Quantifying and Improving DNSSEC Availability

Casey Deccio
Sandia National Laboratories
ctdecci@sandia.gov

Jeff Sedayao and Krishna Kant
Intel Corporation
{jeff.sedayao,krishna.kant}@intel.com

Prasant Mohapatra
University of California, Davis
pmohapatra@ucdavis.edu

***Abstract*—The Domain Name System (DNS) is a foundational component of today's Internet for mapping Internet names to addresses. With the DNS Security Extensions (DNSSEC) DNS responses can be cryptographically verified to prevent malicious tampering. The protocol complexity and administrative overhead associated with DNSSEC can significantly impact the potential for name resolution failure. We present metrics for assessing the quality of a DNSSEC deployment, based on its potential for resolution failure in the presence of DNSSEC misconfiguration. We introduce a metric to analyze the administrative complexity of a DNS configuration, which contributes to its failure potential. We then discuss a technique which uses soft anchoring to increase robustness in spite of misconfigurations. We analyze a representative set of production signed DNS zones and determine that 28% of the validation failures we encountered would be mitigated by the soft anchoring technique we propose.**

I. Introduction

A key part of the Internet is the Domain Name System (DNS). The DNS is a distributed, hierarchical database primarily used for mapping domain names to Internet addresses and is required for nearly every network transaction. To protect the integrity of DNS responses, security solutions have been proposed, most notably the DNS Security Extensions (DNSSEC) [1]. DNSSEC adds the ability to sign and cryptographically validate DNS data. While DNSSEC deployment is still relatively low, the number of DNSSEC-signed zones has increased significantly in the two years [2]–[4], and in 2010 it reached a milestone with the signing of the DNS root zone [5].

As early adopters have begun signing zones and enabling validation, experience has shown that DNSSEC requires significantly more administration than standard DNS. The complexities of DNSSEC are fertile ground for misconfigurations which affect the ability to properly resolve domain names. The availability of dependent domain names is also much more likely to be affected by misconfiguration. For example, in October 2010 signatures accompanying DNS records in the *be* (Belgium) country-code top-level domain expired[1]. Until the signatures were renewed, validating resolvers were unable to successfully resolve domain names under *be*. Regular monitoring and maintenance, coupled with sound protocol understanding, is essential to successful DNSSEC deployment.

In this paper we review the DNS protocol and discuss the extensions related to DNSSEC. We identify DNSSEC-related misconfigurations which affect name resolution and present a metric to quantify their impact, based on resolver behavior. We introduce a metric to analyze the administrative complexity of a DNS zone, a contributing factor to misconfiguration. We propose a system for soft anchoring to minimize the impact of misconfigurations on name resolution. Using production DNSSEC data we show the pervasiveness of DNSSEC-related misconfigurations and show how our soft anchoring approach helps maintain availability of otherwise unreachable zones. We list the following as the major contributions of this paper:

- Metrics for quantifying the potential for resolution failure of zones due to DNSSEC misconfiguration, based on resolver behavior.
- A survey of failure potential due to misconfiguration in production DNSSEC zones.
- A mechanism for soft anchoring to increase robustness in spite of DNSSEC misconfiguration.

In Section II we review the fundamentals of DNS and DNSSEC. In Section III we present metrics for evaluating DNSSEC availability and analyze complexity of DNS configurations which increase the potential for failure. Section IV describes our proposal for increasing robustness through soft anchoring. Section V discusses the methodology we employed for data collection and our analysis of production DNS data in light of our metrics. Related work is summarized in Section VI, and we conclude in Section VII.

II. DNS Background

Name resolution in the Domain Name System (DNS) [6], [7] typically involves three roles: a *stub resolver*, a *recursive resolver*, and an *authoritative server*. The resolver library of an operating system is a stub resolver. It is configured with n recursive resolvers, $r_1, r_2, \ldots, r_n$, to which it directs name lookups (e.g., in `/etc/resolv.conf` for UNIX systems). The recursive resolver receives queries and performs iterative requests to authoritative name servers to find the answers, following downward referrals from servers in higher level domains until it receives an answer from a server authoritative for the name in question. Figure 1 illustrates the roles of stub resolver, recursive resolver, and authoritative server in name resolution.

This research was supported in part by the National Science Foundation through the grant CNS-0716741.

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

[1]http://dnssec-deployment.org/pipermail/dnssec-deployment/2010-October/004513.html

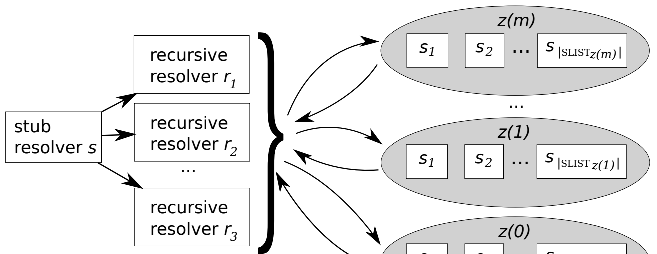
stub resolver s
recursive resolver r_1
recursive resolver r_2
...
recursive resolver r_3
$z(m)$
s_1 s_2 ... $s_{|\text{SLIST}_{z(m)}|}$
...
$z(1)$
s_1 s_2 ... $s_{|\text{SLIST}_{z(1)}|}$
$z(0)$
s_1 s_2 ... $s_{|\text{SLIST}_{z(0)}|}$

Fig. 1. A typical DNS setup in which a client (stub resolver) is configured to use n recursive resolvers which resolve a name in zone z with an ancestry of size m.

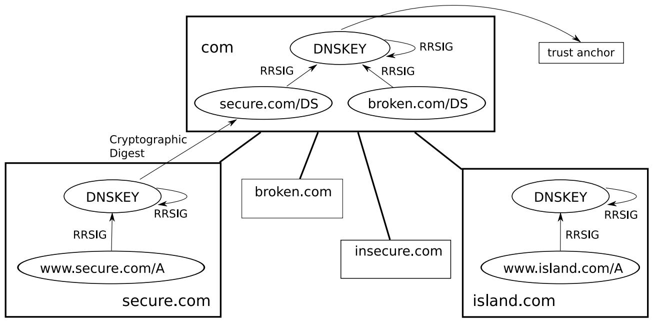

Fig. 2. The DNSSEC authentication chain for several fictitious zones. RRSIGs are represented by upward arrows extending from the RRset they cover to the DNSKEY which can validate it. SEP DNSKEYs are mapped to their corresponding trust anchor or DS RR with an arrow. Self-signatures at each SEP DNSKEY are represented by a self-loop.

DNS questions and answers consist of *resource records* (RRs), each of which has a name (e.g., *www.example.com*), a time-to-live (TTL) value, a type (e.g., A), and record data specific to its type (e.g., an Internet address for an A RR). RRs of the same name and type comprise a *resource record set* (RRset). Let $z(i)$ denote the zone i generations above zone z, such that $z(0) = z$, $z(1) = Parent(z)$, and $z(m)$ is the root zone. Figure 1 uses this notation.

The DNS Security Extensions (DNSSEC) [1], [8], [9] add authentication to DNS. Public keys are included in the zone data for each zone using a DNSKEY RR. Each RRset in a zone is signed by the zone's private key, and each signature is included in an RRSIG RR. An RRSIG also includes validity dates, and any RRSIGs covering an RRset must be included in the response to a DNSSEC query.

A resolver must authenticate a DNSKEY before it can be used to authenticate an RRset. The resolver is initially seeded with a *trust anchor* corresponding to a DNSKEY that signs its own DNSKEY RRset (i.e., is self-signing). This anchor provides a *secure entry point* (SEP) into the zone and authenticates all the DNSKEYs in the RRset. A DS (*delegation signer*) RR is maintained in a parent zone and contains a cryptographic digest of a DNSKEY in the child zone, which creates a SEP into the child zone. With the DS signed by the parent zone, a resolver is able to form a *chain of trust* from an RRset to a trust anchor in an ancestor zone. An *island of security* is a chain of trust comprised of one or more zones whose "top" zone is not securely linked to its parent. Figure 2 illustrates the chain of trust for an example DNS hierarchy. The *secure.com* zone is linked to its parent, while *island.com* is an island of security. Neither *broken.com* nor *insecure.com* are signed.

When there is no secure link from a parent zone to its child, the parent authoritative server must prove that no DS RRs exist, yielding an *insecure delegation*. Parent authoritative servers return signed NSEC RRs with such negative responses to demonstrate this proof. For example, a server authoritative for the *com* zone in Figure 2 should send the appropriate NSEC RR(s) in response to a query for DS RRs for the *insecure.com* and *island.com* zones.

When a resolver is configured to validate DNS responses, there are three primary outcomes with regard to validation of an RRset using DNSSEC [9]: *secure*, *insecure*, and *bogus*. Secure responses result from an unbroken chain of trust between the RRset and a trust anchor. If the resolver cannot establish an chain of trust but can show securely show that no such chain should exist, the result is an insecure response. A response is bogus if resolver cannot complete a chain of trust and cannot prove that no such chain should exist.

III. DNSSEC Availability

While DNSSEC has the obvious advantage of allowing a resolver to cryptographically verify the answer given for a domain name query, it adds complexity to the requirements for name resolution, and increases the potential for failure. Any server or zone misconfiguration in the line of trust between anchor and query name widens the target of error. In this section we model name resolution in a DNSSEC deployment and provide metrics for measuring the potential for validation failure based on misconfiguration. We focus solely on the issue of availability due to improper DNSSEC configuration and do not consider incorrect responses which are the result of malicious tampering.

A. Failure potential

We group misconfigurations contributing to bogus validation into three classes:

- *Zone*: Missing, expired, or otherwise invalid RRSIGs covering zone data; *or* missing DNSKEY RRs required to verify RRSIGs.
- *Delegation*: Bogus delegations caused by lack of appropriate DNSKEYs in a child zone corresponding to DS RRs in the parent zone; *or* insufficient NSEC RRs to prove an insecure delegation to a resolver.
- *Anchor*: Stale trust anchors in a resolver, which no longer match appropriate DNSKEYs in the corresponding zone.

Failure potential measures the probability that a resolver encounters a misconfiguration which would consequently result in the validation failure of an RRset. The two clearest cases are those in which failure potential is either 0 or 1—that is validation either certainly succeeds or certainly fails. Those cases occur when authoritative servers all respond consistently.

In practice, there is often variance between server behavior, either in terms of data synchronization or levels of DNSSEC support. For example, if an authoritative server fails to transfer a new version of a zone from its master, it may continue to serve expired `RRSIG`s. Likewise, if a server lacks DNSSEC support, then its responses will not contain the RRs required for validation, such as `RRSIG` or `NSEC`.

We base our metric on the following behavior: a non-error response received by a resolver does not induce a follow-up query to another authoritative server, even if the response is lacking valid DNSSEC information; alternatively if a resolver receives no response, or if the response has a `SERVFAIL` status, the resolver will query another server. More discussion on different resolver behavior is discussed later in this section. We use $\texttt{SLIST}_z$ to denote the set of addresses of servers authoritative for zone z, populated either by glue records or resolved independently by the resolver [7], [10]. The subset that respond without errors is denoted $\texttt{SLIST}'_z \subseteq \texttt{SLIST}_z$.

Let $B(z) \subseteq \texttt{SLIST}'_z$ denote the set of servers authoritative for zone z which serve bogus or incomplete DNSSEC data for zone z, resulting in a *zone*-class misconfiguration. Specification dictates that a particular resolver prefer authoritative servers with the best response history, after each has been tried at least once [7]. However, we assume a distribution of resolvers and authoritative servers such that any authoritative server for a zone has an equal chance of being selected for query by an arbitrary resolver. Under such circumstances the probability, $P'_f(z)$, that the resolver queries a server whose response results in a bogus validation is:

$$P'_f(z) = \begin{cases} 1.0 & \text{if } \texttt{SLIST}'_z = \emptyset \\ \frac{|B(z)|}{|\texttt{SLIST}'_z|} & \text{otherwise} \end{cases} \tag{1}$$

Because validation must follow the authentication chain from the zone in question to a trusted anchor, we include the servers authoritative for zones in z's ancestry, $z(0)$ through $z(m)$, as illustrated in Figure 1. Let $z(a), a \in [0, m]$ denote the zone for which the resolver is configured with a trust anchor. While it is possible that a resolver is configured with multiple trust anchors within the same hierarchy, for the purposes of this paper we assume that for a given zone hierarchy at most one trust anchor exists.

We also consider the case of insecure delegation. Let $z(j), j \in (0, a]$ denote a zone such that 1) the delegations between $z(a)$ and $z(j)$ are linked with a chain of trust and 2) the delegation from zone $z(j)$ to zone $z(j-1)$ is insecure. Figure 3 illustrates the delegation model with this notation.

When extending failure potential for zone z to include its entire ancestry we must now consider *zone*-class DNSSEC problems in zones $z(i), i \in [j, a]$, *delegation*-class problems in zone $z(i)$ affecting delegation to $z(i-1)$, $i \in [j, a]$, and *anchor*-class problems in zone $z(a)$. We denote the sets of servers serving such bogus data for zone $z(i)$ as $B_z\big(z(i)\big)$, $B_d\big(z(i)\big)$, and $B_a\big(z(i)\big)$, respectively. We combine the sets to form the comprehensive set referenced in Equation 1:

$$B\big(z(i)\big) = B_z\big(z(i)\big) \cup B_d\big(z(i)\big) \cup B_a\big(z(i)\big) \tag{2}$$

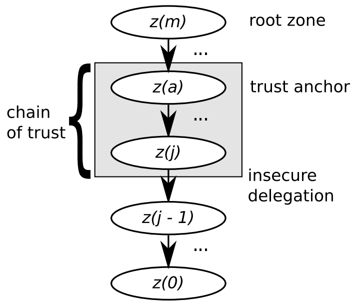

Fig. 3. An illustration of the delegation model for zone z with ancestry $z(0)$ to $z(m)$ and chain of trust extending from $z(j)$ to $z(a)$, anchored at zone $z(a)$.

Note that for other zones, outside the chain of trust, non-anchor-class DNSSEC errors are innocuous, regardless of whether or not $z(i)$ is signed, such that $B_z\big(z(i)\big) = \emptyset$ and $B_d\big(z(i)\big) = \emptyset$. We now combine the probabilities of querying a misconfigured server in the preceding zones as independent events to define the potential for zone z to fail validation by a resolver:

$$P^r_f(z) = 1 - \prod_{i=0}^{m} \Big(1 - P'_f\big(z(i)\big)\Big) \tag{3}$$

If a recursive resolver determines that validation of an RRset has failed with bogus status, its response to the requesting stub resolver is a server failure message (`SERVFAIL`). A stub resolver receiving such a response typically fails over to its next configured recursive resolver. If responses across authoritative servers are consistent for all zones $z(i), i \in [j, a]$ then $P^r_f(z)$ will always be 0 or 1—all resolvers from the same vantage point will either fail together or succeed together. However, when authoritative servers exhibit inconsistent DNSSEC behavior, failure potential is a fraction and is reduced exponentially with each validation attempt by a distinct recursive resolver since each validates independently:

$$P^s_f(z) = \left(1 - \prod_{i=0}^{m} \Big(1 - P'_f\big(z(i)\big)\Big)\right)^n \tag{4}$$

where the resolver is configured with n recursive resolvers.

We have only considered a common DNS configuration exemplified in Figure 1. In this paper, our figures for failure potential are calculated based on a configuration involving a single validating resolver (i.e., $n = 1$).

B. Failure potential factors

A DNSSEC deployment must be carefully coordinated and monitored both hierarchically (between a zone and its ancestors) and laterally (between servers authoritative for the same zone). Issues with either can result in increased failure potential for descendant namespaces. We address the hierarchical component in Section IV, presenting a novel protocol for mitigating problems in ancestor zones. In this section we analyze the lateral contributor to failure potential, inconsistency across authoritative servers.

Administrative complexity describes the diversity of a zone, with respect to organizations administering its authoritative servers. While third-party hosting is often advantageous to gain geographic and network diversity for high availability, this collaboration increases the potential for failure. Differences in server implementation or unilateral changes to server address, firewalls, or server configuration could ultimately result in an inconsistent behavior between servers from distinct organizations. One means of measuring this diversity is simply examining the number of distinct organizations operating authoritative servers for a zone. However, to characterize the distribution of organizations we propose a metric which determines the probability that given n random selections, with replacement, from the servers authoritative for z, the servers selected are not administered by the same organization $o \in O_z$:

$$AC_n(z) = 1 - \sum_{o \in O_z} \left(\frac{|\texttt{SLIST}_z^o|}{|\texttt{SLIST}_z|} \right)^n \tag{5}$$

where O_z denotes the set of organizations administering servers which are hosting zone data for z and $\texttt{SLIST}_z^o \subseteq \texttt{SLIST}_z$ denotes the subset of servers in $\texttt{SLIST}_z$ administered by organization $o \in O_z$.

For example, assuming the servers *ns.example.com* and *ns.example.net* are operated by two separate organizations and are the only authoritative servers for *example.com*, the administrative complexity of *example.com* with $n = 2$ is:

$$AC_2(\textit{example.com}) = 1 - \left(\left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2 \right) = 0.5$$

Managing administrative complexity is a matter of the owners of the authoritative DNS service of a zone. However, problems caused by inconsistent responses may also be mitigated by the validating resolver, using what we refer to as *validator diligence*. If validation of an RRset fails because of bogus or incomplete validation data received from an authoritative server, the resolver re-issues the query to other authoritative servers, attempting to complete the chain of trust. Such is the case with Internet Systems Consortium's (ISC) Berkeley Internet Name Domain (BIND) version 9.6 [11].

However, even if a resolver practices validation diligence, inconsistent response behavior results in reduced redundancy of hosted zones and more traffic to those servers serving legitimate responses. Additionally, if the resolver is behind one or more *proxy* resolvers to which it is configured to forward its requests, it is at the whim of the proxy resolvers, regardless of whether or not the upstream resolvers are configured for validation or have practiced validation diligence to obtain appropriate DNSSEC responses.

IV. Soft Anchoring

In Section III-B we discussed the effects of improper lateral coordination a zone, and mentioned validation diligence as one potential technique to mitigate the effect of inconsistent responses from authoritative servers. In this section we propose a mitigation technique the use of additional trust anchors by the resolver for each zone in an authentication chain to decrease the reliance of each ancestor "link" in the chain. We provide the reasoning behind and implementation of our proposal.

A. Extending the DNSSEC trust model

The authenticity of `DNSKEY`s in zones $z(j)$ through $z(a-1)$ is established by following the authentication chain to the trust anchor $z(a)$. The `DNSKEY` RRset for each zone is cached by validating resolvers until its TTL expires or its covering `RRSIG` expires—whichever occurs first [9]. Upon expiration it must be re-authenticated to the trust anchor at zone $z(a)$. However, `DNSKEY` values often remain static well beyond their expiration. Current recommended rollover practices suggest rollovers be performed on granularity of months or years, depending on algorithm, key size, and key function [12]. We therefore propose that validating resolvers install additional trust anchors at descendant zones, as `DNSKEY`s are authenticated through the established chain of trust.

If the resolver is configured with a trust anchor at each zone $z(i), i \in [j, a]$, then validation of each is self-contained. By installing additional anchors from authenticated `DNSKEY`s, the burden carried by ancestor zones to properly validate is eased, and each signed zone must only ensure the correctness of its own data. The result is that validation of zone z is unaffected by: zone- and anchor-class misconfigurations from zones $z(i), i \in (j, a]$; and delegation-class misconfigurations affecting delegations from zone $z(i)$ to $z(i-1)$, for $i \in (j, a]$.

B. Soft anchor management

To implement our trust anchor extension, we introduce the notion of *soft* anchors, in addition to traditional trust anchors, which we refer to as *hard* anchors. Hard and soft anchors are both used by resolvers for validation and are formally defined by the following rules of transitivity:

- A trust anchor installed on a resolver by a DNS administrator and verified out-of-band is a *hard anchor*.
- A trust anchor authenticated within the same `DNSKEY` RRset as a hard anchor is also a *hard anchor*.
- A trust anchor authenticated by establishing an authentication chain from a hard or soft anchor to a SEP in a descendant zone is a *soft anchor*.
- A trust anchor authenticated within the same `DNSKEY` RRset as a soft anchor is also a *soft anchor*.

Resolvers must maintain the source of each trust anchor in its repository as either a hard or soft anchor.

Hard anchors are maintained following procedures detailed in RFC 5011 [13]. Resolvers periodically poll the anchored zone for updates. A resolver adds a new trust anchor when a new `DNSKEY` RR with the SEP bit set is detected in the `DNSKEY` RRset of the anchored zone, after it has existed for longer than a specified *hold-down* time. Resolvers learn of `DNSKEY` revocation when they see a self-signed `DNSKEY` with the revoke bit set.

Our proposed procedures for managing soft anchors extend RFC 5011 for anchor addition and removal. A resolver periodically polls to re-authenticate the chain leading to the soft anchor. Soft anchors are added by a resolver when:

- the resolver detects a self-signed `DNSKEY` RR with the SEP bit set and which corresponds to an authenticated `DS` RR in the parent zone; or
- the resolver detects a new `DNSKEY` with the SEP bit set within a `DNSKEY` RRset already authenticated by an existing a soft anchor, and the `DNSKEY` persists for a hold-down period.

A hold-down period is only required in the second case because the existence of a `DS` RR in the first case suggests a legitimate addition by the zone administrator. The hold-down period specified by RFC 5011 is the greater of 30 days or the TTL of the `DNSKEY` RRset. However, soft-anchored zones don't have reason to implement RFC 5011 because they are linked via a chain of trust to their parent zone and don't anticipate being anchored by resolvers. For such zones with `DNSKEY` TTL less than 30 days (quite likely), a new key might be introduced and used to sign the `DNSKEY` RRset exclusively as a SEP before a hold-down period is complete. The resolver would be unable to add it as a soft anchor in that case (unless its `DS` RR was also detected). This would undermine the continuity needed to protect against a bad SEP key rollover, in which the new SEP `DNSKEY` is rolled successfully, but the corresponding `DS` RRs are not. Without a new soft anchor, a resolver cannot legitimately validate the `DNSKEY` RRset until the remainder of the 30 days has passed. To remedy this we suggest a hold-down period matching the lesser of the `DS` TTL and `DNSKEY` TTL, which is sufficient for the KSK rollover procedure outlined in RFC 4641 [12].

Soft anchors are removed from a resolver's repository when:

- a `DNSKEY` RR is revoked, following RFC 5011;
- the `DS` RR previously in existence for a soft anchor has been removed; or
- the chain of trust from hard to soft anchor has been securely unlinked, such that there is no longer a path to authenticate the soft anchor.

The first two items indicate explicit revocation either by RFC 5011 standards or by removing the link from its parent. The third item recognizes the intent of the administrator of a zone or its ancestor to prevent a path for validation; soft anchors were not necessarily intended by the zone administrators to be maintained by resolvers as trust anchors.

Validation is first attempted with the soft anchor in the closest ancestor zone. When validation with soft anchors at each ancestor zone have failed, then the hard anchor is used for validation. Thus, our approach is backwards compatible with plain DNSSEC.

Although the soft anchoring approach will decrease dependence on the DNSSEC correctness of ancestor zones in the chain of trust, it is only a mitigation technique. Most notably if a problem exists at zone $z(i), i \in [0, j]$, then the availability of zone z will still be affected because there is no further soft anchoring below $z(j)$. Additionally, we note that a resolver encountering a broken chain of trust without previously having authenticated a soft anchor through a chain from a hard anchor will be unable benefit from our approach.

Another consideration for soft anchoring is implementation. The computational and resource overhead required to maintain soft anchors is a function of the number of zones for which soft anchors are being maintained and the TTLs of the corresponding `DNSKEY` and `DS` RRsets. Such maintenance will likely require resources beyond those required by current implementations, and existing platforms may be insufficient for the maintenance of soft anchors for unlimited secure zones. To mitigate this concern, we suggest that implementations limit the number of zones for which soft anchors are maintained and that a least-recently-used policy be applied to select which are maintained.

V. Data Collection and Analysis

We performed a study of production DNS data using the metrics from this paper. Our seed data came from three sources: hostnames extracted from URLs indexed by the Open Directory Project (ODP) [14]; names queried to recursive resolvers at the 2008 International Conference for High Performance Computing, Networking, Storage, and Analysis (SC08) [15]; and names submitted via the Web interface of the DNSViz analysis tool [16].

From our seed data we selected a representative subset of production signed zones for analysis. With this objective, our data set includes fewer signed zones than other analyses [2], [3]. However, because names in our data set were either indexed by ODP, queried by clients at SC08, or submitted by interested parties, we justify our data set as a representative subset of production zones. We excluded zones whose names contained "test", "bogus", "bad", and "fail" or that were subdomains of known DNSSEC test namespaces (e.g., *dnsops.gov* and *dnsops.biz*, of the Secure Naming Infrastructure Pilot [17]). We further filtered zones by including only islands of security that had some public intent to be validated by resolvers—those with an authentication chain to the root zone trust anchor (after the July 2010 signing of the root [5]) or with an authentication chain to the trust anchor at ISC's *DNSSEC Look-aside Validation* (DLV) service [18]. DLV [19] was introduced to allow an arbitrary zone to be securely linked to a zone other than its hierarchical parent, for scalable validation prior to the signing of the root. We note that other DLV services exist [2], [3], but are populated with `DNSKEY`s discovered through DNSSEC polling, which means that users may not have explicitly opted in for production validation. We therefore consider only zones registered with ISC's DLV service as production signed zones.

Based on our qualifications for production signed zones, we considered 2,242 zones production. We polled the production signed zones every four hours for approximately five months, from June to November of 2010. Some zones were only present for part of the survey because they were added after

Production signed zones	2,242
Total errors resulting in non-zero failure potential	2,634
Errors resulting in possible failure ($0 < P_f^s(z) < 1.0$)	1,998 (76%)
Errors resulting in certain failure ($P_f^s(z) = 1.0$)	636 (24%)
Zone-class errors resulting in $P_f^s(z) = 1.0$	460 (72%)
Delegation-class errors resulting in $P_f^s(z) = 1.0$	176 (28%)
Errors resulting from misconfigured ancestor zones	178 (28%)

TABLE I
STATISTICS FOR THE DNS DATA COLLECTED FOR OUR ANALYSIS.

our polling began or because they were at some point unlinked from their parent zone, resulting in a non-production status.

Our analysis examined the authentication chain from each zone's SOA RR to the root or DLV trust anchor. Some zones had multiple paths to a trust anchor (i.e., due to multiple ancestral zones being registered with the DLV service). In such cases we optimistically selected the path with the most valid results. The results from our analysis are summarized in Table I.

A. DNSSEC misconfigurations

We identified validation problems lasting two or more consecutive polling periods (i.e., at least four hours) and classified them as either zone- or delegation-class misconfigurations. Each misconfiguration may account for multiple errors in our data if affected subdomains are also included in the analysis. Over duration of the survey we detected a total of 2,634 instances of non-zero failure potential, 24% of which resulted in certain failure. Of the errors resulting in certain failure, 28% were caused by delegation-class misconfiguration or misconfiguration in an ancestral zone. These 28% would have been resolvable had a soft anchoring system been in use by resolvers, such as that proposed in Section IV of this paper.

For each zone we averaged the failure potential calculated at each poll during our survey. The resulting average and maximum for each production signed zone is displayed as a cumulative distribution function (CDF) in Figure 4. The average failure potential was zero for nearly 83% of the production signed zones. However, 17% had a failure potential of 1 for at last four hours during our survey. Surprisingly, almost 2% of the production signed zones were completely unresolvable due to misconfigurations that persisted through every poll we made to them.

Our analysis of failure potential considers only production signed zones whose ancestry is linked to the root zone or ISC DLV; this doesn't account for unsigned zones that may be affected by misconfigurations of signed zones at higher levels. For example, the expiration of RRSIGs in the *be* zone occurred during our analysis period and affected the authenticity of its NSEC RRs for insecure delegations, such as *sw.be*, which resulted in bogus responses for *sw.be*. However, such were not included in our analysis. We also analyzed insecure delegation by signed zones by testing for proper use of NSEC RRs by authoritative servers. We identified 36 signed zones for which one or more servers failed to return NSEC responses for authenticated denial of existence.

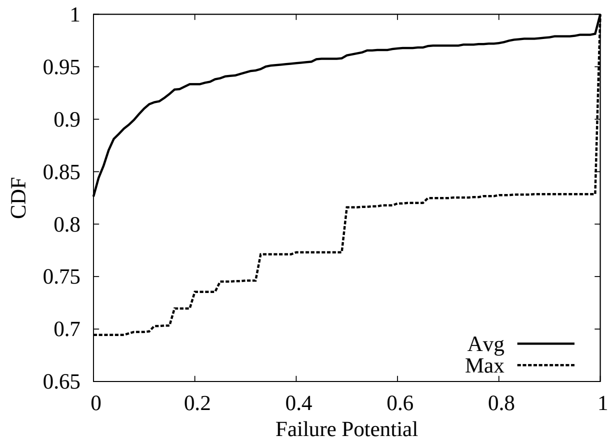

Fig. 4. The CDF describing the average and maximum failure potential for production signed zones during our analysis period.

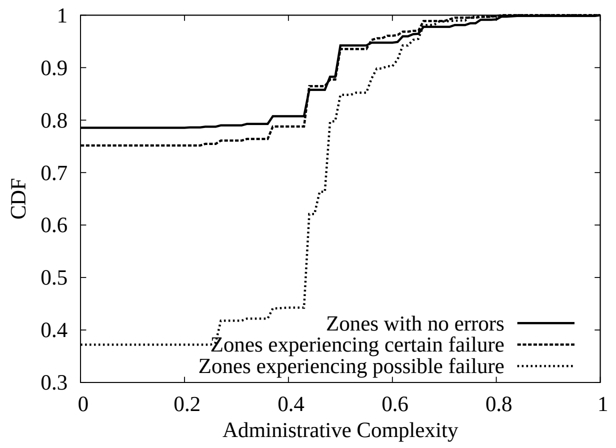

Fig. 5. The administrative complexity of production signed zones with no errors during our survey and those experiencing certain or possible validation failure.

B. Administrative complexity

We analyzed the administrative complexity for each production signed zone to see how it correlated with failure potential. We used the email suffix of the RNAME field in the SOA RR corresponding to the name of each server authoritative for a zone to estimate administering organization. The results of our analysis of administrative complexity with $n = 2$ is represented as a CDF in Figure 5.

The graph contrasts the administrative complexity for zones with no errors during our survey with that of zones that experienced certain or possible failure, at the time that the error was detected. Nearly 80% of the zones with no errors and 75% of those that experienced certain failure potential had zero administrative complexity. However, of the zones that experienced possible failure only 38% had zero administrative complexity. This supports our claim that administrative complexity can result in increased failure potential if not properly managed.

The large vertical jump at 0.5 reflects a common case where just two servers are authoritative for a zone, each administered by a different organization. In such cases it is equally likely to query a server administered by either of the two organizations and hence have a different view of zone data, depending on configuration and data consistency.

VI. Related Work

Other studies have been performed to quantify DNSSEC deployment in terms of pervasiveness, availability, and quality. One such project, run by IKS Jena [2], maintains an ongoing status of DNSSEC signed zones. SecSpider [3], [20], discovers signed zones through several means: as discovered by a Web search engine; by user submission to the SecSpider Web interface; and by traversing NSEC RRs on known signed domains. It has for several years polled these zones from different worldwide locations, verifying consistency of results from different vantage points. Zone data collected from SecSpider was used perform an assessment of availability, verifiability, and validity of DNSSEC deployments [4].

Our objectives are similar to those of previous analyses, but our approach differs. In this paper we focus on consistency of behavior across authoritative servers from a particular vantage point (under the assumption that our path is reliable), rather than consistency of experience querying a zone from different client locations. SecSpider declares a zone DNSSEC-enabled only if all authoritative servers for a zone serve DNSSEC-type RRs [20]. However, our analysis is based on actual resolver behavior when configured with a valid trust anchor: unless an chain of trust extends from an anchor at the resolver, DNSSEC deployments are irrelevant from the perspective of a resolver.

Other solutions for trust anchor distribution have been proposed, although having a different overall objective. SecSpider, and its sister project, Vantages [21], [22], collect DNSKEYs through polling and collaborative sharing over peer-to-peer networks. DLV services, provided by ISC and others [2], [3], [18], assist with reliable distribution of trust anchors for zones that would otherwise remain islands of security (i.e., no authentication path to the root zone). However, the objective and methodology of our soft anchoring approach in Section IV is to strengthen chains of trust that already exist for zones that have "opted in" to DNSSEC validation by publishing DS or DLV RRs.

VII. Conclusion

DNS is an essential component of the Internet's architecture. DNSSEC deployment is under way to protect its integrity, but the additional complexity of DNSSEC challenges the availability of DNS. Various DNSSEC-related server and zone misconfigurations can affect not only the corresponding zones, but also the entire namespace below.

In this paper we have reviewed the DNS protocol and its security extensions. We have presented a metric for assessing DNSSEC availability by calculating failure potential, based on the behavior of validating resolvers. Our analysis considers administrative complexity, which contributes to failure potential, and the results of our empirical analysis support our assertion that it is a factor. Our proposal of a soft anchoring system to help maintain availability of signed zones in the presence of misconfiguration of ancestor zones. Our survey showed that 28% of the errors resulting in certain failure might have been mitigated by the soft anchoring system we described. This technique will be particularly helpful in the case of errors at the highest levels, such as that experienced by *be* in October 2010.

As the deployment of DNSSEC continues, a thorough understanding of the protocol accompanied by proper configuration will allow it to be successful. A system such as soft anchoring will mitigate the impact of misconfigurations resulting from its early deployment. In addition, administrators should be acquainted with the impact associated with misconfigurations.

References

[1] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "RFC 4033: DNS security introduction and requirements," 2005.

[2] IKS, "DNSSEC." [Online]. Available: https://www.iks-jena.de/leistungen/dnssec.php

[3] "SecSpider." [Online]. Available: http://secspider.cs.ucla.edu/

[4] E. Osterweil, M. Ryan, D. Massey, and L. Zhang, "Quantifying the operational status of the DNSSEC deployment," in *Proceedings of the 6th ACM/USENIX Internet Measurement Conference (IMC'08)*, October 2008.

[5] "Root DNSSEC." [Online]. Available: http://www.root-dnssec.org/

[6] P. Mockapetris, "RFC 1034: Domain names - concepts and facilities," 1987.

[7] ——, "RFC 1035: domain names - implementation and specification," 1987.

[8] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "RFC 4034: Resource records for the DNS security extensions," 2005.

[9] ——, "RFC 4035: Protocol modifications for the DNS security extensions," 2005.

[10] R. Elz and R. Bush, "RFC 2181 - clarifications to the DNS specification," 1997.

[11] ISC BIND. [Online]. Available: http://www.isc.org/products/BIND/

[12] O. Kolkman and R. Gieben, "RFC 4641: DNSSEC operational practices," 2006.

[13] M. StJohns, "RFC 5011: Automated updates of DNS security (DNSSEC) trust anchors," 2007.

[14] Open Directory Project. [Online]. Available: http://www.dmoz.org/

[15] SC08: The International Conference for High-performance Computing, Networking, Storage and Analysis. [Online]. Available: http://sc08.supercomputing.org/

[16] Sandia National Laboratories, "DNSViz." [Online]. Available: http://dnsviz.net/

[17] National Institude of Standards and Technology, "Secure naming infrastructure pilot." [Online]. Available: http://www.dnsops.gov/

[18] Internet Systems Consortium, "DNSSEC look-aside validation registry." [Online]. Available: https://dlv.isc.org/

[19] S. Weiler, "RFC 5074: DNSSEC lookaside validation," 2007.

[20] E. Osterweil, D. Massey, and L. Zhang, "Deploying and monitoring DNS security (DNSSEC)," in *25th Annual Computer Security Applications Conference (ACSAC '09)*, December 2009.

[21] E. Osterweil and L. Zhang, "Interadministrative challenges in managing DNSKEYs," *Security and Privacy Magazine: Securing the Domain Name System*, September 2009.

[22] "Vantages." [Online]. Available: http://www.vantage-points.org/