SatelliteLab: Adding Heterogeneity to Planetary-Scale Network Testbeds

Marcel Dischinger
MPI-SWS

Andreas Haeberlen
MPI-SWS / Rice University

Ivan Beschastnikh
University of Washington

Krishna P. Gummadi
MPI-SWS

Stefan Saroiu
University of Toronto

ABSTRACT

Planetary-scale network testbeds like PlanetLab and RON have become indispensable for evaluating prototypes of distributed systems under realistic Internet conditions. However, current testbeds lack the heterogeneity that characterizes the commercial Internet. For example, most testbed nodes are connected to well-provisioned research networks, whereas most Internet nodes are in edge networks.

In this paper, we present the design, implementation, and evaluation of SatelliteLab, a testbed that includes nodes from a diverse set of Internet edge networks. SatelliteLab has a two-tier architecture, in which well-provisioned nodes called *planets* form the core, and lightweight nodes called *satellites* connect to the planets from the periphery. The application code of an experiment runs on the planets, whereas the satellites only forward network traffic. Thus, the traffic is subjected to the network conditions of the satellites, which greatly improves the testbed's network heterogeneity. The separation of code execution and traffic forwarding enables satellites to remain lightweight, which lowers the barrier to entry for Internet edge nodes.

Our prototype of SatelliteLab uses PlanetLab nodes as planets and a set of 32 volunteered satellites with diverse network characteristics. These satellites consist of desktops, laptops, and handhelds connected to the Internet via cable, DSL, ISDN, Wi-Fi, Bluetooth, and cellular links. We evaluate SatelliteLab's design, and we demonstrate the benefits of evaluating applications on SatelliteLab.

Categories and Subject Descriptors

C.4 [**Computer Systems Organization**]: Performance of Systems; C.2.1 [**Computer Systems Organization**]: Computer-Communication Networks—*Network Architecture and Design*

General Terms

Experimentation, measurement, performance

SIGCOMM'08, August 17–22, 2008, Seattle, Washington, USA.

Keywords

Internet, network testbeds, distributed systems

1. INTRODUCTION

Internet testbeds, such as PlanetLab [23] and RON [1], have become indispensable for evaluating networking and distributed systems research. Researchers deploy prototypes of new systems over these testbeds and study their performance to estimate how well these systems would work over the real Internet. PlanetLab, the state-of-the-art Internet testbed, has been used by over a thousand researchers for evaluating several hundred Internet-scale distributed systems, including peer-to-peer systems [27], routing, multicast, and QoS overlays [1, 7, 31], content distribution networks [35], as well as for network measurements [29]. Recently the success of PlanetLab has inspired numerous efforts to build next-generation testbed environments, such as GENI [14] and FIRE [11].

In this paper, we focus on a widely recognized problem with existing network testbeds: they lack the heterogeneity that characterizes the commercial Internet [28]. Most nodes in existing testbeds are located in well-connected academic and corporate networks, and the network paths between them are often restricted to well-provisioned research backbones [5]. This is in sharp contrast to the Internet, where most end nodes connect via a diverse set of edge networks, such as residential broadband, wireless, and cellular networks. Prior studies have shown that the paths in PlanetLab have very different characteristics from the paths in the commercial Internet [5,25]. More alarmingly, researchers have found that the behavior of some systems can vary considerably between Internet and testbed deployments [17].

Our basic idea is to improve the heterogeneity of current testbeds by recruiting nodes from the Internet edges. These nodes can be desktops, laptops, or handhelds, and they can be connected to the Internet via residential broadband, wireless, or cellular networks. However, adding edge nodes with diverse resource constraints to existing testbeds creates new challenges. Current testbeds are designed to be hosted by large institutions in academia and industry, and testbed administrators expect dedicated, well-provisioned nodes that are connected to the Internet via high-speed networks. For example, PlanetLab nodes are required to be server-class machines, they must run PlanetLab OS, and they must be configured with static, public IP addresses [24]. While these requirements make the testbed easy to manage and easy to use, they are also a high barrier to entry for nodes in Internet edge networks, many of which are hosted and managed

by individual end users. For instance, most edge nodes are not server-class machines, and many cannot get a public, static IP address from their ISPs.

The need to support nodes with a wide range of hardware, software, networking, and management resources requires us to fundamentally rethink existing testbed designs. While current architectures treat all nodes equally, we propose a hierarchical testbed architecture that assigns different roles to nodes based on their available resources.

We present the design, implementation, and evaluation of SatelliteLab, a highly heterogeneous testbed that includes nodes from a diverse set of edge networks. SatelliteLab has a two-tier architecture. The nodes in the top tier are called *planets*; they play the role of classical well-provisioned testbed nodes. The nodes in the bottom tier belong to a new class of light-weight testbed nodes called *satellites*. In existing testbeds, each node performs two tasks: it executes application code, and it forwards traffic over its access links. The key insight behind SatelliteLab's design is to separate these two tasks. Satellites do not run any application code; they delegate this task to well-provisioned planets. Instead, satellites collaborate with the planets to detour traffic along the Internet links to which they are connected. Thus, SatelliteLab's routing design subjects traffic to the network conditions that it would experience if the application was run on the satellites. In this way, satellites improve the heterogeneity of a testbed by contributing access to edge links without having to contribute any resources for code execution.

SatelliteLab's architecture allows us to leverage existing testbed infrastructures like PlanetLab by augmenting them with satellite nodes from the Internet edges. It also significantly lowers a testbed's barrier to entry while preserving its ease of use and ease of management. We show that an end host can support the limited routing functionality of a satellite by running just a few hundred lines of Java code. Researchers who have been using the existing testbed do not need to modify their application code to use SatelliteLab; they only need to specify which satellites should forward their application traffic. Finally, because satellites do not execute application code, they do not need to be managed or monitored by testbed administrators.

To understand the extent to which our design facilitates recruiting of end hosts in edge networks, we implemented SatelliteLab as an extension to the popular PlanetLab testbed. Within a period of two weeks, the five authors of this paper were able to recruit 32 satellite nodes from our friends and colleagues, who were willing to donate their spare network resources to our experiments. These nodes included desktops, laptops, and handhelds that were being actively used by our contributors, and they were connected to the Internet through a variety of residential cable, DSL, ISDN, Wi-Fi, Bluetooth, and cellular links. For comparison, only five out of more than 800 PlanetLab nodes are located in such edge networks. Our experience suggests that SatelliteLab can potentially scale to thousands of nodes if the several hundred researchers using PlanetLab collaborate to grow the testbed.

We used our preliminary deployment to evaluate SatelliteLab's design. Our results show that, despite the limited functionality supported by the satellites, SatelliteLab can forward traffic from testbed experiments over heterogeneous edge networks. Further, we demonstrate that SatelliteLab can be used to conduct realistic experiments in the radically heterogeneous network environments that comprise today's Internet. We illustrate our testbed's benefits by presenting a brief evaluation of network coordinate and overlay multicast systems in residential broadband environments, as well as a preliminary measurement study of UMTS [32] cellular networks.

The rest of this paper is organized as follows. We begin by making the case for recruiting Internet edge nodes as testbed nodes in Section 2. Next, we discuss the challenges and design requirements for a testbed using edge nodes in Section 3, and we describe the design and implementation of SatelliteLab in Sections 4 and 5. In Section 6, we present evaluation results demonstrating that SatelliteLab can closely approximate the network characteristics of a real deployment, and in Section 7, we illustrate the benefits of SatelliteLab by using it to evaluate two distributed systems and to perform a measurement study. Finally, we discuss related work in Section 8 and conclude the paper with Section 9.

2. BACKGROUND AND MOTIVATION

In this section, we make the case for recruiting nodes from Internet edge networks to improve the heterogeneity of testbeds. We first demonstrate a pressing need for more diverse testbeds, and then show how adding a small number of edge nodes can dramatically improve the heterogeneity of PlanetLab, the state-of-the-art Internet testbed.

2.1 The need for more heterogeneous testbeds

The lack of heterogeneity in current Internet testbeds is widely known and has been identified as an important concern by the designers of PlanetLab [28]. This creates several challenges for networking researchers: networked system designs cannot be rigorously evaluated in an environment that lacks the diversity present in the Internet, protocols are shielded from complications caused by middle boxes (such as proxies and NATs) that are present on many Internet paths, and network measurement results are not always representative and are difficult to generalize.

Many research projects have already shown these consequences either directly [5, 25] or indirectly by pointing out surprising differences between evaluations performed in testbeds and those performed in highly heterogeneous networks. For example, the paths between PlanetLab nodes and the paths in access networks have different reliability and packet loss characteristics [15]. Evaluation of the Vivaldi network coordinate system showed that the accuracy of the system differs vastly between a testbed and the Internet at large [17]. A recent study of residential networks found that transport protocols, such as TCP, behaved differently in broadband access networks than in PlanetLab [9]. Finally, it is already widely accepted that PlanetLab does not capture the characteristics specific to network paths in wireless and mobile environments [4].

Current testbeds lack heterogeneity in multiple dimensions: the participating nodes have similar hardware configurations, run similar software, and are connected to networks with similar characteristics. While all of these aspects are important, we believe that a diverse set of network paths is particularly important for Internet testbeds. Unlike simulators and emulators, testbeds can accurately capture path properties to provide a realistic network environment for high-level protocols and applications. A testbed with poor network path diversity adversely impacts evaluations

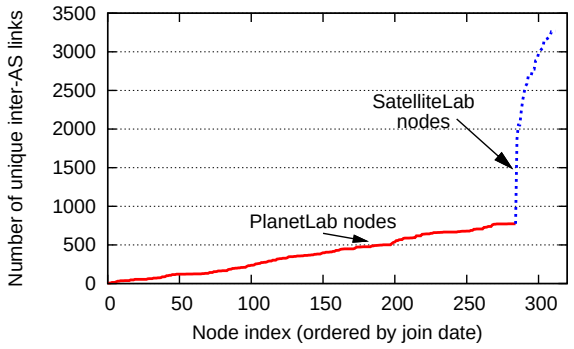


Figure 1: Inter-AS links covered by PlanetLab vs. SatelliteLab: By adding a small number of edge nodes to PlanetLab, we can increase the number of inter-AS links covered by the testbed paths by more than a factor of three.

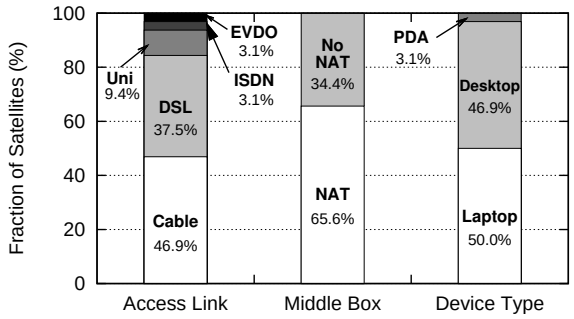


Figure 2: Heterogeneity in SatelliteLab: Our 32 testbed nodes were connected to various access networks, such as DSL, cable, and EVDO. Less than 10% of nodes are located in University (Uni) networks. Many of the nodes were located behind NATs, and some of them were mobile.

by causing them to be less realistic, which ultimately detracts from the value of the testbed itself.

2.2 Adding edge nodes increases testbed heterogeneity

Our proposal is to augment testbed heterogeneity by recruiting nodes from Internet edge networks. To illustrate the potential benefits of this approach, we compare the diversity of network paths in the PlanetLab testbed with paths in SatelliteLab, a testbed prototype we built by extending PlanetLab with 32 nodes from a variety of edge networks. Since all of our SatelliteLab nodes are in Europe and North America, we only consider PlanetLab nodes in these regions here to ensure comparability.

Although PlanetLab continues to attract new participants, additional nodes do not necessarily improve a testbed's path heterogeneity. We illustrate this by using the number of distinct inter-AS links that are covered by paths in a testbed as a proxy for path heterogeneity. Figure 1 shows that adding a new node to PlanetLab increases the AS-path heterogeneity only slowly; on average, each additional node increases coverage by only 2.7 inter-AS links. We believe this is because most PlanetLab nodes are located in closely coupled academic networks and thus the AS paths between them are similar. However, Figure 1 also shows that, if we increase PlanetLab's size by just 10% using nodes from the commercial Internet, we can more than triple its coverage of inter-AS links.

Figure 2 further illustrates the diversity of our SatelliteLab testbed. In contrast to PlanetLab, most nodes in SatelliteLab are located behind a diverse set of access links, including cable and DSL. A few are even connected using Bluetooth and cellular links. Two-thirds of the nodes are behind NATs, and half of the nodes are mobile devices, such as laptops and handhelds, that use Wi-Fi. Thus, adding nodes from Internet edge networks can considerably improve the heterogeneity of Internet testbeds in multiple dimensions.

3. CHALLENGES AND REQUIREMENTS

In this section, we discuss the challenges in enabling arbitrary end hosts, including resource-constrained nodes in Internet edge networks, to be used for testbed experiments. From this discussion, we derive two primary requirements that our SatelliteLab design must satisfy.

3.1 Goals

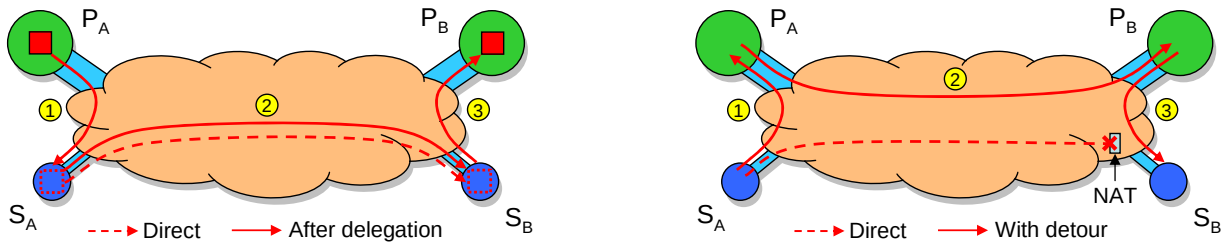
Our basic design goal is to preserve the numerous benefits of existing testbeds. For example, PlanetLab provides experimenters with a stable software environment, supports complete management of private virtual slices, and offers an extensive API on top of which useful distributed services can be built. We share all of these goals, and additionally want to support heterogeneous edge nodes. In the remainder of this section, we will outline the important challenges and requirements particular to this last goal.

3.2 Challenges

Recruiting volunteer nodes from the edge of the Internet imposes challenges to SatelliteLab's design that are fundamentally different from the challenges faced by existing testbeds such as PlanetLab. We describe three challenges unique to SatelliteLab below.

1. Edge nodes provided by volunteers are not dedicated testbed nodes. Most of the edge nodes we recruited for our testbed were personal computers owned by friends and colleagues, who were willing to forward experiment traffic in the background using their spare resources. Based on our experience, we believe that volunteers will resist giving up administrative control over their systems and will not agree to installing a particular OS. This is in contrast to node management in PlanetLab and RON, where sites are required to share (or even relinquish) control over their root account. We also realized that, due to security and accountability concerns, contributors do not want to run arbitrary experiment code on their machines. For example, a BitTorrent experiment could cause others to suspect copyright violations, and a network measurement could generate complaints about unwanted traffic. These concerns must be addressed because they will discourage many volunteers from participating. The challenge is to do so while imposing as few management requirements on edge nodes as possible.

2. Edge nodes often have limited storage and processing resources. We cannot make strong assumptions about the capabilities of participating edge nodes. They may be laptops, handhelds, or cell phones with limited storage



(a) Delegating code execution to planets

(b) Detouring traffic via the planets

Figure 3: Mechanisms used in SatelliteLab: (a) Satellites choose a nearby planet to run application code on their behalf, and (b) traffic is detoured through the planets because satellites cannot communicate directly.

and processing resources. Thus, the testbed nodes' software stack must be flexible and light-weight, which contrasts with inflexible policies of existing testbeds. For example, PlanetLab requires nodes to be x86-based server-class machines with fast CPUs and large amounts of memory and storage. The challenge is to allow all sorts of nodes – even restricted ones – to contribute to the testbed.

3. Edge nodes are often located behind middle boxes. It is well known that many Internet users are connected to broadband access networks that almost always use dynamic IP addresses, NATs, and/or firewalls. Existing testbeds like PlanetLab or RON require nodes to have publicly reachable, static IP addresses and DNS names that can be resolved with both forward and reverse lookups. To elicit broad participation and adoption, these requirements must be relaxed. The challenge here is to design a testbed using nodes that may not be able to communicate directly with each other.

3.3 Requirements

From our discussion above, we identify two crucial design requirements:

1. The testbed design should accommodate nodes that cannot run application code. Even resource-constrained nodes should be able to participate, and node owners should retain administrative control.

2. The testbed design should accommodate nodes that cannot communicate directly with each other. For example, nodes behind NATs or firewalls should be able to join the testbed.

4. DESIGN

In this section, we present the design of SatelliteLab, a testbed that can accommodate nodes from a diverse set of edge networks. We start with a high-level overview of our testbed architecture. Next, we describe two key mechanisms to overcome the design challenges we identified in the previous section. Finally, we show how SatelliteLab leverages these mechanisms to create a highly heterogeneous network testbed.

4.1 Overview

At a high level, SatelliteLab has a two-tier architecture. The nodes in the upper tier – the *planets* – are well-provisioned and professionally managed nodes located in high-capacity research networks. The nodes in the lower tier – the *satellites* – belong to a new class of light-weight testbed nodes that is introduced by SatelliteLab. Unlike planets, satellites can be any type of end host with Internet connectivity, such as desktops, laptops, or handhelds, including hosts behind NATs and firewalls. Satellites are owned and managed by individual users.

Separating testbed nodes into powerful planets and light-weight satellites enables us to assign different responsibilities to the two node types. This distinction is crucial to meeting the two design requirements imposed by inclusion of satellites: their inability to execute arbitrary experiment code or to communicate directly with each other. We first describe two mechanisms that address these requirements and then explain how SatelliteLab works.

4.2 Delegating code execution to the planets

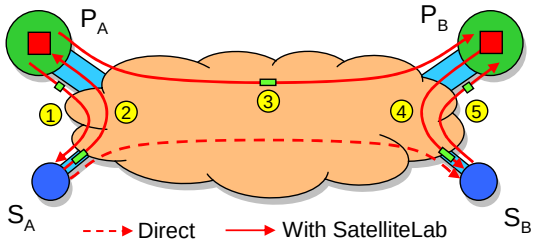
Since satellites cannot run application code, we associate each satellite with a nearby planet and run the application code on that planet. However, this changes the network path that connects the application instances. In particular, if the traffic were sent directly from one well-connected planet to another, we would miss the access links of the satellites, which are often the bottlenecks on Internet paths and have a significant impact on the path characteristics [9].

To subject the application traffic to the network conditions that exist along the direct paths between the satellites, SatelliteLab must re-route packets via the satellites. Figure 3(a) illustrates this through an example in which two satellites, S_A and S_B, delegate the execution of their application instances to two nearby planets P_A and P_B.

There are two problems with this approach: First, the traffic along the path segment between the satellites is often blocked by NATs and firewalls, making the path unusable. Second, compared to the direct path between satellites, the data packets traverse the access links of the satellites twice, once each in the upstream and the downstream directions. We propose a detour routing-based mechanism in Section 4.3 to solve the first problem. We resolve the second problem in Section 4.4, where satellites do not receive and forward actual data packets, but instead they use small 'control' packets in one direction, and 'dummy' packets that are of the same size as the data packets in the other direction.

4.3 Detouring traffic via the planets

To avoid the path segment between two satellites, we can detour all traffic between satellites via their closest planets. Figure 3(b) shows an example. When satellite S_A needs to send a packet to another satellite S_B, the packet is first sent to the nearest planet P_A, then forwarded to the planet P_B



#	Direction	Packet Type	Size
1	$P_A \rightarrow S_A$	Control	40 bytes
2	$S_A \rightarrow P_A$	Dummy	(same as data)
3	$P_A \rightarrow P_B$	Data	(variable)
4	$P_B \rightarrow S_B$	Dummy	(same as data)
5	$S_B \rightarrow P_B$	Control	38 bytes

Figure 4: SatelliteLab paths: The application instances run on the planets, but the traffic between them is detoured through the satellites.

nearest to the target satellite S_B, and finally delivered to satellite S_B.

We expect the detour path latency, loss, and throughput characteristics to be similar to the direct path between the satellites for the following two reasons. First, the detour path and the direct path typically share the same bottleneck links, namely the access links of the satellites. Second, because each satellite is associated with the planet closest to it (in the network topology), the base latency between the planets will approximate the latency between the satellites. Our evaluation results in Section 6 show that these expectations typically hold in practice.

Note that when detouring is used, each satellite only needs to communicate with a single node – the nearest planet. In the final protocol, satellites are required to register with this planet a priori, and are not allowed to communicate with any other node. This ensures that satellites will not send traffic to arbitrary nodes, even if a malicious non-testbed node sends them forged packets. This constraint shields contributors from complaints about unwanted traffic, and provides accountability for all traffic generated by the satellites.

4.4 How SatelliteLab works

Our design of SatelliteLab is based on a combination of these two insights. First, satellites delegate the execution of application code to nearby planets. The planets in turn forward the application traffic via the satellites to subject it to the network conditions that exist on the path between the satellites. Finally, satellites communicate by forwarding their traffic through their nearby planets, overcoming their inability to communicate with each other directly.

Figure 4 illustrates in detail how SatelliteLab works. The figure shows two satellites S_A and S_B, as well as their closest planets P_A and P_B. Each planet is running an instance of the application that is being evaluated. When the instance on planet P_A needs to send a data packet to the instance on planet P_B, P_A sends a small control message to its satellite S_A (1) and instructs it to forward a 'dummy' packet along the detour path. The dummy packet has the same size as the data packet and is initially empty. When the dummy packet reaches P_A (2), it is filled with the actual data and forwarded to P_B (3). At P_B, the data is removed, and a dummy packet is forwarded to S_B (4). When the packet arrives at S_B, S_B responds with a small control message to P_B acknowledging its receipt (5). Finally, when P_B receives this receipt, it delivers the original packet to its local application instance.

The detour path chosen by SatelliteLab includes two path segments, S_A to P_A and P_B to S_B, in addition to the direct path between the planets. While these two segments introduce additional delays and losses, SatelliteLab minimizes their effects by (a) selecting the planets closest to the satellites to minimize the extra latency, and (b) keeping the sizes of the control packets traversing the segments to just a few bytes to minimize the additional network load. Section 6 demonstrates that the additional delays and losses introduced by the detour path are minimal in practice.

5. IMPLEMENTATION

Although our design can leverage any existing testbed infrastructure, we implemented SatelliteLab as an extension to the PlanetLab testbed. In this section, we describe the technical details of our implementation. The code is available from `http://satellitelab.mpi-sws.mpg.de/`.

5.1 Overview

Our implementation of SatelliteLab has two components: a *planet proxy* and a *satellite helper*. Each PlanetLab node runs a planet proxy and each satellite runs a satellite helper. The role of the planet proxy is threefold: it intercepts network traffic sent by applications, it forwards network traffic along the appropriate detour path, and it communicates with the helpers running on satellites that are assigned to its local node. Our planet proxy runs in user space on the PlanetLab nodes; this helped with deployment, but also prevented us from using fast kernel-level hooks to intercept network traffic [34]. Thus, our planet proxy is subject to delays associated the kernel's scheduling policy.

5.2 The planet proxy

We implemented the planet proxy as a Linux daemon process in approximately 2,400 lines of C++ code.

5.2.1 Intercepting application traffic

When the proxy is started, it creates a virtual Ethernet device (a TAP device [33]) configured with a private subnet, such as `10.0.0.0/8`. This private subnet is shared by all planet proxies and all application instances running on the planets. To run an application on SatelliteLab, the experimenter only needs to configure it to bind to the TAP device's IP address. Thus, all traffic sent by the application is intercepted by its local planet proxy.

When two application instances on the same planet exchange traffic, this traffic would normally be delivered directly by the kernel without being routed through the TAP device. To avoid this, we use a simple technique borrowed from Modelnet [34]. Each application modifies certain bits in all destination addresses to ensure that they appear as remote addresses; once the planet proxy intercepts a packet, these bits are set back to their original value. This technique uses a dynamic library and works with unmodified application binaries.

5.2.2 Communicating with the satellites

The proxy interacts with its satellites using a very simple probe/response protocol. The proxy can send a d-byte UDP

message `PROBE(i,d,u)`, where i is an identifier and u is the size of the requested response packet. In response, the satellite sends a u-byte UDP message `RESPONSE(i,u)`. To mask the occasional loss of packets, each probe (response) message includes information about the two most recently exchanged probe (response) packets between the proxy and the satellite.

Let A and B be application instances running on planets P_A and P_B, respectively; let S_A and S_B be the corresponding satellites, and let σ be the size of the data packet in bytes. The complete packet forwarding process is as follows:

1. P_A chooses an identifier i and stores the packet in an internal buffer indexed by i.
2. P_A sends `PROBE(i,12,`σ`)` to S_A, which responds with `RESPONSE(i,`σ`)`.
3. P_A retrieves the packet and forwards it to P_B.
4. P_B sends `PROBE(i,`σ`,10)` to S_B, which responds with `RESPONSE(i,10)`.
5. P_B retrieves the packet and delivers it to B.

This forwarding process subjects application traffic to network conditions on the access links of the satellites. Since the access links are often the bottlenecks of the direct network path between the satellite nodes, the detour paths have similar access link delays, losses, and bandwidths, as the direct paths.

5.3 The satellite helper

We implemented the satellite helper in Java to enhance its portability across different software and hardware configurations. Our Java implementation has 118 lines of code, as counted by the number of semicolons. Additionally, we developed OS-specific packages – one for Windows, one for Mac OS X, and one for Linux. These packages have installers and are integrated with the respective OSes; for example, the Windows version can be minimized to the system tray.

During startup, the helper pings several planet nodes and then registers with the one closest to it in terms of network latency. Then it waits for, and responds to, any incoming UDP probes from that planet. To minimize the possibility of complaints or abuses directed at the satellite's owner, the helper does not communicate with any other node.

5.3.1 Handling NATs and DHCP

In our experience, many satellites are behind a NAT. This creates two challenges for the satellite helper: First, it knows its local IP address, but not its publicly visible IP address, and second, control packets from the planet cannot reach it unless it first initiates a connection. We use a heartbeat mechanism to solve both problems. The satellite helper periodically sends a small status message to its planet, which allows the proxy to find out the satellite's publicly visible IP. These packets also prevent the address translation rule in the NAT from expiring and thus ensure that the satellite remains reachable from the planet.

Some satellites acquire their IP address using DHCP. This can create problems if DHCP reassigns a satellite's IP address. In this case, the planet proxy must no longer send probes to that address. We also use the heartbeat mechanism to handle this problem: when the proxy no longer receives status messages from a satellite, it stops sending probes to that satellite.

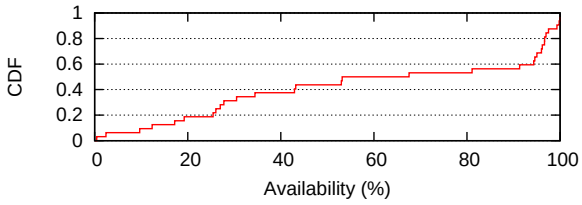


Figure 5: Satellite availability: Availability varied widely among the different satellite node types.

5.3.2 Allocating satellites to experiments

Like the PlanetLab testbed, SatelliteLab can be used by researchers to run different experiments. In our implementation, we restrict satellites to participate in only one experiment at a time, which prevents interference between experiments and avoids overloading the satellite's access link. However, researchers can serially allocate satellites to different experiments over time. Our implementation seamlessly enables a satellite to leave one experiment and to join another. For this, the planet proxy can either bind the satellite to a different application running on its local planet, or the satellite can re-register with a different planet.

We believe that allocation of satellites to different experiments should be subject to a testbed-wide policy. This policy would depend on SatelliteLab's incentive mechanism for attracting satellites. There are a variety of incentive schemes suitable for experiments of different scales. For small-scale experiments, researchers can convince their friends to deploy satellites on their personal machines. Experiment requiring hundreds or thousands of nodes can use a tit-for-tat incentive scheme in which contributors are granted priority when they utilize the testbed. Researchers using SatelliteLab may also provide per-node-hour financial compensation to contributors or conduct a lottery [10]. A detailed discussion of incentives is outside the scope of this paper.

6. EVALUATION

At a high level, our evaluation answers three questions about SatelliteLab's design. First, we illustrate one of its key advantages – a lower barrier to entry. Second, we show that although satellites have lower availability than planets, their session durations are adequate for running most experiments. Lastly, we demonstrate that the characteristics of SatelliteLab's detour paths closely match those of the direct paths between the satellites.

6.1 SatelliteLab makes it easy to recruit edge nodes

To support satellite nodes with a variety of software profiles, we implemented the satellite helper in Java. We created easy-to-install packages for Windows, Mac OS X, and Linux, and we set up a web page with instructions for installing our software. To compare SatelliteLab's detour paths to direct paths, we added a test harness that enabled us to send packets directly between the satellites where possible. Since this test harness makes the nodes vulnerable to abuse, we disabled it once we had completed our evaluation experiments.

In a period of only two weeks, we recruited 32 satellite nodes by asking our friends, family members, and colleagues to install SatelliteLab on their private machines (Table 1).

#	Location	Access link	NAT	Type	Mob.
1	Canada	Cable	no	Desktop	no
2	Canada	DSL	no	Desktop	no
3	Canada	Uni+Wi-Fi	no	Laptop	yes
4	Canada	Uni+Wi-Fi	no	PDA	yes
5	Canada	Cable+Wi-Fi	yes	Laptop	no
6	Germany	DSL+Wi-Fi	yes	Desktop	no
7	Germany	Cable	no	Desktop	no
8	Germany	DSL	yes	Desktop	no
9	Germany	DSL+Wi-Fi	yes	Laptop	yes
10	Germany	Cable+Wi-Fi	yes	Laptop	yes
11	Germany	DSL	yes	Desktop	no
12	Germany	DSL+Wi-Fi	yes	Desktop	no
13	Germany	ISDN+BT	no	Laptop	no
14	Hungary	DSL	yes	Laptop	yes
15	Portugal	Cable	no	Laptop	no
16	UK	DSL	no	Laptop	yes

#	Location	Access link	NAT	Type	Mob.
17	CA, USA	DSL+Wi-Fi	yes	Laptop	no
18	CA, USA	EVDO	no	Laptop	no
19	CO, USA	Cable+Wi-Fi	yes	Laptop	no
20	IL, USA	Cable	yes	Desktop	no
21	LA, USA	DSL	yes	Desktop	no
22	MA, USA	Cable+Wi-Fi	no	Laptop	yes
23	MD, USA	Uni	no	Desktop	no
24	MD, USA	Cable+Wi-Fi	yes	Laptop	yes
25	NJ, USA	DSL+Wi-Fi	yes	Laptop	no
26	NJ, USA	Cable+Wi-Fi	yes	Laptop	no
27	TX, USA	Cable+Wi-Fi	yes	Desktop	no
28	WA, USA	Cable	yes	Desktop	no
29	WA, USA	Cable	yes	Desktop	no
30	WA, USA	Cable+Wi-Fi	yes	Desktop	no
31	WA, USA	Cable+Wi-Fi	yes	Laptop	yes
32	WI, USA	DSL	yes	Desktop	no

Table 1: Overview of the satellite nodes: Our nodes use a variety of access links such as cable, DSL, wireless (Wi-Fi), Bluetooth (BT), or high-capacity access links located in Universities (Uni). Some of the nodes' access links combine two types of networks; for example, a DSL+Wi-Fi means that the host is connected to a DSL modem via a wireless link. The last column indicates whether or not the host was mobile.

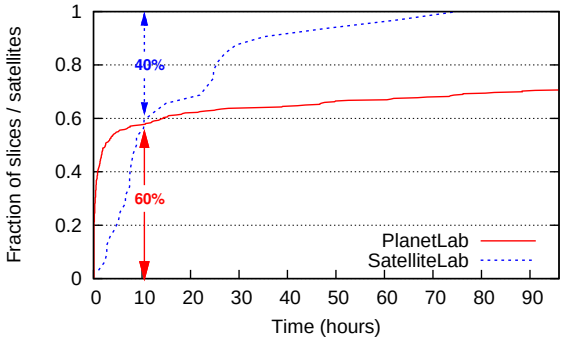


Figure 6: Run-time of experiments on PlanetLab: 40% of our satellites had median session times longer than 10 hours, while 60% of PlanetLab slices did not run any experiment that lasted more than 10 hours.

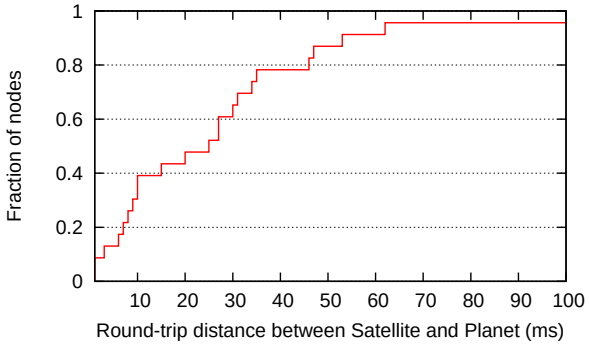


Figure 7: Minimum distance between a satellite and its closest PlanetLab node: 80% of the satellites in our testbed had an RTT of 35 ms or less.

Our testbed includes sixteen satellite nodes from the U.S., eight nodes from Germany, five nodes from Canada, and one node each from Hungary, Portugal, and the United Kingdom. These nodes connect via DSL, cable, ISDN, and cellular links; many of them have an extra wireless hop (802.11 or Bluetooth), and several of them are behind NATs.

As demonstrated in Section 2.2, SatelliteLab's lower barrier to entry does not just improve access link diversity, it increases the heterogeneity of other testbed characteristics as well. For example, our deployment included mobile nodes (a PDA and several laptops), which were connected to different access networks at different times (e.g., to the university network at work and to a cable network at home). Also, as shown in Figure 5, satellite uptimes varied between close to zero and close to 100% because some of the nodes were switched off overnight and/or were occasionally suspended (e.g., for travel). We believe that this additional heterogeneity could be equally valuable to experimenters.

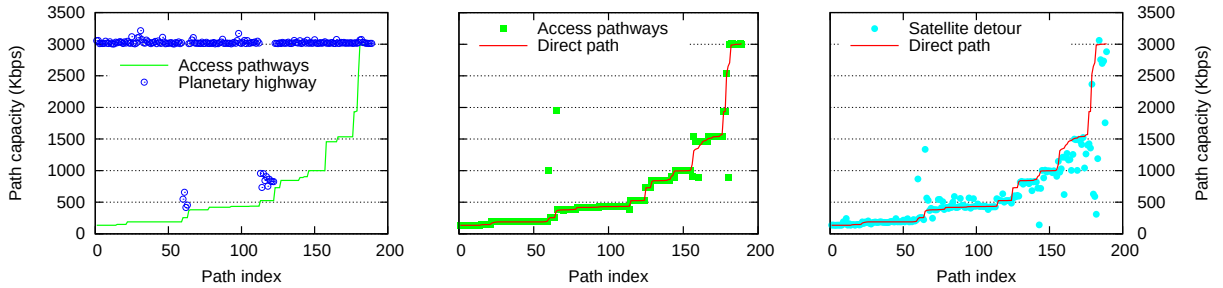
Based on our initial experience, we believe that if the several hundred researchers using PlanetLab collaborated and each recruited a handful of satellites, future SatelliteLab deployments could grow to include thousands of nodes in Internet edge networks.

6.2 The availability of satellites is adequate for many testbed experiments

To test whether availability of SatelliteLab nodes is sufficient to support potential experiments, we compared the median session times of our satellites to run-times of experiments on PlanetLab. We used the CoTop monitor data [8] for the month of November 2007 to compute for each slice the longest continuous session during which a node sent or received data over the network. A session ended when a node sent and received no traffic for 15 minutes. Figure 6 compares this to the median node availabilities in our testbed; it shows that 40% of satellites have enough availability to support the workloads of about 60% of the PlanetLab slices. The figure also shows that a significant proportion of slices are active for days or weeks. Most of these slices are services, such as CoMon and OpenDHT, or long-lived measurement experiments. SatelliteLab was not designed to support such workloads (see Section 7 for further discussion).

6.3 Satellites can find planets in their close proximity

Detouring application traffic through satellites leads to an increase in latency as compared to the direct path. However,



(a) Planetary highway vs. Access pathways (b) Direct path vs. Access pathways (c) Direct path vs. Satellite detour

Figure 9: Path capacities: The capacity bottlenecks are typically on the access links, so they are shared by both the access pathway and the direct path. As a result, the capacity of the satellite detour matches well with that of the direct path.

we expect this increase to be small with respect to the overall latency as long as satellites can find a planet in their close proximity. Figure 7 shows that 80% of the satellites had a round-trip time (RTT) of less than 35 ms to the nearest PlanetLab host, and all but one had an RTT of less than 62 ms. The satellite using EVDO had an RTT of 109 ms because of the high transmission delays in cellular networks.

6.4 Detour and direct paths are bottlenecked at the same access links

The characteristics of Internet paths are driven by their bottlenecks. In edge networks, path bottlenecks often occur on the "last mile" – that is, at or close to the access link [9]. Because both the direct path and the detour path between two satellites share this "last mile", they are likely to share the same bottleneck. This observation is the key to understanding why the characteristics of SatelliteLab's detour paths closely match those of the direct paths.

We conducted a series of experiments to show how the capacity, jitter, and loss rates of detour paths are similar to those of direct paths. We measured five different paths for each pair of satellites (Figure 8): the *direct path* between the satellites, the *satellite detour* path used between the two satellites by SatelliteLab, and the three components of the satellite detour, namely the *planetary highway* and two *access pathways*. The planetary highway is the segment of the satellite detour between the PlanetLab nodes, while an access pathway is the segment between a PlanetLab node and a satellite. We used two probe types to measure each of the five paths[1]:

Small UDP ping/pong probes: We sent a long sequence of small (100-byte) UDP ping/pong packets along the paths. We paced the ping packets using a Poisson distribution with a mean sending rate of one packet per second; the pongs were returned immediately after the receipt of a ping. Since the average data rate was just 1 Kbps, the probe packets reflect the RTTs experienced by packets under normal operating conditions.

Large UDP flood probes: We sent large (1,000-byte) UDP probes at the rate of 3 Mbps along the paths. Each flood lasted for three seconds and typically saturated the bottleneck links, which were below 3 Mbps in most cases. These probes reflect the conditions of the paths under load.

[1]The number of paths measured with these probes can be different because not all testbed nodes were available at the same times. For the small ping/pong probes we used only paths for which at least three hours of measurements were available.

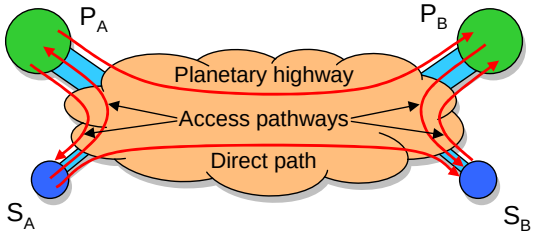


Figure 8: Paths used for evaluation: We separated the detour path into two segments, the planetary highway and the access pathways.

6.4.1 Path capacity

We studied path capacities by measuring the bandwidth of their bottleneck links. When designing SatelliteLab, we expected the detour and the direct path to have the same path capacity because they share the same access links, which often tend to be the bottlenecks. To verify this, we estimated the capacities of paths from the packet delivery rate of the large UDP[2] flood probes [9]. For each path, we took the maximum across all the measurements to remove noise from potential cross-traffic at the bottleneck link.

1. Access pathways are the bottleneck of the satellite detour paths. To understand where the capacity bottlenecks in the satellite detours are located, we compared the capacities of their constituent planetary and access pathways. Figure 9(a) shows capacities of 3 Mbps for most planetary highways, which suggests that their bottleneck links were not saturated by our 3 Mbps floods. In contrast, most access pathways show capacities of less than 2 Mbps.

2. Access links shared by the direct and the detour paths are often the capacity bottlenecks. Figure 9(b) compares the bottleneck capacities of the access pathways to those of the direct paths between the satellites. The path capacities closely match, suggesting that direct and detour paths share their bottlenecks on the access links.

3. SatelliteLab's detour routing preserves the direct path capacities. Finally, we plot the capacities of direct

[2]We found that most NATs allow UDP-based hole punching, while many have trouble allowing TCP-based hole punching.

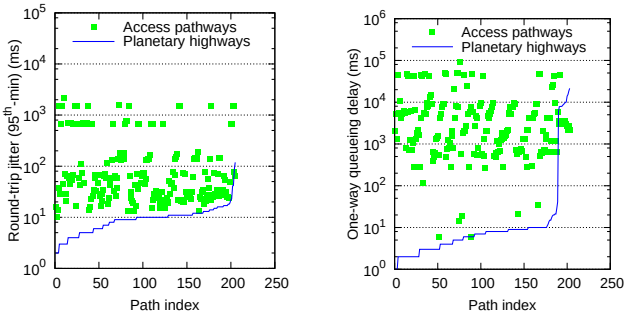


Figure 10: A comparison of (a) jitter and (b) queueing delay between access pathways and planetary highways.

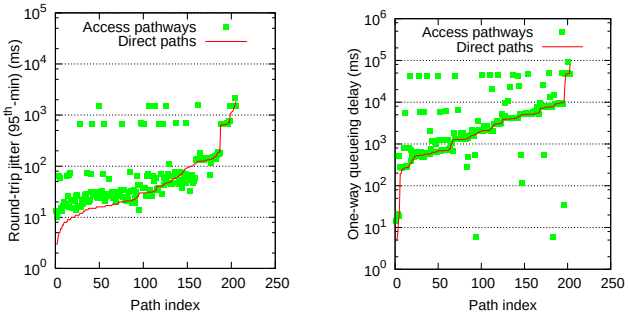


Figure 11: A comparison of (a) jitter and (b) queueing delay between access pathways and direct paths.

Loss rate	$< 0.1\%$	0.1-0.5%	0.5-1.0%	$\geq 1.0\%$
Planetary hwy	95.4%	4.6%	0.0%	0.0%
Access pathway	38.1%	41.7%	7.8%	12.4%
Satellite detour	25.4%	46.0%	10.3%	18.4%
Direct path	66.4%	18.4%	5.3%	9.9%

Table 2: Packet loss rates along different paths.

and detour paths in Figure 9(c). As before, the capacities closely match; the differences are within 10% in almost all cases.

6.4.2 *Path jitter and queuing delay*

We define *path jitter* as the variation in packet RTTs due to queueing at bottleneck routers along the path, and we estimate it as the difference between the 95th percentile path RTT and the minimum path RTT. We define *queueing delay* as the maximum increase in one-way delay for a path under load, and we measure it using the large UDP floods that saturate the path bottleneck router by filling its queue. We estimate queueing delay as the difference between the minimum and the 95th percentile one-way path delay.

1. Queuing occurs primarily along the access links. Since path bottlenecks are likely at the access links, we expect the access pathways to dominate jitter and queueing delay. Figure 10(a) compares the jitter along planetary highways and access pathways for different satellite detours (note that the vertical axis is a log scale). The results show that the jitter along the access pathways is significantly higher than jitter along the planetary highways. This indicates that queueing primarily occurs along the access links. Figure 10(b) confirms this result: most planetary highways have low queuing delay, whereas access paths experience significant queuing delays, often exceeding 1 second.

2. Access pathways and direct paths have matching jitter and queueing delay. As detour and direct paths share the access link, we expect them to experience similar jitter and queueing delay. Figure 11(a) demonstrates that jitter along both paths closely matches. As before, Figure 11(b) confirms the similarity of queueing delays.

6.4.3 *Path loss rates*

To compare loss rates for detour and direct paths, we used the UDP ping/pong probe experiment data. We recorded a path failure event whenever three or more consecutive packets were lost (most of these episodes lasted at least for three seconds). A vast majority of the measured paths (over 90%) did not show any path failure events at all. The remaining paths typically experienced one or two path failures during a single six-hour experiment. Next we computed the paths' average loss rates ignoring the losses during the path failure events.

Table 2 shows the observed loss rates for different paths. Overall, the loss rates were low across all the paths, which is consistent with recent studies of edge networks in the Internet [9]. The table shows that all measured planetary highways experienced loss rate below 0.5%, while 20.2% of the access pathways, 28.7% of the satellite detours, and 15.3% of the direct paths saw loss rates above 0.5%. This suggests that most of the packet loss occurs at the satellite node access links.

6.5 Summary

Our evaluation demonstrates that it is possible to use SatelliteLab to enlist a highly heterogeneous set of nodes as satellites. We showed that the availability of edge nodes is adequate for running many testbed experiments, and our measurements of a 32 node SatelliteLab testbed indicate that important path characteristics, such as capacity, jitter, and loss rates of direct paths between satellites are typically preserved in SatelliteLab's detour paths.

7. APPLICATIONS

Testbeds like PlanetLab have been generally used for three broad classes of experiments: deployments of public Internet services, such as Coral [12] and CoDeeN [36]; evaluation of networked systems; and Internet measurement studies. SatelliteLab cannot accommodate the first class of experiments because application code cannot run on the satellites. However, the remaining two classes of experiments can greatly benefit from using SatelliteLab. SatelliteLab's node and network diversity allow researchers more flexibility in selecting an appropriate set of nodes for their experiments. This flexibility makes it possible to experiment with settings that closely mirror the intended deployment, more so than is possible today with PlanetLab. In the remainder of this section, we illustrate the benefits of using SatelliteLab for evaluating networked systems and for performing Internet measurement studies.

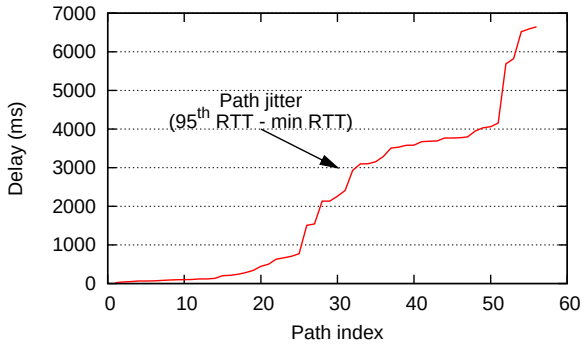


Figure 12: Measured jitter in a network coordinate scheme: Paths between satellites running BitTorrent experience very high jitter.

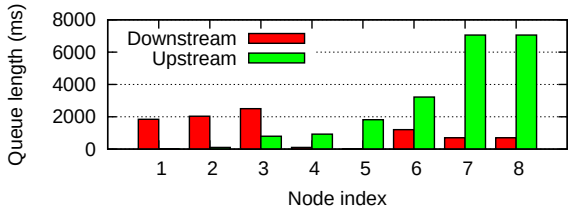


Figure 13: Queue sizes of access routers for satellites: Our broadband satellites have very large queue sizes both upstream and downstream.

7.1 Evaluation of networked systems

We used SatelliteLab to evaluate two popular networked systems: a network coordinate system and an overlay multicast system. Although both systems have been previously evaluated over PlanetLab, our evaluation over SatelliteLab led us to substantially different conclusions. As we show, this is due to the characteristics of the satellites' network links, which differ drastically from the links between PlanetLab nodes.

7.1.1 Network coordinate system

Internet systems use network coordinate systems to cheaply and rapidly obtain estimates of network latency. The basic idea is to assign a set of coordinates to participating nodes, which can then be used to obtain rough estimates of network latencies. Although PlanetLab experiments have shown network coordinate systems to be accurate, a recent study found that their accuracy is significantly lower when they are used by BitTorrent participants [17]. In this study, the latency variations between the BitTorrent hosts were so high that the network coordinate system failed to converge on a single coordinate set. However, the authors could not explain the cause of these variations.

We used SatelliteLab to investigate this phenomenon. We began by repeating the experiment described in [17] on a smaller scale. We used eight broadband hosts as satellites, and we installed the Azureus BitTorrent client on their corresponding planets. The *seeder* host served a large file, and seven *leechers* downloaded it. The experiment re-started when all hosts finished downloading the file.

We duplicated the findings of [17] with little effort. Figure 12 plots the distribution of the jitter in the latencies measured across the paths among the eight SatelliteLab nodes. More than half of the paths experienced a jitter of more than two seconds!

In addition to reproducing the results, SatelliteLab enabled us to find an explanation for them. We measured the queue sizes of the access routers for the eight broadband hosts using techniques from [9]. We found that all access routers had long downstream and upstream queues. As Figure 13 shows, almost all routers had a queue length of at least one second's worth of traffic. As BitTorrent hosts exchange large amounts of data, bottleneck queues fill up and cause the hosts' latency and jitter to vary by several orders of magnitude. This explains the findings in [17].

7.1.2 SplitStream overlay multicast

In another experiment, we used SatelliteLab to evaluate SplitStream, a tree-based overlay multicast system that streams content from a source to a set of client nodes [7]. SplitStream was evaluated on PlanetLab, but it has been shown that the performance of such systems depends strongly on the available bandwidth [30]. Therefore we expected its performance on SatelliteLab to be very different from performance observed on PlanetLab.

In our experiments, we used the FreePastry 2.0_02 [13] implementation of SplitStream configured with $k = 16$ trees. We ran three trials with five nodes each in three different environments – a local cluster, PlanetLab, and SatelliteLab. We used two metrics of SplitStream's performance: the chunk delivery ratio (CDR), which measures the fraction of chunks successfully delivered, and the latency of the chunk delivery.

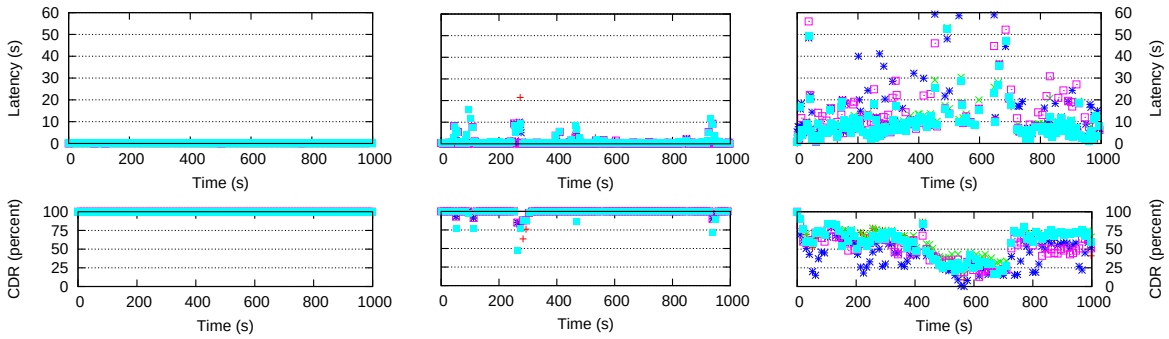
Figure 14 shows our results. As we anticipated, SplitStream's performance on SatelliteLab was much worse than its performance on PlanetLab or on the cluster. On the cluster, performance was almost ideal; each node in the cluster received all content with minimal delay. On PlanetLab, performance was only slightly lower; even though the nodes experienced occasional losses, the delays and the throughput were consistently high. SatelliteLab nodes, however, experienced peak delays three times above those of PlanetLab nodes and suffered from significant throughput variations.

7.2 Internet measurement studies

To illustrate the benefits of SatelliteLab as a platform for Internet measurement, we present a small-scale study of TCP throughput over cellular links. TCP flows are likely to experience highly fluctuating throughput because cellular links suffer from interference, spotty coverage, and poor signal strength. Today, little is known about the characteristics of such links and how they affect the behavior of TCP flows.

Using SatelliteLab, we ran a series of TCP transfers between a mobile laptop and a well-provisioned server. The laptop was equipped with a UMTS modem and communication software that always chose the available network with the highest data rate (GPRS or UMTS). Each transfer ran for 30 seconds and was spaced 4 minutes apart from the previous one. For each transfer we recorded the average throughput.

Figure 15 shows the cumulative distribution function of our TCP throughput measurements over UMTS/GPRS both upstream and downstream. We find that downstream flows can have a throughput of anywhere between 10 Kbps to 320 Kbps (an order of magnitude difference!). In con-



(a) Local cluster (b) PlanetLab (c) SatelliteLab

Figure 14: SplitStream experiment: Latency between transmission and reception of each delivered chunk, and fraction of chunks delivered. The five point types represent the five nodes in the experiment.

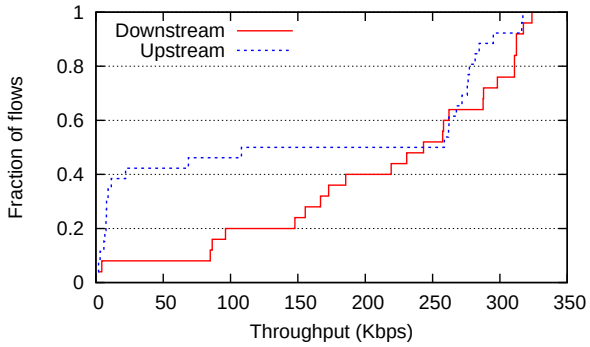


Figure 15: Cumulative distribution function of TCP throughput over UMTS: TCP over cellular links can experience very different throughput based on whether UMTS is available or whether it has to fall-back to GPRS.

trast, upstream flows have a bi-modal distribution. Half of them have very low throughput of up to 10 Kbps, whereas the other half have high throughput of over 250 Kbps. This simple experiment illustrates how SatelliteLab can be used to conduct measurement studies of new network environments.

7.3 Summary

Our experiments illustrate several key benefits of SatelliteLab. Even at small scale, SatelliteLab allows networking researchers to evaluate their prototypes over highly heterogeneous networks. By contrasting their SatelliteLab evaluations to PlanetLab, researchers can gain additional insight into the behavior of their systems in diverse network environments. When a system behaves in a surprising manner on the Internet, SatelliteLab can be used to re-create some properties of the environment to investigate the underlying cause of a system's behavior. Finally, SatelliteLab can be used to conduct measurement studies of different network environments, including wireless and broadband networks.

8. RELATED WORK

Simulation/Emulation: Network simulators like ns-2 [21] and emulators like Modelnet [34] are useful for exploring complex parameter spaces and enable experimentation under controlled conditions, which ensures repeatability. SatelliteLab complements these tools by offering a highly realistic environment with real user behavior, a deployed network, and real-world effects such as routing instabilities or congestion.

Testbeds: A variety of testbeds are in use by the academic community, including PlanetLab [23], RON [1], NIMI [22], and VINI [6]. However, their nodes are biased towards academic networks and commercial links are severely underrepresented [25]. SatelliteLab offers a way to remove this bias by augmenting testbeds with other link types such as broadband links donated by residential users.

Flexlab [26] closely couples a testbed with a network emulator, parameterizing the emulator's network properties with values measured on the testbed. This is a compromise between realism and control; in particular, it makes experiments repeatable. SatelliteLab does not use a model; it directly measures network conditions at per-packet granularity, which provides additional realism but less control.

Volunteer computing: Several projects have explored the idea of leveraging resources provided by end users. Condor [18] was one of the first systems to utilize idle workstations for computation. BOINC [2] is a more recent platform built to support volunteer supported computational projects. SETI@home [3] which uses spare CPU time to analyze radio signals, and NETI@home [20] which gathers statistics for research purposes are examples of applications built on top of BOINC. End-system multicast [16] uses spare network capacity to distribute content, and ePOST [19] uses spare disk capacity to store email messages. Like all these projects, SatelliteLab relies on donated resources, except that these resources are used to improve the heterogeneity of today's network testbeds.

9. CONCLUSIONS

We presented SatelliteLab, a system that adds heterogeneity to existing planetary-scale network testbeds. SatelliteLab introduces satellites, a new class of lightweight testbed nodes located in a diverse set of edge networks, including broadband and wireless networks. Satellites do not run application code, nor do they forward traffic to each other. Instead, satellites only communicate with nearby planet nodes. In this way, SatelliteLab can easily recruit participants from the Internet at large, even if their machines do not meet

the requirements for joining the core testbed. Our evaluation shows that SatelliteLab captures the heterogeneity of the Internet edges while preserving a realistic testbed network environment. Our experiments conducted over both PlanetLab and SatelliteLab showed a number of differences stemming from the additional heterogeneity in SatelliteLab. This confirms the benefits of using SatelliteLab as a testbed for evaluating networked systems and for performing Internet measurements.

10. ACKNOWLEDGMENTS

We would like to thank all our friends and colleagues who helped us with our evaluation by hosting SatelliteLab nodes. We also thank Harsha Madhyastha for providing us with tools for our analysis, and Peter Druschel for his valuable feedback on the early design of SatelliteLab. Finally, we are grateful to our shepherd David Andersen and our anonymous reviewers for providing detailed and helpful feedback on this paper.

11. REFERENCES

[1] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient Overlay Networks. In *Proc. of SOSP '01*, Banff, AB, Canada, Oct 2001.

[2] D. P. Anderson. BOINC: A System for Public-Resource Computing and Storage. In *Proc. of the 5th International Workshop on Grid Computing*, Pittsburgh, PA, Nov 2004.

[3] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer. SETI@home: An Experiment in Public-resource Computing. *CACM*, 45(11):56–61, 2002.

[4] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. H. Katz. A Comparison of Mechanisms for Improving TCP Performance over Wireless Links. *IEEE/ACM ToN*, 5:756–769, Dec 1997.

[5] S. Banerjee, T. G. Griffin, and M. Pias. The Interdomain Connectivity of PlanetLab Nodes. In *Proc. of the Passive and Active Measurement Conference*, Antibes Juan-les-Pins, France, Apr 2004.

[6] A. C. Bavier, N. Feamster, M. Huang, L. L. Peterson, and J. Rexford. In VINI Veritas: Realistic and Controlled Network Experimentation. In *Proc. of SIGCOMM'06*, Pisa, Italy, Sep 2006.

[7] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. SplitStream: High-Bandwidth Multicast in Cooperative Environments. In *Proc. of SOSP'03*, Lake George, NY, Oct 2003.

[8] CoTop Monitoring Tool. `http://codeen.cs.princeton.edu/cotop/`.

[9] M. Dischinger, A. Haeberlen, K. P. Gummadi, and S. Saroiu. Characterizing Residential Broadband Networks. In *Proc. of IMC'07*, San Diego, CA, Oct 2007.

[10] J. R. Douceur and T. Moscibroda. Lottery Trees: Motivational Deployment of Networked Systems. In *Proc. of SIGCOMM'07*, Kyoto, Japan, Aug 2007.

[11] FIRE: Future Internet Research and Experimentation. `http://cordis.europa.eu/fp7/ict/fire/`.

[12] M. J. Freedman, E. Freudenthal, and D. Mazières. Democratizing Content Publication with Coral. In *Proc. of NSDI'04*, San Francisco, CA, Mar 2004.

[13] The FreePastry web site. `http://freepastry.org/`.

[14] GENI: Global Environment for Network Innovations. `http://www.geni.net`.

[15] K. P. Gummadi, H. Madhyastha, S. D. Gribble, H. M. Levy, and D. J. Wetherall. Improving the Reliability of Internet Paths with One-hop Source Routing. In *Proc. of OSDI'04*, San Francisco, CA, Dec 2004.

[16] Y. hua Chu, A. Ganjam, T. S. E. Ng, S. G. Rao, K. Sripanidkulchai, J. Zhan, and H. Zhang. Early Experience with an Internet Broadcast System Based on Overlay Multicast. In *USENIX Annual Technical Conference*, Boston, MA, Jun 2004.

[17] J. Ledlie, P. Gardner, and M. Seltzer. Network Coordinates in the Wild. In *Proc. of NSDI'07*, Cambridge, MA, Apr 2007.

[18] M. Litzkow, M. Livny, and M. Mutka. Condor - A Hunter of Idle Workstations. In *Proc. of ICDCS'88*, San Jose, CA, Jun 1988.

[19] A. Mislove, A. Post, A. Haeberlen, and P. Druschel. Experiences in Building and Operating ePOST, a Reliable Peer-to-Peer Application. In *Proc. of EuroSys'06*, Apr 2006.

[20] NETI@home. `http://neti.gatech.edu/`.

[21] ns-2 Network Simulator. `http://www.isi.edu/nsnam/ns/`.

[22] V. Paxson, A. K. Adams, and M. Mathis. Experiences with NIMI. In *Proc. of Symposium on Applications and the Internet*, Nara, Japan, Feb 2002.

[23] PlanetLab. `http://www.planet-lab.org/`.

[24] Planetlab hosting requirements. `http://www.planet-lab.org/hosting/`.

[25] H. Pucha, Y. C. Hu, and Z. M. Mao. On the Impact of Research Network Based Testbeds on Wide-area Experiments. In *Proc. of IMC'06*, Rio de Janeiro, Brazil, Oct 2006.

[26] R. Ricci, J. Duerig, P. Sanaga, D. Gebhardt, M. Hibler, K. Atkinson, J. Zhang, S. Kasera, and J. Lepreau. The Flexlab Approach to Realistic Evaluation of Networked Systems. In *Proc. of NSDI'07*, Cambridge, MA, Apr 2007.

[27] A. Rowstron and P. Druschel. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. In *Middleware'01*, Heidelberg, Germany, 2001.

[28] N. Spring, L. Peterson, A. Bavier, and V. Pai. Using PlanetLab for Network Research: Myths, Realities, and Best Practices. *SIGOPS OSR*, 40(1):17–24, 2006.

[29] N. Spring, D. Wetherall, and T. Anderson. Scriptroute: a public internet measurement facility. In *Proc. of USITS'03*, Seattle, WA, 2003.

[30] K. Sripanidkulchai, A. Ganjam, B. Maggs, and H. Zhang. The Feasibility of Supporting Large-Scale Live Streaming Applications with Dynamic Application End-Points. In *Proc. of SIGCOMM'04*, Portland, OR, Aug 2004.

[31] L. Subramanian, I. Stoica, H. Balakrishnan, and R. H. Katz. OverQoS: Offering Internet QoS using Overlays. *SIGCOMM CCR*, 33(1):11–16, 2003.

[32] The 3rd Generation Partnership Project. `http://www.3gpp.org/`.

[33] Universal TUN/TAP driver. `http://vtun.sourceforge.net/tun/index.html`.

[34] A. Vahdat, K. Yocum, K. Walsh, P. Mahadevan, D. Kostic, and D. Becker. Scalability and Accuracy in a Large-Scale Network Emulator. In *Proc. of OSDI'02*, Boston, MA, Dec 2002.

[35] L. Wang, V. Pai, and L. Peterson. The Effectiveness of Request Redirection on CDN Robustness. *SIGOPS OSR*, 36(SI):345–360, 2002.

[36] L. Wang, K. Park, R. Pang, V. S. Pai, and L. Peterson. Reliability and Security in the CoDeeN Content Distribution Network. In *USENIX Annual Technical Conference*, Jun 2004.