DNS: a statistical analysis of name server traffic at local network-to-Internet connections

Chris J. Brandhorst[1], Aiko Pras[2]

[1] Electrical Engineering, Mathematics and Computer Science, University of Twente, Campuslaan 71-105, 7522 NK Enschede, The Netherlands, c.j.brandhorst@cs.utwente.nl
[2] Electrical Engineering, Mathematics and Computer Science, Design and Analysis of Communication Systems Group, University of Twente, P.O. BOX 217, 7500 AE Enschede, The Netherlands, pras@cs.utwente.nl

Abstract. This paper puts forward a purely statistical analysis of name server traffic captured at four different locations: two links to residential networks, and two to the Dutch academic and research institute. Analysis of the system can give insight in the use and performance of the protocol, which is helpful for future improvement. Multiple analyses can show the development of the performance over time and help create quality models. The analysis shows that a little more than 12% of all queries are not answered upon. Three quarters of the lookups are successful: they give the client the correct IP address mapping for the requested hostname. 90% is answered within 275 ms, with an average of 152 ms. In 9% of all cases, clients ask for a hostname which does not exist. At one of the locations, a client is discovered which sends queries to two DNS servers at a remarkable rate: one each 11 to 22 ms.

1 Introduction

The Domain Name System is one of the most vital protocols of the modern Internet. This protocol is responsible for translating Internet addresses (e.g. `www.server.com`) to network locations, also called IP addresses (e.g. `125.114.163.15`). This translation occurs at DNS servers to which a client must send its queries, asking for the IP address of a specific Internet address. The DNS is vital, because virtually all applications that connect to the Internet (e.g. mail programs, web browsers and FTP utilities) make use of the protocol for their operation. Therefore, it is necessary that the system must be operable virtually 100% and have a high degree of efficiency, to provide the clients with low response times to there queries.

Analysis of the DNS traffic will give insight in the use of the protocol and may assist in improving it in the future or designing a new version. Multiple analyses can show the development of the use of the protocol over time and "studies involving certain DNS performance measures would be greatly strengthened by data from many locations" [1]. This, in turn, can address performance issues. Furthermore, when these statistics are known, quality models can be created for evaluating new protocols, algorithms and architectures [2].

1.1 Related Work

Past work is available on this subject. The most quoted of them was done by Danzig et al. [3]. Others are Brownlee et al. [4], Jung et al. [5] and Liston et al. [1]. This work will differ from the above in three aspects.

The first is that it will take measurements from at least four local network-to-Internet connections, all with a varying number of users, types of users and bandwidth. Jung et al. use data gathered at just two networks, both of which belong to research institutes and Liston et al. gather their data at 75 *client* locations, which is far less than the number of clients connected to the networks used in this study.

The second difference is the topographical network location at which the measurements have taken place. The data used for this research was captured at a switch which connects the local networks to the Internet. Danzig et al. and Brownlee et al. have concentrated on measurements at a DNS root server, where the first also takes three domain servers into account. Liston et al. have taken a different approach by measuring the DNS performance at client computers. One can say that this research can be placed closer to Liston's, because of the proximity of the clients to the switch.

Thirdly, only Jung et al. include (limited) basic trace statistics in their research. In this research an extensive breakdown of all DNS responses will be given.

1.2 Research Questions

There are a number of statistics that can be extracted from DNS traffic data. This analysis will include a breakdown of DNS queries into those that: are erroneous, are refused, ask for non-existing domains, are omitted because of a server failure, are not replied to at all, are of a type which is not implemented by the receiving DNS server and those that are answered normally. Besides this, information can be gathered about the delay between the sending and receiving of DNS queries, also called *latency*, and how many queries are to be processed recursively and which fraction of the total traffic can be accounted for by the DNS.

This results in the following research questions, by which the performance of the DNS, as perceived by the client, can be analyzed:

1. What is the fraction of total traffic that can be accounted for by the DNS, measured in packets as well as bytes?
2. What fraction of the received DNS queries is to be processed recursively by the DNS servers?
3. Given a collection of DNS queries, which fractions of these queries are answered normally and which fraction results in some kind of error? We will determine the fractions of the following error conditions: queries that 1) are erroneous, 2) are refused by the DNS servers, 3) ask for non-existing domains, 4) are omitted because of a server failure, 5) are not replied to at all, and 6) are of a type which is not implemented by the receiving DNS servers.

4. What is the average delay between the time that the query is sent and the answer is received? (the measured delay is the time measured at the place in the network where the DNS data is collected, *not* the latency the client observes)
5. What are the differences between these results and prior research on DNS traffic, mostly focused on [5]?

In order to answer the above research questions, the repository set up by the M2C-project (*Measuring, Modeling and Cost Allocation*) at the University of Twente will be used [6]. This collection includes data captured at four locations, at different times. The locations include Ethernet links which connect 1) a residential network of a university to the core network of this university, 2) a research institute to the Dutch academic and research network, 3) a large college to that same research network and 4) a couple of hundred ADSL customers, mostly student dorms to an aggregated uplink of an ADSL access network.

In the following chapter a general overview of the Domain Name System is given. After this, the method of research is laid out in chapter three and in chapter four the results will be presented out of which conclusions will be drawn in the final chapter.

2 The Domain Name System

The Domain Name System is a standard and is defined in [7] and [8]. Here, only an overview of the most important basic functionality is given.

The Internet's DNS function is to translate human-readable hostnames (e.g. `www.server.com`) to IP addresses (e.g. `125.114.163.15`), which are used by network-hardware to identify client computers. This translation can be done for all kinds of servers, from web- and mail- to FTP-servers and is employed by the corresponding application-layer protocols (HTTP, SMTP and FTP respectively), which are in turn used by computer applications. The application will ask the DNS for the IP address of the host of which the client entered the hostname by sending a DNS *query* onto the network. After a delay, the client will receive a *response* stating the hostname or possibly an error message.

The DNS can be used on top of both the UDP and TCP protocol.

2.1 Name Servers

To achieve its functionality, the DNS consists of a large number of *name servers* which are scattered around the world and ordered hierarchical. There is not one server that has *DNS records* for *all* hostname to IP-mappings for the Internet. Instead, a certain DNS server has records for a specific domain (like `utwente.nl`) or local network. Besides this, it can route DNS queries for mappings that that server does not contain to a DNS server which does. Therefore, there are two kinds of name servers [5]:

- **Root name servers**. If the name server in the local network of the client does not have a mapping for the requested hostname, the local name server sends a query to one of the 13 [9] root name servers. These servers contain DNS records of *authoritative name servers*, one of which is returned to the local name server.
- **Authoritative name servers**. A name server is authoritative for a certain host or domain if it contains DNS records for that host or domain or has a record for another DNS server that does has these records. This layering of DNS servers can be repeated more than once.

2.2 Recursive Queries

An example of a *recursive query* is shown in Fig. 1 [10]. The query and its response together often are called a *lookup*. The host `192.168.1.203` here requests the IP address of `www.utwente.nl`. It sends this query to its local name server, `192.168.1.1`. The local server does not have the correct mapping, so it sends a query to the root server (`b.root-servers.net`). This root server knows the address of the authoritative name server for the `.nl` *top-level-domain*, which is returned to the local name serer (`ns.domain-registry.nl`). This server in turn, when asked by the local name server, returns the address of the authoritative name server for the `utwente.nl` domain (`ns1.utwente.nl`). Finally, after querying this server, the IP-address of `www.utwente.nl` is obtained and returned to the client. This query is called recursive, because the client only sends one query and receives one answer. The local name server handles all the other queries in favor of the client. When a query is *iterative*, it means that the name server to which the query is sent will always send a response to the requesting host instead of to an other name server, whether it does have a DNS record for the requested hostname (it will send the requested IP address) or it does not (it will send the IP address of an authoritative name server). Not all name servers support recursive querying. Note that the local name server in the example only sends iterative queries.

2.3 Response Types

When a query is answered by a name server, this response is of one of the following types [8]:

- **OK**. Everything went correctly and embedded in this response packet is the IP address belonging to the hostname the requesting host supplied.
- **Format error**. The name server was unable to interpret the query.
- **Server failure.** The name server was unable to process this query due to a problem with the name server.
- **Name error**. The domain name referenced in the query does not exist.
- **Not implemented**. The name server does not support the requested kind of query.
- **Refused**. The name server refuses to perform the specified operation for policy reasons. For example, a name server may not wish to provide the information to

the particular requester, or a name server may not wish to perform a particular operation for particular data.

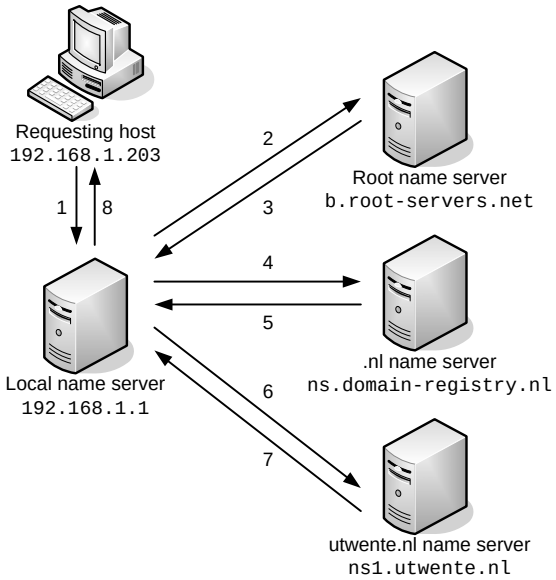


Fig. 1. An example of a recursive query [10].

2.4 Latency

The time between the sending of a DNS query and the receiving of a response is called the *latency* of that query. The latency can vary between milliseconds and tens of seconds [11].

3 Research Methodology

3.1 Used Data

In order to analyze the DNS traffic, this study makes use of the Internet data repository set up by the M2C project (*Measuring, Modeling and Cost Allocation*) [6]. This repository consists of 555 traces (in December 2004), each of which contains 15 minutes of packet data. The measurements were taken at four different locations, all in the Netherlands, and on different days and times [12].

1. “On location #1 the 300 Mbit/s (a trunk of 3 x 100 Mbit/s) Ethernet link has been measured, which connects a residential network of a university to the core network of this university. On the residential network, about 2000 students are connected,

each having a 100 Mbit/s Ethernet access link. The residential network itself consists of 100 and 300 Mbit/s links to the various switches, depending on the aggregation level. The measured link has an average load of about 60%. Measurements have taken place in July 2002" and are spread over 15 traces.

2. "On location #2, the 1 Gbit/s Ethernet link connecting a research institute to the Dutch academic and research network has been measured. There are about 200 researchers and support staff working at this institute. They all have a 100 Mbit/s access link, and the core network of the institute consists of 1 Gbit/s links. The measured link is only mildly loaded, usually around 1%. The measurements are from May to August 2003" and are spread over 183 traces.
3. "Location #3 is a large college. Their 1 Gbit/s link (i.e., the link that has been measured) to the Dutch academic and research network carries traffic for over 1000 students and staff concurrently, during busy hours. The access link speed on this network is, in general, 100 Mbit/s. The average load on the 1 Gbit/s link usually is around 10-15%. These measurements have been done from September - December 2003" and are spread over 255 traces.
4. "On location #4, the 1 Gbit/s aggregated uplink of an ADSL access network has been monitored. A couple of hundred ADSL customers, mostly student dorms, are connected to this access network. Access link speeds vary from 256 kbit/s (down and up) to 8 Mbit/s (down) and 1 Mbit/s (up). The average load on the aggregated uplink is around 150 Mbit/s. These measurements are from February - July 2004" and are spread over 102 traces.

3.2 Method of Collection

On every location, the method of collection is identical. "The measurements are performed by capturing the headers of all packets that are transmitted over the (Ethernet) 'uplink' of an access network to the Internet." [13], see also Fig. 2. To the switch (can also be a router), a measurement PC was connected to which the switch copied all packet data that was coming through. To capture the 15 minutes of packets into binary files, the utility *tcpdump* [14] was used. These traces were anonymized, compressed and published on the Internet [13], [12].

The *anonymization* is done for privacy reasons, which means 1) that only the packet headers are captured, so it is unknown *what* data is sent from the network to the Internet and 2) that the source and destination IP addresses of the packets are scrambled *consequently*, so that is unknown from and to which host the packets are sent. But within a trace, a single IP address is each time scrambled to the same anonymized IP address and two IP addresses that are within the same network, will have anonymized counterparts that also can be placed within the same network. For a more detailed explanation of this method, see [13].

In chapter 2 it was said that the DNS protocol can be used on top of both TCP and UDP, but a few samples from each location revealed zero DNS packets that made use of the TCP protocol. Therefore it is assumed in this paper that the use of this protocol for the Internet is negligible.

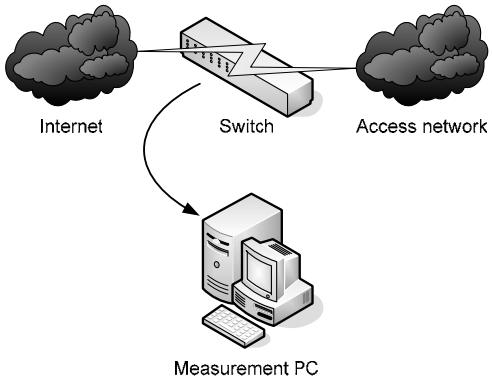


Fig. 2. The setup for the collection of the packet data [13].

3.3 Method of Analysis

For the analysis of the data, a self-built Java application called *ReadDump* was developed. This application opens a packet-trace file and analyses the packets within to retrieve the data necessary to answer the above research questions. When it is done, it writes a report to a log file. Because there are 555 traces, another program was developed to send the data files to the Java so it could analyze all data autonomously. *Runner* was written in VB.NET and for each packet trace file, it would decompress it and send it to ReadDump for processing.

4 Results

In Table 1 and 2, the statistical results of the analysis are presented.

4.1 Location #1: Server Failures

What immediately came to attention is the enormous amount of returned server failures at location #1 (see Table 1). 60.99% of all sent queries get this response. The first thought was that a broken DNS server was the cause of these high amounts of server failures. After further research [15], it seemed that not a DNS server but (most probably) one single client was the cause. This will be discussed further in section 4.3. Because this abnormal behavior was observed in 7 of the 15 traces at location #1, only the remaining 8 traces were used to calculate new statistics which are laid out in Table 2. Important to note here is that retransmissions of DNS queries are taken into account as a ‘new’ query. The retransmissions cannot be measured because of the anonymized nature of the repository; it cannot be shown when a query is retransmitted, because of the lack of the real data of the packets.

Table 1. Trace statistics including 7 traces in which a client with abnormal behavior is detected. Percentages are with respect to: (1) Total bytes; (2) Total # of packets; (3) # of DNS packets; (4) # of Queries.

	Location #1	All
Date	2002/05/23 – 06/26	
Total bytes	282,717,144,046	2,670,914,636,701
DNS bytes (1)	146,613,052 (0.05%)	1,279,580,232 (0.05%)
Total packets	429,846,054	4,085,445,051
DNS packets (2)	1,556,032 (0.36%)	10,450,329 (0.26%)
Queries (3)	780,538 (50.16%)	5,527,878 (52.90%)
Recursive (4)	670,425 (85.89%)	1,011,762 (18.30%)
Iterative (4)	110,113 (14.11%)	4,516,116 (81.70%)
Unanswered (4)	5,044 (0.65%)	605,427 (10.95%)
OK (4)	233,965 (29.97%)	3,715,741 (67.22%)
Format error (4)	667 (0.09%)	167,726 (3.03%)
Server failure (4)	476,055 (60.99%)	536,729 (9.71%)
No such name (4)	64,171 (8.22%)	450,601 (8.15%)
Not implemented (4)	274 (0.04%)	4,145 (0.07%)
Refused (4)	362 (0.05%)	47,509 (0.86%)
Average latency (ms)	687, σ = 364	156, σ = 206

4.2 Statistics

4.2.1 DNS Fraction of Total Traffic

The percentage of total bytes which are DNS traffic varies little; between 0.01% and 0.26%, on average 0.05%. A little more variation is observed for the percentage of total packets which are DNS packets; between 0.03% and 1.41%, on average 0.23%. The data shows that there are two locations which in both cases have (by far) the least percentage of DNS traffic: locations #1 and #4.

4.2.2 Recursive or Iterative

Whether the queries are sent recursive, varies much; between 0.86% and 97.53%. Locations #1 and #4 have high values for recursive queries, #2 and #3 show many iterative queries. This difference most probably comes from the fact that at locations #2 and #3, a DNS server that can receive recursive queries is present *inside* the local network. It can be assumed that the clients of those networks use those recursive DNS servers as a standard. This implies that virtually all DNS queries that are directed outside the local networks (the queries that are observed) will be iterative, because that traffic travels from the local DNS servers to a root server (see section 2.2).

In contrast, at locations #1 and #4, the local DNS servers are placed *outside* the network. If we assume the same as for locations #2 and #3, this implies that *all* queries will be directed out of the network and measured, and because of the standard setup of the clients, these queries will be recursive. Still, at location #4, the amount of recursive and iterative queries is almost equal. The reason for this may be a privately setup DNS server inside the network.

[5] reports similar variation in the number of recursive or iterative queries in the observed networks.

Table 2. Corrected trace statistics. Percentages are with respect to: (1) Total bytes; (2) Total # of packets; (3) # of DNS packets; (4) # of Queries.

	Location #1	Location 2	Location 3	Location 4	All
Date	2002/05/23 – 06/26	2003/05/13 – 08/28	2003/09/02 – 11/25	2004/02/04 – 05/07	
Total bytes	140,359,988,095	115,895,661,957	950,853,472,873	1,321,448,357,825	2,528,557,480,750
DNS bytes (1)	31,206,868 (0.02%)	299,235,088 (0.26%)	754,663,500 (0.08%)	79,068,592 (0.01%)	1,164,174,048 (0.05%)
Total packets	220,314,110	167,772,874	1,346,774,765	2,141,051,358	3,875,913,107
DNS packets (2)	228,883 (0.10%)	2,365,875 (1.41%)	5,885,995 (0.44%)	642,427 (0.03%)	9,123,180 (0.24%)
Queries (3)	115,679 (50.54%)	1,280,706 (54.13%)	3,130,427 (53.18%)	336,207 (52.33%)	4,863,019 (53.30%)
Recursive (4)	112,825 (97.53%)	11,011 (0.86%)	149,367 (4.77%)	180,959 (53.82%)	454,162 (9.34%)
Iterative (4)	2,854 (2.47%)	1,269,695 (99.14%)	2,981,060 (95.23%)	155,248 (46.18%)	4,408,857 (90.66%)
Unanswered (4)	2,475 (2.14%)	195,537 (15.27%)	374,859 (11.97%)	29,987 (8.92%)	602,858 (12.40%)
OK (4)	72,308 (62.51%)	868,755 (67.83%)	2,404,557 (76.81%)	208,464 (62.00%)	3,554,084 (73.08%)
Format error (4)	291 (0.25%)	58,050 (4.53%)	90,234 (2.88%)	18,775 (5.58%)	167,350 (3.44%)
Server error (4)	1,761 (1.52%)	28,619 (2.23%)	28,181 (0.90%)	3,874 (1.15%)	62,435 (1.28%)
No such name (4)	38,534 (33.31%)	127,214 (9.93%)	225,566 (7.21%)	33,650 (10.01%)	424,964 (8.74%)
Not implemented (4)	118 (0.10%)	75 (0.01%)	2,348 (0.08%)	1,448 (0.43%)	3,989 (0.08%)
Refused (4)	192 (0.17%)	2,456 (0.19%)	4,682 (0.15%)	40,009 (11.90%)	47,339 (0.97%)
Average latency (ms)	919, σ = 399	114, σ = 64	85, σ = 38	332, σ = 344	152, σ = 205

4.2.3 Answering rates

All locations show that a little more than 50% of all DNS packets consist of queries, implying that there are few queries that are not answered. Location #2 tops with 15.27% of all queries not being answered, location #1 has the least; 2.14%. On average this is 12.40%. Compared to the results of Jung et al., these DNSs perform far better; [5] reported 20.1-23.5% unanswered.

4.2.4 Return types

Of the queries that *were* answered, on average 73.08% were queries for hostname mappings that could be correctly retrieved by the DNS server(s). Jung et al. found similar values (twice 64% for two locations) and one far worse (36% for the third location). As can be classified as logical, the percentage of no such name responses is not negligible. On average 8.74% of all requests are for hostnames that doesn't exist. Location #1 tops here with 33.31%. [5] also reports a large variation (10% to 42%). Their explanation lies in "inverse lookups for IP addresses with no inverse mappings" and "particular invalid names such as `loopback`, and […] records that point to names that do not exist." Because of the anonymized nature of the traces, it cannot be confirmed that the same causes for these rates are applicable here.

Not implemented and refused are returned rarely, with an exception at location #4, which has a high refused rate of 11.90%. The reason for this cannot be uncovered. Format error is returned a little more, with tops of 4.53% and 5.58%. The amount of server failure responses also is quite low.

4.2.5 Latency

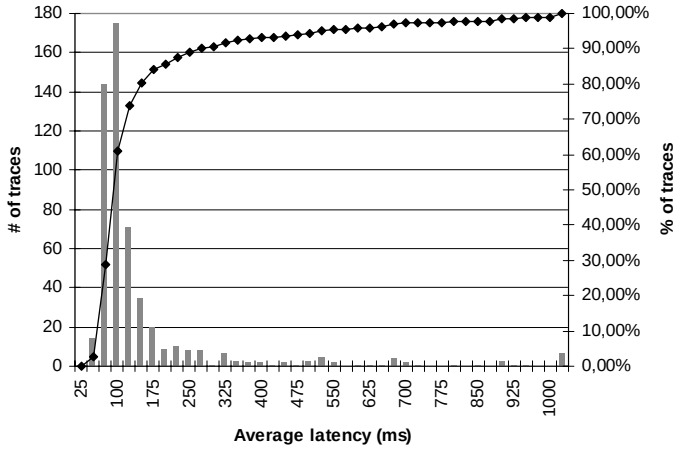
The observed latency times are mostly low; on average 152 ms. Location #1 tops with 919 ms. Compared to some prior studies, this is far better. Wills and Shang [as quoted by [1]] "report lookup times exceeding 2.0 seconds for as many as 29% of lookups to random servers, and Cohen and Kaplan report lookup times exceeding 3.0 seconds for

as many as 10% of lookups." [5] reports that 90% of all lookups on the MIT network had a latency of 447 ms in January 2000 and 11 months later, this number was about 2.5 times greater. In our study merely 7 traces had an average latency of over 1 second and only one over than 2 seconds. Also, 90% of the traces have an average latency of 275 ms or less. 32% of all traces time between 75 and 100 ms. See Fig. 3.

It can be seen that at locations #1 and #4, the average latency is far greater compared to the other locations. If the assumptions made in 4.2.2 are correct, this difference can be addressed to the fact that the fraction of observed DNS recursive queries at locations #1 and #4 is far greater. A client that sends a recursive query gets a response only after all iterative queries that are needed to resolve the client's query are resolved by the recursive DNS server. So the total latency for a recursive query is higher than for an iterative query. This high latency can be seen at location #1 and (to a lesser degree, because of the higher fraction of iterative queries) at location #4, where the recursive queries from the clients to the DNS servers are observed. This in contrast to locations #2 and #3, where the observed queries are (virtually all) the iterative queries sent from the local DNS servers to the root servers.

The higher latency standard deviation at locations #1 and #4 can be explained by noting that one recursive query requires that more iterative queries are performed by the DNS servers than are needed for an other. The more iterative queries with a deviation from the average, the higher the total standard deviation.

Fig. 3. Cumulative latency distribution. The last column represents all traces that have an average latency of 1000 ms or greater.



4.3 Client with abnormal DNS behavior

It seemed that virtually 100% of the server failure responses within the 7 discarded location #1 packets (see section 4.1) were directed to one single client. This cannot be said with full certainty, because in every trace the IP addresses were scrambled differently, but it is very reasonable to assume this.

This client was in each trace receiving DNS server failures from two different name servers. This receiving furthermore occurred at an exceptionally high rate: one received packet per 22 to 11 ms. This rate of reception implies that the client should also have approximately the same rate of sending DNS queries.

So all 'evidence' points to a client that was either deliberately sending all these queries to the name server for some reason, or who's computer was misconfigured, which caused the continuing sending of queries. It was revealed that the name servers on the university network are specifically configured to answer queries for certain zones only when these queries come from specific clients [15]. If any other client asks for an IP from one of these zones, a server failure is returned. This complies to a certain extent with RFC 2308 [16], which declares that server failures fall in two classes, one of which states that "This may be where it (the name server, red.) has been listed as a server, but not configured to be a server for the zone, or where it has been configured to be a server for the zone, but cannot obtain the zone data for some reason." This 'some reason', most probably, is the fact that the DNS servers may not return the IP mappings for certain zones to any clients but the specified ones.

5 Conclusion

This paper put forward a statistical analysis of DNS data captured at four different locations: two links to residential networks, and two to the Dutch academic and research institute. The performance of the DNS, as seen by the client, was analyzed by measuring the latency and a breakdown of the various response types of the protocol. The networks analyzed here show a better DNS performance than prior research.

The percentage of queries that never receives a response is far lower than the 20+% that [5] measured. This study reports an average of a little more than 12%, which can be called acceptable, because of the non-retransmissive nature of the UDP protocol.

Three quarters of all queries are responded to with a correct IP mapping with on average a latency of 152 ms. 90% of these lookups take 275 ms or less, which is at least twice as fast as earlier research showed.

Clients ask in 9% of all cases for an IP address for a hostname that does not exist. The amount of queries that are responded to by one of the other return types is very small. Only the amount of format errors is names worthy: a little less than 3.5%. This can be accounted to bugs in client software and transmission errors.

Further research is suggested on DNS data that is not anonymized, so the reasons for the observed format and no such name errors can be analyzed. Location #4 may look into their network in order to find the reason for the large amount of refused responses.

The strange behavior of the client observed in the traces of location #1 could be worth to investigate further. Do these actions occur more often and do they bring down the performance of the DNS, or more broad, of the entire network link? Besides this, one might think of changing the settings of the DNS server in question, so that it responds with a refused message in these situations, which is more logical than a server error response.

Acknowledgements

We would like to thank Remco van de Meent for pointing us to and making available the M2C repository and collecting the data. We also thank Remco van de Meent for his extra information on the DNS system at location #1. Furthermore, thanks go out to the reviewers for useful comments on the draft version of the paper.

References

1. Liston, R., Srinivasan, S., Zegura, E., *Diversity in DNS performance measures*, Proceedings of the second ACM SIGCOMM Workshop on Internet measurement, November 2002, Pages 19-31
2. ACM SIGCOMM, *Call for papers: Measuring the Internet's Vital Statistics*, http://www.acm.org/sigs/sigcomm/ccr/ivs/IVS-CFP.pdf, available: September 5, 2004
3. Danzig, P.B., Obraczka, K., Kumar, A., *An analysis of wide-area name server traffic: a study of the Internet Domain Name System*, ACM SIGCOMM Computer Communication Review, Conference proceedings on Communications architectures & protocols, Volume 22 Issue 4, October 1992, Pages 281-292
4. Brownlee, N., Claffy, K., Nemeth, E., *DNS measurements at a root server*, Global Internet 2001, November 2001
5. Jung, J., Sit, E., Balakrishnan, H., Robert Morris, *DNS Performance and the Effectiveness of Caching*, IEEE/ACM Transactions on Networking, Volume 10, No. 5, October 2002, Pages 589-603
6. *M2C: Measuring, Modeling and Cost Allocation for Quality of Service*, http://arch.cs.utwente.nl/projects/m2c/, updated: March 2, 2004, available: September 22, 2004
7. Mockapetris, P.V., *Domain names - concepts and facilities*, RFC 1034, available at: http://www.rfc-editor.org/rfc.html, November 18, 2004
8. Mockapetris, P.V., *Domain names - implementation and specification*, RFC 1035, available at: http://www.rfc-editor.org/rfc.html, November 18, 2004
9. Root-servers.org, *Root Server Technical Operations Assn*, available at: http://www.root-servers.org, November 18, 2004
10. Wanrooij, W. van, *DNS zones revisited*, Proceedings of the 2nd Twente Student Conference on Information Technology, January 21, 2005
11. Kurose, J.F., Ross, K.W., *Computer Networking: A Top-Down Approach Featuring the Internet*, Addison-Wesley, 2003
12. *M2C Measurement Data Repository*, http://m2c-a.cs.utwente.nl/repository/, updated: unknown, available: September 22, 2004
13. Van de Meent, R., *M2C Measurement Data Repository*, University of Twente, Enschede, The Netherlands, December 22, 2003
14. *TCPDUMP public repository*, http://www.tcpdump.org/, updated: June 6, 2004, available: September 10, 2004
15. Van de Meent, R., University of Twente, email communication during October and November 2004
16. Andrews, M., *Negative Caching of DNS Queries (NCACHE)*, RFC 2308, available at: http://www.rfc-editor.org/rfc.html, November 28, 2004