Impact of Configuration Errors on DNS Robustness

Vasileios Pappas
UCLA Computer Science
vpappas@cs.ucla.edu

Zhiguo Xu
UCLA Computer Science
zhiguo@cs.ucla.edu

Songwu Lu
UCLA Computer Science
slu@cs.ucla.edu

Daniel Massey
Colorado State University
massey@cs.colostate.edu

Andreas Terzis
Johns Hopkins University
terzis@cs.jhu.edu

Lixia Zhang
UCLA Computer Science
lixia@cs.ucla.edu

ABSTRACT

During the past twenty years the Domain Name System (DNS) has sustained phenomenal growth while maintaining satisfactory performance. However, the original design focused mainly on system robustness against physical failures, and neglected the impact of operational errors such as misconfigurations. Our recent measurement effort revealed three specific types of misconfigurations in DNS today: lame delegation, diminished server redundancy, and cyclic zone dependency. Zones with configuration errors suffer from reduced availability and increased query delays up to an order of magnitude. Furthermore, while the original DNS design assumed that redundant DNS servers fail independently, our measurements show that operational choices made at individual zones can severely affect the availability of other zones. We found that, left unchecked, DNS configuration errors are widespread, with lame delegation affecting 15% of the DNS zones, diminished server redundancy being even more prevalent, and cyclic dependency appearing in 2% of the zones. We also noted that the degrees of misconfiguration vary from zone to zone, with most popular zones having the lowest percentage of errors. Our results indicate that DNS, as well as any other truly robust large-scale system, must include systematic checking mechanisms to cope with operational errors.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Reliability, availability, and serviceability; C.2.2 [**Computer-Communication Networks**]: Network Protocols—*DNS*

General Terms

Measurement, Reliability, Performance

Keywords

DNS, Misconfigurations, Resiliency

1. INTRODUCTION

The Domain Name System (DNS) is one of the most successfully designed and operated Internet services today. It provides a fundamental service for end users, i.e., name resolution, and is used by a number of different applications ranging from load balancing to service discovery. In essence, DNS is a global scale hierarchical database, managed in a fully distributed manner. It seeks to provide acceptable performance without setting any limit on the size of the database [24]. According to [2], the number of host records (only one of the record types carried by DNS) has grown from 20,000 in 1987 to 171,638,297 in January 2003. The number of independent zones has skyrocketed from a handful in 1982 to around 60 million in 2003. Despite such phenomenal growth in size, DNS has been able to deliver satisfactory performance at the user level. Jung *et al.* [15] showed that a DNS query is answered by sending 1.3 messages on average and the mean resolution latency is less than 100ms. Moreover, DNS has been able to meet unforeseen challenges. For example, DNS survived widely publicized DDoS attacks targeted at the root nameservers in October 2002 [11], demonstrating the robustness of its distributed design against brute-force attacks.

Despite its tremendous success, DNS is not without limitations. If one puts a higher standard on DNS resilience, its design warrants further examination, as evidenced by the following example. During January 2001 all the authoritative servers for the Microsoft DNS domain became inaccessible [10]. Despite the fact that the need for geographically dispersed DNS servers is well documented [8], Microsoft placed all its DNS servers behind the same network switch which failed. Furthermore, what followed this failure was largely unexpected. The number of DNS queries for the Microsoft domain seen at the F root server surged from the normal 0.003% among all the queries to over 25%. As explained later in the paper, DNS implementation choices can introduce a coupling between the reachability to a zone's DNS servers and the load at the top level DNS servers. Had the fault persisted, or had additional domains failed simultaneously, the additional load might have overwhelmed the F root server.

The previous example illustrates that local decisions in configuring a specific DNS zone can have globally adverse impact. In this work, we study the effect of misconfigurations on DNS robustness and performance. We used a combination of passive and active measurements collected over a six-month period to study the type and the extent of misconfigurations observed in the global DNS infrastructure, and the impact these misconfigurations have on DNS query response times and service availability. The passive measurement traces were collected from a typical university campus environment and recorded about 3 million queries sent to over 55,000 distinct zones. The active measurements were done by querying a sample set of DNS zones randomly selected from the ISC reverse zone files [2].

Our key contributions and results can be summarized as follows.

SIGCOMM'04, Aug. 30–Sept. 3, 2004, Portland, Oregon , USA.

First, our analysis shows that on average 15% of the DNS zones suffer from a specific misconfiguration called *lame delegation*, in which the parent of a DNS zone points to wrong name servers for the child zone. We found that 70% of such lame delegation cases reduced the number of available name servers for a zone by half. Because DNS queries sent to non-existing servers timeout, and have to be resent to a different server, lame delegation translates to increased query time. While DNS queries are normally resolved in 100ms, queries sent to lame delegated servers resolve in 3 seconds on average, a thirty-fold increase in response time.

Second, while DNS design documents [8, 14] call for diverse placement of a zone's authoritative servers, our measurements show that configuration errors can lead to *diminished server redundancy*. We found that in 45% of the measured zones, all the servers reside under the same /24 address prefix. This means that any routing anomaly affecting that prefix will disrupt services to the DNS zone. We also discovered that 82% of the zones place their servers in the same geographical region and 77% of the zones have all their servers in a single Autonomous System (AS). Servers placed in the same geographic region are vulnerable to external failures, such as fiber cuts or power outages. Similarly, servers placed within the same AS can be simultaneously affected by failures within the AS.

Third, our study discovered a previously unknown type of misconfiguration, *cyclic zone dependency*. In this type of misconfiguration, information required to resolve a name in zone X depends on information in zone Y which in turn depends back on zone X. While the percentage of the zones involved in such misconfigurations seems relatively small, 2% based on our measurements, these cases are much harder to detect. Moreover, cyclic zone dependencies reduce a zone's availability and may substantially increase query response times. Our results show that, for zones involved in cyclic zone dependencies, the error may cause more than 25% of the zone's servers to become unreachable.

Overall, our findings indicate that even for a seemingly well-functioning Internet system like DNS, human errors such as misconfigurations exist to a surprising extent. As a result, the degree of system resiliency for many zones is not as high as the number of redundant servers indicates. Such unanticipated reduction of resiliency may incur grave consequences when unexpected failures occur. Enhancing DNS resilience requires systematic detection and reaction mechanisms to handle operational errors.

The rest of the paper is structured as follows. In Section 2 we give a brief overview of the DNS design. We present the methodology used to measure the different types of DNS misconfigurations in the global DNS system in Section 3. In Section 4 we describe results from our measurements. We discuss the implications of our findings in Section 5, and compare with the related work in Section 6. We conclude the paper in Section 7.

2. DNS BACKGROUND

Developed in the 1980s, the primary goal of the Domain Name System (DNS) [21, 22] is to provide a robust and scalable name-to-address mapping service. DNS data is stored using a simple data structure called a *Resource Record* (RR). A RR can be viewed as a tuple $< name, TTL, class, type, data >$, where $name$ is a fully qualified domain name (FQDN) such as *www.google.com*, TTL is the time-to-live value in seconds, $class$ is typically Internet (IN), $type$ identifies the resource record format, and $data$ contains the format-specific data. Examples of common RR types include A RRs that store IP addresses, MX RRs that identify mail servers, PTR RRs that map IP addresses to names, and NS RRs that store name server information. DNS makes no distinctions between different types of data records, such as MX RRs used by end-user applications, and *infrastructure* RRs, such as NS RRs, used only to establish and navigate the DNS hierarchy.

The DNS database is organized as a tree to match the hierarchical nature of Internet names. Scalable management of this large-scale and continuously growing namespace is achieved via a distributed management hierarchy. Below the DNS root are a number of Top Level Domains (TLDs). The root delegates the management of each TLD to a specific administrative body, which in turn delegates subdomain allocations under that TLD. The TLDs are divided into two classes: generic TLDs (gTLDs) such as *com, edu, biz,* and country-code TLDs (ccTLDs) such as *uk, jp, nl.* There is also a special purpose TLD, the *arpa* domain, which is used for the reverse mapping of IP addresses to names. No limit is set on either the depth of the tree or the number of branches at each node. A DNS *zone* is defined as one or multiple domains, which *i)* occupies a continuous subspace in the DNS name tree and *ii)* is under the same administrative control and served by the same set of *name servers*. In the remainder of this paper, we refer to name servers simply as servers. The servers specified by the owner of the zone store the *authoritative* data for all the names in that zone.

Operators of each zone determine the number of authoritative servers and their placement, and manage all changes to the zone's authoritative data. Each authoritative server of the same zone should have an identical copy of the zone data, thus the zone data is available as long as any authoritative server is reachable. The best current practice recommends placing authoritative servers in diverse locations for maximum protection against network failures [14].

Although zone administration is autonomous, some inter-zone coordination is required to maintain the DNS hierarchy. In particular, the parent zone must provide information on how to reach its children's servers[1]. Each child provides its parent with a list of authoritative servers, more precisely an NS RRset for the child zone. The parent stores this NS RRset, and refers the resolvers to the child zone by including this NS RRset in the responses. Ideally, the NS RRset stored at the parent should match the child's set of authoritative servers exactly, although the DNS design provides no mechanism to ensure this consistency. DNS continues to work as long as the parent NS RRset correctly identifies at least one authoritative server.

3. MEASUREMENT METHODOLOGY

The DNS design, as described in Section 2, assumes that operators correctly configure and manage the zones. In particular, reliable DNS operations depend on the following correct actions: appropriate placement of redundant servers for high availability, manual input of each zone's database for correct setting, and coordination between parent and child zones for consistency. We broadly define human errors in any of these actions as *configuration errors*.

It is well known in the operational community that DNS configuration errors exist [13, 20]. However, there has been no systematic study to quantify their pervasiveness and impact. In this work, we conducted large-scale measurements to assess the pervasiveness of configuration errors, their effect on the user-perceived performance, and their impact on DNS robustness. We conducted two types of measurements, passive and active as described next.

3.1 Passive Measurements

We collected two sets of DNS traces from the UCLA Computer Science department network. The first set was collected during the week of 8/29/03–9/5/03, and the second set during 9/27/03–

[1] The records containing this information are called *glue records* in DNS parlance.

	Trace Period	Number of Queries	Number of Responses	Level 2 Domains
Trace 1	08/29/2003 - 09/05/2003	2,470,370	2,284,744	54,564
Trace 2	09/27/2003 - 10/04/2003	3,097,028	2,693,907	56,058

Table 1: **DNS packet traces used for the passive measurements**

	Number of Zones	Type of Sampling
Sample 1	51,515	Random Sampling
Sample 2	18,522	Zones from Sample 1 that allow zone transfer
Sample 3	500	Zones hosting the 500 most popular web servers

Table 2: **DNS zones used for the active measurements**

10/4/03. Table 1 summarizes the total number of queries and responses in each set, as well as the total number of second level domains queried. We use the last two labels of a domain name to estimate the number of second level domains[2].

Our data collection observes DNS packets sent over the department's external links and captures all the DNS packets exchanged between the department and external sites. We count only the DNS traffic exchanges with external sites; we exclude the local DNS traffic between end hosts and the local caching servers. We also exclude all DNS queries sent to the department's authoritative servers. We measure the delay between the first query packet and the final response. In other words, we measure the delay associated with obtaining an answer that is not present in the local server's cache. We also analyze the content of each intermediate response packet to detect whether it reflects any configuration errors.

Given that the data-set is taken from a university environment, it is possible that there is a bias in the interests of the observed user population. Therefore, the visited zones may not be a representative sample of the entire DNS space, and the number of configuration errors among all the visited zones may not give a good quantitative estimate on the degree of configuration errors in general. However, for all the cases of DNS configuration errors that we identified, our measured delay should serve as a good estimate of the DNS query delay in the presence of configuration errors.

3.2 Active Measurements

To overcome the potential bias in the passive measurement and to gauge the pervasiveness of DNS configuration errors, we also conducted active measurements. We implemented a specialized DNS resolver and used it to query a randomly selected subset of the DNS namespace. Our resolver added the following additional features on top of the standard resolver function. First, when it receives a referral for zone Z with a list of DNS servers for Z, it sends a query to *each* of the servers to verify whether all of them can provide correct replies. Second, it attempts to make use of the DNS zone transfer functionality to retrieve the entire zone data which allows us to determine the number of delegations and compare the results for the various delegations. We utilized external information, such as BGP tables [5] and geographic location information [4], to estimate the topological and geographic locations of each zone's authoritative servers.

Our active measurements use three sample sets of DNS zones. To create Sample 1, we used the ISC reverse zone data [2] to obtain the pool of zones used for the active measurements. The ISC data included PTR records that map IP address to DNS names, and we examined the names in the PTR data field. From these names, we stripped off the first element to obtain a *potential* zone name. We then used a set of relatively simple sanity checks to eliminate bogus entries that included non-existent top level domains. This left us with a pool of about 2 million potential zone names, from which we picked 3% of the names through a uniformly random selection. Finally, we queried each selected name to verify that it indeed belonged to an existing zone. This created our measurement set Sample 1, as shown in Table 2.

For zones belonging to Sample 1, we checked for configuration errors by sending queries to all servers associated with each zone. If our initial query failed to receive a reply, we retried twice before declaring the server *unreachable*. Thus our classification of unreachable servers in Sample 1 could be the result of either transient outages of the servers themselves, or of the network.

Sample 2 consists of all the Sample 1 zones that enable public zone transfers from their authoritative servers. We created Sample 2 for two reasons. First, the number of zones in Sample 1 was too big for periodic monitoring. Second, by allowing zone transfers, the zones in Sample 2 can provide us with additional information, such as the delegation records of the zone, the number of delegations, and changes in the delegation records. For zones in Sample 2 we continuously monitored their status once every week for the duration of one month and thus the results were less sensitive to transient failures.

Finally we created Sample 3 in order to measure the pervasiveness of configuration errors in "important" zones, that is zones that host at least one popular web-server. We collected the 500 most popular web-servers by combining the web-server rating information given by two independent sources [25, 6]. While Samples 1 and 2 gauge the extent of configuration errors in the current DNS infrastructure, we use Sample 3 to estimate how frequently a typical user may be affected by these errors.

Using active probing, we can identify the number of zones in our sample sets that appear to have a specific configuration error. From this number, we infer the percentage of zones in the DNS system that may experience a given type of configuration error. In addition, active measurements provide some insight to the questions of where and why these errors occur. For example we can identify whether there is a correlation between a specific error and the number of delegations a zone has, or whether a configuration problem is associated with newly created zones or with older ones.

Given that the active measurements are based on statistical sampling, the results are subject to statistical errors. Specifically, the confidence interval for each measurement depends on the total number of zones in each sample. Even though our sample size is large enough to incur very small statistical errors (with 95% confidence interval the errors are less than 0.3%), when we group the results by top level domains, statistical errors start to vary depending on the total number of zones in our sample that are under a given TLD. For this reason, we provide the error estimates for a confidence interval of 95% whenever we group the results by TLDs. We also note that the original pool of zones is not a complete collection of the DNS space; as a consequence, it may add a skew in our sample of zones.

[2] Labels are separated by ".". However, the presence of "." in a name does not necessarily signify a zone delegation. For example, "a.b.c.example" may belong to zone "b.c.example" or zone "c.example" if there was no delegation to "b.c.example". This ambiguity can occur at any level, but the presence of a "." near the last labels does tend to indicate a delegation and this allows us to reasonably infer the second level zone name from the query name.

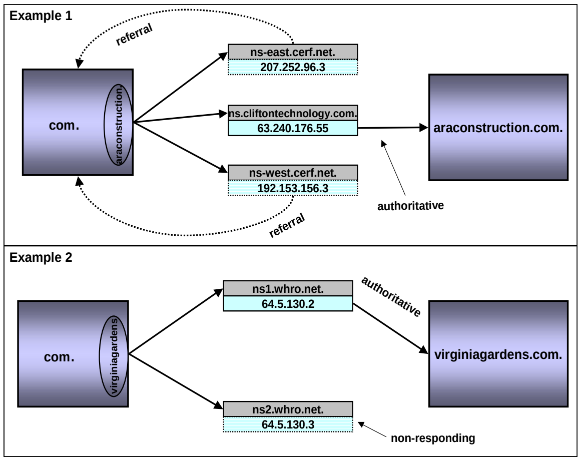


Figure 1: Lame Delegation Examples: A) "authoritative" servers *ns-east* and *ns-west* answer queries for zone *araconstruction.com.* with a referral to the *com.* servers; B) "authoritative" server *ns2.whro.net* for zone *virginiagardens.com.* zone does not respond to queries for that zone.

Example 1		(Date: 12/07/03)
$ORIGIN com.		
araconstruction.com	NS	ns.cliftontechnology.com
araconstruction.com	NS	ns-east.cerf.net
araconstruction.com	NS	ns-west.cerf.net
Example 2		(Date: 12/07/03)
$ORIGIN com		
virginiagardens.com	NS	ns1.whro.net
virginiagardens.com	NS	ns2.whro.net

Table 3: The configuration of: A) *araconstruction.com.* and B) *virginiagardens.com.* at the *com.* zone

4. MEASUREMENT RESULTS AND ANALYSIS

In this section, we describe three specific types of DNS configuration errors and provide measurement results for each of them. The first type of error is *lame delegation*. The second type is *diminished server redundancy*. While both types of errors are known problems in the DNS community [8, 13, 14], to the best of our knowledge there has been no quantitative study to gauge their impact on DNS performance and the extent of the problems in the Internet. The third type is *cyclic zone dependency*, a new type of DNS configuration error that has not been previously documented.

4.1 Lame Delegation

When a parent zone P delegates part of its namespace to a child zone C, P stores a list of NS resource records for the authoritative servers of zone C. This list of NS resource records are kept both at the parent zone P and the child zone C. Whenever the operator of zone C makes changes to one or more of C's authoritative servers, he must coordinate with the operator for zone P to update P accordingly. A lame delegation occurs when a DNS server that is listed as an authoritative server for a zone cannot provide authoritative answers for names in that zone. Such errors occur when changes at a child zone are not reflected to the NS RRs at the parent zone, resulting in some of the NS RRs at the parent zone pointing to either non-existing or non-authoritative servers. Lame delegation can also occur when the parent and child NS records are consistent but both point to non-existing or non-authoritative servers.

Table 3 shows the configuration of two DNS zones that suffered from lame delegation. As Figure 1 shows, the *com* zone had three NS records for the *araconstruction.com* zone, pointing to servers *ns.cliftontechnology.com*, *ns-east.cerf.net*, and *ns-west.cerf.net*, respectively. When a query for a name belonging to the *araconstruction.com* zone was sent to each of the three servers, only the first one replied with an authoritative answer; the other two servers replied with a referral to the servers for the *com* zone, indicating that they were not authoritative servers for the *araconstruction.com* zone. In the second example, *com* zone indicated that the zone *virginiagardens.com* was served by two name servers. When queried, however, only the first one provided authoritative answers; the second server, *ns2.whro.net*, did not respond at all.

The existence of lame delegations can affect DNS performance in at least two ways. First, it decreases the zone's availability: in the previous examples, out of a seemingly redundant set of servers for both zones, only a *single* server served each zone. Second, it increases the query response time. Queries sent to lame servers either do not elicit any answer, in which case a resolver waits until the query timer expires (usually set to 3-4 seconds), or receive a useless referral. In either case the resolver has to contact the next authoritative server until it receives an authoritative answer for the zone, or give up after retrying all the known authoritative servers for the zone [22]. In the best case a lame server may reply with a non-authoritative answer to a query if it happens to have cached the name in question.

A number of RFCs and operational directives have been written –the first one published in 1996 [8]– to address the lame delegation problem. Attempts to solve the problem so far have focused on informing the operators of the involved administrative domains to fix them [13]. However our measurements show that the problem is still widespread, indicating that the approach of manually detecting and fixing lame delegation errors has not been very effective.

4.1.1 Measurement Details

We used our custom-built DNS resolver to assess the pervasiveness of lame delegations. The resolver identifies cases of lame delegation in the following way: First, for each zone C the resolver iteratively queries the DNS system, starting at the root servers, until it reaches the parent zone P of C. At each step, the resolver queries for the SOA (Start of Authority) resource record of zone C; it makes use of previously cached entries whenever they are available to avoid unnecessary queries. The iterations stop when the resolver queries a server of the parent zone P[3], which replies with a referral to the servers of zone C. Up to this point, our resolver behaves as a standard DNS resolver. Next, the resolver tests whether all the servers, returned by the referral from the parent zone P, are indeed authoritative servers for the child zone C. An authoritative server should reply with the actual answer and the DNS header AA (authoritative answer) bit set. Note that if the zone C exists then the SOA resource record is always stored in C's zone file, and thus we can always expect a valid answer. On the other hand if we do not get an answer to the SOA query, the server is considered to be lame for the zone.

We sort lame delegations into the following three types based on what happens during the querying process:

- *Type I: non-responding server*. The server does not respond

[3] In a small number of cases, one server may be authoritative for both the child and its grandparent zones and thus our resolver never encounters the parent servers. We account for this case by using the child NS RR set to test for lame delegations in such rare cases.

TLD	Sample 1	Sample 2
com	17.80±0.5	14.66±0.80
net	18.21±1.19	15.26±1.95
org	17.49±1.45	18.08±2.49
edu	16.07±2.83	15.53±4.95

Table 4: Percentage of zones that are lame delegated: Sample 1 and Sample 2 results

to DNS queries. This could be due to multiple reasons, for example no machine is assigned to that IP address, a machine does exist but no DNS server listens on port 53, or even a (misconfigured) firewall is blocking DNS queries.

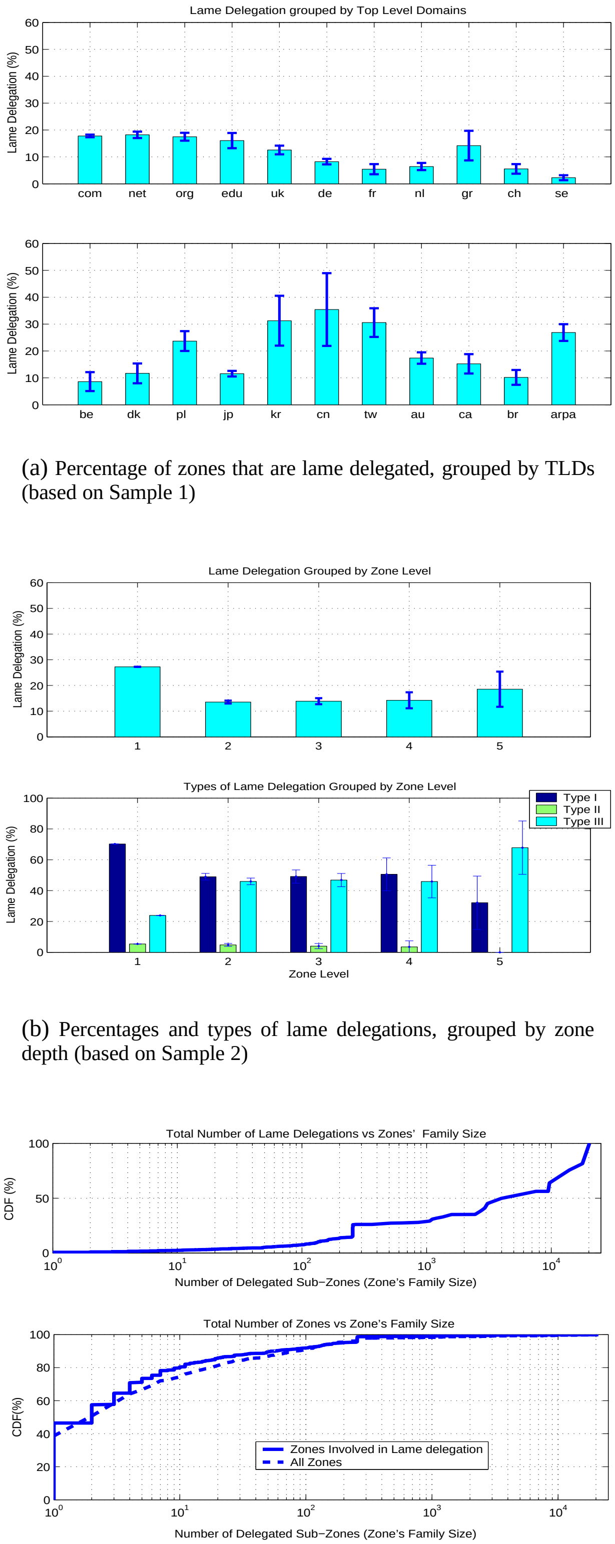
- *Type II: DNS error indication*. The server replies with an error indication (ServFail or Refused error code). These errors indicate that the server is not properly configured. This can possibly happen in cases of wrongly configured access lists and/or incorrectly defined views[4] [3].
- *Type III: non-authoritative answer*. The server does not return authoritative answers (the AA bit is not set). The server either replies with a referral to another zone, likely higher up in the DNS tree and usually the root zone, or it replies with the actual answer, if the requested name happens to be locally cached.

4.1.2 Measurement Results

Our main goal is to obtain a quantitative estimate of the pervasiveness of lame delegation among DNS servers. Along the way we also tried to identify whether there is any relation between the occurrence of lame delegations and the zones' geographic locations, the depth of the zone in the DNS hierarchy, and the number of delegations associated with the zone. We define the number of delegations associated with a zone as the zone's *family size*.

Figure 2(a) shows the percentage of zones that are lame delegated, grouped by top level domains. These results are based on measurements done with Sample 1 and the selected TLDs are a representative sample of gTLDs and ccTLDs. By representative we mean two things: i) there is a large number of zones in our samples that belong to the selected TLD; ii) for the case of ccTLDs, the selected TLDs cover different continents. The figure shows that there is a relation between the pervasiveness of lame delegation and geographical distribution: most of the zones that belong to the RIPE[5] region have a low percentage of lame delegations, lower than 10% in most cases, whereas many zones in the APNIC[6] region have a lame delegation percentage higher than 20%. Zones belonging to gTLDs lie somewhere between, with the exception of the zones under $arpa$, which have a considerably higher percentage of lame delegations.

Table 4 compares the frequency of lame delegations for four gTLDs, as measured by using zones from Sample 1 and Sample 2. There is a perception that zones allowing zone transfer (those in Sample 2) are managed by "sloppier" administrators, thus one might expect these zones to have higher percentage of lame delegations. Our results show that this is not the case. The four gTLDs are chosen because our Sample 2 contains a large number of zones in each of the four gTLDs and thus the results have small statistical



(a) Percentage of zones that are lame delegated, grouped by TLDs (based on Sample 1)

(b) Percentages and types of lame delegations, grouped by zone depth (based on Sample 2)

(c) Relation between lame delegation and number of delegations (based on Sample 2)

Figure 2: Results from active measurement on lame delegation

[4] A DNS server can be configured with access lists and views which can control the content of the reply, based on the client's IP address. Thus, a server can reply differently to hosts "behind" and "outside" a firewall.

[5] The uk, de, fr, nl, gr, ch, se, be, dk and pl zones.

[6] The jp, kr, cn, tw, au zones, for example.

gTLD	Type I	Type II	Type III
com	47.51±2.75	4.11±1.90	48.22±2.75
net	52.61±6.45	3.48±2.37	43.41±6.41
org	42.78±6.96	3.61±2.62	53.61±7.02
edu	45.83±14.10	6.25±5.85	47.92±14.13

Table 5: Types of lame delegation and their appearance frequency

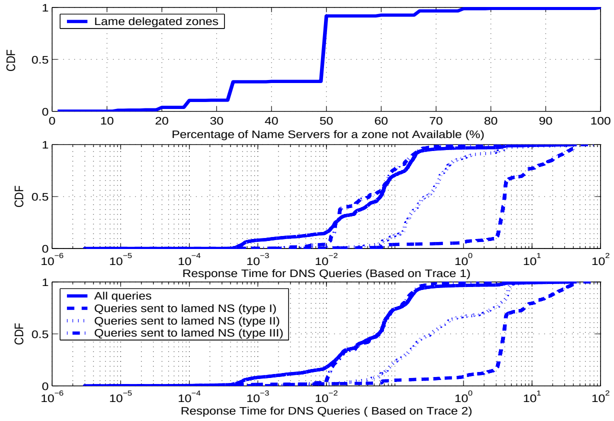


Figure 3: Impact of lame delegation on DNS availability and performance

errors with a confidence interval of 95%. Figure 2(a) uses Sample 1 because its much larger size provides a better representative coverage of all the zones; the remainder of the graphs uses the smaller Sample 2 data. Zones belonging to Sample 2 were periodically monitored over a month, thus the results are much less likely to be affected by transient network failures.

Table 5 shows the frequency for each of the three different types of lame delegations, grouped by the same four gTLDs. We observe that the first and third types of lame delegations are responsible for the majority of the cases. The first type, non-responding servers, accounted for nearly 50% of all the lame delegation cases and, as we will show later, this type of lame delegation can increase query response time by one order of magnitude.

Figure 2(b) shows the percentage of lame delegation at different levels of the DNS hierarchy. More specifically the upper graph in Figure 2(b) gives the lame delegation occurrence frequency, whereas the lower graph shows the types of lame delegation at different levels; level 1 zones are TLDs. With the exception of level 1 zones, both graphs show that the percentage of lame delegation errors is independent from the zone's location in the DNS hierarchy, suggesting that the lack of coordination between parent and child zone administrators is likely to be the main reason behind the delegation errors.

We repeated the same measurements using Sample 3 in order to gauge the pervasiveness of these errors on the "most important" zones. The results show that 7.6% of the popular zones are lame delegated, and none of the top 100 zones has any lame delegation errors. These results indicate that, as one might expect, popular zones are much better administered compared to a randomly chosen zone. On the other hand, the results on the different types of lame delegation for the popular zones are consistent with the ones we obtain from Samples 1 and 2, with type I appearing in nearly 50% of the cases and type III in the other 50%.

Figure 2(c) shows the relation between the number of delegated zones, i.e. a zone's *family size*, and the percentage of zones with that family size that have delegation errors. The upper graph gives the cumulative distribution function (CDF) of the absolute number of lame delegated zones across the number of delegated zones. The graph shows that zones with large family sizes contribute the largest percentage of lame delegations, even though the number of "small" zones is considerably larger than the number of "large" zones.

Similarly the lower graph of Figure 2(c) shows the relations between the percentage of lame delegated zones across the number of delegated zones. The solid line gives the CDF of the percentage of zones that are lame delegated, for a given number of delegations. The dashed line on the same graph shows the CDF of the number of zones with a given number of delegations. Since the two distributions, the percentage of lame delegations and the number of delegations, are very close, the probability of a zone being lame is the same for large and small family sizes. If one expects that larger zones are managed by administrators who are more likely to be aware of the lame delegation problems, then this result further suggests that the lack of coordination between the parent and child zones is the main cause of lame delegations.

4.1.3 *Impact of Lame Delegation*

The upper graph in Figure 3 shows the impact of lame delegation on zone availability. We plot the CDF for the percentage of unavailable servers in a lame delegated zone. We note that for about 70% of the zones which suffer from lame delegations, the number of available servers is reduced by about 50% or more, that is, those zones are served by half or less of their servers.

The lower two graphs in Figure 3 show the impact of lame delegations on query response time by using results from our passive traces. They plot the CDF of the total response time for queries that encountered at least one lame server of type I, II or III, and the CDF of total response times for all the other queries. The results show that lame delegations increase the DNS response time considerably. For normal queries (those that do not encounter lame servers), the mean response time is about 60 msec; for queries that encountered at least one lame server of type I, the response time is longer than 3 seconds in most cases, and can even exceed 30 seconds in rare cases. Moreover, the response time for queries that encountered at least one lame server of type II is increased by several hundreds of milliseconds, compared to the response time for normal queries. Finally, queries sent to type III lame servers experienced response times similar to normal queries. A possible explanation is that the non-authoritative servers replied with the correct answer which had been locally cached.

Finally, Table 6 gives the number of queries sent to lame servers. It shows that lame delegation related queries contributed around 8% of the total number of queries in Trace 1, and 13% of the queries in Trace 2. The total number of queries sent to lame servers depends highly on the users' profiles, thus we cannot conclude that these numbers represent typical cases for sites other than the ones we observed. Note that the percentage of queries that went to non-responding servers is much larger than other types of lame related queries; this is because the resolvers in our traces repeatedly sent queries to non-responding servers. One may also note that the number of type II queries is much higher than type III, while the number of zones containing a type II lame delegation is relatively small (see Table 5). Further examination shows that 92.6% of type II queries went to the *arpa* domain for Trace 1 and 88.4% for Trace 2, and queries that went to the *arpa* domain are about 20-30% of the total queries. Overall, these numbers show that traffic due to lame delegations can make a considerable portion of the total DNS traffic.

4.2 Diminished Server Redundancy

DNS uses redundancy as one of the two mechanisms for high

Type of lame delegation	Number of queries sent to lame servers	
	Trace 1	Trace 2
Non-responding NS (Type I)	117,753	310,975
NS replying with ServFail (Type II)	58,278	65,141
NS replying with Refused (Type II)	1,162	1,740
NS replying with non-authoritative answers (Type III)	25,180	24,904

Table 6: Number of queries sent to lame servers: Trace 1 contained 2,470,370 queries in total; Trace 2 contained 3,097,028 queries in total

Example 1		(Date: 12/07/03)
$ORIGIN pik-net.pl		
bieszczady.pik-net.pl	NS	ns3.pik-net.pl
bieszczady.pik-net.pl	NS	ns1.pik-net.pl
bieszczady.pik-net.pl	NS	ns2.pik-net.pl
ns3.pik-net.pl	A	213.241.68.129
ns1.pik-net.pl	A	213.241.68.198
ns2.pik-net.pl	A	213.241.68.146

Example 2		(Date: 12/07/03)
$ORIGIN nl		
saxcompany.nl	NS	ns.vuurwerk.nl
saxcompany.nl	NS	ns2.vuurwerk.net.
saxcompany.nl	NS	ns3.vuurwerk.net.
ns.vuurwerk.nl	A	62.250.2.2
ns2.vuurwerk.net.	A	212.204.221.71
ns3.vuurwerk.net.	A	213.136.0.173

Table 7: Diminished Server Redundancy Examples: A) all of the *bieszczady.pik-net.pl* authoritative servers are under the same /24 prefix, advertised by the same AS and located in the same city. B) all three *saxcompany.nl* servers are under different /24 prefixes, advertised by three different ASs, and located in three different cities.

availability - the other one is caching. The level of availability provided by redundant servers is a function not only of their number but also of their location. An operational server may not be able to answer DNS requests if the network path between the server and the clients is unavailable due to physical failures or routing problems. If all the replicated servers are connected to the same local network, then the redundancy is lost when that network fails. If all the servers are assigned addresses from the same address prefix, they will all be unavailable when that prefix becomes unreachable due to routing problems. If all the servers are in the same geographic location, natural disasters (e.g. earthquakes or floods) or large scale power failures may again cause all the replicated servers to fail. Therefore, to build redundancy against *unexpected* failures, the replicated servers must be placed in diverse locations that are unlikely to fail at the same time. Diverse server location not only increases service reliability but also reduces query response time since diversely placed servers can better cover widely distributed client populations.

Table 7 shows different configurations of redundant servers for two sample zones. In the first case all the authoritative servers for the *bieszczady.pik-net.pl* zone were assigned addresses from the same /24 prefix, which was advertised by a single autonomous system. Furthermore, all of the servers were located behind the same last hop router [7], and they were also placed in the same geographic location. In contrast, the three authoritative servers for the *saxcompany.nl* zone were assigned addresses from different address prefixes, advertised from three different autonomous systems, and were located in three different cities (based on the NetGeo database [4]).

The need for diverse placement of redundant servers is a well known requirement in the DNS community. Several RFCs [14, 8] state the requirement for geographic distribution of the authoritative servers. However despite these RFC guidelines, operators often decide where to place the zone's DNS servers based on their *perception* of expected failures. In the case of Microsoft DNS incident in January 2001, operators (correctly) believed that redundant servers would have protected the domain against individual *server* failures but overlooked the possibility of *network* failures. The result was that the entire *microsoft.com* zone became unreachable when an unexpected router failure occurred. Unfortunately, in the absence of any systematic mechanisms to enforce adequate diversity in server placement, the actual decision is often made under the constraints of cost and management by operators that may have partial understanding of the consequence.

We believe that diverse placement of redundant servers should be a requirement for both large and small organizations. One could imagine a scenario where all the hosts of a small organization depend on a single upstream link to the Internet. If this upstream link fails, none of the hosts in the organization will be reachable. Even in this case, we argue that diverse DNS server placement has clear advantages. If all the organization's DNS servers are placed behind the failed upstream link, resolvers will slowly try each server before finally abandoning the query. This potentially adds a long delay before the application learns the host is unresolvable. Furthermore, applications (and users) may respond differently based on a “hostname unresolvable” or “host unreachable”. One should not assume these two errors are equivalent for all applications. By keeping DNS service available through diversely placed servers, applications (and users) learn the desired host IP address exists. Then they can determine whether the host is reachable and respond appropriately.

The need for diverse placement of redundant servers goes even beyond the availability issue of affected zones. According to DNS specification ([21], Section 5), a resolver may go back to query the root servers if it fails to receive a response from any of the servers for a requested zone. There is at least one popular DNS resolver implementation which, after failing to receive response from a requested zone, will continuously query higher level DNS servers [16]. Such DNS implementations introduce a coupling between the availability of any specific zone's DNS servers and the load put on the top level DNS servers, resulting in undesired global impact due to local failures.

Unlike the lame delegation problem, the lack of diversity in server placements is not easily observed, making the problem more difficult to detect. Because failures are rare events, and large scale failures are more so, even zones with vulnerable server placements appear to work correctly under normal conditions. As a result, administrators often discover the underlying vulnerability only *after* a failure disables the service.

4.2.1 Measurement Details

Our main goal is to estimate the number of authoritative servers that have independent failure modes. In order to measure the server redundancy we need to estimate whether two servers share a common point of failure. Specifically, we try to identify if two servers share the same network subnet, which in most cases implies that they are behind the same router, if they are served by the same autonomous system, and if they are placed in the same geographic location.

[7] We discovered this fact by running `traceroute` to each of the three listed addresses.

Degree of redundancy	Geographic Level	AS Level	Prefix (/24)	Host
1	82.30	77.19	45.80	14.69
2	16.51	19.56	42.73	65.20
3	0.78	2.26	8.94	14.45
4	0.36	0.56	1.81	4.50
5	0.03	0.36	0.57	0.86
6	0.02	0.04	0.10	0.23

Table 8: Name server redundancy for different redundancy definitions.

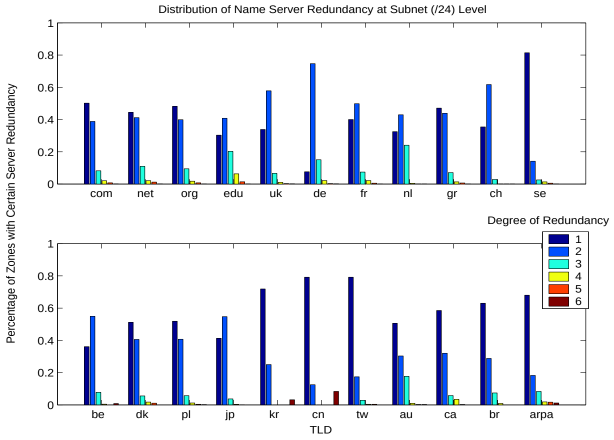


Figure 4: Distribution of redundancy at /24 prefix level

Since we do not know how remote networks allocate addresses, the results presented in this section assumes a subnet prefix length of /24 since it is the most commonly used. We also tried a variable size prefix length, ranging from /24 to /30, to decide whether two servers are in the same subnet, and did not observe any substantial differences compared to using /24 prefixes. We use the BGP routing tables provided by the RouteViews project [5] to locate the AS that each server is located in. We perform a longest prefix match in the BGP table for the server's IP address; the server is assumed to reside in the last AS on the AS Path associated with that prefix.

Finally, we estimate the geographic location of different servers using the NetGeo database [4], which provides a mapping between AS numbers and geographic locations. NetGeo provides location information in three levels: city level, region (state) level, and country level. We use the city level to estimate the servers' geographic location, since servers in the same city are likely to be affected by the same external failures. In cases where we cannot extract city level information, we use the most specific common level to compare the two locations. We note that geographic mapping techniques may not be accurate [26], nevertheless we believe they provide adequate results for our aggregate measurements.

4.2.2 Measurement Results

Table 8 shows the degree of redundancy for different definitions of redundancy. At the host level we can see that most zones (65%) have two authoritative servers, and a small percentage (20%) have three or more servers. At the prefix level, 55% of the zones have two or more authoritative servers located at different /24 prefixes. An even smaller percentage of zones, i.e. less than 25%, have two or more servers located at different ASs or different geographic locations.

We also computed the server redundancy at the /24 prefix level for zones grouped by their TLD. From Figure 4, we can observe that there is a relation between server redundancy at the prefix level and the TLD where the zone belongs to. We did not observe any substantial difference among the TLDs for the other levels of redundancy. The main reason is that in most cases all servers for the same zone are placed in the same location or advertised by the same AS.

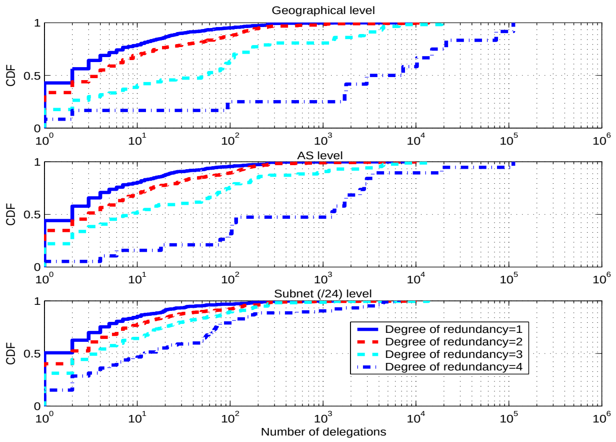


Figure 5: The relationship between percentage of zones with a specific degree of redundancy and zone family size.

Degree of Redundancy		Prefix Level	AS Level	Geographic Level
1	zones unreachable (%)	15.90%	14.73%	13.32%
	zone availability (%)	99.89%	99.92%	99.97%
2	zones unreachable (%)	11.39%	4.23%	3.95%
	zone availability (%)	99.94%	99.98%	99.99%
3	zones unreachable (%)	6.11%	3.84%	4.49%
	zone availability (%)	99.98%	99.99%	99.99%

Table 9: Impact of diminished server redundancy on availability

We repeated the same measurements for the popular zones of Sample 3. The results show again that popular zones are better administered compared to a randomly chosen zone. For example only 1% of the zones have one DNS server and 24.4% of the zones have all their authoritative servers under the same prefix. 42% of the zones have at least two of their servers placed at different ASs, and 32% of the zones have servers located in multiple cities.

Finally, we computed the percentage of zones with a specific family size that have redundancy degree from 1 to 4 for our three measures of redundancy (prefix, AS and geographic location), where redundancy degree is measured by the diversity in prefixes, ASes, and locations of redundant DNS servers. Figure 5 shows one graph for each definition of redundancy. Each line in the graph represents a specific degree of redundancy. From these figures we can observe that zones with larger family sizes tend to have a higher degree of redundancy. For example, at the geographic level, 50% of zones with family size 10,000 have degree of redundancy 4 compared to only 10% of zones with family size 10. This effect is more prominent for geographic and AS level redundancy. Our observations seem to indicate that larger zones have a higher degree of redundancy compared to smaller zones.

4.2.3 Impact of Diminished Server Redundancy

While it is intuitive that a smaller degree of redundancy translates to a reduced level of availability, we conducted quantitative measurement of server placement's impact on availability. To assess the availability of a zone, we sent probes every 30 minutes

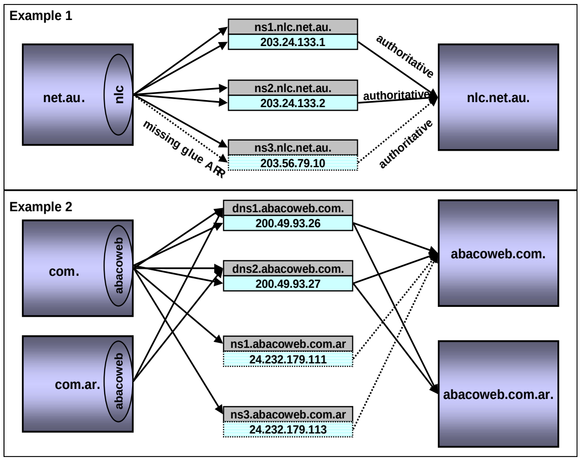


Figure 6: Cyclic Zone Dependency Examples: A) a cyclic zone dependency at the *nlc.net.au.* zone; if both *ns*{*1,2*}*.nlc.net.au.* are unavailable, then the *nlc.net.au.* cannot be resolved. B) a cyclic zone dependency between two zones, the *abacoweb.com.* and the *abacoweb.com.ar.*; if both *dns*{*1,2*}*.abacoweb.com.* are unavailable, then the *abacoweb.com.* cannot be resolved.

for a duration of two weeks to all the authoritative servers of every zone of Sample 1 that has three servers[8]. We define each 30 minute probing as a round, and a zone as *unreachable* in one round if none of the authoritative servers replied to the probing. The *availability* of the zone is defined as the ratio of rounds that received at least one answer, coming from an authoritative server, to the total number of rounds.

Table 9 shows the results of our measurements. We group zones based on their redundancy (1, 2 or 3) at the prefix (/24), AS, and geographic level. For each of these groups we calculate the percentage of the zones that are unavailable for at least one round. We also calculate the average zone availability for each of the three groups. This table shows clearly that zones with all their servers under the same prefix tend to have the worst availability: around 16% of such zones are unavailable and their availability on the average is under 99.9%[9]. In contrast, zones whose servers are placed at three different geographic locations, or in three different ASs, have notably higher availability (99.99%). Moreover, the number of zones that are unavailable at least one time is considerably lower when servers are correctly placed (4.49% and 3.84% respectively).

4.3 Cyclic Zone Dependency

To achieve the desired geographic and network diversity for a zone's authoritative servers, operators often establish mutual agreement to host each other's DNS services. For example, *ns.bar.com* may serve as a secondary authoritative server for zone *foo.com*. Authoritative servers located in other zones are normally identified by their names instead of IP addresses. As a result, to resolve a DNS name in zone *foo.com* requires one to first resolve the IP address of *ns.bar.com*. A cyclic zone dependency happens when two or more zones' DNS services depend on each other in a circular way: to resolve a name in zone $Z1$, one needs to first resolve a name in zone $Z2$, which in turn requires some information from zone $Z1$. This type of inter-dependency creates a “chicken and egg” problem; one cannot resolve a name in zone $Z1$ without first resolving a name in $Z2$ and vice versa. This scenario can occur due to configuration errors in either or both of the zones, however it is more often the case where none of the involved zones has any noticeable configuration error, yet the combination of two or more correctly configured zones results in cyclic zone dependency.

Example 1		(Date: 12/07/03)
$ORIGIN .net.au.		
nlc.net.au.	NS	ns1.nlc.net.au.
nlc.net.au.	NS	ns2.nlc.net.au.
nlc.net.au.	NS	ns3.nlc.net.au.
ns1.nlc.net.au.	A	203.24.133.1
ns2.nlc.net.au.	A	203.24.133.2
Example 2		(Date: 12/07/03)
$ORIGIN com.		
abacoweb.com.	NS	ns1.abacoweb.com.ar.
abacoweb.com.	NS	ns3.abacoweb.com.ar.
abacoweb.com.	NS	dns1.abacoweb.com.
abacoweb.com.	NS	dns2.abacoweb.com.
dns1.abacoweb.com.	A	200.49.93.26
dns2.abacoweb.com.	A	200.49.93.27
$ORIGIN com.ar.		
abacoweb.com.ar.	NS	dns1.abacoweb.com.
abacoweb.com.ar.	NS	dns2.abacoweb.com.

Table 10: The configuration of: A) *nlc.net.au.* at the *net.au.* zone; B) *abacoweb.com.* at the *com.* zone and *abacoweb.com.ar.* at the *com.ar.* zone

Figure 6 shows two real examples of cyclic zone dependencies we captured during our measurement and Table 10 lists the configuration details of the involved zones. The first example involves a single zone only and the problem was due to a configuration error in the parent zone, which should have included the glue A record for the *ns3.nlc.net.au.* Without that glue record, the third server was reachable only after one was able to resolve its IP address by querying one of the other two servers. In case these two servers became unavailable, it was impossible to obtain the IP address of *ns3.nlc.net.au.* The second example is slightly more complicated and involves two correctly configured zones. The parent zone, *com*, listed four servers for the *abacoweb.com* zone. However, in order to reach both *ns*{*1,3*}*.abacoweb.com.ar* servers, one had to first contact the *abacoweb.com.ar* zone. The *abacoweb.com.ar* zone was served by two servers, *dns*{*1,2*}*.abacoweb.com.* In case both of them became unavailable, the zone *abacoweb.com* was unavailable. Even though *ns1.abacoweb.com.ar* and *ns3.abacoweb.com.ar* might have functioned correctly, they were not reachable because their IP addresses couldn't be resolved.

The above examples illustrate the failure dependency between zones, the failure of some servers in one zone leads to unreachability of all authoritative servers in another zone. We have found cases where, due to cyclic zone dependency, a zone that appears to have several redundant servers actually relies solely on the availability of one server, which effectively becomes a single point of failure. Similar to the diminished server redundancy problem, these errors can significantly reduce the system's redundancy and they are not immediately visible to the operators. Moreover, it is possible that some incorrectly implemented DNS resolvers may be trapped in a query loop when they encounter a cyclic dependency case and certain servers are unavailable.

Among the three configuration problems discussed in this paper, cyclic zone dependency is the most difficult to detect. The system operates correctly in the absence of server failures and, when inspected individually, each of the zones involved in a cyclic zone de-

[8] We chose zones with three servers because they qualify as 'properly' managed zones from a redundancy perspective.

[9] 99.9% availability is equal to 8.76 hours of downtime per year, compared to 52.55 minutes of downtime for 99.99% availability.

	Percentage of Zones Involved	Percentage of Zones Affected
All glue A records	2.39%	5.95%
Necessary glue A records only	2.99%	33.34%

Table 11: Cyclic zone dependency appearance frequency

pendency appears to be configured correctly. The inter-dependency loop can be discovered only when one brings together the combined configuration of all the involved zones. Although our measurements show that this configuration error is not as widespread as the previous two types of errors, most of the existing cases reduce system redundancy substantially. There is also a concern that, without a systematic mechanism to detect this problem, it may spread more widely in the future as the DNS system continues to grow and the operators responsible for configuring new DNS servers may not be aware of this subtle problem.

4.3.1 *Measurement Details*

In order to detect cyclic zone dependency errors, our DNS resolver follows all the possible paths in resolving a query. When the resolver gets a referral answer, it tries to resolve the IP address for all the NS records that do not have a glue A record in the additional section of the answer. This starts a new sequence of iterative queries , and puts the previous question in a pending state. If during that process it happens to ask information from a zone that is already in pending state, then a cyclic zone dependency is detected. With a careful examination of the servers that are part of the cycle, we can identify the servers that depend on the availability of other servers in the loop.

Cyclic zone dependencies could be eliminated by the inclusion of specific glue A resource records. For example in the second example shown in Table 10, had the *com* zone server included glue A RRs for the two servers *ns*{*1,3*}*.abacoweb.com.ar*, one would have been able to resolve names in *abacoweb.com* zone even when the two *dns*{*1,2*}*.abacoweb.com* servers were unavailable. However the current DNS implementations discourage the loading of *unnecessary* glue A RRs. A glue A RR is considered "unnecessary" if it points to servers that belong to a domain different from the delegated domain. In the second example in Table 10, all the glue A RRs defined at the *com* zone are necessary; glue A RRs for *ns*{*1,3*}*.abacoweb.com.ar* are unnecessary and thus are not included in the *com* zone. This constraint is introduced to reduce the chance of lame delegation. When a zone includes the glue A RR that belongs to another zone, it must updates the A RR when any change to the A RR is made at the authoritative zone. Unfortunately, such updates are done manually and errors or delays in the manual operation lead to lame delegations. The inclusion of unnecessary glue A RRs has reportedly led to lame delegations in the past. In one anecdotal example, a particular server had differing glue A RRs (IP addresses) listed in thirteen different zones and only one of the IP addresses was correct [1].

4.3.2 *Measurement Results*

Since operational practice may differ regarding the inclusion of unnecessary glue A RRs, we repeated each type of measurement with and without accepting unnecessary glue A records. Table 11 shows the appearance frequency of the cyclic zone dependency problem based on the measurements of zones from Sample 1. About 2% of the zones were involved in a cyclic dependency error, however the number of zones that were affected by these problems is around 5%. A zone is affected if any of following three conditions hold: *i)* the zone is directly involved in a cyclic dependency, *ii)* one or more zone servers lie in a zone with a cyclic dependency, or *iii)* if any ancestor zone is involved in a cyclic zone dependency. Note that measurements based on Sample 3 show that none of the popular zones is involved in cyclic dependencies. Table 11 also shows the percentage of zones that are involved and the percentage of zones that are affected by the cyclic zones dependency errors, when only necessary glue A RRs are accepted. While the percentage of involved zones is not increased much by the elimination of unnecessary glue A RRs, the number of affected zones is increased to 33%, primarily because some top level domains are involved in cyclic zone dependency which are currently concealed with the use of unnecessary glue A records.

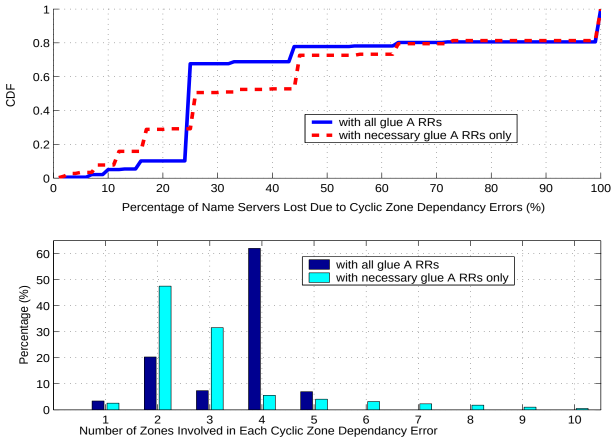


Figure 7: A) Servers lost due to cyclic zone dependency errors; B) Number of zones involved in cyclic dependency errors.

As the number of zones involved in a specific cyclic zone dependency increases, the manual detection of the problem becomes harder because one must bring together the configuration data from each of the zones. Fixing the problem becomes even harder because it requires coordinated changes in multiple zones. The lower graph of Figure 7 gives the number of zones that are involved in each cyclic zone dependency, with and without considering unnecessary glue A RRs. If all glue A RRs are accepted, cyclic dependency errors involve four zones in most cases. If only necessary glue A records are accepted, the number of two-zones cyclic dependency errors increases greatly. This indicates that cyclic dependency errors exist in the current system, but are concealed due to the unnecessary glue A RRs. Including these unnecessary glue A RRs in a zone increases chances of lame delegation, yet not including these unnecessary RRs leads to increased cyclic zone dependency.

4.3.3 *Impact of Cyclic Zone Dependency*

Cyclic zone dependency errors can reduce the DNS service availability. If a zone is involved in a cyclic dependency, the failure of DNS servers in some other zones can affect its own DNS service availability. The upper graph of Figure 7 shows the CDF for the percentage of a zone's authoritative servers that are a part of a cyclic zone dependency problem and may become unavailable due to dependency on other servers. As a result of cyclic dependency, a zone loses more than 25% of its servers in most cases. The graph also shows the CDF when no unnecessary glue A records are accepted. In this case, the impact of the cyclic zone dependency errors on the server availability becomes even greater.

5. DISCUSSION

The DNS design is an example of great engineering. It tried to balance the essential functionality requirement – scalable name resolution – against the complexity of different design choices. The result is a disarmingly simple system which has been proven extremely successful in both achieving the original objectives and adapting to changes. However the original design focused mainly on robustness against physical failures and gave no consideration to operational errors such as misconfigurations. As our measurements show, left unchecked, configuration errors have effectively reduced the level of redundancy in the deployed system. Thus the actual degree of redundancy is likely to be lower than the number of servers suggests for a significant portion of DNS zones. Although the system seems to be operating satisfactorily today, unexpected partial failures may occur at any time which may easily disable DNS services to those zones having one or more types of errors lurking in their configurations.

5.1 Detecting Misconfigurations

One can develop a set of simple mechanisms to detect all of the lurking errors identified in this study. Lame delegation can be detected by a simple protocol between parent and child zones to periodically check the consistency of the NS records stored at each place. Cyclic zone dependency can be detected via automatic checking by trying to resolve a name through *each* of the authoritative servers in the zone. Although there may not be a single check to detect the diminished server redundancy problem, automatic periodic measurement between servers of the same zone on their IP address distance, hop count distance, and AS distance can effectively reflect the diversity degree in their placement. These simple checks are absent from the original DNS design, not because they are difficult to do but a lack of appreciation of the severity of human-introduced errors. As part of our ongoing effort we are developing a software [27] that can proactively detect DNS configuration errors.

5.2 Comparison with Other Efforts

The IETF community has initiated several efforts to address DNS resilience issues. One is deploying anycast routing for root servers. The anycast routing solution removes the limit on the number of replicated servers for a zone. Moreover, it also eliminates the need to change configurations when the number or locations of the servers change. While anycast is well suited to the root and certain heavily used top-level servers, it is less practical for lower-level zones due to the scalability concern in global routing. It is also important to note that, unless redundant servers are placed in diverse locations and the anycast address is announced from those locations, anycast itself does not automatically solve the diminished server problem. It simply shifts burden from DNS configurations to routing configurations. Furthermore, anycast raises its own issues in problem debugging. For example, if a lame server exists among a set of anycast enabled servers, it can be difficult to pin down which one is the culprit.

The DNS Security Extensions (DNSSEC) [7] are another effort to improve DNS resilience. The original DNS design provided no authentication and current DNS service is vulnerable to a wide range of attacks [9]. The DNS Security Extensions add cryptographic authentication into the system, allowing a resolver to authenticate that the data received in the response matches the data entered by the zone operator. However authentication does not address any of the lame delegation, cyclic zone dependency, or diminished server redundancy problems. In fact, DNSSEC deployment may exacerbate these problems as cryptographic misconfigurations may further reduce the availability of DNS services. Moreover, DNSSEC defines an authentication chain where a new DS RR at the parent zone identifies a public key (DNSKEY) at the child zone. Just as a traditional zone must ensure the parent and child's NS RR sets are consistent, a secure zone must also ensure the DS RR at the parent matches the DNSKEY RR at the child. The addition of DNSSEC will only increase the importance of consistency checks and correct configurations.

5.3 Human Errors in Other Internet Systems

It is also important to note that configuration errors are not limited to the DNS system alone, although its dependency on the vast scale distributed database management across multiple administrative domains may have left plenty room for human errors. There are many manually configured parts in the operations of the global Internet, such as BGP configurations and resulting problems found in [19]. One lesson that can be drawn from our and other studies is that future protocol designs should take into account the impact of human errors.

6. RELATED WORK

Danzig *et al.* [23] provided an extensive analysis of the DNS traffic observed on a root name server. They identified a number of DNS implementation bugs and found that such bugs incurred unnecessary wide-area DNS traffic by a factor of twenty. In a followup work, Brownlee *et al.* [10] measured the impact of some new implementation errors on the DNS traffic directed toward the F root server. In this work, our focus is to identify configuration errors in the DNS system and to measure their impact on the zone's availability and performance.

Jung *et al.* [15] measured the DNS performance in term of query response time perceived by DNS resolvers, and studied the effectiveness of caching. They observed that the query response time is highly related to the number of referrals, and that the majority of queries complete in less than 100 milliseconds. They further concluded that DNS caching works effectively even when the TTL value of host names is as low as a few hundred seconds, as long as the domain servers' A RRs are cached. Our interest is in identifying the performance degradation, in terms of query response time, due to the configuration errors.

Liston *et al.* [18] studied the diversity in DNS performance perceived by a number of geographically distributed clients. They showed that the mean response time for completed name lookups at different sites varies greatly and the performance of root servers and TLD servers have the least impact on lookups for non-cached entries. In this paper we examine the diversity in server placement and its impact on zones availability.

Lee *et al.* [17] studied the optimal placement of root servers and showed that geographical distribution of root servers can help improve overall performance perceived by clients. In this paper, we further show that true server redundancy for a zone requires more than just a number of physically replicated servers. The geographical location diversity, AS diversity, and subnet diversity of the servers have a clear impact on the service availability as well as the performance.

Mahajan *et al.* [19] measured the appearance frequency of BGP misconfigurations and the impact on Internet connectivity. They showed that BGP misconfigurations happen often but have a small impact on the global connectivity. Although BGP is a large system, it is still possible to capture all the visible configuration errors on a global scale. In contrast, in this work we can infer the pervasiveness of DNS configuration errors based only through statistical sampling. On the other hand, it is more difficult to verify BGP misconfigurations than DNS misconfigurations because of hidden routing policies.

Finally there are a number of companies and individuals that look into the problem of lame delegation. Men & Mice [20] periodically measures the lame delegation as it appears under the *com* domain; Team Cymru [13] collects the BIND log files from a number of DNS servers and extracts from them a list of lame servers; and Credentia [12] provides lame delegation statistics on the TLD zones.

7. CONCLUSION

DNS is one of the best-known Internet systems providing indispensable name resolution services for end users and applications. Its design relies on redundant servers to achieve reliability. Adverse events, such as DDoS attacks against the DNS root servers, illustrate the critical dependence on distributed replicated servers. However, our measurements show that lame delegations and cyclic zone dependencies reduce the number of reachable servers and thus the actual system redundancy can be much lower than expected. Our results also show that a large portion of the zones have all their DNS servers placed either behind the same routing prefix or in the same geographical location, thus a physical disaster or network disruption can simultaneously incapacitate all of these servers, invalidating the assumption that server redundancy should provide resilience in the face of failures.

Distributed management is crucial in achieving DNS system's scalability, however our measurements show that it also leads to inconsistencies due to mistakes in coordinating zone configurations and changes. While human induced configuration errors are a well-known fact, DNS delegation configurations require consistency *across* administrative boundaries, a condition that is even more prone to errors. Unfortunately the current system does not provide any automated means to communicate for coordination. Today configurations are communicated manually, and as we have seen this process is highly subject to errors.

We draw two broad conclusions from our measurement study on the DNS configuration errors. First, large-scale, distributed systems should *expect* human errors and therefore should *proactively* develop systematic checking mechanisms against such errors. Second, in distributed systems such as DNS, acceptable system performance at the user level is not a reliable indicator that the system is error free. Minor errors may be lurking under the surface, and when a number of these errors cascade or get triggered simultaneously, the system can fail in unexpected and catastrophic ways. Other fields have provided numerous examples on how complex systems may fail through a number of cascaded small failures (e.g., failures of airplanes, nuclear plants and the power grid); the same lessons apply to the Internet as well. We are developing a set of active mechanisms to detect and eliminate neglected configuration errors in today's DNS system. Through this process we hope to gain further understanding on how to design such proactive checking into distributed systems in general.

8. ACKNOWLEDGMENTS

We would like to thank Jennifer Rexford, our shepherd, and the anonymous SIGCOMM reviewers for their valuable comments. We would also like to thank Patrik Faltstrom, Duane Wessels, Rob Austein, and Matt Larson for helping us clarify some implementation details of the DNS software and deployment. Finally we thank the UCLA IRL and WING groups for providing comments on earlier versions of this paper.

9. REFERENCES

[1] Private communication with Mark Andrews. Nov. 2003.

[2] Internet Domain Survey. http://www.isc.org/ops/ds/, 2004.

[3] ISC BIND. http://www.isc.org/sw/bind/, 2004.

[4] NetGeo Database. http://netgeo.caida.org/aslatlong.txt, 2004.

[5] Route Views Project. http://www.routeviews.org/, 2004.

[6] Alexa.com. http://www.alexa.com/, 2004.

[7] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS Security Introduction and Requirements. Work in progress: draft-ietf-dnsext-dnssec-intro-08, December 2003.

[8] D. Barr. Common DNS Operational and Configuration Errors. RFC 1912, 1996.

[9] S. M. Bellovin. Using the Domain Name System for System Break-Ins. In *Proceedings of the Fifth Usenix UNIX Security Symposium*, pages 199–208, 1995.

[10] N. Brownlee, k claffy, and E. Nemeth. DNS Measurements at a Root Server. In *Proceedings of the IEEE Globecom' 01*, pages 1672–1676, 2001.

[11] CAIDA. Nameserver DoS Attack October 2002. http://www.caida.org/projects/dns-analysis/, 2004.

[12] Credentia. http://www.credentia.cc/research/cctlds/, 2004.

[13] T. Cymru. http://www.cymru.com/DNS/lame.html, 2004.

[14] R. Elz, R. Bush, S. Bradner, and M. Patton. Selection and Operation of Secondary DNS Servers. RFC 2182, 1997.

[15] J. Jung, E. Sit, H. Balakrishnan, and R. Morris. DNS Performance and the Effectiveness of Caching. In *Proceedings of the First ACM SIGCOMM IMW*, pages 153–167. ACM Press, 2001.

[16] M. Larson and P. Barber. Observed dns resolution misbehavior. Work in progress: draft-ietf-dnsop-bad-dns-res-02, February 2004.

[17] T. Lee, B. Huffaker, M. Fomenkov, and kc claffy. On the Problem of Optimization of DNS Root Servers' Placement. In *Passive Measurement and Analysis Workshop '03*, 2003.

[18] R. Liston, S. Srinivasan, and E. Zegura. Diversity in DNS Performance Measures. In *Proceedings of the Second ACM SIGCOMM IMW*, pages 19–31. ACM Press, 2002.

[19] R. Mahajan, D. Wetherall, and T. Anderson. Understanding BGP Misconfiguration. In *Proceedings of the ACM SIGCOMM'02*, pages 3–16. ACM Press, 2002.

[20] Men & Mice. http://www.menandmice.com/, 2004.

[21] P. Mockapetris. Domain Names–Concepts and Facilities. RFC 1034, 1987.

[22] P. Mockapetris. Domain Names–Implementation and Specification. RFC 1035, 1987.

[23] P. B. Danzig and K. Obraczka and A. Kumar. An Analysis of Wide-Area Name Server Traffic. In *Proceedings of the ACM SIGCOMM'92*, pages 281–292. ACM Press, 1992.

[24] P. Mockapetris and K. J. Dunlap. Development of the Domain Name System. *SIGCOMM Comput. Commun. Rev.*, 18(4):123–133, 1988.

[25] Ranking.com. http://www.ranking.com/, 2004.

[26] V. N. Padmanabhan and L. Subramanian. An Investigation of Geographic Mapping Techniques for Internet Hosts. In *Proceedings of the ACM SIGCOMM'01*, pages 173–185. ACM Press, 2001.

[27] V.Pappas, P. Faltstrom, D. Massey, and L. Zhang. Distributed DNS Troubleshooting. In *Proceedings of the ACM SIGCOMM workshop on Network Troubleshooting*, 2004.