Is the Internet Ready for DNSSEC Evaluating Pitfalls in the Naming Infrastructure

Haya Shulman and Michael Waidner
Fraunhofer Institute for Secure Information Technology
Darmstadt, Germany

***Abstract*—We study the challenges of deploying DNSSEC on Domain Name System (DNS) name servers.**

DNSSEC, a defence mechanism for DNS, was designed to provide cryptographic assurance for DNS records against cache poisoning attacks. Although standardised more than 15 years ago, DNSSEC is still not widely deployed. Multiple efforts are focused on identifying deployment obstacles and it is generally believed that adopting DNSSEC is mainly a matter of motivation.

In this work we systematically explore this widely held, folklore belief. Utilising wide-scale measurements of DNS servers in the forward and the reverse DNS trees, we show that a large fraction of servers in popular domains fail with DNSSEC signed DNS requests, hence breaking the backward compatibility property of DNSSEC. This further reduces the motivation for clients to adopt DNSSEC. Our evaluation results indicate that DNSSEC deployment is a cost-benefit decision, and full adoption thereof requires upgrading significant parts of the DNS infrastructure. In particular, deploying DNSSEC on the unsigned domains *today* would render a large fraction thereof unreachable.

Our study shows two intertwined obstacles that impede the adoption of DNSSEC for DNS. One is legacy infrastructure, the other is lack of protocol support.

I. Introduction

The correctness and availability of Domain Name System (DNS), [RFC1034, RFC1035], are critical to the security and functionality of the Internet. Initially designed to translate domain names to IP addresses, the DNS infrastructure has evolved into a complex ecosystem – it is increasingly utilised to facilitate a wide range of applications and constitutes an important building block in the design of scalable network infrastructures. In particular, a secure DNS would facilitate a wide range of systems, such as secure routing (with `ROVER` [1]) and secure email, (with PGP keys distribution [2]). The significance and wide use of DNS also made it a lucrative target for attacks. Indeed, there is a long history of attacks against DNS, most notably, DNS cache poisoning, [3]–[8]. In the course of a DNS cache poisoning attack, the attacker provides spoofed records in DNS responses, thus redirecting the victim clients to incorrect hosts, e.g., for credentials theft, malware distribution, censorship and more. DNS cache poisoning has a detrimental impact on the stability and availability of the Internet, and inflicts economic losses and privacy damages to Internet clients and services.

To mitigate the detrimental damages of cache poisoning attacks, the IETF designed and standardised a cryptographic defence for DNS: Domain Name System Security Extensions (DNSSEC) [RFC4033-RFC4035]. DNSSEC was proposed in 1997, but it is still not widely deployed; measurement studies show that only about 3% of the DNS resolvers validate DNSSEC records, [9], [10]. Furthermore, although many important domains, such as the root and top-level domains (TLDs), are signed, most lower (second) level domains are not. As we show with the results collected via our large scale measurements, the fraction of the signed zones in lower domains, such as the second level domains (SLDs), is negligible. This is unfortunate, since the second level domains host the services which clients and systems typically use, such as email (e.g., `alice@bank.foo`) and websites (e.g., `www.bank.foo`).

While there has been a considerable effort to identify the challenges to DNSSEC deployment, the focus was generally on the issues that large DNSSEC responses incur with the legacy firewalls and middleboxes on the resolver side, e.g., [11], [12], [12], [13]. Zone signing was thought to be just a matter of motivation, and a folklore belief was that incentivised domain operators could sign their domains *today*. In particular, experts, [14], [15] suggest that more than 90% of the unsigned domains could deploy DNSSEC *right now* since the main hurdles (believed to be integral in impeding DNSSEC adoption) were largely resolved, specifically: (1) the root zone, as well as most top-level domains (TLDs), are already signed, and (2) automated scripts for signing the zone-files are readily available online [16]. Hence, the belief is that DNSSEC adopters can easily sign their domains and provide their clients with the immediate security benefits.

In this work we systematically explore this claim. Our study takes a complementing approach to state-of-the-art on DNSSEC deployment on the domains, [9], [17], and in addition to measuring the fraction of signed zones, we also focus on studying the challenges that the *unsigned* domains face with DNSSEC enabled DNS request, and will face when deploying DNSSEC. We ask the following question: what are the protocol and infrastructure challenges in handling DNSSEC-enabled DNS requests and in signing the zonefiles with DNSSEC?

To answer this question we utilise Internet scale measurements of the domains and name servers in forward and in reverse DNS trees, and evaluate the interoperability of the DNS name servers architecture with DNSSEC. Our evaluation shows failures, from name servers serving popular domains, when receiving DNSSEC enabled DNS requests. We also show that there are failures with DNSSEC enabled DNS responses (containing DNSSEC specific records), even when the requests are handled successfully. Since DNSSEC is defined to support backward compatibility, such failures introduce an obstacle towards DNSSEC adoption, and demotivate adoption on the client side. In particular, failures result in increased traffic on the resolvers' networks as well as to the name servers (due to timeouts and retransmissions), and degrade the performance on the client side, when resulting in failures to resolve domain names.

Our study of the challenges of DNSSEC support and interoperability on the name servers reveals obstacles imposed by: (1) legacy server-side caching (proxy) DNS resolvers, and (2) lack of support for Transmission Control Protocol (TCP), [RFC793]. In this work we illustrate the scope and the extent of both issues.

A. Contributions

Our contributions are twofold. *Conceptual:* our work highlights the challenges inherent in adopting cryptographic schemes on large scale distributed systems. In particular, design and adoption of new protocols requires careful study of the existing infrastructure. *Practical:* we perform a study of the DNS infrastructure. Our key observations are that: (1) a significant fraction of the DNS infrastructure is not ready for DNSSEC deployment; (2) the challenges are largely due to legacy infrastructure and a lack of support for basic protocols.

The implications of (1) are twofold: adopting DNSSEC on the client side will result in failed DNS resolution (which will further demotivate clients from adopting DNSSEC), and mitigation will require large upgrading efforts of the server side naming infrastructure.

Our measurement methodology for evaluating interoperability with cryptographic defences, and for locating caches, is of a general interest and can be used for studies of other distributed systems, such as Content Distribution Networks (CDN).

B. Organisation

We provide a background on DNS and the transport mechanisms used by it in Section II. We present our datasets and evaluation methodology in Section III. In Section IV we study DNSSEC failures in domains with server side caches. In Section V we evaluate failures with TCP. Finally, in Section VI we compare our results to related work and conclude in Section VII.

II. Background

In this section we provide a background on Domain Name System (DNS). Domain Name System (DNS) is a client-server protocol, used by the resolvers to retrieve domain records stored in the zone files maintained by the name servers. The resolvers communicate to the name servers using a simple request-response protocol (typically over UDP); for instance, (abstracting out details) to translate `www.foo.bar` resolvers locate the name server `ns.foo.bar`, authoritative for `foo.bar`, and obtain the IP address of the machine hosting the web server of the website `www.foo.bar`. Resolvers store the DNS records, returned in responses, in their caches for the duration indicated in the Time To Live (TTL) field of each record set.

The resource records in DNS correspond to the different services run by the organisations and networks, e.g., hosts, servers, network blocks. The zones are structured hierarchically, with the root zone at the first level, Top Level Domains (TLDs) at the second level, and millions of Second Level Domains (SLDs) at the third level. The IP addresses of the 13 root servers are provided via the *hints* file, or compiled into DNS resolvers software and when a resolver's cache is empty, every resolution process starts at the root. According to the query in the DNS request, the root name server redirects the resolver, via a `referral` response type, to a corresponding TLD, under which the requested resource is located. There are a number of TLDs types, most notably: *country code TLD* (ccTLD), which domains are (typically) assigned to countries, e.g., `us, il, de`, and *generic TLD* (gTLD), whose domains are used by organisations, e.g., `com, org`, and also US government and military, e.g., `gov, mil`. Domains in SLDs can also be used to further delegate subdomains to other entities, or can be directly managed by the organisations, e.g., as in the case of `ibm.com, google.com`.

The original DNS specifications fix the maximal size of a DNS packet, when sent over UDP, to 512 bytes [RFC1035]. Longer responses were unusual and *truncated*, by returning a partial response and signaling truncation (TC bit set). Upon receiving truncated response, resolvers should resend the request over TCP. However, using TCP for DNS requests imposes significant overhead, requires state in the name server, and adds latency to responses.

To support larger DNS responses over UDP, Extension Mechanisms for DNS (EDNS), [RFC6891], was introduced. EDNS is a hop by hop extension to DNS, and was designed to extend DNS to support new functionalities, which could not otherwise be added, since all the fields in DNS header were preallocated for different purposes. EDNS is an optional, but widely deployed, mechanism, that enables clients to advertise new capabilities to the name servers.

III. Data and Evaluation Methodology

We study the prevalence of two issues that need to be considered when adopting DNSSEC: support for TCP and the impact of legacy infrastructure hosted on complex name servers' architectures, whereby the name servers utilise server-side caching resolvers.

We first describe the data that we used for our study and then our measurement methodology.

A. Data Collection

Our study encompassed 50K-top Alexa domains, 568 Top Level Domains (TLDs), and domains in a reverse DNS tree; these domains are summarised in the left most column in Table I. We collected the IP addresses of the name servers, used by these domains; the name servers, corresponding to each set of domains, are listed in the second (left most) column in Table I.

B. Methodology

In this section we describe the settings used for evaluation of DNSSEC and TCP failures on popular nameservers.

1) Evaluation of DNSSEC: Our measurement methodology uses the name servers' infrastructure as a black box. We send DNS requests to the DNS name servers (collected from domains listed in Table I). We then utilise the responses from these DNS name servers to identify failures. More specifically, a DNSSEC enabled DNS request to a DNS server, that does not support DNSSEC, results in a timeout or an error message: **time out** – no response arrives when the `DO` bit is set in a DNS request (in contrast, we also measure that not setting the `DO` bit in a DNS request results in a response); **errors** – a request with `DO` bit is responded with a format error (`FMTERROR`) or server fail (`SRVFAIL`).

The failure may be triggered during the processing of the DNSSEC flag in a DNS request, or due to the processing of the DNSSEC records in a DNS response. We illustrate the failures in a messages exchange diagram in Figure 1. In phase (1) the server-side resolver fails (either returns an error message or does not return a response – a timeout at the client) during the processing of the `DO` bit in the EDNS record of a DNS request. In phase (2) the DNS server fails with a DNSSEC enabled DNS request, which results in a timeout or an error message on the client side.

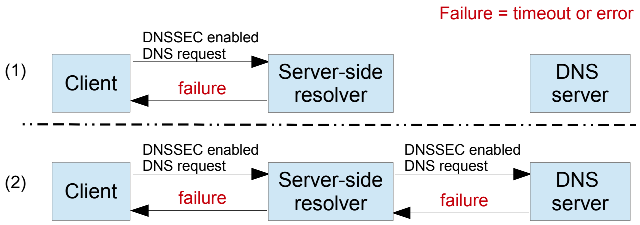

Fig. 1: Failures of handling DNSSEC enabled DNS requests or responses.

We design two measurement methodologies: *timing side-channel*, whereby we measure the latency to the name servers via DNS requests to their domains, and utilising *open recursive resolution* supported by some server-side resolvers, we send queries to our name server, under a domain that we set up. In what follows we describe the setup and application of both methodologies.

a) Timing Side Channel: We design timing side channel which allows to identify whether the DNSSEC failure occurs on the server-side caching resolver or on the name server side. The technique is illustrated in Figure 1. The idea is to utilise the difference in the latency of the responses returned by the server-side resolvers that query (hidden) name servers vs. the latency of the responses returned directly by the server-side resolvers. We illustrate the measurement of the name servers which use server-side resolvers in Figure 2; these are the name servers listed in the two left most columns in Table I. First (1) we measure the latency τms to the name server, via a DNS request to a random subdomain, then (2) we measure the latency λms to the server-side resolver, with the same DNS record (so it is responded from the cache and not forwarded to the name server); it holds that $\tau >> \lambda$ms, i.e., the latency to the name server is larger than the latency for a response from the cache. Indeed, as [18] measured, in most such configurations the name server and the cache are located on different networks, hence the latency for response from the cache of the server-side resolver is lower than that from the name server.

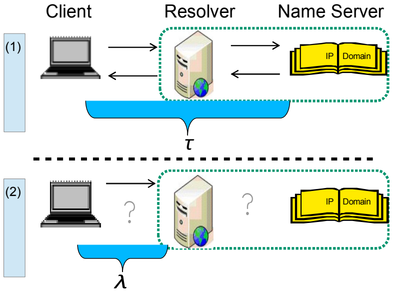

Fig. 2: Detection of server-side resolvers via latency.

b) Open Recursive Resolvers: We first introduce open recursive resolution on the server-side, and then present our methodology for evaluation of DNSSEC.

Open recursion is a known bad practice both on the client and on the server side. On the client side, the resolvers should be restricted to serve only the clients on their networks, and to provide recursive resolution for *any* domain that the authorised clients may request; see `client-side recursive resolution` in Figure 3. In contrast, on the server side, recursive server side caches should provide resolution services for *any* client, but, only to domains which the DNS cache serves; see `server-side recursive resolution` in Figure 3. Server side recursive caches, supporting open recursion, are willing to resolve *any* domain name for *any* client. We use this property for design of our methodology of the evaluation of DNSSEC support, and failures on the domains.

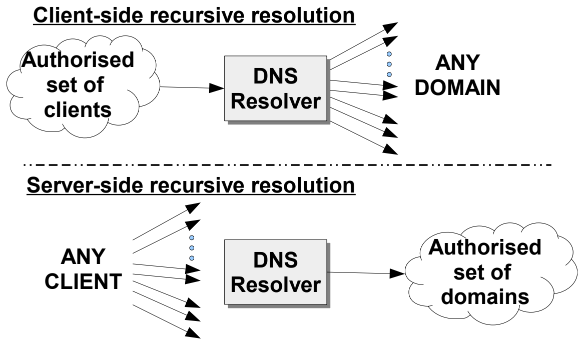

Fig. 3: Recursive DNS resolving caches on the client side and on the server side.

2) Evaluation of TCP: In our study we employ active measurements to evaluate TCP support among the 170K name servers serving domains on the 50K-top Alexa list. Our client is invoked with a list of name servers, then traverses the list and establishes TCP connections to every name server on the list. We analyse the responses (or a lack thereof) from the name servers to infer information about TCP support and configuration. Our analysis proceeds in three phases: (1) we check whether a given name server supports TCP, if so (2) we check whether the name server can serve DNS responses over TCP, and finally (3) if the TCP is correctly configured.

IV. Evaluating DNSSEC Support

In this section we measure the interoperability of DNSSEC with domains that use server-side recursive resolvers, and evaluate the difficulties in processing DNSSEC enabled DNS requests, and in deploying DNSSEC on the remaining unsigned domains. When domains use server side resolvers, all the DNS requests are relayed through a proxy resolver instead of being sent directly to a name server, see Figure 6. Hence, failures at the proxy impact the performance for queries to a name server.

Our study provides a novel angle to the deployment of DNSSEC, investigating the interoperability of the name servers' infrastructure with the extensions and the cryptographic records of DNSSEC.

In our measurement we set forth to identify which domains fail with DNSSEC or `EDNS` (an extension for DNS, [RFC6891]), whether the failure is with requests or with responses, and if the failure occurs on the name server or caused by the server-side resolvers.

A. *Measuring DNSSEC support with timing side-channels*

We study the fraction of the unsigned domains, which use server-side resolvers, that would incur failures if the zone owner adopted DNSSEC, or when the domain receives DNSSEC enabled DNS requests. The latter introduces an obstacle towards adoption of DNSSEC on the client side – DNSSEC enabled responses incur failures on some name servers, which result multiple retransmissions at the resolvers and increased latency on the client side, hence deterring clients from adopting DNSSEC. This also means that DNSSEC is not backward compatible with those name servers, in contrast to the guarantees by the DNSSEC standards, [RFC4033-RFC4035].

We report on our measurement results of latency differences between the caches in server-side resolvers and the name servers in Figure 4 (based on the evaluation described in Section III-B1). As can be seen in Figure 4 the difference is more than $100ms$ for 50% of the measured servers, and for some servers is even higher than $700ms$; such large latency indicates that they are often located in different networks. Then, (3) we send a DNS request with a `DO` bit set, and measure the latency to receive the response. Focusing on domains which fail with DNSSEC, we distinguish between the following outcomes: in case of an error, by measuring the latency of the error response we can evaluate whether the failure is on the cache or the name server. Specifically, if the error arrives in τ and $\tau >> \lambda$ ms, then the exception is on the name server, otherwise, if the error arrives in λ, then the exception is on the cache.

In case of a timeout, the error is either on the name server of the cache; the two options are illustrated in phase (2) in Figure 1.

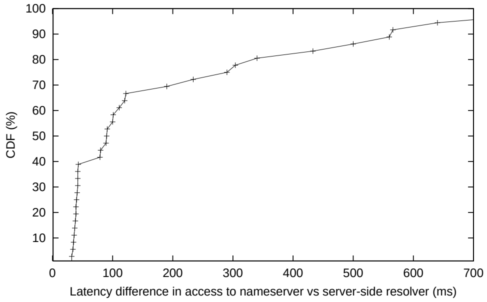

Fig. 4: Latency difference for communication to nameservers and server-side resolver.

We repeat the same experimental evaluation to study the support and interoperability with `EDNS`. The results for DNSSEC and `EDNS` are reported in Table I.

B. *Measuring DNSSEC support with open recursive resolution*

We use the setup described in Section III-B1 and seek to measure two quantities: (1) What fraction of open-recursive server-side caches are 'DNSSEC compatible', i.e., can operate with responses containing DNSSEC records. (2) What fraction of open-recursive server-side caches perform strict validation of DNSSEC responses. The second question is only possible to measure for domains with open recursive caches (since most domains are still not signed). During the measurement, we trigger queries to our *signed* domain. We use the setup (see Section III-B1 for setup details)We seek to measure two quantities: (1) What fraction of open-recursive server-side caches are 'DNSSEC compatible', i.e., can operate with responses containing DNSSEC records. (2) What fraction of open-recursive server-side caches perform strict validation of DNSSEC responses. The second question is only possible to measure for domains with open recursive caches (since most domains are still not signed). During the measurement, we trigger queries to our *signed* domain. We set up three domains: (1) without DNSSEC, (2) correctly signed with DNSSEC, (3) incorrectly signed. Domain (1) was not signed, and served plain DNS responses. Domains (2) and (3), were signed with 1024 bit keys RSA/SHA-1 (algorithm 5), [RFC3110,RFC4034]. Domain (3) served expired keys in `DNSKEY` records and invalid signatures in `RRSIG` record.

We performed three sets of tests using the domains we setup: (a) we sent DNS requests for `A` record of `www.our-domain.tld`, with `EDNS0` record and set the `EDNS buffer size` to 4096; (b) we sent requests for `DNSKEY` record of `bad-domain.tld` and for `A` `www.bad-domain.tld`, with `EDNS0` record and set the `DO` bit in `EDNS` record; (c) we sent requests for `DNSKEY` record of `good-domain.tld` and for `A` `www.good-domain.tld`,

domain	regs	NS-es	signed	fail w/EDNS	fail w/DNSSEC	caching	open-caching
Forward DNS Alexa	50K	32.5K	0.46%	12.7%	23%	38%	6%
Forward DNS TLDs	568	3.2K	62%	1.6%	2%	12%	3.73%
rDNS `x.in-addr.arpa.`	229	1.5K	84%	1%	2%	14%	8%
rDNS `y.x.in-addr.arpa.`	28K	97K	0.4%	19%	32%	41%	19%
rDNS `z.y.x.in-addr.arpa.`	2,767K	9,687K	0.06%	19.5%	34.5%	38%	21%

TABLE I: Summary of the domains and incompatibility with DNSSEC.

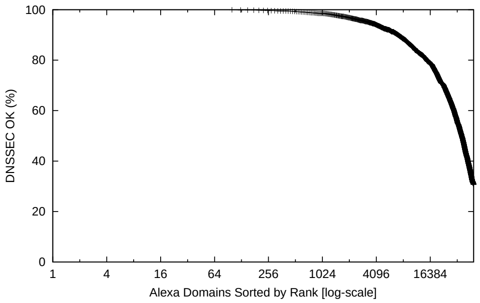

Fig. 5: Support of DNSSEC-signed DNS packets among open recursive name servers of top Alexa domains.

with `EDNS0` record and set the `DO` bit in `EDNS` record. We ran a `tcpdump` capture on the name servers, authoritative for our domains.

C. Compatibility with DNSSEC and EDNS

To test compatibility of open recursive authoritative name servers with DNS extention mechanisms, EDNS and DNSSEC, we performed test (c). We found that 39.2% of the open recursive authoritative name servers could not process signed DNSSEC responses, and returned `FMTERROR/SRVFAIL`, 29.8% stripped DNSSEC records from responses and returned plain DNS responses (without signatures and keys). Together this resulted in a bit more than 69% of servers that can not support DNSSEC. We found that only 30.9% of the open recursive authoritative name servers return DNSSEC enabled responses, in return to requests for records in signed domains.

We further tested support of `EDNS0` among the open recursive authoritative name servers. Clearly, the 30.9% of them that support DNSSEC also support `EDNS0`. What about the remaning 69% that do not support DNSSEC. We found that 52% of them support `EDNS0`. In total, we observed that 82% of the open recursive name servers support `EDNS0` while 18% do not.

We measured the failure rate with DNSSEC among the Alexa domains, and found that the failure rate increases for less popular domains. In Figure 5, we plot the ability to handle DNSSEC enabled packets, as can be seen, the less popular domains have less success rate and the curve, plotting the success rate, decreases for lower ranked domains, as the failures rate increases. This result supports the intuition that the more popular the domains are the better they are maintained.

D. Strict DNSSEC Validation

To test whether open recursive authoritative name servers perform strict validation of DNSSEC responses, we ran test (b) on the 30.9% of the open recursive authoritative name servers that could serve signed responses.

None of the queries failed, namely, the open recursive authoritative name servers do not support strict DNSSEC validation. This result is consistent with measurements reported in [9], which showed that most recursive DNS resolvers, that support DNSSEC, do not perform strict DNSSEC validation.

We summarise our findings pertaining to DNSSEC support in complex server infrastructures in Table I. In forward DNS, we found that 62% of the Top-Level Domains (TLDs) are signed, but only less than one percent (0.46%) of the top million Alexa domains (SLDs) are signed.

In the reverse DNS tree the number of signed domains is even lower: only 0.07%. This is surprising since the reverse DNS is commonly utilised by the security mechanisms, hence one would expect better awareness to securing the DNS records. We found that very few of the reverse DNS domains, that correspond to classes B and C (i.e., `x.y.in-addr.arpa` and `x.y.z.in-addr.arpa`), are signed (only 0.4% and 0.06% respectively). This means that even if the lower level domains decide to adopt DNSSEC, the resolvers will not be able to establish a chain of trust to them, to enable validation of their DNSKEY (public verification key) records.

The next columns 'fail w/EDNS' and 'fail w/DNSSEC' report the amount of domains that do not support the `EDNS` record, [RFC6891], or DNSSEC [RFC4034-RFC4035] respectively, and fail with an exception or a timeout.

The two rightmost columns 'caching' and 'open caching' contain the number of name servers which use caching (resp. open) recursive DNS resolvers, studied in [18]; open recursive resolvers are willing to lookup names in *any* domain and not only in the domain for which they are 'authoritative'. In such a configuration, the server-side caches are configured to relay all the communication between the client-side resolvers and the name servers; see Figure 6. The IP address of the server-side resolver is registered as the name server in the zone file of the target domain (and in its parent). The server-side resolver receives DNS requests from the client-side resolvers and forwards them to the name server hosting the zone file for the target domain. Upon responses from the name server, the server-side resolver caches the DNS records and subsequently returns them to the requesting client-side resolver. Similarly to the standard DNS resolvers functionality, if the requested record is in the cache, the server-side resolver does not forward the query to the name server, but responds from the cache. The setting is illustrated in Figure 6.

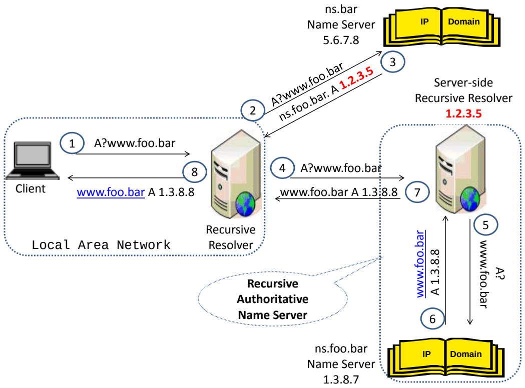

Fig. 6: Resolution for `www.foo.bar`. The name server is hidden behind server-side resolver.

V. Evaluation of TCP Implementations

TCP is essential for the delivery of large DNSSEC enabled responses. The original DNS specifications fix the maximum size of a DNS packet, when sent over UDP, to 512 bytes [RFC1035]. Larger responses were sent over TCP. To support larger DNS responses over UDP, Extension Mechanisms for DNS (EDNS), [RFC6891], was introduced. In the absence of EDNS, the normal behaviour is to send packets exceeding 512 bytes over TCP. Indeed, some server still do not support EDNS. But, TCP may be required to deliver large DNS response even when EDNS is supported, e.g., firewalls often block fragments or large UDP datagrams (exceeding 512 bytes). TCP is critical for delivery of DNSSEC responses. In this section we perform a study of TCP support among the name servers. Our results show that almost 20% of the name servers cannot serve responses over TCP. Worse, 2% of those servers do not support EDNS, which means that they will not be able to serve DNSSEC enabled responses, or even plain DNS responses that exceed 512 bytes. In what follows we explain our evaluation and results of the phases listed in Section III-B2.

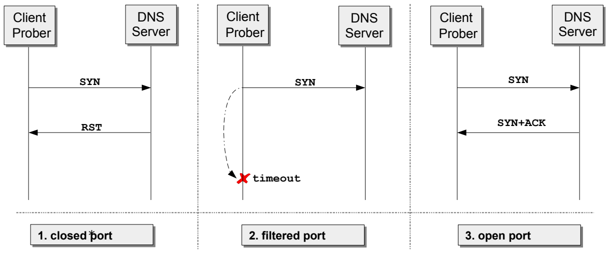

Fig. 7: Identifying state of TCP on a nameserver: closed, filtered or open.

A. No TCP (phase 1)

To check if the name server has a TCP daemon listening on port 53, we send a TCP `SYN` packet and analyse the response from the name server. As a general rule, a TCP `RST` packet should be sent if the connection does not exist (closed). We mark port as filtered if the port is behind a firewall, and `RST` packets are filtered. In that case a timeout will occur on the client. Finally, if the port is in listening state, a `SYN ACK` packet will be returned by the server, and we indicate that the TCP port is open (i.e., TCP support exists on the name server). We illustrate the three cases in Figure 7. For ports that were classified as open we proceed to phase 2, and check whether the name server can respond to DNS requests over TCP. Out of the checked servers 6% have closed or filtered TCP port 53.

B. Failure to Send Data Over TCP (phase 2)

We continue the evaluation with name servers that responded with a `SYN ACK` in phase 1. Next, we send a DNS request, with a query for a resource record within a domain for which the name server is authoritative. The behaviour of the name servers in this phase varies significantly. We identified three main categories of failures when serving DNS responses over TCP.

Premature Connection Closure. This group contains servers (settings A and B in Figure 8) contains those that close a connection before a DNS transaction is complete, namely, before a response to a DNS request is sent. We identified two cases: (1) servers that send a `RST` packet in response to a DNS request and (2) servers that send a `FIN` packet. The source IP address in both the `RST` and `FIN` packets is that of the DNS server. The cause for the former is *short connection timeout*, described in [RFC2525]. The latter is due to a misconfiguration, as a result the server indicates to the client that it will not send any more data.

Infinite Loop. The second group (setting C in Figure 8) contains servers which keep resending a `SYN ACK` packet, without sending the actual DNS response.

Blocking TCP at Network Layer. This group of servers (setting D in Figure 8) contains those that send an `ICMP` packet (`type=3, code=10`) in response to a DNS request which indicates that the server cannot answer and is *administratively prohibited*, [RFC1812]. The ICMP message is generated if the router cannot forward the packet due to administrative filtering, and signals that it filters DNS requests over TCP to port 53, by sending an ICMP packet. To identify the sources of the `RST` and `ICMP` packets we used the `traceroute` tool to the destination host, and in particular to identify the IP address of the router/firewall on the network of the DNS server. Then we compare that IP address, to the source IP address in the IP header of `ICMP` packet. When the addresses are equivalent, the filtering is done at the firewall/router on the network of the DNS server. Our study shows that 13% of the name servers fail to serve responses over TCP, with the majority of the servers exhibiting a *premature connection closure* (more than 60% of the servers).

C. TCP Failures At Data Transfer (phase 3)

The misconfigurations are illustrated in Figure 9. Settings A, B and C do not cause failures on the client side, but result in degraded performance. In A the server advertises an unstable

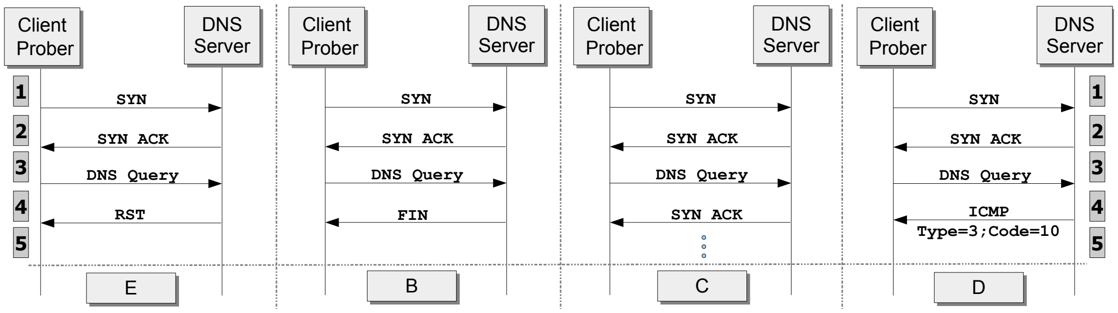

Fig. 8: Failures to send DNS responses.

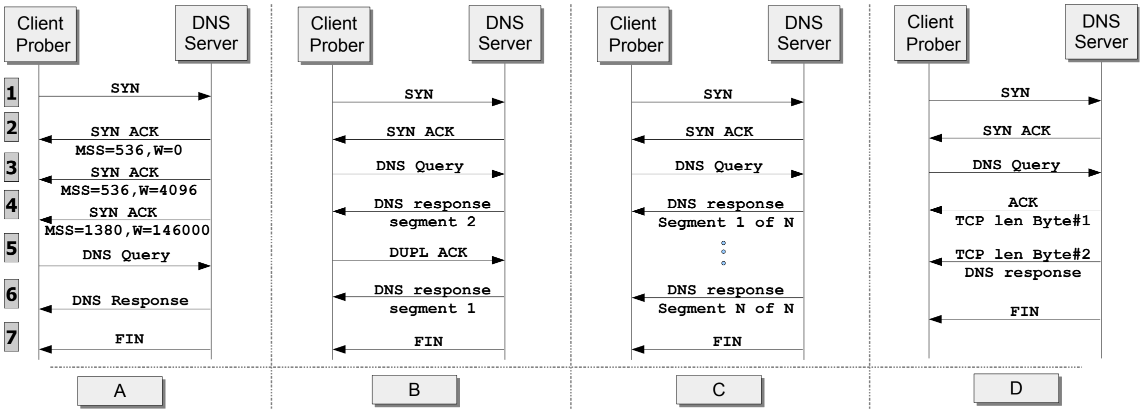

Fig. 9: Misconfigurations in TCP implementations.

TCP flow-control window and Maximum Segment Size (MSS) values, in B the segments arrive out of order (first segment arrives after the second), in C the DNS response is split to segments, each up to 100 bytes. Finally, in D, the two byte TCP length field, required to parse the payload in a DNS response, is split among two segments (first byte appears in segment ACK-ing the DNS query, and second byte in the DNS response). This prevents the TCP implementation at the client side from reassembling the DNS response, hence resulting in failure and timeout on the client side. Such misconfigurations are not common, 4%, but nevertheless hinder performance, and may (as in setting D in Figure 9) result in failures.

VI. Related Work

Previous work, on DNSSEC adoption, either measured the fraction of the *signed* zones, [11], [12], or the fraction of *validating resolvers*, [9], [19], however, the ability and the readiness of the name servers to adopt DNSSEC or serve signed responses did not receive sufficient attention. We initiate this investigation in our work. Our work takes a complementary approach to state-of-the-art on the discovery of the DNS infrastructure. First, in contrast to previous work, [20]–[22], that studied the client-side infrastructure, we focus on exploring the name servers. Second, our probing strategy is different – we do not scan the IPv4 address space for discovery of DNS services, like [20]–[22], but we send queries for records within domains in forward and reverse DNS trees; our work also utilises queries to the name servers, similarly to [23], which measured the latency for queries to common domains, and studied the consistency of time-to-live (TTL) values in DNS records in the responses. In [24] the authors also utilised the latency to the DNS name servers, via inspection of the DNS traces collected on the resolvers, to study the performance of the DNS name server selection algorithm. Recently, [18] studied the operators of third party caches, showing that often these are not the owners of the domains.

In addition to architecture challenges pertaining to adoption of DNSSEC, we also investigate transport protocols support within the domains, specifically focusing on TCP and EDNS. Both transport mechanisms, the TCP and the EDNS, are a basic assumption in DNS, and were standardised in [RFC1035,RFC5966] and [RFC6891] respectively. The proposals for defences of DNS assume the existence of TCP in the transport layer, e.g., [25]; and e.g., see [RFC7626] for a survey of privacy proposals for DNS. Some of the privacy mechanisms were already integrated into the DNS resolvers' software, such as `unbound`. However, our evaluation has not detected name servers supporting privacy mechanisms within the domains in the forward DNS tree (Alexa and TLDs) and in the reverse DNS tree.

Prior to our work, [26] found that 17% of the clients cannot receive DNS traffic over TCP. Nevertheless, it was believed that the name servers support TCP by default. In this work we measure the existence and support of TCP among the name servers. Our evaluation shows that many (popular) domains either do not have TCP at all, or run a bogus TCP implementation. Our work is the first to show that TCP is a non-trivial assumption *also* on the name server side.

VII. Conclusions

In this work we present a study of the challenges in adopting cryptographic defences on large scale distributed systems. Deploying a security mechanism over an existing infrastructure may often introduce new obstacles and failures as well as communication and computation overhead, e.g., see [27] for a study of the overhead introduced by DNSSEC; for instance, failures among the signed zones are much more common than among unsigned zones, [9], [28]. Hence, understanding the infrastructure and foreseeing potential failures is critical prior to adoption of a new security mechanism.

In this work we studied obstacles towards adoption of DNSSEC. We show that deploying DNSSEC on many unsigned zones, which use server-side caches, may disrupt the DNS functionality for clients to those domains. In addition, we show failures to unsigned zones, when the clients signal support of DNSSEC via the `DO` bit, which contradicts the backward compatibility requirement of DNSSEC, and further reduces the motivation to adopt DNSSEC on the client side.

The obstacles are often an artifact of hosting the DNS services on a legacy DNS infrastructure.

In our study we also detected a significant fraction of servers which cannot serve responses over TCP, and hence some clients would often fail to receive signed DNSSEC responses from those domains. Since TCP was mandatory for DNS, [RFC1035], and it is a natural assumption that DNS servers can serve responses over UDP or TCP, we find it surprising that a large fraction of popular domains (according to Alexa ranking) still do not support TCP.

The deployment of any new mechanism in the Internet is a challenging task, that is tightly coupled with understanding the existing infrastructure and protocols' support.

Our measurements methodology, for study of obstacles and interoperability of the existing DNS infrastructure with cryptographic defences, as well as techniques for detection of caches, are of independent interest and are applicable for studies of other distributed systems.

VIII. Acknowledgements

We thank the anonymous referees for their thoughtful feedback on our work, and in particular Gareth Tyson for his useful comments. Part of this research was supported by a Microsoft Azure Research Award.

References

[1] J. Gersch and D. Massey, "Rover: Route origin verification using dns," in *Computer Communications and Networks (ICCCN), 2013 22nd International Conference on*. IEEE, 2013, pp. 1–9.

[2] P. Wouters, "Using DANE to Associate OpenPGP public keys with email addresses," 2014, http://tools.ietf.org/html/draft-wouters-dane-openpgp-02.

[3] J. Stewart, "Dns cache poisoning–the next generation," 2003.

[4] A. Herzberg and H. Shulman, "Fragmentation Considered Poisonous: or one-domain-to-rule-them-all.org," in *IEEE CNS 2013. The Conference on Communications and Network Security, Washington, D.C., U.S.* IEEE, 2013.

[5] Chipotle, "Chipotle Twitter DNS Hijacking," https://twitter.com/mzbat/status/564428557514858496/photo/1, 2015.

[6] A. Herzberg and H. Shulman, "Security of patched DNS," in *Computer Security - ESORICS 2012 - 17th European Symposium on Research in Computer Security, Pisa, Italy, September 10-12, 2012. Proceedings*, 2012, pp. 271–288. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-33167-1_16

[7] ——, "Vulnerable delegation of DNS resolution," in *Computer Security - ESORICS 2013 - 18th European Symposium on Research in Computer Security, Egham, UK, September 9-13, 2013. Proceedings*, 2013, pp. 219–236. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40203-6_13

[8] ——, "Socket Overloading for Fun and Cache Poisoning," in *ACM Annual Computer Security Applications Conference (ACM ACSAC), New Orleans, Louisiana, U.S.*, C. N. P. Jr., Ed., December 2013.

[9] W. Lian, E. Rescorla, H. Shacham, and S. Savage, "Measuring the Practical Impact of DNSSEC Deployment," in *Proceedings of USENIX Security*, 2013.

[10] K. Fukuda, S. Sato, and T. Mitamura, "A technique for counting dnssec validators," in *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013, pp. 80–84.

[11] H. Yang, E. Osterweil, D. Massey, S. Lu, and L. Zhang, "Deploying cryptography in internet-scale systems: A case study on dnssec," *Dependable and Secure Computing, IEEE Transactions on*, vol. 8, no. 5, pp. 656–669, 2011.

[12] A. Herzberg and H. Shulman, "Retrofitting Security into Network Protocols: The Case of DNSSEC," *Internet Computing, IEEE*, vol. 18, no. 1, pp. 66–71, 2014.

[13] ——, "Right-signing: Efficient distribution of signed data for dnssec, wsns and beyond," in *Conference on Cryptology and Network Security (CANS)*, 2014.

[14] G. Huston and G. Michaelson, "Recounting DNSSEC," 2012, http://www.potaroo.net/ispcol/2012-10/counting-dnssec-2.html.

[15] APNIC, "Measuring DNSSEC Use," 2013, http://labs.apnic.net/presentations/store/2013-10-15-dnssec.pdf.

[16] I. S. Consortiuml, "BIND 9.7.2 and automatic DNSSEC signing," 2014, http://www.isc.org/blogs/bind-9-7-2-and-automatic-dnssec-signing/.

[17] E. Osterweil, "SecSpider deployment stats," http://secspider.cs.ucla.edu/stats.html, May 2013, accessed May 2013.

[18] H. Shulman and M. Waidner, "Towards Security of Internet's Naming Infrastructure," in *Computer Security - ESORICS 2015 - 20th European Symposium on Research in Computer Security, Wien, Austria, September, 2015. Proceedings*, 2015.

[19] O. Gudmundsson and S. D. Crocker, "Observing DNSSEC Validation in the Wild," in *SATIN*, March 2011.

[20] D. Leonard and D. Loguinov, "Demystifying service discovery: implementing an internet-wide scanner," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM, 2010, pp. 109–122.

[21] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig, "Comparing dns resolvers in the wild," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM, 2010, pp. 15–21.

[22] K. Schomp, T. Callahan, M. Rabinovich, and M. Allman, "On measuring the client-side dns infrastructure," in *Proceedings of the 2013 conference on Internet measurement conference*. ACM, 2013, pp. 77–90.

[23] R. Liston, S. Srinivasan, and E. Zegura, "Diversity in dns performance measures," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*. ACM, 2002, pp. 19–31.

[24] K. Fukuda, S. Sato, and T. Mitamura, "Towards evaluation of dns server selection with geodesic distance," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*. IEEE, 2014, pp. 1–8.

[25] L. Zhu, Z. Hu, J. Heidemann, D. Wessels, A. Mankin, and N. Somaiya, "T-dns: Connection-oriented dns to improve privacy and security," in *SIGCOMM*, vol. 14, 2014, pp. 17–22.

[26] G. Huston, "A Question of DNS Protocol," 2013, http://www.circleid.com/posts/.

[27] ——, "The Cost of DNSSEC," http://www.potaroo.net/ispcol/2014-08/dnsseccost.html, 2014.

[28] COMCAST, "Analysis of DNSSEC Validation Failure," http://dns.comcast.net/images/files/dnssec_validation_failure_nasagov_20120118_final.pdf, 2012.