

# Characterizing Anycast in the Domain Name System

USC/ISI Technical Report ISI-TR-681, May 2012\*

Xun Fan  
Information Sciences Institute  
Univ. of Southern California  
xunfan@isi.edu

John Heidemann  
Information Sciences Institute  
Univ. of Southern California  
johnh@isi.edu

Ramesh Govindan  
Computer Science Dept.  
Univ. of Southern California  
ramesh@usc.edu

## ABSTRACT

IP anycast is a central part of production DNS. While prior work has explored proximity, affinity and load balancing for some anycast services, there has been little attention to third-party discovery and enumeration of components of an anycast service. Enumeration can reveal abnormal service configurations, benign masquerading or hostile hijacking of anycast services, and can help characterize the extent of anycast deployment. In this paper, we discuss two methods to identify and characterize anycast nodes. The first uses an existing anycast diagnosis method based on CHAOS-class DNS records [45] but augments it with traceroute to resolve ambiguities. The second proposes Internet-class DNS records which permit accurate discovery through the use of existing recursive DNS infrastructure. We validate these two methods against three widely-used anycast DNS services, using a very large number (60k and 300k) of vantage points, and show that they can provide excellent precision and recall. Finally, we use these methods to evaluate anycast deployments in top-level domains (TLDs), and find one case where a third-party operates a server masquerading as a root DNS anycast node as well as a noticeable proportion of unusual anycast proxies. We also show that, across all TLDs, up to 72% use anycast, and that, of about 30 anycast providers, the two largest serve nearly half the anycasted TLD name-servers.

## 1. INTRODUCTION

Rapid response and high availability requires that large network services be distributed widely, often with a single logical service is provided by distributed replicas accessed using a single logical identifier. Content delivery networks (for example, [16]), mirroring services (for example, [15]), URNs [40], and IP anycast [35] all fit this model.

In IP anycast, as standardized by the IETF [35, 1],

\*This work is partially supported by the United States Department of Homeland Security contract numbers N66001-10-C-0081 (“AMITE”) and NBCHC080035 (“LANDER-2007”). All conclusions of this work are those of the authors and do not necessarily reflect the views of DHS.

an anycast service operates on a single IP address, but multiple *anycast nodes* replicate that service at different physical locations. Each node may be implemented with one or more *servers* (a physical or virtual machine), each of which listens to the anycast address and often also one or more unicast addresses. Standard interdomain routing directs clients to the nearest replica and handles fail-over to other nodes as required. (We review details and terms in Section 2.)

Anycast is used for many core services of the Internet today. It is widely used for DNS [22]: as of April 2011, 10 out of 13 root name servers employ anycast [37]. Other uses include discovering IPv6-to-IPv4 relay routers [24] and sinkholes [21] and for load distribution [41, 17]. Anycasted services benefit from anycast’s load balancing and ability to mitigate denial-of-service attacks [1], and research proposals have discussed improvements to scale to many anycast destinations [26].

The use of anycast for core Internet services suggests we need to understand its performance, use, and robustness. In this paper, we focus on understanding anycast use in DNS. Extensive prior work (Section 7) has measured server proximity, the affinity between clients and anycast services, and the performance of load balancing of anycasted DNS. However, to date there has been no effort to systematically discover and map anycast use in DNS. As we show in Section 5, such a capability can aid in diagnosing abnormal name service configurations, and help understand the extent of anycast deployment.

The first contribution of our work is to evaluate different approaches to automatically *discover* and *enumerate* all nodes of an anycast service. To understand the challenges in anycast discovery, we first taxonomize anycast deployment configurations (Section 2). Anycast discovery is challenging because anycast configurations can be somewhat complex, existing diagnosis methods are not standardized and can result in measurement ambiguity, and the visibility of anycast servers can be topologically scoped requiring a large number of vantage points.

We then discuss the design of *two methods to enumerate anycast nodes*. The first method uses an existing anycast diagnosis technique based on CHAOS-class TXT DNS records [45], but augments it with traceroute to identify non-cooperative anycast nodes and resolve ambiguities in CHAOS-based discovery (Section 3.1). This approach requires specific measurement support to be widely deployed, sometimes limiting its coverage. Our second method (Section 3.2) proposes the use of Internet-class (IN) TXT DNS records to enable the use of tens of thousands of recursive DNS servers as vantage points. Finally, we identify security concerns to anycast diagnosis and describe how providers can control its use (Section 6).

A careful validation of these methods, using 60k and 300k vantage points, reveals interesting trade-offs (Section 4). CHAOS queries issued from 60k Netalyzr clients discover (measured using *recall* from information retrieval) 93% of F-root anycast servers. However, because the CHAOS query format is not standardized, different providers use different conventions to identify anycast servers; this results in a measurement ambiguity that can be resolved using traceroute probes. A smaller scale experiment on PlanetLab using 238 nodes reveals that *precision* of CHAOS queries can be improved from 65% to 89% using traceroute, and to 100% if the provider’s CHAOS labeling conventions are known. Finally, we show that, up to 90% recall is possible on-demand, when we shift to IN queries and 300k recursive DNS servers, as evaluated on the AS112 anycast service.

More important, we find that 10,000 or more vantage points are required to reach a recall of 80% for either method (Section 4.2). For context, almost all prior work on anycast performance (the exception being [9]) has used only hundreds of vantage points. Interesting future work may be to examine whether their conclusions would be significantly altered by a broader perspective as suggested by our approaches.

Our second contribution is to understand how anycast is used in practice over many services (Section 5). Until recently, the AS112 anycast service used manual methods to track their extent of deployment; our evaluations find that the manual list is out-of-date and incomplete, with about 26% of listed nodes no longer operational, and four providers operating multiple nodes. Recently, AS112 operators have adopted a discovery method similar to what we propose. Second, we evaluate anomalous anycast usage (Section 5.1). We found one third-party DNS server *masquerading* as an anycast node for a public root server, and hundreds of users observe what are likely in-path proxies. This demonstrates the importance of dynamic discovery methods to audit critical Internet infrastructure. Finally, in Section 5.3, we apply anycast discovery to servers for all top-level domains, showing that up to 72% of TLDs may now be

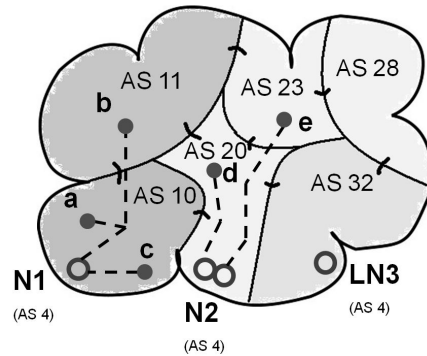


Figure 1: Anycast and routing: three anycast nodes (N1, N2, and LN3) and their catchments (dark, light and medium grey regions). Observing two nodes from vantage points *a* to *e*.

using anycast. Moreover, we are able to estimate the distribution of TLDs across providers of anycast service and find that almost half the TLDs are hosted by two anycast providers. Thus, our methods can lead to new insights about anycast usage and, in the future, enable an understanding of how this usage evolves over time.

Data we generated for this paper is no cost [29].

## 2. A TAXONOMY OF ANYCAST CONFIGURATIONS

IP anycast provides clients of a distributed service with a single-server abstraction [35]. Clients send traffic to a designated IP address identifying the *anycast service*. However, the service itself is implemented by a service provider using one or more *anycast nodes* that can be physically distributed around the Internet. Standard routing protocols such as BGP ensure that the user’s packets are sent to a nearby anycast node. Since successive packets from a client can be routed to different anycast nodes (e.g., as a result of network dynamics) anycast is usually used only for stateless, single-packet-exchange services like DNS [22] or datagram relay [24].

Figure 1 shows how three anycast nodes cover six ASes; each node covers a different region or *catchment* as shown by different shades of gray. We now discuss anycast routing terminology (RFC4786 [1]) and present a taxonomy of anycast node configurations; this taxonomy informs the design of our anycast enumeration methods.

**Routing configurations:** Anycast nodes have two levels of external visibility: global and local. Global nodes can be seen across multiple ASes, while local nodes are visible only within the hosting or adjacent AS. In Figure 1, anycast nodes N1 and N2 are global, each with catchments encompassing multiple ASes (AS10 and AS11; and ASes 20, 23, and 23, respectively), Node LN3





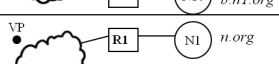
		Node topology	Label	PR
T1	One router One server		<i>l</i>	1
T2	Multi-router One server		<i>l</i>	>1
T3	One router Multi-server		> <i>l</i>	1
T4	Multi-router Multi-server		> <i>l</i>	>1
T5	Multi-nodes With same CHAOS		<i>l</i>	>1

Figure 2: Anycast node configurations. Observed labels in italics and penultimate-hop traceroute routers in bold. “VP” indicates vantage points, “R1, R2” indicates penultimate routers, “N1a, N1b” indicates servers in the same node, “N1, N2” indicates different nodes.

is local and so affects only AS32.

Global nodes advertise an anycast IP prefix to BGP and are visible Internet-wide [1]. Local nodes advertise an anycast IP prefix with the *no-export* BGP attribute [3] and are visible only to adjacent autonomous systems. Larger anycast services often include both local and global nodes, but either may be omitted.

Anycast is available in both IPv4 and IPv6. In this paper we consider only IPv4, although to our knowledge our approaches generalize to IPv6.

**Node configurations:** While routing allows clients to access a nearby anycast node, a node itself can be structurally complex; each node may be implemented by one or more *anycast servers*. Figure 2 taxonomizes all important configurations that we have encountered.

The top row (T1) shows the simplest case, where a single server provides service at a given anycast node. This server listens to traffic on the service anycast address. To provide access to management functions, it also typically listens on a second unicast address, unique to the server, but not shown in Figure 2.

Since anycast nodes are often placed in Internet exchanges (IXP) with complex local topologies, row T2 of Figure 2 and node N1 in Figure 1 show a single server with links to multiple adjacent routers, either connected by a shared LAN or with multiple network interfaces.

For large services such as a top-level domain server, a service at an anycast node may be provided by multiple physical servers. Cases T3 and T4 show multiple servers behind one (T3) or two or more (T4) routers. Node N2 in Figure 1 shows the T4 case.

Nodes or servers often have labels for diagnostic purposes. Current diagnostic practices encourage the use

of unique labels [45], but in some cases, either due to misconfiguration or hijacking, different nodes can end up with the same labels. Case T5 identifies this incorrect case. We cannot distinguish T5 from T2 by external observation; we see neither case in our ground truth but do observe such cases in our study of TLDs (Section 5.3).

### 3. METHODS FOR ANYCAST DISCOVERY

We wish to discover all anycast nodes and servers. Anycast nodes and servers *cannot* be enumerated simply by sending an IN-class DNS query to an anycast address, since standard responses contain no information specific to the anycast node. Instead, we must send queries from within the catchment of anycast node that elicit unique information from that node, as shown in Figure 1.

We describe two such active probing methods below. First, we extend the existing diagnostic convention, that uses *CHAOS queries*, by adding traceroutes. Second, we develop a new proposal for a standardized type of *IN query*. These methods differ in how much information they return and what vantage points they can use.

#### 3.1 CHAOS Queries

Anycast providers require methods to observe and debug their services. Their current methods use DNS records to identify individual anycast nodes and servers as documented in RFC-4892 [45]. Although not mandatory, we find these conventions used widely (Section 5.3).

Since anycast is often used for DNS services, and DNS provides a convenient external query mechanism, RFC-4892 uses distinct DNS records to identify specific anycast servers. It re-purposes CHAOS-class network records from the now defunct Chaosnet to provide anycast diagnosis. Standard DNS records [33] with class CHAOS, type TXT, and name *hostname.bind* or *id.server* are defined to return a unique string per anycast server. The contents of the string are provider-defined and not formally standardized, although we have identified common conventions (see [19]).

In principle, presence of these records should make identifying anycast servers trivial. Standard DNS tools (such as dig or nslookup) can retrieve this information. Because CHAOS records are tied to individual servers, they correctly identify single-server nodes (cases T1 and T2 in Figure 2) and can also detect each server in multi-server nodes (cases T3 and T4).

In practice, CHAOS records are not always sufficient to identify anycast servers. They are specified in an informational RFC, and not in a mandated standard, so providers may choose to not to provide them. They were initially implemented in the BIND implementation of DNS (hence “bind” in the record name). As of Dec. 2011, half of the 16 different DNS implementa-

tions listed in Wikipedia support CHAOS records [44], including all implementations we know that are used to host large services (BIND, NDS, Nominus ANS, Microsoft DNS, PowerDNS, and UltraDNS). In addition, CHAOS records indicate anycast servers, but conventions to relate anycast servers to nodes are unspecified. Thus, the multi-server cases T3 and T4 in Figure 2, or example N2 in Figure 1, require additional information to determine the two servers located at the anycast node. Finally, CHAOS records may be misconfigured (case T5 of Figure 2). For example, a DNS masquerader or hijacker may intentionally omit or provide duplicate CHAOS records (as shown in Section 5.1).

These shortcomings motivate our design of a qualitatively different method based on IN queries (Section 3.2). However, it is possible to overcome some of these limitations by augmenting CHAOS queries with traceroute information.

**Using Traceroute for Disambiguation:** Consider a traceroute from a vantage point to its nearest anycast node. We simplify the path and focus on the *penultimate router*, or *PR*. Our hypothesis behind this method is that each anycast node will have one PR, as exemplified by case type T1 in Figure 2.

In practice, this hypothesis is only partially correct, since anycast nodes with a rich local topology sometimes have multiple PRs (case T2 of Figure 2) or multiple servers per node (cases T3 and T4). These cases complicate our analysis. Since these routers are nearly always co-located with the anycast node in the same IXP, we use simple heuristics to partially address this problem. We assume routers with “nearby” addresses are in the same IXP; currently, we define nearby as within the same /24 address block. In Section 4.3 we show how a combination of the methods can help. Development of better PR alias resolution is an area of ongoing work.

From Figure 2, we see that traceroute complements CHAOS queries. Sometimes both methods work (case T1), or one of the two works (cases T2, T3, and T5). In case T4, both methods fail with an overcount of the anycast node, and when no vantage point is in the node’s catchment, they undercount. When possible we use them together: if either method results in a single identifier, we know there is a single anycast node, even if the other suggests multiple nodes. We take observations from all vantage points as input, separately merge records with duplicate CHAOS and PR identifiers, and finally merge these lists to get a combined estimate.

**Vantage Points:** As a result of the specific naming convention used for anycast identification (*hostname.bind*), these records cannot be retrieved using recursive DNS queries. As such, use of this method requires customized software in each catchment. One option is to use public

research infrastructure like PlanetLab. In our experiments we generally use 238 PlanetLab nodes, about one per unique site. However, as a research platform, PlanetLab servers do not provide the geographic and topological diversity we need to cover all catchment areas. Even today, with “only” around 500 sites, PlanetLab cannot cover all ASes.

To overcome this limitation, we have also crowd-sourced anycast discovery. We requested the Netalyzr [27] developers to add our methods their service. They implemented CHAOS queries, but omitted traceroute due to constraints of Java. In Section 4, we examine Netalyzr data obtained from about 62k vantage points.

## 3.2 IN Queries

While the CHAOS query is current practice, its use requires diagnostic software at a vantage point in each anycast catchment. (For example, LN3 in Figure 1 is missed for this reason.) While Netalyzr’s clients provide reasonable coverage, we consider an alternative that provides more convenient anycast discovery.

Regular Internet-class (IN) DNS records support *recursive DNS (rDNS) queries*, allowing the use of tens of thousands of open recursive DNS-servers to serve as vantage points which can be accessed easily from a centralized measurement site. We therefore propose a new approach using IN TXT records for anycast enumeration.

For IN queries, we propose that each anycast service define a designated subdomain `_ns-diagnostics` delegated to the anycast server. Inside that subdomain, dedicated TXT-type resource records identify anycast servers (label `_instance-id`) and nodes (`_node-id`) anycast instances. Thus, a node of the F-root could be identified by querying `_node-id._ns-diagnostics.f.root-servers.net`. The key advantage of IN records is that, unlike CHAOS records, they can be retrieved through recursive queries. We place them as a subdomain of the service domain so they require no new protocols; we use an unusual designated subdomain so their label is unlikely to conflict with existing domains. Our mechanism therefore requires that each anycast service create a separate zone for diagnostic information, and that each server populate that zone with server-specific resource records following our convention.

We have offered this proposal for standardization [18], but it is under consideration and not yet deployed. However, the AS112 anycast service uses a similar approach; it provides a proxy to evaluate our approach in Section 4.2.

**Vantage Points:** Our IN-class records can be queried using recursive DNS servers (rDNS), so they do not require custom diagnostic software (in PlanetLab or Netalyzr) at each vantage point. Many DNS servers offer recursive service, and a few hundred thousand of these

Vantage points	number	countries	ASes
PlanetLab	238	40	186
Netalyzr clients	61,914	164	4153
rDNS	318,988	220	15,210

Table 1: Vantage points diversity

support public queries. By sending queries indirectly through rDNS, each rDNS server effectively becomes a vantage point, potentially covering many more ASes and anycast catchments. We use open rDNS servers to quantify the performance of IN query based enumeration.

## 4. VALIDATION

We next evaluate the accuracy of CHAOS and IN queries, and illustrate the role that traceroute plays in improving the accuracy of CHAOS queries.

### 4.1 Methodology

We are interested in the efficacy of the anycast discovery methods discussed in the previous section. We evaluate this from many global vantage points to three large anycast services for which we have ground truth.

**Vantage points:** We use three different sets of vantage points: Netalyzr clients, rDNS servers, and PlanetLab. The number and reach of these vantage points is shown in Table 1. We note that, with the exception of PlanetLab, the number of vantage points in our study is an order of magnitude higher than in previous work.

**Targets:** We study three anycast services in our experiments. In most cases we study the F-root DNS service run by ISC, and the Packet Clearing House (PCH) Anycast service that provides service for 56 TLDs. We selected these providers as targets for two reasons. First, as large, professional anycast providers serving important major domains, they are representative of other major anycast providers. Second, both are non-profit organizations willing to support research, both with public descriptions of their infrastructure and willingness to respond to our queries.

To evaluate IN queries, we use AS112 an anycast service providing reverse name lookups for private address space.

**Ground truth:** We consider two types of ground truth: oracle truth, and authority truth. By *oracle truth*, we mean the actual set of nodes that respond to an anycast address in the Internet at any instant. We identify it as “oracle” truth because defining it requires a perfect snapshot of network routing from all parts of the Internet—an impossible goal. We define *authority truth* as the list of anycast nodes that we get from the anycast

service provider.

Oracle and authority truth can diverge for two reasons. First, third parties may operate anycast nodes for a given service with or without the provider’s knowledge. Such third party nodes would not be part of authority truth. We discuss an example of a third-party node for F-root in Section 5.1. Second, we derive authority truth from public web pages, which can sometimes lag the current operational system, as discussed in Section 4.3.

**Metrics:** Our active probing methods can result in several categories of results, with respect to authority and oracle truth. This influences our choice of metrics.

Consider Figure 1 where five vantage points (*a* through *e*) probe three anycast nodes. Probes from *a* through *c* find N1, and *d* and *e* find N2, the *true positives* (or *tp*). Node LN3 is omitted because there are no vantage points in its catchment, so it is a *false negative* (an undercount, *fn*). To correct this error, we need a new vantage point in LN3’s catchment; we study this relationship in Section 4.2.

There are three cases that might be classified as false positives. If we are unable to distinguish that two machines at N2 represent a single anycast node, then we would *overcount*. In an overcount, neither observation is completely wrong (since both N2a and N2b are anycast servers), but they result in a mis-estimate of the number of anycast *nodes*. When we detect a node that we confirm is operated by the anycast provider but is not in authority truth, we have a *missing authority* (“missauth” for short). Finally, if a non-authorized anycast node appeared in the AS with vantage point b, we record an *extra* node. An extra node is a false positive when compared to authority truth, but it is a true positive when compared to oracle truth.

We define *precision* against authority and oracle truth:

$$precision_{authority} = \frac{tp}{tp+overcount} \quad (1)$$

$$precision_{oracle} = \frac{tp+missauth+extra}{tp+missauth+extra+overcount} \quad (2)$$

In general, we do not have false positives (because everything we find is an anycast server). Therefore authority precision reflects our level of accidental overcounts due to multi-server or multiple PR nodes.

*Recall* captures the coverage of authority truth:

$$recall = \frac{tp}{tp+fn} \quad (3)$$

We do not define a recall for oracle truth because we do not have a complete set of the oracle population.

### 4.2 Recall

Ultimately our recall is dominated by our ability to see different anycast nodes. At best, each vantage point is in a different catchment and sees a new node; at

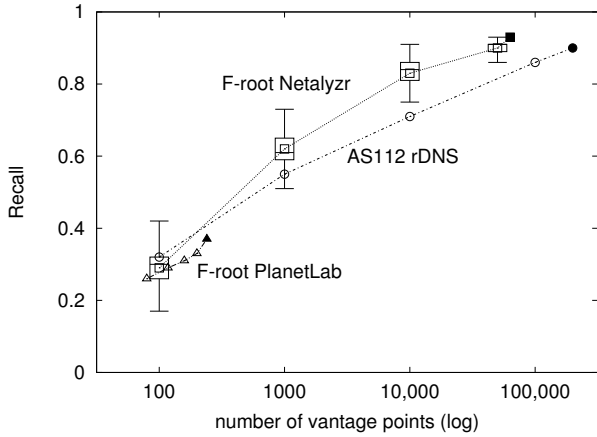


Figure 3: Recall of the CHAOS method on F-root PlanetLab, F-root Netalyzr and AS112 rDNS, as number of vantage points vary. For F-root Netalyzr each box shows median and quartiles, with the whiskers showing extrema. Lines show mean values. Exact values (filled squares, circle and triangle) represent our best experimental results (93% for Netalyzr, 90% for AS112 and 37% for PlanetLab).

worst, they are all in the same catchment and we are uncertain if the target is using anycast at all. In Figure 1, we see that vantage points  $a$ ,  $b$  and  $c$  duplicate each other, as do  $d$  and  $e$ . We next explore how query method and numbers of vantage points affect recall.

**Recall for CHAOS Queries:** We first consider recall for CHAOS queries. Figure 3 shows recall as a function of number of vantage points for F-root from PlanetLab and Netalyzr. For each line, the right-most point represents the complete observation. We also plot recall from smaller subsets of vantage points by taking 1000 random samples of smaller populations to estimate the effect of numbers of vantage points on recall. For Netalyzr, we show quartiles and extrema with box plots; other cases have similar distributions, omitted for clarity.

First, we see that with 62k vantage points, Netalyzr finds nearly all F-root anycast nodes at 93% recall (53 of the 57 official F-root nodes). By contrast, the 238 vantage points in PlanetLab provide a recall of only 37%.

We also see a roughly logarithmic relationship between recall and the number of vantage points: recall grows very slowly with increasing numbers of vantage points. On average, about 10,000 vantage points are required to achieve 80% recall; we note that, with the exception of [9] which used 20K rDNS servers, all prior anycast measurement studies have used far fewer van-

tage points.

**Recall for IN Queries:** Our proposal for standardizing IN queries for anycast identification is not yet widely deployed. Fortunately AS112’s anycast service is ideal to test our IN queries approach because its providers have adopted the convention that each anycast node include a unique `hostname.as112.net` IN TXT DNS record; these records can serve as a proxy for our IN query based approach.

We queried AS112 using over 300,000 rDNS servers, and found 65 servers; in contrast, issuing IN queries for AS112 from PlanetLab reveals only 17 of these servers. These statistics reveal the scale of vantage points required in order to enumerate anycast servers; almost 3 orders of magnitude more vantage points were required to quadruple the number of observed servers. This is more evident in Figure 3 which shows that 300,000 rDNS servers achieved a recall of 90%. Furthermore, our analysis of the recall achieved by subsets of rDNS servers reveals that almost 100,000 rDNS servers are required to achieve a recall of 80%. Intriguingly, rDNS exhibits lower recall than using Netalyzr clients (the line for AS112 is consistently lower than the line for F-Root); we have left to future work an understanding of whether this difference results from differences in the two anycast services, or arises from the type or placement of vantage points.

To compute recall, we needed to calculate the authority list of anycast servers for AS112, and this proved tricky. The AS112 project maintains a voluntary list of known providers [5]. However, because all AS112 nodes are run by volunteers, who use public information to set up new nodes [2] and who only coordinate loosely with each other, this list is both incomplete and out-of-date. In particular, AS112 coordinators and our data confirms that the list is missing some providers and has others that have ceased providing service. Each entry of the list identifies a provider by name and AS number. Some entries include one or more unicast IP addresses for an anycast node’s DNS server. The list identifies providers, not anycast nodes, so even when up-to-date, it can under-represent providers that run multiple anycast nodes.

Table 2 compares anycast nodes found by our IN queries approach to this list. We found that rDNS discovered 35 nodes that were *not* in the AS112 list, confirming that the voluntary list is incomplete and that *automatic diagnosis is important*. Moreover, rDNS discovered 42% of the provider’s list; this value does not represent recall, however, because that list is also out-of-date—some providers shut their services down but neglected to remove themselves from the list.

To build a more accurate “ground truth”, we evaluate which entries on the list are actually alive or we can confirm are no longer operational. By default, we assume

				<b>authority</b>	<b>rDNS</b>
Found by rDNS, but not in ground truth	35			missing	<b>new</b>
provider not present	26			missing	<b>new</b>
nodes run by providers with multiple nodes	9	(of 4 providers)		missing	<b>new</b>
Operator list (authority truth)	70	100%		both known	
found by rDNS	30	42%		both known	
not found by rDNS	40	58%		known	<i>possible missing</i>
possible alive	52	74%	[100%]		
definite alive	37		[71%]		
found by rDNS (and not BGP)	30	(rDNS recall)	[58%]	both known	
found by PlanetLab (and not BGP)	14	(PlanetLab recall)	[27%]	both known	
found by BGP information (and not rDNS)	7		[13%]	known	<i>missing</i>
have rDNS in address block	1				
no rDNS in address block	6				
not found by any means	15		[29%]	interpretation uncertain	
possible down	18	26%		out-of-date	<b>corrected</b>
unicast-IP know and down	12				
unicast-IP unknown, but have rDNS in address block	6				
Conservative ground truth (52 + 35)	87		100%		
found by rDNS (30 + 35)	65	(oracle recall)	75%		
Realistic ground truth (37 + 35)	72		100%		
found by rDNS (30 + 35)	65	(higher bound recall)	90%		

Table 2: Evaluation of IN queries coverage compared to the AS112 providers list as ground truth.

all nodes are alive (a conservative choice). We confirm nodes are down two ways. First, when the list provides a unicast IP address for the node, we can confirm its presence with unicast DNS queries. Second, we probe the anycast prefix from 40 open BGP looking glasses and Merit’s BgpTables service [32] which provides 38 BGP peers. From each BGP peer we probe the well-known AS112 anycast prefix and search for the provider’s AS number in any AS paths; we interpret that presence as an active anycast node. For six providers, we have no rDNS server in their address blocks (as determined by whois); these are 6 of the 7 cases where BGP identifies an anycast node and rDNS does not. In the seventh case, the rDNS server in the provider was in the catchment of another AS, suggesting a complex network. These examples show that even extensive vantage points may not reach 100% coverage.

Using these methods, we confirm that 18 nodes in the list (26%) are no longer operational. Finally, turning to the subset of 52 nodes in the list that we cannot prove are down, we see that IN queries alone find 30.

From this analysis, there are two ways to define ground truth: a) the nodes on the list that are possibly alive (52), plus those found by rDNS but not on the list (30), or b) the nodes on the list known to be definitely alive (37), plus those found by rDNS but not on the list (30). Recall defined by (a) is 75%, while that defined by (b) is 90%. We argue that (a) is a conservative choice, and that the true recall is likely to be closer to 90% (since we were able to determine that 18 of the nodes are no longer operational).

The AS112 community has recently recognized the need for automated methods of node discovery, and have recently implemented an automated discovery method

<b>CHAOS queries:</b>	<b>F-Root</b>	<b>PCH</b>
authority truth	57	53
oracle truth	58	53
records considered ( $ \hat{A} $ )	216	215
estimated anycast nodes ( $ \hat{A} $ )	34	26
him true positives	21	26
overcounts	12 (0)	0
missing authority	1	0
extra	0	0
authority precision	64% (100%)	100%
oracle precision	65% (100%)	100%

Table 3: Accuracy of CHAOS queries without traceroute.

that also uses rDNS servers obtaining similar corrections to their public ground truth [6].

### 4.3 Precision for CHAOS Queries

While determining the ground truth (and therefore recall) for AS112 was challenging, CHAOS queries face a different challenge: since CHAOS queries are not standardized, providers adopt different conventions for labeling servers and nodes, and this can affect precision. To evaluate precision, we use our PlanetLab experiments on F-Root and PCH; this is the only set of vantage points from which we were able to issue both CHAOS queries and traceroutes, and precision can be affected by the choice of whether to use traceroutes are not.

Table 3 describes the precision of using *CHAOS queries alone* (without traceroute). PCH precision is 100%. F-root precision falls to 64%, mostly because of 12 overcounts. These overcounts are due to T3 or T4 configurations where multiple servers provide service for

Combined Method:	F-root	PCH
authority truth	57	53
oracle truth	58	53
records considered ( $ \hat{A} $ )	225	223
estimated anycast nodes ( $ \hat{A} $ )	27	26
true positives	21	26
overcounts	3 (0)	0
missing authority	2	0
extra	1	0
authority precision	88% (100%)	100%
oracle precision	89% (100%)	100%

Table 4: Accuracy of CHAOS queries augmented with traceroute.

a single node. Since ISC’s CHAOS records are per-server (not per-node), multi-server configurations result in overcounts.

CHAOS records also reveal one case of incomplete authority truth for F-root. Although missing from the public web page, ISC confirmed that the one anycast node we found should have been listed. This missing authority makes our oracle precision slightly better than authority precision, from 64% to 65%.

Our basic CHAOS algorithm does not interpret the contents of the reply, because there is no formal standard. However, each anycast service provider has their own convention, something we explore in Section 5.4. As an experiment, we decoded ISC’s convention, to extract identities of both the anycast node and the specific server. We show the results of this F-Root-aware CHAOS algorithm in parenthesis in Tables 3 and 4. This provider-specific interpretation makes the CHAOS method completely correct, suggesting it would be beneficial to standardize reply contents, or other means of making this distinction.

In the absence of being able to confirm provider-specific conventions, it is also possible to use traceroute to improve precision.

Combining traceroute with CHAOS queries introduces one new source of failure: if routing changes between the CHAOS observation and traceroute, analysis could incorrectly combine observations from different nodes. We detected these cases and identified them as *false combinations* removing them before analysis; they occurred primarily because our prototype took CHAOS and traceroute data hours apart. We plan to take CHAOS observations before and after traceroutes to automate detection of routing changes.

Table 4 measures how much our results improve by augmenting CHAOS queries with traceroute. Combining the two sources allows true positives to follow the larger of the two stand-alone methods for both targets. It reduces overcounts by 75% (3 instead of 12 or 13) for F-root, even without decoding F-root CHAOS replies, and eliminates overcounts for PCH.

These improvements translate into better precision for the combined method. For F-Root, precision rises to 88% (compared to 64% or 58% authority precision, with similar results for oracle precision), and PCH precision remains at 100%, the maximum of the single-source algorithms. Thus, because CHAOS conventions are not standardized, augmenting CHAOS queries with traceroute can improve precision significantly (from 65% to 89% for F-Root).

## 5. EVALUATION

Methods for identifying anycast server can help uncover anomalies in anycast configuration, characterize the level of deployment of anycast among root name servers and TLDs, and help us understand how anycast is managed as a service by providers for use by DNS root and TLD operators. This section demonstrates these capabilities.

### 5.1 Anomalous Anycast Configurations

**Root Masquerading:** While validating CHAOS queries on F-Root, we encountered an anycast server that was not on ISC’s list of F-Root anycast nodes, and which returned an empty CHAOS response. Discussions with ISC confirmed this site was *masquerading* as an F-Root anycast node—a non ISC server operating on the F-root IP address.

ISC described two general cases where third parties operate nodes at the F-Root anycast address. First, some organizations operate local copies of the root zone, and internally *masquerade* responses to `*.root-servers.org`. While ISC discourages this behavior, redirection of internal traffic is generally left up to the organization. Second, other organizations have attempted to hijack root DNS server from others, often to promote a modified root zone.

We observed this masquerading host from two vantage points inside CERNET, the China Education and Research Network. In both case the PR of the target is 202.112.36.246, at AS4538 in CERNET. The contents of the two zones appeared the same based on the SOA record, although we did not exhaustively compare the zones. ISC identified this non-ISC anycast node as a masquerading node, not a hijacked one, and we concur.

While this case represents a network provider choosing to handle requests from their own users using masquerading, nearly the same mechanisms can be used to detect hijacking. This potential illustrates the benefits of actively monitoring anycast services, at least until use of DNSsec becomes pervasive.

**In-Path Proxies and Others:** Beyond masquerading, our anycast server discovery methods can identify other abnormal configurations. We detected these anomalies when analyzing our Netylyzr dataset.



Total observations	61,914	100%		
expected replies	59,509	96.1%		
no reply	1,289	2.1%		firewall discards or routing failure
abnormal replies	1,116	1.8%		
observations have fake F-root CHAOS record	355	0.6%	[100%]	in-path proxies
Got facebook or non-existent-domain	354		[99.7%]	in-path proxies
neither facebook nor non-existent-domain	1		[0.3%]	in-path proxy or hijack/masquerade
observations got empty F-root CHAOS	761	1.2%	(100%)	
Got facebook or non-existent-domain	550	0.8%	(72%)	in-path proxies
no facebook and non-existent-domain	211			
got empty CHAOS records for all roots	93	0.15%		firewall or hijack/masquerade
got valid CHAOS records for some other roots	117	0.2%		hijack/masquerade
got fake CHAOS records for some other roots	1			in-path proxy

Table 5: Anomalies found for F-root CHAOS records in Netalyzr data.

While Netalyzr does not augment CHAOS queries with traceroute, it does include CHAOS queries to each root server, IP resolution requests for `www.facebook.com` and for a non-existent domain name `RANDOM.com` (where `RANDOM` is string longer than 40 characters, that triggers a non-existent domain error message). It also reports when the CHAOS queries timeout without response; we ignore these cases. We next use this information, with additional manual probing, to infer possible root causes of these abnormal responses for F-root.

In this dataset, we see two abnormal responses to CHAOS queries: incorrect CHAOS records and missing CHAOS records, making up about 1.8% of the observations (Table 5). We believe these observations detect *in-path proxies*. Usually end-systems are configured to use a local DNS resolver. An in-path proxy is a network middlebox that captures and redirects all DNS traffic directly. We believe that incorrect F-root CHAOS records (355 cases, 0.6%) indicate in-path proxies that modify CHAOS queries, since we know all F-root nodes provide correct CHAOS records. We believe these are in-path proxies because almost always (354 of the 355 cases) the client also gets a direct reply for Facebook from the supposed-F-root node. A true F-root server would not have directly responded with an entry for Facebook, but would have redirected to the `.com` DNS server. (The one case that omits Facebook is located in China, where Facebook is blocked.)

Empty CHAOS records occur more often (761 cases, 1.2%). In most of these cases (550, 72% of 761, 0.8% of all), we observe Facebook or non-existent domain replies, suggesting an in-path proxy for the same reasons as above. However, in some of these cases, we see an empty CHAOS record for F-Root, but also get neither a Facebook nor the non-existent domain reply. In some cases we get empty CHAOS records for all roots, in others we see some valid and other invalid CHAOS records. Without additional data (like traceroutes) we cannot diagnose these problems with certainty. We believe they are either firewalls or masqueraders.

DNS root servers	measured	published	found
A (Verisign)	4 <	<b>6</b>	66%
B (ISI)	1 =	1	100%
C (Cogent)	6 =	6	100%
D (Univ. of Maryland)	1 =	1	100%
E (NASA)	9 >	1	900%
F (ISC)	<b>53</b> >	49	108%
G (DISA)	6 =	6	100%
H (U.S. ARL)	<b>3</b> >	2	150%
I (Automica)	<b>58</b> >	38	153%
J (Verisign)	59 <	<b>70</b>	84%
K (RIPE)	17 <	<b>18</b>	94%
L (ICANN)	78 <	<b>107</b>	73%
M (WIDE)	6 =	6	100%

Table 6: Comparing measured against published numbers of anycast nodes for all anycast root servers.

To summarize, in about 62k unique IP addresses using Netalyzr, our data suggests that 0.2% appear to be behind potentially masquerading F-root nodes, while 1.4% (0.6% + 0.8%) see in-path proxies, and about 0.15% see other unusual behavior. These observations suggest that DNS manipulation is not common, but does occur. They also suggest the need for external monitoring as our IN-queries, and for additional information to disambiguate these cases, as with our use of traceroute.

## 5.2 Characterizing Anycast in Other Roots

We have confirmed that with nearly 62k vantage points, CHAOS queries discover almost all F-root nodes. We next examine other anycast root servers discovered by Netalyzr clients, and compare our findings to public records [37].

We began by examining the CHAOS record formats for each root, finding that 9 use CHAOS records that embed location information in the string, while 2 have location information in some records but not all. Our measurements above assume providers use unique CHAOS strings.

In Table 6 we compare the number of measured any-

CHAOS (# recs.)	traceroute (number of PRs)		
	> 1	1	0
> 1	anycast; T4 unicast	anycast; T3 unicast	anycast; T3 unicast
1	T2 unicast; mis-config anycast	unicast	unicast; mis-config anycast
0	non-BIND anycast; T2/T4 unicast	unicast	insufficient information

Table 7: Interpretation of CHAOS queries and traceroute on TLD nameservers.

cast nodes, from CHAOS queries with 62k Netalyzr vantage points, against the published number from `root-servers.org`. We expect 10 of the 13 to use anycast. In 3 of the 10 cases (F, H, and I) we detect anycast nodes not reported, suggesting public information is out-of-date, omitting up to 20 nodes. In addition in one case, E, we find that it uses anycast although that use is not published. Examination of the CHAOS strings suggests that NASA has outsourced E-root anycast to PCH. In 4 cases (A, J, K and L), we miss some nodes, either because recall with Netalyzr is not perfect, or because the published list is out-of-date. Finally, for three cases (C, G, and M), each with relatively few nodes, we find all.

### 5.3 Anycast Use in Top-Level Domains

Anecdotal evidence suggests that anycast is widely used in DNS, but to our knowledge there has been no systematic study of *how extensively* it is used. In this section, we determine how many TLDs use anycast by using CHAOS queries (with traceroute) on PlanetLab. Although PlanetLab’s recall is low, that should not affect the results discussed in the section since we are not trying to enumerate all of the anycast servers in each TLD. Rather, we try to determine if more than one server responds to a CHAOS query sent to a TLD nameserver.

**Target:** The targets for our study are the authoritative name servers for the country-code top-level domain names (ccTLDs), and the generic TLDs (gTLDs), as listed by IANA [25]. Together there were 1133 IP addresses providing TLD nameservice in April 2012.

**Methodology:** We use CHAOS queries and traceroute against each IP address for each name server, querying from 240 PlanetLab vantage points. (We omit IN queries because, until further standardization, only AS112 supports them.) We collected data on May 2011 and April 2012, and present the data collected on 2 April 2012. (We see similar results on other days in 2012, and fewer in 2011.)

In Table 7 we interpret these results to identify definite and possible anycast services, since in this case

2012 April Results				
CHAOS (# recs.)	traceroute (# PRs)			(definite, possible) anycast
	> 1	1	0	
> 1	255 (238, 0)	14 (3, 0)	159 (0, 159)	(241, 159)
1	99 (2, 1)	117 (0, 2)	312 (0, 0)	(2, 3)
0	44 (0, 44)	32 (0, 0)	101 (0, 0)	(0, 44)
total TLD name servers and anycast:				1133 (243, 206)

Table 8: Anycast discovered for TLD name servers. The first number in braces is definite anycast, the second number in braces is possible anycast.

Number of TLD names	definite anycast	possible anycast	higher bound
314 (100%)	<b>177 (56%)</b>	48	<b>225 (72%)</b>

Table 9: Anycast services discovered for TLD names, obtained by ACE.

there is no ground truth. Of these cases  $CHAOS > 1 \wedge PR > 1$  is the strongest evidence for anycast, though our combined method still finds a few T4 unicast cases. The other cases where  $CHAOS > 1$  or  $PR > 1$  are likely partially observed anycast addresses. We classify these results in three ways. *Definite anycast* means our method finds multiple nodes. *Possible anycast* means there are multiple records but we cannot guarantee anycast, such as when  $CHAOS == 0 \wedge PR > 1$  or  $CHAOS > 1 \wedge PR == 0$ . Finally, *definite unicast* means the method confirms that there is only one node.

**Results:** Table 8 shows our results of anycast deployment in TLD name servers. We report definite anycast as the first number in braces, and the second number is possible anycast. We observe that about 21% (243 of 1133) of TLDs nameservers definitely use anycast, while another 18% (206 of 1133) are possibly anycasted. If we adopt definite anycast as a lower bound and definite plus possible as an upper bound, then at least 21% and perhaps 39% of TLDs nameservers use anycast.

A complementary view classifies use of anycast by the *name* of the TLD, rather than by IP address. As there are always several authoritative name servers for a top level domain name, we count a TLD name as definitely anycasted if at least one of its authoritative name servers is definitely using anycast. Table 9 shows anycast deployment in TLD names. When there are no definite anycasted name servers, but at least one is possible anycast, then we count the TLD name as a possible anycast. We see that at least 56% of the TLD names are definitely anycast, and 72% of TLD names possibly so. Thus, more than half and perhaps three-quarters of TLDs include anycast as part of their DNS service.

The main implication of these findings is that anycast is an important component of the DNS, and needs to be continuously monitored for abnormal configurations, masquerading or, worse, hijacking (Section 5.1).

#### 5.4 How Many Anycast Providers Exist?

In the operational Internet, several providers provision anycast services; TLD managers operate their own servers or use these providers. Thus, a single anycast provider may support multiple TLDs. In this section, we measure the number of anycast providers, and the distribution of TLDs across these providers. We next use our data of CHAOS queries to all TLDs from Section 5.3 to explore these providers. Our results here depend on CHAOS naming conventions, and do not require full enumeration, so they are not hindered by so the moderate recall from PlanetLab’s few VPs.

To identify anycast providers, we review the CHAOS queries to confirmed anycast nodes; we study 243 definite anycast nodes (Section 5.3).

To go from services to providers, we examine the patterns in their replies. We identify a potential provider based on either a unique pattern, or a provider-specific identifier in a standard pattern. For example, several organizations include the provider’s domain name in their reply, while others use distinctive patterns. We see two general patterns: in the most common (22 likely providers found), the reply uses hostname-like strings, often encoding geographic information along with server and node identity. Examples of this format include *lax1b.f.root-servers.org* for a server *b* at an IXP-1 in Los Angeles, and *host.bur.example.net* for server *host* at an IXP near Burbank airport. The second format we identify, with 10 likely providers, is even more provider specific, with just a serial number *example1* for server 1 by provider *example*, or server plus a geographic code *s1.lax* for server 1 in Los Angeles. From these kinds of patterns we find 32 unique providers in the entire set. This count represent a likely lower bounds: “likely” since it seems unlikely for providers to use very dissimilar patterns, and a lower bound since the second format is general enough that two providers may have adopted the same scheme coincidentally.

Figure 4 shows how many services each provider operates. We see that some providers are unique to one service (about two thirds, 20 of 32). A few large providers operate many services, with the top two providers (PCH and DynDNS) accounting for more than 58% of services.

## 6. SECURITY IMPLICATIONS

Root DNS operations are critical infrastructure, so we next explore security implications of our approach. Some anycast providers consider *any* details of their infrastructure proprietary, to avoid giving information to attackers or competitors. Second, attackers can use

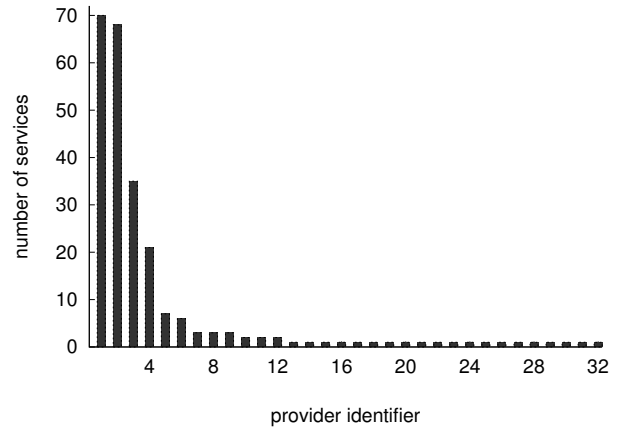


Figure 4: Estimates of number of services (anycast IP addresses) operated by each estimated provider (as identified by CHAOS response patterns).

our discovery methods to masquerade or hijack anycast services. We discuss solutions to these security threats in the rest of this section.

### 6.1 Limiting diagnosis to the provider

While some anycast providers welcome open diagnostic tools, discussions of our proposed diagnosis mechanisms on IETF mailing lists suggest that several providers require the ability to limit their use. One concern is disclosure of details of anycast operation to competitors. A second concern is that diagnostic information may assist on attacks on anycast infrastructure. For example, disclosure of the unicast address for an anycast server may enable a targeted denial-of-service attack.

The challenge in limiting access to diagnostic information is that diagnosis requires probing from many and possibly unknown public sites, such as PlanetLab nodes or rDNS. Traditional access control, such as a whitelist implemented by a firewall, is therefore likely porous and not completely effective, difficult to maintain, and may also reduce diagnosis accuracy.

Instead, we propose *coordinated, changing private query names*. To restrict access, providers can move information under a private name, so instead of placing `_node-id` in `_ns-diagnostics.example.org` (discussed in Section 3.2), it instead is stored under: `nonce._ns-diagnostics.example.org`, where `nonce` is a time-changing value known only to the provider and their authorized queriers. One way to compute the nonce is as a cryptographic hash of the time since an epoch concatenated with a secret value. Time provides a globally changing value; it should be rounded to a reasonable lifetime for the nonce (say, a few minutes), and both the current and prior nonce could be active to avoid requiring tight time synchronization. An attacker will not

know the secret, so they can do nothing without snooping query traffic. An attacker can easily snoop traffic (say, by running an open recursive nameserver), but an attacker can only masquerade as the server during the nonce’s relatively brief lifetime.

We are implementing nonce-based access in BIND-9.

## 6.2 Discouraging Masquerader spoofing

A second threat is masqueraders who may attempt to pretend to be a legitimate anycast nodes. The masquerader will likely receive queries from our diagnosis system. The masquerader then has two possible actions to prevent itself from being identified. It could discard the probe and not reply, or it could generate its own reply, possibly the response from a valid node.

To protect against non-replying masqueraders, all legitimate anycast nodes must reply, and diagnostic queries must be retried several times to rule out packet loss.

Protection from reply replay is difficult, since a masquerader could forward the reply to a legitimate node out-of-band. If necessary, replays could be prevented by assigning each legitimate anycast node a public key and using a cryptographic challenge-response protocol.

## 6.3 Relationship to DNSsec

DNSsec provides origin authentication and data integrity in DNS [4]. In development for more than a decade, it has recently been deployed on root domains. However, DNSsec deployment does not fully address the problems we explore. By cryptographically insuring the integrity of answers, DNSsec provides end-to-end validation of DNS contents. Our work complements this role by providing diagnostic tools for DNS providers who use anycast. In addition, we provide auditing tools for the end user to assess service quality. We also provide tools to detect masquerading, helping identify some cases of possibly unexpected traffic diversion (although not guaranteeing query confidentiality).

## 7. RELATED WORK

While there has been significant work exploring the DNS performance and anycast use in root nameservers, to date there has been little work exploring anycast discovery, at least outside the operational community. We review that work, and broader related work in route hijack detection.

**Anycast discovery:** The DNS operational community has developed several techniques to support anycast diagnostics. The CHAOS query was first defined in RFC-4892 [45], and although originally developed in the BIND implementation of DNS, the approach is now supported in other DNS server software (see Section 3.1 for a partial list). We carefully validate the precision and recall of this method, and suggest ways to improve its precision using traceroute.

Subsequent standards activity has suggested the need for additional diagnostic methods. RFC-5001 [7] defines a new NSID (name server identifier) option for DNS. By using a new option, it differs from RFC-4892 in specifying that rDNS will not forward NSID requests. Although the RFC explores several possible payloads NSID could return, it explicitly defers standardizing contents to future work. A recent Internet Draft proposes using unique-per-node AS numbers for anycast node identification [31]. When this method is widely deployed, it can be used for anycast enumeration. We expect that analysis of recall will apply here as well.

Complementary to our enumeration of anycast servers, Gibbard relies on published root and TLD server information to analyze their geographic distribution [20]. Our work focuses on automatic instead of manual methods to identify nodes, but we do not geolocate nodes.

**Root nameserver performance:** Complementary to our work is a rather large body of work on measuring the proximity (client-to-server latency), affinity (the stability of client-server association), and load balancing for DNS anycast. In general, methods to study proximity compare anycast query latency with unicast latency to anycast servers from several vantage points [9, 14, 38]. However, at least one piece of work has explored proximity by measuring server-side accesses by clients, and geolocating clients to estimate latency [30]. Several pieces of work have explored affinity by periodically probing anycast servers to determine when routing changes cause anycast packets to be routed to a different server [8, 12, 9]. Boothe et al. observe that anycast affinity measurement techniques can be used as a lightweight approach to understand BGP dynamics, since anycast routes are propagated using BGP [11]. Finally, load balancing is assessed by measuring client accesses at anycast servers [9, 10].

Our work is inspired by these works, but differs in several respects. While other work has explored the use of CHAOS records to study affinity [12, 11, 8, 39], we extensively validate CHAOS query use for anycast server enumeration and use it to characterize the use of anycast in TLDs. Most prior work listed above have used hundreds of vantage points, usually from PlanetLab; as our work shows, anycast recall is modest at the scale of PlanetLab implying that the conclusions drawn by prior work may need to be revisited. One exception is the work of Ballani et al. who have used 20,000 rDNS servers [9]; our evaluations contain an order of magnitude larger vantage points. Finally, the use of IN-class records for identifying anycast servers is not new; it has been used in AS112, and Ballani et al. [9] use a similar mechanism to study anycast load balance in a controlled setting. Our primary contribution is a concrete proposal to standardize anycast identification using IN queries, and a careful characterization of its

recall properties.

**Route Hijack Detection:** Closest to our study of DNS masquerading is work on general route hijacking and protection of routing to name servers. Before widespread deployment of anycast, Wang et al. [43] proposed a BGP path-filtering method to protect routes to critical TLD name servers. Others have discussed the importance of detecting hijacked *unicast* prefixes [34], and proposed methods for hijack detection using the control plane [28], data plane [47, 46, 36], and hybrid control-and-data plane approaches [23]. Detecting anycast hijacking is qualitatively harder than detecting unicast hijacking, since by definition, anycast packets can be sent to one of many destinations, one or more of which may be suspect while with unicast routing any examples of multiple destinations are illegitimate.

We also compare our work to DNSsec in Section 6.3.

## 8. CONCLUSIONS

Through its wide use in DNS, anycast has become an indispensable part of the Internet. We developed new methods that combine CHAOS queries with tracersoutes, or use new IN records to support tens of thousands of open recursive DNS servers as vantage points. We find our methods have generally good precision and high recall. In particular, we find that the topological dispersion of anycast requires a very large number of vantage points to enable high recall; on average, 10,000 vantage points are required for a recall of 80%. Finally, our studies of F-Root and PCH anycast infrastructure detect one third-party site masquerading as an anycast node, reveal several abnormal anycast configurations, and our evaluation of all country-code and generic top-level domain servers shows anycast is possibly used by 72% of the TLDs.

## Acknowledgments

We thank ISC and PCH for their openness and for their patience answering our questions about their infrastructure: Paul Vixie, Leo Bicknell and Suzanne Woolf at ISC, and Bill Woodcock and Ross Stapleton-Gray at PCH. We thank PlanetLab for its service, and Brad Karp and John Andrews at UCL, and Renata Teixeira at lip6 for help at their sites. We thank Dan Massey (CSU) for comments about DNSsec. We thank Nichoals Weaver for adding CHAOS probing to Netalyzr, and him, Veron Paxson, and Christian Kreibich for providing us Netalyzr data. We thank Duane Wessels and The Measurement Factory, Inc. [42] for information about rDNS, and William F. Manton Sotomayor for help about AS112.

## 9. REFERENCES

- [1] J. Abley and K. Lindqvist. Operation of anycast services. *RFC 4786*, 2006.
- [2] J. Abley and W. Maton. AS112 nameserver operations. RFC 6304, Internet Request For Comments, July 2011.
- [3] Joe Abley. Hierarchical anycast for global service distribution. Technical Report ISC-TN-2003-1, ISC, March 2003.
- [4] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS security introduction and requirements. *RFC 4033*, 2005.
- [5] AS112 Project. AS112 Server Operators Listing. <http://public.as112.net/node/10>, 2012.
- [6] AS112 Project. How many public AS112 nodes are there as of March 31, 2012? <http://public.as112.net/node/30>, April 2012.
- [7] R. Austein. DNS Name Server Identifier (NSID) Option. *RFC 5001*, 2007.
- [8] H. Ballani and P. Francis. Towards a global IP anycast service. In *Proc. of ACM SIGCOMM*. ACM, August 2005.
- [9] Hitesh Ballani, Paul Francis, and Sylvia Ratnasamy. A measurement-based deployment proposal for IP anycast. In *Proc. of ACM IMC*, pages 231–244, Rio de Janeiro, Brazil, October 2006.
- [10] Piet Barber, Matt Larson, Mark Kosters, and Pete Toscano. Life and Times of J-root. In *NANOG 34*, October 2004.
- [11] Peter Boothe and Randy Bush. Anycast measurements used to highlight routing instabilities. In *NANOG 34*, May 2005.
- [12] Randy Bush. DNS Anycast Stability: Some Initial Results. CAIDA/WIDE workshop, <http://www.caida.org/workshops/wide/0503/slides/050311.wide-anycast.pdf>, March 2005.
- [13] Lorenzo Colitti. Effect of anycast on K-root. DNS-OARC Workshop, July 2005.
- [14] Eric Cronin, Sugih Jamin, Cheng Jin, Anthony R. Kurc, Danny Raz, and Yuval Shavitt. Constrained mirror placement on the Internet. *IEEE JSAC*, 20(7):1369–1383, September 2002.
- [15] John Dille, Bruce Maggs, Jay Parikh, Harald Prokop, Ramesh Sitaraman, and Bill Weihl. Globally distributed content delivery. *IEEE Internet Computing*, 6(5):50–58, September 2002.
- [16] Robert Engel, Vinod Peris, Debanjan Saha, Erol Basturk, and Robert Haas. Using IP anycast for load distribution and server location. In *Proc. of Global Internet*, December 1998.
- [17] Xun Fan, John Heidemann, and Ramesh Govindan. Improving diagnostics of name server anycast instances (draft-anycast-diagnostics-01.txt). discussed on [dnsop@ietf.org](mailto:dnsop@ietf.org) mailing list and at <http://www.isi.edu/~xunfan/research/draft-anycast-diagnostics.txt>, October 2011.

- [18] Xun Fan, John Heidemann, and Ramesh Govindan. Characterizing anycast in the domain name system (extended). Technical report, Information Sciences Institute, May 2012.
- [19] Steve Gibbard. Geographic implications of DNS infrastructure distribution. *The Internet Protocol Journal*, 10(1):12–24, 2007.
- [20] B. Greene and D. McPherson. ISP security: Deploying and Using Sinkholes. NANOG talk, <http://www.nanog.org/mtg-0306/sink.html>, June 2003.
- [21] T. Hardie. Distributing authoritative name servers via shared unicast addresses. *RFC 3258*, 2002.
- [22] Xin Hu and Z. Morley Mao. Accurate real-time identification of IP prefix hijacking. In *Proc. of IEEE Symposium on Security and Privacy*, 2007.
- [23] C. Huitema. An anycast prefix for 6to4 relay routers. *RFC 3068*, 2001.
- [24] Internet Assigned Numbers Authority. Root zone database. <http://www.iana.org/domains/root/db/>.
- [25] Dina Katabi and John Wroclawski. A framework for scalable global IP-anycast (GIA). In *Proc. of ACM SIGCOMM*, pages 3–15, Stockholm, Sweden, August 2000. ACM.
- [26] Christian Kreibich, Nicholas Weaver, Boris Nechaev, and Vern Paxson. Netalyzr: Illuminating the edge network. In *Proc. of ACM IMC*, pages 246–259, 2010.
- [27] Mohit Lad, Dan Massey, Dan Pei, Yiguo Wu, Beichuan Zhang, and Lixia Zhang. Phas: A prefix hijack alert system. In *Proc. of USENIX Security Symposium*, 2006.
- [28] LANDER. LANDER: Los Angeles Network Data Exchange and Repository. Project website <http://www.isi.edu/ant/lander>, 2004.
- [29] Ziqian Liu, Bradley Huffaker, Marina Fomenkov, Nevil Brownlee, and kc Claffy. Two days in the life of the dns anycast root servers. In *Passive and Active Network Measurement*, pages 125–134, 2007.
- [30] Danny McPherson, Ryan Donnelly, and Frank Scalzo. Unique per-node origin ASNs for globally anycasted services. *Active Internet-Draft*, November 2010.
- [31] Merit Network, Inc. bgptables. <http://bgptables.merit.edu/>.
- [32] P. Mockapetris. Domain names—concepts and facilities. RFC 1034, Internet Request For Comments, November 1987.
- [33] Ola Nordström and Constantinos Dovrolis. Beware of BGP attacks. *ACM SIGCOMM Computer Communication Review*, 34(2):1–8, 2004.
- [34] C. Partridge, T. Mendez, and W. Milliken. Host anycasting service. *RFC 1546*, 1993.
- [35] Tongqing Qiu, Lusheng Ji, Dan Pei, Jia Wang, Jun Xu, and Hitesh Ballani. Locating prefix hijackers using lock. In *Proc. of USENIX Security Symposium*, pages 135–150, 2009.
- [36] ROOT-DNS. <http://www.root-servers.org/>.
- [37] Sandeep Sarat, Vasileios Pappas, and Andreas Terzis. On the use of anycast in DNS. Technical report, Johns Hopkins University, 2004.
- [38] Yuji Sekiya. Passive and active DNS measurement: update. CAIDA/WIDE workshop, <http://www.caida.org/workshops/wide/0503/slides/sekiya.pdf>, March 2005.
- [39] Karen Sollins and Larry Masinter. Functional requirements for uniform resource names. RFC 1737, December 1994.
- [40] The Domain Name System Operations Analysis and Research Center. The AS112 Project. <http://www.as112.net>.
- [41] The Measurement Factory, Inc. <http://www.measurement-factory.com/>.
- [42] Lan Wang, Xiaoliang Zhao, Dan Pei, Randy Bush, Daniel Massey, and Lixia Zhang. Protecting BGP routes to top-level DNS servers. *IEEE Transaction on Parallel and Distributed Systems*, 14(9):851–860, September 2003.
- [43] Wikipedia. Comparison of DNS server software. [http://en.wikipedia.org/wiki/Comparison\\_of\\_DNS\\_server\\_software](http://en.wikipedia.org/wiki/Comparison_of_DNS_server_software).
- [44] S. Woolf and D. Conrad. Requirements for a mechanism identifying a name server instance. *RFC 4892*, 2007.
- [45] Zheng Zhang, Ying Zhang, Y. Charlie Hu, Z. Morley Mao, and Randy Bush. iSPY: Detecting IP Prefix Hijacking on My Own. In *Proc. of ACM SIGCOMM*, pages 327–338, August 2008.
- [46] Changxi Zheng, Lusheng Ji, Dan Pei, Jia Wang, and Paul Francis. A light-weight distributed scheme for detecting IP prefix hijacks in real-time. In *Proc. of ACM SIGCOMM*, August 2007.