A Global Reference Model of the DNS

Y. Koç[1,2,⋆], A. Jamakovic[2,⋆⋆] and B. Gijsen[2,⋆⋆⋆]

[1]Delft University of Technology
Deparetment of Multi-actor Systems, Section System Engineering
P.O. Box 5015, 2600 GA Delft, The Netherlands
[2]TNO: Netherlands Organization for Applied Research
P.O. Box 5050, 2600 GB Delft, The Netherlands

Abstract. The Domain Name System (DNS) is a crucial component of today's Internet. At this point in time the DNS is facing major changes such as the introduction of DNSSEC and Internationalized Domain Name extensions (IDNs), the adoption of IPv6 and the upcoming extension of new generic Top-Level Domains. These changes can have impact on the behaviour of the DNS. In this paper we present a first global DNS reference model with the aim to predict the DNS traffic behaviour under specific conditions. In fact, this quantitative model is intended to be used for analyzing what-if scenarios. For example, how will DNS query rates at the recursive and authoritative name servers increase in case DNSSEC validation errors lead to sending more Servfail responses towards DNS clients? The DNS reference model takes into account all relevant components present in the DNS architecture. To characterize the system variables describing the query behaviour at each of these independent system components, we statistically analyze real world data from recursive resolvers. In addition, we use experimental results that characterize DNS client behaviour and data from the literature to characterize the behaviour of authoritative name servers. In order to validate our reference model we compare the model predictions to the real world data. The validation results show that the model predictions are rather accurate. At the end of the paper we present a specific what-if scenario to demonstrate the applicability of the model.

1 Introduction

In the last decade the Internet gained more and more importance such that it became an essential part of our society. As a consequence, the stability of the Internet including the Domain Name System (DNS) as a key Internet component, is crucial. The DNS is primarily used to translate the human readable domain names into the corresponding Internet protocol (IP) addresses, which are used for the routing purposes. For instance, thanks to the DNS, one just needs to recall "cnn.com" instead of "157.166.255.19". The data for this mapping between domain names and IP addresses is stored in a tree-structured distributed database, where the mapping responsibility for each domain is assigned to designated authoritative name servers (NSs). The authoritative NSs are thus

⋆ email: yakupkoc@gmail.com
⋆⋆ email: almerima.jamakovic@tno.nl
⋆⋆⋆ email: bart.gijsen@tno.nl

assigned to be responsible for their particular domains which typically are the root, top-level domain (TLD) and second-level domain (SLD). This mechanism makes the DNS distributed and resilient against failure [7].

The top layer of DNS hierarchy is facing major changes: cryptographically signing the authoritative NS with DNSSEC, deploying new generic TLD names by allowing domains such as .bank as well as deploying Internationalized Domain Name extensions (IDNs), including non-ASCII characters. In addition, the uptake of IPv6 that is required to make the Internet future proof has impact on the DNS. These developements can have consequences to the stability of DNS and indirectly, to the continuity of the entire Internet. For example, the query load towards the authoritative NS is expected to increase [12, 10] and a specific type of DNS query response, i.e. Servfail responses, is expected to increase significantly [9]. All the mentioned challenges have triggered the need for public awareness and more research on proper understanding of the DNS behaviour in the increasingly evolving DNS landscape.

In this paper we present a global DNS reference model aimed at analysing what-if scenarios. For example, how will DNS query rates at the recursive and authoritative name servers increase in case DNSSEC validation errors lead to sending more Servfail responses towards DNS clients? The contribution of this paper is twofold. First, we present a global reference model taking into account the typical DNS architecture: starting from client's OS with its stub resolver and application browser, then recursive resolver present mostly at an Internet Service Provider (ISP), to the authoritative NS which include the root, TLD and SLD servers. To characterize the system variables describing the query behaviour at each of these independent system components, we statistically analyze real-world data from recursive resolvers. The data is provided by SURFnet who serves a large number of academic customers in The Netherlands. In addition, we use a characterization of DNS client behaviour from an experimental study by TNO and SIDN, and data from the literature to characterize the DNS behaviour of authoritative name servers in more detail. Second, we validate our reference model by using Monte Carlo simulation to generate DNS behaviour predictions and compare them to the real world data. The validation results show that the model predictions are rather accurate. In addition to these main contributions we discuss shortcoming related to the real-world data and possible extensions of the model. Finally we present a specific what-if scenario to demonstrate the applicability of the model. Overall, this paper establishes a path towards the proper understanding of the DNS behaviour in the increasingly evolving DNS landscape.

The paper is organised as follows. Section 2 gives an overview of the former work on modelling efforts in the DNS community, which is followed by the description of our DNS reference model in Section 3. Section 4 explains the overall operation of the model and the model validation with real-world data is presented in Section 5. Section 6 treats the usage of the DNS reference model for the impact assessment of the increase of a specific DNS query response by a certain percentage. In Section 7 we present a summary of our results and identify further research for the DNS reference model.

2 Related work

We already mentioned several important works in the field of measurement and characterisation of DNS traffic. They all present and discuss the DNS query behaviour and corresponding data analysis tools, focussing mainly on the upper DNS hierarchy. For example, a vast majority of papers attempt to address the question of characterisation of DNS traffic at the root: CAIDA and the Measurement Factory have done numerous monitoring studies on the traffic at the root NS, among which the more recent ones are [4, 10, 13]. Besides this traffic analysis at the core component of the upper DNS hierarchy, authors attempt to address the question of characterisation of DNS traffic at the recursive resolver and at the client side. For example, in [3] authors give a statistical analysis of DNS traffic at the recursive resolver and in [15, 1] authors compare the performances of caching recursive resolvers with respect to query response time and querying behaviour towards the root, while in [2] authors attempt to characterize the querying behaviour of specific client types (e.g. a client with Linux as OS and Firefox as application browser). Equally relevant for our work are those publications that present experimental studies carried out to understand the effectiveness of DNS caching [6, 5, 16]. Furthermore, many authors point to the lack of data with which to do the long-term research and analysis in support of DNS performance, stability and security, as being one of the main concerns of the DNS community. For example, in [10] authors rise the awareness of this problem to evaluate the DNS during the expected transition phase the DNS is facing in a short time interval.

Although there is a substantial literature on the characterization of the traffic and querying behaviour of each individual hierarchical level of the DNS, we are not aware of much work that attempted to study the entire DNS. Perhaps good to mention here is that there are several works, similar to [1], which pinpoint the limitations of the current DNS deployment and its foremost influence on the performance of applications. However, to this day there is a little understanding of the way the DNS behaves as whole, especially when the expected changes are incorporated and the querying mechanism need further detailed analysis. In this respect, by introducing a reference model of the entire DNS, we make an important step in fundamentally understanding the DNS behaviour in the increasingly evolving DNS landscape.

3 The DNS reference model

3.1 General features and assumptions

Our primary concern is the scalability of the DNS system when for example redundant DNS traffic towards the recursive resolvers and authoritative NS occurs. We therefore create a reference model at the flow level, being only interested in the query flow distribution at an arbitrary point in time. Consequently, the time notion does not play a role and the distribution of the DNS queries is only dependent on the behaviour of various components of the DNS system. We therefore chose to distinguish between the following generic components in the DNS: a) client with its OS (and the corresponding stub resolver) and application browser, b) recursive resolver, and c) authoritative

NS with the root, TLD and SLD NSs. Figure 1 shows these generic components of the DNS system and also the interactions between them. In our model we assume that all clients (of the same configuration type) are independent and have identical querying behaviour, so they can be modelled as one client. The same holds for recursive resolvers and the authoritative NS being either the root, TLD or SLD. This assumption enables us to control the entire system by adjusting only input parameters for a single client, a single recursive resolver, and single root, TLD and SLD. Furthermore, we model the querying behaviour with a system variable referred to as Query Multiply Factor, which in fact reflects how many queries will be reinitiated by a component in reaction a negative query response. Then, the caching behaviour of a component: it depends strongly on TTL values and inter-arrival time of the queries, having a stochastic and state dependent behaviour [6]. However, we do not model the caching mechanism as a state, i.e. weather the domain name is in the cache or not, but rather by a probability that a queried domain name will be in the cache of the corresponding system component. We call this system variable Cache Hit Ratio. Finally, we assume that the type of a response to an initial query at the authoritative NS follows a certain distribution. This variable is referred to as Response Distribution at Authoritative Name servers. Values of these system variables are obtained by analyzing the real-world data which consists of 30.000 DNS packets, captured at an UNBOUND resolver for the duration of 14 sec.

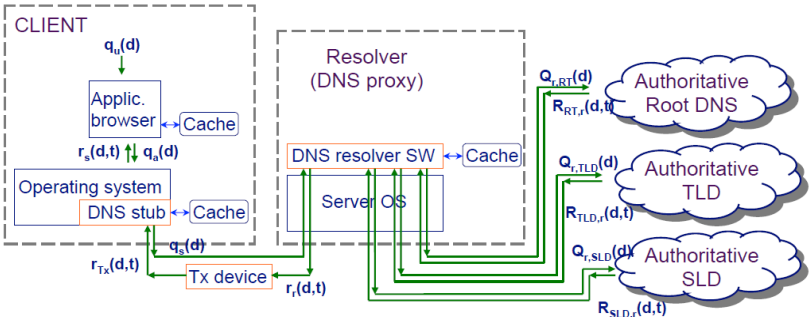

Fig. 1. The DNS overall modeling structure

3.2 System variables

Cache Hit Ratio is the value which indicates the probability that a queried domain name will be in the cache of a system component under consideration. The notion of Cache Hit Ratio is different for client and recursive resolver side, therefore we treat them separately. The values for the Cache Hit Ratio at the client side indicate the probability that a query will be answered with a certain response type from the cache. Whether the received DNS data can be cached or not depends on the response type. For instance, application browsers cache only the DNS response types that provide valid data (i.e. Valid, Valid>512B and Truncated), and the NXdomain response type.

The OS and application browser Cache Hit Ratio's are rather complicated to determine. Therefore, considering the relative scale nature of the DNS reference model, we assume "rule of thumb" values for OS and application browser Cache Hit Ratio's. These values are given in Table 1 and Table 2. Note please that the model has the possibility of filling in the missing values, as soon as they are available to the DNS community.

Table 1. Cache Hit Ratio values for three application browser types

Response type	IE8(%)	Firefox(%)	Safari(%)
Total	25	25	25
Valid	22	22	22
Valid (>512B)	1	1	1
NXdomain	1	1	1
Truncated	1	1	1

Table 2. Cache Hit Ratio values for four OS types

Response type	Windows XP(%)	Windows 7(%)	Linux(%)	MAC OSX(%)
Total	25	25	0	25
Valid	22	22	0	22
Valid (>512B)	1	1	0	1
NXdomain	1	1	0	1
Truncated	1	1	0	1

In Table 1 and Table 2, "Total" stands for the amount of the total traffic which will be responded from the application browser/OS cache. Correspondingly, the percentage of 22% for example for the IE8 application browser indicates the amount of traffic that will be responded with the "Valid" response type. The Cache Hit Ratio values for OS and application browser are relatively smaller than the recursive resolver Cache Hit Ratio values since these are client specific caches.

The Cache Hit Ratio values at the recursive resolver are rather different from the Cache Hit Ratio at the client side. Queries arriving at the recursive resolver are classified into four different groups from the caching point of view: a) Non-cached queries are queries which are not in the cache. These queries have to be sent to the root directly and domain name resolution will be performed by the resolver until whole name is resolved. b) TLD-cached queries are those whose top level domain is known by caching resolver. This means that TLD-cached queries will be sent directly to TLD NS by skipping the root. c) SLD-cached queries will be directly sent to SLD NS. TLD and SLD of those queries are known by caching resolver. d) Domain-cached queries occur when the entire request is in the cache. The probability that an incoming query will be located in one of these groups is given by the system variable Cache Hit Ratio. The Cache Hit Ratio values for the UNBOUND resolver are determined by analyzing the SURFnet data captured at an UNBOUND resolver. This data set consists of 300.000 DNS packets which we divide in 10 smaller data subsets of 30.000 DNS packets. For one of the subsets we

give the Cache Hit Ratio values for UNBOUND recursive resolver in Table 3. Besides this common resolver type, we also leave the possibility of having another type of the recursive resolver, for example BIND9.

Table 3. Cache Hit Ratio values for UNBOUND recursive resolver.

Cached Domain	UNBOUND(%)
TLD-cached	4.1
SLD-cached	41.1
Domain-cached	54.7
Noncached	0.1

Table 4. Normal distribution with mean and variance of Cache Hit Ratio for UNBOUND

Cached Domain	Mean(%)	Variance(%)
TLD-cached	4.5	0.51
SLD-cached	38.5	9.11
Domain-cached	56.9	12.37
Noncached	0.11	0.005

To bring the stochastic nature in the DNS reference model, we find distributions for each of the four caching types of the Cache Hit Ratio by analyzing the 10 data sets of 30.000 DNS packets. We test first whether the Cache Hit Ratio values, obtained from each set, are independent. The independency is tested by using Von Neumann test [8]. Distributions for each of the four Cache Hit Ratio caching types is estimated and verified by using distribution fitting techniques. It is important to note that our sample size is relatively small, i.e. n=10. However, we use Shapiro-Wilk normality test [11] for which the sample size is large enough, to conclude that the each of the four Cache Hit Ratio groups is normally distributed. Additionally, we verify this assumption by using quantile-quantile (Q-Q) plots in which we show that the plots are almost linear, pointing to the almost identical behaviour of the two compared distributions. This is given in Figure 2. Table 4 gives the mean and the variance of the estimated distributions for each of the four Cache Hit Ratio caching types.

Response Distribution at Authoritative Name servers is the system variable which indicates the fraction of response types that are given, in response to incoming initial queries, at the authoritative NS. These values are different for the root, TLD and SLD NSs. Distribution values are determined by analyzing UNBOUND data sets of 30.000 DNS packets. These values are given in Table 5b while in Table 5a a detailed breakdown of response types at the authoritative NSs is given. It is interesting to see that just seven initial queries are sent to the root and all these queries are replied by NXdomain response type. The latter observation points to the proper working of the caching mechanism of UNBOUND recursive resolver while former observation is due to the fact that our data set covers 14 seconds of DNS traffic. Since our dataset is not large enough to determine Response distribution at root NS, we use the values which are derived from

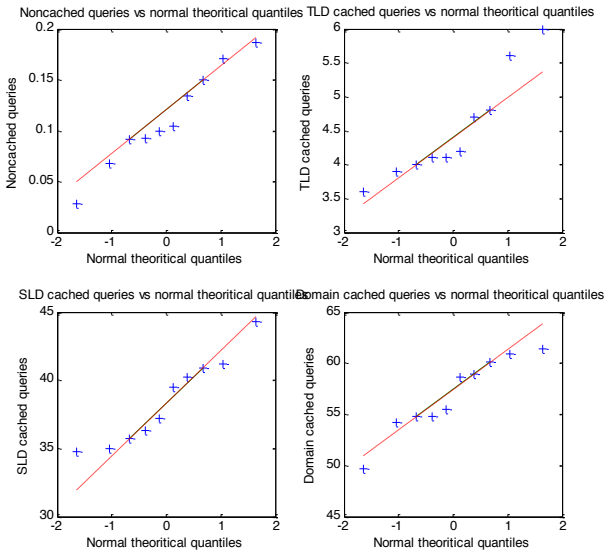

Fig. 2. Q-Q plots of resolver Cache hit ratio values.

[14]. These values are included in Table 5b. SLD NS is the last step in the domain name resolution process. We therefore observe the diversity in SLD response types unlike it is the case for TLD responses.

For this system variable we also find distributions for each of the four response types, again by analyzing the 10 data sets. Following the process explained previously, we first test the independency and then make sure that the obtained distributions are validated by using Shapiro-Wilk normality test and Q-Q plots. We found that the obtained distributions for each of the four response types follow normal distributions with the mean and the variance given in Table 6a and Table 6b, for TLD and SLD responses, respectively.

Query Multiply Factor is a system variable which indicates how many queries will be reinitiated by a component in reaction to a negative response. In other words, it reflects how the component behaves when it receives a negative response to a query. Determining the values for this system variable involves the detailed characterisation of the querying behaviour at both the client and the recursive resolver.

For the client, the experiments in the lab environment have shown that when a negative response is received for an initial query, the client may automatically resend new identical repeat queries [2]. The amount of repeat queries depends strongly on the type of client's OS and application browser: clients with various OS and application browser combinations react differently when they receive different type of responses for their

Table 5. Values for Response Distribution at Authoritative Name servers.

Response Type	Root	TLD	SLD
Referrals	0	377	741
A	0	0	1072
AAAA	0	0	20
CNAME	0	0	673
MX	0	1	23
PTR	0	2	105
NXdomain	7	31	510
Not Imp.	0	0	89
Refused	0	5	270
Servfail	0	2	199
NS	0	0	3
SOA	0	0	1
TXT	0	0	6
Format Error	0	0	5

(a)

Response Type	Root(%)	TLD(%)	SLD(%)
Valid	8.1	90.9	71.1
NXdomain	91.5	7.4	13.7
Servfail	0.4	0.5	7.9
Refused	0	1.2	7.3

(b)

Table 6. Normal distribution of responses with their mean and variance at TLD(a) and SLD(b).

Response Type at TLD	Mean(%)	Variance(%)
Valid	94.7	3.7
NXdomain	5.0	3.2
Servfail	0.2	0.04
Refused	0.05	0.02

(a)

Response Type at SLD	Mean(%)	Variance(%)
Valid	80.2	3.23
NXdomain	16.9	2.23
Servfail	1.5	0.13
Refused	1.3	0.04

(b)

initial queries. Table 7 displays Query Multiply Factors for any possible response type and for different application browsers and OS types [2]. It should be noted that the depicted numbers also include the initial query, e.g. in case of the Servfail response, a Linux-Firefox client will send in total eight queries, including the initial query.

Table 7. Query Multiply Factor values for various client's OS and application browser types.

Response Type	Windows XP	Windows 7	Linux	MAC OSX	IE8	Firefox	Safari
Valid	1	1	1	1	1	1	1
NXdomain	1	1	2	2	1	2	1
Partial	1	1	2	2	1	2	1
Servfail	1	1	4	4	1	2	1
Time-out	4	4	4	4	1	2	1
Refused	4	4	4	4	1	2	1
Truncated	2	2	2	2	1	1	1

To understand the querying behaviour of the recursive resolver, the analysis of the data of the two most popular resolvers is performed: UNBOUND and BIND9. The result, Query Multiply Factor for the recursive resolvers, is given in Table 8.

Table 8. Values for Query Multiply Factor for the two recursive resolvers.

Response Type	UNBOUND	BIND9
Valid	1	1
NXdomain	1	1
Partial	1	1
Servfail	5	2
Time-out	7	7
Refused	5	1
Truncated	1	1

4 Operation of the DNS model

As previously mentioned, the model considers the typical DNS architecture: a) client with its OS (and the corresponding stub resolver) and application browser, b) recursive resolver, and c) authoritative NS with the root, TLD and SLD NSs. Figure 3 depicts the DNS reference model as it is implemented in the Microsoft Excel. As seen in Figure 3, the left hand side (i.e. the client side) of the reference model is divided into three different parts: a) query to root, b) query to TLD c) query to SLD. The aim of doing this partition was to be able to determine the number and the sort of response types going back to the client from the root, TLD and SLD NSs. Following this objective, the operation of the DNS reference model will be divided in three different steps: step I)

the initial queries are going from the client side to the authoritative NS side, step II) the responses to the initial queries are returned from the authoritative NS side to the client side, and step III) the repeat queries due to the negative responses are reinitiated from the client side to the authoritative NS side. In the rest of this section, the operation of the DNS reference model will be explained by considering each step separately. The model will be explained by using the following input parameters, depicted in Table 9.

Table 9. Input parameters of reference model.

Number of simultaneously active DNS clients	1000
Fraction of IPv6 clients wrt to the total number of clients	10%
Number of simultaneously active recursive resolvers	100
Primary & secondary NS: average number	1

Step I In the first step, the initial queries are generated by the client and sent to the authoritative NS via client's OS, application browser, and recursive resolver, respectively. This process can be observed at the first row of the DNS reference model in Figure 3. In this example the client generates 1.1 qpt. It sends queries towards the application browser which forwards 0.83 qpt to the OS of the client. Note the difference between the two query rates which is due to the caching property of the Firefox application browser. As shown in Table 1, Firefox caches 25% of the total queries, meaning that it handles 25% of the incoming queries by itself and 75% of the queries are forwarded to the OS of the client. In Figure 3, the number of incoming queries at the OS, which is Linux, is equal to the outgoing query number in Linux. This is because Linux does not implement caching, as shown in Table 2. Then, at the recursive resolver there are 825 qpt which is due to the in Table 9 given number of simultaneously active clients. Afterwards, queries arriving at the recursive resolver will be classified into four different groups. The distribution of the queries over different classes is based on the system variable Cache Hit Ratio for the resolver, which is given in Table 3. According to the Table 3, 0.1% of the queries belong to the Non-cached group (i.e. the queries will be forwarded directly to the root) and 4.1%, 41.1% and 54.7% to respectively TLD, SLD and the Domain-Cached group. Consequently, 54.7% of the queries will be directly answered by the recursive resolver while 45.3% of the queries will undergo the recursive resolution process. Once the recursive resolution has been initiated at the root, it will be performed until the entire domain name is resolved. This means that the queries which are responded at the root with the Valid response type, will be sent to the TLD NS by the recursive resolver. The queries which are again qualified as valid at the TLD NS will be sent to the SLD NS. After receiving the response from the SLD NS, the domain name resolution process for the initial queries will be completed. The distribution of response types at the root, TLD and SLD can be found by using the system variable Response Distribution at Authoritative Name servers, given in Table 5b. The total number of the queries going from one particular UNBOUND resolver to the root, TLD and SLD NSs is found to be respectively 0.83, 40.49 and 368.2 qpt. Recall that 454.6 qpt

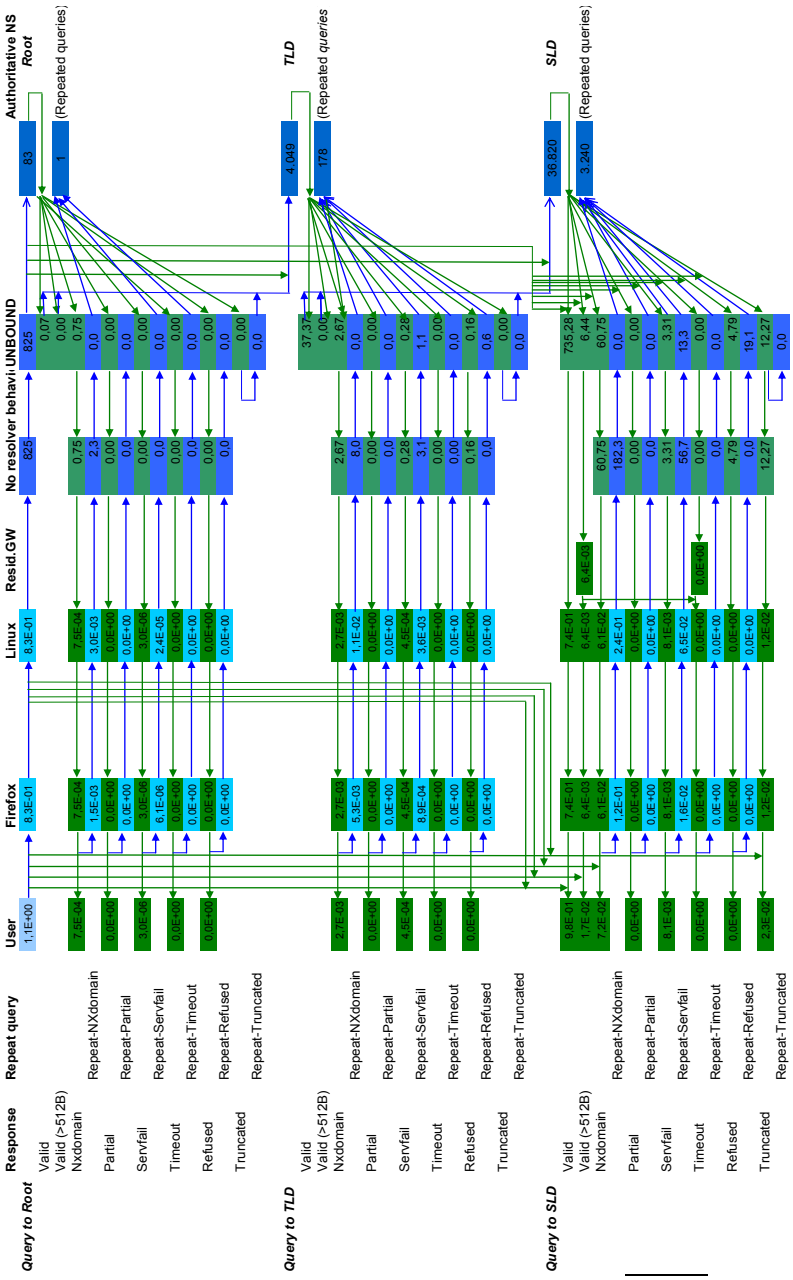

Fig. 3. The DNS reference model overview as implemented in Microsoft Excel.

will be answered by UNBOUND itself. Taking into account the input stating that there are 100 UNBOUND resolvers querying the root, TLD and SLD NSs, the total numbers of the initial queries at root, TLD and SLD NSs is 83, 4049 and 36820 respectively.

Step II In the second step, the responses from the authoritative NS will be sent back to the client. The response stream from the authoritative NS to the client is classified, and this classification is based on the response type. We assume that the authoritative NS will answer all the queries. As seen in Table 8, for each Servfail response, UNBOUND initiates four extra repeat query towards the authoritative NS, while for each Timeout response six new repeat queries will be initiated. In the DNS reference model, we assume that the repeat query will have the same response as the initial query. Therefore, in total five Servfail responses will be gathered at the UNBOUND although just one of them is sent back to the client. The same will be done for Timeout responses, i.e. UNBOUND initiates six extra repeat query towards the authoritative NS and only one response will be sent back to the client. Following this line of reasoning, negative responses from the root, TLD and SLD NSs are sent back from UNBOUND to the client side while positive ones are only sent after the recursive resolution process has been completed. As a consequence of the assumption that the repeat queries will have the same response as the initial queries, positive responses at SLD (resulting in a completed recursive resolution process) can only be result of the positive responses starting from the root. Recall that we assumed that each particular UNBOUND serves 1.000 identical users simultaneously, therefore the number of responses at OS (in this example Linux) can be found by simply dividing the value at the resolver by 1.000. These responses are sent from OS to the application browser (Firefox) and from the application browser to the user.

There are two important points that have to be mentioned about the transferring Valid responses to the user. The first point is about a fraction of the Valid>512B responses which leads to the Timeout responses when going from the recursive resolver to the OS. This point is included in the reference model so as to be able to analyze the effect of the residential gateways which can block the packets with size larger than 512B. In such a case, a Valid>512 response is perceived and treated as a Timeout response by the client. The second important point concerns the responses which are given by the application browser and the OS of the client. As explained in step I, a fraction of the initial queries is immediately returned as the two mentioned components have queried domain names in their caches. Those responses, in this example from Firefox and Linux, are aggregated to the total response and seen in Figure 3 at the place between "User", "Firefox" and "Linux" by means of green arrows pointing to the Valid, Valid>512B, NXdomain and Truncated responses. Recall that these two client's components are caching only valid query response types (Valid, Valid>512B and Truncated) and the NXdomain response type.

Step III In this step, for each negative response, there will be new reinitiated repeat queries from the client to authoritative NS. Since response streams, coming from authoritative NS, are kept separated, it is possible to determine how many new repeat queries will be reinitiated from the client to the authoritative side. The repeat queries

from the application browser and OS will be reinitiated based on the values given in Table 7. Whether a repeat query is sent again towards authoritative NS depends on the type of the recursive resolver and the type of the response for which a repeat query is reinitiated. Different types of the recursive resolvers have different caching properties, to be seen in Table 8. For example, UNBOUND caches Valid and NXdomain responses. Hence, all repeat queries due to NXdomain responses will be in the cache and they will be answered by UNBOUND.

In Figure 3, for instance, we can observe that for NXdomain responses 0.061 qpt are going back from SLD NS to the client. From Table 7, the Query Multiply Factor for an NXdomain response is two for both Firefox and Linux. Therefore, 0.061 is multiplied by two when passing through Firefox and again by two when going through Linux. Consequently, 0.24 qpt will be gathered at the client's OS to be sent to UNBOUND. As previously explained, for Query Multiply Factor, the obtained value of 2 for NXdomain response type means that one extra repeat query will be resent for each initial query. Therefore, before sending the repeat queries from OS to the recursive resolver, the number of initial NXdomain responses, which is 0.061 qpt, has to be subtracted from 0.24 qpt (sum of initial and repeat queries). Hence, 0.179 repeat queries will be sent from the OS to the recursive resolver. The same procedure will be followed for each response type and all the repeat queries will be gathered at the resolver. However, the resolver, based on its caching property, will send to the authoritative NS only those for which the caching does not play a role. For example, repeat queries due to the NXdomain responses will arrive at the UNBOUND but they will be not sent to the authoritative NS since the UNBOUND deploys negative caching. As a result, 1, 178 and 3240 repeat queries will arrive at the root, TLD and SLD, respectively.

5 Validation of the DNS model

In this section, we validate the DNS reference model by using a new data set captured also at an UNBOUND recursive resolver but in the different environmental setting. The new data set consists of 30.000 DNS packets with duration of 51 seconds. Although we are aware of the fact that validating the model with a single dataset captured at a specific time of day is a rather limited model validation, it still provides a good indication about the capability of the DNS reference model to capture the DNS querying behaviour. To validate the model, we first analyze the data and obtain the input parameters so as to run the simulations and compare the model output to the statistics found in the real-world data. Lastly, we perform the sensitivity check of the model by using coefficient of variance indicator.

Before starting data processing, it should be ensured that all the anomalies are cleaned from the data set. For example, we observed that some misconfigured clients send lots of repeat queries for the same domain names although they receive positive answers on their queries. We determined that the most repeat queries are sent for domains "allmx.tue.nl" and "edgesmtp.uu.nl" and excluded the DNS traffic related to these domains from the dataset. Having obtained a cleaned dataset, the number of the initial

queries can be determined at different point of interest (POI) in the system. The determination of the initial query numbers is crucial since the DNS reference model will be calibrated with the initial queries at the different POIs in the system. To determine the number of initial queries, first a repeat definition has to be formalized.

Repeat definition Considering two queries, the second query will be defined as a repeat query if it has the same domain name, query type and destination level as the first query. Additionally, the time difference between two queries has to be smaller than a certain number δ. For repeats at the recursive resolvers, δ is determined to be 13 seconds while for repeats at the authoritative NS, δ is 3 seconds. These values are determined by analyzing the client and the recursive resolver behaviour. Recall that in the case of a Servfail response Linux client sends seven repeat queries towards the recursive resolver. The time difference between the initial query and the last repeat query is measured to be around 13 seconds. On the other hand, in the case of a Servfail response, the time difference between the initial query and the last repeat query from the UNBOUND towards the authoritative NS is measured to be 3 seconds. Having defined the repeat query notion, initial queries from a given data set of aggregated queries (i.e. initial and repeat queries) are obtained for both the recursive resolver and the root, TLD and SLD NSs. Table 10 shows these values.

Table 10. Initial and repeat queries at different POI in real-world data.

Query Type	Resolver	Root	TLD	SLD
Initial	7131	13	204	3414
Repeat	2360	0	8	723

The last model input parameter to be obtained from the real-world data is the distribution of initial queries' OS types. In fact, this parameter indicates the fraction of the initial queries, generated by a specific client type. OS's fingerprint on each DNS packet is found by using IP TTL values upon which different types can be distinguished. Table 11 shows the result.

Table 11. Distribution of initial queries' OS types, based on only initial queries.

OS	Linux(%)	Windows(%)	MAC(%)
Fraction	60.1	30.2	9.7

The input parameter values obtained from the real-world data serve as a starting point for the validation process. As the data is captured at one single UNBOUND recursive resolver, the "Number of simultaneously active resolvers" will be 1. Additionally, as data is captured at one UNBOUND recursive resolver, the model will be calibrated at this POI with the number of initial queries, instead of the number of initial queries at the

users, i.e. before the client's OS and application browser. As a consequence, the "Number of simultaneously active DNS clients" is determined by trial and error method: we found that with a query rate of 1 qpt, 9700 users generate 7131 initial queries at the recursive resolver. This number of 9700 users concerns thus "Number of simultaneously active DNS clients". Furthermore, we will ignore the effect of secondary NSs by assuming that there will be no secondary NSs and assume that the fraction of IPv6 is with respect to all clients is 0. Now that we have obtained the input parameters for the model, we can run the model. Recall that we are interested in the number of initial and repeat queries at different POI in the model. The simulation is repeated (30000 times) for each of the three different combinations of the client's OS and application browser: Windows-IE, MAC-Safari and Linux-Firefox. The outcomes of the simulations are histograms showing the distribution of the initial and the repeat queries at each POI. An example of a histogram showing the repeat query distributions at the recursive resolver can be seen in Figure 4. The most probable values from the distribution of initial and repeat queries at each POI are given in Table 12. Recall that the outcome values are weighted by the distributions of client OS as found in real-data set (given in Table 11).

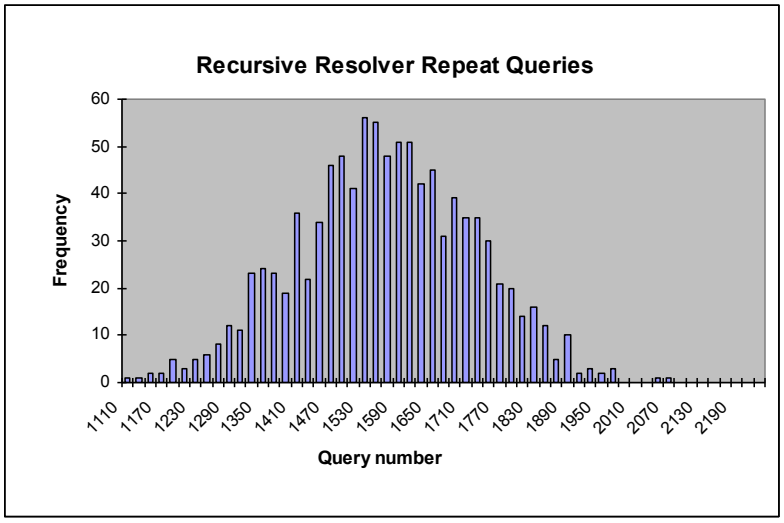

Fig. 4. Repeat query distribution at recursive resolver.

Table 12. Initial and repeat queries at different POI in reference model.

Query Type	Resolver	Root	TLD	SLD
Initial	7204	8	350	3030
Repeat	1530	0	5	350

In the following step we compare these results with the results obtained from the real-world data. We compare both, the fraction of total queries as well as the repeat-

initial query ratio at POIs. Query ratio at POIs indicates the ratio between the total number of queries (i.e. initial and repeat queries) at POI and the total number of the queries in the entire system. For instance, in the DNS reference model, the fraction of total queries at the recursive resolver can be found as the ratio between the number of queries at resolver and the number of queries in the entire system, i.e.: (7204 + 1530) / (7204 + 1530 + 8 + 350 + 5 + 3030 + 350) = 70%.

Table 13 shows the fractions of total queries at POIs in the real-world data and the DNS reference model. It can be seen that DNS reference model predicts well the query ratio over the POIs in the system. Small errors are most probably due to the stochastic nature of the system variables.

Table 13. Query ratio at POIs, obtained from real-world data and DNS reference model.

Query ratio	Resolver(%)	Root(%)	TLD(%)	SLD(%)
Real world data	68.5	0.1	1.5	29.9
DNS reference model	70	0.1	2.8	27.1

The second test point of the DNS reference model concerns the repeat-initial query ratio at POIs. This ratio indicates the fraction of the repeat queries with respect to the total number of queries at a particular POI. For instance, in the DNS reference model, the faction of repeat-initial queries at the resolver can be found as follows: 1530 / (1530 + 7204) = 17.5%.

Table 14. Initial-repeat query ratio at different POI obtained from real-world data and reference model.

Initial-repeat ratio	Resolver(%)	TLD(%)	SLD(%)
Real world data	24.8	3.8	17.5
DNS reference model	17.5	1.4	10.4

Table 14 shows the repeat-initial ratio at POIs in the real-world data and in the DNS reference model. At recursive resolver a difference of 7,3% is observed. We expect this error occurs due to effect of IPv6 clients. IPv6 clients send two queries in pair for address resolution: A and AAAA query. When they receive a negative response from the recursive resolver, then they resend repeat queries also in pair meaning that they send more repeat queries than the Query Multiply Factor values given in Table 7. The error at the authoritative NS might be due to the effect of secondary NSs. For example, we observed that in case of Servfail response, each additional NS causes five extra queries: at first, two repeat queries are sent to primary NS. If it again receives a Servfail response, then it queries the secondary NS. If secondary NS also returns Servfail responses, then UNBOUND will again query the primary NS. This querying pattern continues until each NS is queried five times. In this way, a Servfail response causes in total ten queries

instead of five as in the case of only one NS.

As a last step we answered the question of how the variation in the system variables affects the outcome of the DNS reference model. In other words, how sensitive is the model output with respect to the stochastic system variables. This question is answered by using coefficient of variance (CoV) metric. CoV is a statistical measure of dispersion around the mean in a probability distribution. We found that CoVs at the output are smaller than 1, meaning that the dispersion in the distributions is small and all the values are concentrated around the mean. Then, CoVs of system variables and output values are comparable, which implies that the DNS reference model does not amplify the uncertainty due to the random system variables.

6 Case study

In order to illustrate the value of our reference model we briefly indicate how the model can be applied to a specific case. As stated earlier, the DNS is facing several major changes, among which the introduction of DNSSEC. Potentially DNSSEC introduces the risk of an increase in Servfail responses due to validation errors or other factors. For example, any error made in DNSSEC signatures at the authoritative side will result in a validation error at the recursive NS. And by default the recursive NS will feed back the validation error to the client side as a Servfail response. We evaluate the impact of this potential increase in Servfail responses and quantify the increase of DNS traffic towards the recursive resolver, the TLD and SLD NSs as a function of the relative increase of Servfail responses.

First, we investigate the impact of an increase of responses (in %) from the TLD that trigger the resolver to return a Servfail response to the client. In this case we keep the fraction of Servfail responses at the root and SLD constant. In particular, we vary the value of our model input parameter Response Distribution for Servfail at the TLD (see Table 6a) and observe the values predicted by our model for the DNS traffic increase from clients towards the resolver, and from the resolver towards the TLD NS. For the other model input parameters we use the default parameter values presented in the tables in Section 3.2. In this case we focussed on UNBOUND resolver behavior. For the DNS query volumes and the distribution of traffic per OS we used the real-world data as described in the previous section. The results are presented in Figure 5. The x-axies represent the percentage of increase in Servfail response at the authoritative NS (TLD and SLD). For example, 2% implies that additional 2% of Servfails are added to the percentage of Servfail responses specified in Table 6a. On the y-axis the predicted, relative increase of DNS traffic towards the recursive resolver, respectively towards the authoritative NS is plotted. The figure shows a more or less linear relation between the Servfail increase and the DNS traffic. However, the DNS traffic increase towards the TLD is much stronger, than from the client towards the resolver. More detailed analysis of the model results (not presented here) can explain these results. Not explicitly shown in Figure 5 is the observation that additional Servfail responses triggered by responses

from the TLD does not increase DNS traffic between any other POIs.

Similarly, we investigate the traffic increase towards the recursive resolver or towards the authoritative NS as a consequence of the Servfail increase at SLD. This is given in Figure 6. We observe that the DNS traffic towards the authoritative NS increases significantly by the increase of Servfail responses. For example, in case of 10% of additional Servfail response at SLD, the DNS traffic would increase for almost 35% towards SLD NS. We further observe that the increase in Servfail at SLD NS has larger impact on the DNS traffic than the increase in Servfail at TLD NS.

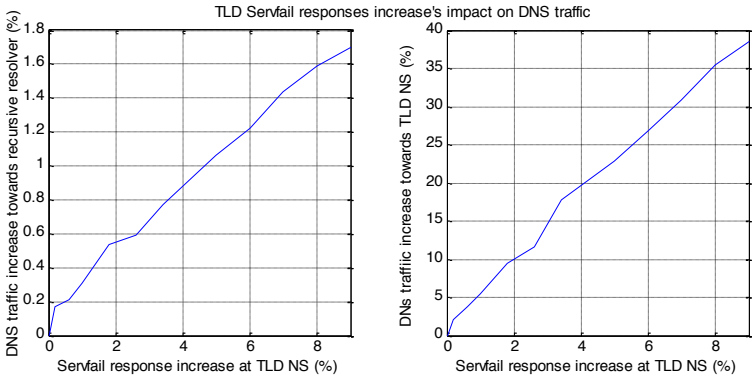

Fig. 5. Impact of increase in Servfail responses at TLD NS on DNS traffic.

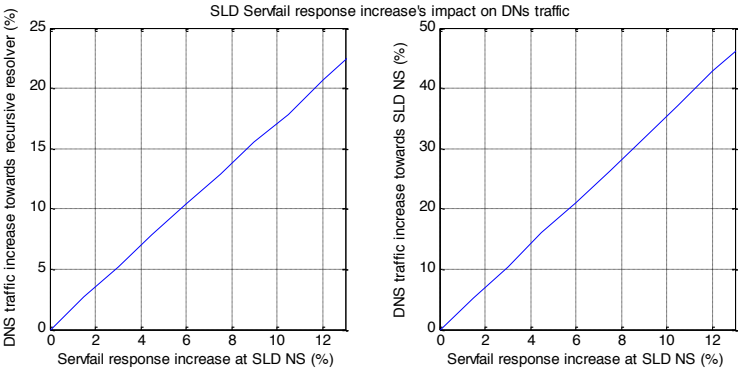

Fig. 6. Impact of increase in Servfail responses at SLD NS on DNS traffic.

7 Conclusion and further research

In this paper we have introduced a global DNS reference model with the aim to assess the scalability of the DNS system in case of certain what-if scenarios. The DNS reference model consists of components that model the DNS behaviour of client OS and application browser, the recursive resolver and the authoritative name servers. The values of the parameters for these components are obtained from real-world data (captured on the operational SURFnet DNS infrastructure), results from experiments with DNS clients, and complemented with results from data analyses published by other researchers. We validated the model by comparing the DNS model output to the statistics found in the real-world data and conclude that the model predictions are rather accurate. In addition, we discussed the shortcomings and possible extensions of the model and how additional analysis based on real-world data can be done to further increase the accuracy of three model variables that we introduced:

- Cache Hit Ratio, used to characterize the caching property of the client and the recursive resolver,
- Response Distribution at Authoritative Name servers, used to characterize the response behaviour of the authoritative NSs, i.e. the root, TLD and SLD,
- Query Multiply Factor, used to characterise the query behaviour, in reaction to negative query response, of the client and the recursive resolver.

For the first two system variables, the probabilistic distributions are found by analyzing real-world data. We have shown that these system variables can be approximated by a Gaussian distribution. For the Query Multiply Factor, we relied on the lab experiments but also on results published in [2]. Having determined the probabilistic distributions for the system variables we have accounted for the stochastic behaviour of the DNS. For the validation of the model, we relied on the approach of Monte Carlo simulation. We have compared the results from the real-world data and the results from the DNS reference model, and shown that the DNS reference model captures the DNS behaviour properly. We have tested the model performance based on the two test points: the fraction of total queries and the initial-repeat queries ratio at various POIs in the system. We have observed a negligible error at the first test point while the error in the second test point was relatively small. We attributed the error in the second test point to the effects of IPv6 enabled clients and the secondary NS, possibly present in the real-world data set. After validating the model, we have used CoV metric to show that the output of the DNS reference model output is not sensitive to the variations in the system variables. Finally, we demonstrated the applicability of the model by evaluating the impact of a potential increase in Servfail responses.

For future work we propose to validate the model with data from different DNS environmental settings e.g. a different UNBOUND recursive resolver. Additionally, although we used a data set consisting of 300.000 DNS packets, analysis of a larger data set will be needed in order to determine the response distribution at the root. Remarking that the system variable Response Distribution at Authoritative Name servers has a crucial importance for the initial-repeat query ratio, we recommend to extend the data

analysis with more and larger data sets to obtain representative numbers for all the system variables. Furthermore, extending the model with the effects of secondary NS and IPv6 enabled hosts, belongs to the possible ways this work could be extended. We expect that modelling these factors would reduce the error and leads to a more accurate DNS reference model prediction of DNS query behaviour.

Previous publications have mentioned that a significant part of the DNS traffic in the real-world data sets is generated by a very limited number of clients. In our analysis we confirmed this effect, that may be resulting from e.g. misconfigured name servers or client. Detecting this kind of behaviour requires some data engineering in order to obtain clear interpretation of the DNS data. More advanced algorithms for detecting these DNS anomalies would contribute significantly to better understand DNS behaviour and would help improving DNS research results.

References

[1] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Comparing dns resolvers in the wild. In M. Allman, editor, *Proc. Internet Measurement Conference*, pages 15–21. ACM, 2010.

[2] B.Gijsen. Dns(sec) client analysis, 2011. DNS OARC Workshop, San Fransisco.

[3] C. Brandhorst and A. Pras. Dns: a statistical analysis of name server traffic at local network-to-internet connections. In *Proc.IFIP International Workshop on Networked Applications*, 2006.

[4] S. Castro, D. Wessels, M. Fomenkov, and K. C. Claffy. A day at the root of the internet. *Computer Communication Review*, 38(5):41–46, 2008.

[5] B. M. Duska, D. Marwood, and M. J. Feeley. The measured access characteristics of world-wide-web client proxy caches. In *Proc. USENIX Symposium on Internet Technologies and Systems*, 1997.

[6] J. Jung, E. Sit, H. Balakrishnan, and R. Morris. Dns performance and the effectiveness of caching. *IEEE/ACM Trans. Netw.*, 10(5):589–603, 2002.

[7] P. Mockapetris and K. J. Dunlap. Development of the domain name system. *SIGCOMM Comput. Commun. Rev.*, 18(4):123–133, 1988.

[8] J. Neumann. *The Annals of Mathematical Statistics*, (4):367.

[9] R.Arends. Protocol modifications for the dns security extensions, 2005. RFC 4035.

[10] e. a. S. Castro. Understanding and preparing for dns resolution. In *Lecture Notes in Computer Science*, pages 1–16, 2010.

[11] S. Shapiro and M. Wilk. An analysis of variance test for normality (complete samples). *Biometrika.*, 52(3):591–611, 1956.

[12] K. T.Toyono, K.Ishibashi. Clear and present increase in number of dns aaaa queries, 2006. DNS OARC Workshop, San Fransisco.

[13] D. Wessels. Is your caching resolver polluting the internet? In *Proc. ACM SIGCOMM NetTs*, pages 271–276, 2004.

[14] D. Wessels and M. Fomenkov. Wow, thatâĂŹs a lot of packets. In *Passive and Active Measurement Workshop (PAM)*, 2003.

[15] D. Wessels, M. Fomenkov, N. Brownlee, and K. C. Claffy. Measurements and laboratory simulations of the upper dns hierarchy. In C. Barakat and I. Pratt, editors, *PAM*, volume 3015 of *Lecture Notes in Computer Science*, pages 147–157. Springer, 2004.

[16] A. Wolman, G. M. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. M. Levy. On the scale and performance of cooperative web proxy caching. In *Proc. of the 17th ACM Symposium on Operating System Principles (SOSP 1999)*, pages 16–31, 1999.

AUTHOR BIOGRAPHIES

Yakup Koç received his B.Sc and M.Sc in Electrical Engineering at the Delft University of Technology (TU Delft), Delft, the Netherlands, in 2009 and 2011, respectively. He is currently pursuing the Ph. D. degree in the Section of System Engineering, Department of Multi-Actor Systems, TU Delft. His Ph.D. work focuses on designing smart energy systems, so called SmartGrids. His research interests include network reliability and robustness of complex networks.

Almerima Jamakovic graduated and received a M.Sc. degree in Electrical Engineering at the faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS) at Delft University of Technology (TU Delft), the Netherlands, in 2004. After finishing her study Almerima joined the Network Architecture and Services (NAS) Group of TU Delft to work towards a PhD degree, which she obtained in the third quarter of 2008. During her PhD she performed research in the field of complex networks, focusing on the quantitative characterization of such âĂIJcomplexâĂİ structures and its application to robustness analysis. From June 2008 till present day Almerima is employed at TNO, the Netherlands institute for applied research. As a member of the department of Performance of Networks and Systems she works on the topic of quantitative analysis of networks and systems in a wide range of performance- and optimization-related projects in the field of Information systems and Telecom networks.

Bart Gijsen After receiving his Msc. degree in both computing science and mathematics Bart joined KPN Research in 1996 as a researcher in the field of performance analysis of Internet technology based information and communication systems. From 2003 till present day Bart is employed at TNO as a senior innovator in the expertise group Performance of Networks and Systems. One of his focus areas is quantitative modelling and impact prediction of Internet security and stability. Since 2010 Bart is also part-time director of the Dutch ICT Innovation Platform "Critical ICT infrastructures".