Does Anycast Hang Up on You (UDP and TCP)?

Lan Wei and John Heidemann, *Fellow, IEEE*

***Abstract*—Anycast-based services today are widely used commercially, with several major providers serving thousands of important websites. However, to our knowledge, there has been only limited study of how often anycast fails because routing changes interrupt connections between users and their current anycast site. While the commercial success of anycast CDNs means anycast usually works well, do some users end up shut out of anycast? In this paper, we examine data from more than 9000 geographically distributed vantage points (VPs) to 11 anycast services to evaluate this question. Our contribution is the analysis of this data to provide the first quantification of this problem, and to explore where and why it occurs. We see that about 1% of VPs are *anycast unstable*, reaching a different anycast site frequently (sometimes every query). Flips back and forth between two sites in 10 s are observed in selected experiments for given service and VPs. Moreover, we show that anycast instability is *persistent* for some VPs—a few VPs never see a stable connections to certain anycast services during a week or even longer. The vast majority of VPs only saw unstable routing toward one or two services instead of instability with all services, suggesting the cause of the instability lies somewhere in the path to the anycast sites. We point out that for highly unstable VPs, their probability to hit a given site is constant, suggesting load balancing might be the cause to anycast routing flipping. Finally, we directly examine TCP flipping and show that it is much rarer than UDP flipping, but does occur in about 0.15% (VP, letter) combinations. Moreover, we show concrete cases in which TCP connection timeout in anycast connection due to per-packet flipping. Our findings confirm the common wisdom that anycast almost always works well, but provide evidence that a small number of locations in the Internet where specific anycast services are never stable.**

***Index Terms*—Computer network reliability, content distribution networks, domain name system, IP networks, network topology.**

I. Introduction

A CONCERN about anycast is that BGP routing changes can silently shift traffic from one site to another—we call this problem *potential anycast instability*. Without centralized control, such a shift will cause the connection to break. Yet this problem cannot possibly be widespread—anycast's wide use across many commercial providers suggests it works well. This observation is supported by multiple studies that have shown routing changes interrupt connections rarely [1], [10], [12]. Internet applications already must include some form of recovery from lost connections to deal with server failures and client disconnections, so anycast instability should not be a problem provided it is infrequent. Moreover, most Web connections are only active for short periods of time, so the fraction of time when a route change will directly affect users is small.

In addition to BGP changes that may cause anycast instability, *load balancers* are widely used in many places in the Internet. While load balancing at the destination is usually engineered to provide stable destinations for each client, load balancing in the wide-area network is not always so careful. Prior work has observed that WAN-level load balancing can disrupt RTT estimation [14]; we believe it can also result in anycast instability. While such problems may be very rare (affecting only users that cross a specific link, and perhaps only certain traffic types), such effects in the WAN are particularly concerning because they happen *outside the control* of both the user and the service provider. It is extraordinarily difficult to detect problems that affect a tiny fraction of users, while being able to provide service to the vast majority of users. With billions of users, even a fraction of percent is a serious problem.

This paper provides the first *quantitative evaluation of the stability of anycast routing*. While very rare, we find that about 1% of combinations of vantage point and anycast service are *anycast unstable*, frequently changing routes to different sites of a service (Section IV-B). We call these route changes *anycast flips*, and they can disrupt anycast service by losing state shared between the VP and the server with which it was previously communicating.

This result follows from the study of 11 different anycast deployments, each a global *Root DNS Letter* with an independent architecture, with sizes varying from 5 to about 150 anycast sites (each a location with its own anycast catchment and one or more servers).

Our second contribution is to demonstrate the *severity and potential causes of* of route flips through a number of measurement studies. This study provides a *broad* view by examining all combinations of about 9000 VPs and 11 anycast services. We use several measurement methods to examine how frequent flips are, proving they often flip between anycast sites in tens of seconds (Section IV-E), and strongly suggesting they may flip more frequently, perhaps every packet as shown in our data (Section IV-F). We also find that anycast instability is often continuous and persistent: 80% of unstable pairs of VP and anycast services are unstable for more than a week. For a few, these problems are very long lasting: 15% VPs are still unstable with some service even 8 months later (Section IV-C).

Manuscript received October 30, 2017; revised January 7, 2018; accepted January 19, 2018. Date of publication February 12, 2018; date of current version June 8, 2018. Lan Wei and John Heidemann's work is partially sponsored by the Department of Homeland Security (DHS) Science and Technology Directorate, HSARPA, Cyber Security Division, BAA 11-01-RIKA and Air Force Research Laboratory, Information Directorate under agreement number FA8750-12-2-0344 and via contract number HHSP233201600010C. The associate editor coordinating the review of this paper and approving it for publication was M. Mellia. *(Corresponding author: Lan Wei.)*

The authors are with the Information Sciences Institute, University of Southern California, Marina del Rey, CA 90292 USA (e-mail: weilan@isi.edu; johnh@isi.edu).

Digital Object Identifier 10.1109/TNSM.2018.2804884

We show that *anycast instability is specific to paths*: with almost all VPs with instability seeing it in only a few services (one to three), not all 11 (Section IV-D). Although we cannot definitively know the root causes of anycast instability, we do show from our measurement that certain (VP, service) pairs flip very frequently, likely every packet (Section IV-F).

In earlier work we suggested a possible explanation is load balancers on WAN links [25]. Here we report evaluation of both UDP and TCP connections (Section IV-G), showing that UDP flipping is rare, occuring in about 1% of (VP, service) pairs, and that TCP flipping is rarer still, with regular TCP flipping occurring in for only about 15 pairs (less than 0.15% VPs). However, when TCP flipping occurs, more than 20% of DNS queries over TCP do not complete.

Our results have three important implications. First, *anycast almost always works without routing problems*: for 99% of combinations of VP and anycast service, routes are stable for hours, days, or longer. With multiple successful commercial CDNs, this result is not surprising, but it is still important to quantify it with a clear, public experiment. Second, we show that *anycast does not work for **all** locations, and a few VP/anycast combinations (about 1%) see persistently UDP route instabilities* with paths flipping frequently. Third, TCP anycast flipping does occur in certain cases resulting in connections terminating, although not happening for every anycast service in our data. We also show that some users behind a per-packet balancer might be terribly affected by one anycast service, but not by other anycast services. This result suggests that commercial anycast CDNs and those providers who want to provide service to *all* users may wish to study locations that have anycast unstable routes and investigate ways to reduce this instability.

This work is originally published in Network Traffic Measurement and Analysis Conference 2017, co authored by Wei and Heidemann [25]. Here we add new measurement and analysis of TCP (Section IV-G).

II. Anycast Routing Instability

In IP anycast, an anycast service uses a single IP address, and a user's traffic is directed to a "nearby" site selected by BGP routing. Typically, "nearby" is defined by the length of the path in AS hops, but BGP supports multiple mechanisms that allow service operators and ISPs to impose policy decisions on routing (for details, see an overview [3]). Policies can be political (this path is only for academic traffic), commercial (send more traffic to the less expensive peer), or technical (load balance across these links).

Anycast flips can be a problem because changes in routing shift a client to a new server without any notification to either. If the client and server have some shared states, such as active TCP connections, these will break because of a TCP reset and need to be restarted.

CDNs often keep persistent TCP connections open to clients when sending streaming media such as video. While applications need to be prepared for unexpected termination of TCP connections, a route flip will greatly increase latency as the problem is discovered and a new connection is built.

TABLE I
Datasets Used in This Paper: **a, b, c**: Datasets Observing Catchments From UDP-Based CHAOS Queries; **d**: BGP Routing Updates; **e, f**: Datasets Observing Catchments From TCP-Based CHAOS Queries; **g**: Traceroute Datasets

	start	**duration**	**VPs** *number*	**probing** *interval*	**targets** (root letters)
a	**2015**-12-05 00:00	7 days	9184	240s	`--CDEFG-IJKLM`
b	**2016**-08-01 00:00	7 days	9254	240s	`A-CDEFG-IJKLM`
c	**2017**-01-29 21:00	30 minutes	100	20s	`---D---------`
d	**2016**-08-01 00:00	7 days	192 peers	cont.	`A-CDEFG-IJKLM`
e	2017-07-08 00:00	17 hours	100 per root	20m	`ABCDEFG-IJKLM`
f	2017-07-19 22:30	17 hours	15	20m	`ABCDEFG-IJKLM`
g	2017-07-19 22:30	17 hours	15	30m	`ABCDEFG-IJKLM`

Most DNS today is sent over UDP, but zone transfers use TCP, and recent work has suggested widespread use of TCP and TLS for DNS privacy [11], [27]. For DNS, a route flip results in a much larger response time. For a CDN or video streaming, it might result in playback stalls and "buffering" messages.

A key factor affecting the degree of impact that an anycast flip has on the client is how long its TCP connections are open, plus how long they are active. For video, connections may be open for many tens of minutes, during which they may be active around 10% of the time. For DNS, connections may be open for tens of seconds and active briefly, but multiple times.

III. Methodology

This section explains the essential features of the datasets and the methodology we use to analyze the dataset.

A. Sources and Targets

Our paper uses five CHAOS query datasets listed in Table I. All use the RIPE Atlas infrastructure [16]. Two are existing public datasets they collect via UDP queries [17], the third is an additional publicly-available dataset we collect to improve time precision [21]. We also collect another two datasets [18], [19], both using RIPE Atlas VPs to query DNS letters via TCP. Additionally, we use RouteViews dataset [23] to check the updates of BGP routing table. We also use Paris-traceroute records from RIPE Atlas VPs [20] to study routing paths.

The target of RIPE data collection is all 13 Root DNS Name Servers (or *Root Letters*), shown in Table II. Of these services, our study considers all Root Letters that use anycast at the time of measurement. We omit A-Root from the 2015 dataset, because at that time it was only probed every 30 minutes. We omit B- and H-Root from both datasets because, at these times, B is unicast and H uses primary/secondary routing. Root Letters are operated by 12 organizations and use 13 different deployment architectures, and a wide range of sites (5 to 144), providing a diverse set of targets.

We actively probe each anycast letter, sending queries from more than 9000 RIPE Atlas probes, embedded computers we call *Vantage Points* (VPs). VPs are geographically distributed around the world, although North Africa and China are only sparsely instrumented. Our results may underrepresent anycast problems in these two areas.

TABLE II
TARGETS OF OUR STUDY ARE MOST OF 13 ROOT LETTERS, WITH THEIR REPORTED NUMBER OF SITES [22], AND HOW MANY SITES WE OBSERVE IN EACH DATASETS

letter	operator	sites reported	observed *2015*	observed *2016*
A	Verisign	5	—	5
C	Cogent	8	8	8
D	U. Maryland	87	63	71
E	NASA	71	74	66
F	ISC	59	51	48
G	U.S. DoD	6	6	5
I	Netnod	49	51	56
J	Verisign	98	65	89
K	RIPE	33	32	40
L	ICANN	144	110	118
M	WIDE	7	6	6

We use active queries rather than passive analysis of BGP because we are most concerned about frequent flipping (Section IV-E) and prior work has shown that BGP changes are relatively infrequent, often hours or more apart [13]. We also use BGP data from RouteViews to confirm the BGP stability (Section IV-F).

The targets of our queries are 11 root-letters which are operated by 10 organizations, listed in Table II. Different letters have different deployment architectures, and they have different number of anycast sites. Although we study most letters, we do not see all anycast sites of each letter. We sometimes miss sites because RIPE Atlas VPs are sparse in some parts the world (particularly Africa), and because some anycast sites are local-only and so will be seen only by a VP in the same AS. Fortunately, answers to our research questions do not require complete coverage.

We do not directly study anycast-based CDNs. Like root letters [22], CDNs vary widely in size, from ten or tens of sites [2], even approaching 1000 sites [4], although hybrid architectures may use a subset of all sites [8]. In Section IV-D and Section IV-F we show that instability often results from load balancer in the middle of the network, so our results likely apply to CDNs.

B. Queries From RIPE Atlas

Each VP queries each Root letter every 4 minutes (except for A root in the 2015 dataset). The query is a DNS CHAOS class, for a TXT record with name `hostname.bind`; this query is standardized to report a string determined by the server administrator that identifies server and site [26]. Queries are directed at anycast IP addresses that are served by a specific Root Letter. (RIPE Atlas queries each specific IP address of the root-servers, allowing us to study each letter as an independent service.)

The above query results in a record listing the time, the VP's identity, and the response to the CHAOS query (or an error code if there is no valid response). The responses are unique to each server in use. There is nothing to prevent third parties from responding on an anycast service address, and we see evidence of that in our data. We call responses by third parties other than the operator *spoofed*.

We map the CHAOS responses we see to the list of sites each self-reports [22], following practices in prior studies [7]. While CHAOS responses are not standardized, most letters follow regular patterns, and the same pattern from many different VPs gives us some confidence that it is valid. For example, if lax1a.c.root-servers.org and lax1b.c.root-servers.org regularly appear, we assume city.c.root-servers.org is C-Root's pattern.

CHAOS responses usually identify specific servers, not sites. Some letters have multiple servers at a given site. Continuing the above example, lax1a.c.root-servers.org and lax1b.c.root-servers.org suggest C-root has two servers 1a 1b at the lax site. Not all letters identify servers inside large sites, but all provide unique per-site responses.

We study flipping between *sites* and ignore changes between *servers* in each site, since operators can control server selection if they desire (perhaps with a stateful or consistent load balancer), but not changes between sites.

We detect spoofed strings as those that do not follow the pattern shown by that letter, those seen only from a few VPs in specific networks, and because spoofers typically reply with very low latency (a few ms instead of tens of ms). Typically, about 0.7% of VPs see spoofed replies, and those VPs *always* see the same replies, suggesting their ISPs intercept DNS. While we work to remove spoofed chaos replies from our data, our methods do not prevent a malicious party from generating correct-looking replies.

C. Other Sources: High Precision Queries, TCP and BGP

In addition to the standard RIPE Atlas probes of Root letters, we also request our own measurements at a more frequent time interval via UDP, later perform experiment via TCP, and gathered BGP information to understand routing.

For high precision queries we use the RIPE Atlas infrastructure, but select 100 VPs of interest, based on those that see anycast instability. For these VPs, we request that they query D-Root every 60, 70, 80, and 90 s for 30 minutes. Although RIPE Atlas limits queries to once per minute, by scheduling concurrent measurement tasks on the same VPs we can get results that provide precision approaching 20 s or even less from the unevenly-distributed queries.

For the two datasets **e** and **f** in Table I collected from TCP queries, we use the same RIPE Atlas infrastructure. We select 100 different VPs of interest for the first dataset, based on those that see the top most UDP anycast instability. For these VPs, we request that they query all DNS root letters every 20 minutes for 17 hours. For the second dataset, we select 15 VPs based on the previous datasets, and repeat the experiment.

To rule out BGP as the cause of flipping we use data from RouteViews [23] from 2016-08-01 to 2016-08-07. Although the peers that provide routing data are in different locations than our RIPE VPs, the multiple RouteViews peers provide a guiding picture of Internet routing for BGP.

D. Detecting Routing Flips

We define a *routing flip* as when a prior response for a VP's query indicates one site, and the next response indicates a different site. For missing replies, we assume that the VP

is still associated with the same site as in the prior successful reply.

Most VPs miss ten or fewer replies per day and so loss does not change our results. About 200 VPs (around 2%) miss all or nearly all replies; we exclude these from our datasets.

E. Identifying Load Balancers With Paris Traceroute

Although our end-to-end measurements suggest flipping somewhere on the path, these tests do not identify specific load balancers. We therefore use traceroutes [20] to identify potential locations of load balancers that may cause flipping. We confirm the presence of multipath routing by detecting changes to paths observed at a list of continuing timestamps in dataset **g** in Table I. If such appearance keeps a certain proportion in any rolling time window, such as a half or two thirds in any one-hour window or two-minute window, we consider that a load balancer shapes the traffic this way. The load balancer is likley hop before this intermittent hop, or in a private network before this hop.

We check whether a load balancer alters the final destination of packets. Load balancers may send packets over different paths, but if they the packets end up at the same destination, problems may be limited. On the other hand, if a load balancer sends alternative packets to different sites, the connection-oriented communication will not be successful (TCP will fail with a reset). We look up the geolocation of the penultimate routers of the final destinations to see whether the destination changes, by checking its IP address when they are public ones. The root servers usually drops ICMP packets, and the last hops in the traceroute are often the penultimate routers of root servers.

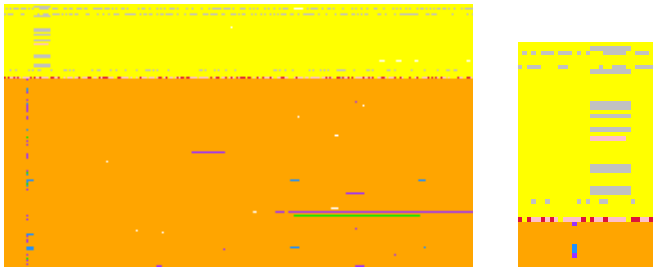
IV. Evaluation

We next apply our analysis to evaluate anycast stability. We first identify examples of routing stability, then quantify how often it happens, how long it persists, and then discuss possible causes for the instability.

A. What Does Anycast Instability Look Like?

We first look to see if there is *any* anycast instability. While successful anycast-based CDNs suggest that most users will be stable, but perhaps a few are less fortunate. We look at the data from RIPE Atlas to the Roots Letters as described in Section III, looking for VPs that change sites between consecutive queries.

Before looking at stability statistics, we first show a sample of direct observations to characterize typical anycast stability. We selected 140 VPs from the dataset for C-Root in 2015 dataset and plotted which sites they access for each 40-minute period of the week-long dataset. To better present the data, we select C-root because we can assign each of its 6 sites a unique color (or shade of gray). We choose 140 VPs that mainly associate with the MAD and ORD sites as representative of all sites for C. To show our full week of data on the page, we report only the last site selected by each VP in each 40 minute period. (This summarization actually reduces the apparent amount of changes.)



(a) 140 VPs for 1 week

(b) The top left: 50 VPs for 20 hours

Fig. 1. Sites accessed by 140 VPs: each row represents a VP, and each column represents a 40-minute period, and the colors show what site that VP reaches for C-Root. (yellow: MAD (Madrid), orange: ORD (Chicago), gray: CDG (Paris), red: BTS (Bratislava), pink: FRA (Frankfurt), purple: IAD (Herndon), blue: JFK (New York), green: LAX (Los Angeles), white: no response). Dataset: 2015.

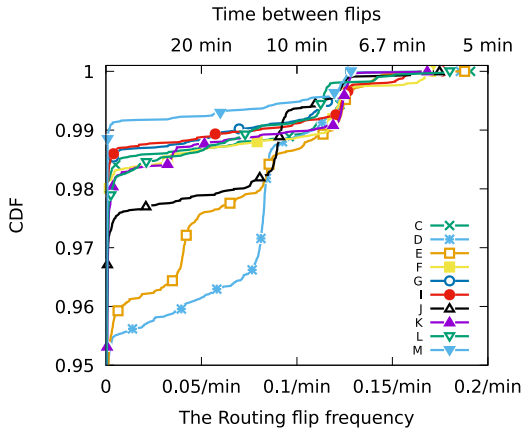
Figure 1 is a timeseries showing which sites 140 VPs reach over 1 week in the 2015 dataset. Each row is a VP, and each column is a 40-minute period, and color indicates the currently active catchment (or white if no reply). Figure 1b zooms in on the top left 50 VPs for 20 hours. We see similar results for other letters, and for other datasets.

Overall stability: Figure 1 shows very strongly that anycast usually works well—*most VPs are very stable*. Many of these VPs access one site, with most of top 39 VPs for MAD (the yellow band), while the most of the bottom 101 VPs for ORD (orange). We expect general stability, consistency with wide, successful use of anycast.

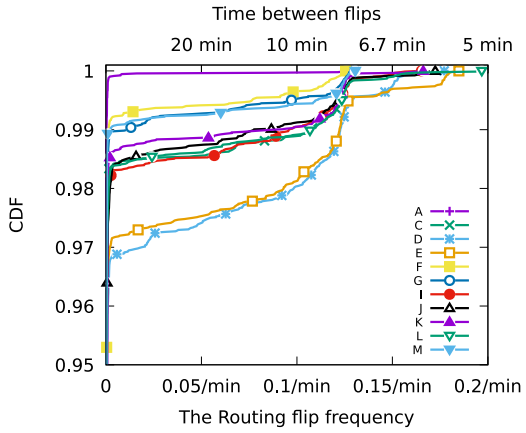
While most VPs are stable, we next look at three groups of routing flips as shown by color changes in the figure.

Groups of Routing Flips: These routing changes happen and affect many VPs at the same time; if these are occasional they are benign. On the left of Figure 1a, there is a tall vertical "stripe" affecting many VPs for ORD (orange), and another wider strip affecting many of the VPs for MAD (yellow). In each of these cases we believe there was a change in routing in the middle of the network that affected many (but not all) users, changing them from ORD or MAD to blue JFK. In both cases, the routes changed back fairly quickly (after 36 minutes for MAD-CDG-MAD, and 6 hours for ORD-LAX/IAD/JFK-ORD). Group flips that happen occasionally will require TCP connection restarts, but two events in two weeks will have minimal impact on users. These kind of normal routing changes reflect anycast automatically re-routing as ISPs reconfigure due to traffic shifts or link maintenance.

Individual, Long-term Changes: Other times we see individual VPs change their active site, perhaps reducing latency. For example, the bottom-right of Figure 1a, about 10 VPs change from ORD to IAD or LAX (orange to purple or green), and stay at that site for the remainder of the period, about three days. Again, we believe these changes in routing represent long-term shifts in the network, studied elsewhere [24]. Because these changes are infrequent, long-term shifts cause minimal harm to users, and sometimes they may help if they result in a lower latency path. They may also represent routing changes by operators to re-balance load on sites.



(a) Week of 2015-12-05.



(b) Week of 2016-08-01.

Fig. 2. Cumulative distribution of mean flip time for each VP, broken down by anycast service. (Note the *y*-axis does not start at zero).

Frequent Routing Flips: Finally, we see a few cases where *VPs see persistent routing flips*, suggesting that, *for them, anycast will not work well*. In Figure 1a we see four cases: three VPs flip between MAD and CDG and back (gray and yellow, all in the yellow band), and one VP alternates between FRA-BTS-MAD (pink, red and yellow, shown at the boundary of the yellow and orange bands). This behavior continues throughout the week. While the VPs sometimes reach the same site in consecutive measurements, this kind of frequent flipping greatly increases the chances of breaking TCP connections.

B. Is Anycast Instability Long Lasting, and for How Many?

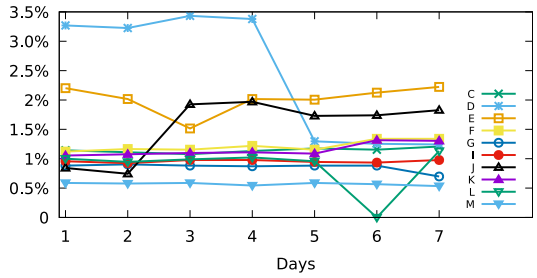
We have seen some unstable users (Figure 1a), but *how many* are unstable? To answer that question, we must first consider how long instability lasts.

To evaluate the stability of each VP, we compute the mean duration that VP is at each site, then report the cumulative distribution for each root letter for 2015 and 2016 dataset in Figure 2.

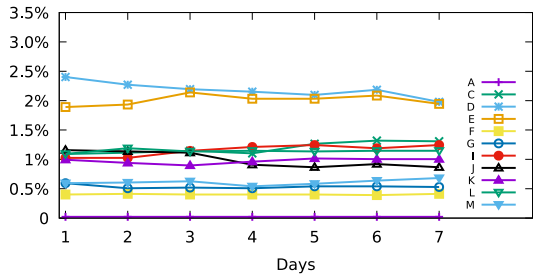
The result confirms the prior observation that *overall, anycast is very stable for most VPs*. The *y*-axis of the CDFs (Figure 2) does not start at zero, and we see that 90% of VPs see two or fewer changes for all Root Letters we study but one

TABLE III
NUMBER OF FLIPS PER VP, FOR EACH ROOT LETTER, FOR THE WEEK OF 2016-08-01

Root Letter	mean	(sd)	flips (% VPs) =0	≤ 1	≤ 2	≤ 3
A	2.0	(21.2)	23%	25%	98%	98%
C	16.7	(133.2)	80%	80%	90%	91%
D	32.4	(188.5)	50%	52%	89%	91%
E	30.9	(190.0)	66%	69%	90%	90%
F	7.1	(81.8)	81%	82%	91%	92%
G	11.3	(93.5)	12%	12%	51%	52%
I	17.2	(134.3)	72%	76%	89%	90%
J	15.6	(128.3)	69%	72%	90%	92%
K	14.5	(124.8)	76%	78%	86%	86%
L	17.1	(137.8)	71%	75%	90%	92%
M	8.9	(98.1)	90%	91%	95%	95%



(a) Week of 2015-12-05.



(b) Week of 2016-08-01.

Fig. 3. The percentage of anycast unstable VPs for each day in a week.

(Table III). In fact, A-root **barely** saw any route changes for any VPs in the week starting from 2016-08-01. (Figure 2b).

Stability means *most VPs are in one catchment for a long time*. Table III shows overall statistics per letter, for each dataset. Most VPs are very stable.

However, it also confirms *a few VPs experience frequent routing flips*. We define a VP as *anycast unstable* when the mean time between flips is 10 minutes or less. We select this threshold because it is slightly longer than two measurement intervals (each 4 minutes), tolerating some measuring jitter. Based on the threshold of 10 minutes, we see that about 1% of VPs are anycast unstable for almost all Root Letters for both 2015 and 2016 datasets. One exception is A-root, who shows high stability in the 2016 dataset. This analysis suggests that, at least in these datasets, some VPs will have a difficult time using anycast and may experience TCP connection breaks.

To confirm these results are typical, Figure 3 examines the fraction of anycast unstable VPs each day. Most VPs consistently have about 1% of VPs as unstable, although there is

some variation in a few letters (for example, D has 3.2% in part of Figure 3a).

The precision of results in Figure 2 is limited by the 4 minute frequency of basic RIPE observations. We later return to this question with more frequent request rate and analysis to suggest that actually flipping rates are much higher than every 4 minutes (Section IV-E), and likely every packet (Section IV-F).

C. Is Anycast Instability Persistent for a User?

We have shown that about 1% VPs are anycast unstable, and that this count is relatively consistent over time (Figure 3). But does instability haunt specific users, or does it shift from user to user over time? That is: is the *set of unstable users* itself stable or changing?

To evaluate if anycast instability is persistent, we split each week into its first half and second half. We identify anycast unstable VPs in each half using our 10 minute threshold, then we compare the two sets to see how much overlap they have.

Table IV shows the number of unstable VPs in each half of the week, for both datasets. While the absolute number of unstable VPs varies by letter, the *most VPs that are unstable keep being unstable over the whole week*—the percent that overlap in the two halves of the week is at least 63% and typically around 90%. Anycast instability is a stable property between a VP and its anycast service. Although the two weeks we checked are more than half a year apart, we check the overlap over two different weeks and found there are still around 13% overlap.

It is also possible we see large amounts of overlap because many VPs are on the same networks—we rule this case out with additional validation. To check for bias from clustered VPs, we manually examined unstable VPs and their ISPs. We found these VPs belong to different ISPs.

This analysis shows unlucky VPs (those that are anycast unstable) are likely to continue to be unlucky. This result suggests that we must take care in interpreting the commercial success of anycast CDNs. Although they work well for most users, and analysis of their own data shows few broken TCP connections, it may be that their sample is not abundant enough, especially the VPs' coverage, because we just showed the instability is sticky with specific VPs over time. People will use anycast CDNs that work, but unlucky people that are anycast unstable for a particular CDN may simply turn away from that CDN (or its clients) because it doesn't "work" for them.

D. Is Anycast Instability Near the Client?

We next look at *where in the network* anycast instability appears to originate. Is it near the VP (the client), or near the anycast service's sites (the servers)? This question is of critical importance, because we have shown that some VPs are consistently anycast unstable. If the problem is located near the VP, it is likely that they will be unstable with *many* anycast services, and if an important service (like a CDN or Root DNS services) is provided *only* by anycast, then it might impossible for that VP to get service.

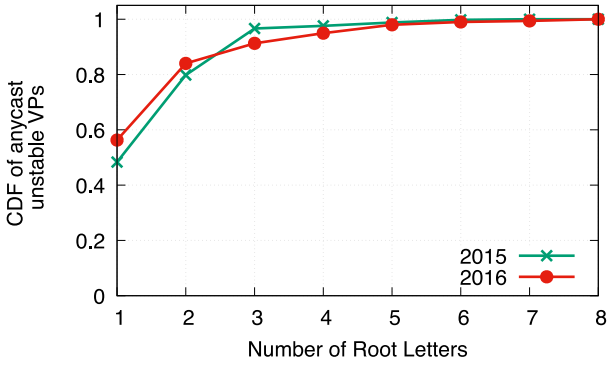


Fig. 4. The CDF of unstable VPs for how many root DNS services. The vast majority of VPs only experience instability towards one to three services.

To explore this question, we use the same approach we used to study the persistence of anycast instability (Section IV-C), but rather than comparing two halves of the same week, we compare different anycast services (different Root Letters). We consider three cases: (1) If instability occurs near an anycast site, then *many VPs* reaching that site should see instability. (2) If anycast instability is near a specific VP, we expect that this VP will be unstable with *many services*. (3) On the other hand, if a VP is unstable with only one service, then flipping likely occurs on the *path* between that VP to its current site.

We define *near* as occurring within the first (or last) three hops of the path, likely within the same ISP. The rest of the path is the *middle* of the network, since few organizations have more than a few hops before reaching another ISP.

We also assume that operators do not configure routers to treat different root letters differently. For example, operators will not configure the router as per-packet for traffic to one root-server, but per-flow for other root-servers.

We rule out case (1), since the service operator would notice and correct a site-specific problem. In addition, our study of number of unstable VPs per service that showed that there are at most a few anycast unstable VPs for each service (Figure 2).

For the 2015 dataset we identify 416 VPs that are anycast unstable for some root letter. Anycast instability is a property between the VP and a specific service, and Figure 4 shows for how many anycast services each of these VPs find to be unstable.

Our first observation is that *almost half of VPs are only unstable with one service*. Of the 416 VPs, 200 (48%) are unstable with only one of the 11 IP anycast services we study. We conclude that *the most common location of anycast instability is the middle of the network*, somewhere on a unique network path, not near the VP or an anycast site.

About the same number are anycast unstable with two or three services—202 of the 416 VPs, again about 48%. We conjecture that in these cases the problem is closer to the VP. Fortunately, it does not affect all services.

Only 2% of VPs are anycast unstable with more than 3 services, and none are unstable with more than 7. Since very few VPs have problems with all anycast services, we rule out case (2), that there are problems that are not tied up with the VPs.

The distribution of 2016 dataset is similar to 2015. The new weeks saw 494 unstable VPs, more than the 2015 datasets. The

TABLE IV
OVERLAP OF ANYCAST INSTABILITY FOR SPECIFIC VPS IN HALF-WEEKS

Root Letter	week of 2015-12-05 unstable VPs 1st	2nd	both	Overlap (percent)	week of 2016-08-01 unstable VPs 1st	2nd	both	Overlap (percent)	both weeks Overlap (percent)
A	—	—	—	—	2	2	2	100%	—
C	102	108	98	93%	97	113	67	64%	10%
D	301	106	99	63%	190	186	142	75%	14%
E	119	180	110	76%	173	183	129	72%	13%
F	107	111	107	98%	34	35	26	75%	7%
G	82	82	76	92%	44	49	30	64%	16%
I	84	85	76	89%	84	107	68	72%	9%
J	68	157	48	50%	94	74	64	77%	12%
K	99	100	94	94%	86	93	75	89%	18%
L	87	67	62	81%	93	102	80	82%	20%
M	53	52	46	87%	55	57	32	57%	24%

fact that highest number of letters the VP experiencing instability simultaneously goes from 8 to 7 is normal considering we add another letter B-root in our dataset.

One source of instability are paths that are load balanced over multiple links, where link selection is a function of information in packets that change in each packet. For example, the UDP source port is randomized in each of our queries; if it is included in a hash function for load balancing, packets could take different links. This problem has previously been observed in ping-based latency measurements [14]. Additional work is provided in following sections Section IV-F trying to understand these root causes.

We do not consider correlations between number of sites and degree of flipping. One might look at Figure 2 for correlations, but with only 11 architectures, each unique, it seems difficult to make statistically strong comparisons.

We conclude anycast instability does not correlate with a specific VP, or with a specific anycast site, but instability is a factor of the path between (VP, service) combinations, depending on their relative locations. The good news is that this conclusion means clients that see problems with one anycast service will likely be successful using some alterative services. Since Root DNS resolution is implemented by 13 underlying IP anycast services, this result implies that anycast instability is unlikely to impede access to Root DNS resolution. For CDNs [8], this result suggests the same content should be provided by multiple independent anycast deployments.

E. Higher Precision Probing Shows More Frequent Flipping

The long-term RIPE Atlas datasets (examined in Section IV-A) provide broad coverage for years, but each VP observes its catchment every 4 minutes, and we would like greater precision on flip frequency. We use this data to identify anycast unstable VP/service pairs, but these measurements are hugely *undersampled*—we expect some sites are flipping every packet, but 4 minute measurements of a VP flipping between two sites every packet will see a median flip time of 8 minutes. Improving the precision of this estimation is important because TCP connections are active for short times, often few tens of seconds, so proof of 4 minute flipping does not demonstrate TCP problems. In this section we take additional, direct measurements from RIPE Atlas to evaluate if these pairs are actually flipping more frequently than standard RIPE Atlas measurements are able to observe. (We cannot run custom measurement code on the VPs because RIPE does not support that, we have no way of contacting VP owners, and we require data from the few, specific VPs that show frequent flipping.)

To test this question we select 100 VPs to probe towards D-root in 30 minutes with unevenly distributed 95 probes, roughly one query per 20 seconds.

Figure 5 compares how many flips we see when the same VPs probe at 4 minute intervals (green filled dots on the bottom) compared to probes sent with about 20 s intervals (top open squares), for these 100 VPs with frequent flips. (We report counts of flips rather than mean flip duration because it is difficult to assess mean duration with this hour-long measurement.)

This data shows that more observations result in more flips—the open squares are always above filled dots. Of course more observations make more flips possible, but this data shows that 4 minutes measurements are *undersampled* and the path is flipping much more often. In fact, two VPs marked with asterisks show no flips during 4 minute observations, even though they flip frequently about at least every 30 seconds. If we assume every packet flips, then with fewer samples, these VPs just get "lucky" and appear stable with undersampling.

Since our measurements are not synchronized, sometimes we take measurements very close in time. As two specific examples, we saw one VP (84.246.12.69) flip from London to Frankfurt and back with three measurements in 7 s, and another (201.217.128.115) flip from Miami to Virginia and back in 10 s. In the next section we provide statistic evidences that suggest per-packet flipping are happening for specific VPs.

F. Does Per-Packet Flipping Occur?

We have shown that some VPs see very frequent flipping to some anycast services—as short as tens of seconds (Section IV-E). It seems unlikely that BGP is changing so frequently, since route flap damping is usually configured to suppress multiple changes within a few minutes.

To rule out BGP as the source of instability we analyse the dataset of RouteViews [23] from 2016-08-01 to 2016-08-07, finding that BGP is quite stable. Of the total 192 BGP RouteViews peers we studied, 0% to 23% RouteViews peers will ever see a BGP change, for each root letter, on each day.

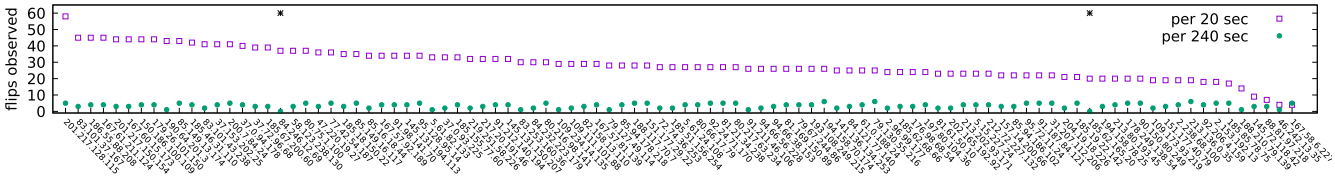


Fig. 5. Counting site flips from 100 VPs to D-Root. Measurements with about 20 s intervals (blue open squares on top) are compared to every 4 minutes (green filled dots on bottom). Two VPs with no flips in 4 minute data are marked with an asterisk (*).

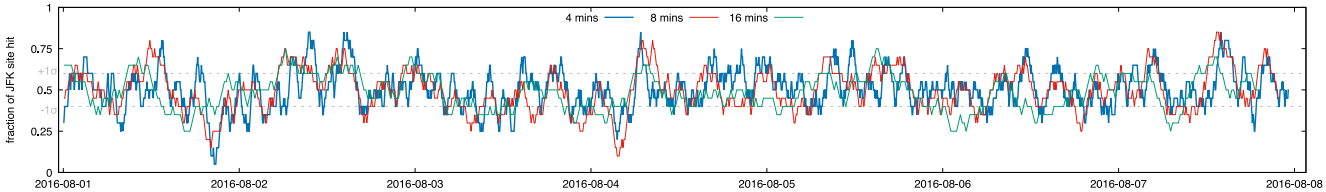


Fig. 6. Fraction of time one VP (146.186.115.74) spends at the JFK site of C-Root. Each point is the mean of a 20-observation sliding window, done at four timescales, 4 minutes (wide blue), 8 minutes (red), and 16 minutes (blue).

The mean time of BGP change seen by those RouteViews peers is always fewer than twice a day, so such infrequent routing changes cannot explain anycast flips that occur multiple times per minute. This new analysis supports previous studies that show BGP changes are relatively infrequent for Root DNS [13]. Instead, we suggest that these very frequent flips result from per-packet decisions made by load balancers in the path.

We cannot *directly* evaluate very frequent flips, because they occur only from specific VPs to certain anycast services. While we find them with RIPE Atlas, it limits probing intervals to 60 s, and even with multiple concurrent experiments, sub-second probing is impossible on RIPE. Neither can we reproduce these flips from another site, since they are specific to the path from that VP to the service.

However, we can *indirectly* show it is likely that these flips are per-packet by looking at how they respond in sliding time window over time. If the path is flipping every packet, then the probability of reaching a specific site should be *almost consistent over time*. We measure consistency by sliding a window over all observations and looking at the fraction of queries that go to different anycast sites. If flipping is per-packet, the fraction should be similar for any window duration and time period.

To evaluate this hypothesis, we return to the 2016 dataset, and we focus on the 100 VPs that have frequent site-slips towards C-Root (measured as those with time-to-flip around 10 minutes, roughly twice the measurement frequency). For each VP, we compute how many times their requests reached a specific anycast site in a window of 20 observations. We slide the window forward with one observation at a time, so windows overlap.

Figure 6 shows a representative example for one VP (146.186.115.74), which flips between the JFK and ORD sites of C-Root. We report the fraction of time the VP is at JFK, measured with a 20-observation moving window. We compute this moving window at three timescales, first using all the data (4 minute samples, the wide line), and also down-sampled two times (8 and 16 minutes). First, we see that the long-term average is around 0.5, consistent with each packet going one way or the other. There are peaks and valleys, as we expect with any long-term average, sometimes we get a run of one site or the other, but standard deviations is ± 0.1184 (shown as the dashed lines), and most of the time the average is within this range. However, lack of any repeating pattern suggests that there are not long-term flips, but per-packet.

In addition, when we compare the three timescales, all show similar properties. This result is consistent with all being drawn from random samples of per-packet flipping. These trends supports the suggestion that we would see similar results if we increase sampling frequency, as we showed experimentally to 20 s in Section IV-E.

We see the same behavior for this example VP in most of the other 100 VPs we observed. Figure 7 shows the mean and standard deviation of all selected VPs, sorted by mean. Most of these VPs show a ratio around 0.5 and a standard deviation around 0.1, consistent with our example, and consistent with random selection per-packet. Some sites on the right of the graph show an uneven split; we expect these are due to uneven load balancing, or multiple load-balanced paths.

Taken together, our experiments and this analysis present a strong case for per-packet flipping. Experiments at 4 minutes and 20 s (Section IV-E) directly support this claim, and our analysis at multiple timescales and across many VPs indirectly supports it.

G. Are TCP Connections Harmed by Anycast Flipping?

When a route flips, an active TCP connection will shift from one server to anther, almost certainly resulting in a connection reset. In the previous sections we studied route flipping with long-term UDP data; we next compare these UDP observations with TCP connections and TCP flipping.

1) The Relationship Between UDP Flipping and TCP-Flipping: Because TCP connections interact badly with route changes, most load balancers try to send TCP flows to the same destination, balancing *flows* instead of *packets*. Measurements of UDP flipping are therefore likely to overestimate the degree of TCP flipping. We expect that TCP instability will occur only

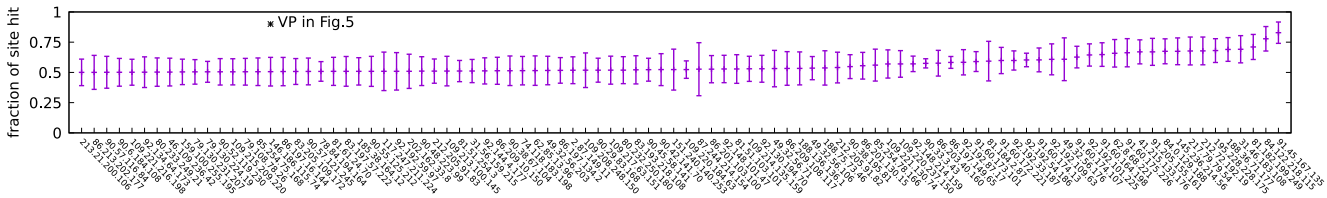


Fig. 7. Mean and standard deviation of site hit ratio across all sliding time window in a week.

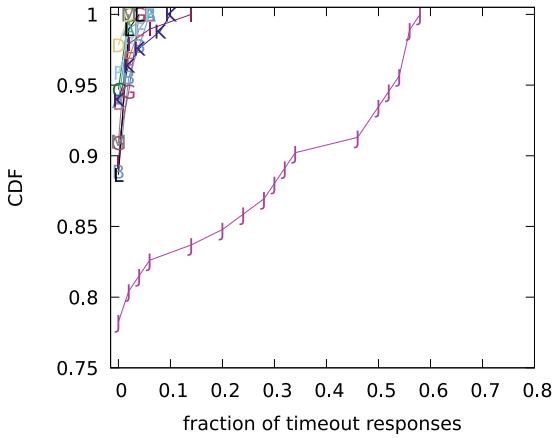


Fig. 8. Cumulative distribution of fraction of timeout responses of TCP query for each VP, broken down by anycast service. For each letter, the selected VPs are different, depending on the UDP flipping case. (Note the *y*-axis does NOT start at zero).

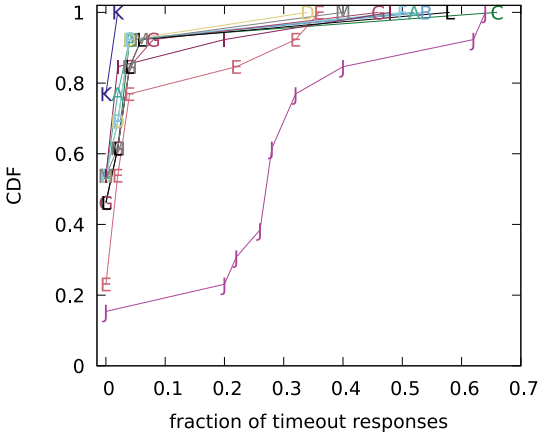


Fig. 9. Cumulative distribution of fraction of timeout responses of TCP query for each VP, broken down by anycast service. For each letter, the selected VPs are from a same set (Note the *y*-axis DOES start at zero).

on a subset of the (VP, letter) combinations than show UDP instability.

We next measure TCP connection success directly. We send TCP-based DNS queries from RIPE Atlas VPs to Root DNS letters, selecting the combinations that UDP-based measurements suggest are most unstable. We are looking for TCP connection failures, measured by how many TCP connections time out. We believe these timeouts indicate per-packet load balancing, while (VP, letter) combinations that show UDP flipping but not TCP flipping indicate load balancers that use per-flow scheduling.

For our experiment we begin with the 100 VPs for each letter that show the most frequent UDP flipping. (These VPs often are different for different letters, although there is some overlap.) We then make 50 TCP DNS queries from each VP to its letter, each 20 minutes apart (thus the experiment lasts 17 hours).

Figure 8 shows a CDF of what fraction of all TCP queries time out for all VPs and queries, broken out by letter. This experiment shows that *TCP is very stable most (VP, letter) combinations*, except for J-Root. In Figure 8, for all letters including J-root, more than 75% (VP, letter) combinations show no timeouts. This experiment shows that *most* load balancers *are* switching flows as a whole, not packets.

However, J-Root shows many more timeouts than the others, with 15 of VPs showing many timeouts (20% or more of all tries), and 7 VPs timing out on half or more of their queries. We later refer to these 15 VPs that timeout for more than 20% of all queries as *frequent TCP flippers*. We use 20% as a safe threshold, because in Figure 8, for all letters other than J, the normal timeout range is less than 20%. We emphasize that TCP timeouts are still *very, very rare* for J-Root: 99% of VPs see neither UDP nor TCP flipping to J-Root, so only about 0.15% see frequent TCP timeouts. Next sections examine this case to understand it better.

2) Why Does J-Root See More TCP Timeouts?: We find J-Root sees more TCP timeouts than other letters, and that this problem occurs only for a few VPs. We next examine if these VPs see problems with other roots, and look at the root cause of their TCP instability.

First, to see if these 15 frequent TCP flippers have problems with other letters, we repeat the experiment in Section IV-G by using the same 15 VPs to query all 12 roots that use anycast at the time of this experiment.

For these, almost all see frequent timeouts only in TCP queries to J-root, not to other root letters. In the Figure 9, out of total 15 VPs, 12 VPs have more than 20% of their queries to J-Root time out, but do not see such frequent timeout towards other roots. We refer to these 12 VPs as *J-Root-TCP-flippers*. This observation implies these timeouts are not caused by per-packet load balancing near the VPs, since load balancing near the VP would affect many letters.

One or two VPs time out in TCP queries to other letters besides J-root. In Figure 9, for every root except K-root, one or more VPs time out frequently as well. Our previous UDP-based sample selection in Figure 8 might leave out a few VPs that possibly saw frequent TCP query timeout as well (a result of per-packet flipping), but those VPs are included in Figure 9.

We believe TCP-flipping is closely associated with the routing path between a VP and its possible anycast destinations,

caused by a router in the path. The amount of frequent TCP flippers varies for different service. Also, VPs that timeout for one root DNS service usually do not timeout for other letters.

3) Can We Locate the Per-Packet Balancer?: We next examine the 12 J-Root-TCP-flippers in Figure 9 to identify what makes them unusual. From the RIPE atlas data recording the VPs' information, we find 11 of them are in Iran in several different ISPs, and 1 is in Finland.

We consider two possible reasons why these 11 VPs timeout frequently: DNS-based filtering, or a load balancer. While some countries do DNS-based filtering to censor Internet access, we rule out filtering because no other root letters see timeouts. Second, we do not see frequent timeouts on any other Iranian VPs in Figure 9 and Figure 8, suggesting this is not caused by a country-wide policy. Third, the 11 Iran J-Root-TCP-flippers sometimes see successful replies from J-root, suggesting an intermittent problem and not a systematic censorship.

To look for a load balancer in common across these 11 VPs. We use 34 traceroutes from each of these VPs to each DNS root letters. These traceroutes are already taken by RIPE Atlas and from the same time period as our the other Atlas data we use.

All VPs have traceroutes that show their traffic passes through private network address space [15] (except two VPs that have a router on their path blocks ICMP echo requests). Although VPs have public IP addresses, traffic of each VP passes through 1 to 9 hops of private address space before reentering public IP address space in a neighboring country. Paths often diverge in the private address space, but never with the same IP address, so we cannot identify a obvious specific router doing per-packet load balancing. Sometimes several paths share routers with common private /24 prefixes, but with private address space, it is difficult to judge devices for certain.

We believe per-packet load balancing happens in these private networks, although we cannot say more. Moreover, the traceroute directly shows, those Iran VPs' penultimate routers to the J-root destination shift among different sites because of the load balancing.

Traceroute records also explains why timeouts from those per-packet balancers occur only for J-Root and not for other letters. If we look at traceroutes from a specific VP to all letters, we see the penultimate routers to the J-Root destination shift between Malaysia and South Africa, two different J-Root sites. However, for A-Root, this same VP traverses the same private networks in Iran exiting at two different places in the public Internet, but both terminate at the same anycast site in Virginia, U.S. The same is true for B-Root, with most traffic terminating Los Angeles (and not the recently Miami site) C- and D-Root both have multiple sites, but all Iranian traffic terminates at one IP address in Los Angeles for C-Root and in Tokyo for D-Root. Other letters show similar patterns with different paths, but generally one consistent destination. Other VPs show the same pattern as the above VP, that they go through different traceroute path. Then for J-root, they end in different penultimate routers, but for other roots, they end in a single penultimate routers or multiple routers in one city.

These observations show that a per-packet load balancer will often cause TCP connections to timeout and be unavailable for flows that cross it, *but* that such configurations are very, very rare. This experiment result agrees with our findings in Section IV-D that load balancers are in the middle of the network. We believe there are one or more hops are configured as per-packet in the middle of private network space. In our dataset, a few Iranian VPs only timeout in TCP queries to J-roots but not to other roots, and not all Iranian VPs timeout in TCP queries to J-root.

V. Related Work

Prior studies have considered many aspects of anycast: latency [5], [24], geography [1], usage and traffic characteristics [6], [7], [9], [10], CDN load balancing [8], and performance under DDoS attack [13]. However, only a few studies have considered the stability of anycast [1], [12], and their conclusions are largely qualitative. Unlike this prior work, our goal is to *quantify the stability* of anycast.

Direct measurements: Prior stability studies either directly or indirectly measured catchments. Direct measurement studies of anycast stability use data from end-users or monitors that contact the anycast site. Microsoft has used Bing clients to study anycast and evaluate latency and load balancing. They observed 21% of end-users change sites at least once per a week [5]. However, the FastRoute system is concerned about small file downloads in their availability studies [8]. They also showed that anycast availability dipped from 99.9% to 99.6% once during their week-long observation, but do not discuss why.

LinkedIn [1] evaluated anycast with a synthetic monitoring service to evaluate latency and instability, and did not find "substantial instability problems". Our results suggest that most VPs are stable, so long-duration observation is unlikely to see new results, unless one studies from more vantage points located at other different places in the Internet.

Finally, recent studies of DNS Root anycast showed frequent routing flips during DDOS [13], but that paper did not study stability during normal periods.

Our work is also direct measurement like these prior studies, but unlike prior work we use many geographically dispersed VPs (more than 9000 from RIPE Atlas) multiple services (the 11 anycast Root DNS services, some with 100 sites), under normal behavior.

Indirect evaluation: Inference can estimate changes in anycast catchments by looking for changes in latency or hop counts (IP time-to-live). Cicalese and Giordano examined anycast CDN traffic by actively sending queries to each prefixes announced by 8 CDN providers [6]. The found anycast stable, with nearly constant RTT and time-to-first-byte, and consistent TTLs over a month. They later studied the duration of TCP connections for DNS and show that most last tens of seconds, suggesting that DNS will not be affected by infrequent anycast catchment changes [9]. Unlike their work, we directly observe site flips with CHAOS queries, rather than infer it. More important, we use 9000 VPs geographically dispersed across the world, while their study is based on VPs only in Europe.

VI. Conclusion

In this paper we used data from more than 9000 vantage points (VPs) to study 11 anycast services to examine the stability of site selection. Consistent with wide use of anycast in CDNs, we found that anycast almost always works—in our data, 98% of VPs see few or no changes . However, we found a few VPs—about 1%—that see frequent route changes and so are *anycast unstable*. We showed that anycast instability in these VPs is usually "sticky", persisting over a week of study. The fortunate fact, that most unstable VPs are only affected by one or two services, shows instability causes may lie somewhere in the middle of the routing path. By launching more frequent requests, we captured very frequent (back and forth within 10s) routing change in our experiments using the unstable VPs we discovered from previous analysis, the statistical analysis shows they are possibly affected by per-packet flipping, which is potentially caused by load balancer in the path. Also, we perform experiment by the same sources and targets but with TCP connection. We find TCP anycast instability is even rarer but exists and harms. Our results confirm that anycast generally works well, but when it comes to a specific service, there might be a few users experiencing routing that is never stable.

Acknowledgment

This research has been partially supported by measurements obtained from RIPE Atlas, an open measurements platform operated by RIPE NCC. We thank them for sharing their data.

Lan Wei's work is technically and spiritually supported by her very first network research colleagues, Ricardo de Oliveira Schmidt, Wouter Bastiaan de Vries, and her advisor John Heidemann. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views contained herein are those of the authors and do not necessarily represent those of DHS or the U.S. Government.

References

[1] P. Bret, K. Prashanth, J. Samir, and A. K. Zaid. (Sep. 2010). *TCP Over IP Anycast—Pipe Dream or Reality?* [Online]. Available: https://engineering.linkedin.com/network-performance/tcp-over-ip-anycast-pipe-dream-or-reality

[2] (Apr. 2017). *CacheFly Network Map*. [Online]. Available: https://web1.cachefly.net/assets/network-map.html

[3] M. Caesar and J. Rexford, "BGP routing policies in ISP networks," *IEEE Netw. Mag.*, vol. 19, no. 6, pp. 5–11, Nov./Dec. 2005.

[4] M. Calder, X. Fan, Z. Hu, E. Katz-Bassett, J. Heidemann, and R. Govindan, "Mapping the expansion of Google's serving infrastructure," in *Proc. ACM Internet Meas. Conf.*, Barcelona, Spain, Oct. 2013, pp. 313–326.

[5] M. Calder, A. Flavel, E. Katz-Bassett, R. Mahajan, and J. Padhye, "Analyzing the performance of an anycast CDN," in *Proc. ACM Conf. Internet Meas. Conf.*, Tokyo, Japan, 2015, pp. 531–537.

[6] D. Cicalese, D. Giordano, A. Finamore, M. Mellia, M. Munafò, D. Rossi, and D. Joumblatt, "A first look at anycast CDN traffic," *arXiv preprint arXiv:1505.00946*, 2015.

[7] X. Fan, J. Heidemann, and R. Govindan, "Evaluating anycast in the domain name system," in *Proc. IEEE INFOCOM*, Turin, Italy, 2013, pp. 1681–1689.

[8] A. Flavel, P. Mani, D. A. Maltz, N. Holt, J. Liu, Y. Chen, and O. Surmachev, "FastRoute: A scalable load-aware anycast routing architecture for modern CDNs," in *Proc. USENIX Symp. Netw. Syst. Design Implement.*, Oakland, CA, USA, May 2015, pp. 381–394.

[9] D. Giordano, D. Cicalese, A. Finamore, M. Mellia, M. Munafo, D. Joumblatt, and D. Rossi, "A first characterization of anycast traffic from passive traces," in *Proc. IFIP Workshop Traffic Monitoring Anal. (TMA)*, 2016, pp. 30–38.

[10] J. Hiebert, P. Boothe, R. Bush, and L. Lynch, "Determining the cause and frequency of routing instability with anycast," in *Proc. Asian Internet Eng. Conf. (AINTEC)*, Pathum Thani, Thailand, Nov. 2006, pp. 172–185.

[11] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman, "Specification for DNS over transport layer security (TLS)," Internet Eng. Task Force, Fremont, CA, USA, RFC 7858, May 2016.

[12] M. Levine, B. Lyon, and T. Underwood, "TCP anycast—Don't believe the FUD," presented at the NANOG 37, Jun. 2006. [Online]. Available: https://www.nanog.org/meetings/nanog37/presentations/matt.levine.pdf

[13] G. C. M. Moura, R. de O. Schmidt, J. Heidemann, W. B. de Vries, M. Müller, L. Wei, and C. Hesselman, "Anycast vs. DDoS: Evaluating the November 2015 root DNS event," in *Proc. ACM Internet Meas. Conf.*, Santa Monica, CA, USA, Nov. 2016, pp. 255–270.

[14] C. Pelsser, L. Cittadini, S. Vissicchio, and R. Bush, "From Paris to Tokyo: On the suitability of ping to measure latency," in *Proc. ACM Internet Meas. Conf.*, Barcelona, Spain, Oct. 2013, pp. 427–432.

[15] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear, "Address allocation for private Internets," Internet Eng. Task Force, Fremont, CA, USA, RFC 1918, Feb. 1996.

[16] RIPE NCC. (2015). *DNSMON*. [Online]. Available: https://atlas.ripe.net/dnsmon/

[17] RIPE NCC. (2015). *RIPE Atlas Root Server Data*. [Online]. Available: https://atlas.ripe.net/measurements/ID

[18] RIPE NCC. (2017). *RIPE Atlas Self TCP Measurement 1*. [Online]. Available: https://atlas.ripe.net/measurements/ID

[19] RIPE NCC. (2017). *RIPE Atlas Self TCP Measurement 2*. [Online]. [Available]: https://atlas.ripe.net/measurements/ID

[20] RIPE NCC. (2017). *RIPE Atlas Self Traceroute Measurement*. [Online]. Available: https://atlas.ripe.net/measurements/ID

[21] RIPE NCC. (2017). *RIPE Atlas Self UDP Measurement*. [Online]. Available: https://atlas.ripe.net/measurements/ID

[22] (Apr. 2016). *Root Operators*. [Online]. Available: http://www.root-servers.org

[23] RouteViews. (Aug. 2016). *Routeviews2, Routeviews3, Routeviews4*. [Online]. Available: http://bgpmon.io/archive/help

[24] R. de Oliveira Schmidt, J. S. Heidemann, and J. H. Kuipers, "Anycast latency: How many sites are enough?" in *Proc. Passive Active Measur. Workshop*, Sydney, NSW, Australia, May 2017, pp. 188–200.

[25] L. Wei and J. Heidemann, "Does anycast hang up on you?" in *Proc. IEEE Int. Netw. Traffic Measur. Anal. Conf.*, Dublin, Ireland, 2017, pp. 1–9.

[26] S. Woolf and D. Conrad, "Requirements for a mechanism identifying a name server instance," IETF, Fremont, CA, USA, RFC 4892, Jun. 2007.

[27] L. Zhu, Z. Hu, J. Heidemann, D. Wessels, A. Mankin, and N. Somaiya, "Connection-oriented DNS to improve privacy and security," in *Proc. 36th IEEE Symp. Security Privacy*, San Jose, CA, USA, May 2015, pp. 171–186.

Lan Wei is currently pursuing the Ph.D. degree with the Computer Science Department, University of Southern California. She is an active researcher in networking and systems. Her research interests include Internet measurement, traffic engineering, and performance of CDNs, DNS, and DDoS.

John Heidemann (S'90–M'95–SM'04–F'14) received the B.S. degree from the University of Nebraska-Lincoln in 1989, and the M.S. and Ph.D. degrees from the University of California, Los Angeles in 1991 and 1995, respectively. He is a Senior Project Leader with the University of Southern California/Information Sciences Institute (USC/ISI) and a Research Professor in computer science with USC. At ISI, he leads the Analysis of Network Traffic Laboratory, studying how to observe and analyze Internet topology and traffic to improve network reliability, security, protocols, and critical services. He is a Senior Member of ACM.