Assessing Performance of Internet Services on IPv6

David Plonka
University of Wisconsin-Madison
Email: plonka@cs.wisc.edu

Paul Barford
University of Wisconsin-Madison
Email: pb@cs.wisc.edu

***Abstract*—The exhaustion of the IPv4 address space significantly increases the urgency for transitions to IPv6. Since native IPv6 support is not yet ubiquitous, a major concern of users and service providers (*e.g.,* Facebook, Google, etc.) is that end-to-end performance via IPv6 could be substantially worse than IPv4. In this paper, we develop an analysis method and framework that matches DNS rendezvous information to flows so that we can compare and contrast performance over both protocols for a variety of Internet services. Our initial analyses focus on the basic services that are accessed using both protocols, observed client behaviors, and a presentation of performance characteristics of services using both IPv4 and IPv6. Our objective is to detect and expose differences by passive measurement without access to application traffic payloads. To demonstrate our method, we present results of an empirical feasibility study that considers the issue of Internet services performance over IPv6. Our study uses data collected on the World IPv6 Day, including both DNS requests/responses and flow export records for dual-stack hosts operating at a large research university. Our results expose various performance characteristics of Internet services that support IPv6: *(1)* Robust measures of services' flow bit rate distributions vary significantly by time of day, by number of active local clients, and by IP protocol version (6 or 4). *(2)* These rate characteristics differ amongst services. *(3)* There are regimes of time in which IPv6 flow bit rates exceed those of IPv4 and others where the IPv4 flow rates exceed those of IPv6.**

I. Introduction

With the near exhaustion of the IPv4 address space, the deployment and operation of Internet Protocol version 6 (IPv6) in production networks is upon us. Indeed, years have passed since World IPv6 Day, June 8, 2011, when numerous Internet service providers demonstrated their readiness by providing their services over IPv6, and many continue to do so. There are myriad reports of significant amounts of IPv6 traffic being transited [5], suggesting that more and more users and services are utilizing IPv6.

However, achieving good end-to-end performance over IPv6 is challenging for a number of reasons. First, IPv6 must be deployed and operated in parallel with the existing IPv4 infrastructure. Second, new network configuration tasks must be introduced and performed. Also, network monitoring and performance assessment is more complicated with IPv6 because there are now populations of users with differing environments due to multiple IP versions; many users have IPv4, some might have only IPv6, and an increasing number have both, *i.e.,* "dual-stack" hosts. Such challenges motivate the need for general methods to assess IPv6 performance during deployment and during the long-lived simultaneous operation of both protocol versions.

In this paper we present a new method for assessing the performance of Internet services over IPv6. Our objective is to (i) accurately assess performance based on passive measurements, (ii) provide the capability to compare and contrast IPv6 performance with that of IPv4, and (iii) provide an assessment that is both independent of the end-hosts and the Internet services they access. We perform this assessment based on passive measurements gathered at two observation points: one at or near the clients recursive Domain Name System (DNS) resolver and the other at any point along the end-to-end path. Our approach does not need privileged knowledge of the Internet services nor special access to the end-hosts involved in the exchange of traffic. We then develop a framework and tools to detect and inspect performance differences between IPv6 and IPv4 for Internet services.

Our performance assessment method is predicated on the fact that client hosts, on IPv4 and IPv6, necessarily employ some *rendezvous* mechanism to discover the IP address(es) of an Internet service before interacting with it. For many types of port-based services (*e.g.,* HTTP, HTTPS, SMTP, and IMAP) and for most of those with early IPv6 support, the DNS is that rendezvous mechanism. For instance, users' client hosts rendezvous with Facebook, an Internet service, by resolving the domain name "www.facebook.com." When a service supports IPv4 and IPv6 simultaneously, clients use a common rendezvous mechanism for both Internet protocol versions, such as the DNS.

Our framework determines service performance in three steps: *(1)* **measurement** in two forms: *(a)* full capture of low-volume DNS query/response packets and *(b)* collection of 5-tuple IP flow export records with duration and byte count of high-volume application traffic; *(2)* **classification** of flows source and/or destination IP addresses by matching them to their corresponding domain names (when possible) based on query names and the resulting IP addresses in DNS responses; *(3)* **performance inference** by constructing a distribution of flow bit rates and applying statistical techniques.

The common DNS rendezvous mechanism is an aid in the transition to IPv6 because it keeps the user from having to make the difficult decision about whether to use IPv4 or IPv6. Modern dual-stack Internet hosts issue DNS queries to request IPv6 and/or an IPv4 address(es) for the domain name of the desired service. When a given service supports both IPv4 and IPv6, the client host, having retrieved one type of address or both, makes a decision as to which peer address to use; this decision is typically performed in the IP imple-

mentation, resolver, or application. For example, a dual-stack client wishing to access a World-Wide Web (WWW) service named "www.example.com" might issue a AAAA (or "quad-A") query for that service's IPv6 address: 2001:0db8::2:1. It might also, simultaneously or subsequently, issue an A address query for that service's IPv4 address: 192.0.2.1. One or the other of these peer addresses will be selected, possibly influencing the resulting performance and user experience.

We demonstrate the capabilities of our method by collecting traffic trace data for a campus population having dual-stack client hosts. This trace data, collected on World IPv6 Day, consists of 24 hours of: (i) all recursive DNS queries issued by those client hosts and (ii) complete flow export records (from NetFlow configured without sampling, *i.e.,* non-packet-sampled) for those client hosts' traffic as it traverses a campus core router.

During the course of this work we encountered a number of challenges. We desire a common stream and storage format that can encapsulate rendezvous information (*i.e.,* DNS queries and responses), packet capture, and flow export information, for online and offline processing. Furthermore, we require high-performance C data structures and other APIs available in a scripting language for rapid prototyping and ad hoc analysis and reporting. Another challenge arose from the complication inherent in dual-stack hosts: namely, they have multiple IP address "identities" and it is non-trivial to determine by passive observation whether or not a given IPv4 and IPv6 address are actually bound to the same host. This is pertinent because hosts often use one IP address for their DNS queries and responses, and the other to interact with other remote hosts; by observation, it can seem as if the host has a covert channel by which it gleans rendezvous information. We addressed this by developing a consensus-based strategy to infer IPv6 rendezvous information. Other challenges arose from the way in which Internet operators are deploying the IPv6 portion of their services. For instance, we found that many domain names were multiplexed to one IPv6 address, sometimes for seemingly unrelated services. This practice, sometimes called "virtual hosting," was popularized in the IPv4 Internet, in part, because of address scarcity. This suggests IPv6 service architects and operators mimic techniques used with IPv4 that may not be necessary nor be ideal with IPv6.

Our results capture various performance phenomena for Internet services on IPv6: *(1)* Flow bit rates vary significantly even when measured by robust statistics such as interquartile range and vary by time of day, by active client hosts, and by IP protocol version. *(2)* Flow bit rates differ significantly amongst services, *e.g.,* Facebook and Google. *(3)* For both services, there are regimes in which IPv4 flow bit rates are greatest and other regimes in which IPv6 flow rates are greatest.

There are a number of instances of related work on IPv6 that are based on both passive and active measurements [6], [16], [10], [8], [7], [15], [12], [5]. These differ from our study in that they primarily focus on uptake and deployment, use different measures (*e.g.,* latency), or measure traffic (typically in aggregate) with a vastly different methodology.

The recent work of Bermudez *et al.,* who developed DN-Hunter [4], is the most similar to ours. While their goals differ and they do not assess service performance, they employ our rendezvous-based method from prior work [13], [14] to annotate flows in an online fashion. This paper augments our prior method with a transport and storage framework for annotating flows. To the best of our knowledge, ours is the first study that presents a scalable and robust methodology for passively assessing IPv4 versus IPv6 performance for services with which clients rendezvous via the DNS.

II. Method and Implementation

Our goal is to develop a performance assessment framework with the following characteristics or features:

- a method employing rendezvous-based traffic classification and robust statistics to determine and expose IPv6 and IPv4 performance phenomena for Internet services.
- an extensible mechanism for encapsulating the requisite rendezvous and traffic trace data prior to classification and for annotating IP traffic trace data afterward.
- a mechanism for transmitting streams of the encapsulated data to distributed framework components for online analysis in near real time.
- a serialized data file format for storing the encapsulated and annotated data for offline analysis, as in this study.
- a scripting language interface for the high-performance C code data structures and APIs that can be used to build distributed framework components and to conveniently run ad hoc analyses and reports.

We utilize this framework to assess the performance of traffic exchanged between hundreds of IPv6-capable campus hosts and Internet services with which these hosts rendezvous via the DNS. To this end, we've reimplemented TreeTop (previously a standalone tool) by incorporating the features above as the *TreeTop Framework*. TreeTop now processes "nmsg" streams rather than pcap input. The nmsg format [3] is an extensible encapsulation scheme, based on Protocol Buffers [9], that provides both transmission (for online analysis) and serialization to a data file (for storage and offline analysis.) Here, we first give a brief overview and highlight our TreeTop framework's features.

TreeTop processes incoming streams of DNS and flow export data for application traffic. The streams are network messages (nmsg) of two types: (i) DNS query and response (dnsqr) and (ii) NetFlow data (nfdump). The dnsqr messages contain all the interesting parts of a DNS query and response plus the time observed and delay between the corresponding query and response. The nfdump message contains flow data with typical transport header information and other attributes of unidirectional 5-tuple IPv4 and IPv6 flows. We introduced the nfdump message type to ISC's nmsg framework and based it upon the records from the nfdump tool [2]. A sample nfdump message is shown in Figure 1.

In an input nmsg stream, TreeTop observes the dnsqr messages that contain DNS reply information to each client; when there is a successful response to a DNS query for an

IP address, TreeTop *(a)* stores the query name in a central *domain tree* (an n-ary prefix search tree), *(b)* stores the IPv4 and/or IPv6 address answers in a client-specific *address tree* (a binary prefix search tree), and *(c)* links nodes in the client's address tree to their corresponding nodes in the domain tree. Thus, these data structures store per-client *DNS rendezvous state information* as to which remote IP addresses are known by domain name(s). Subsequently, when TreeTop observes application traffic flows (in nfdump messages in the input stream), it uses the prior rendezvous state information to annotate the nfdump message with source or destination domain names (corresponding to the source and destination addresses), and accumulates per-client traffic counters (in bytes or packets) for those meta-categories as well as for hierarchical sub-categories by domain name.

For the data sets in this work, we collect a pcap trace of DNS traffic on the recursive name servers used by the campus population described in Section III. We also collect NetFlow version 9 flow export data from a campus core router that forwards traffic between the population and the Internet beyond the campus border. To prepare the data sets for TreeTop, we convert the pcap traces of DNS using nmsgtool [3] and convert nfdump using a tool of our own (the nfdump2nmsg script). Subsequently, these sets of messages are merged by timestamp so that the DNS and flow information are properly interleaved to form one coherent input stream in which DNS query responses will be observed prior to their associated application traffic flow data.

```
[2011-06-08 21:52:26.000000000] [7:1 WISC nfdump]
sa: 203.0.113.71
da: 192.0.2.32
sp: 80
dp: 55983
pr: 6
ibyt: 396630
td: 0.064000
snamed: CLIENT_DNS_NAMED
sn: static.ak.facebook.com
ip_version: IPV4
```

Fig. 1: An nfdump message in presentation form, annotated with source domain name (`sn`). The source name is known because the client (with the destination address in `da`) performed an A query that resulted in this peer source address (`sa`) as the answer.

To annotate flows with domain names (or domain suffixes) as classification labels, we utilize two rendezvous-based labeling methods: *direct* and *consensus*.

A. Direct Labeling

Direct DNS rendezvous-based labeling is performed when TreeTop discovers that a given client end-host knows a peer remote IP address by a domain name as the result of a canonical "forward" DNS query to translate that name to an address. In this case, an nfdump record can be annotated because the client involved has used the DNS to resolve the name of its peer; we call this "CLIENT_DNS_NAMED". For instance, the sample nfdump record in Figure 1 has been annotated with "static.ak.facebook.com", as shown in the `sn` (source name) field.

This direct labeling is the most reliable, but it requires the client host to use the same IP address as both the source of its DNS queries and as its local address when exchanging related application traffic. If that is not the case or if TreeTop does not observe a given client's DNS query response that contained a given peer IP address in an answer, we resort to "consensus" labeling.

B. Consensus Labeling

In our observations, the dual-stack hosts use just one of their IP addresses as the source of their DNS queries: a host's IPv4 address. Thus, for IPv6 flows in this work, we often can't perform direct labeling since the client host's IPv6 address is not usually the client address in the corresponding dnsqr messages. To label these flows' sources or destinations, we, instead, use a *consensus-based* approach based on the domain names resolved by other DNS clients in the population studied. If another host or hosts resolved a name to the peer address in question, it is generally agreed, by rough consensus of the population, that this host could also have used the DNS and named the peer (source or destination) similarly; we call this "INFERRED_DNS_NAMED". As can be seen in Figure 2, the `snamed` annotation has been set to `INFERRED_DNS_NAMED` meaning that we have determined the source name by consensus to be the value shown in the `sn` source name annotation, *i.e.,* "*.facebook.com". Note that this is not a fully-qualified domain name (FQDN), but rather a domain suffix; this is because the consensus of the population was that multiple names resolve to the given source address. To improve the classification, it is useful to *sample* and present those names.

```
[2011-06-08 20:14:11.000000000] [7:1 WISC nfdump]
sa: 2001:0db8::face:b00c:0:3
da: 2001:0db8::2:1
sp: 443
dp: 53646
pr: 6
ibyt: 34297
td: 0.064000
snamed: INFERRED_DNS_NAMED
sn: *.facebook.com.
sn_sample: de-de.facebook.com.
sn_sample: check6.facebook.com.
sn_sample: ar-ar.facebook.com.
ip_version: IPV6
```

Fig. 2: An IPv6 nfdump message annotated with a partially ambiguous source name (`sn`) determined by consensus and samples (`sn_sample`) of the resolved FQDNs that matched. The source name is inferred because this client (with the destination address in `da`) was not observed to have resolved a name to this source address (`sa`), but other clients in the locally monitored population resolved at least three names to this source address.

C. Name Sampling

Whether direct or consensus labeling was performed, it is certainly possible that a given flow's source or destination may be known by more than one domain name. For instance,

a service with the name "www.example.com" might also be known as "login.example.com". In such a case, we would like to annotate an nfdump record with a single name for the peer, such as "*.example.com", but also with what caused the ambiguous label. To expose this information, we perform "name sampling."

In Figures 2 and 3, note that the source name `sn` and destination name `dn`, respectively, do not contain an FQDN. Instead they contain ambiguous domain suffixes: "*.facebook.com" and "*." (the DNS root), respectively. This is somewhat unsatisfactory for traffic classification, especially when we only have the DNS root or merely a Top-Level Domain (TLD).

To deal with this situation, we sample and report a number (*e.g.,* 3) of the FQDNs that led to the ambiguity. Specifically, we search the resolved domain names to report a diverse set of FQDNs that differ in the DNS label where the ambiguity occurs, *i.e.,* where the "*" is in the aforementioned examples. Once these samples are gathered (and added as annotations to the nfdump records) we can either (a) compose class labels from multiple domain names or (b) consider whether or not the names are likely associated with one common service. In the former situation, for example, we might relabel an ambiguous "*.com" to "Gmail" if the samples were "mail.google.com" and "gmail.com"; in fact, this is exactly what we do for Internet services that have conflicting labels or TLDs, such as Gmail, in the results in the Section IV. In the latter case, for example in Figure 3, "rss.slashdot.org" and "www.beantownbloggery.com" seem to be unrelated; this confounding ambiguity results from (i) the hosting of unrelated services on the same IPv6 address and by (ii) using dual-stack hosts during the transition to IPv6.

```
[2011-06-08 00:11:10.000000000] [7:1 WISC nfdump]
sa: 2001:0db8::2:1
da: 2001:0db8:fff4::79
sp: 56451
dp: 80
pr: 6
ibyt: 849
td: 0.128000
dnamed: INFERRED_DNS_NAMED
dn: *.
dn_sample: rss.slashdot.org.
dn_sample: www.beantownbloggery.com.
ip_version: IPV6
```

Fig. 3: An IPv6 nfdump message annotated with an ambiguous destination name (`dn`) determined by consensus and samples (`dn_sample`) of the resolved FQDNs that resolved to the same IP address.

D. Port-based Classification

To complement the aforementioned DNS rendezvous-based classifications, we employ traditional port-based application labels from an existing classifier [1] that has been used in prior work. [11] These are: "WWW," "P2P," "FTP," "Streaming," etc., and allow one to distinguish amongst multiple service types that happen to be identified by a single domain name or prefix, such as distinguishing IMAP from HTTPS traffic for Gmail.

The last part of our methodology to assess performance of IPv6 (and IPv4) flows is to calculate the bit rates based on fields already present in the nfdump records: `ibyt` (input bytes) and `td` (time, duration), which we do for all flows having a non-zero duration. We rely on the flow export implementation (in the commercial router) in that we assume sufficient granularity, range, and accuracy of these values for the distributions of rate values used in our performance analyses and results.

III. Empirical Data set

Since we are interested in assessing the performance of Internet services over IPv6 (as compared with IPv4), we select a campus population whose network and client hosts are IPv6-capable. On our campus, there are thousands of dual-stack hosts that reside within 22 IPv4 subnets and one IPv6 subnet and are mixed-use in campus offices and labs.

To gather the traffic traces and input data for this work, we monitor campus traffic at two observation points: *(1)* the campus clients' recursive name servers, and *(2)* a campus core router that forwards traffic between the client hosts and the commodity Internet. We perform full packet capture at the campus domain name servers, and collect non-packet-sampled NetFlow version 9 data at a campus core router. Thus, the payload of the DNS traffic is recorded, but the application traffic payload is neither needed nor recorded. Such monitoring of DNS traffic between the client end-hosts and their recursive DNS service and router-based flow export is feasible within the typical networks of large institutions, enterprises, or Internet service providers. Our interest is in the "canonical" DNS traffic, *i.e.,* the standard DNS traffic expected to precede application traffic that consists of a query by FQDN and an answer containing one or more IP addresses associated with the query name. Because we assess performance using flow bit rates, we use non-packet-sampled flow export data that has complete byte and packet counts as well as start time and flow duration.

Both the DNS and flow export data were collected for the 24 hours of World IPv6 Day. We collected ˜14.2M DNS query responses for 2028 total IPv4 and 23 IPv6 client addresses; of these, ˜114,300 AAAA queries resulted in ˜6,200 NOERROR responses. The client hosts' total traffic was represented as ˜58.8 million IPv4 flows and ˜2.4 million IPv6 flows. The number of active IPv6 and IPv4 client hosts numbered in the the hundreds and is shown in Figures 4a and 4b (Section IV).

IV. Results

In this section we provide a sample assessment of the IPv6 and IPv4 performance for the World-Wide Web traffic (HTTP and HTTPS) involving two popular services, Facebook and Google Mail (Gmail), as observed during the 24 hours of World IPv6 Day (June 8, 2011). We selected these services due to the high number of active local client hosts that utilized them, thus providing a larger sample of hosts and their respective flows for each hour of the day. First we consider how the traffic was classified as being associated with each

service and the differences by IP protocol version, then the active clients, and finally, the flow bit rate as distributions in time series with hourly bins.

A. Service Domain Names

As discussed in Section II, we perform our analyses with scripts that process an nmsg stream of nfdump messages annotated with the domain names that client hosts resolved to the source and/or destination addresses of each flow. The traffic is labeled by domain name (FQDN or domain suffix) and that label is the basis for classification, *i.e.,* Facebook or Gmail.

The IPv4 Facebook traffic is that labeled with one of 950 FQDNs that have the suffix "facebook.com" (and were resolved by this population), such as "www.facebook.com", "developers.facebook.com", "ssl.facebook.com", "login.facebook.com", "upload.facebook.com", etc., including 867 different FQDNs matching "*.channel.facebook.com". The IPv6 Facebook traffic is that labeled with domains including: "www.facebook.com", "developers.facebook.com", "check6.facebook.com", and various others matching "*.facebook.com". This Facebook classification yielded ˜618K IPv4 flows and ˜128K IPv6 flows.

The IPv4 Gmail traffic is that labeled with the following domains: "gmail.com", "mail.google.com", and "www.gmail.com". The IPv6 Gmail traffic is that labeled with the following domains: "gmail.com", "mail.google.com", "www.gmail.google.com". This Gmail classification yielded ˜785K IPv4 flows and ˜463K IPv6 flows.

B. IPv4 and IPv6 Service Asymmetries

We observe that these services exhibit some asymmetry with respect to the specific DNS names resolved to access them over IPv4 versus IPv6. This is apparently due to differences in implementation of the IPv4 and IPv6 portions of the service. For Facebook, we see that the FQDNs matching "*.channel.facebook.com" were not resolved by AAAA queries (but were resolved by A queries for IPv4 addresses), thus it's probable that Facebook Chat was not yet supported via IPv6 and may fall-back to IPv4 on a dual-stack host. Alternatively, it's possible that the Chat service via IPv6 was overloaded on another FQDN or that it used a non-DNS rendezvous mechanism, and thus may be structured differently (with respect to DNS names).

For Gmail, similarly, names such as "imap.gmail.com" and "smtp.gmail.com" were not resolved by AAAA queries; thus, we believe that these Google Mail features (IMAP and SMTP access) were not available via IPv6 at the time. To accommodate this in these results, we select only WWW traffic (by selecting flows with the port numbers for HTTP and HTTPS) so that IPv4 Gmail traffic involving IMAP and SMTP would not be mixed into the performance results for comparison (below).

Such asymmetries or differences in service implementation between IPv4 and IPv6 are a challenge to attempts to directly compare service performance between IPv4 and IPv6. The initial performance analysis presented here assumes that the IPv4 and IPv6 traffic classifications are equivalent for these two services, ignoring the Facebook Chat complication noted above.

C. Active Hosts

In Figures 4a and 4b we plot the total number of active local host IP addresses, IPv4 (solid line) and IPv6 (dashed line) for Facebook and Gmail, respectively. The horizontal axis above the plot in Figure 4a is labeled with the hour of day in local time, five hours west of UTC; the lowest level of activity is at about 0600 and the highest (for these services) during the noon hour, with activity decreasing toward the end of the work day (after 1700 hours). Also, note that there are two regimes: roughly the first 12 hours of World IPv6 Day have low activity and thus fewer flows for which we examine their bit rates; the latter 12 hours have high activity with many more hosts and flows being used to calculate bit rate distributions.

D. Flow Rates

Bit rate distributions for unidirectional flows were calculated for all non-zero duration flows, simply by dividing the number of bits by the flow duration (in seconds). Bit rate is labeled on the vertical axis in the remaining plots in Figure 4, and the horizontal axis is the hour of the day.

Figures 4c, 4d, 4e, 4f are box plots presenting the hourly outbound and inbound flow bit rate distributions for each service. The outbound rates (from local clients) are plotted above the horizontal access, and the inbound rates (to local clients) are plotted below as negative values. The boxes plot the 25th percentile, median, and 75th percentile, and the error bars plot the 1st percentile and the 99th percentile. For each hour bin, the IPv4 flow rates are darker with a solid line and the IPv6 rates are lighter with a dashed line.

Figure 4c shows the Facebook flow rates on a coarse scale, to highlight the error bars extending to the 1st and 99th percentile. Note that the 99th percentile flow in hour 21 (UTC) had a rate of nearly 50 megabits per second; this is the flow shown in Figure 1 (Section II), where we see that it is very short-lived: ˜0.064 seconds. In this plot we also see that the 99th percentile inbound flows from Facebook via IPv4 greatly exceed the rates via IPv6. Furthermore, inbound flow rates from Facebook generally peak higher than those outbound to Facebook; this might be expected for this service when most of its bulk data transfer is content pushed to the client host's web browser application.

Figure 4d shows the Gmail flow rates on a coarse scale. Here we see that the 75th to 99th percentile flow rates outbound and inbound are roughly equal, as are the 99th percentile flow rates for both IPv4 and IPv6; note, though, that there are hours of the day, *e.g.,* hours 1, 3, 5, 7, 9 UTC, when the 99th percentile IPv6 rate exceeded the IPv4 rate, suggesting that the IPv6 service, at least occasionally, provided similar throughput. We also see that there are similar 75th to 99th percentile flow rates outbound to Gmail and inbound from Gmail, suggesting that this bidirectional symmetry is characteristic of a WWW email

service; both sending and receiving messages result in similar workload in HTTPS flows.

In Figures 4e and 4f we show a finer-detail representation, where the vertical axis has been clipped to highlight the interquartile range and median in the box plots of the flow rate distributions; these correspond to the distributions plotted in Figures 4c and Figure 4d, respectively. Figure 4e plots the Facebook flow rates lower than 200K bits per second. Here we see that the flow rate distributions vary with activity level and/or the number of flows. Such measurements, whether due to load or sample size, suggest a direction for subsequent forensic investigation.

Lastly, by examining Figure 4f, we see differing performance between local nighttime and daytime. First, the interquartile range of IPv4 flow rate distributions exhibit higher bit rates than the corresponding interquartile range for IPv6 flows; note the number of active IPv4 and IPv6 hosts are nearly identical (Figure 4b). Second, in the high-activity local daytime, the IPv4 and IPv6 performance seem roughly comparable, until after 1700 hours UTC (noon local time), when the interquartile range for IPv6 flows contains consistently higher values than those for IPv4 flows. This change in IPv6 performance is not correlated merely with a change in the number of active hosts observed.

In these results we employed robust statistics to broadly compare IPv4 and IPv6 performance for two popular services. We find that the number of active hosts (observed via their flows) greatly influences the bit rate distributions. Second, we see evidence of wildly varying near peak (99th percentile) rates in flow export data for a given Internet service. Third, we see that there are regimes in which IPv6 rates are higher and others in which IPv4 rates are higher.

These results show that ostensibly the same services over IPv4 and IPv6 exhibit different performance as measured by the clients' sessions' flow bit rate distributions, meeting our objective to develop an analysis method and presentation by which one could expose performance phenomena and assess IPv6 performance. These observations motivate and guide future work including other visualizations and forensic tasks to determine the root causes of performance anomalies for services on both IPv4 and IPv6.

V. Conclusion

In this paper we present a method to examine the performance of Internet services on IPv6 and IPv4, with which clients rendezvous via the DNS. Our approach is a new application of TreeTop's traffic classification technique that doesn't perform active measurements, doesn't need "insider" knowledge about those services IP addresses, and doesn't require inspection of application traffic payloads that may be encrypted, obscured, or otherwise unavailable. Instead, it relies on low-volume DNS query/response traffic and easily-obtained application transport information from packet headers.

We demonstrate the feasibility of the approach by implementing our method in the TreeTop Framework and a set of assessment tools. We demonstrate its utility by analyzing DNS traces and flow export data gathered from a campus network with an advanced deployment of IPv6 via dual-stack hosts, focusing on their traffic on the World IPv6 Day. A large proportion of traffic involving services running IPv6 is arranged via the DNS, allowing the associated service (*e.g.,* Facebook or Gmail) to be directly identified. While we find that dual-stack IP implementations complicate measurement, our method is able to infer service identities by a consensus of hosts in the monitored population.

These sample results demonstrate how a rendezvous-based technique can be effective in assessing the IPv6 performance of services during their deployment and side-by-side operation over IPv4. We apply robust statistics to observed flow bit rate distributions for Facebook and Gmail traffic and present them as graphs allowing visual detection of performance differences and anomalies. Such capability can inform developers, operators, and users with respect to selection of which Internet Protocol version is likely to yield better performance, making it more likely that Internet services' transition to IPv6 will meet or exceed expectations.

References

[1] CoralReef. http://www.caida.org/tools/measurement/coralreef/, 2008.

[2] NFDUMP. http://nfdump.sourceforge.net/, 2012.

[3] nmsg: network message library. http://rsfcode.isc.org/git/nmsg/, 2012.

[4] I. Bermudez, M. Mellia, M. Munafo, R. Keralapura, and A. Nucci. DNS to the Rescue: Discerning Content and Services in a Tangled Web. In *Proceedings of ACM Internet Measurement Conference (IMC '12)*, Boston, MA, November 2012.

[5] RIPE Network Coordination Centre. IPv6 Measurement - A Compilation. https://labs.ripe.net/Members/mirjam/content-ipv6-measurement-compilation, 2012.

[6] K. Cho, M. Luckie, and B. Huffaker. Identifying IPv6 Network Problems in the Dual-stack World. In *Proceedings of the ACM SIGCOMM Workshop on Network Troubleshooting: Research, Theory and Operations Practice Meet Malfunctioning Reality*, New York, NY, 2004.

[7] K. Claffy. Tracking IPv6 Evolution: Data We Have and Data We Need. *ACM SIGCOMM Computer Communications Review*, 41(3), July 2011.

[8] L. Colitti, S. Gunderson, E. Kline, and T. Fefice. Evaluating IPv6 Adoption in the Internet. In *Proceedings of the Passive and Active Measurement Conference*, Zurich, Switzerland, April 2010.

[9] Google. Protocol Buffers. https://developers.google.com/protocol-buffers/, 2012.

[10] E. Karpilovsky, A. Gerber, D. Pei, J. Rexford, and A. Shaikh. Quantifying the Extent of IPv6 Deployment. In *Proceedings of the Passive and Active Measurement Conference*, Seoul, South Korea, April 2009.

[11] H. Kim, kc claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee. Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices. In *Proceedings of the 4th ACM International Conference on emerging Networking EXperiments and Technologies (ACM CoNEXT 2008)*, Madrid, Spain, December 2008.

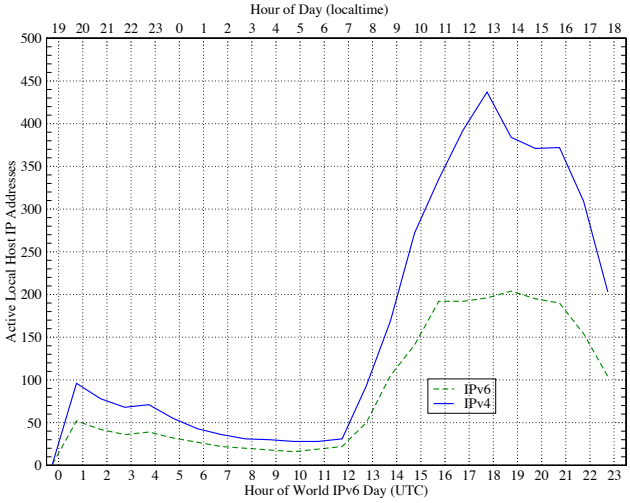
[12] C. Labovitz. Six Months, Six Providers and IPv6. http://ddos.arbornetworks.com/2011/04/six-months-six-providers-and-ipv6/, April 2012.

[13] D. Plonka and P. Barford. Context-aware Clustering of DNS Query Traffic. In *Proceedings of the ACM SIGCOMM / USENIX Eighth Internet Measurement Conference (IMC 2008)*, Vouliagmeni, Greece, October 2008.

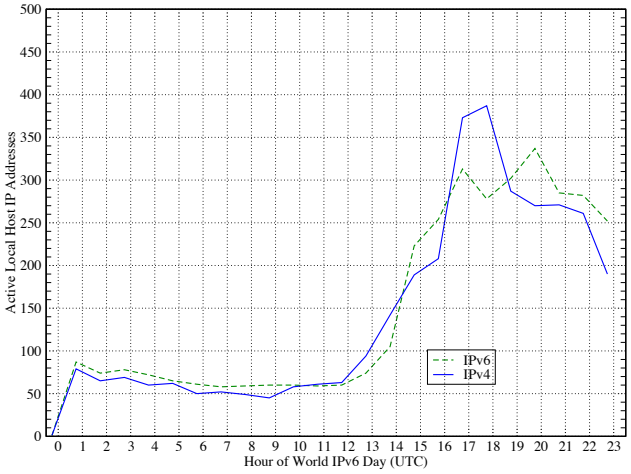
[14] D. Plonka and P. Barford. Flexible Traffic and Host Profiling via DNS Rendezvous. In *Proceedings of the Securing and Trusting Internet Names Workshop (SATIN 2011)*, Teddington, UK, April 2011.

[15] N. Sarrar, G.Maier, B. Ager, R. Sommer, and S. Uhlig. Investigating IPv6 Traffic - What Happened on the World IPv6 Day? In *Proceedings of the Passive and Active Measurement Conference*, Vienna, Austria, March 2012.

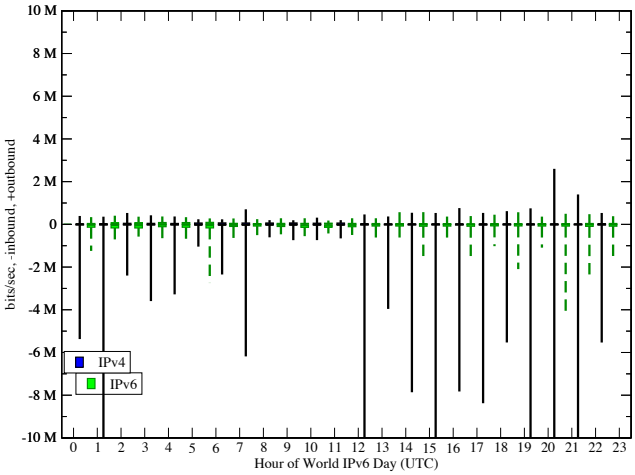
[16] X. Zhou and P.V. Mieghem. Evaluating IPv6 Adoption in the Internet. In *Proceedings of the Passive and Active Measurement Conference*, Boston, MA, April 2005.



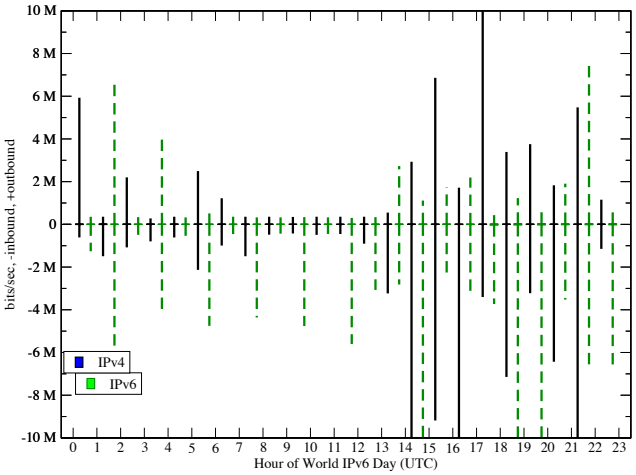
(a) Facebook Active Client IP Addresses



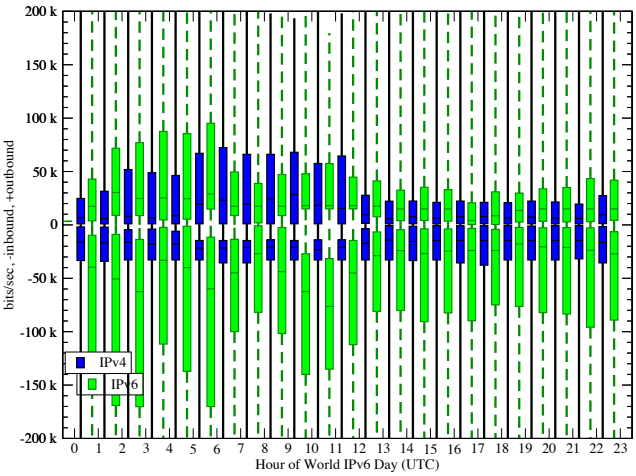
(b) Gmail Active Client IP Addresses



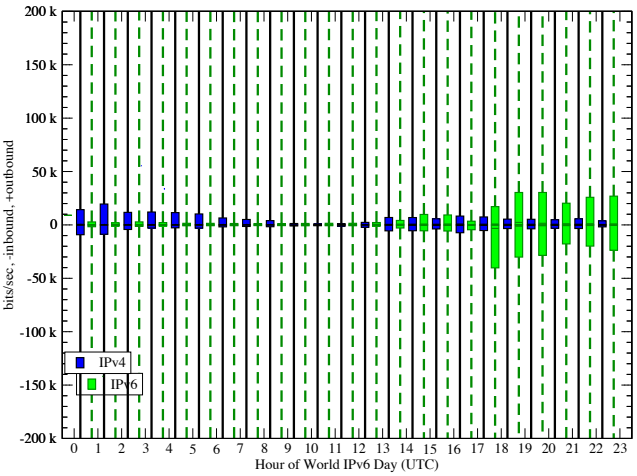
(c) Facebook WWW Flow Bit Rates



(d) Gmail WWW Flow Bit Rates



(e) Facebook WWW Flow Bit Rates (detail)



(f) Gmail WWW Flow Bit Rates (detail)

Fig. 4: Active clients (4a, 4b), flow peak (4c, 4d) and detail (4e, 4f) performance for Facebook and Gmail on World IPv6 Day.