Workload and Failure Characterization on a Large-Scale Federated Testbed

Brent N. Chun
Intel Research Berkeley
bnc@intel-research.net

Amin Vahdat
University of California, San Diego
vahdat@cs.ucsd.edu

ABSTRACT

Recently, a number of federated distributed computational and communication infrastructures have emerged, including the Grid, PlanetLab, and Content Distribution Networks. In these environments, mutually distrustful autonomous domains pool resources together for their mutual benefit, for instance to gain access to: unique computational resources, multiple vantage points on the network, or more computation than available locally. Key challenges for such federated infrastructures include resource allocation, scheduling, and constructing highly available services in the face of faulty end hosts and unpredictable network behavior. Developing such appropriate mechanisms and policies requires an understanding of the usage characteristics and operating environment of the target environment. In this paper, we present a detailed characterization of the actual use of the PlanetLab network testbed. PlanetLab consists of 240 nodes spread across 100 autonomous domains with over 500 active users. Using a variety of measurement tools, we present a three-month study on the network, CPU, memory and disk usage of individual PlanetLab nodes and sites. On the consumer side, we further characterize the consumption of individual users. Next, we present results on the availability and reliability of system nodes and the network interconnecting them. Finally, we discuss the implications of our measurements for emerging federated environments.

1. INTRODUCTION

A number of forces have contributed to the recent popularity of federated, distributed computation and communication infrastructures. The vision of a computational grid [13] promises the ability to leverage statistical multiplexing and unique hardware resources across the network to carry out computations larger than might be possible within any single site or administrative domain. Next, the advent of large-scale distributed systems such as distributed hash tables, peer-to-peer file sharing, and network measurement and characterization has lead a number of researchers to build personal testbeds consisting of available machines at various points in the network. These testbeds facilitate distributed system development, evaluation as well as network measurement and characterization. Given commonality in the requirements of this community—a set of available machines at a diversity of sites across the network—a variety of shared testbeds have been developed over the years. PlanetLab [18] is the latest, largest and perhaps most advanced of these testbeds. Finally, on the production side, companies deploying content distribution networks and shared hosting environments are considering techniques to pool their resources across trust boundaries to more cost effectively deliver high levels of performance and availability to their customers.

These emerging federated testbeds are growing to significant size, geographic and administrative diversity. For instance, as of October 2003, the PlanetLab infrastructure consisted of 240 nodes at over 100 distinct administrative domains in 19 countries and 500 active users. As the size and reach of a federated infrastructure grows, important challenges include resource discovery, scheduling, resource allocation policies, and reliability among mutually distrustful users and sites. Clearly, the appropriate mechanisms and policies depend on the exact usage characteristics of the system under consideration. For instance, if resources were never constrained, very simple resource allocation policies would be appropriate.

While such federated infrastructures are growing in popularity and importance, there is currently little understanding of how the resources are actually used. Thus, the goal of this work is to characterize the aggregate, per-site, and per-user resource consumption characteristics of the PlanetLab testbed. In addition, since system resources are under the control and administration of a wide variety of authorities, we also measured the availability of the testbed, with an eye toward the mechanisms appropriate for supporting reliable large-scale distributed systems.

We instrumented the PlanetLab infrastructure to capture a broad range of per-node characteristics and present the results of our study over a three-month period, from July-October 2003. Our high-level findings include: i) the system transitions from periods of light load to periods of heavy contention, ii) a small number of users account for the majority of system activity, iii) active distributed services typically remain active for less than 5 minutes, though some services remain active for weeks, iv) when averaged across a day and the entire infrastructure, the majority of services consume less than 1% of a single machine's resources in aggregate, v) most nodes demonstrate high levels of availability and low mean times to repair; however, the tail is long with 10% of nodes demonstrating extremely low levels of reliability, vi) node failures can be correlated significantly beyond the level predicted by correlation of node failures at a single site.

Of course, we cannot claim that the specifics of our measurements are representative of how such federated infrastructures may be used in general. However, we discuss the implications of our measurements for emerging PlanetLab infrastructure services in Section 6. Further, we believe that the general trends displayed by our testbed are likely to reflect at least some of the characteristics of emerging federated distributed systems.

2. PLANETLAB OVERVIEW

Our study examines PlanetLab, an open, global network testbed for developing, deploying and accessing widely distributed network services. The goal of PlanetLab is to grow to 1000 geographically distributed nodes situated in a variety of diverse locations on the Internet (e.g., colocation centers, edge sites, etc.). PlanetLab targets services that require broad geographic coverage for reasons including leveraging multiple vantage points on the network, providing physical proximity to data sources and sinks, providing multiple independent failure domains, and spanning multiple administrative and political boundaries.

In October 2003, the testbed consisted of 240 nodes at over 100 sites in 19 countries. It has been in production use since July 2002, currently supports over 120 active research projects and over 500 users around the world. The model for sites joining PlanetLab is that a site contributes some set of local resources (e.g., 2-3 machines plus network connectivity) and joins the testbed. In exchange, the site gains access to remote resources at other sites. Implicit here is the idea that remote resources are more valuable than local resources. This idea follows naturally given the nature of the widely distributed network services PlanetLab aims to enable.

Slice	Service
cmu5	IrisNet [10]: XML-based distributed query processing
mit4	Chord [21]: Distributed lookup, distributed hash table
northwestern2	Nemo: Resilient overlay multicast protocol
princeton6	Sophia [22]: Prolog-based distributed query processing
princeton9	CoDeeN [17]: Open content distribution network
tennessee7	IBP [8]: Internet Backplane Protocol
ucb5	PIER [16]: Distributed query engine
ucb8	Bamboo: Churn resilient distributed hash table
utah1	Emulab [25]: Emulab-PlanetLab integration service
uw9	ScriptRoute [20]: Network measurement/debugging

Table 1: Example wide-area services running continuously on PlanetLab.

The abstraction of a *slice* is fundamental. A slice is a horizontal cut of global PlanetLab resources. A slice comprises a network of virtual machines spanning some set of physical nodes, where each virtual machine (VM) is bound to some set of local per-node resources (e.g., CPU, memory, network, disk). Just as processes serves as the fundamental OS abstraction for single-node applications, slices serve as the distributed abstraction for widely distributed network services. Throughout the paper, when we refer to slices, we are referring to network services that are running in particular slices. Table 1 provides a handful of example services which currently run continuously on PlanetLab.

Currently, users access their slice through `ssh` access to the private virtual machine[1] residing on all global nodes that make up the slice. Currently, PlanetLab allocates resources to competing virtual machines on a best effort basis. Hence, the local nodes operating system, currently a heavily patched Linux 2.4.19 kernel, allocates memory, CPU, network bandwidth, and disk storage according to demand with no per-user resource arbitration. However, a number of efforts [6, 7, 9, 14] are investigating appropriate resource allocation mechanisms and policies. One of the motivations for this study is to gain an understanding of resource usage models on PlanetLab to gain a better understanding of the appropriate resource allocation policies. While our specific conclusions are restricted to PlanetLab's usage patterns over a recent three month time period, we believe the general trends are likely applicable to the broad class of emerging federated, networked computation and communication infrastructures.

[1]Currently, we use lightweight Linux *vservers* for this mechanism. However, investigating the appropriate structure for such virtual machines is an active area of research [11, 24].

3. MONITORING DATA

We use monitoring data from five different sources in our analysis. Our five sources of monitoring data include:

- **AllPairsPing** Minimum, average, and maximum ping times (over 10 ping attempts) between all pairs of nodes in PlanetLab. Measurements were taken and collected approximately every 30 minutes from each node. Failed ping attempts were also recorded.
- **Ganglia** Node resource statistics collected by Ganglia using the `/proc` filesystem. Measurements were taken every 15 seconds. However, to save storage, Ganglia compacts older per-node information. Thus, the long-term node data used was available only as per-day averages.
- **PLNetflow** Number of packets, number of bytes sent, network protocol, source and destination IP addresses, and source and destination ports for every slice on every node. Updated data was collected every five minutes.
- **Scout** Number of bytes sent and received for each slice virtual machine on every node. Updated byte counts were taken every five minutes from a special directory in `/proc` (i.e., `/proc/scout`), which exports per-slice statistics on PlanetLab nodes.
- **SliceStat** CPU, physical memory, and network bandwidth usage for each virtual machine on every node. Bandwidth usage was computed over 1, 5, and 15 minute windows. Measurements were taken every five minutes and logged to local files.

Data Set	Dates	MaxNodes	Size (MB)
AllPairsPing	2003-07-01 to 2003-10-01	211	1214
Ganglia	2003-07-01 to 2003-10-01	192	129
PLNetflow	2003-09-10, 2003-09-15	166	8331
Scout	2003-07-01 to 2003-10-01	176	5480
SliceStat	2003-08-22 to 2003-10-04	152	4543

Table 2: Summary of monitoring data sets.

For each source, measurements were taken locally on individual nodes and periodically archived at a centralized location. In each case, the set of nodes being monitored varied over time but generally tended to cover a large fraction of PlanetLab. Variation in the node sets was due primarily to the introduction of new PlanetLab nodes and the time for the maintainers of the various monitoring systems to incorporate these nodes into their monitoring by installing their monitoring software. Table 2 summarizes each of the five data sets used in terms of time coverage, number of nodes covered, and the size of the raw monitoring data. All five monitoring sources used are publicly available to users on PlanetLab. Long term archives for two of the five data sources are also available on the web [2].

[2]Archives for AllPairsPing can be found at `http://www.pdos.lcs.mit.edu/~strib/pl_app`. Archives for Scout can be found at `http://www.planet-lab.org/logs/scout-monitor`.

4. WORKLOAD MEASUREMENT AND ANALYSIS

4.1 Global Resource Demand

Figures 1, 2 and 5 plot aggregate CPU, network, and disk utilization across all PlanetLab nodes between July 1 and October 1 2003. PlanetLab is a dynamic, constantly evolving system. Hence, the set of nodes present in the system did not stay constant during the course of the experiment, with the number of global nodes increasing from 132 to 169 over the three month period, as depicted in Figure 1. The modest increase in system size is insufficient to explain the widely varying system load. For each of the graphs, the x-axis depicts time progressing in days while the y-axis plots aggregate resource utilization averaged across the entire day.

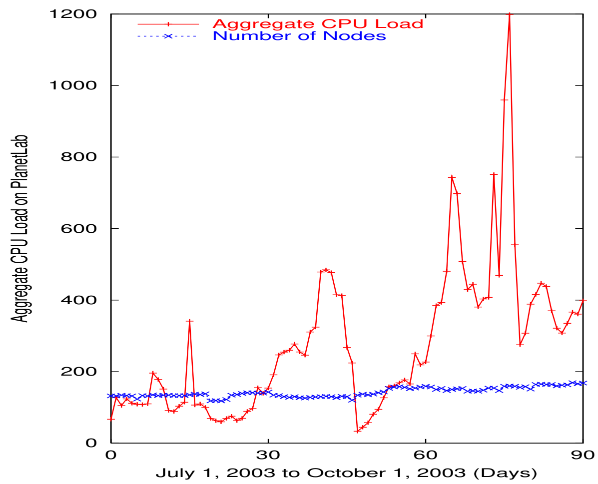


Figure 1: Aggregate CPU load on PlanetLab based on the Ganglia data set.

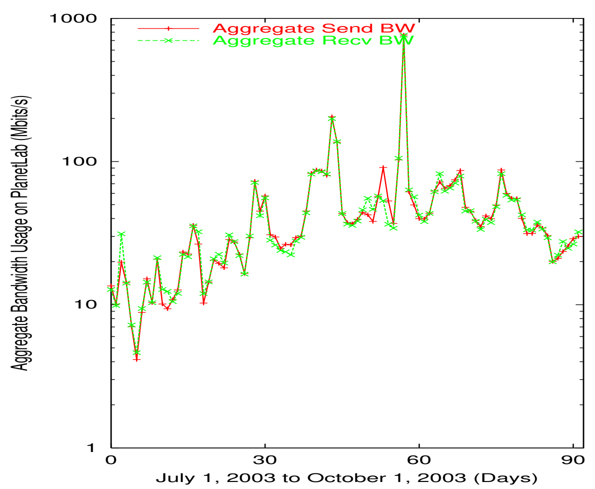


Figure 2: Aggregate send and receive bandwidth usage on PlanetLab based on the Scout data set. Note the large bandwidth spike on August 27, 2003.

These results show that CPU and network utilization are bursty. CPU utilization varies by an order of magnitude, going from a low point of an aggregate load average of 100 system wide (meaning 100 processes are in the ready queue averaged over each five minute interval) to an aggregate load average of 1200. Note that these values are averaged over an entire day; intra-day bursts place significantly larger load on the system. Network utilization is even more bursty, varying by more than 2 orders of magnitude. Aggregate send and receive bandwidth varies from approximately 5 Mbps (aggregate, system-wide) averaged across a single day to 800 Mbps aggregate. Note that 1 Gbps of aggregate bandwidth corresponds to approximately 10 TB of data transferred across PlanetLab during the course of the day. Figure 3 zooms in on the average per-hour resource consumption on the heaviest day in the trace, August 27, 2003. This figure shows that while the system averaged approximately 800 Mbps during the day, per-hour bandwidth varied from a peak of nearly 1.3 Gbps sustained over an hour (0200 UTC) to approximately 50 Mbps for the last few hours of the day. Finally, Figure 4 shows bandwidth consumption averaged over 5-minute intervals (the finest data granularity available) for the three busiest hours (0100-0400 UTC) on August 27, 2003. Bandwidth usage climbs steadily from approximately 400 Mbps to 1.5 Gbps over a

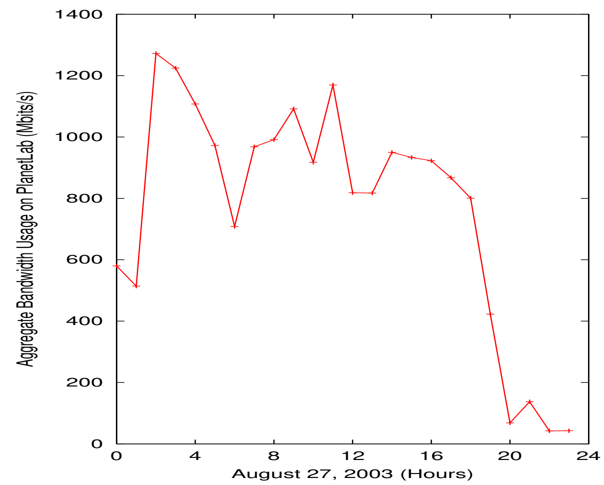


Figure 3: Aggregate send bandwidth on PlanetLab by hour on August 27, 2003 based on the Scout data set.

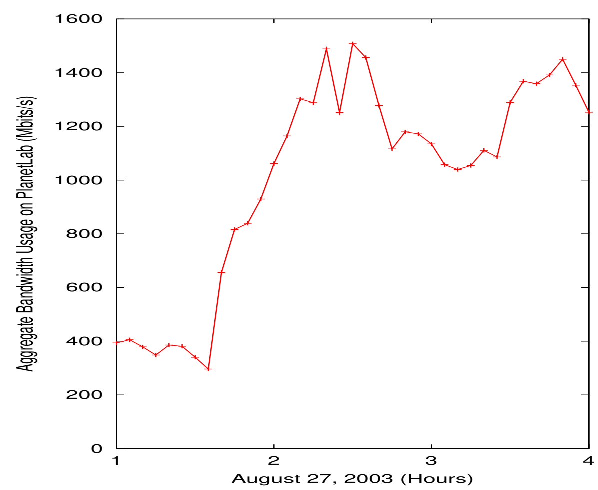


Figure 4: Aggregate send bandwidth on PlanetLab from 1:00am UTC to 4:00am UTC on August 27, 2003 based on the Scout data set.

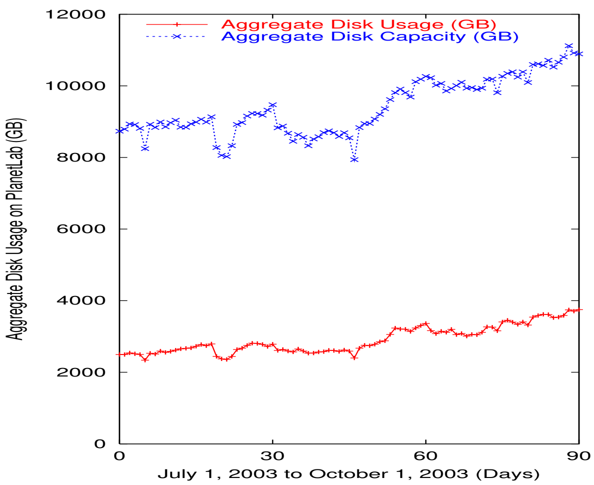


Figure 5: Aggregate disk usage and capacity on PlanetLab based on the Ganglia data set.

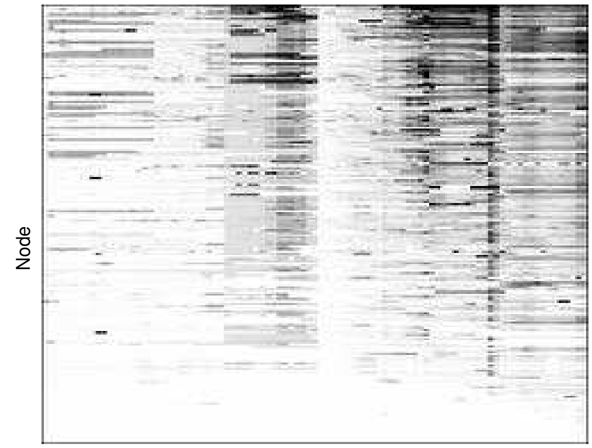


Figure 6: Per-node CPU load on PlanetLab based on the Ganglia data set. CPU load is represented as a grayscale ranging from white (no load) to black (CPU load $\geq$ 10.0).

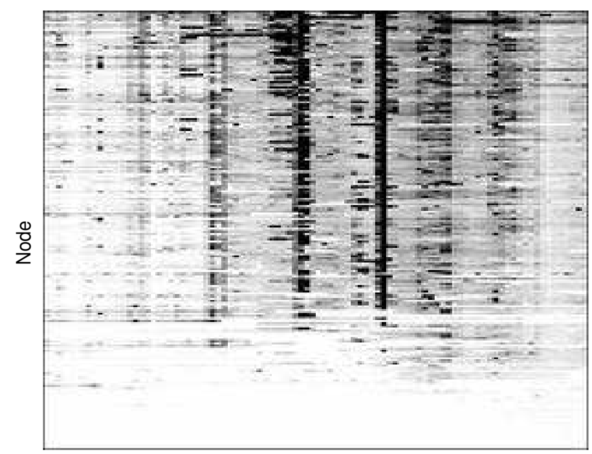


Figure 7: Per-node send bandwidth on PlanetLab based on the Scout data set. Bandwidth is represented as a grayscale ranging from white (0 Mbps) to black ($\geq$ 1 Mbps).

one hour period, before plateauing in the 1.2-1.4 Gbps range.

As expected, disk utilization is much less bursty as shown in Figure 5. Disk capacity grows with the number of nodes in PlanetLab while utilization also grows steadily as users bring additional experiments and data sets online. We expect the gap between disk capacity and disk usage to shrink as time passes.

4.2 Node Resource Contention

Having considered aggregate system resource utilization, Figures 6-8 analyze per-node resource utilization. In each figure, time once again progresses on the x axis over the target 3 month period. The y-axis has bands for each node in PlanetLab sorted by total resource utilization over the entire 3 month period. Thus, the node that is busiest on average for the entire 3 months is presented at the top of graph, while the least busy node is at the bottom. Individual squares within the band represent total resource utilization at a particular node on a given day, with the shade indicating the level of resource utilization. Lighter shades indicate lower levels of resource utilization while darker shades indicate higher levels of resource utilization (see the figure captions for precise quantification). Figure 6 shows fairly light per-node CPU utilization across PlanetLab for the three month period. However, there are periods of time where 30-40 of the nodes show high levels of CPU utilization.

We believe that the relatively light level of PlanetLab CPU utilization can be attributed in part to the types of services and applications currently running on the infrastructure. However, note that during periods of contention (e.g., during the week leading to the NSDI paper submission deadline, Sep 15-22 2003), there was significant CPU contention, with many nodes maintaining 5-minute load averages over 10 for the entire day (many have sustained load averages over 50). On the other hand, PlanetLab services make more heavy and constant use of available bandwidth, as borne out by the level of bandwidth utilization depicted in Figure 7. There are significant periods of heavy network utilization where a large fraction of nodes are averaging more than 1 Mbps of sustained traffic. Since most PlanetLab services currently are attempting to characterize the Internet or improve its communication behavior it stands to reason that network bandwidth is the most constrained system resource.

As expected, Figure 8 shows that disk utilization does not demonstrate significant variation in utilization. Approximately 15-20% of the nodes are at or near capacity during the entire lifetime of our measurement with the rest of the nodes demonstrating increasingly higher levels of utilization over the 3 month period. Note that during this time period, PlanetLab did not enforce any per-user disk quota, validating the notion that without external stimuli disk usage will simply grow unabated.

Having described some of the aggregate and per-node characteristics of PlanetLab, we now turn to geographic characteristics of the system. Table 3 plots the mean and standard deviation for the number of slices hosted per day in each geographic region over the 3 month period. Interestingly, while there are significantly more nodes located in the United States, the average number of slices in different geographical regions of the world is approximately the same. In fact, the top seven regions when measured by average

Region	$\mu_{VMs/day}$	$\sigma_{VMs/day}$
australia	20.4	0.0
netherlands	19.7	0.7
brazil	19.3	0.1
denmark	19.0	0.1
hongkong	18.0	1.4
sweden	17.9	1.1
italy	17.8	0.6
us.central	17.8	5.3
us.eastern	17.7	3.8
germany	16.9	1.9
us.mountain	16.1	7.7
taiwan	16.0	2.4
canada	15.1	5.2
france	15.1	0.0
us.pacific	15.0	5.3
uk	11.5	6.6
china	9.6	0.9
israel	9.0	1.6
russia	7.8	0.0

Table 3: For each region, this table shows the average and standard deviation number of VMs running on each node per day based on the Scout data set.

Region	$\mu_{uniqslices}$	$\sigma_{uniqslices}$
australia	117.0	8.5
denmark	116.5	0.7
sweden	112.5	7.8
italy	106.0	7.1
hongkong	102.5	0.7
germany	101.0	9.6
us.central	98.2	38.6
brazil	98.0	0.0
us.eastern	97.5	31.5
netherlands	97.5	2.1
us.mountain	96.6	60.8
us.pacific	84.3	39.0
taiwan	72.5	13.4
france	71.0	0.0
canada	66.3	45.0
uk	46.2	46.4
china	35.0	1.4
israel	25.5	16.3
russia	15.0	0.0

Table 4: For each region, this table shows the average and standard deviation number of unique slice VMs active on each node during the entire trace period for the Scout data set.

Site	$\mu_{VMs/day}$	$\sigma_{VMs/day}$
duke	23.3	3.7
cmu	22.5	1.3
arizona	22.1	1.4
utexas	21.8	1.0
caltech	21.3	1.0
rice	21.2	1.4
ucla	21.2	0.0
princeton	21.0	3.5
columbia	21.0	1.8
ucb	20.8	2.8

Table 5: Top 10 sites ranked by average number of VMs per node based on the Scout data set.

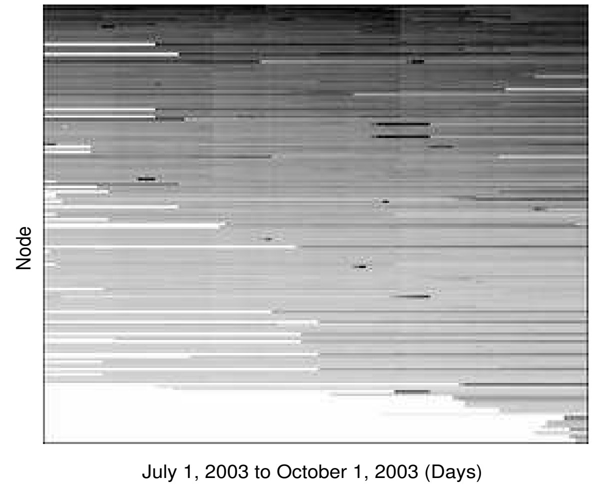


Figure 8: Per-node disk utilization on PlanetLab based on the Ganglia data set. Disk utilization is represented as a grayscale ranging from white (0% utilization) to black (100% utilization).

number of slices are all outside of the United States. This can partially be explained by the fact that the types of services running on PlanetLab desire multiple vantage points on the network, making sites outside of the U.S. more desirable. Further, there are many more nodes currently in the United States than outside it, meaning that system load can be more diffuse if one desires access to U.S. resources compared to non-U.S. resources—if one wishes to run in Australia for example there are only a small number of nodes to choose from. At the same time, Table 5 shows that the 10 most busiest sites, in terms of average number of slices hosted, are all within the United States.

Table 4 plots the mean and standard deviation for the number of unique slices hosted in that geographic region in the 3 month period. Note that certain regions, such as Russia, came up during the course of our measurements, naturally leading to both a smaller number of unique slices and smaller average number of slices per day as system users slowly learn of the existence of a new set of nodes. The Table shows that once again, nodes outside of the U.S. were among the most popular from the perspective of running distributed experiments and services.

4.3 Per-Site Resource Consumption

To this point, we have discussed how PlanetLab resources are used in aggregate across the system. We now turn our attention to who is consuming the resources, with the associated implications on resource allocation scheduling. We use our SliceStat infrastructure (described in Section 3) to measure per-slice resource consumption over a six week period. Tables 6, 7 and 8 depict the amount of CPU, memory, and network send bandwidth consumed in aggregate averaged per-day over a six week period, ranked in order by total per-site resource consumption over the entire period. Table 6 shows that average CPU consumption is fairly modest during this period. SliceStat measures the percentage of CPU utilized by each slice on each node, and hence our CPU utilization numbers are different from the Ganglia profiles we used to monitor overall PlanetLab characteristics earlier, which used 5-minute load average as an indication of CPU load. Table 6 shows that slices originating from site `ast` consume 21 CPUs on average over the entire period with a standard deviation of 11 CPUs, suggesting periods where the site was consuming *all* of the CPU at 30-40 machine equivalents over an entire day. The rest of the top 10 consumed a more modest .2-7 machine equivalents on average.

Site	μ CPU%	σ CPU%
ast	2131.91	1118.82
mit	688.09	1039.01
ucb	614.41	678.42
idsl	609.04	262.02
princeton	402.29	281.62
nyu	364.71	910.48
uiuc	144.61	299.45
duke	77.96	162.44
northwestern	67.30	63.08
cornell	22.22	61.75

Table 6: Top 10 sites ranked by mean aggregate CPU usage over six weeks. For each site, the table shows the mean and standard deviation aggregate CPU usage.

Site	μ GB	σ GB
mit	8.928	4.953
ucb	8.299	5.512
princeton	4.715	1.252
irb	1.389	0.684
northwestern	1.269	1.167
idsl	1.120	0.407
pl	1.087	0.811
uiuc	0.643	1.180
nyu	0.544	1.259
emulab	0.403	0.523

Table 7: Top 10 sites ranked by mean aggregate physical memory usage over six weeks. For each site, the table shows the mean and standard deviation aggregate physical memory usage.

Site	μ Mbps	σ Mbps
mit	23.4	77.5
ucsb	7.0	22.4
ucb	3.5	4.1
rice	3.1	9.0
princeton	1.9	1.3
cmu	1.9	4.4
tennessee	1.2	1.9
nyu	1.2	3.2
uiuc	1.0	3.1
duke	0.7	1.3

Table 8: Top 10 sites ranked by mean aggregate send bandwidth. For each site, the table shows the mean and standard deviation aggregate send bandwidth usage.

As discussed below, the average number of nodes per slice is significantly higher than these values, suggesting that slices typically use a subset of available CPU on a larger number of individual machines. Interestingly, no PlanetLab site contributed more than 3 machines to the infrastructure during the measurement period, clearly indicating that there PlanetLab users do wish to make use of statistical multiplexing to gain access to not just additional vantage points on the network, but also to access more resources than they might contribute.

As depicted in Figure 7, average memory consumption is similar to CPU utilization. The top site, `mit`, consumes an average of 9 GB of global physical memory over the six month period. Each PlanetLab machine has at least 1 GB of physical memory, so this approximately corresponds to MIT users saturating the available physical memory on 9 machines on average across the trace period.

Finally, Table 8 shows that network bandwidth is the most heavily used and most bursty global system resource. For instance, MIT users average 23.4 Mbps per day across the six week period. This corresponds to 253 MB transferred per day on average by MIT users. However, network bandwidth utilization is much more bursty than other resources, with a standard deviation of 77.5 Mb/s for MIT. In fact, each site in the top ten has a standard deviation higher than its mean utilization of network resources. On one day during the trace period, MIT users averaged nearly 700 Mb/s, corresponding to 7.4 TB transferred across the PlanetLab infrastructure on behalf of that site in one day.

One conclusion from these measurements is that individual sites or administrative domains in any shared infrastructure will consume a disproportionately large fraction of global system resources. This suggests the need for a resource allocation infrastructure where each site receives its “fair” (for various definitions of fair) portion of global system resources. At the same time, resource consumption is highly bursty, implying that any static resource allocation mechanism will either leave resources significantly under-utilized or will not allow certain classes of application and services to obtain their full resource needs. It may then be appropriate to leverage statistical multiplexing to guarantee sites and individual users some minimum portion of system resources under constraint. However, in the common case, many users will not require their full allotment, allowing other currently more demanding users to pick up the slack.

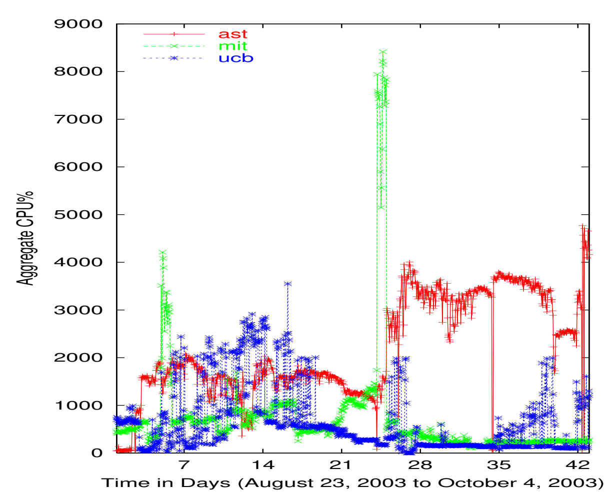


Figure 9: Aggregate CPU usage for the top three sites (ranked by average aggregate CPU usage) based on the SliceStat data set.

Having presented the per-day average resource utilization for the most active sites on PlanetLab, we now focus on variation in behavior over the six-week period. Figures 9, 10 and 11 depict the per-day average resource utilization for the top three sites for CPU, physical memory and send bandwidth respectively. The figures depict significant variation in utilization for all three resources and, once again, in particular for communication bandwidth (note the log scale on the y-axis in Figure 11). Figure 9 shows a significant

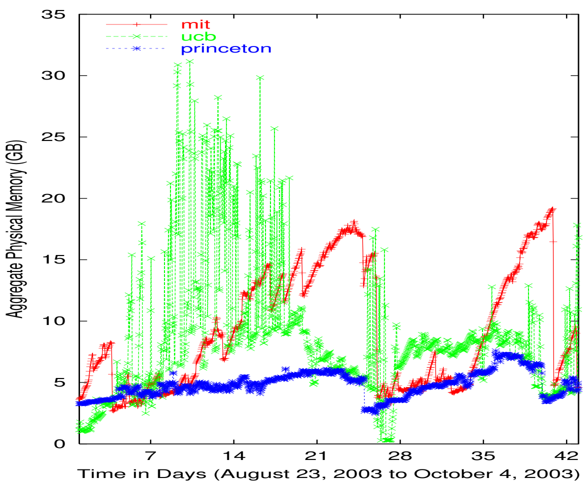


Figure 10: Aggregate physical memory usage (MB) for the top three sites (ranked by average aggregate physical memory usage) based on the SliceStat data set.

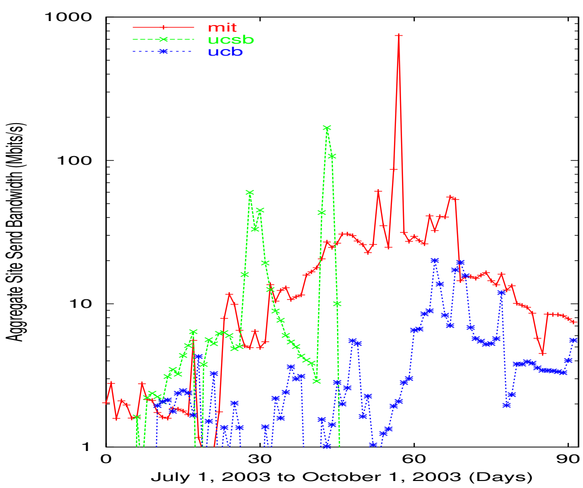


Figure 11: Aggregate site send bandwidth over time for the top three sites (ranked by average site send bandwidth) based on the Scout data set.

spike in CPU utilization, growing from approximately 10 machine equivalents in aggregate to saturating approximately 80 machine equivalents for a 2-day period. Site `ast` moved from averaging 10-20 machine equivalents for the first three weeks of the measurement period to averaging 30-40 machine equivalents for the last three weeks. Interestingly, the NSDI paper submission deadline coincided with the transition to 30-40 aggregate CPUs system wide. This transition suggests that system resources were simply constrained for the first three weeks of the trace and there was actually pent-up demand on the part of `ast`. As soon as the deadline passed, other sites freed up CPU resources, freeing up `ast` to consume additional global resources[3]. Site `ucb` demonstrates yet qualitatively different behavior, rapidly varying between 10-30 machine equivalents within the same day for a two week period.

[3]Recall that each PlanetLab node runs a best effort scheduler to arbitrate local CPU contention.

Figure 10 shows somewhat different behavior for physical memory consumption. The heaviest memory consumer, `mit`, consumed a baseline of at least 5 GB of global physical memory for the entire six week period, with steady climbs to 20 GB twice during the trace period. The second largest consumer, `ucb`, demonstrates highly bursty behavior, coinciding with the bursts in its CPU consumption. While `ucb` consumes just 1 GB of aggregate physical memory at the beginning of the trace, total consumption rises quickly and erratically, bursting to over 30 GB for individual xxx-hour periods over a two week period. At this point, memory consumption returns to a steadier baseline of between 5-10 GB for the last three weeks of the trace period. Finally, the third biggest memory consumer, `princeton` demonstrated very steady memory utilization over the entire 6 week period, hovering just under 5 GB of aggregate utilization with a steady climb to 6 GB over a one week period near the end of the trace.

Finally, Figure 11 shows the highly bursty nature of per-site bandwidth consumption over the entire 3 month period. Site `mit` is the top consumer in aggregate, with its transmission bandwidth increasing steadily from 1 Mbps at the beginning of the period to nearly 1 Gbps two months into the trace. After dropping its average network utilization by nearly two orders of magnitude, it jumps back up by those same two orders of magnitude to 700 Mbps three weeks later. Finally `mit` settles back into the 10 Mbps range for the last two weeks of the trace. The second largest consumer, `ucsb`, demonstrates similarly bursty behavior. Its consumption grows from the 1 Mbps range, peaking first at 100 Mbps, dropping sharply, and rising abruptly back to 200 Mbps before tailing off to zero for the last half of the trace. We were able to determine that this final peak corresponded to a graduate student completing the "last" set of experiments before having his PhD dissertation signed. Finally, the third largest consumer, `ucb`, shows (relatively) the most stable behavior. Its bandwidth varies from 0-5 Mbps for the first two months of the trace before holding more steady in the 3-10 Mbps for the last month of the trace.

4.4 Per-Slice Resource Consumption

The previous subsection discussed resource consumption on a per-site basis. We now delve one level deeper into resource consumption on a per-slice basis. Recall that a slice corresponds to the resource consumption of a single PlanetLab user or service. We begin by measuring the active number of individual slices on the PlanetLab infrastructure and the number of nodes that they are using as a function of time. To gather this information, we track the number of nodes that showed any activity (defined to be transmitting at least one byte of data during the day) on behalf of a slice on a given day. Figure 12 plots these results. It shows that the number of slices that had activity on at least one node grows from approximately 70 to over 160 over the 3 month period. Note that the number of active slices varies widely. In the first two months of the period (from July 1 to Sep 1), the number of slices hovers around 70, varying from 40 up to 90. Starting September 1, the number of active slices begins to grow steadily, perhaps corresponding with the conference submission deadline in mid September. We believe that the number of slices is likely to remain elevated and grow as an increasing number of users become aware of the utility of the infrastructure with each passing deadline. The number of slices that consume a larger number of nodes also grew steadily over the 3 month period, with more than 30 slices running across at least 64 nodes continuously by the end of the trace period.

Figure 13 shows the number of active slices per node, for 7 specific

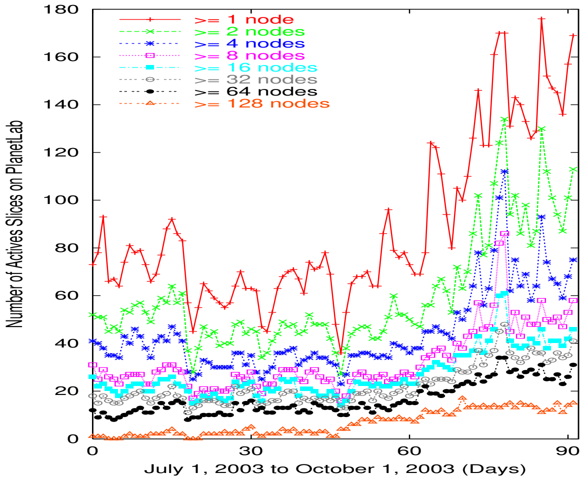


Figure 12: Number of active slices on PlanetLab based on the Scout data set.

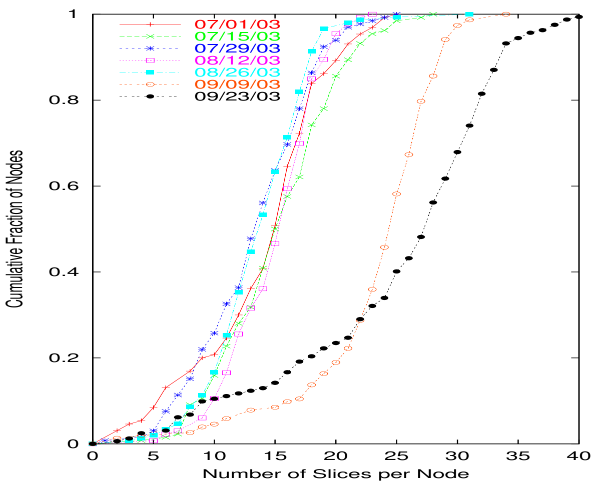


Figure 13: Cumulative distribution of slices per node on PlanetLab based on the Scout data set. Each curve represents the cumulative distribution of slices per node on the first day of each two week interval from July 1, 2003 to October 1, 2003.

dates during the trace period (the dates were chosen to be interspaced by approximately 2 weeks). For the first two months of the trace, 50% of the nodes hosted 12 slices or less on a given day, with 3-15% of the nodes hsoting 20 or more active slices. In the last few weeks of the trace period (corresponding with the conference deadline), 50% of the nodes hosted at least 25 active slices with some popular nodes hosting between 35-40 slices. While not all slices are active simultaneously, if this trend were to continue, the PlanetLab infrastructure would require additional nodes per site to handle the CPU demands (currently there are typically 2-3 nodes per site) and efficient scheduling mechanisms would be required to ensure that nodes do not thrash context switching among a large number of active slices.

Next, we consider the question of how long a slice remains active once it runs. We define a slice to be active if *any* of its constituent virtual machines transmit a threshold amount of data in a given

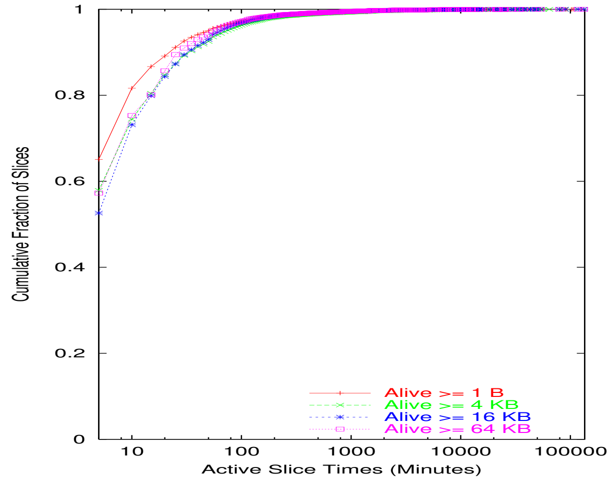


Figure 14: Cumulative distribution of slice lifetimes based on the Scout data set. Each curve plots a slice lifetime CDF for a different definition of slice liveness based on a minimum number of bytes sent or received per day.

time period. The finest time granularity we can consider is 5 minutes given the resolution of our available data sets. Figure 14 plots CDFs of slice lifetime for different data thresholds of activity. Note the logarithmic scale on the x-axis. The figure shows that if we consider a slice to be active if even one of its virtual machines transmits 1 byte of data, then approximately 65% of slices are active for less than 5 minutes or less and 80% are active for 10 minutes or less. Less than 5% of slices are active for more than 100 minutes while a few slices are active for the duration of the trace.

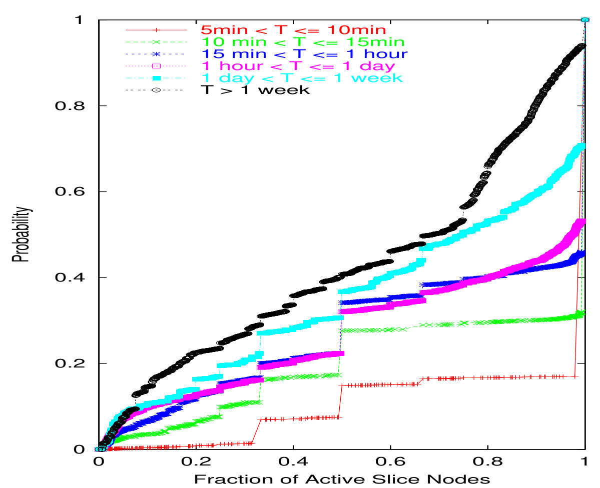


Figure 15: Cumulative distribution of slice activity for varying slice lifetimes across all slices based on the Scout data set.

Next, Figures 15 and 16 plot the fraction of nodes that a slice does use during over its lifetime. First, we determine the maximum number of nodes that are active during a particular lifetime, as measured by the total number of distinct virtual machines that show any network activity (transmitting at least one byte) during the slice's lifetime. Next during each 5 minute interval in the slice's lifetime we consider what fraction of the overall virtual machine set were active. Figure 15 plots this result for slices of different overall dura-

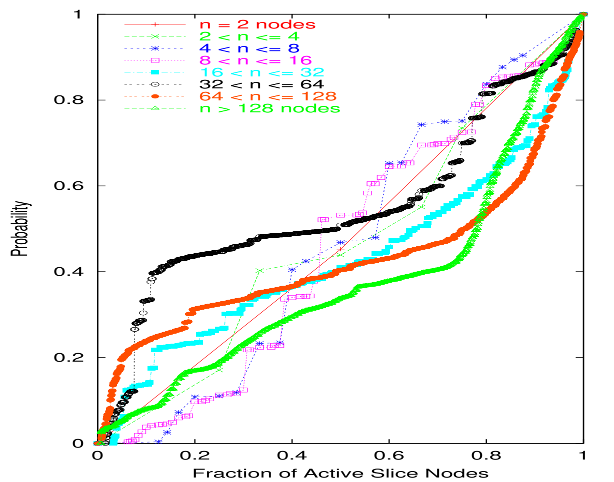


Figure 16: Cumulative distribution of slice activity for varying slice sizes across all slices based on the Scout data set.

tion while Figure 16 does so for slices of different maximum size. Figure 15 shows that short-lived slices tend to run on all of their virtual machines for the duration of their activity. Slices that run for less than 10 minutes run on all of their VMs 80% of the time. The trend is fairly general with most slices running on most of their VMs for the duration of their experiment. Slices that run for more than one week are active on at least 75% of their VMs in 50% of their periods activity. Figure 16 shows that the percentage of active VMs is largely independent of the overall size of the slice. For slices that ran on a maximum of between 32 and 64 VMs during their period of activity, 75% of the VMs were active more than 60% of the time. Interestingly however, less than 10% of the VMs were active 40% of the time, suggesting that a large fraction of a slice's VMs will see significant periods of inactivity.

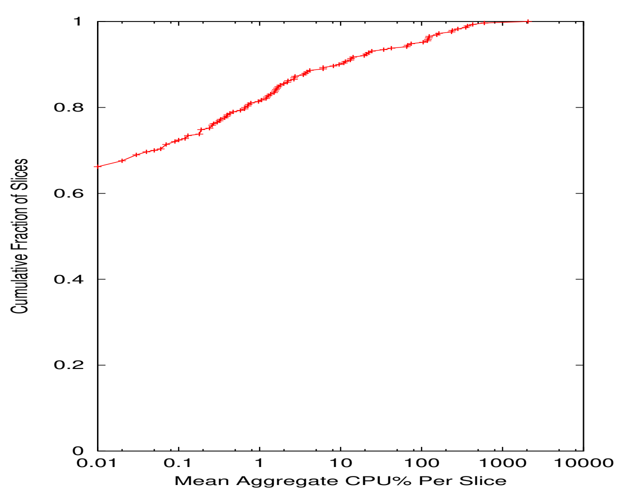


Figure 17: Cumulative distribution of mean, per-day aggregate CPU used by per slice on PlanetLab based on the SliceStat data set.

We now discuss the wide range of resource consumption on a per-slice basis. Figures 17, 18, and 19 depict cumulative distribution functions of per-slice consumption of CPU, physical memory, and network send bandwidth over a six week period. Note the log scale

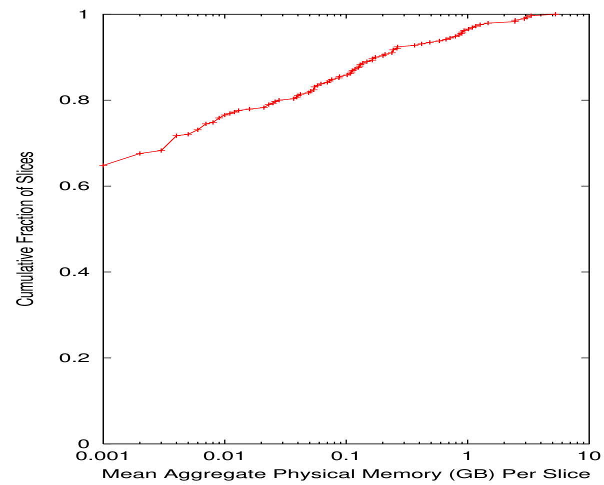


Figure 18: Cumulative distribution of mean, per-day aggregate physical memory in GB used by per slice on PlanetLab base on the SliceStat data set.

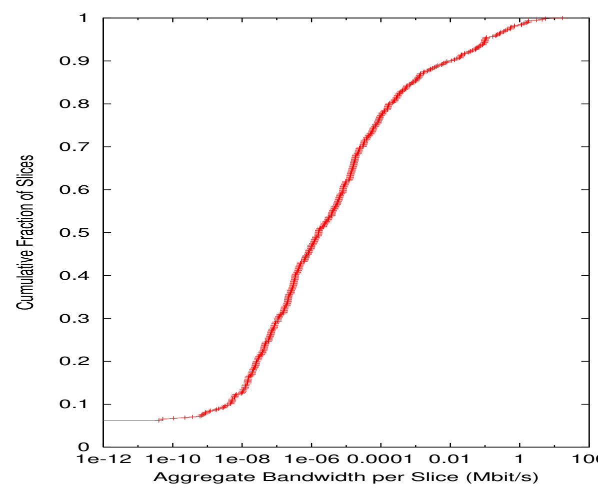


Figure 19: Cumulative distribution of mean, per-day aggregate send bandwidth in Mbps based on the Scout data set.

on the x-axis for all three figures. Figure 17 shows that most slices consume relatively little CPU. In fact, nearly 65% of PlanetLab slices consume less than .01% of aggregate CPU per day. The vast majority of consumed CPU can be attributed to the top 5% of slices. One slice averaged 21 CPU equivalents over the entire 6 week period.

Next, Figure 18 shows that typical physical memory consumption is even more modest. 65% of the slices consume less than 1 MB of physical memory aggregated across the entire infrastructure. While more spread out than CPU utilization, most of aggregate system memory is concentrated in the top 20% of slices, each of which consume more than 30 MB of aggregate memory each. One slice averaged over 5 GB of aggregate memory consumption.

Finally, Figure 19 shows that among the three measured resources, network bandwidth once again shows the widest range in resource consumption. 70% of active slices in the six week period average

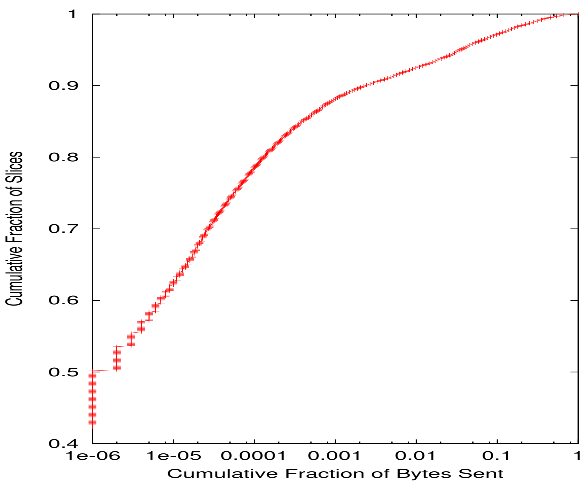


Figure 20: Cumulative distribution of average per-day bandwidth consumption across all slices based on the Scout data set.

less than 1 Kbps of aggregate per-day bandwidth. However, the number of "heavy" users of network bandwidth is larger. Approximately 10% of active slices average more than 100 Kbps of average bandwidth (with much larger bursts). 3% of slices averaged over 1 Mbps of traffic, with one slice averaging approximately 20 Mbps during the six week period. Figure 20 plots a cumulative distribution function for the percentage of slices responsible for transmitting various portions of *all* the bytes sent across PlanetLab for the three month period. The figure shows that the top 10% of slices are responsible for transmitting 99% of the data transmitted across PlanetLab in the three month period. Further, three of the slices are responsible for more than 50% of all transmitted bytes. We expect the set of heavy users to expand as the infrastructure matures, but the concentration of resource utilization among a small set of users is consistent with similar measurements in other distributed settings [1, 2, 4, 15, 19].

4.5 Intra-Slice Communication Patterns

Finally, we consider the communication patterns among the virtual machines that make up the distributed slice. Figure 21 plots the intra-slice communication patterns for eight large-scale services running across PlanetLab in September 2003. Each square is a grid, where each (x, y) point plots the communication intensity from node y to node x over the entire day. The darker the shade, the more communication between the pair of nodes. Grid shade intensity is normalized relative to the heaviest communicating pair of nodes. The axes are sorted by the IP address of the nodes, such that nodes in the same administrative domain are clustered together. The eight communication patterns show a wide range of patterns. Here, we highlight a few. Figure 21(b) shows one service running on 44 nodes, with a maximum of 3.2 MB of data transferred between any pair of nodes. Among the 44 nodes, 5 dark vertical bands correspond to a subset of the nodes receiving data from all slice participants. On the other hand, the two horizontal bands correspond to two nodes sending significant amounts of data to all slice participants. Figure 21(d) shows the communication pattern of a centralized service. In response to user-specified queries to a central location, the central node communicates with all slice participants, collects responses and returns the collected answer to the user. Next, Figure 21(f) plots the communication pattern of a distributed hash table running across 82 nodes. As expected, each node communicates with a subset of global participants (consisting of its routing and leaf table entries). This particular DHT structure attempts to maintain some proximity in its entry corresponding to the cluster of heavy communication around the point $y = -x$ (IP addresses grow down on the y-axes and to the right on the x-axes) resulting from locating nodes in the same site. Finally, Figures 21(a) and (h) plot two variations of slices that demonstrate all-to-all communication.

5. FAILURE CHARACTERISTICS

5.1 Node and Network Failures

Figure 22 shows network availability of PlanetLab nodes over a three month time period based on the AllPairsPing data set. Each curve in the figure plots the CDF of network availability for PlanetLab nodes with a round-trip timeout T. For each value of T, we define the availability of a node as the fraction of average ping times to the node with a round-trip time less than or equal to T. Availability, as defined here, captures three types of failures: (i) local node failures, since a node that is down is unpingable, (ii) network routing failures, and (iii) network failures in terms of *performance* availability [3]. Given PlanetLab's focus on planetary-scale network services, we believe that this definition provides a more useful measure of node availability as compared to focusing strictly on local node crashes. Note that given that our all-pairs ping measurements are taken every 30 minutes, the availability numbers here may not correspond to actual system availability. However, given the long-term nature of these measurements, we believe that the samples represent a relatively unbiased view of system availability.

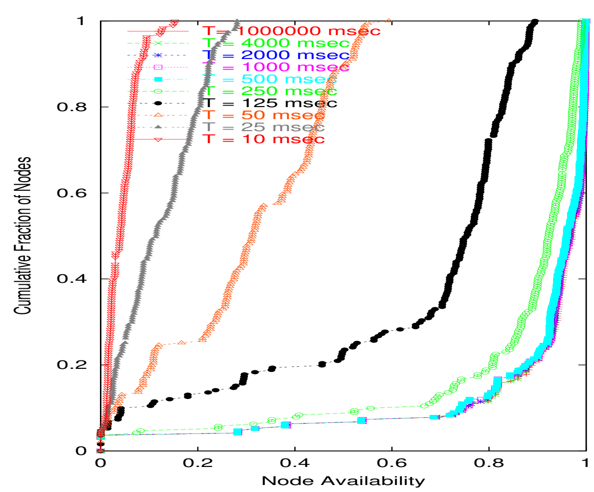
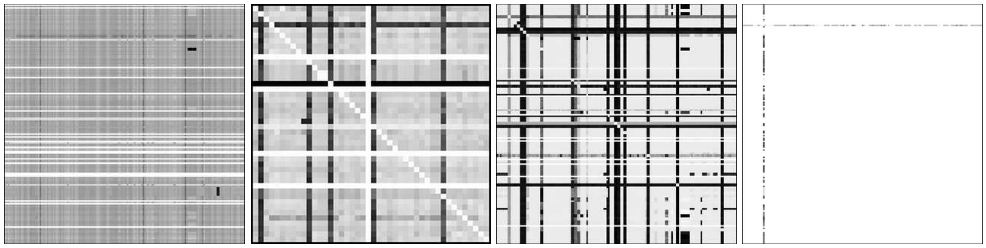


Figure 22: Node availability distribution based on the AllPairsPing data set. For each timeout T, availability of a node is defined as the fraction of average ping times to the node with a round-trip time less than or equal to T.

These results show that, while node availability is high for a significant fraction of nodes, a number of nodes exhibit failures that result in significant total downtime. The rightmost curve ($T = 10^6$), for example, characterizes the extent that PlanetLab nodes are up and reachable via `ping` independent of network performance. Raw data for this curve indicates that 37.7% of all nodes realize greater than 99% availability. However, the curve also shows that approximately 6% of all PlanetLab nodes are unavailable half of the time and that approximately 12% of all nodes display availabilities less than than 80%, when averaged across all nodes in the system.

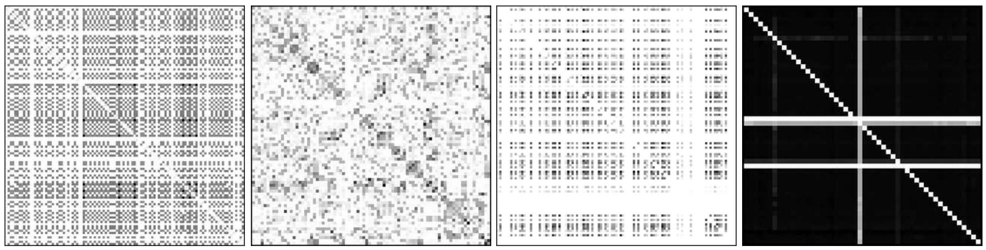


(a) `ast5` (165 nodes, max = 493 KB)

(b) `michigan6` (44 nodes, max = 3.2 MB)

(c) `mit2` (155 nodes, max = 1.1 MB)

(d) `princeton6` (165 nodes, max = 36 KB)



(e) `ucb6` (108 nodes, max = 1.1 MB)

(f) `ucb8` (82 nodes, max = 45 MB)

(g) `uw9` (109 nodes, max = 50 KB)

(h) `vrije8` (50 nodes, max = 298 KB)

Figure 21: Inter-node slice communication patterns based on a day's worth of PLNetflow data before the NSDI deadline in September 2003. The intensity of each point (x,y) represents the amount of data sent from node y to node x normalized relative to the heaviest communicating pair of nodes.

These results also suggest that if a PlanetLab node is up and reachable, it is usually reachable from all other PlanetLab nodes with a round-trip time under 500 msec. This follows from considering performance faults and observing that the availability curves for moderate timeouts (500 msec to 4 secs) virtually overlap the rightmost availability curve. As we decrease timeouts even further, the curves eventually shift to the left as speed of light delays and typical levels of network congestion begin to manifest. With a 50 msec timeout, we see that the highest observed availability is about 60%, which suggests that 40% of PlanetLab is effectively "partitioned" off in a network distance sense.

In Figure 23, we plot CDFs for node mean time to failure (MTTF) and node mean time to repair (MTTR). Here, we break three months of AllPairPings data down into a sequence (approximately 30 minutes apart) of all pairs ping measurements and, for each set of ping measurements, we define each node as being up if one or more ping attempts were successful to the node and down if all ping attempts to the node failed. For each node, we then compute alternating runs of up time and down time and use those runs to compute the MTTF and MTTR for that node. For each down to up transition, we assume the node recovered at the midpoint of the transition. The figure shows the CDF of MTTF and MTTR values across all PlanetLab nodes. The underlying data reveals a median node MTTF of 321.7 hours (about two weeks) and a median node MTTR of about 2.5 hours.

The primary observation is that node MTTF and MTTR vary sub-

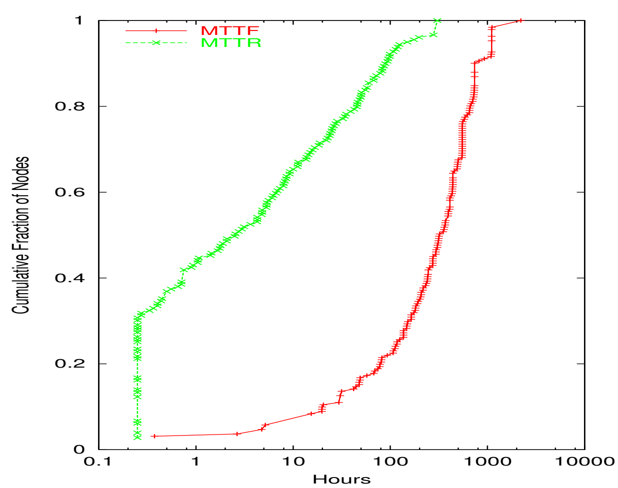


Figure 23: Cumulative distribution of node mean time to failure (MTTF) and mean time to repair (MTTR) in hours based on the AllPairsPing data set.

stantially across PlanetLab nodes and span several orders of magnitude. For MTTF, the minimum MTTF recorded was 22.5 minutes, while the maximum MTTF was the entire three month interval. In the data set, three nodes at distinct sites were online and observed to be up the entire time (i.e., during any hour, at least one ping was successful). Nine other nodes, which came online after the start of the three month interval, were also observed to be up the remainder of the three month time period. For MTTR, the minimum MTTR was 15 minutes, while the maximum MTTR was 12.7 days (if the node came back online at all). The minimum MTTR was likely smaller than 15 minutes (e.g., reboots, transient network outages), but this was not measurable given the granularity of the AllPairsPing data set.

Upon closer examination of the data and discussions with the PlanetLab operations team, we found the primary sources of failures to be software (kernel and device drivers) bugs, hardware failures (e.g., memory parity errors), Internet path outages, system meltdowns under heavy load, reboots due to resource exhaustion (e.g., file descriptors), frequent reboots and downtime as new nodes come into production use, and downtime caused as a result of site-specific administrative issues that needed to be resolved in response to excessive, anomalous, or inappropriate network traffic.

Most of these failure classes have been observed in previous systems. The last category of failures, however, is new and could become increasingly relevant as federated systems grow in popularity and applications use the network in new and interesting ways in the presence of network intrusion detection systems. One interesting observation from PlanetLab's operation is that, in some cases, sites must be taken down for non-technical reasons. For instance, one (remote) PlanetLab service began consuming nearly a third of the bandwidth exiting a site's primary network connection. This particular site payed for bandwidth by the byte. As opposed to CPU, memory, and disk resources, which all in some sense carry a fixed up-front cost, network bandwidth is a renewable resource that can incur a recurring cost depending on the payment structure. It became necessary to rate limit traffic out of that site to ensure that the monetary costs of participating in the PlanetLab testbed did not outweigh the benefits gained locally.

Overall, we found that transient failures occur often and that human response times appear to be the dominating factor with respect to downtime and MTTR. For small MTTRs, we observed 30% of the nodes as having an MTTR of 15 minutes. (Some were probably even lower, but the granularity of our data prevented us from verifying that.) These times suggest transient failures with automatic recovery not involving a human operator (e.g., rebooting a node remotely, short-term network path outages [12] near end hosts, etc.). For larger MTTRs, we see that the range of MTTRs varies over several orders of magnitude with many nodes having an MTTR of multiple hours or even days. Given the types of failures that occur on PlanetLab, this is not surprising given that kernel panics, hardware faults, and resolving network traffic issues with human administrators all ultimately require power cycling a machine (manually in most cases). Moving forward, we believe that automated mechanisms for remotely power cycling machines can substantially improve MTTR, by removing the need to gain the attention of a particular operator at a particular site.

5.2 Correlated Failures

Next, we characterize to what extent node failures are correlated on PlanetLab. For correlation, we use conditional failure probabilities as proposed by Bakkologlu et al. [5] for characterizing correlated failures in survivable storage systems. To quantify the failure correlation between nodes X And Y, we compute the conditional probabilities P(X is down | Y is down) and P(Y is down | X is down). We use conditional probabilities, as opposed to the classic definition of correlation, because failure is typically a rare event. Since most nodes are up the majority of the time, the classic definition of correlation would lead to many node pairs having high correlations, values which would be largely independent of how failures were actually distributed. With conditional failure probabilities, if node failures always overlap, conditional failure probabilities are 1. Similarly, if downtimes never overlap, conditional failure probabilities are 0.

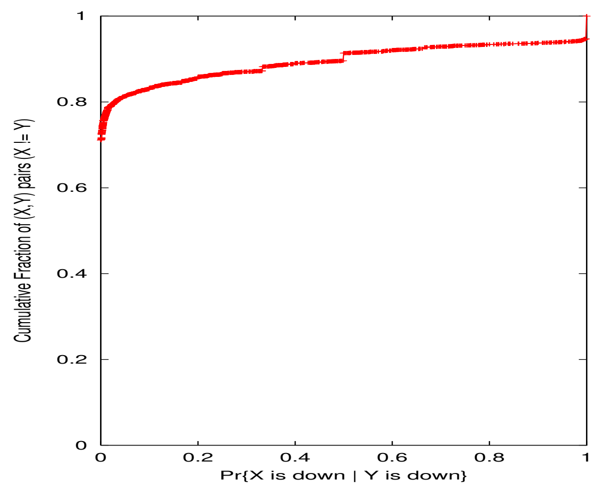


Figure 24: Cumulative distribution of conditional failure probabilities, i.e., P(X is down | Y is down) for all X, Y where X $\neq$ Y.

Figure 24 plots the CDF of conditional failure probabilities for all node pairs (X,Y) where X $\neq$ Y. Failures are based on the same definition used in computing MTTF and MTTR in Section 5.1, except here we consider nodes pairwise at an hour granularity. The data shows that approximately 70% of node pairs have conditional failure probabilities of 0, meaning that 70% of all node pairs were never observed to fail at the same time. On the other hand, we also observe that the remaining 30% of node pairs display non-zero conditional failure probabilities over a wide range of values. Note that a fraction of these correlations are caused by some nodes simply being down for long periods of time. The next visualization, however, shows that concurrent failures are observed on other node pairs besides those that are the chronically down.

Figure 25 depicts correlated failures using a plot where the pixel intensity of point (X,Y) is based on the conditional failure probability P(Y is down | X is down). Nodes are sorted based on IP address, which roughly clusters nodes by sites on either axis. The diagonal band corresponds to P(X is down | X is down), which necessarily is 1 (black). We plot (X,Y) pairs where no data was available between the pair of nodes as white.

The key result from this data is that correlated failures do occur on PlanetLab and that such failures are not entirely caused by sites that are down for extended periods of time. Nodes that are down for long periods of time appear as dark horizontal bands. For example, the dark band in the upper part of the figure corresponds to three

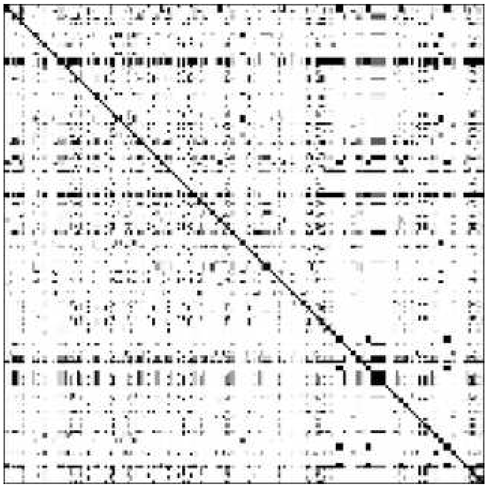


Figure 25: Conditional failure probabilities. Each point (X,Y) shows P(Y is down | X is down) ranging from 0 (white) to 1 (black).

nodes at a site that had to be taken offline for many weeks while issues pertaining to anomalous network traffic were resolved. In general, horizontal bands depict cases where the failure of a set of nodes together predict the failure of a given node. Vertical bands on the other hand indicate the cases where the failure of a single node (on the x axis) predicts the failure of a set of other nodes. While relatively short vertical bands indicate correlation for nodes at the same site going down simultaneously (recall that the axes are sorted according to IP address and that nodes at the same site have similar IP addresses), longer bands cannot be necessarily explained by network proximity.

6. SUMMARY AND IMPLICATIONS

Given our detailed system analysis, we now present a high-level summary of our findings based on our three-month study of system usage of the PlanetLab federated infrastructure.

- Resource utilization is highly bursty, varying by one order of magnitude for CPU and memory (peak to trough) and by two orders of magnitude for network bandwidth.
- Following from the above, there is significant contention for testbed resources in time.
- Demand for system resources is truly global. Users desire access to resources across the world.
- Resource consumption is highly asymmetric on both a per-site and a per-slice basis. Most of global system resources are consumed by a small number of users running resource intensive slices. The top three users of the testbed consumed more than 50% of global network resources over a three-month period.
- The typical testbed node simultaneously hosted 25-30 active slices during busy periods with significant contention for all node resources, including secondary storage.
- Most slices show less than 5 minutes of system activity. Less than 3% of slices are active for more than 2 hours and less than 0.4% for more than 1 day.
- 50% of slices consume less than .01% of CPU, 1 MB of memory, and 1 bit/sec of bandwidth when averaged over an entire day.
- 60% of the nodes have availability above 98% over the 3 month period. However, approximately 10% of the nodes have an availability of under 50%. The median mean time to repair (MTTR) for all nodes is 2 hours. However, 20% of the nodes have an MTTR of 1 day or more. The higher MTTRs typically resulted from needing human attention at a remote site to power cycle a failed node.
- Nearly 75% of all pairs of nodes show no correlation of failure. However, 10% of the node pairs, (X, Y), have a conditional probability of failure of 0.5 or more, i.e., P(X is down | Y is down) $>$ 0.5.

We believe that the above findings have implications for resource discovery, resource allocation, service placement, and construction of highly available services. Given the highly bursty access patterns and the skewed distribution of resource demand among the user population, we believe that some form of resource arbitration (allocating portions of global resources to individual users) will be necessary for the infrastructure. A best effort scheduler will render the system unusable during times of contention for the vast majority of users who have fairly modest resource requirements. At the same time, a strict resource reservation scheme is unwarranted given high levels of resource availability during the times when the system is not constrained. Finally, given system usage patterns, some form of user-initiated resource trading infrastructure may increase overall system utility. That is, during times of contention, all users may fall back to some guaranteed baseline minimum of global system resources. However, it may make sense for users to trade their resource privileges in time to temporarily gain access to more than their baseline share of global system resources.

In our measurements, system users often wish to run on a large number of machines spread across the world. Currently, users manually choose the set of sites that they run on. Moving forward, we believe that an automated resource discovery scheme would greatly simplify access to PlanetLab resources, for instance, automatically recruiting a set of nodes based on user-specified preferences. Given the demand for resources, spread across the world, the testbed would benefit from additional resource availability outside of North America. At the same time, given significant resource contention for nodes outside of North America, any resource discovery scheme must monitor the connectivity and CPU load of nodes spread across the infrastructure to meet user requirements while making best use of aggregate global system resources. Finally, we believe that some portion of resource contention can be attributed to the lag in time from when additional PlanetLab nodes become available to when existing users manually learn of their presence. An automated resource discovery mechanism could certainly mitigate this problem.

Our measurements also indicate that individual testbed nodes host a large number of active virtual machines (up to 40, with 25-30 typical). We further found that individual virtual machines within a slice were often inactive, for instance not transmitting a single byte over a 5-minute interval. This implies that the operating system scheduler must be lightweight and able to quickly swap in and out among active virtual machines. The current PlanetLab design statically creates a virtual machine (with all the associated disk space) on *all* PlanetLab nodes in anticipation of its potential use. Given significant contention for especially disk resources, the system would benefit from a more dynamic model where virtual machines are instantiated on demand or perhaps in response to a resource discovery request.

Our availability measurements indicate that in a heterogeneous, federated testbed there will be significant variation in the uptime and upkeep of individual nodes in the system. This observation has a number of implications. Any resource discovery infrastructure must account for a site's availability history in making allocation decisions. The system may require some incentive for individual sites to maintain a high level of node availability. Further, rather than requiring an a remote administrator to reboot a machine, infrastructure for automated remote power cycling may greatly decrease MTTR for certain sites, at the same time increasing overall node availability at those sites.

Finally, our measurements show a significant and perhaps surprising level of correlation among pairs of node failures, going beyond the expected level of correlated failure one would expect among nodes at a single site. While we have not had the opportunity to explore the causes for such correlated failures, their presence and our ability to measure them have implications for replica placement in the construction of highly available services [23]. Such historical information should be made available to the resource discovery infrastructure as well as possibly the service itself to aid in decisions of node recruitment and data placement to ensure high levels of overall availability and survivability in the face of correlated failures.

Acknowledgments

We thank Paul Brett for making long-term archives of PLNetflow data available and Jeremy Stribling for providing a single tarball of the AllPairsPing data. We are also grateful to Matt Massie for providing scripts to assist in the processing of Ganglia monitoring data. Finally, we thank Robert Adams and Mic Bowman for helpful discussions on Scout and PlanetLab operations in general.

7. REFERENCES

[1] Eytan Adar and Bernardo A. Huberman. Free Riding on Gnutella. Technical report, Xerox PARC, August 2000.

[2] Remzi Arpaci, Andrea Dusseau, Amin Vahdat, Lok Liu, Thomas Anderson, and David Patterson. The Interaction of Parallel and Sequential Workloa d on a Network of Workstations. In *Proceedings of Performance/SIGMETRICS*, May 1995.

[3] Remzi H. Arpaci-Dusseau. *Performance Availability for Networks of Workstations*. PhD thesis, University of California, Berkeley, 1999.

[4] Mary Baker, John Hartman, Mike Kupfer, Ken Shirriff, and John Ousterhout. Measurements of a Distributed File System. In *Proceedings of the 13th ACM Symposium of Operating Systems Principles*, October 1991.

[5] Mehmet Bakkaloglu, Jay J. Wylie, Chenxi Wang, and Gregory R. Ganger. On correlated failures in survivable storage systems. Technical Report CMU-CS-02-129, Carnegie Mellon University, School of Computer Science, May 2002.

[6] Gaurav Banga, Peter Druschel, and Jeffrey C. Mogu l. Resource Containers: A New Facility for Resource M anagement in Server Systems. In *Third Symposium on Operating Systems Design and Imp lementation*, February 1999.

[7] Andy Bavier, Thiemo Voigt, Mike Wawrzoniak and̃ Larry Peterson, and Per Gunningberg. SILK: Scout Paths in the Linux Kernel. Technical Report 2002-009, Department of Information Technology, Uppsala University, Uppsala, Sweden, February 2002.

[8] Micah Beck, Terry Moore, and James Plank. An end-to-end approach to globally scalable network storage. In *Proceedings of the ACM SIGCOMM '02 Conference on Communications Architectures and Protocols*, August 2002.

[9] Brent Chun, Yun Fu, and Amin Vahdat. Bootstrapping a Distributed Computational Economy with Peer-to-Peer Bartering. In *Proceedings of 1st Workshop on Economics of Peer-to-Peer Systems*, June 2003.

[10] Amol Deshpande, Suman Nath, Phillip B. Gibbons, and Srinivasan Seshan. Cache-and-query for wide area sensor databases. In *Proceedings of the 2003 ACM SIGMOD Conference*, June 2003.

[11] Boris Dragovic, Keir Fraser, Steve Hand, Tim Harris, Alex Ho, Ian Pratt, Andrew Warfield, Paul Barham, and Rolf Neugebauer. Xen and the Art of Virtualization. In *Proceedings of the ACM Symposium on Operating Systems Principles*, October 2003.

[12] Nick Feamster, David Andersen, Hari Balakrishnan, and Frans Kaashoek. Measuring the effects of internet path faults on reactive routing. In *Proceedings of the 2003 ACM SIGMETRICS Conference*, June 2003.

[13] Ian Foster, Carl Kesselman, and Steven Tuecke. The anatomy of the grid. *International Journal of Supercomputing Applications*, 15(3), 2001.

[14] Yun Fu, Jeffrey Chase, Brent Chun, Stephen Schwab, and Amin Vahdat. SHARP: An Architecture for Secure Resource Peering. In *Proceedings of the 19th ACM Symposium on Operating System Principles*, October 2003.

[15] Krishna P. Gummadi, Richard J. Dunn, Stefan Saroiu, Steven D. Gribble, Henry M. Levy, and John Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles*, October 2003.

[16] Ryan Huebsch, Joseph M. Hellerstein, Nick Lanham, Boon Thau Loo, Scott Shenker, and Ion Stoica. Querying the internet with pier. In *Proceedings of the 29th International Conference on Very Large Data Bases*, September 2003.

[17] Vivek Pai, Limin Wang, KyoungSoo Park, Ruoming Pang, and Larry Peterson. The dark side of the web: An open proxy's view. In *Proceedings of HotNets–II*, November 2003.

[18] Larry Peterson, Tom Anderson, David Culler, and Timothy Roscoe. A blueprint for introducing disruptive technology into the internet. In *Proceedings of HotNets–I*, October 2002.

[19] Stefan Saroiu, P. Krishna Gummadi, and Steven D. Gribble. A Measurement Study of Peer-to-Peer File Sharing S ystems. In *Proceedings of Multimedia Computing and Networking 2002 (MMCN'02)*, January 2002.

[20] Neil Spring, David Wetherall, and Tom Anderson. Scriptroute: A public internet measurement facility. In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems*, March 2003.

[21] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM SIGCOMM '01 Conference on Communications Architectures and Protocols*, September 2001.

[22] Mike Wawrzoniak, Larry Peterson, and Timothy Roscoe. Sophia: An information plane for networked systems. In *Proceedings of HotNets–II*, November 2003.

[23] Hakim Weatherspoon, Tal Moscovitz, and John Kubiatowicz. Introspective Failure Analysis: Avoiding Correlated Failures in Peer-to-Peer Systems. In *Proceedings of International Workshop on Reliable Peer-to-Peer Distributed Systems*, October 2002.

[24] Andrew Whitaker, Marianne Shaw, and Steven D. Gribble. Denali: Lightweight Virtual Machines for Distributed and Networked Applications. In *Proceedings of Operating Systems Design and Implementation*, December 2002.

[25] Brian White, Jay Lepreau, Leigh Stoller, Robert Ricci, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Barb, and Abhijeet Joglekar. An integrated experimental environment for distributed systems and networks. In *Proceedings of the 5th USENIX Symposium on Operating Systems Design and Implementation*, pages 255–270, December 2002.