Measuring the Effects of Happy Eyeballs

Vaibhav Bajpai
Jacobs University Bremen, DE
v.bajpai@jacobs-university.de

Jürgen Schönwälder
Jacobs University Bremen, DE
j.schoenwaelder@jacobs-university.de

ABSTRACT

The IETF has developed protocols that promote a healthy IPv4 and IPv6 co-existence. The Happy Eyeballs (HE) algorithm, for instance, prevents bad user experience in situations where IPv6 connectivity is broken. Using an active test (`happy`) that measures TCP connection establishment times, we evaluate the effects of the HE algorithm. The `happy` test measures against ALEXA top 10K websites from 80 SamKnows probes connected to dual-stacked networks representing 58 different ASes. Using a 3-years long (2013 - 2016) dataset, we show that TCP connect times to popular websites over IPv6 have considerably improved over time. As of May 2016, 18% of these websites are faster over IPv6 with 91% of the rest at most 1 ms slower. The historical trend shows that only around 1% of the TCP connect times over IPv6 were ever above the HE timer value (300 ms), which leaves around 2% chance for IPv4 to win a HE race towards these websites. As such, 99% of these websites prefer IPv6 connections more than 98% of the time. We show that although absolute TCP connect times (in ms) are not that far apart in both address families, HE with a 300 ms timer value tends to prefer slower IPv6 connections in around 90% of the cases. We show that lowering the HE timer value to 150 ms gives us a margin benefit of 10% while retaining same preference levels over IPv6.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network monitoring*

Keywords

Happy Eyeballs, IPv6, SamKnows

1. INTRODUCTION

The Happy Eyeballs (HE) algorithm [23,31] (2012) provides recommendations to application developers to help prevent bad user experience in situations where IPv6 connectivity is broken. The algorithm when combined with the default address selection policy [29] (2012), gives a noticeable advantage (300 ms) to connections made over IPv6 as shown in Fig. 1. The HE timer value was chosen during a time when IPv6 brokenness was quite prevalent, which made applications stall for several seconds before attempting a connection over IPv4. For instance, Savolainen *et al.* in [7] (2011) reported browser connection timeouts to be in the order of 20 seconds. A 300 ms HE timer value allowed applications to fast fallback to IPv4 in such situations. The IPv6 brokenness has been largely attributed to failures caused by Teredo [26] and 6to4 relays [22]. Studies [24, 32] (2012, 2010) have shown that even in situations where relays work, Teredo / 6to4 add noticeable latency when compared to native IPv4 and IPv6. With considerable efforts made by the IPv6 operations community, these transition mechanisms appear to steadily decline over the last 5 years. For instance, Christopher Palmer in [3] (2013) announced that Microsoft will stop Teredo on Windows and deactivate its public Teredo servers in 2014. The 6to4 anycast prefix recently has been obsoleted [30] (2015) and future products are recommended to not use 6to4 anycast anymore. Geoff Huston [11] (2016) recently showed that as a consequence, failure rates over IPv6 have dropped from 40% (2011) to 3.5% (2015). In fact unicast IPv6 failure rates have also gone down from 5.3% (2011) to 2% (2015).

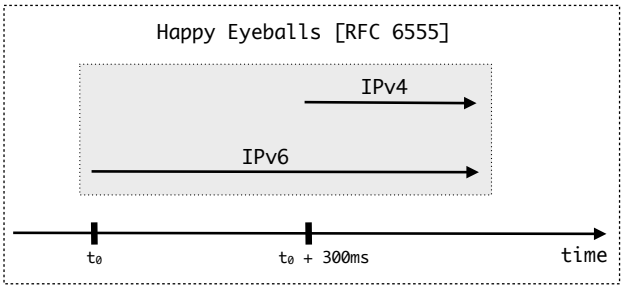

Figure 1: The Happy Eyeballs (HE) algorithm that gives a 300 ms advantage to a TCP connection request over IPv6. The competition runs fair after 300 ms by starting a parallel TCP connection request over IPv4.

Today, IPv6 adoption has reached 12.2% (native) with Teredo / 6to4 at around 0.01% according to Google IPv6 adoption statistics [13] (as of Jun 2016). The Google over IPv6 (whitelist) program no longer exists, but has been replaced by an IPv6 blacklist [19]. In fact, today Google will not return AAAA entries to DNS resolvers where latency over IPv6 is consistently 100 ms or more slower [14] than IPv4. In such a changed landscape, the effect of the HE timer value (300 ms) on the overall experience of a dual-stacked user remains largely unclear. We want to know − What are the percentage of cases where HE makes a bad decision of choosing

ANRW '16, July 16 2016, Berlin, Germany

DOI: http://dx.doi.org/10.1145/2959424.2959429

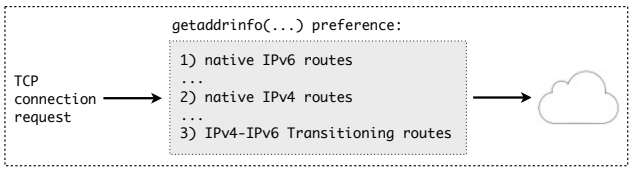

Figure 2: `getaddrinfo()` behavior as dictated by the default destination address selection policy. The policy makes applications iterate over endpoints in an order that prefers an IPv6-upgrade path.

IPv6 when it's slower? Furthermore, in such situations what is the amount of imposition (in terms of latency impact) a dual-stacked user has to pay as a result of the high HE timer value? – This is critical since applications on top of TCP not only apply HE in scenarios where IPv6 connectivity is broken, but also in scenarios where IPv6 connectivity is comparable.

The fragmentation of the algorithm due to the high HE timer value is visible in browser implementations (see § 2.1) today. For instance, Mozilla Firefox (since v15) [15] and Opera (since v12.10) [16] by default use parallel TCP connections over IPv4 and IPv6. Apple (since OS X 10.11 and iOS 9) [2] uses a considerably smaller 25 ms timer value in favor of IPv6 connections. Google Chrome (since v11) [12] is the only browser that sticks to the 300 ms timer value. These values are arbitrarily chosen. We want to empirically determine the right HE timer value that provides the same preference levels over IPv6 as is today but also reduces the performance penalty in situations where IPv6 is slower.

Towards this pursuit, we have developed an active test (`happy`) [19] that measures (see § 2.3) TCP connection establishment times. We deploy this test on 80 geographically distributed SamKnows [18] probes connected to dual-stacked networks (see Fig. 3) representing 58 different ASes to provide diversity of network origins. The test measures against ALEXA top 10K websites. Using a 3-years long (2013 - 2016) dataset of TCP connection establishment times obtained from our metric, we are able to calculate decisions a HE enabled application would have taken and experiment with variations of the HE algorithm to propose changes to it.

Our **contributions** – *a*) We show that TCP connect times to popular websites over IPv6 (see § 3.1) have considerably improved over time. As of May 2016, 18% of websites are faster over IPv6 with 91% of the rest being at most 1 ms slower, *b*) Only around 1% of the TCP connect times over IPv6 were ever above the HE timer value (300 ms), which leaves around 2% chance for IPv4 to win a HE race towards these websites. As such, 99% of these websites prefer IPv6 connections (see § 3.2) more than 98% of the time and *c*) Although absolute TCP connect times (in ms) are not that far apart in both address families, HE with 300 ms timer value tends to prefer slower IPv6 connections (see § 3.3) in around 90% of the cases. A lowering of the HE timer value to 150 ms (see § 3.4) gives us a margin benefit of 10% while retaining same preference levels over IPv6.

2. BACKGROUND

A dual-stacked host with native IPv6 connectivity establishing a TCP connection to a dual-stacked website will prefer IPv6. This is due to the function `getaddrinfo()` that resolves a dual-stacked website to a list of endpoints in an order that prefers an IPv6 upgrade path [29] (2012) as shown in Fig. 2. The dictated order can dramatically reduce the application's responsiveness in situations

NETWORK TYPE	#
RESIDENTIAL	55
NREN / RESEARCH	11
BUSINESS / DATACENTER	09
OPERATOR LAB	04
IXP	01

RIR	#
RIPE	42
ARIN	29
APNIC	07
AFRINIC	01
LACNIC	01

Figure 3: Measurement trial of 80 dual-stacked SamKnows probes as of Jun 2016. The entire metadata for each probe is available online: http://goo.gl/PwD4yN

where IPv6 connectivity is broken. In fact, an attempt to connect over an IPv4 endpoint will only take place when the IPv6 connection attempt has timed out, which can be in the order of several seconds. This degraded user experience can be subverted by implementing the Happy Eyeballs (HE) algorithm [31] (2012) in applications. The HE algorithm recommends that a host, after resolving the DNS name of a dual-stacked website, tries a TCP `connect()` to the first endpoint (usually IPv6). However, instead of waiting for a timeout, which is typically in the order of seconds, it only waits for 300 ms, after which it must initiate another TCP `connect()` to an endpoint with a different address family and start a competition to pick the one that completes first.

The HE algorithm biases its path selection in favor of IPv6 (see Fig. 1) by design. The TCP connection establishment race has been handicapped for the following reasons – *a*) A Carrier-Grade NAT (CGN) [28] establishes a binding for each connection request. Dual-stack hosts by preferring IPv6 connection routes, reduce their contention towards the critical IPv4 address space, *b*) The IPv4 traffic may be billed by Operation Support Systems (OSS). Techniques that help move this traffic to IPv6 networks reduce costs, and *c*) Middleboxes maintain state for each incoming connection request. If dual-stacked hosts prefer IPv6 paths, the load on load balancers and peering links reduces automatically. This reduces the investment on IPv4, and encourages IPv6 migration.

The HE algorithm honors this IPv6 upgrade policy. It is therefore not designed to encourage aggressive connection requests over IPv4 and IPv6, but instead to satisfy the following goals – *a*) The connection requests must be made in an order that honors the destination-address selection policy [29], unless overridden by user or network configuration. The client must prefer IPv6 over IPv4 whenever the policy is not known, *b*) The connection initiation must quickly fallback to IPv4 to reduce the wait times for a dual-stack host in situations where the IPv6 path is broken, and *c*) The network path and destination servers must not be thrashed by mere doubling of traffic by making simultaneous connection requests over IPv4 and IPv6. The connection requests over IPv6 must

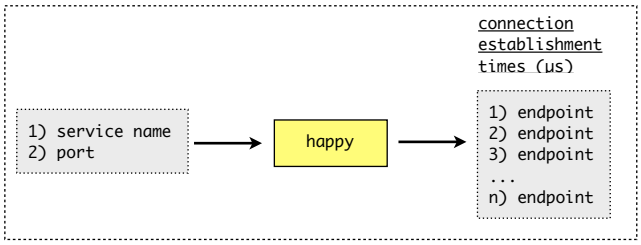

Figure 4: `happy`: A tool to measure TCP connection establishment times. The input parameter is a tuple (service name, port number) and the output is the TCP connection establishment time for each endpoint (measured in microseconds). The tool is available online: http://happy.vaibhavbajpai.com

be given a fair chance to succeed to reduce load on IPv4, before a connection over IPv4 is attempted.

2.1 Browser Implementations

The fragmentation of the algorithm due to the high HE timer value is visible in browser implementations today. For instance, Google Chrome has an implementation of the HE algorithm since v11.0.696.71 [12], which was released in 2011. It uses a 300 ms timer, which is fired after the first TCP SYN request has been sent. Once the timer expires the browser switches to a different address family and starts a competition between IPv4 and IPv6 connection requests to pick the one that completes first.

Mozilla Firefox released its first HE implementation with v7.0. The implementation received multiple bug reports leading to a stable implementation by v15.0 [15]. Firefox by default, unlike Google Chrome follows a more aggressive approach by starting parallel TCP connections to the first endpoints of each address family. However, once one of the connections has been successfully established, the second connection request is not closed by sending a TCP RST, instead the connection request is allowed to continue until exhaustion. Opera, since v12.10 [16] has an implementation similar to that of Mozilla Firefox. It tries simultaneous TCP connections to the first endpoint of each address family and chooses whichever completes first. It remains unclear whether parallel connection attempts can be deemed as a flavor of HE, since the algorithm is designed to honor the IPv6 upgrade policy and therefore does not encourage aggressive connection requests over IPv4 and IPv6. As such, Mozilla Firefox also allows to disable parallel connection attempts by setting a parameter, `network.http.fast-fallback-to-IPv4` to `false`, after which the browser starts preferring IPv6 connection requests with a 250 ms timer value.

Apple Safari prior to OS X 10.11 (since OS X 10.7) [1] used a more hybrid approach. The OS X networking APIs maintained a history of the previously witnessed latencies to each destination along with a combined mean for each address family. Apple Safari instead of using `getaddrinfo()` used these higher level APIs to prefer the fastest connection. Moreover, Apple Safari did not switch to a different address family if no response was received from the first endpoint, instead it tried a TCP connection with the next endpoint in the same address family. This took a long time for an address family switch-over. Apple with OS X 10.11 and iOS 9 has a new simplified HE implementation [2] which uses a 25 ms timer value in favour of IPv6 connections.

These HE timer values are arbitrarily chosen. We want to empirically determine the right HE timer value that provides the same preference levels over IPv6 as is today but also reduces the performance penalty in situations where IPv6 is considerably slower.

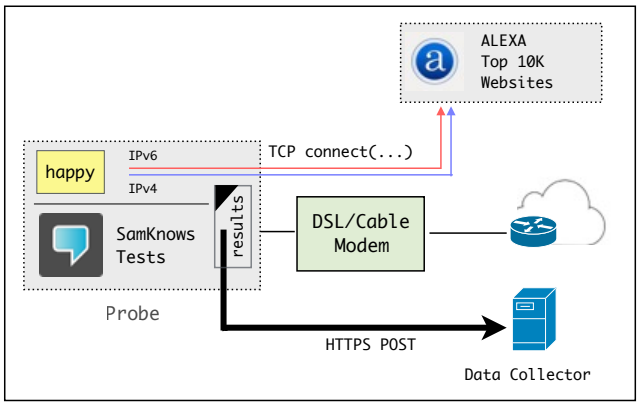

Figure 5: A measurement setup on top of the SamKnows platform. A dual-stacked probe in addition to the standard SamKnows tests, executes a happy test. The happy test runs every hour and measures TCP connect times to top 10K ALEXA websites both over IPv4 and IPv6. The locally collected measurement results are pushed every hour to a data collector.

2.2 Related Work

Jakub Czyz *et al.* in [25] (2014) provide a survey of studies measuring IPv6 adoption on the Internet. We in [19] (2015) have recently provided a short survey on studies measuring IPv6 performance. In this work, we therefore scope our survey to studies measuring HE.

Studies [6, 9, 10] (2011-2012) in the past have analysed HE implementations in Mozilla Firefox 7 and 8, Google Chrome 11, Opera 11 and Apple Safari on OS X 10.7. It was witnessed that Google Chrome (with a 300 ms timer) helps reduce the degraded user experience in situations where a dual-stacked host's IPv6 connectivity is broken. Mozilla Firefox (with the fast-fallback parameter disabled) has an HE behaviour is similar to that of Google Chrome. Apple Safari on OS X 10.7 tends to prefer the fastest connection, but in the process also prefers legacy IPv4 connectivity even where IPv6 connectivity is relatively similar, a situation referred to as *hampering eyeballs*, since it tends to delay the transition to IPv6. These studies however are dated since HE behavior in browser implementations has changed (see § 2.1) with time.

Fred Baker in [21] (2012) describes HE metrics and testbed configurations in a controlled setting to measure how quickly an application can reliably establish connections from a dual-stacked environment. Sebastian Zander *et al.* in [33] (2012) showed that 20% of the hosts had a HE implementation, out of which 75% of the connection attempts preferred IPv6. We show that this preference (due to decreased latencies over IPv6) has increased to 98% today. They observed that HE was used by hosts running Google Chrome (9% of connections), Apple Safari (4%) and Mozilla Firefox (1%). We recently showed [20] (2013) that HE (with a 300 ms timer value) never prefers IPv6 using Teredo except in situations where IPv4 reachability of the destination endpoint is broken. We further showed [17] (2015) that HE (with a 300 ms timer value) prefers a connection over IPv6 to YouTube media servers even when the measured throughput over IPv4 is better. This results in lower bit rates and lower resolutions when streaming a video than can be achieved if streamed over IPv4.

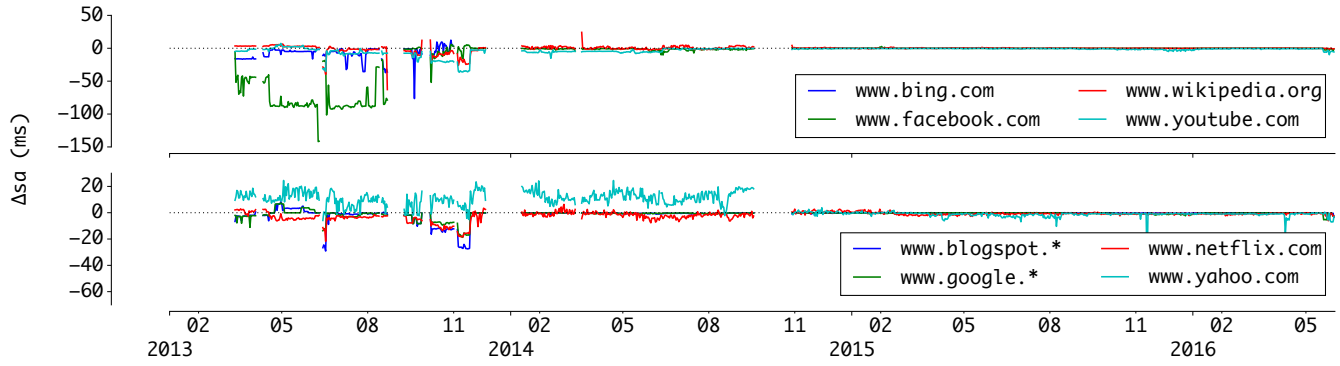

Figure 6: Time series (gaps represent missing data) of absolute difference in TCP connect times to dual-stacked websites. TCP connect times to popular websites over IPv6 have improved over time.

2.3 Methodology

We have developed a `happy` test [19] (2015) that measures TCP connection estalishment times as shown in Fig. 4. The test takes a DNS name and port number of a website as input and returns the TCP connect time for all IP endpoints of the website. It uses `getaddrinfo()` to resolve the DNS name to A and AAAA resource records and non-blocking TCP `connect()` calls to concurrently establish connections to all resolved IP endpoints. It calculates the time it takes for the TCP `connect()` call to complete as a measure of the TCP connect time. As such, the DNS resolution time is not accounted in this measure. This is intentionally done to avoid slow resolvers from biasing the calculation of TCP connect times to a website. We deployed the `happy` test on 80 SamKnows probes (see Fig. 3) connected to dual-stacked networks representing 58 different ASes. To put numbers into perspective, this is more than the number of CAIDA Archipelago (Ark) [27] probes (64 as of Jun 2016) deployed with native IPv6 connectivity today. The `happy` test is executed on the top 10K ALEXA websites and the measurement repeats every hour as shown in Fig. 5. We refer the reader to our previous work [19] (2015) for a more detailed description of our methodology. The rest of the paper presents analysis using a 3-years long (2013 - 2016) dataset of TCP connect times collected from these probes.

3. DATA ANALYSIS

We begin by presenting a terminology. Let u denote a website identified by a URL. We call the time taken to establish a TCP connection towards a website u as $t(u)$. Since we study the impact of accessing websites using different IP protocols, we denote the TCP connect time of u accessed over IP version v as $t_v(u)$. We use *slowness* to adjudicate the performance difference over IPv4 and IPv6. We use both absolute slowness (s_a) and relative slowness (s_r) as described in Eq. 1. Absolute slowness (s_a) is the difference between TCP connect times over IPv4 and IPv6, while relative slowness (s_r) is the fraction of absolute slowness over the observed TCP connect times using IPv4.

$$\Delta s_a(u) = t_4(u) - t_6(u)$$
$$\Delta s_r(u) = \frac{\Delta s_a(u)}{t_4(u)} \quad (1)$$

We use $\hat{\Delta} s_a(u)$ and $\hat{\Delta} s_r(u)$ to represent the median of the sample of $\Delta s_a(u)$ and $\Delta s_r(u)$ values across all probes respectively. The median is taken to ensure measured performance does not get biased by a specific vantage point. This terminology will be used in the rest of the data analysis.

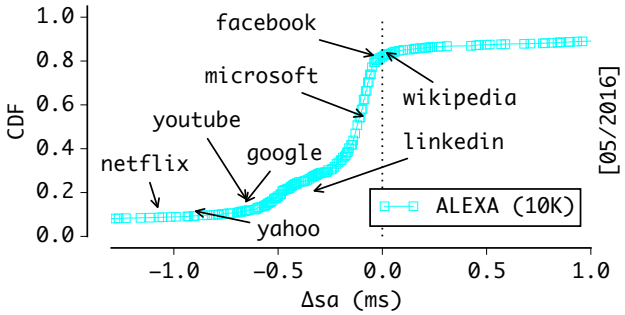

Figure 7: CDF of absolute difference of TCP connect times between IPv4 and IPv6 as of May 2016. 18% of the top 10K ALEXA websites are faster over IPv6 today, although 91% of the rest are at most 1 ms slower.

3.1 Trends

Fig. 6 shows timeseries of absolute slowness, $\hat{\Delta} s_a(u)$ towards popular dual-stacked websites. Note, observations from all google and blogspot websites are clubbed together as `www.google.*` and `www.blogspot.*` since they are served by the same CDN [19] and therefore tend to offer similar performance over each address family. It can be seen that TCP connect times to popular websites over IPv6 appear to have considerably improved over time. It can also be noticed that `www.bing.com` permanently stopped (even though `www.microsoft.com` and `www.office.com` are still IPv6 enabled) providing IPv6 services in Sep 2013. The time series however, does not reveal whether TCP connections over IPv6 are faster (or slower) to these websites today. It can be seen that there is marginal variation in 2016. As such, we aggregated the absolute slowness over May 2016. Fig. 7 shows the absolute slowness, $\hat{\Delta} s_a(u)$ for ALEXA top 10K websites as of May 2016. It can be seen that 18% of the websites connect faster over IPv6 today, although 91% of the rest are at most 1 ms slower. Around 3% of the websites are at least 10 ms slower, with 1% being at least 100 ms slower (not shown) over IPv6. Facebook recently showed [8] (2015) that their news feeds load 30% faster over IPv6 from a US mobile service provider (undisclosed). Our analysis using more diverse vantage points reveals that `www.facebook.com` connects

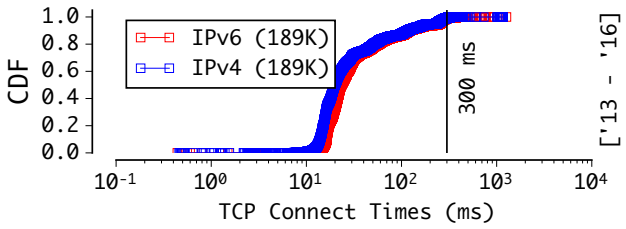

Figure 8: CDF of TCP connect times over IPv4 and IPv6 over the entire 3 years long duration. Only around 1% of the samples exhibit TCP connect times over IPv6 above HE timer value of 300 ms.

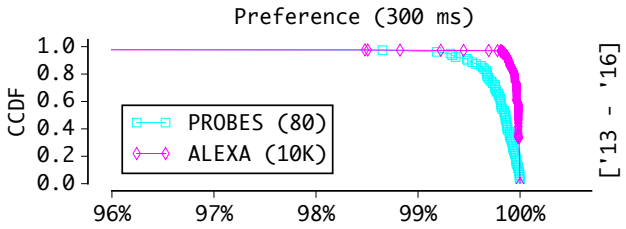

Figure 9: Complementary CDF of TCP connection establishment preference over IPv6 both from source (probes) to destinations (websites). A 300 ms timer values leaves around 2% chance for IPv4 to win a HE race to popular dual-stacked ALEXA websites.

as fast over IPv6 as over IPv4.

3.2 Measuring Preference

Fig. 8 shows the distribution of TCP connect times over IPv4 and IPv6 over the entire 3 years long duration. As can be seen, only around 1% of the samples over IPv6 exhibit TCP connect times above the HE timer (300 ms) value. In fact 90% of the samples over IPv6 are below 100 ms with 82% of the samples below 50 ms. Similarly, 86% of samples over IPv4 are below 50 ms with 75% below 30 ms. Fig. 9 shows the preference calculated over a 3 years long dataset using the HE timer (300 ms) value. It can be seen that during the last 3 years, all probes (sources) preferred IPv6 at least 93% of the time with 99% of probes preferring it more than 98% of the time. Similarly TCP connections over IPv6 to 99% of websites (destinations) were preferred more than 98% of the time. Note, the probe CCDF is invariant of the websites (out of 10K samples), while the website CCDF is invariant of the probes (out of 80 samples). The only probe with less than 98% (93.5%) IPv6 preference is a probe behind a TWC subscriber. The subscriber has a Motorola SB6183 cable modem which is known [4] to drop TCP segments over IPv6 when the TCP timestamp option is set (set by default in Linux). As such, each TCP SYN packet lost can add to a second delay thereby perturbing the IPv6 preference calculation from this vantage point. This is the reason why we prefer to take median aggregation across all probes to remove bias introduced by issues closer to the vantage point. The only website with less than 75% (28.7%) IPv6 preference is `www.qq.com`, where we witnessed that all TCP connect times over IPv6 are more than 200 ms with 63% of the values being more than 300 ms. On the other hand, 88% of TCP connect times over IPv4 are less than 50 ms. This makes about half of the probes to not prefer connecting over IPv6 to this website with 80% of probes having less than 50% preference

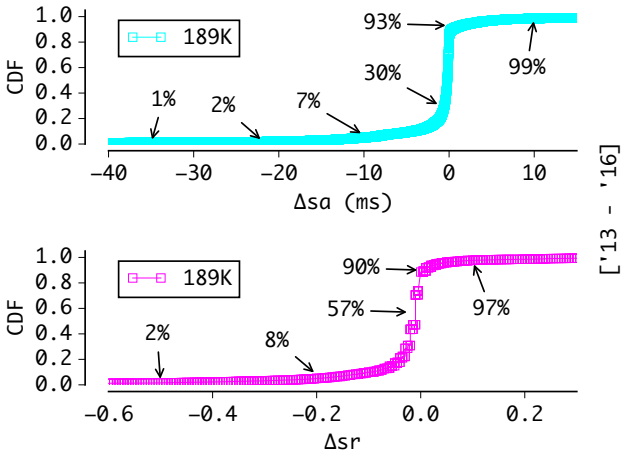

Figure 10: CDF of absolute (above) and relative (below) difference of TCP connection establishment times over IPv4 and IPv6 for situations where HE prefers IPv6 using 300 ms timer value. HE tends to prefer slower IPv6 connection in around 90% of the samples, but absolute TCP connect times are not that far apart from IPv4.

over IPv6. We can conclude that with a HE 300 ms advantage, a dual-stack host tends to use IPv4 connections only around 2% of the time.

3.3 Measuring Slowness

Fig. 10 shows relative slowness $\hat{\Delta}s_r(u)$ for situations where HE prefers IPv6 using the 300 ms timer value. Note, this only includes cases where HE prefers connections over IPv6. The positive values on x-axis represent samples where IPv6 is faster which is around 10% of the total samples. IPv6 is more than 10% faster in around 3% of the samples. On the other hand, IPv6 is more than 2% slower in half of the samples with being more than 20% slower in 8% of the samples. Worse, it is more than 50% slower in 2% of the samples. Fig. 10 also shows the corresponding absolute slowness, $\hat{\Delta}s_a(u)$. It can be seen that around 7% of the samples exhibit TCP connect times that are at least 1 ms faster over IPv6 with around 1% samples that are at least 10 ms faster. On the other hand, around 30% of the samples are at least 1 ms slower with 7% of samples that are at least 10 ms slower. In fact only 2% of the samples are at least 22 ms slower with 1% samples being at least 35 ms slower over IPv6. As such, IPv6 may be slower in 90% of the cases where HE prefers it, but the TCP connect times are not that far apart from IPv4. We know that a 300 ms timer value leaves around 2% chance for IPv4 to win a HE race (see Fig. 9). In 90% of these cases, HE tends to prefer slower IPv6 connection. This shows that the timer value (300 ms) used by the HE algorithm has past its time and is not suitable in today's landscape. Perhaps a lower HE timer value can give the same (99%) preference to IPv6 (see Fig. 9) but not penalise IPv4 in rare cases where IPv6 is (such as `www.qq.com`) slower.

3.4 HE Timer by Preference

We experimented by lowering the HE timer advantage. We know that by using 300 ms HE timer, IPv6 connections to 99% of ALEXA websites are preferred more than 98.6% of the time (see Fig. 9). The idea towards finding a better HE timer value is to control these two parameters (99% websites prefer IPv6 connections 98.6% of

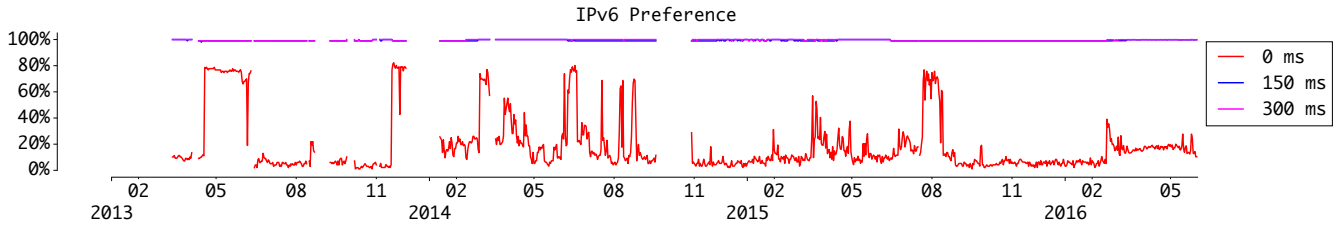

Figure 11: Time series (gaps represent missing data) of IPv6 preference to dual-stacked websites. Disabling HE entirely (0 ms) hampers IPv6 preference. A 150 ms advantage retains same preference levels as the 300 ms timer value.

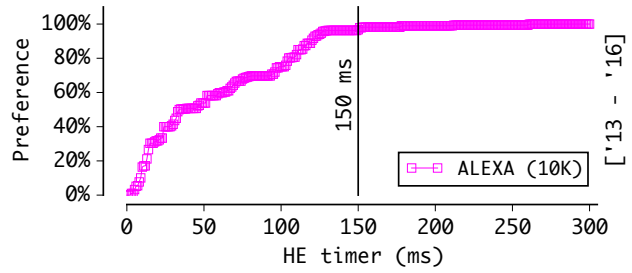

Figure 12: TCP connection establishment preference over IPv6 towards ALEXA websites by varying the HE timer value. A HE timer value of 150 ms allows same 99% of the websites to still prefer connections at least 98.5% of the time.

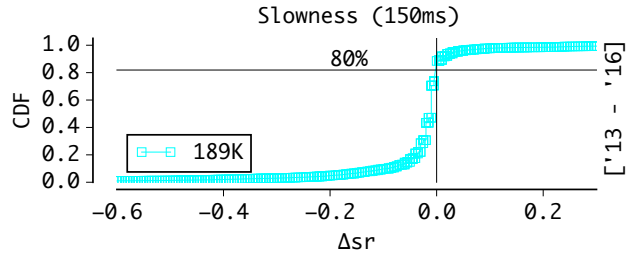

Figure 13: CDF of relative difference of raw TCP connection establishment times over IPv4 and IPv6 for situations where HE prefers IPv6 using 150 ms timer value. By lowering the timer, HE tends to save 10% (18.9K) of the samples from preferring slower IPv6 connections.

the time) and lower the HE timer value to see until when this precedence remains true. This is important because the timer value cannot be lowered to zero (parallel connections over IPv4 and IPv6), since HE must still adhere to the IPv6 upgrade policy (see § 2) to prefer IPv6 paths. Fig. 11 shows that disabling HE entirely by using parallel TCP connections (such as used by Mozilla Firefox and Opera) hampers preference to IPv6 since only 18% of top ALEXA 10K websites are faster (see Fig. 7) over IPv6 today. As such, the timer value by design should give IPv6 a fair chance to succeed to reduce load on IPv4, but at the same time reduce wait time for a dual-stack host in situations where IPv6 is considerably (such as `www.qq.com`) slower. Fig. 12 shows TCP connection establishment preference over IPv6 towards ALEXA websites by varying the HE timer value. Each data point is the 1th percentile preference towards dual-stacked websites. As can be seen, a HE timer value of 150 ms allows same 99% of the websites to still prefer connections at least 98.5% of the time. Fig. 13 shows that lowering HE timer to 150 ms gives us a margin benefit of 10%. A 300 ms timer value preferred 90% of the connections where IPv6 was slow (see Fig. 10) which has been reduced to 80% with a 150 ms timer value. This means 10% (around 18.9K connections with a daily aggregate) of the samples where $t_6(u)$ is at least 150 + $t_4(u)$ ms but less than 300 + $t_4(u)$ ms (because HE timer with 300 ms was preferring IPv6 in these cases) now prefer IPv4 because the timer cuts it early. These may be cases where content over IPv6 is served from a different continent. The new HE timer value is ideal because it comes with no IPv6 preference penalty to observed dual-stacked websites.

4. LIMITATIONS

In this work, we only measured TCP connection establishment times towards websites. As such, the comparison over both address families reflect the performance as seen over TCP port 80 only. Furthermore, around 69K websites [5] (as of Jun 2016) are dual-stacked, however our measurements cover ALEXA top 10K websites only. Finally, the results are biased by the number and location of our vantage points which largely cover US, EU and JP regions. However, in all fairness, it must be noted that a large fraction of IPv6 deployment today is also centered in these regions only, but we concur that the state of IPv6 adoption may change in the future.

5. CONCLUSION

We measured the effects of the HE algorithm. Using a 3-years long trend, we showed that TCP connect times to popular dual-stacked websites over IPv6 have improved over time. As of May 2016, 18% of the top 10K ALEXA websites are faster over IPv6 while 91% of the rest are at most 1 ms slower. A 300 ms timer value therefore leaves only around 2% chance for IPv4 to win a HE race to these websites. In 90% of these cases, HE tends to prefer slower IPv6 connection, although the TCP connect times are not that far apart from IPv4. We showed that a HE timer value of 150 ms provides a margin benefit of 10% while retaining similar IPv6 preference levels for 99% of the dual-stacked websites.

6. ACKNOWLEDGEMENTS

We like to thank all volunteers who host a SamKnows probe to help us in this measurement study. We thank Sam Crawford (SamKnows) and Jamie Mason (SamKnows) for providing us technical support on the SamKnows infrastructure. We also like to thank Dan Wing (Cisco), Andrew Yourtchenko (Cisco) and Steffie Jacob Eravuchira (SamKnows) for reviewing our manuscripts. This work was partly funded by Flamingo, a Network of Excellence project (ICT-318488) supported by the European Commission under its Seventh Framework Programme. This work was also supported by the European Community's Seventh Framework Programme in (FP9/2007-2013) Grant No. 317647 (Leone).

7. REFERENCES

[1] Apple - Lion and IPv6. http://goo.gl/uAPlV8. [Online; accessed 25-January-2016].

[2] Apple and IPv6 - Happy Eyeballs. https://goo.gl/1nzMs6. [Online; accessed 25-January-2016].

[3] Christopher Palmer - Teredo at Microsoft: Present and Future. http://goo.gl/9I65Wy. [Online; accessed 10-February-2016].

[4] Dan Drown - SB6183 dropping IPv6 traffic. http://goo.gl/4zwaHQ. [Online; accessed 03-February-2016].

[5] Dan Wing - AAAA and IPv6 Connectivity Statistics. http://www.employees.org/~dwing/aaaa-stats. [Online; accessed 11-January-2016].

[6] Emile Aben - Hampering Eyeballs: Observations on Two Happy Eyeballs Implementations. https://goo.gl/3xVUIO. [Online; accessed 10-February-2016].

[7] Experiences of host behavior in broken IPv6 networks. http://goo.gl/4NnRiH. [Online; accessed 25-January-2016].

[8] Facebook News Feeds Load 20-40% Faster Over IPv6. http://goo.gl/e5RWHh. [Online; accessed 11-January-2016].

[9] Geoff Huston - Bemused Eyeballs: Tailoring Dual Stack Applications for a CGN Environment. http://goo.gl/LMPc4h. [Online; accessed 10-February-2016].

[10] Geoff Huston - Dual Stack Esotropia. http://goo.gl/N1qUib. [Online; accessed 10-February-2016].

[11] Geoff Huston - Measuring IPv6 Performance. https://goo.gl/n78W1t. [Online; accessed 10-February-2016].

[12] Google Chrome - Revision 85934: Add a fallback socket connect() for IPv6. https://goo.gl/nPhilZ. [Online; accessed 25-January-2016].

[13] Google IPv6 Adoption Statistics. http://goo.gl/i12Qhu. [Online; accessed 11-January-2016].

[14] Google no longer returning AAAA records? https://goo.gl/6Z7gZM. [Online; accessed 11-January-2016].

[15] Mozilla Firefox 15 - Release Notes. http://goo.gl/hA15eu. [Online; accessed 25-January-2016].

[16] Opera 12.10 - Changelog. http://goo.gl/MGsn4K. [Online; accessed 25-Jan-2016].

[17] S. Ahsan, V. Bajpai, J. Ott, and J. Schönwälder. Measuring YouTube from Dual-Stacked Hosts. Passive and Active Measurement Conference (PAM) '15, pages 249–261, 2015. http://dx.doi.org/10.1007/978-3-319-15509-8_19.

[18] V. Bajpai and J. Schönwälder. A Survey on Internet Performance Measurement Platforms and Related Standardization Efforts. IEEE Communications Surveys and Tutorials (COMST) '15, pages 1313–1341, 2015. http://dx.doi.org/10.1109/COMST.2015.2418435.

[19] V. Bajpai and J. Schönwälder. IPv4 versus IPv6 - who connects faster? IFIP NETWORKING '15, pages 1–9, 2015. http://dx.doi.org/10.1109/IFIPNetworking.2015.7145323.

[20] V. Bajpai and J. Schönwälder. Measuring the Effects of Happy Eyeballs. Internet-Draft, July 2013. http://goo.gl/BP6m6G.

[21] F. Baker. Testing Eyeball Happiness. RFC 6556, 2012. https://tools.ietf.org/html/rfc6556.

[22] B. Carpenter and K. Moore. Connection of IPv6 Domains via IPv4 Clouds. RFC 3056, Feb. 2001. https://tools.ietf.org/html/rfc3056.

[23] G. Chen, C. Williams, D. Wing, and A. Yourtchenko. Happy Eyeballs Extension for Multiple Interfaces. Internet-Draft, 2016. https://goo.gl/2UHUuc.

[24] L. Colitti, S. H. Gunderson, E. Kline, and T. Refice. Evaluating IPv6 Adoption in the Internet. Passive and Active Measurement Conference (PAM) '10, pages 141–150, 2010. http://dx.doi.org/10.1007/978-3-642-12334-4_15.

[25] J. Czyz, M. Allman, J. Zhang, S. Iekel-Johnson, E. Osterweil, and M. Bailey. Measuring IPv6 adoption. ACM SIGCOMM '14, pages 87–98. http://doi.acm.org/10.1145/2619239.2626295.

[26] C. Huitema. Teredo: Tunneling IPv6 over UDP through Network NATs. RFC 4380, Feb. 2006. https://tools.ietf.org/html/rfc4380.

[27] kc claffy. The 7th Workshop on Active Internet Measurements (AIMS7) Report. Computer Communication Review (CCR) '16, pages 50–57, 2016. http://doi.acm.org/10.1145/2875951.2875960.

[28] S. Perreault, I. Yamagata, S. Miyakawa, A. Nakagawa, and H. Ashida. Common Requirements for Carrier-Grade NATs (CGNs). RFC 6888 (Best Current Practice), Apr. 2013. https://tools.ietf.org/html/rfc6888.

[29] D. Thaler, R. Draves, A. Matsumoto, and T. Chown. Default Address Selection for Internet Protocol Version 6 (IPv6). RFC 6724, Sept. 2012. https://tools.ietf.org/html/rfc6724.

[30] O. Troan and B. Carpenter. Deprecating the Anycast Prefix for 6to4 Relay Routers. RFC 7526, May 2015. https://tools.ietf.org/html/rfc7526.

[31] D. Wing and A. Yourtchenko. Happy Eyeballs: Success with Dual-Stack Hosts. RFC 6555, 2012. https://tools.ietf.org/html/rfc6555.

[32] S. Zander, L. L. H. Andrew, G. J. Armitage, G. Huston, and G. Michaelson. Investigating the IPv6 Teredo Tunnelling Capability and Performance of Internet Clients. Computer Communication Review (CCR) '12, pages 13–20, 2012. http://doi.acm.org/10.1145/2378956.2378959.

[33] S. Zander, L. L. H. Andrew, G. J. Armitage, G. Huston, and G. Michaelson. Mitigating Sampling Error when Measuring Internet Client IPv6 Capabilities. Internet Measurement Conference (IMC) '12, pages 87–100, 2012. http://doi.acm.org/10.1145/2398776.2398787.