DNSSEC meets real world: dealing with unreachability caused by fragmentation

Gijs van den Broek*†, Roland van Rijswijk-Deij†, Anna Sperotto*, Aiko Pras*
*University of Twente
†SURFnet bv

***Abstract*—The Domain Name System (DNS) provides a critical service on the Internet: translating host names into IP addresses. Traditional DNS does not provide guarantees about authenticity and origin integrity. DNSSEC, an extension to DNS, improves this by using cryptographic signatures, at the expense of larger response messages. Some of these larger response messages experience fragmentation, and may, as a result of that, be blocked by firewalls. As a consequence, resolvers behind such firewalls will no longer receive complete responses from name servers, leading to certain Internet zones becoming unreachable because no translation into IP addresses can be performed.**

Our research shows that despite ongoing efforts to educate firewall and resolver administrators, as much as 10% of all resolvers suffer from fragmentation-related connectivity issues. Given that some major Internet companies were reluctant to adopt even a technology like IPv6 if it meant that a small percentage of their users would have connectivity issues, it is clear that we cannot rely on resolver/firewall operators alone to tackle this issue.

The contribution of this paper is that it a) quantifies the severity of these DNSSEC deployment problems, based on extensive measurements at a major National Research and Education Network (NREN) and backed up by validation of these findings at an independent second location, b) proposes two potential solutions at the DNS authoritative name server side, and c) validates both solutions, again based on extensive measurements on the operational network of this major NREN. The paper concludes with a recommendation favoring our first solution. The first solution is relatively simple to implement and gives DNS zone operators control over this problem without having to rely on all resolver operators solving the issue.

I. Introduction

THE Domain Name System (DNS) provides a critical service on the Internet. DNS is responsible for translating easily recognizable host names into IP addresses.

DNS data is contained in zones, like surfnet.nl, in which *resource records* are specified, such as www.surfnet.nl. An *authoritative name server* for a zone responds to queries from resolvers for these records. *Resolvers* are responsible for querying authoritative name servers on behalf of end users. The majority of DNS messages are transmitted using the UDP protocol, although TCP can be used as a fallback.

Traditional DNS has no mechanism to ensure data origin integrity and authenticity. This makes it possible to alter DNS traffic. Multiple vulnerabilities have been identified since the 1990's [1], [2], RFC 3833. Such incidents have led to proposals to secure DNS.

In the secure version of DNS (DNSSEC, RFC 4033-4035), resource records are digitally signed to provide data origin integrity and authenticity. Resolvers can verify the authenticity of data received from an authoritative name server based on these digital signatures.

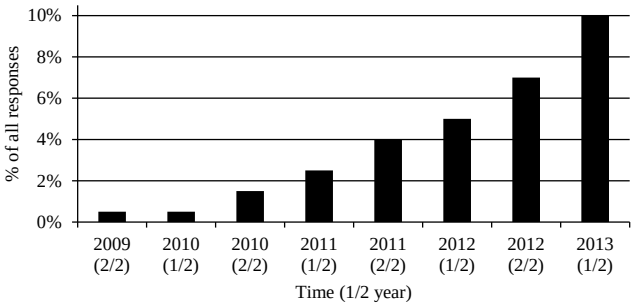

Fig. 1. DNSSEC response messages at SURFnet resolvers (by approximation).

DNSSEC is implemented in many DNS software distributions and its use is increasing, as shown in Fig. 1. Since the DNS root was signed in 2010, the percentage of responses that could be validated using DNSSEC on resolvers of SURFnet[1] has grown to about 10% by mid-2013. More information on the current deployment state and other DNSSEC related resources can be found at the Internet Society[2].

Traditional DNS specifies a maximum UDP message size of 512 bytes. That is no longer sufficient for most DNSSEC responses, because of the added burden of having to include digital signatures in each response. Cowperthwaite and Somayaji [12] measured an increase of 11 times in response size. An extension mechanism, called EDNS0 (RFC 2671) allows for larger messages. In some cases, DNSSEC responses are so large that they are fragmented into multiple IP fragments. Fragmentation occurs when the message size exceeds the maximum amount of data (*MTU*[3]) that can be transported in one packet. This poses a problem, however, since fragments are often blocked by firewalls to prevent some types of cyber attacks (but at the expense of VPN interoperability, see [3] for a discussion of the tradeoffs).

Firewalls that block fragments introduce problems as depicted in Fig. 2. The resolver sends a query (1) to an authoritative name server. The name server sends back a large response that exceeds the MTU and is fragmented. The fragments arrive

[1]The Dutch National Research and Education Network (NREN).
[2]http://www.internetsociety.org/deploy360/dnssec
[3]Maximum Transmission Unit

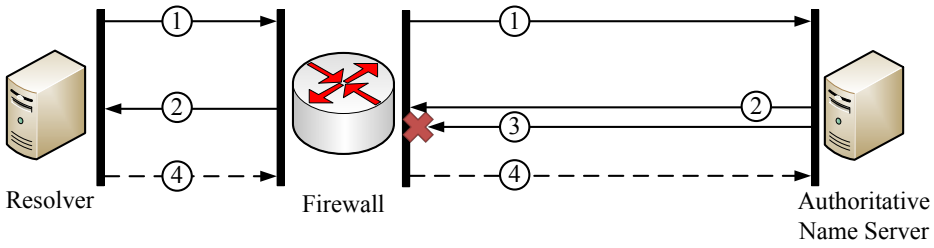

Fig. 2. DNS resolver incapable of receiving fragmented response messages because of firewall restrictions.

at a fragment blocking firewall, where the first fragment (2) is allowed to pass, but remaining fragments (3) are blocked. The resolver is now unable to reassemble the fragments to get the original response. After some time the resolver will signal that it did not receive all fragments by sending back an ICMP Fragment Reassembly Time Exceeded (FRTE) message (4). The end effect of fragment blocking firewalls is that complete zones, even the DNS root [4], become unreachable for resolvers behind such firewalls and consequently to every client using these resolvers [5]. In the remainder of this paper we will refer to such resolvers as *problematic resolvers*.

One might consider using signature algorithms that result in smaller signatures, or propose the development of a mechanism for cipher-suite negotiation [13]. This may result in smaller responses and hence less fragmentation. However, it is questionable if this would solve the problem.

The obvious approach for now is to solve this problem at the problematic resolver's side. Educational materials like[4] explain this issue in detail. Besides configuring firewalls correctly, a solution could be that problematic resolvers switch to TCP, instead of UDP, in order to avoid fragmentation. TCP, however, is expensive compared to UDP in terms of response time and system resource consumption. Another solution would be to detect problems with fragmentation at the problematic resolver side, for example by inspecting the maximum MTU from the authoritative name server, and alter future requests to avoid fragmentation [6], [7]. To a certain extent, modern name server software will use this strategy.

The problem with relying on resolver operators, however, to deal with this issue is two-fold. Firstly, it gives very little control to DNS zone operators whose zones may become unreachable for end users behind problematic resolvers. This may hinder wide-scale deployment of DNSSEC, as parallels with IPv6 show, where large companies were reluctant to deploy IPv6 because it potentially meant that a small percentage of their users would experience connectivity issues. Secondly, resolver operators may be unaware that they suffer from this problem. The majority of DNS resolver implementations request DNSSEC data regardless of whether or not that data is validated and have default settings that inevitably lead to fragmented responses to some of their queries.

Potential problems associated with deploying DNSSEC have already been described in literature [8]; the novel contribution of this paper is that it:

a) quantifies the severity of this problem for operators, based on extensive measurements on the live infrastructure at a major NREN;
b) proposes two potential solutions to avoid fragmentation at the authoritative name server side;
c) validates both solutions, again based on extensive measurements on the operational network of this major NREN.

The paper is organized as follows. Section II analyzes the extent of the problem, based on real-world measurements, and validates findings at a second location. After the severity of the problem is known, two solutions for an authoritative name server to avoid response fragmentation are proposed. Section III discusses a solution that avoids response fragmentation in general, for all resolvers. Section IV discusses a more sophisticated solution, which detects problematic resolvers and modifies responses for those resolvers only. Section V compares both solutions, and Section VI contains our conclusions.

II. Extent of the Problem

This section presents real-world observations of the problem with fragmented responses. These observations result from network traces recorded on an authoritative name server of SURFnet. This server is authoritative for ±4000 zones, including ±300 DNSSEC-signed zones. It receives ±500 queries per second on average.

The traces recorded in early 2012 over a period of 6 hours contain about 8.5 million DNS(SEC) messages. In these traces we identified 231,391 unique resolvers (based on IP addresses). ±75% of all queries used EDNS0, indicating that the querying resolver is capable of receiving responses with EDNS0. The average response size (UDP and IP headers not included) is 840 bytes. Note that this is higher than the limit for traditional DNS messages (512 bytes). The cumulative distribution of the response size is shown in Fig. 3. About 36% of all responses are fragmented at the authoritative name server (with an MTU of 1232 bytes). Moreover, 57% of all resolvers received a fragmented response at some time during the measurements.

In the following we identify and discuss 5 behavioral patterns indicating that we are dealing with a problematic resolver. We note that only the first pattern is a definite indicator that the resolver has problems with fragments. Because this pattern may be affected by the firewall that causes the problems for the resolver we have also looked at 4 heuristic indicators for problematic resolvers. The patterns are

[4]http://www.surfnet.nl/Documents/rapport_Deploying_DNSSEC_v20.pdf

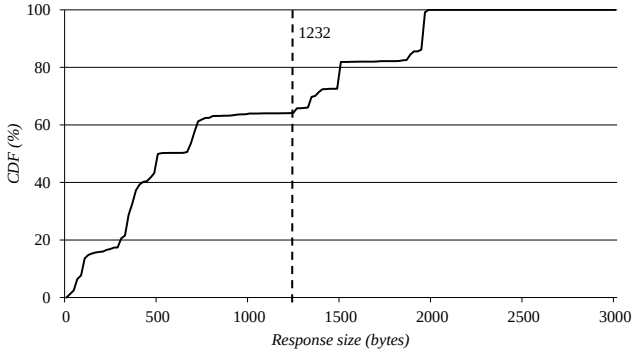

Fig. 3. CDF of DNS response size in traces.

Problematic Resolver Characteristic	Occurrence
CASE 1: Send ICMP Fragment Reassembly Time Exceeded	1.3%
CASE 2: Fallback to traditional DNS	2.4%
CASE 3: Reduce advertised max. response size in retries	3.5%
CASE 4: TCP fallback w/o truncated UDP response	<0.1%
CASE 5: Retries for large responses (>512 bytes)	9.7%

TABLE I
PROBLEMATIC RESOLVER CHARACTERISTICS.

summarized in Tab. I together with their occurrence in the recorded traces.

CASE 1 The simplest pattern to detect is receiving ICMP Fragment Reassembly Time Exceeded (FRTE) messages at the authoritative name server, as discussed in Sec. I. These ICMP messages confirm that the resolver experiences problems with fragmented responses. 1.3% of the resolvers seen in the traces exhibit this behavior. Note that, as mentioned above, the firewall affecting a problematic resolver may also block ICMP messages sent back to an authoritative name server for security reasons, as discussed in [3].

CASE 2 Some DNS software will, when not receiving a response to a query that uses EDNS0, retry a query using traditional DNS (probably under the assumption that the authoritative name server being queried does not support EDNS0). A resolver cannot distinguish not receiving a response because of blocked fragments from not receiving an answer because EDNS0 is not supported. We therefore expect resolvers that use this strategy to apply it in the problem under consideration. In the traces, we detected this behavior in 2.4% of all resolvers.

CASE 3 When using EDNS0, resolvers advertise the maximum response size they support in all queries. These advertisements are used by authoritative name servers to ensure that responses do not exceed this size limit. Consequently, if the advertised maximum response size exceeds the MTU of the link between the resolver and the authoritative name server, this will lead to fragmentation. Our traces show that 69% of all queries contain a maximum response size advertisement set to the default[5] value of 4096 bytes. Considering that the MTU is mostly $\leq$1500 bytes, advertising a maximum response size of 4096 bytes likely results in fragmented responses.

Some resolver software dynamically adapts the advertised maximum response size if it fails to receive responses to queries. Thus, if we detect this behavior, this is indicative of a problematic resolver. Our traces show 3.5% of all resolvers applying this strategy.

Interestingly, our traces also show that 1.8% of all queries used EDNS0 with a maximum response size advertisement of just 512 bytes and that 2% of all queries have a maximum response size advertisement between 1280 and 1472 bytes. This range is likely chosen by resolver operators such that most responses do not get fragmented, since it is below the MTU of Ethernet[6].

CASE 4 If an answer does not fit in the maximum response size advertised by a resolver, the authoritative name server will send back a *truncated response*, indicating to the resolver that the response is incomplete. In order to avoid response size restrictions for UDP, DNS then allows the resolver to retry the query using TCP. Some 0.1% of all resolvers, however, use TCP after a UDP response that is not truncated. We have strong suspicions that these are problematic resolvers attempting to avoid fragmentation issues.

CASE 5 Finally, there are resolvers (9.7% of all resolvers) for which we only detect series of retries that always result in responses larger than 512 bytes. Since we only see retries, it becomes difficult to determine which of these resolvers are actually problematic resolvers. We will discuss this problem in more detail in Sec. IV.

Weaver et al. [7] state that up to 9% of all Internet hosts may have problems receiving fragmented UDP messages. Our measurements confirm this; traces show that most resolvers receive fragmented responses and we identified 5 different resolver behaviors that can indicate problems receiving fragmented responses. As much as 10.5% of all resolvers showed one or more of these behaviors. We verified our results using traces from an authoritative name server at the University of Pennsylvania, which showed the same distribution of behavioral patterns indicative of problematic resolvers.

III. AVOIDING RESPONSE FRAGMENTATION IN GENERAL

This section presents a simple solution to the problem by attempting to avoid most response fragmentation in general. This involves changes to authoritative name servers, as shown in Fig. 4.

As mentioned in Section II (CASE 3), resolvers advertise a maximum response size in EDNS0 queries. Problematic resolvers advertise maximum response sizes that are too high (as they cannot receive fragmented responses). Generally speaking, the response size on an authoritative name server is only limited by the size advertised by querying resolvers and not by configuration settings on the authoritative name server itself.

The solution we propose here is to restrict the maximum response size in the configuration of the authoritative name server, such that (most) response fragmentation is avoided.

[5]That is: the default for the most popular name server software.

[6]78% of all paths between any 2 nodes on the Internet have the Ethernet MTU (1500 bytes) and 96% are $\leq$1500 bytes [7]. The maximum size for DNS responses (without IP and UDP headers) that avoids fragmentation, given an MTU of 1500 bytes, is 1472 bytes for IPv4 and 1452 bytes for IPv6.

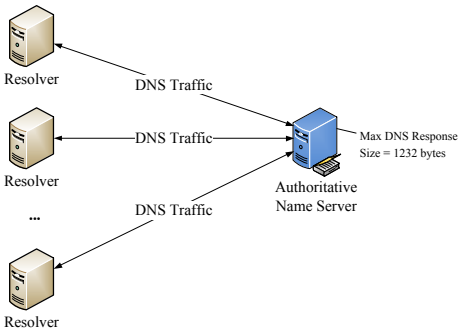

Fig. 4. Authoritative name server with limited response size.

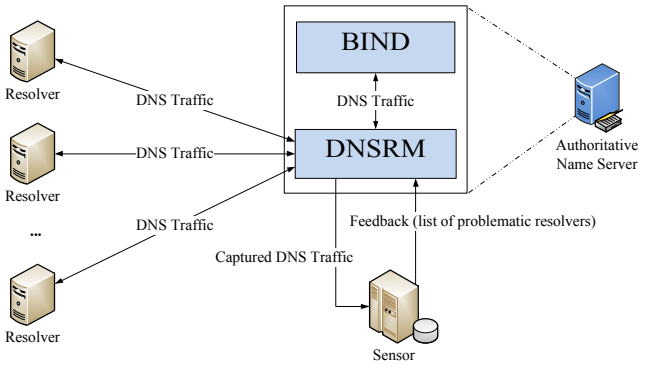

Fig. 5. Authoritative name server with a host-proxy (DNSRM) and an external sensor.

When responding to queries the authoritative name server will then use the minimum of the configured response size and that advertised in the query.

The response size limit for the authoritative name server should be as high as possible, while avoiding most fragmentation. If the limit is chosen too low, this may result in undesirably high levels of truncated responses, resulting in retries over TCP. Considering that the MTU is usually 1500 bytes[6], the limit should be below that value. We propose to use a limit of 1232 bytes, based on the minimum MTU for IPv6 of 1280 bytes (RFC 2460). This ensures that any response sent over IPv6 by the authoritative name server is not fragmented. The minimum IPv4 MTU is 68 bytes (RFC 791), which is too small to be used here, but since 1232 bytes is well below the expected MTU of 1500 bytes we expect this limit to avoid fragmentation for IPv4 as well. Note that other limits are possible and that 1232 bytes may be too small for some zones [9]. This limit can be set in most DNS software using a single parameter.

The response size limit will help most problematic resolvers, even if just one of the authoritative name servers per zone returns responses of limited size. This is because resolvers query all authoritative name servers for a zone in case they do not receive a response.

The solution presented in this section avoids most response fragmentation thus helping problematic resolvers. Although this solution is very simple, care must be taken that a proper size limit is chosen.

IV. Selectively avoiding response fragmentation

This section presents a solution that avoids response fragmentation by limiting the response size for problematic resolvers only. Fig. 5 shows the setup of this solution. This solution is based on DNSRM[7] (DNS Router/Modifier), a tool we developed specifically for this purpose. DNSRM operates as a host-proxy on an authoritative name server and acts on information supplied by a separate sensor tool that detects problematic resolvers.

A. Modifying queries using DNSRM

The purpose of DNSRM is to allow an authoritative name server to differentiate in response size, depending on the querying resolver.

DNSRM achieves this goal by acting as a host-proxy between the local authoritative name server process (e.g. BIND) and the outside world. It is transparent for resolvers in the outside world. DNSRM also forwards a copy of all DNS traffic to a sensor, which is tasked with detecting problematic resolvers and their maximum response size. DNSRM receives updates to a list of IP addresses of detected problematic resolvers from the sensor.

Finally, DNSRM modifies queries from problematic resolvers detected by the sensor, before it passes them on to the local authoritative name server process. DNSRM overwrites the advertised maximum response size in a query. It does so when a query from a problematic resolver has a maximum response size advertisement larger than the maximum response size for that problematic resolver detected by the sensor. DNSRM does not modify responses.

B. Detecting problematic resolvers using a sensor

A sensor detects problematic resolvers based on DNS traffic forwarded by DNSRM. The sensor analyses the traffic per resolver, based on the 5 problematic resolver behaviors discussed in Sec. II. The sensor uses a system of thresholds when detecting a problematic resolver, in order to avoid false positives. For instance, network issues causing packet drops may result in resolver behavior that could incorrectly suggest a problematic resolver. The actual values of the thresholds depend on the network characteristics of the authoritative name server running DNSRM[8]. The sensor works as follows for each problematic resolver behavior pattern:

- Case 1 The sensor detects ICMP Fragment Reassembly Time Exceeded messages; resolvers sending ICMP FRTE messages are marked as problematic.
- Case 2 The sensor analyses retries of queries. If a resolver changes from EDNS0 to traditional DNS for the same query in a retry, it is marked as problematic.
- Case 3 A resolver that reduces its advertised maximum response size in retries, will also be marked as a problematic resolver.

[7]https://svn.surfnet.nl/svn/dnstools/

[8]We used an expected packet loss of 4% for responses sent to a resolver, based on Wang et al. [10].

CASE 4 The sensor detects the use of TCP in retries. As mentioned in Sec. II this is only allowed when a truncated UDP response precedes the fallback to TCP (RFC 1123). Resolvers that use TCP without a preceding truncated UDP response are marked as problematic.

CASE 5 There are a number of resolvers that do not show any of the behaviors described above, but do send sequences of similar queries that appear to be retries. Since we only see these retries, it is difficult to determine if these resolvers are problematic resolvers. Some will be *non-caching resolvers*, that do not store a response and therefore frequently send the same query. Non-caching resolvers are not necessarily problematic resolvers. If the number of 'retries' from a resolver after responses $\leq$512 bytes exceeds a pre-set threshold we consider them to be non-caching. Here, we assume that fragmentation issues do not affect responses $\leq$512 bytes. If we see retries from a resolver that is not considered a non-caching resolver, then it will be marked as problematic if the number of retries after responses $>$512 bytes exceeds a pre-set threshold.

The sensor not only detects problematic resolvers but also the maximum response size it supports. This is done by marking the lowest response size for which problematic resolver behavior still occurs. The maximum response size for a problematic resolver is reduced stepwise, starting at the MTU of the medium (e.g. Ethernet[6]) being used by the authoritative name server running DNSRM. The process continues for as long as problematic resolver behavior is detected[9] and will help a problematic resolver quicker with a larger step size.

The solution presented in this section avoids response fragmentation, but only for problematic resolvers detected by a sensor. The sensor detects a maximum response size per problematic resolver that avoids response fragmentation. The size of a response from an authoritative name server is reduced by overwriting the advertised maximum response size in queries using DNSRM. This solution leaves the settings of a problematic resolver unchanged.

V. EVALUATION

This section evaluates the two solutions that were presented in Sections III and IV. First, the characteristics of the solutions are compared. Next, the results of real-world testing are discussed.

A. Comparing characteristics

The first solution avoids response fragmentation by limiting the size of all responses sent by an authoritative name server by altering a single parameter on the server. The advantage of this approach is that it is trivial to implement. The biggest disadvantage, however, is that it is a blanket approach that affects all resolvers. As a consequence, well-behaved resolvers ($\pm$90% of all resolvers) may suffer performance penalties because they do not receive optimal answers to their queries. Additionally, problematic resolvers are not given an incentive to alter their configuration. Thus one could claim that this approach rewards bad behavior. Finally, this solution does not address all issues of problematic resolvers (e.g. blocking DNS messages $\geq$512 bytes).

The second solution only avoids response fragmentation for problematic resolvers. Its main advantage is that it is adaptive; it only limits the response size for resolvers that are suspected to be problematic. Also, it limits the response size dynamically. Thus, unlike the first solution, it also helps problematic resolvers that suffer additional constraints on the path between themselves and the authoritative name server. A final advantage of this solution is that it only assists problematic resolvers for a limited amount of time, giving administrators of these hosts a bigger incentive to improve their behavior. What makes this solution less attractive, especially in a production environment, is its complexity. It requires two additional components, a host-proxy directly in the path to the authoritative name server and a sensor application that is CPU intensive. If, however, the decision is made to deploy the second solution, then it is imperative that the system(s) hosting the DNSRM and the sensor application be properly sized and engineered such that the second solution deployment is not a bottleneck, i.e., that the effective throughput of the name server is not decreased.

B. Real-world testing

Both solutions were tested for 6.5 hours during office hours on the same authoritative name server of SURFnet that was used to measure the extent of the problem (Sec. II). Network traces were recorded during the tests for later analysis.

1) Avoiding response fragmentation in general: To test the first solution we reduced the response size from a typical 4096 bytes to 1232 bytes. Traces show that no responses were fragmented at the authoritative name server. Consequently, we saw no ICMP FRTE messages, which indicates that this solution effectively helped the hosts that were previously sending these error messages. We saw a slight increase in the number of truncated UDP responses, but the increase was not statistically significant.

2) Selectively avoiding response fragmentation: The second solution was tested with DNSRM on the same server and a separate sensor. Traces show that fragmentation was down about 50% compared to normal operations; note that fragmented responses still occur for resolvers that are not marked as problematic resolvers.

The number of ICMP FRTE messages was 18% of normal. This number is not zero, because the sensor first needs to detect a problematic resolver. Only after detection will the resolver be helped and will these ICMP messages disappear.

The number of truncated UDP responses almost doubled in this experiment. This suggests that the detected maximum response size for some problematic resolvers may have been too small. Analysis of the detected sizes confirms this. Approximately 18% of all problematic resolvers were assigned a maximum response size of 512 bytes, likely resulting in

[9]It will never go below 512 bytes since that is the lower limit for EDNS0.

truncated UDP responses. Most of these problematic resolvers may actually be non-caching resolvers, that have no problem receiving fragmented responses. Our mechanism for detecting non-caching resolvers may therefore need improving.

Our evaluation suggests that the first solution we proposed is preferred in production environments due to its simplicity and effectiveness. If just one of the authoritative name servers for a zone limits its response size to a value that avoids most response fragmentation, then problematic resolvers may already be able to receive responses. Depending on the zone, a proper response size limit needs to be chosen, in order to avoid an unacceptable increase in truncated UDP responses and TCP fallbacks [9]. We believe that our second solution is not without merit, however. Implementing it yielded a useful categorization of the problems encountered by resolvers in the form of 5 problem cases. Our tests also showed that despite its complexity the solution performs well in a production environment.

VI. Conclusions and Future Work

The introduction of DNSSEC creates new problems, such as Internet zones becoming unreachable due to poorly configured firewalls that block fragmented UDP packets. Although these problems can easily be solved by changing the firewall rules or by changing the behavior of DNS resolver software, our research shows that DNS zone operators cannot rely on these solutions always being implemented in practice. To avoid becoming unreachable for clients behind faulty resolvers, DNS zone operators have to take measures to prevent DNS responses getting fragmented and blocked by firewalls.

To the best of our knowledge, this paper is the first of its kind that investigates these problems from the perspective of the DNS zone operator. Data collected at the National Research and Education Network in the Netherlands (SURFnet) was analyzed, and results were validated using similar data acquired at the University of Pennsylvania. Two solutions were investigated and tested within the SURFnet network.

The first contribution of this paper is to what extent real-world DNS resolvers experience problems caused by fragmented responses. We recorded more than 6 hours of DNS traffic on the authoritative name servers of SURFnet; in total we collected around 8.4 million DNS(SEC) messages. Our analysis shows that 58% of all resolvers received a fragmented response at some point in time. To identify which resolvers experience problems, this paper identified five ways problematic resolvers can behave. We found that about 10.5% of all resolvers showed problematic behavior (see also Sec. II).

The second contribution of this paper, is how authoritative name servers can avoid fragmentation of DNS responses for problematic resolvers. Two solutions are presented. In the first solution an authoritative name server will limit the size of all its responses, which usually requires configuration changes within a *single* authoritative name server for that zone (see Sec. III). The second solution is more sophisticated and requires special software that alters DNS queries, but only for detected problematic resolvers. These altered queries result in smaller responses and avoid response fragmentation for problematic resolvers (see Sec. IV).

Finally the paper investigated how both solutions perform in a real-world environment. We evaluated both solutions on an authoritative name server of SURFnet. We concluded that the benefits of the second solution, where only detected problematic resolvers are assisted, do not outweigh the simplicity and real-world results of the first solution. Our proposal for production environments would therefore be to implement the first solution.

A. Future work

Further research is required to determine the advantages and disadvantages of limiting the response size on more than one authoritative name server for a zone (or even on all of them).

Secondly, we suspect that some problematic resolvers experience other issues besides receiving fragmented responses. A firewall could block DNS messages with EDNS0, or a proxy could limit UDP/DNS responses to 512 bytes [11]. Research is required to detect and help these problematic resolvers.

Acknowledgments

We would like to express our gratitude to Shumon Huque from the University of Pennsylvania for providing data to validate our research.

Our thanks also go out to the reviewers Bart Gijsen (TNO), Niels den Otter (SURFnet) and Frans Panken (SURFnet) for their constructive and detailed comments. Part of this work has been supported by the EU-FP7 FLAMINGO Network of Excellence Project (318488).

References

[1] Ariyapperuma, S., Mitchell, C.: Security vulnerabilities in DNS and DNSSEC. In: *Second International Conference on Availability, Reliability and Security* (ARES'07), p. 335-342, Washington, DC (2007).

[2] Musashi, Y., Kumagai, M., Kubota, S. and Sugitani, K.: Detection of Kaminsky DNS Cache Poisoning Attack. In: *Proceedings of the 2011 4th International Conference on Intelligent Networks and Intelligent Systems*, p. 121-124, Washington, DC (2011).

[3] Scarfone, K., Hoffman, P.: Guidelines on Firewalls and Firewall Policy. *Recommendations of the National Institute of Standards and Technology* (NIST), Gaithersburg, MD (2009).

[4] Akkerhuis, J., Chapin, L., Fältström, P., Kowack, P., Liman, L., Manning, B.: Scaling the Root. *Report on the Impact on the DNS Root System of Increasing the Size and Volatility of the Root Zone* (TNO), Delft, Netherlands (2009).

[5] Osterweil, E., Ryan, M., Massey, D., Zhang, L.: Quantifying the operational status of the DNSSEC deployment. In: *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement* (IMC'08), Vouliagmeni, Greece (2008).

[6] Rikitake, K., Nakao, K., Shimojo, S. and Nogawa, H.: UDP Large-Payload Capability Detection for DNSSEC, IEICE Trans. Inf. Sys., Vol. E91-D, No. 5, pp. 1261-1273 (2008).

[7] Weaver, N., Kreibich, C., Nechaev, B., and Paxson, V.: Implications of Netalyzr's DNS Measurements. In: *Proceedings of the First Workshop on Securing and Trusting Internet Names* (SATIN), Teddington, United Kingdom, (2011).

[8] Rikitake, K., Nakao, K., Shimojo, S., and Nogawa, H.: DNSSEC Feasibility Issues and the Transport Validation Assessment. In: *Proceedings of IPSJ Computer Security Symposium 2006* (CSS2006) (2006).

[9] Rikitake, K., Nogawa, H., Tanaka, T., Nakao, K. and Shimojo, S.: An Analysis of DNSSEC Transport Overhead Increase, IPSJ SIG Technical Reports 2005-CSEC-28, Vol. 2005, No. 33, pp. 345-350 (2005). ISSN 0919-6072.

[10] Wang, Y., Huang, C., Li, J., Ross, K.: Queen: Estimating Packet Loss Rate between Arbitrary Internet Hosts. In: *Proceedings of the 10th International Conference on Passive and Active Network Measurement* (PAM'09), Seoul, Korea (2009).

[11] Bellis, R., Phifer, L.: Test Report: DNSSEC Impact on Broadband Routers and Firewalls September 2008. Oxford, United Kingdom (2008).
[12] Cowperthwaite, A. and Somayaji, A.: The futility of DNSSec. In: *Proc. 5th Annual Symp. Information Assurance* (ASIA10), pp. 28, (2010).
[13] Herzberg, A. and Shulman, H.: Towards Adoption of DNSSEC: Availability and Security Challenges. In: *Communications and Network Security (CNS)*. IEEE Conference, (2013).

VII. BIOGRAPHIES

Gijs van den Broek Gijs van den Broek obtained a B.Sc. and a M.Sc. degrees in Telematics at the University of Twente, The Netherlands, in 2007 and 2012 respectively. This paper is related to the final research assignment of his M.Sc. degree. Since 2011 he works for ZorgTTP, a Dutch foundation operating in the field of data pseudonymisation. His expertise is in cryptography and information security.

Roland van Rijswijk-Deij Roland van Rijswijk-Deij works as Technical Product Manager for SURFnet and is responsible for SURFnet's DNS infrastructure. In this capacity he is responsible for innovation projects w.r.t. DNS, including DNSSEC. He participates in several national and international projects dealing with DNSSEC deployment in the broadest sense. Roland obtained a Master of Science degree in Computer Science from the University of Twente (2001), after which he worked in software development for Philips, Advanced Encryption Technology (AET) and InTraffic. His expertise is in the application of high-end cryptography. Roland joined SURFnet in 2008.

Anna Sperotto Anna Sperotto is postdoctoral researcher at the Design and Analysis of Communication Systems Group (DACS) of the University of Twente, The Netherlands. She received a M.Sc. degree in Computer Science from the Ca'Foscari University, Venice, Italy, in 2006 and a PhD degree from the University of Twente, in 2010. Her main topics of interest include intrusion detection and traffic modeling.

Aiko Pras Aiko Pras is Professor at the Design and Analysis of Communication Systems Group (DACS) of the University of Twente, The Netherlands. He received a PhD degree for his thesis titled "Network Management Architectures". His research interests include network management technologies, network monitoring, measurements and security. He is chairing the IFIP Technical Committee 6 on "Communications Systems", and is Project Leader of the European Network of Excellence on "Management of the Future Internet" (FLAMINGO). He is steering committee member of several conferences, including IM/NOMS and CNSM, and series/associate editor of ComMag and IJNM.