlay network for developing and accessing new network services.” Planetlab aims to grow to thousands of distributed nodes that can be used for both research *and* long-term service deployment. Like RON testbed nodes, Planetlab nodes are more autonomous than Emulab nodes, running a single base operating system image. Planetlab nodes can be deployed alone or in local clusters.

Planetlab diverges from the RON testbed in its goal of being open to service development and deployment by (eventually) almost anyone. The RON testbed is primarily an experimentation platform, and researchers who wish to provide services to the Internet are encouraged to migrate their services elsewhere. In contrast, Planetlab embraces the idea of being a substrate for future Internet services. The cost of this openness is some loss of control by researchers, who give up more privileged access to the nodes for security. We hope to eventually host PlanetLab services on the RON nodes, while still making the nodes available for lower-level experiments.

6. FUTURE WORK

Our work with testbeds is far from complete. On one hand, we are working with the Planetlab effort, and hope that our experiences can prove useful to this new effort. On the other hand, because the two projects retain a different focus and capabilities, we are still working to improve the RON testbed. As Planetlab has shown, a vibrant user community is beneficial to the development of any testbed, and we have not provided sufficient infrastructure to connect our users to help them share tools and ideas. We continue to work with the Netbed and Planetlab researchers to reduce the administrative burden on both ourselves and our users.

Users with root access to the testbed find it extremely convenient, particularly when non-root mechanisms do not exist to provide the capabilities they require. While unhindered access cannot scale to larger numbers of users, we believe that the development of additional safe, “root-like” interfaces would benefit not just our testbed, but most operating systems. We are investigating this through FreeBSD’s “jail” mechanism, while the Planetlab effort is exploring such interfaces using Linux’s “vservers” capability.

Finally, we would like to increase the size of the RON testbed though the deployment of additional nodes and through alliances with other testbeds. This scaling requires the development of scalable probing and measurement facilities and will place new stresses on our management tools, a good proving ground for new techniques.

7. CONCLUSION

The MIT RON testbed is an ongoing research artifact that has been useful to many projects. We believe that its success, in large part, has been due to its diverse network connectivity, reasonable size, and easy to use, homogenous interface. The development and ongoing maintenance of this testbed presented us with a number of challenges, and we hope that the design principles and lessons we have learned will benefit the designers of future testbeds.

Acknowledgements

We would like to extend our deepest thanks to Leigh Stoller and Mike Hibler of the University of Utah for their implementations of the Netbed account management and node updating, and to Jay Lepreau for providing us with Netbed resources to manage the testbed. Nick Feamster provided the incentive and the ability to collect BGP feeds at the RON nodes. DARPA provided us with the financial resources to pursue this project, and without its support none of this would have happened.

We are grateful to the many users who both made our testbed useful, and provided us with the feedback that helped to improve it; this list includes, but is not limited to, Alex Snoeren, Alex Yip, Allen Miu, Amin Vahdat, Athicha Muthitacharoen, Benjie Chen, Chuck Blake, Constantine Dovrolis, David Mazieres, Elisha Ziskind, Emil Sit, Frank Dabek, Idit Keidar, Lopa Roychoudhuri, Michael Freedman, Michael Walfish, Nick Feamster, Ningning Hu, Omar Bakr, Peter Druschel, Roie Melamed, Thomer Gil, Timo Burkard, and Z. Morley Mao. Thank you all!

Without the time, space, and bandwidth donated by our site hosts, no experiments would have gotten off the ground. We thank Andre Barrette, Ant Rowstron, Bjorn Ablad, Chris North, Chris Pohlabel, Eric Bates, James Edwards, Jeffrey Papen, Jennifer Rexford, John Jannotti, John Todd, Joonbok Lee, Kees J Bot, Kent Lewin, Kevin Fall, Lars-Ake Larzon, Leigh Stoller, Mike Biesele, Mike Cutler, Mike Hibler, Mike Sanders, Nancy Miller, Randy Bush, Robbert van Renesse, Shawn Shira, Thanasis Douitsis, and Venkatesh S. Obanaik for hosting RON nodes.

8. REFERENCES

[1] Resilient overlay networks webpage. `http://nms.lcs.mit.edu/ron/`, 2002.

[2] Abone. `http://www.isi.edu/abone/`.

[3] ANDERSEN, D., BALAKRISHNAN, H., KAASHOEK, M., AND MORRIS, R. Resilient Overlay Networks. In *Proc. 18th ACM SOSP* (Banff, Canada, Oct. 2001), pp. 131–145.

[4] ANDERSEN, D. G., SNOEREN, A. C., AND BALAKRISHNAN, H. Best-path vs. multi-path overlay routing. In *Proc. Internet Measurement Conference* (Miami, FL, Oct. 2003).

[5] CAIRN Home Page. http://www.isi.edu/div7/cairn/, 1996.

[6] CHESWICK, B., BURCH, H., AND BRANIGAN, S. Mapping and visualizing the Internet. In *Proc. USENIX Technical Conference* (2000).

[7] DABEK, F., KAASHOEK, M. F., KARGER, D., MORRIS, R., AND STOICA, I. Wide-area cooperative storage with CFS. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP ’01)* (Banff, Canada, Oct. 2001).

[8] Dummynet. `http://www.iet.unipi.it/~luigi/ip_dummynet`, September 1998.

[9] FEAMSTER, N., ANDERSEN, D., BALAKRISHNAN, H., AND KAASHOEK, M. F. Measuring the effects of Internet path faults on reactive ro uting. In *Proc. Sigmetrics* (San Diego, CA, June 2003).

[10] FREEDMAN, M. J., AND MAZIÈRES, D. Sloppy hashing and self-organizing clusters. In *2nd International Workshop on Peer-to-Peer Systems (IPTPS)* (Berkeley, CA, Feb. 2003).

[11] Internet2. `http://www.internet2.edu/`.

[12] JACOBSEN, K., ZHANG, X., AND MARZULLO, K. Group membership and wide-area master-worker computations. In *Proc. 23rd ICDCS* (Providence, RI, 2003).

[13] JAIN, M., AND DOVROLIS, C. End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput. In *Proc. ACM SIGCOMM* (Aug. 2002).

[14] KATABI, D., AND BLAKE, C. Inferring congestion sharing and path characteristics from packet interarrival times. Tech. Rep. MIT-LCS-TR-828, MIT, 2002.

[15] MAO, Z. M. BGP beacons. `http://www.psg.com/~zmao/BGPBeacon.html`, 2002.

[16] MAZIÈRES, D., KAMINSKY, M., KAASHOEK, M. F., AND WITCHEL, E. Separating key management from file system security. In *Proc. SOSP* (Dec. 1999), pp. 124–139.

[17] MUTHITACHAROEN, A., MORRIS, R., GIL, T., AND CHEN, B. Ivy: A read/write peer-to-peer file system. In *Proceedings of the 5th USENIX Symposium on Operating Systems Design and Implementation (OSDI '02)* (Boston, MA, Dec. 2002).

[18] PAXSON, V. End-to-End Internet Packet Dynamics. In *Proc. ACM SIGCOMM* (Cannes, France, Sept. 1997), pp. 139–152.

[19] PAXSON, V., ADAMS, A., AND MATHIS, M. Experiences with NIMI. In *Proceedings of the Passive & Active Measurement Workshop* (Apr. 2000).

[20] PETERSON, L., ANDERSON, T., CULLER, D., AND ROSCOE, T. A blueprint for introducing disruptive technology into the Internet. In *Proc. HotNets-I* (Princeton, NJ, Oct. 2002).

[21] ROWSTRON, A., AND DRUSCHEL, P. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *Proc. 18th IFIP/ACM International Conference on Distributed Systems Platforms* (Nov. 2001).

[22] STOICA, I., MORRIS, R., KARGER, D., KAASHOEK, M. F., AND BALAKRISHNAN, H. Chord: A scalable peer-to-peer lookup service for Internet applications. In *Proc. ACM SIGCOMM* (San Diego, California, Aug. 2001).

[23] STRAUSS, J., KATABI, D., AND KAASHOEK, F. A measurement study of available bandwidth estimation tools. In *Proc. Internet Measurement Conference (IMC)* (Oct. 2003).

[24] TANG, L., AND CROVELLA, M. Virtual landmarks for the Internet. In *Proc. Internet Measurement Conference (IMC)* (Oct. 2003).

[25] THE VINT PROJECT. *The* ns *Manual*, Apr. 2002. http://www.isi.edu/nsnam/ns/ns-documentation.html.

[26] TOUCH, J. Dli95 dartnet overview. `http://www.isi.edu/~touch/dli95/dartnet-dli.html`, 1995.

[27] WHITE, B., LEPREAU, J., STOLLER, L., RICCI, R., GURUPRASAD, S., NEWBOLD, M., HIBLER, M., BARB, C., AND JOGLEKAR, A. An integrated experimental environment for distributed systems and networks. In *Proc. OSDI* (Boston, MA, Dec. 2002), pp. 255–270.