NORTHWESTERN
UNIVERSITY

Electrical Engineering and Computer Science Department

Technical Report
Number: NU-EECS-13-09

September, 2013

Dasu: A measurement experimentation platform at the Internet's edge

M. Sánchez*, J. Otto*, Z. Bischof*, D. Choffnes^, F. Bustamante*,
B. Krishnamurthy, W. Willinger+.
**Northwestern U, ^U of Washington, +AT&T Labs-Research*

Abstract

Poor visibility into the network hampers progress in a number of important research areas, from network troubleshooting to Internet topology and performance mapping. This persistent well-known problem has served as motivation for numerous proposals to build or extend existing platforms by recruiting larger, more diverse vantage points. However, capturing the edge of the network remains an elusive goal. We argue that at its root the problem is one of incentives. Today's measurement platforms build on the assumption that the goals of experimenters and those hosting the platform are the same. As much of the Internet growth occurs in residential broadband networks, this assumption no longer holds. We present Dasu, a measurement experimentation platform built on an alternate model that explicitly aligns the objectives of the experimenters with those of the users hosting the platform. Dasu is designed to support both network measurement experimentation and broadband characterization. In this paper, we discuss some of the challenges we faced building a platform for the Internet's edge, describe our current design and implementation, and illustrate the unique perspective our current deployment brings to Internet measurement. Dasu has been publicly available since July 2010 and is currently in use by over 95,000 users with a heterogeneous set of connections spreading across 1,802 networks and 151 countries.

Dasu: A measurement experimentation platform at the Internet's edge

Mario A. Sánchez† John S. Otto† Zachary S. Bischof† David R. Choffnes‡
Fabián E. Bustamante† Balachander Krishnamurthy* Walter Willinger*
†Northwestern University ‡University of Washington *AT&T Labs-Research

***Abstract*—Poor visibility into the network hampers progress in a number of important research areas, from network troubleshooting to Internet topology and performance mapping. This persistent well-known problem has served as motivation for numerous proposals to build or extend existing platforms by recruiting larger, more diverse vantage points. However, capturing the edge of the network remains an elusive goal.**

We argue that at its root the problem is one of incentives. Today's measurement platforms build on the assumption that the goals of experimenters and those hosting the platform are the same. As much of the Internet growth occurs in residential broadband networks, this assumption no longer holds.

We present Dasu, a measurement experimentation platform built on an alternate model that explicitly aligns the objectives of the experimenters with those of the users hosting the platform. Dasu is designed to support both network measurement experimentation and broadband characterization. In this paper, we discuss some of the challenges we faced building a platform for the Internet's edge, describe our current design and implementation, and illustrate the unique perspective our current deployment brings to Internet measurement. Dasu has been publicly available since July 2010 and is currently in use by over 95,000 users with a heterogeneous set of connections spreading across 1,802 networks and 151 countries.

I. Introduction

Our poor visibility into the network hampers progress in a number of important research areas, from network troubleshooting to Internet topology and performance mapping. This well-known problem [1], [2] has served as motivation for several efforts to build new testbeds or expand existing ones by recruiting increasingly large and diverse sets of measurement vantage points [3]–[6]. Today's measurement and experimentation platforms offer two basic incentive models for adoption – cooperative and altruistic. In cooperative platforms such as PlanetLab [7] and RIPE Atlas [8] an experimenter interested in using the system must first become part of it. Other platforms such as SatelliteLab [3] and DIMES [5] have opted instead for an altruistic approach in which users join the platform for the betterment of science. Despite all these efforts, however, capturing the diversity of the commercial Internet (including, for instance, end-hosts in homes and small businesses) at sufficient scale remains an elusive goal [9], [10].

This paper presents Dasu — a software-based measurement platform that is hosted by voluntary nodes at the edge of the network and relies on a direct incentive model to ensure large-scale adoption. Compared to existing alternatives, our platform is the first to support safe, extensible and globally coordinated measurements from hosts located in edge networks while providing an alternate incentive model that explicitly aligns the objectives of the experimenters with those of the users hosting it. Dasu[1] is designed to support both broadband characterization and Internet measurement experiments and leverage their synergies. Both functionalities benefit from wide network coverage to capture network and broadband service diversity. Both can leverage continuous availability to capture time-varying changes in broadband service levels and to enable long-running and time-dependent measurement experiments. Both must support dynamic extensibility to remain effective in the face of ISP policy changes and to enable purposefully-designed, controlled Internet experiments. Finally, both functionalities must be available at the edge of the network to capture the end users' view of the provided services and offer visibility into this missing part of the Internet [11].

Using our prototype we show that today's home networks are a feasible environment to host vantage points and that a programmable, edge-based platform enables complex, coordinated measurements across participating hosts. Dasu has been publicly available since June 2010 and is currently in use by 95,222 users with a heterogeneous set of connections spreading over 1,802 networks and across 151 countries.

The remainder of this paper is structured as follows. We put our work in context and provide further motivation in Section II. In Section III we examine the complexity of home networks and show that hosting a platform for experimentation at the network edge is an attainable goal. Then, in Sections IV and V we present the design and implementation of Dasu, including a description and evaluation of its current deployment. Section VI presents four different case studies that illustrate the unique perspective offered by a platform such as Dasu and serve as examples of experiments made possible from it. Finally, we discuss future work and present our conclusions in Section VII.

II. A case for experimentation at the Internet's edge

The lack of network and geographic diversity in current Internet experimentation platforms has been long recognized [1], [2]. Most Internet measurement and system evaluation studies rely on dedicated infrastructure [7], [12], [13] which include nodes primarily located in well-provisioned academic

[1]Dasu is a Japanese word that means "to reveal" or "to expose".

or research networks and are not representative of the larger Internet — with end-hosts in homes, small businesses and Internet cafes, connected over DSL, dial-up and cable.

Several research projects have pointed out the implications this "lack of representativeness" has on efforts to generalize the results of network measurements and have cast doubts on the conclusions drawn from evaluations of networked systems (e.g. [2], [14]–[17]). A comparative analysis of the paths between PlanetLab nodes and between nodes in residential networks illustrates some of these issues. These two sets of paths have been shown to traverse different parts of the network [11], exhibit different latency and packet loss characteristics [18], [19] and result in different network protocol behaviors [20].

A. Goals and Approach

An experimental platform for the Internet edge should be deployed at scale to capture network and service diversity. It should be hosted at the network edge, to provide visibility into this missing part of the Internet. To support time-dependent and long-running experiments, such a platform should also offer (nearly) continuous availability. Last, it should facilitate the design and deployment of experiments at the network edge while controlling the impact on the resources of participating nodes and the underlying network resources.

Dasu is an experimental platform designed to match these goals. To capture the diversity of the commercial Internet, Dasu supports both Internet measurement experimentation and broadband characterization, and leverages their synergies. In its current version, Dasu is built as an extension to the most popular large-scale peer-to-peer system – BitTorrent.[2] The typical usage patterns and comparatively long session times of BitTorrent users means that Dasu can attain nearly continuous availability to launch measurement experiments. More importantly, by leveraging BitTorrent's popularity, Dasu attains the necessary scale and coverage at the edge of the network. Dasu is tailored for Internet network experimentation and, unlike general-purpose Internet testbeds such as PlanetLab, does not support the deployment of planetary-scale network services.

B. Challenges

Both strengths and challenges of a platform like Dasu stem from its inclusion of participating nodes at the Internet's edge. For one, the increased network coverage from these hosts comes at a cost of higher volatility and leaves the platform at the "mercy" of end user behavior. The types of experiments possible in such a platform depend thus on the clients' availability and session times since these partially determine the maximum length of the experiment that can be safely assigned to clients. Such a platform must provide a scalable way to share measurement resources among concurrent experiments with a dynamic set of vantage points. It must also guarantee the safety of the volunteer nodes where it is hosted (for instance, by restricting the execution environment), and ensure secure communication with infrastructure servers. Last, to control the impact that experiments may have on underlying network and system resources, the system must support coordinated measurements among large numbers of hosts worldwide, each of which is subject to user interaction and interference.

[2]A stand-alone version of Dasu has been developed and was released in June 2013.

C. Related work

Our work shares goals with and builds upon ideas from several prior large-scale platforms targeting Internet experimentation. Most active measurement and experimentation research relies on dedicated infrastructures (PlanetLab [7], Ark [12], Looking Glass servers). Such infrastructures provide relatively continuous availability and nearly continuous monitoring at the cost of limited vantage point diversity. Dasu targets the increasingly "invisible" portions of the Internet, relying on a direct incentive model to ensure large-scale adoption at the Internet edge.

Several related projects use passive measurements or restricted active measurements from volunteer platforms to capture this same perspective (e.g., [3]–[6], [8], [21], [22]). In contrast, Dasu is a software-based solution with a much broader set of measurement vantage points that has been achieved by altruistic and hardware-based systems, and supports a programmable interface that enables complex, coordinated measurements across the participating hosts. As such, Dasu shares some design goals with Scriptroute [23] and SatelliteLab [3]. Unlike Scriptroute, Dasu is intended for large scale deployment on end users' machines, and relies on incentives for user adoption at scale. Dasu also enables programable measurements without requiring root access, avoiding potential security risks and barriers to adoption. SatelliteLab adopts an interesting two-tier architecture that links end hosts (satellites) to PlanetLab nodes and separates traffic forwarding (done by satellites) from code execution. In Dasu, experiment code generates traffic directly from hosts at the network edge.

Several systems have proposed leveraging clients in a P2P system to measure, diagnose and predict the performance of end-to-end paths (e.g., [24], [25]. Dasu moves beyond these efforts, exploring the challenges and opportunities in supporting programmable experimentation from volunteer end hosts.

III. The Home Network – A complex Environment

In contrast with the well-understood and -provisioned academic and research networks that host dedicated infrastructure platforms such as PlanetLab, home network environments and broadband access links vary widely and present several unique challenges to Dasu's measurement approach (e.g. the presence of cross-traffic from other devices). In this section we characterize home networks and explore the dynamics of home device usage both at the macro – the frequency with which devices in the home network are active – and micro – the rate and volume of traffic generated by these device – levels. We then demonstrate how Dasu addresses some of the unique challenges presented by this environment.

A. Data Collection and Dataset

We conduct our analysis using data collected with Dasu. We use a combination of passive and limited active measurements gathered over a 6-month period between February 24, 2012 and August 23, 2012. This dataset includes traces of BitTorrent and overall home network activity collected by Dasu from 13,605 homes spanning 151 countries.[3]

Each Dasu client periodically (at 30s intervals) collected anonymized traffic traces from BitTorrent's activity, including the number of bytes uploaded and downloaded as well as the current transfer speed. Traffic on the host's network interface (i.e. the total number of bytes sent/received) was also captured using `netstat`. Beyond this passively collected data, clients also scanned the local network in search of Internet gateway devices using UPnP, following an approach based on DiCioccio et al. [26], [27].

For each gateway device responding to UPnP discovery messages, Dasu pulled their device definition XML data and collected the following configuration parameters: (a) current state of NAT for this connection, (b) external IP address, (c) current connection type (Cable, DSL), (d) maximum upstream/downstream bit rate available, (e) device model name and version. At the same rate, clients also retrieved dynamic information from the gateway including (f) cumulative count of bytes and packets received and (g) sent, as well as (h) the connection status.

Additionally, a subset of clients periodically broadcasted UPnP discovery messages and recorded, for each responding device: (a) devices' UUID and UDN, (b) device type, (c) manufacturer, (d) model name and (e) model number.

B. Exploring the complexity of home networks

As a measure of home network complexity, we count the number of networked devices found for a subset of ≈4.6K of our client's home networks using UPnP discovery messages.

Figure 1 shows the histogram of home networks by the number of UPnP announced devices found. While 34.5% of sampled home networks have no UPnP devices announcing their presence, over 65% of them have at least 1 device, and over 16% have 3 or more devices. Although this approach is prone to both undercounting (missing devices without UPnP) and over-counting (devices with multiple UPnP services), we plan to refine this approach to address these issues as part of our future work. Regardless, these results indicate that, for nearly two-thirds of locations, we must consider the impact of other devices on the network.

a) Device diversity: To study the diversity of home network devices we classify the found UPnP-enabled devices and study their prevalence. For common devices we use the DLNA's "Home Network Device" specification[4] to categorize them and divide the rest into functional classes such as *storage*, *cameras* or *television*. We labeled each device class as *Internal* and *External* based on their dominant network role – *externally-facing* devices that exchange traffic with the outside

[3]The dataset is available to other researchers upon request.

[4]http://dlna.org/dlna-for-industry/digital-living/how-it-works/dlna-device-classes

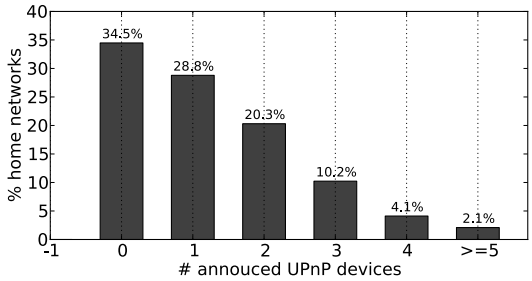


Fig. 1: UPnP-enabled devices in home networks.

Device Type	Connection	Perc.
Gateway	Gateway	36.7%
Digital Media Player (DMP)	Internal	34.7%
Digital Media Server (DMS)	Internal	10.2%
Digital Media Renderer (DMR)	Internal	9.5%
Digital Media Printer (DMPr)	Internal	1.2%
Digital Media Controller (DMC)	Internal	1.2%
Digital Organizer	Internal	1.3%
Storage	Internal	1.1%
Game Console	External	0.5%
TV	External	0.3%
Camera	External	0.2%
SetupBox	External	0.1%
House Automation	External	$<$ 0.1%
Other	External	1.5%

TABLE I: Different classes of UPnP-enabled devices and their prevalence.

world (e.g. TV) or *internally-facing* devices that exchange traffic mostly within the home network (e.g. Storage). Given that the purpose of DLNA devices is to share media within the home (e.g. Digital Organizers, and Storage), each of these device classes are labeled as *Internal*. We classify the remaining classes of devices as external, including *Others*. We treat the *Gateway* category (e.g. DSL modems, WiFi routers) as its own class.

Table I shows the different device classes identified in our traces. From the ≈6K devices seen across ≈3K peers the most popular device type are gateways (over 35%) followed by a large number of DLNA-compliant devices, including Digital Media Players (34.7%), Digital Media Servers (10.2%) and Digital Media Renderers (9.5%).

In the context of ISP characterization and Internet measurements, we are particularly interested in the distribution of *internally-* and *externally-facing* devices. Figure 2 shows the fraction of home networks within each group for which at least one externally-facing device was identified. Not surprisingly, as the number of announced devices in the network increases, so does the probability that at least one of those devices is an external device, hence increasing the likelihood that the access link will be used by multiple devices simultaneously.

b) Device dynamics: To study device dynamics, we leverage the fact that Dasu runs for long periods at a time (the median session time is about 3 hours long) and is thus able to take multiple snapshots of the active UPnP-enabled devices present on the network over time. We focus on the set of home networks for which we have at least 10 different sample snapshots and where there is more than one UPnP-

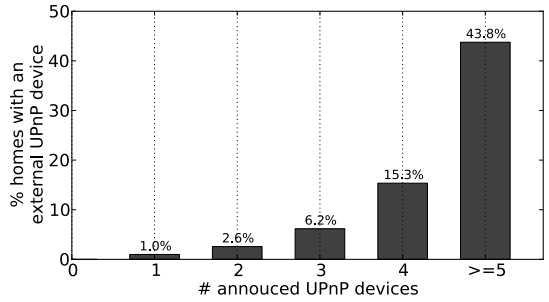


Fig. 2: Percentage of homes with external devices based on number of UPnP-announced devices.

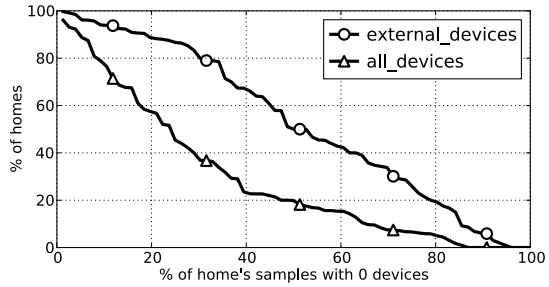


Fig. 3: Distribution of the fraction of homes vs the fraction of samples for which no other UPnP-device is present in the network.

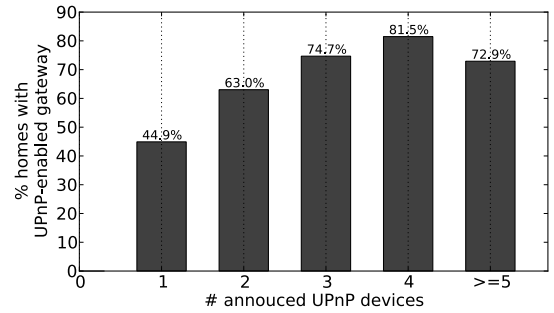


Fig. 4: Prevalence of UPnP-enabled gateways in sampled homes, clustered based on number of UPnP-enabled devices announced.

enabled device announced and at least one of those is *outer-facing*. This set consists of 502 different home networks.

We rank all locations based on the percentage of measurement samples where we find no other device/no external device active other than the host machine. Figure 3 plots the CDF for both – any device active (labeled *all_devices*) and external device active (labeled *external_device*). As the figure shows, for nearly 85% of the locations, the host computer where our measurement client is running is the only active external device in the network for at least 10% of measurement samples. For the median location, about 20% of the measurement samples occur when the host computer is the only active device in the network and nearly 50% of them when there is no other external device present.

C. Detecting Cross-traffic

Two sources of concern for network experimentation from end systems are the presence of cross-traffic from other applications in the hosting devices and from other devices in the home network. Dasu uses `netstat`, a network statistics tool available in most platforms, to capture the number of bytes sent and received from the host and compares it against the amount of traffic monitored by our client. This allows us to identify situations where significant amount of traffic is being generated by other applications in the host device.

The second type of cross traffic is the one generated by other devices in the network. To identify such cases Dasu employs the technique described by DiCioccio et al. [28] where UPnP-enabled home gateways are periodically queried to measure traffic in the home network. In cases where the UPnP-supplied data is both available and accurate, the authors showed that this technique provides a rich source of information for inferring the presence of cross traffic in the home network. Thus, for homes with UPnP-enabled gateways, Dasu periodically queries for traffic counters across its WAN interface (the number of bytes and packets sent and received). When Dasu identifies times where the number of packets or bytes sent or received is high enough to affect our measurements it simply discards (if passive) or postpone (if active) its measurements. While gateway UPnP traffic counters are not always accurate [28], such instances can be easily identified and accounted for.

c) Prevalence of UPnP-enabled gateways: UPnP-enabled gateways are helpful for managing resources and monitoring the state of the network. Although UPnP-enabled gateways are not always available, their presence is particularly important in home networks with high number of devices, where cross-traffic could interfere with characterization and measurements.

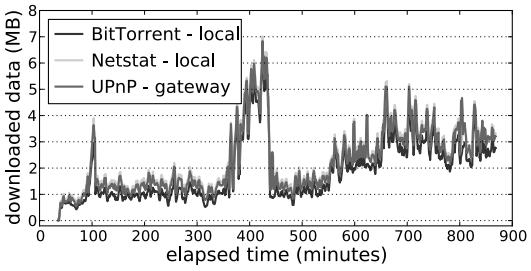
Figure 4 shows the availability of UPnP-enabled home gateways in our sample. The figure plots the fraction of homes, with a given number of UPnP devices, in which such a gateway is present. As the number of UPnP-enabled devices in the local network increases, so does the likelihood that the home gateway supports UPnP.

D. The Value of UPnP-Counters

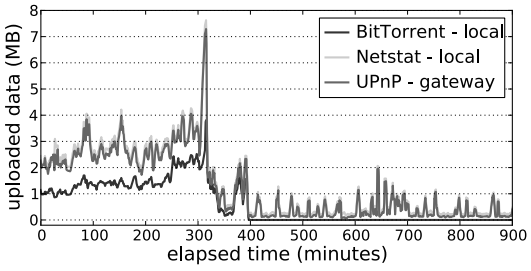
We now present some concrete examples of how traffic counters from UPnP-enabled gateways allow Dasu to disambiguate between different scenarios inside the home network. Using data collected from our Dasu users we show, for instance, how the presence of internal traffic can be identified and separated from traffic that uses the access link, both from the local host and other devices within the network.

As mentioned before, our traces contain the network activity as seen by each individual Dasu client at three different granularities: (i) Because Dasu runs as part of a network intensive application (BitTorrent) our traces contain traffic statistics generated by the application itself. (ii) By using `netstat`, our traces also contain overall traffic activity of the host, including traffic generated by all running applications. Finally (iii) the client collects UPnP-supplied data from the gateway including traffic statistics across the gateway's WAN interface.

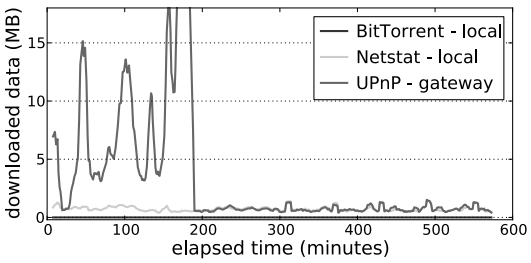
d) No cross-traffic.: Figure 5a shows the simplest scenario – where BitTorrent is solely responsible for the network



(a) No cross-traffic.



(b) Local cross-traffic (up).



(c) Cross-traffic (down).

Fig. 5: Traffic scenarios within the home network: (5a) download with no cross-traffic, (5b) local cross-traffic from other applications and (5c) download cross-traffic.

traffic using the access link and the only source of traffic generated by the host. The figure plots the download activity of one Dasu client in a span of 15 hours in August 2012. Each of the three signals in the graph represents the number of downloaded bytes as reported by BitTorrent (blue), `netstat` (black), and the gateway counters (red), respectively, in intervals of 30 seconds increment. As the figure shows, all three signals overlap when Dasu' hosting application (BitTorrent) is the only network active application.

e) Local cross-traffic from other applications.: Figure 5b plots the upload activity of another client, also for a span of 15 hours in June 2012. As before, the client is solely responsible for all the traffic present in the access link, but here BitTorrent is not the only network active application. As the figure shows, the curves that correspond to the local `netstat` counters (black) and the UPnP-counters at the gateway (red) overlap through the entire collection period (i.e., the client is the only device using the access link), but the curve that corresponds to BitTorrent traffic (blue) is much lower than that of `netstat` for the first five hours (300 minutes) of the session.

f) Cross-traffic from other devices.: Figure 5c shows our last scenario, where there is significant cross-traffic from other devices in the home network. The figure plots download activity seen from a client over a span of five hours. In this case, there's no BitTorrent content being downloaded (the BitTorrent signal is a flat horizontal line around 0 bytes), but there is local traffic being generated by other applications in the host device (denoted by the black signal). However, for the first $\approx$ 200 minutes of the session, the traffic generated by the host devices represents only a small fraction of the total traffic present in the access link (red signal). The figure also shows the easily identifiable point at which the cross-traffic disappears.

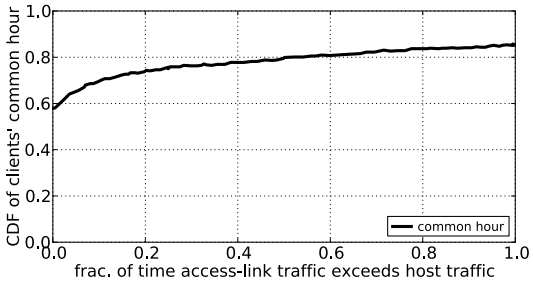


Fig. 6: Fraction of time the host machine shares the access-link with other devices in the network.

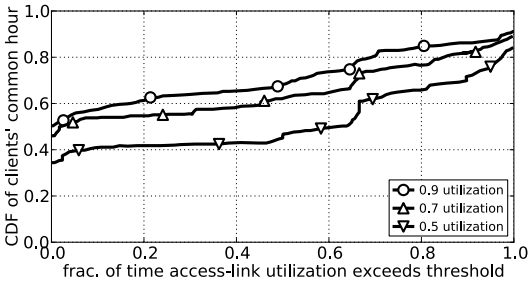
E. Access-link Utilization

Finally we measure the fraction of time users are alone in the network and quantify the utilization of home networks' access link.

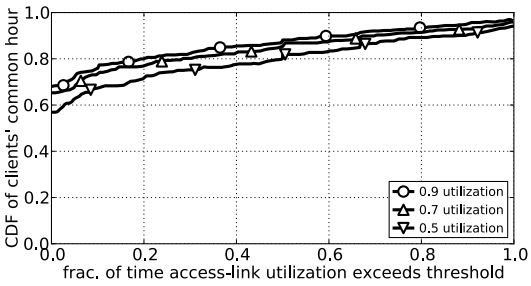
For this analysis we use data collected between December 2012 and January 2013 from 1,377 different end users and select the ones with UPnP-enabled gateways that properly report traffic counters ($\approx$40% of them). For every hour each individual user is online, the fraction of time in which the traffic in their access link (captured using UPnP counters) is higher than that generated by the client (captured using `netstat`) and sort these utilization values in increasing order. Using this information we then define the *common hour* for each user as the median utilization hour; i.e. the hour over which 50% of sampled-hours fall, and use this measurement as representative of each user.

We first look at the fraction of time the user's traffic shares the access link with traffic generated by other devices in the network. Figure 6 shows that for the *common hour* over 60% of users see no other traffic in the network (i.e. they are sole users of the access link) and for an additional 23% of users the fraction of time that the link is also utilized by other devices is less than 50% ($<$30 minutes).

The figure also shows that for 13% of the users their common hour have a utilization of 1, which means that in their representative hour the access-link is always shared with other devices in the network. We next look at the access-link utilization for this set of users. For each hour the users are online we compute the fraction of time in which the utilization of their access-link (captured using UPnP counters) is higher than different thresholds of their maximum link capacity (obtained through the use of NDT). Figure 7a shows the fraction of time the utilization of the access-link surpasses different utilization thresholds. As the figure shows, for 35% of these clients, the traffic present in their access-link never exceeds 50% of their link capacity at the common hour,



(a) Fraction of users with common hour utilization of 1 (access-link shared with other devices).



(b) All users.

Fig. 7: Fraction of time access-link utilization surpasses various utilization thresholds, (7a) for users with common hour utilization of 1, i.e. access-link shared with other devices; (7b) for all users.

and this fraction increases to 48% and 52% if we move our thresholds to 70% and 90% respectively. These results suggest that, even for users with high probability of encountering cross traffic in their network, the utilization of their link still allows for some measurements.

Finally we extend this analysis to include every user in our dataset, this is shown in Figure 7b. As the figure shows for 60% of the clients, the traffic in their access-link never exceeds 50% of their link capacity at the common hour, and that fraction increases to almost 70% when looking at link capacity of 90%. These results indicate that for the majority of users there is significant available capacity for network measurement without adversely affecting other applications using the network.

In conclusion, the results from this section suggest that despite the increased complexity of today's home networks, the deployment of a measurement platform hosted by end users is a viable proposition. Our analyses show that a measurement platform that continuously runs on an end-host can successfully avoid periods of interference from cross traffic inside the network. Furthermore they suggest it is possible to launch measurement probes without negatively impacting the performance of hosts hosting the platform.

IV. Dasu Design

In this section, we provide an overview of Dasu's design, discuss several system's components and briefly describe the API supporting measurement experiments.

A. System Overview

Dasu is composed of a distributed collection of clients and a set of management services. Dasu clients provide the desired coverage and carry on the measurements needed for broadband characterization and Internet experimentation. The *Management Services*, comprising the Configuration, Experiment Administration, Coordination and Data services, distribute client configuration and experiments and manage data collection. Figure 8 presents the different components and their interactions.

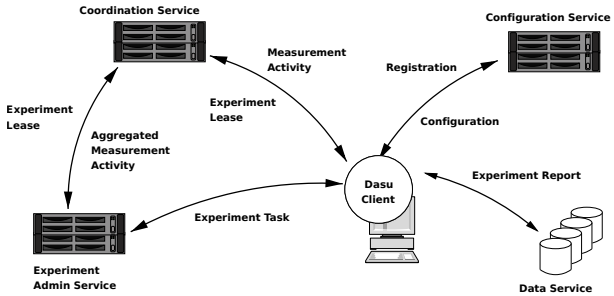


Fig. 8: Dasu system components.

Upon initialization, clients use the *Configuration Service* to announce themselves and obtain various configuration settings including the frequency and duration of measurements as well as the location to which experiment results should be reported. Dasu clients periodically contact the Experiment Administration Service, which assigns measurement tasks, and the Coordination Service to submit updates about completed probes and retrieve measurement limits for the different experiment tasks. Finally, clients use the Data Service to report the results of completed experiments as they become available.

B. The Dasu Client

Dasu clients run at the edge of the network within the context of a network-intensive hosting application. Each Dasu client (Fig.9) includes a set of *Probe Modules* to passively collect application metrics and run active measurements.

Dasu provides low-level measurement tools in the form of active probe modules that can be combined to build a wide range of measurement experiments. Currently available measurement primitives include traceroute, ping, Network Diagnostic Tool (NDT) [29], HTTP GET and DNS resolution.

A particularly important probe module is the *Passive Monitoring* module, responsible for collecting relevant signals from the host application in real time. This module constantly monitors the activities of BitTorrent and collects information such as (1) per torrent statistics, which include, number of RST received, upload rate, download rate, number of leechers, number of seeders, state (seeding, leeching, stopped...), etc; (2) application-wide statistics, which include, total upload and download rates (across all active torrents), number of active torrents, number of connected seeders, number of connected leechers, maximum rates set by the user (if any),etc; (3) system-wide statistics, which include connection-related system-wide statistics reported by the local operating system which include the number of active, current and passive connections, number of connections torn by TCP RST, and number of failed connections.

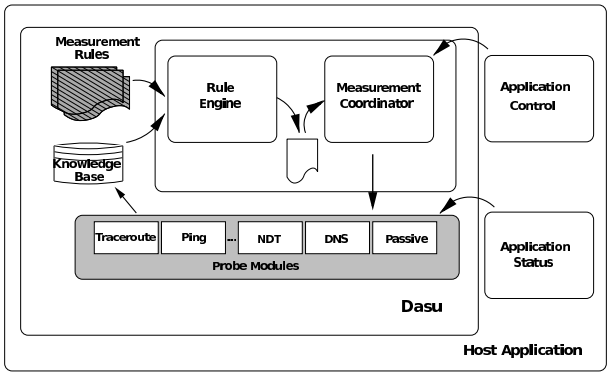


Fig. 9: Dasu client architecture.

C. Experiment Specification

Dasu is a dynamically extensible platform designed to facilitate Internet measurement experimentation while controlling the impact on hosts' resources and the underlying network. A key challenge in this context is selecting a programming interface that is both flexible (i.e., supports a wide range of experiments) and safe (i.e., does not permit run-away programs). We rejected several approaches based on these constraints and our platform's goals. These include offering only a small and fixed set of measurement primitives as they would limit flexibility. We also avoided providing arbitrary binary execution as handling the ramifications of such an approach would be needlessly complex.

We opted for a rule-based declarative model for experiment specification in Dasu. In this model, a rule is a simple `when-then` construct that specifies the set of actions to execute when certain activation conditions hold. A rule's left-hand side is the conditional part (`when`) and states the conditions to be matched. The right-hand side is the consequence or action part of the rule (`then`) i.e., the list of actions to be executed. Condition and action statements are specified in terms of read/write operations on a shared working memory and invocation of accessor methods and measurement primitives. A collection of rules form a program and a set of related programs define an experiment.

The rule-based model provides a clean separation between experiment logic and state. In our experience, this has proven to be a flexible and lightweight approach for specifying and controlling experiments. Experiment logic is centralized, making it easy to maintain and extend. Also, strict constraints can be imposed on rule syntax, enabling safety verification through simple static program analysis.

Dasu provides an extensible set of measurement primitives (modules) and a programmable API to combine them into measurement experiments. While this set is easily extensible (by the platform administrators) we have found it sufficient to allow complex experiments to be specified clearly and concisely. For instance, the experiment for the Routing Asymmetry case study (Sec. VI-B) was specified using only 3 different rules with an average of 24 lines of code per rule. Tables II and III provide a summary of this API and the current set of measurement primitives supported. The API includes some basic accessor methods (e.g. `getClientIps`, `getDnsServers` and `getEnvInfo`). The method `addProbeTask` serves to request the execution of measurements at a given point in time. The `commitResult` method allows results from the experiment to be submitted to the Data Service after completion.

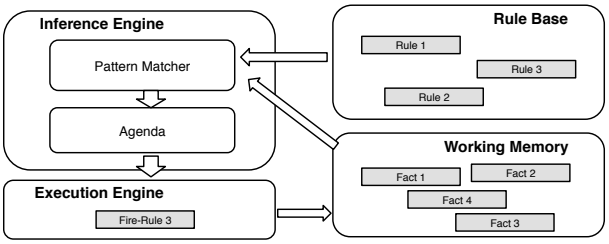


Fig. 10: Rule-based program execution environment.

Measurements primitives are invoked asynchronously by the *Coordinator*, which multiplexes resources across experiments. Progress and results are communicated through a shared *Working Memory*; through this working memory, an experiment can also chain rules that schedule measurements and handle results. (Fig. 10).

In addition to these active measurements, Dasu leverages the naturally-generated BitTorrent traffic as passive measurements (particularly in the context of broadband characterization [30]) by continuously monitoring the end-host Internet connection. Devising an interface to expose these passively collected measurements to experimenters is part of future work.

g) A Simple Example: To illustrate the application of rules, we walk through the execution of a simple experiment for debugging high latency DNS queries. Algorithm 11 lists the rules that implement this experiment. When rule #1 is triggered, it requests a DNS resolution for a domain name using the client's configured DNS server. When the DNS lookup completes, rule #2 extracts the IP address from the DNS result and schedules a ping measurement. After the ping completes, rule #3 checks the ping latency to the IP address and schedules a traceroute measurement if this is larger than 50 ms.

```
rule "(1) Resolve IP address through local DNS"
 when
  $fact : FactFireAction(action=="resolveIp");
 then
  addProbeTask(ProbeType.DNS, "example.com");
end

rule "(2) Handle DNS lookup result"
 when
  $dnsResult : FactDnsResult(toLookup=="example.com")
 then
  String ip = $dnsResult.getSimpleResponse();
  addProbeTask(ProbeType.PING, ip);
end

rule "(3) Handle ping measurement result"
 when
  $pingResult : FactPingResult()
 then
  if ( $pingResult.getRtt() > 50 )
   addProbeTask(ProbeType.TRACEROUTE, $pingResult.ip );
end
```

Fig. 11: Measurement experiment for debugging high latency DNS queries.

Method	Params.	Description
addProbeTask	<probe> <params> [<times>] [<when>]	Submit measurement request of the specified type.
commitResult	<report>	Submit completed experiment results to data server.
getClientIPs	[]	Return network information about the client including the list of IP addresses assigned (both public and private).
getDnsServers	[]	Return the list of DNS servers configured at the client.
getEnvInfo	[]	Return information about the plugin and the host node, including OS information and types of measurement probes available to the experimenter.

TABLE II: Dasu API – Methods.

Probe	Params.	Description
PING	<dest-list (IP/name)>	Use the local host ping implementation to send $ECHO\_REQUEST$ packets to a host.
TRACEROUTE	<dest-list (IP/name)>	Print the route packets take to a network host.
NDT	[<server>]	Run the M-Lab Network Diagnostic Tool [29].
DNS	[<server>] | [<timeout>] | [<tcp/udp>] | [<options>] | <DNS-msg>] | <dest-list>	Submit DNS resolution request to a set of servers.
HTTP	[server] | [<port>] | [<HTTP-Req>] | <url-list>	Submit HTTP request to a a given $< host, port >$ pair.

TABLE III: Dasu API – Measurement modules currently supported.

D. Security and Safety

Safely conducting measurements is a critical requirement for any measurement platform and particularly for one deployed at the Internet edge. We focus on two security concerns: protecting the host and the network when executing experiments. We expand on the former here and discuss the latter in the following section.

To protect the host, Dasu uses a sandboxed environment for safe execution of external code, ensures secure communication with infrastructure servers, and carefully limits resource consumption.

Experiment Sandbox. To ensure the execution safety of external experiments, Dasu confines each experiment to a separate virtual machine, instantiated with limited resources and with a security manager that implements restrictive security polices akin to those applied to unsigned Java applets. In addition, all Dasu experiments are specified as a set of rules that are parsed for unsafe imports at load time, restricting the libraries that can be imported. Dasu inspects the experiment's syntax tree to ensure that only specifically allowed functionality is included and rejects a submitted experiment otherwise.

Secure communication. To ensure secure communication between participating hosts and infrastructure servers, all configuration and experiment rule files served by the EA Service are digitally signed for authenticity and all ongoing communications with the servers (e.g. for reporting results) are established over secure channels.

Limits on resource consumption. Dasu must carefully control the load its experiments impose on the local host, as well as minimize the impact that users' interactions (i.e., with the host and the application) can have on experiments' results. To this end, Dasu limits consumption of hosts' resources[5] and restricts the launching of experiments to periods of low resource utilization; the monitored resources include CPU time, network bandwidth, memory and disk space.

[5]Currently 15% of any monitored resource.

To control CPU utilization, Dasu monitors the fraction of CPU time consumed by each system component (including the base system and each different probe module). Dasu regulates average CPU utilization by imposing time-delays on the activity of individual probe modules whenever their "fair share" of CPU time has been exceeded over the previous monitoring period. Dasu also employs watchdog timers to control for long-running experiments.

To control bandwidth consumption, Dasu passively monitors the system bandwidth usage and launches active measurements only when utilization is below certain threshold (we evaluate the impact of this policy on experiment execution time in Sec. V-C). Dasu uses the 95th percentile of client's throughput rates measured by NDT to estimate the maximum bandwidth capacity of the host and continuosly monitors host network activity (using the commonly available `netstat` tool). Based on pre-computed estimates of approximate bandwidth consumption for each probe, Dasu limits probe execution by only launching those that will not exceed the predetermined average bandwidth utilization limit. Additionally Dasu relies on a set of predefined limits on the number of measurement probes of each type that can be launched per monitored interval. While clients are allowed to dispense with their entire budget at once, the combined bandwidth consumed by all probe modules must remain below the specified limit.

To restrict memory consumption, Dasu monitors the allocated memory used by its different data structures and limits, for instance, the number of queued probe-requests and results. Measurement results are offloaded to disk until they can successfully be reported to the Data Service. Disk space utilization is also controlled by limiting the size of the different probe-result logs; older results are dropped first when the predetermined quota limits have been reached.

E. Delegating Code Execution to Clients

Dasu manages concurrent experiments, including resource allocation, via the Experiment Administration Service. As

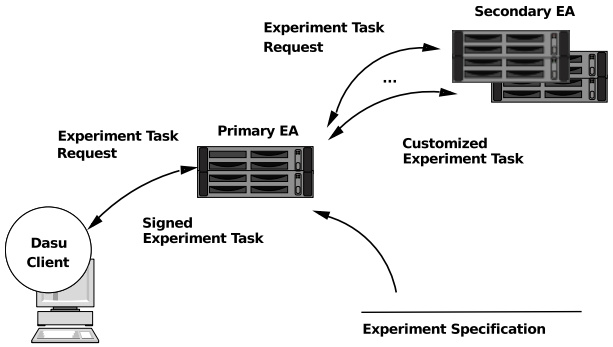


Fig. 12: Interaction between Dasu Clients and the Experiment Administration Service.

clients become available, they announce their specific characteristics (such as client IP prefix, connection type, geographic location and operating system) and request new experiment tasks. The *Experiment Administration (EA) Service* assigns tasks to a given client based on experiment requirements and characteristics of available clients (e.g. random sample of DSL users in Boston).

In the simplest of experiments, every Dasu client assigned to an experiment will receive and execute the same experiment task (specified as a stand-alone rules file). Dasu also enables more sophisticated experiments where experimenters specify which clients to use and how to execute tasks based on client characteristics.

Dasu adopts a two-tiered architecture for the EA Service, with a primary server, responsible for resource allocation, and a number of secondary servers in charge of particular experiments. The *Primary EA* server acts as a broker, allocating clients to experiments, by assigning them to the responsible secondary server, based on clients' characteristics and resource availability. The *Secondary EA server* is responsible for task parameterization and allocation of tasks to clients according to the experiment's logic. While the customized task assigned to a client is generated by the experiment's secondary server, all communication with Dasu clients is mediated by the primary server who is responsible for authenticating and digitally signing the assigned experiments. Figure 12 illustrates the interaction between Dasu clients and the EA Service.

Submitting External Experiments. Dasu supports third-party experiments through the two-tier architecture described above. Authorized research groups host their own Secondary EA server, with security and accountability provided through the Primary EA server.

In addition to providing a safe environment for executing experiments, all experiments submitted to Dasu are first curated and approved by the system administrators before deployment. This curation process serves as another safety check and ensures that admitted experiments are aligned with the platform's stated goals.

F. Coordination

In addition to controlling the load on and guaranteeing the safety of volunteer hosts, Dasu must control the impact that measurement experiments collectively may have on the underlying network and system resources. For instance, although the individual launch rate of `ping` measurements is limited, a large number of clients probing the same destination can overload it.

To this end, Dasu introduces two new constructs - *experiment leases* and *elastic budgets*, to efficiently allow the scalable and effective coordination of measurements among potentially thousands of hosts. In the following paragraphs, we describe both constructs and Dasu's approach to coordination.

Experiment Leases. To support the necessary fine-grained control of resource usage, we introduce the concept of experiment leases. In general, a *lease* is a contract that gives its holder specified rights over a set of resources for a limited period of time [31]. An *experiment lease* grants to its holder the right to launch a number of measurement probes, using the common infrastructure, from/toward a particular network location. Origin and/or targets for the probes can be specified as IP-prefixes or domain names (other forms, such as geographic location, could be easily incorporated).

Experiment leases are managed by the EA Service. The Primary EA server ensures that the aggregated use of resources by the different experiments is within the specified bounds. Secondary EA servers are responsible for managing experiment leases to control the load imposed by their particular experiments. To coordinate the use of resources by the Dasu clients taking part in an experiment, we rely on a distributed coordination service [32]. The Coordination Service runs on well-provisioned servers (PlanetLab nodes) using replication for availability and performance. Clients receive the list of coordination servers as part of the experiment description.

Before beginning an experiment, clients must contact a coordinator server to announce they are joining the experiment and obtain an associated lease. As probes are launched, the clients submit periodic updates to the coordination servers about the destinations being probed. The EA Service uses this information to compute estimated aggregate load per destination and to update the associated entries in the experiment lease. Before running a measurement, the Coordinator checks whether it violates the constraint on the number of probes allowed for the associated source and destination, and if so delays it. After a lease expires, the host must request a new lease or extend the previous one before issuing a new measurement. The choice of the lease term presents a trade-off between minimizing overhead on the EA Service versus minimizing client overhead and maximizing its use.

Elastic Budget. An experiment lease grants to its holder the right to launch a number of measurement probes (i.e., *a budget*) from/toward a particular network location. Due to churn and user-generated actions, the number of measurement probes a Dasu client can launch before lease expiration (i.e., the fraction of the allocated budget actually used) can vary widely. To account for this, Dasu introduces the idea of *elastic budgets* that expand and contract based on system dynamics.

Elastic budgets are computed by the EA Service and used to update bounds on experiment leases distributed to Dasu clients. The EA Service calculates the elastic budget periodically based on the current number of clients participating in the experiment, the number of measurement probes allowed, assigned and completed by each client. The EA Service uses

this elastic budget to compute measurement probe budgets for the next lease period for each participating client.

The budget is computed in the following way:

Let,

d, *destination*
M, *aggregate max # probes per unit time to dest d*
m, *max # of probes per unit time a client will launch*
n, *# of clients in the experiment*
a_i, *# of probes to dest d assigned to client i*
c_i, *# of probes to dest d completed by client i*
p_i, *probability client i will be online in the next unit time*

Then,

$$Budget = \begin{cases} M/n & \text{if } M/n < ppm \\ ppm & \text{if } M/n > ppm \end{cases}$$

where,

$$ppm = \sum_{i=1}^{n} p_i * f(i)$$

$$f(i) = \begin{cases} a_i - c_i & \text{if } (a_i \text{ -} c_i) < m \\ m & \text{if } (a_i \text{ - } c_i) > m \end{cases}$$

This approach is well suited for experiments where the server knows a priori what destinations each client should probe. In the case of experiments where the destinations to be probed are not assigned by the server, but obtained by the clients themselves (through a DNS resolution for example), the same approach can be used if we conservatively assume that a client will launch the maximum number of probes per unit of time whenever it is online.

G. Synchronization

Dasu also provides support for Internet experiments that require synchronized client operation (e.g. [33], [34]).

For coarse-level synchronization, Dasu clients include a cron-like probe-scheduler that allows the scheduling of measurements for future execution. All Dasu clients periodically synchronize their clocks using NTP. Assuming clients' clocks are closely synchronized, an experiment can request the "simultaneous" launch of measurements by a set of clients. We have found this to be sufficient to achieve task synchronization on the order of 1-3 seconds.

For finer-grained synchronization (on the order of milliseconds), Dasu adopts a remote triggered execution model. All synchronized clients must establish persistent TCP connections with one of the coordination servers. These connections are later used to trigger clients actions at a precise moment, taking into account network delays between clients and coordination servers.

Region	Penetration	Dasu Total	Dasu Total Countries
North America	78.6 %	21.45 %	60 %
Oceania/Australia	67.5 %	3.82 %	6 %
Europe	61.3 %	59.25 %	73 %
L. America/Carib.	39.5 %	1.68 %	65 %
Middle East	35.6 %	1.52 %	73 %
Asia	26.2 %	2.59 %	57 %
Africa	13.5 %	9.66 %	34 %

TABLE IV: Internet penetration[6] and Dasu coverage (as percentage of its total population of 95,222) by January 2013.

V. Current Deployment

We have implemented Dasu as an extension to a popular BitTorrent client [35] (publicly available since June 2010) as well as an standalone client (publicly available since June 2013). The following description and analysis are based on the BitTorrent extension, as it offers a large and widespread client population.

To participating users, Dasu provides information about the service they receive from their ISP [30], [36]. Access to such information has proven sufficient incentive for widespread subscription with over 95K users who have adopted our extension with minimum advertisement.[6]

This section demonstrates how Dasu clients collectively provide broad network coverage, sufficiently high availability and fine-grained synchronization for Internet experimentation.

A. Dasu Coverage

We show the coverage of Dasu's current deployment in terms of geography and network topology. Table IV lists broadband penetration in each primary geographic region and compares these numbers with those from our current Dasu's deployment.

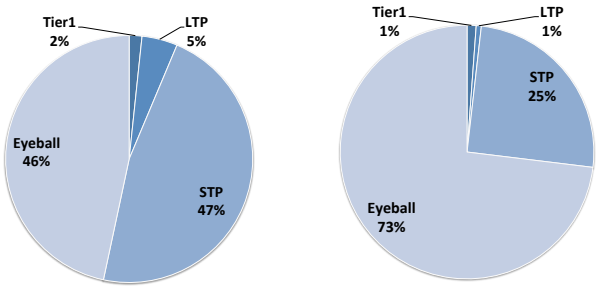
Given the high Internet penetration numbers in Europe and North America, the distribution of Dasu clients per region is not surprising. Note, however, the penetration of Dasu clients per region, measured as the percentage of countries covered. As the table shows, Dasu penetration is over 57% for most regions and is particularly high for Latin America/Caribbean (65%) and the Middle East (73%), two of the fastest growing Internet regions. Even in Africa Dasu penetration reaches 34%.

We also analyze Dasu's network coverage in terms of ASes where hosts are located. With our existing user-base at the end of July 2013, we have Dasu clients in 1,802 different ASes. We classify these ASes following a recently proposed approach [37], as follows:

- Tier-1: 11 known Tier-1s
- LTP: Large (non tier-1) transit providers and large (global) communications service providers
- STP: Small transit providers and small (regional) communication service providers
- Eyeball: Enterprise customers or access/hosting providers

[6]Upon download, users are informed of both roles of Dasu. Users can, at any point, opt to disable experiments from running and/or reporting performance information, without losing access to Dasu's broadband benchmarking information.

[6]http://www.internetworldstats.com



(a) Peer Distribution (b) AS Distribution

Fig. 13: Distribution of Dasu peers per AS (left). Distribution of ASes covered by Dasu peers (right).

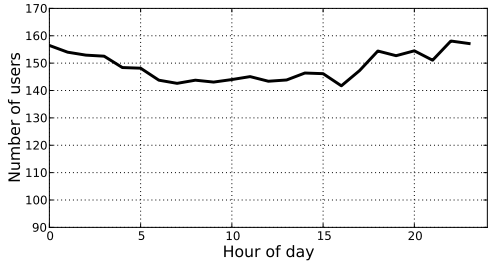


Fig. 14: Number of online Dasu clients over a 24-hour period. The fraction ranges from 39-44% of the total number of unique users, on average.

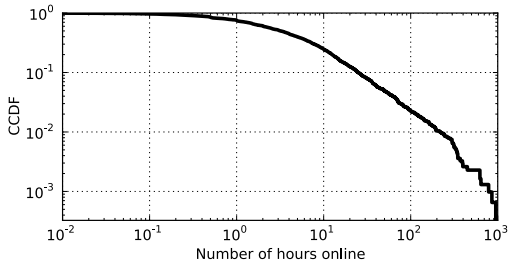


Fig. 15: Session time distribution of Dasu clients (time between their joining and leaving the system).

Figure 13a uses this classification to illustrate where Dasu peers are deployed. As the figure shows, 93% of Dasu peers are located in small transit providers and eyeball ASes; with only minimal presence in large transit and Tier-1 providers. Figure 13b presents the distribution of all the ASes covered by Dasu peers. This figure shows that 73% of the ASes covered by Dasu are eyeball ASes, highlighting the effectiveness of Dasu as a platform for capturing the view from the network edge.

B. Dasu Dynamics

In this section, we show that the churn from Dasu clients is sufficiently low to support meaningful experimentation. This churn is a result of both the volatility of Dasu's current hosting application (i.e. BitTorrent) and that of the end systems themselves. In the following analysis, we focus on the hosting application dynamics. In particular, we investigate what portion of clients are online at any moment, and whether their session times support common measurement durations.

First, we analyze Dasu clients' availability, using the percentage of clients online at any given hour over a 31-day period. Figure 14 plots this for the month of January 2013. The fraction of available clients during the period varies, on average, between 39% and 44% of the total number of unique users seen during a day, with a total of 1,473 active unique users for the month. With respect to the overall stability of the platform, for the same month of January 2013, we saw a total of 1,303 installs, 61 user uninstalls and 21 users who disabled reporting while continuing to run Dasu.

Next, we analyze how the duration of experiments is limited by client session times. Session time is defined as the elapsed time between it joining the network and subsequently leaving it. The distribution of clients' session times partially determines the maximum length of the measurement tasks that can be "safely" assigned to Dasu clients. Figure 15 shows the complementary cumulative distribution function of session times for the studied period. The distribution is clearly heavy-tailed, with a median session time for Dasu clients of 178 minutes or ≈ 3 hours.

Given an average session time, the fraction of tasks that are able to complete depends on the duration of the task – a function of the number of actual measurements and the load at the client. To evaluate the impact of typical client load conditions on task completion times, we designed a controlled experiment in the context of the IXP mapping case study described in Sec. VI, consisting of a fixed number of traceroutes issued by clients to discover potential peerings. The experiment was designed in such a way that given ideal client conditions (sufficiently low CPU and bandwidth load) the minimum time required by any Dasu client to complete it would be $\approx$75 seconds (1.25 minutes); this time would serve as a reference point when comparing against completion times under typical client conditions. For this purpose, we selected a random set of Dasu clients over a one week period in August 2013 and assigned such tasks multiple times during that period. Figure 16 shows a cumulative distribution function of the median task completion time from 164 different clients that completed 10 or more such tasks during that period. The figure shows how for 90% of the Dasu clients the median task successfully completes in < 150 seconds with 50% of clients completing the task in less than 120 seconds (2 minutes).

Now we look at the fraction of assigned tasks to Dasu clients that are successfully completed. Given that we have no control over clients' availability or disconnection times, only a fraction of assigned tasks will be successfully completed. To perform this analysis we look at the tasks assigned to 349 different Dasu clients over a 2-week period in the month of August 2013, taking only into account clients who were assigned 5 tasks or more during that period. Figure 17 shows the complementary cumulative distribution function of the fraction of successfully completed tasks from those assigned to each client. The plot shows that a large fraction of clients are able to complete the majority of assigned tasks in the face of churn with 60% of clients completing 80% or more of the tasks assigned to them.

C. Controlling Experimentation Load

To minimize Dasu's impact on host application performance and to ensure that user interactions do not interfere with

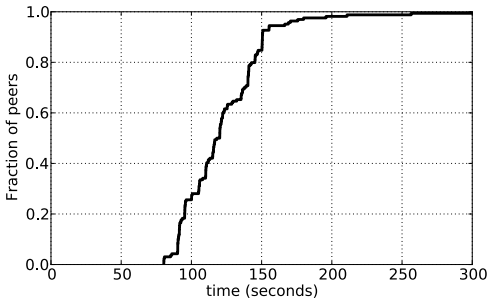


Fig. 16: CDF of median task completion times for tasks completed by Dasu clients. For 90% of the clients, the median task successfully completes in < 150 seconds (2.5 minutes).

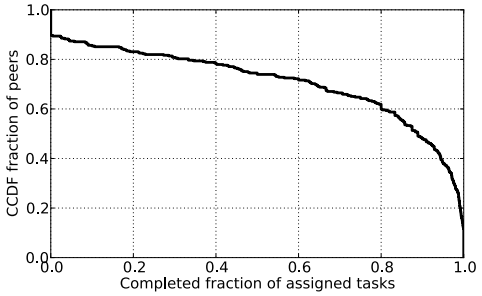


Fig. 17: CCDF of fraction of successfully completed tasks from those assigned. 60% of Dasu clients complete 80% or more of the tasks assigned to them.

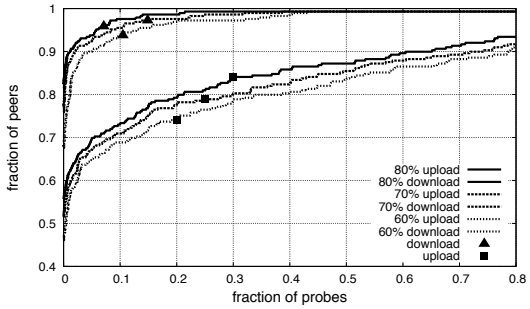


Fig. 18: Distribution of fraction of probes per peer that are delayed due to bandwidth constraints at the client.

scheduled measurements, Dasu enforces pre-defined limits on the number of probes executed per unit time and schedules measurements during low utilization periods. We evaluate the impact of one of these restrictions (on bandwidth utilization) on experiment execution by determining the portion of scheduled measurements delayed.

Figure 18 shows a CDF of the fraction of probes delayed by clients due to different bandwidth utilization constraints (60%, 70% and 80%), taken from a random subset of clients over a two-week period. The distribution shows, for instance, that capping at a download utilization of 80%, every scheduled probe can be launched immediately for 85% of the peers, and that for 98% of the peers less than 20% of the probes would require any delay. In contrast, a smaller fraction of probes (60%) experience no delay when an 80% utilization limit is imposed on the upload direction. This is expected, since broadband users are often allocated lower upload bandwidth than download.

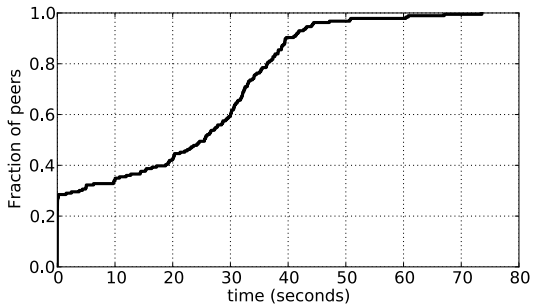


Fig. 19: CDF of median probe queue time for clients. For 30% of the clients, the median probe is launched $< 1sec.$ after being scheduled; with 60% of clients launching probes within 30 seconds of being scheduled.

Finally, we look at the amount of time probes are queued by the clients under typical load conditions from the moment they are requested until launched. Fig. 19 shows the queueing time of probes assigned to 186 Dasu clients for a given experiment over a 1-week period in July 2013. The figure plots the cumulative distribution function of the median probe-queue time on a per client basis for clients with at least 20 launched probes. The figure shows that for 30% of the clients, the median probe is launched $< 1sec.$ after being scheduled; with 60% of clients launching probes within 30 seconds of being scheduled.

D. Client Synchronization

To evaluate the granularity of Dasu's fine-grain synchronization capabilities, we run an experiment where Dasu clients were instructed to simultaneously launch an HTTP request to an instrumented web server. For a span of five minutes, approximately 30 clients were recruited to cooperate in the experiment. Following Ramamurthy et al. [33], as clients joined the experiment they were instructed to measure their latency to the target server as well as to the Coordination Server and to report back their findings.

At the end of the five minutes, clients were scheduled to launch their measurements (having adjusted each request based on their measured latencies) while we logged the arrival times of each incoming HTTP request at the target server. We repeated this experiment 10 times. Figure 20 shows the mean arrival time of each request with a crowd size of 31 clients. About 80% of the requests arrive within 300ms of each other, and 91% of the requests arrive within 1s of each other. This result is on par with the synchronization of 100s of milliseconds reported by Ramamurthy et al. [33]

Variations in the arrival times of the top 20% of requests are due to queuing delays in broadband networks [20] and errors in estimating the latency between clients and the coordinator server.

VI. Case studies of a platform at the Internet's edge

In this section, we present four case studies that illustrate the unique perspective a programmable, edge-based platform with

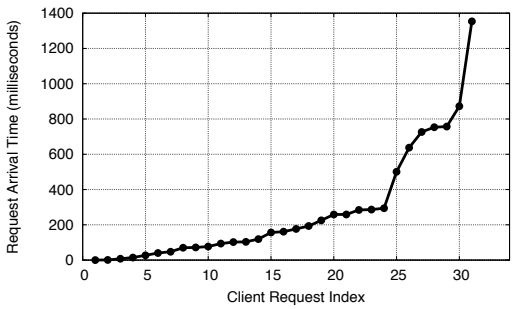


Fig. 20: Request arrival times at the target server. Approximately 80% of requests arrive within 300 ms.

a wide-spread and diverse set of vantage points brings to Internet measurement. These case studies highlight the important role that capabilities such as extensibility, synchronization and coordination play when launching measurements from such a platform.

A. Questioning Existing Experiments: Inferring AS-level Connectivity

The model of the Internet as a hierarchically-structured or "tiered" network of networks is changing [38]–[40]. The emergence of new types of networks (e.g., content providers, web hosting companies, CDNs) and their resulting demands on the Internet have induced changes in the patterns of interdomain connections; however, the precise degree and nature of these changes remains poorly understood.

Internet exchange Points (IXPs) are an important part of the rapidly developing Internet ecosystem because they facilitate the changes, enabling direct connections between member ASes. A recent study of a large European IXP has shown that some of the largest IXPs (e.g., DEC-IX and AMS-IX) handle traffic volumes that are comparable to those carried by some of the largest global ISPs and support peering fabrics consisting of more than 60% of all possible peerings among their 400-500 member ASes [41]. However, despite their importance, there exists little to no publicly available information about who is peering with whom nor about the nature of these peerings.

These changes in the network's structure demand changes to how we have traditionally conducted experiments. For instance to question the standard assumption of homogeneity that has been made when inferring AS-level connectivity at IXPs [42]–[44] – where a single traceroute between two ASes members of an IXP is sufficient to declare that these ASes are, as a whole, connected in the AS graph by a peer-peer link – we require an endemic population of vantage points that allows for finer-grained measurements.

The value of Dasu. Dasu provides an ideal platform to examine the validity of such assumptions. Its widespread and diverse user base provides vantage points in multiple prefixes within the same AS which allows us to identify prefix-specific features that could not be identified from a single location in the network. Additionally, Dasu's near-continuous availability of vantage points allows us to study temporal effects that are critical for the observed kind of peering. Lastly, conducting this kind of targeted experiments involving specific prefixes in specific ASes at particular IXPs relies critically on the programmability of Dasu.

Experiment Setup. To evaluate the validity of this homogeneity assumption, we set up an experiment to launch multiple traceroute probes, between the same pair of member ASes of a given IXP, from vantage points located inside different prefixes of the source AS and at different hours of the day.

Results. We found that about 15% of the peering links that Dasu discovered violated the assumed homogeneity condition. Depending on the prefixes, the probes either crossed the given IXP or were sent instead via one of the source AS's upstream providers.[7] Table V shows a concrete example of such fine-grained peering observed between two ASes at AMS-IX. By probing for peerings between AS1 and AS2 repeatedly from different prefixes in the ASes and separating the probes by the peers' local time, we obtained a view of these well-covered peerings throughout the day. For each data point in the table we corroborated the result across multiple traceroute probes and obtained thus an example of a consistent prefix-based peering – while probes launched from source prefix A towards AS2 are never seen crossing the IXP, probes launched from source prefix B towards AS2 seem to always go through the IXP.

In short, the discovery of such fine-grained or prefix-specific peering arrangements is proof that the traditional view that a single type of AS peering applies uniformly across all prefixes of an AS is no longer tenable. This finding has clear implications for measurement and inference of AS-level connectivity and poses new challenges and requirements for the platforms and techniques used for this type of studies.

B. Extending Earlier Experiments: Routing Asymmetry

A known limitation of traceroute, the most commonly used Internet diagnostic tool, is its one-sided perspective on a path [45], [46]. Traceroute's inability to measure the reverse path (from the target to the source) hinders operators and researchers alike, forcing unrealistic assumptions of symmetry that impact studies from path prediction [24] to prefix hijack detection [47].

To evaluate the extent of the routing asymmetry problem, one would ideally control the hosts at both ends of a path to obtain and compare the forward and reverse path. Other than for a relatively small and potentially not representative set of paths, such as those between hosts in an experimental platform [7], one typically lacks control at both ends of a route. Katz-Bassett et al. [45] propose a useful approach to determine the reverse path, even when one controls only one of the end points, by stitching together fragments of the path captured through a variety of methods that include the use of IP timestamp and record route options as well as limited source spoofing techniques. Their approach is however limited by the presence of routers that do not reply to IP Options probes this relies on.

The Value of Dasu. Dasu makes it possible to conduct an accurate analysis of path asymmetry between nodes in differ-

[7]The various reasons for why certain ASes engage in such non-traditional peering arrangements is beyond the scope of this proposal.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	prefixes
x	*x*	-	-	-	-	-	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	-	*x*	*x*	*A*
✓	✓	✓	✓	✓	-	-	-	-	✓	✓	✓	✓	✓	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	*B*

TABLE V: Prefix-based peering at Amsterdam Internet Exchange (AMS-IX) between two ASes. Columns show the hour, local time. Legend: '✓' probes crossed IXP; '*x*' probes did not cross IXP; '-' no probes.

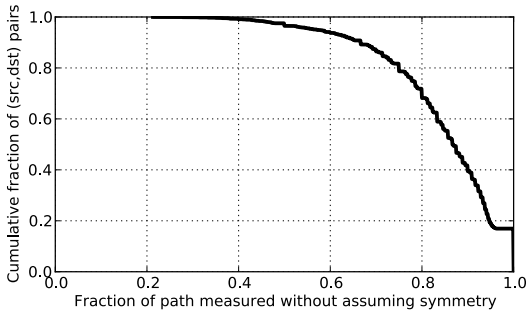


Fig. 21: CCDF of fraction of Dasu-PL path hops that can be directly measured using IP Options probes. 17% of paths reply to probes at each hop, meaning that we can determine the complete reverse path.

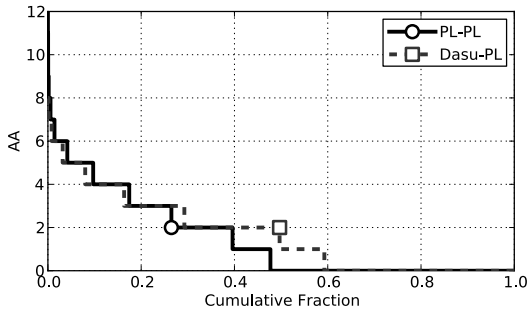
ent networks through its wide network coverage and its ability to schedule experiments and to synchronize the launching of measurements across nodes. Synchronizing measurements across both endpoints enables the concurrent probing of a path to minimize the impact of factors such as network load or time-of-day on routing decisions.

Experiment Setup. For our experimental setup, we extend the work by He et al. [46], comparing routing asymmetry for research and commercial networks, by examining the paths between stub (Dasu) and research (PlanetLab) networks.

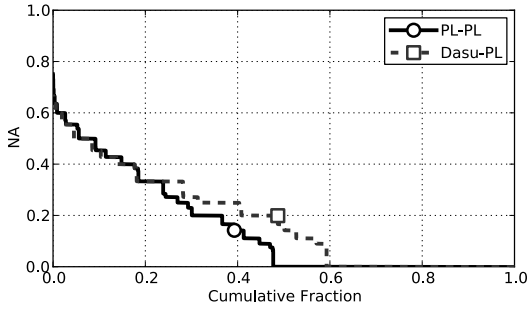
Results. We find that for ≈28% of the paths tested between Dasu clients and PlanetLab nodes (out of 8,046) reverse traceroute would be forced to make an incorrect symmetry assumption because a segment of the reverse path transits at least one AS that does not appear on the forward path and that does not respond to IP Options probes. Figure 21 shows that only 17% of the paths between Dasu and PlanetLab nodes respond to IP Options probes at every hop, this in contrast to the over 40% of paths between PlanetLab nodes reported in [45].

To study routing asymmetry between pairs of Dasu-PlanetLab (Dasu-PL) and PlanetLab-PlanetLab (PL-PL) nodes, we launched probes across 8,046 paths between Dasu clients and PlanetLab nodes, and across 10,067 paths between two PlanetLab nodes. To ensure accurate measures of routing asymmetry we had hosts at both endpoints probe the path concurrently.

We measure routing asymmetry by following the methodology described in [46]. This method maps hops in the forward path to those of the reverse path (either at the link-level or AS-level) and assigns a value of 0 if the hops are identical and a value of 1 if they are different. Through dynamic programming, it then selects the mappings for each path that results in the minimal distance. The minimal composite dissimilarity between a forward and reverse path is referred to as the *Absolute Asymmetry (AA)*, while the length-based *Normalized Asymmetry (NA)* is defined as AA normalized by the length of the round-trip path.



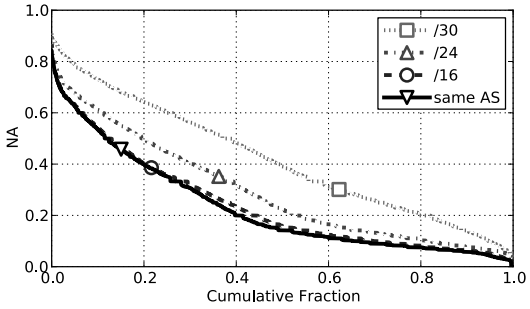
(a) Absolute Asymmetry



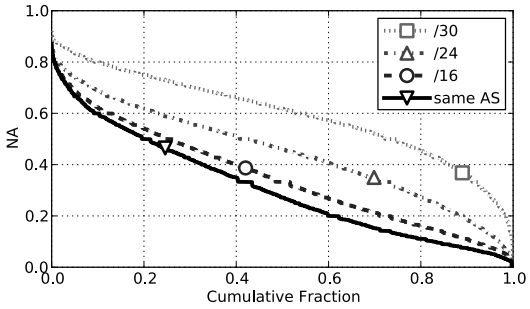
(b) Normalized Asymmetry

Fig. 22: CDFs of AS-level asymmetry in Dasu-PL and PL-PL paths; ≈60% of Dasu-PL paths show some degree of asymmetry, vs. 48% of PL-PL paths.

To compare the asymmetry in the AS-level paths between the two sets of paths (i.e., Dasu-PL and PL-PL), figures 22a and 22b show the cumulative distributions of the AS-level AA and NA metrics, respectively. It can be observed that the Dasu-PL paths not only have a higher percentage of asymmetric routes, but also display a higher magnitude of asymmetry than the PL-PL paths. To compare the two datasets at the link-level, we again follow the approach described in He et al. [46] and use their heuristics to determine if two IP addresses correspond to the interfaces of the same link. These heuristics consider two IP addresses to belong to the same point-to-point link if they belong to the same /30, /24, /16, or AS. For each of the four heuristics, Figures 23a and 23b show the cumulative distributions of the resulting NA metric for the PL-PL and Dasu-PL paths, respectively. As noted in [46], the first (/30) and last (AS) heuristics provide the upper and lower bounds, on the observed Internet routing asymmetry at the link level. While figures 22a and 22b show that the two sets of paths exhibit differences in routing asymmetry at the AS-level, figures 23a and 23b show these differences are significantly more pronounced at the link-level but depend greatly on the heuristics used.



(a) PlanetLab-PlanetLab



(b) Dasu-PlanetLab

Fig. 23: CDFs of link-level normalized asymmetry using different heuristics for IP to link mapping. Link-level NA is much lower for PL-PL paths than Dasu-PL paths.

C. Performing Novel Experiments: Evaluating a Recently-proposed DNS Extension

The *edns-client-subnet* EDNS0 extension (ECS) was developed to address the problems raised by the interaction between the DNS-based redirection techniques commonly employed by CDNs and the increasing use of remote DNS services. CDNs typically map clients to replicas on the location of the client's local resolver; since the resolvers of remote DNS services may be far from the users, this can result in reduced CDN performance. ECS aims to improve the quality of CDN redirections by enabling the DNS resolver to provide partial client location (i.e. client's IP prefix) directly to the CDN's authoritative DNS server. ECS is currently being used by a few public DNS services (e.g., Google DNS) and CDNs (e.g. EdgeCast) and can improve CDN redirections without modifications to end hosts.

The value of Dasu. To understand the performance benefits of the proposed ECS extension and capture potential variations across geographic regions would require access to a large set of vantage points. These vantage points should be located in access networks around the world and allow issuing the necessary interrelated measurement probes. These are some of the unique features that Dasu offers.

Dasu's extensibility allows for the creation and addition of a new probe module to generate and parse ECS-enabled DNS messages. Additionally, Dasu's user base allows us to obtain representative measurement samples from diverse regions and compare trends across geographic areas by looking at the relationships between raw CDN performance, relative proportions of clients affected by the extension, and the degree of performance improvement provided by the extension.

Experiment setup. This experiment extends the work by Otto et al. [48], which examined the impact of varying the amount of information shared by ECS (i.e. prefix length) and compared its performance to a client-based solution. We first obtain CDN redirections to edge servers both with the ECS extension enabled and disabled. Specifically, we query Google DNS (8.8.8.8) for an EdgeCast hostname. To obtain a redirection with ECS disabled, our DNS probe module sends a query with the ECS option that specifies 0 bytes of the client's IP prefix—this effectively disables the extension's functionality. For the ECS-enabled query, we provide the client's /24 IP prefix. After obtaining CDN edge server redirections with and without ECS's help, we conduct HTTP requests to both sets of CDN edge servers to measure the application-level performance in terms of latency to obtain the first byte of content. For the results from each client, we compare the median performance with and without ECS being enabled.

Results. We analyze results from a subset of 1,185 Dasu clients that conducted this experiment over a 4 month period from September 12th, 2011 to January 16th, 2012.[8] Figure 24 shows the relationship between HTTP latency with ECS disabled and the performance benefits (latency savings) with ECS enabled. We classify users by geographic region; the percentages listed in the legend indicate the fraction of all sampled clients from that region. In all regions, sampled clients are located in a diverse set of networks; even in Oceania—the region with fewest clients—we cover 9 ISPs in Australia and 4 in New Zealand. The figure plots the subset of samples in which EDNS impacted HTTP performance.

While we find clients in all these regions that obtained HTTP performance improvements with ECS enabled, the samples tend to cluster by region. Although clients in North America and Western Europe both typically see HTTP latencies between 20 and 200 ms, the North American clients generally obtain higher percentage savings. This would indicate that the CDN's infrastructure in North America is relatively dense in comparison to that of the public DNS service's deployment. Clients in Oceania typically have relatively high HTTP latencies between 200 and 1000 ms with ECS disabled—but commonly realize savings of 70–90% with ECS enabled. This is likely a result of the specific deployments of the CDN and DNS services; although there are actually CDN edge servers near to clients in this region, it appears that the nearest Google DNS servers are farther away, resulting in reduced HTTP performance when ECS is disabled. Finally, we compare the number of clients with benefits from ECS between Eastern Europe and Oceania; while clients in Oceania actually comprise a slightly smaller fraction of the overall sample, the number of clients that actually observed better performance is much higher than for clients in Eastern Europe.

D. Evaluating Earlier Systems: King Evaluation

It is well-known that systems and measurements may exhibit significantly different behaviors when deployed/used at the Internet edge.

[8]Each participating client runs the experiment once over that time.

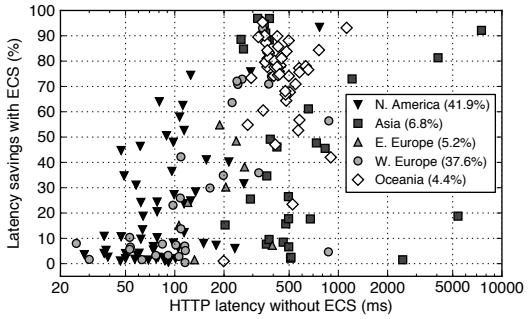


Fig. 24: HTTP latency vs. the performance benefits provided by ECS, by geographic region. Percentages in the legend indicate the geographic composition of the dataset.

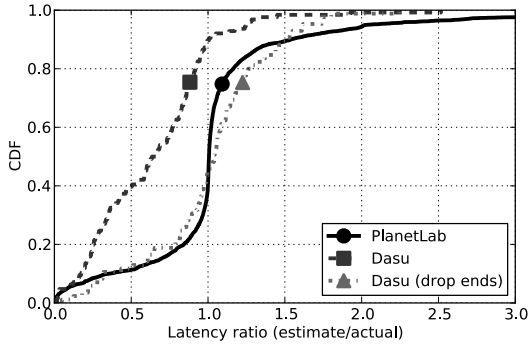


Fig. 25: CDF of the accuracy of King latency estimates (ratio between estimated and actual latency) between pairs of PlanetLab nodes or pairs of Dasu nodes. King provides good estimates for the majority of pairs of PlanetLab nodes, but consistently under-estimates end-to-end latency between Dasu nodes due to the significant access link latencies. However, King does provide comparable accuracy for estimating the in-network latency between Dasu peers, as shown by the "Dasu (drop ends)" curve, which subtracts the access link latencies of Dasu peers.

The value of Dasu. We illustrate the advantages of Dasu as a platform for system evaluation, with a study of King [49], a well-understood measurement tool that estimates the latency between two end hosts. Specifically, we compare the accuracy of King when applied to nodes in GREN networks (PlanetLab [7]) vs. nodes in access networks (Dasu clients).

Experiment Setup. In our experiment, we performed ping and traceroute measurements either between pairs of PlanetLab or Dasu nodes, and concurrently applied the King technique to estimate the latency between the two nodes. The King technique assumes that in most cases it is possible to find DNS name servers that are topologically close to the end hosts; the latency between the name servers serves as an estimate of the actual latency between the two nodes. King also assumes that an authoritative DNS server for a node's domain is likely to be near to the node itself. Such a DNS server can be identified by first doing a reverse DNS lookup for the node's IP address. When applying the heuristics described in [49] to identify a nearby authoritative DNS server, we succeeded for 83% (67%) of PlanetLab (Dasu) IP addresses.

The King technique also requires that at least one of the two DNS servers supports recursive queries coming from remote clients. When King was published in 2002, it was reported that a significant fraction of DNS servers supported this feature – 72% for web servers and 76% for end-user addresses. As a result, the technique could be used to estimate latency between about 92% and 94% of pairs of web servers and end-user addresses, respectively. However, the overall efficacy of the King approach has decreased significantly since 2002, mainly because only about 8.7% of DNS servers in today's Internet support recursive queries [50] – a significant reduction from the 72-76% seen in 2002. Our experimental results are in agreement with the findings reported in [50] and show that only some 15% of PlanetLab nodes and 10% of Dasu clients' DNS servers supported recursive queries from remote clients. Using the King technique, we are able to estimate latency for only about 28% of pairs of PlanetLab nodes and 19% of pairs of Dasu peers.

Results. For the pairs of PlanetLab or Dasu nodes for which we could apply the King technique, we evaluated its accuracy by comparing King's estimated latency to the actual latency between each pair of nodes. Figure 25 shows the cumulative distribution function (CDF) of the ratio between estimated and actual latency for pairs of PlanetLab nodes and pairs of Dasu IP addresses. For PlanetLab, 63% of the pairs we tested had estimated latency within 20% of the actual latency. However, when estimating end-to-end latency between pairs of Dasu peers, only 23% of the estimates were within 20% of the actual latency. The higher error rates for estimates between Dasu users are a result of the high latency encountered on the end-users' access links. Figure 26 shows the CDFs of the fraction of the end-to-end latency due to the first- and last-mile hops and demonstrates that access link latencies are a significant component of end-to-end latency for paths between Dasu nodes – over 50% of the end-to-end latency for 33% of node pairs. This is not the case for paths between PlanetLab nodes. Clearly, the King technique is not able to provide accurate estimates for the end-to-end latency between nodes behind high-latency access links. However, we observed that King does provide good estimates of in-network latency between Dasu peers – $<$ 20% error for 51% of node pairs (see the 'Dasu (drop ends)' curve in Fig. 25). In-network latency is computed by subtracting the access link latencies from the end-to-end latency. In this case, King provides a degree of accuracy that is comparable to that of the estimates between PlanetLab nodes.

VII. Conclusion

We presented Dasu, a measurement experimentation platform for the Internet's edge that supports and builds on broadband characterization as an incentive for adoption. We showed that despite the increased complexity of today's home networks, the deployment of a measurement platform hosted by end users is a viable alternative. We described Dasu's design and implementation and used our current deployment to demonstrate how participating nodes collectively offer broad network coverage, high availability and fine-grained synchronization to enable Internet measurement experimentation.

Dasu represents but a single point in a large design space. We described our rational for our current design choices, but

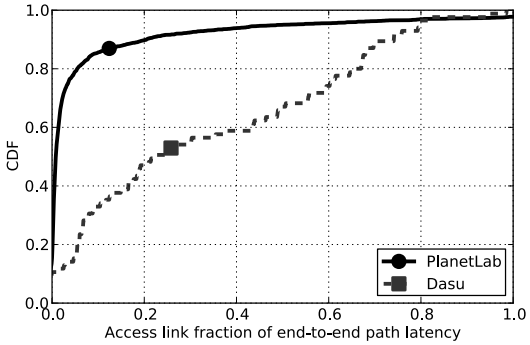


Fig. 26: CDF of the fraction of end-to-end path latency due to first- and last-mile access link latencies, for paths between PlanetLab nodes and paths between Dasu nodes. PlanetLab access link latencies generally make a small contribution to total latency (e.g. <20% of latency for 90% of pairs). In contrast, for 33% of the paths between Dasu nodes, the access link latencies are *more than half* of the total latency.

expect to revisit some of these decisions as we learn from our own and other experimenters' use of the platform.

We presented four case studies that demonstrate Dasu's capabilities and illustrate the unique perspective it brings to Internet measurement. As part of ongoing work, we are exploring the use of node availability prediction for experimentation, approaches to ensure the integrity of experimental results, and allowing fine-grained control of experiments by end users.

The Dasu client is open source and available for download from http://aqualab.cs.northwestern.edu/projects/115-dash-dual-purpose-platform.

VIII. Acknowledgments

We would like to thank Lorenzo Alvisi, Rebecca Isaacs, Aaditeshwar Seth and the anonymous reviewers for their invaluable feedback. We are always grateful to Paul Gardner for his assistance with Vuze and the users of our software.

This research was supported in part by the National Science Foundation through Awards CNS 1218287, CNS 0917233 and II 0855253 and by a Google Faculty Research Award.

References

[1] N. Spring, L. Peterson, A. Bavier, and V. Pai, "Using PlanetLab for network research: Myths, realities and best practices," *ACM SIGOPS Op. Sys. Rev.*, 2006.

[2] M. Casado and T. Garfinkel, "Opportunistic measurement: Spurious network events as a light in the darkness," in *Proc. of HotNets*, 2005.

[3] M. Dischinger, A. Haeberlen, I. Beschastnikh, K. P. Gummadi, and S. Saroiu, "SatelliteLab: Adding heterogeneity to planetary-scale network testbeds," in *Proc. of ACM SIGCOMM*, 2008.

[4] M. Rabinovich, S. Triukose, Z. Wen, and L. Wang, "Dipzoom: The Internet measurements marketplace," in *Proc. IEEE INFOCOM*, 2006.

[5] Y. Shavitt and E. Shir, "DIMES: Let the Internet measure itself," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 5, October 2005.

[6] J. Cappos, I. Beschastnikh, A. Krishnamurthy, and T. Anderson, "Seattle: a platform for educational cloud computing," in *Proc. of the 40th ACM technical symposium on Computer science education*, ser. SIGCSE '09, 2009.

[7] PlanetLab, "An open platform for developing, deploying, and accessing planetary-scale services," Jul. 2013, accessed: 2013-07-26. [Online]. Available: http://www.planet-lab.org/

[8] RIPE, "RIPE atlas," May 2013, accessed: 2013-05-21. [Online]. Available: http://atlas.ripe.net/

[9] Keynote, "Internet health report," Jan. 2013, accessed: 2013-01-7. [Online]. Available: http://internetpulse.net/

[10] FCC, "Broadband network management practices – en banc public hearing," February 2008, http://www.fcc.gov/broadband_network_management/hearing-ma022508.html.

[11] K. Chen, D. R. Choffnes, R. Potharaju, Y. Chen, F. E. Bustamante, D. Pei, and Y. Zhao, "Where the sidewalk ends: extending the Internet AS graph using traceroutes from P2P users," in *Proc. ACM CoNEXT*, 2009.

[12] Caida, "Ark," Jul. 2013, accessed: 2013-07-26. [Online]. Available: http://www.caida.org/projects/ark/

[13] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. ACM SOSP*, 2001.

[14] J. Ledlie, P. Gardner, and M. Seltzer, "Network coordinates in the wild," in *Proc. of USENIX NSDI*, 2007.

[15] D. R. Choffnes and F. E. Bustamante, "Pitfalls for testbed evaluations of Internet systems," *SIGCOMM Comput. Commun. Rev.*, April 2010.

[16] R. Zhang, C. Tang, Y. C. Hu, S. Fahmy, and X. Lin, "Impact of the inaccuracy of distance prediction algorithms on Internet applications: an analytical and comparative study," in *Proc. IEEE INFOCOM*, 2006.

[17] 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*, 2006.

[18] K. P. Gummadi, H. V. Madhyastha, S. D. Gribble, H. M. Levy, and D. Wetherall, "Improving the reliability of Internet paths with one-hop source routing," in *Proc. USENIX OSDI*, 2004.

[19] D. R. Choffnes, M. A. Sánchez, and F. E. Bustamante, "Network positioning from the edge: an empirical study of the effectiveness of network positioning in P2P systems," in *Proc. IEEE INFOCOM*, 2010.

[20] M. Dischinger, A. Haeberlen, K. P. Gummadi, and S. Saroiu, "Characterizing residential broadband networks," in *Proc. of IMC*, 2007.

[21] C. R. Simpson, Jr and G. F. Riley, "NETIhome: A distributed approach to collecting end-to-end network performance measurements," in *Proc. of PAM*, 2004.

[22] S. Sundaresan, W. de Donato, N. Feamster, R. Teixeira, S. Crawford, and A. Pescapè, "Broadband internet performance: a view from the gateway," in *Proc. of ACM SIGCOMM*, 2011.

[23] N. Spring, D. Wetherall, and T. Anderson, "Scriptroute: a public Internet measurement facility," in *Proc. USENIX USITS*, 2003.

[24] H. M. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "iPlane: an information plane for distributed systems," in *Proc. USENIX OSDI*, 2006.

[25] D. R. Choffnes, F. E. Bustamante, and Z. Ge, "Crowdsourcing service-level network event monitoring," in *Proc. of ACM SIGCOMM*, 2010.

[26] L. DiCioccio, R. Teixeira, and C. Rosenberg, "Measuring and characterizing home networks," in *Proc. ACM SIGMETRICS*, 2012.

[27] L. DiCioccio, R. Teixeira, and Rosenberg, "Characterizing home networks with HomeNet Profiler," Technicolor, Tech. Rep., 09 2011, cP-PRL-2011-09-0001.

[28] L. DiCioccio, R. Teixeira, M. May, and C. Kreibich, "Probe and pray: Using UPnP for home network measurements," in *Proc. of PAM*, 2012.

[29] MLabs, "Network diagnostic tool," Jun. 2013, accessed: 2013-06-6. [Online]. Available: http://www.measurementlab.net/run-ndt/

[30] Z. S. Bischof, J. S. Otto, M. A. Sánchez, J. P. Rula, D. R. Choffnes, and F. E. Bustamante, "Crowdsourcing ISP characterization to the network edge," in *Proc. of W-MUST*, 2011.

[31] C. G. Gray and D. R. Cheriton, "Leases: An efficient fault-tolerant mechanism for distributed file cache consistency," in *Proc. ACM SOSP*, 1989.

[32] P. Hunt, M. Konar, F. P. Junqueira, and B. Reed, "ZooKeeper: wait-free coordination for Internet-scale systems," in *Proc. USENIX ATC*, 2010.

[33] P. Ramamurthy, V. Sekar, A. Akella, B. Krishnamurthy, and A. Shaikh, "Remote profiling of resource constraints of web servers using mini-flash crowds," in *Proc. USENIX ATC*, 2008.

[34] A.-J. Su, D. Choffnes, A. Kuzmanovic, and F. Bustamante, "Drafting behind Akamai: Travelocity-based detouring," in *Proc. of ACM SIGCOMM*, Sep. 2006.

[35] Vuze, "Vuze," Jul. 2013, accessed: 2013-07-26. [Online]. Available: http://www.vuze.com/

[36] SamKnows, "Accurate broadband information for consumers, governments and ISPs," Jul. 2013, accessed: 2013-07-26. [Online]. Available: http://www.samknows.com/

[37] A. Dhamdhere and C. Dovrolis, "Ten years in the evolution of the Internet ecosystem," in *Proc. of IMC*, 2008.

[38] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, "Internet inter-domain traffic," in *Proc. of ACM SIGCOMM*, 2010.

[39] A. Dhamdhere and C. Dovrolis, "The Internet is flat: modeling the transition from a transit hierarchy to a peering mesh," in *Proc. ACM CoNEXT*, 2010.
[40] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "The flattening Internet topology: natural evolution, unsightly barnacles or contrived collapse?" in *Proc. of PAM*, 2008.
[41] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger, "Anatomy of a large european ixp," in *Proc. of ACM SIGCOMM*, 2012.
[42] K. Xu, Z. Duan, Z.-L. Zhang, and J. Chandrashekar, "On properties of Internet exchange points and their impact on AS topology and relationship," in *NETWORKING*, 2004.
[43] Y. He, G. Siganos, M. Faloutsos, and S. Krishnamurthy, "Lord of the links: a framework for discovering missing links in the Internet topology," *IEEE/ACM Transactions on Networking*, 2009.
[44] B. Augustin, B. Krishnamurthy, and W. Willinger, "IXPs: mapped?" in *Proc. of IMC*, 2009.
[45] E. Katz-Bassett, H. V. Madhyastha, V. K. Adhikari, C. Scott, J. Sherry, P. Van Wesep, T. Anderson, and A. Krishnamurthy, "Reverse traceroute," in *Proc. of USENIX NSDI*, 2010.
[46] Y. He, M. Faloutsos, S. Krishnamurthy, and B. Huffaker, "On routing asymmetry in the Internet," in *Proceedings of IEEE Globecom*, 2005.
[47] Z. Zhang, Y. Zhang, Y. Charlie, H. Z. Morley, and M. R. Bush, "ispy: Detecting ip prefix hijacking on my own," in *In Proc. ACM SIGCOMM*, 2008.
[48] J. S. Otto, M. A. Sánchez, J. P. Rula, and F. E. Bustamante, "Content Delivery and the Natural Evolution of DNS: remote dns trends, performance issues and alternative solutions," in *Proc. of IMC*, 2012.
[49] K. P. Gummadi, S. Saroiu, and S. D. Gribble, "King: estimating latency between arbitrary Internet end hosts," in *Proc. ACM IMW*, 2002.
[50] G. Sisson, "DNS Survey: October 2010," Oct. 2010, accessed: 2012-09-12. [Online]. Available: http://dns.measurement-factory.com/surveys/201010/