A Case for Comprehensive DNSSEC Monitoring and Analysis Tools

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 Security Extensions (DNSSEC) add an element of authentication to the DNS, which is a foundational component of today's Internet. However, the complexity involved in maintaining a DNSSEC deployment is significantly more than that of its insecure counterpart, and there are more places where problems can occur. Our analysis shows that errors in DNSSEC configuration have been pervasive and have affected DNS namespace at even the highest levels. Effective tools are necessary to recognize, diagnose, and help correct such errors both for administrators of authoritative name servers and of validating resolvers. In this paper we identify some of the issues impacting DNSSEC deployments, analyze their pervasiveness in production DNS, and describe tool functionality that can help administrators understand, maintain, and troubleshoot DNSSEC deployments at both the validating resolvers and authoritative servers.**

I. Introduction

The Domain Name System (DNS) [1], [2] is one of the foundational components of the Internet. The DNS is a distributed database for looking up data based on domain name and query type. The most common use is for mapping domain names (e.g., *www.example.com*) to Internet addresses (e.g., 192.0.2.29).

The DNS Security Extensions [3]–[5] were designed to protect the integrity of DNS responses. DNSSEC allows DNS administrators to cryptographically sign and validate DNS data. Although DNSSEC deployment is still relatively low, the number of DNSSEC-signed zones has increased significantly in the last year, including a significant number of top-level domains (TLDs) [6]–[8]. In July, 2010, a milestone was reached with the signing of the root zone [9].

Despite its security benefits, DNSSEC adds non-trivial complexity to the DNS and increases the chances of DNS outage if not properly deployed or maintained. The effects of misconfiguration have been felt at various levels in the DNS hierarchy, including TLDs, and even the root zone. Essential for proper deployment is an understanding of DNSSEC components, their relationship, and the protocol itself. Effective tools are necessary to accomplish that role and to facilitate troubleshooting and monitoring of DNSSEC deployments.

In this paper we provide a review of DNS and its security extensions, and we identify common scenarios affecting validation of DNSSEC deployments. We present a survey of DNSSEC deployment resulting from periodic polling over an extended time frame and use the results to suggest tool functionality to improve the quality of DNSSEC deployment. We list the following as the major contributions of this paper:

- A study of common configuration problems hindering DNSSEC deployment and their pervasiveness in production DNS.
- Suggestions of tool functionality to minimize configuration problems related to DNSSEC deployment.

Section II contains a review of the fundamentals of DNS and DNSSEC. In Section III we identify common DNSSEC pitfalls that affect validation. In Section IV we describe our survey of DNSSEC deployment and analyze the results. In Section V we suggest tool functionality that would improving DNSSEC deployment by minimizing the chance for misconfiguration. We conclude in Section VI.

II. DNS Background

The Domain Name System (DNS) [1], [2] is a distributed system for resolving Internet names. A DNS *zone* is an autonomously managed piece of namespace, and a set of servers is designated as *authoritative* for each zone. Administration of DNS namespace is *delegated* to different organizations beginning at the *root* zone, which is the top of the namespace hierarchy.

A. Name resolution

In the DNS a *resolver* directs queries to *authoritative servers* to obtain answers. It initially directs a query to one of the servers authoritative for the root zone, which responds with a referral to the appropriate top-level domain (TLD) servers. The resolver then re-issues the query to one of those servers. This downward referral process continues until a response is received from a server authoritative for the zone of the domain name being looked up.

DNS questions and answers are comprised of *resource records* (RRs). Each RR 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`-type RR). A *resource record set* (RRset) is a set of RRs having the same name and type. The `NS` (name server) RRset for a zone is used to designate the names of the servers authoritative for a zone.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

RRsets may be cached by resolvers for the duration of their TTL value.

B. Security extensions

The DNS Security Extensions (DNSSEC) [3]–[5] provide a mechanism for authenticating DNS responses. Each RRset in a zone is signed by a private key, and each resulting signature is included in the record data of an `RRSIG`-type RR, with the same name as the RRset it covers. The corresponding public key is included in the zone data using a `DNSKEY`-type RR with the same name as the zone (e.g., *example.com*). The record data for `RRSIG` RRs includes references to the `DNSKEY` needed to validate the signature. Each `RRSIG` has a limited lifetime, specified by inception and expiration dates in its record data. Any `RRSIG` covering an RRset are included in the response to a DNSSEC query, so the validating resolver (hereafter referred to as simply *validator*) may verify the integrity of the RRset using the appropriate `DNSKEY`.

A validator may only authenticate an RRset with a `DNSKEY` RR that it deems to be authentic. The validator is initially seeded with a *trust anchor* which corresponds to a key that has signed the `DNSKEY` RRset for a zone (i.e., is *self-signing*). This `DNSKEY` provides a *secure entry point* (SEP) into the zone. Having authenticated the `DNSKEY` RRset using the appropriate `RRSIG` and the SEP `DNSKEY`, the validator can then validate `RRSIG`s made by any `DNSKEY` in the `DNSKEY` RRset.

DNSSEC scales by establishing an authentication chain upwards through the namespace hierarchy, so validator may anchor with the `DNSKEY` of a common ancestor zone, typically the root. This *chain of trust* makes it unnecessary for a validator to maintain a trust anchor for every signed zone. The link between zones is accomplished by the introduction of `DS` (*delegation signer*) RRs in the parent zone. The record data of a `DS` RR includes the cryptographic digest of a `DNSKEY` RR in the child zone of the same name. Delegation referrals include the appropriate `DS` RRset, in addition to the `NS` RRset for the child zone, signaling to the validator that the child zone is signed. The `DS` RRset is signed by the parent zone, so a validator can verify its legitimacy. An *island of security* is a chain of trust comprised of one or more zones for which there is no SEP.

A common setup is for a zone to sign only its `DNSKEY` RRset with the SEP key (a *key signing key* or KSK) and sign other zone data with a second key (a *zone signing key* or ZSK). In much of this paper, the implementation of a ZSK/KSK split is abstracted, except as necessary for discussion.

Authenticated denial of existence is accomplished using `NSEC` RRs, which are provided in a response to show a validator where the non-existent RRset would appear (in a canonical ordering of the zone) if it did exist. It has particular relevance in this paper for addressing *insecure delegations*. When there is no SEP into a child zone, the parent zone must effectively prove the end of the chain of trust by showing that there are no `DS` RRs for the child zone using `NSEC` RRs.

Hashed authenticated denial of existence using `NSEC3` RRs was introduced to address challenges introduced by `NSEC` RRs [10]. `NSEC3` provides the same functionality as `NSEC` RRs, except that non-existence of RRsets is demonstrated with an ordering of cryptographic hashes of owner names, rather than clear text owner names. For the purposes of this paper, we use only `NSEC` to represent authenticated denial of existence, unless noted otherwise.

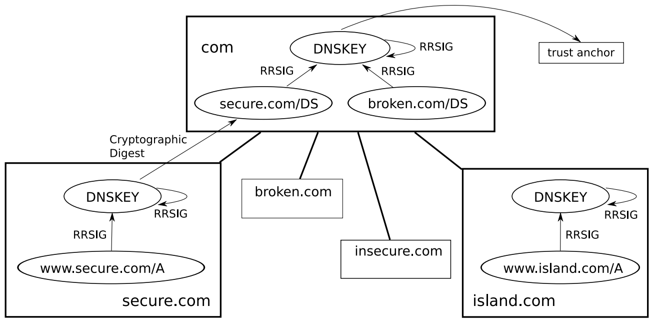

Fig. 1. The DNSSEC authentication chain for several fictitious zones. `RRSIG`s are represented by upward arrows extending from the RRset they cover to the `DNSKEY` which can validate it. SEP `DNSKEY`s are mapped to their corresponding trust anchor or `DS` RR with an arrow. Self-signatures at each SEP `DNSKEY` are represented by a self-directed edge.

Figure 1 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.

C. DNSSEC maintenance

A zone signed with DNSSEC requires more careful maintenance than an unsigned zone. Since `RRSIG`s have a limited lifetime, the RRsets they cover must be periodically re-signed to replace `RRSIG`s that would otherwise go stale.

While `DNSKEY`s technically do not expire, it is recommended that they periodically be replaced to prevent prolonged exposure, which may make them the target of cryptanalysis attack. Such replacement is called a *key rollover*. Current best practices for key rollovers are documented in RFC 4641 [11]. Non-SEP `DNSKEY`s (i.e., with no association with a trust anchor and having no `DS` RR in the parent zone) can be rolled without involving third-parties and are thus self-contained. However, when a SEP `DNSKEY` is rolled, the parent zone must be involved to handle the change in `DS` RRs. Likewise, a validator must be engaged when a configured trust anchor is rolled [12].

III. DNSSEC PITFALLS

In this section we describe some of the deployment and maintenance pitfalls that hinder DNSSEC validation. We begin by describing the possible outcomes resulting from DNSSEC validation and then list some of the causes of failed validation.

A. DNSSEC validation

RFC 4035 defines several possible outcomes resulting from a validation attempt [5][1]. Our examples refer to Figure 1 and

[1]Note that we omit the *indeterminate* status, which is among those listed in RFC 4035, because from a practical standpoint it results in a bogus response.

assume that a validator is anchored with the *com* DNSKEY.

- *Secure*: The validator establishes an unbroken chain of trust between the RRset and a trust anchor. Example: a properly authenticated RRset in the *secure.com* zone is deemed *secure*.
- *Insecure*: The validator has securely proven that the chain of trust from its anchor terminates before reaching the RRset, so there is no trusted path to authenticate the RRset. Example: assuming the NSEC RRs returned in response to a query for DS RRs for *insecure.com* and *island.com* are properly authenticated and sufficient to prove that the delegation is insecure, then responses for those zones are *insecure*.
- *Bogus*: The validator is unable to form a chain of trust between the RRset and a trust anchor and is unable to securely show that no such chain should exist. Example: an expired RRSIG covering an RRset in the *secure.com* results a *bogus* response; likewise, the presence of a DS for the *broken.com* zone, in which there are no DNSKEYs present, results in a *bogus* status for any RRset in the zone.

While a secure validation is ideal, an insecure outcome is also usable and is equivalent to normal, unauthenticated name resolution. However, a bogus outcome is an indicator that validation failed—an alarm that DNS data has been tampered with. The response returned to a *recursive* (i.e., on behalf of another client) query which a validator renders bogus has SERVFAIL error status and contains no DNS data, an indication of general name resolution failure.

Data tampering is not the only cause of a broken chain of trust resulting in bogus validation. Improper setup or maintenance of a DNSSEC deployment may also result in bogus responses. However, an end user cannot distinguish between bogus response due to data tampering and bogus response caused by misconfiguration. In either case, a name cannot be resolved. This study focuses on bogus validation caused by misconfiguration.

B. Bogus validation caused by misconfiguration

In this section we describe some of the causes of validation failures from a perspective of misconfiguration. In any case, an RRset deemed bogus also invalidates any dependent RRsets. For example, a bogus DNSKEY RRset means that none of the RRsets whose RRSIGs would be potentially validated by those DNSKEYs are also bogus.

a) Expired RRSIGs: If an RRSIG is allowed to expire, then the RRset it covers is rendered bogus. A more subtle situation is that in which an RRSIGs is not refreshed within one TTL from its expiration. A validating resolver that authenticates the covered RRset will set the TTL to the minimum of the RRset's TTL, the RRSIG's TTL, and the difference of the RRSIG's expiration and the current time, so there are no concerns with caching beyond the expiration date [5]. However, a non-validating resolver, having no notion of the authenticity of the RRset, will cache it until the TTL expires, even beyond the expiration of the RRSIG. This is problematic in the case where a validating resolver must rely on the cache of a non-validating resolver.

b) Bogus signatures: The signature in the record data of an RRSIG must validate against the RRset it covers, or it is invalid. This might happen, for example, if a DNS-based load balancer sends different A RRsets for requests from different locations but neglects make the RRSIGs match the RRsets in the responses.

c) DS/SEP inconsistency: If DS RRs are present in a parent zone but none correspond to any self-signing DNSKEYs (i.e., SEPs) in the child zone, the result is a bogus delegation, and RRsets in the child zone and below are deemed bogus. Also, DNSKEYs that have been *revoked* may not act as a SEP [12]. Such DS inconsistencies are often caused by a bad KSK rollover (i.e., the DS RRs still reference the previous SEP DNSKEY, which no longer has a signing role). They may also result from incorrectly publishing the DS prior to publishing the signed version of the zone, or prematurely unsigning the zone before their removal.

d) Missing NSEC RRs: If an authoritative server does not provide the appropriate NSEC RRs in a referral or DS query to show that no DS RRs exist for a child zone, then the chain of trust is broken for an insecure delegation, and RRsets in the child zone and below are bogus, regardless of whether or not they are signed (i.e., islands of security). This may be because the zone was not signed properly and does not contain the necessary records or because the server does not have sufficient support for the method of authenticated denial of existence (NSEC or NSEC3) employed in the zone.

e) Missing RRSIG RRs: If an authoritative server does not provide the RRSIGs necessary to complete a chain of trust for a given RRset, then the chain of trust is broken, and the result is a bogus validation. This includes DS, DNSKEY, and NSEC RRsets upon which the RRset is dependent. RRSIGs may be missing because they are erroneously missing from the zone itself or because the authoritative server does not implement DNSSEC and therefore does not include RRSIGs with the RRsets they cover.

f) Missing DNSKEY RRs: If a DNSKEY referenced in an RRSIG is necessary to complete a chain of trust, but not included in the DNSKEY RRset, then the result is a bogus validation. This may occur if the server does not support the DNSKEY RR type and responds with a NOERROR response having no data. It may also happen if the authoritative server is running an inconsistent version of the zone with a DNSKEY RRset that doesn't include the DNSKEY in question.

C. Server consistency

While a valid chain of trust for an RRset may exist, it may not be exempt from the failure-inducing behaviors listed in the previous section if responses returned by authoritative servers are inconsistent. It is possible for a validator to receive one valid and one invalid response given two identical queries to different servers authoritative for the same zone. Inconsistency among authoritative servers can typically be attributed to one of two things: an outdated or otherwise incorrect version of the zone data served by a server; or a level of DNSSEC support inconsistent with the signed zone they are serving.

It is critical that servers authoritative for a zone all have the most current version of the zone. In ordinary DNS, synchronization problems may go unnoticed for long periods of time with few ill effects. However, the time sensitivity of a signed zone requires freshness to avoid expired `RRSIG`s and inconsistent `DNSKEY` RRsets in the wake of key rollovers. If the master server on which the zone is maintained is not configured to notify other authoritative servers of new versions, or if there are other network configurations (e.g., firewall rules) inhibiting the servers from downloading the most current version of the zone, then propagation of the zone is delayed.

Even with proper notification and clear transfer paths, administrators must be sure that a newly signed zone has its *serial* incremented after changes to signal secondary servers that they should download the new version. The `dnssec-signzone` utility distributed with the ISC BIND (version 9.6) software [13] signs (or re-signs) a zone, but by default it does not increment the serial. Improper use of this utility by administrators could lead to server inconsistency.

With regard to DNSSEC support, a server with no notion of DNSSEC will not respond with the appropriate RRs required for DNSSEC, such as `RRSIG` or `NSEC`. If a zone is signed using `NSEC3`, authoritative servers must understand how to return `NSEC3` RRs appropriately. For example, although DNSSEC was implemented in ISC BIND beginning with version 9.3, `NSEC3` support was not implemented until version 9.6.

IV. DNSSEC Deployment Survey and Analysis

Our survey of DNSSEC data consisted of periodic polling of production DNS zones signed with DNSSEC over a timespan of more than five months—June to November, 2010. We analyzed each signed zone six times daily, querying each authoritative server to elicit various DNSSEC-related responses. Our zones 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].

We identified production signed zones in our data set by considering only zones indicating their 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 [9]) or with an authentication chain to the trust anchor at ISC's *DNSSEC Look-aside Validation* (DLV) service [17]. DLV [18] was introduced to allow an arbitrary zone to be securely linked to a zone other than its hierarchical parent for trust anchor scalability prior to the root signing. We note that other DLV services exist [6], [7], but are comprised of `DNSKEY`s discovered through DNSSEC polling rather than explicit opt-in by administrators, so we didn't necessarily consider islands linked to such services as production.

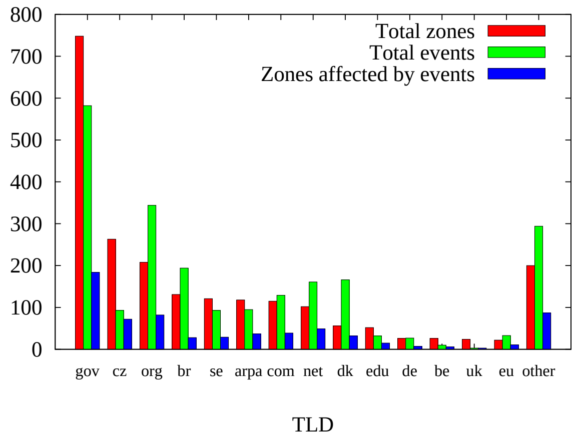

Fig. 2. TLD breakdown of production signed zones analyzed, total failure events, and zones that experienced failure events identified during our polling period.

To further avoid zones set up for non-production testing we excluded zones containing the names "test", "bogus", "bad", and "fail", and those that were subdomains of known DNSSEC test namespaces (e.g., *dnsops.gov* and *dnsops.biz*, of the Secure Naming Infrastructure Pilot [19]). The total number of production signed zones analyzed was 2,242, though the total number analyzed during any given polling period varied as new zones were added or as monitored zones entered or left production DNSSEC. For example, the total number of production signed zones analyzed in our first polling period was 1,276 contrasted with 2,083 at our conclusion.

The breakdown of analyzed zones by TLD is shown in Figure 2. For signed zones under the *gov* TLD, which made up the largest contingency of those analyzed, one in four zones experienced misconfiguration. An even larger percentage of zones were affected in the *cz* TLD, which had the next highest representation in our analysis.

A. Failure events

We grouped configuration errors leading to certain or possible (i.e., due to server inconsistency) validation failure of the `SOA` (start of authority) RRset for the zone, using the following categories:

- `RRSIG` - invalid inception or expiration date
- Missing `RRSIG`
- `RRSIG` - bogus signature
- Missing `DNSKEY`
- `DS`/SEP mismatch

Within each category, we identified an *event* as an occurrence of the problem lasting two or more consecutive periods (i.e., at least four hours), which resulted in 2,257 events over our polling period. However, it is possible that multiple events from different categories contributed to a single outage, such as expired `RRSIG`s on a `DNSKEY` already suffering from a `DS` mismatch. The breakdown of events by the TLD of the zone in which they occur is shown in Figure 2 aside the total

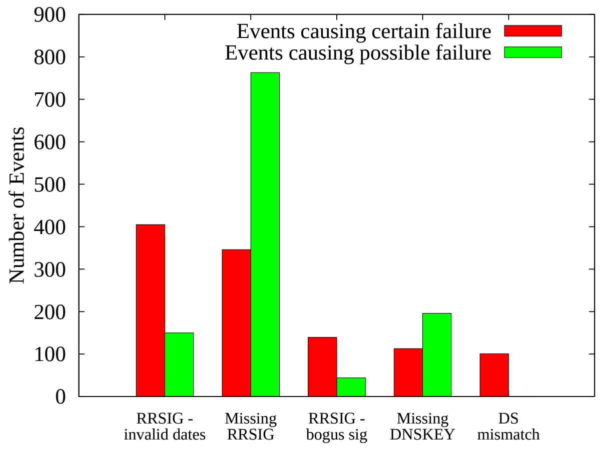

Fig. 3. DNSSEC failure events categorized by type. For each type, the total number of causing certain or possible failure is graphed.

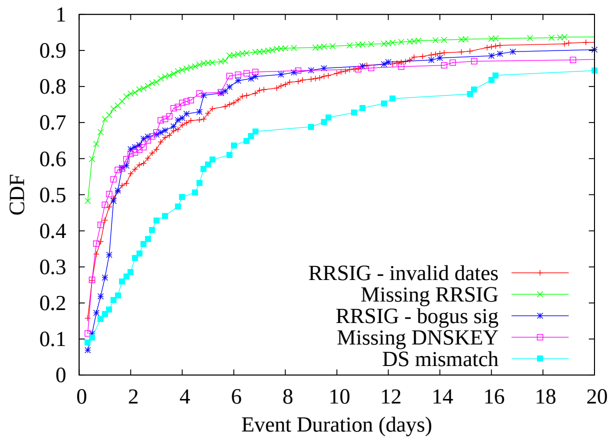

Fig. 4. CDF for the lifetime of DNSSEC failure events, which includes the time to detect and take corrective action for an event.

Event type	Total events	Certain failure	Avg lifetime (days)	Repetition rate
RRSIG - invalid dates	555	405	6.5	27%
Missing RRSIG	1,109	346	5.0	43%
RRSIG - bogus sig	183	139	7.3	43%
Missing DNSKEY	309	113	12.0	38%
DS mismatch	101	101	14.2	7%

TABLE I
A SUMMARY OF DNSSEC-RELATED FAILURE EVENTS, BY TYPE.

number of production signed zones analyzed and number of zones affected by failure events within that TLD.

Although the effects of an event at some domain would be felt by subdomains as well, in this study we only consider misconfigurations within the zone in which they originate. As an example, expired RRSIGs in the *com* zone are not counted in the *example.com* zone as well. Also, we note that some occurrences of DNSSEC misconfiguration are inconsequential and do not result in the possibility of validation failure (although they may, if not corrected). For example, if an RRSIG covering the DNSKEY RRset is made by a non-SEP key (i.e., ZSK), then its validity is irrelevant because it is authenticated by the SEP (i.e., KSK). Such are not considered events in our analysis.

Figure 3 graphs the total events for each category causing both certain and possible failure (i.e., due to server inconsistency), also summarized in Table I. The largest contributor to certain DNSSEC validation failure is RRSIGs with invalid dates. This typically refers to expired RRSIGs, though it may also include RRSIGs published before their inception dates. This mode of failure requires the least administrator intervention to achieve; it occurs with the negligence to sign the zone. Two TLDs were affected by this for a duration long enough to classify it as an event.

Missing RRSIGs are the second highest contributor to certain failure and the leading cause of possible failure. We attribute the latter to zones that were signed, but for which one or more authoritative servers do not support DNSSEC and therefore do not return RRSIGs appropriately. Seven TLDs and the root zone exhibited events in which a fraction of servers did not appropriately return RRSIGs.

Inconsistent DS RRs contributed the lowest number of events, but five such events were associated with TLDs. However, it appears that in each of these cases, the delegation issues were related to an unclean transition from registration with ISC's DLV registry to native hierarchical authentication using DS RRs in the root. The DLV registry was not updated before the SEP DNSKEYs referenced in the DLV were rolled, leaving dangling DLV RRs for the zones in question, even though the delegation from the root was properly configured. Because these events occurred after the signing of the root zone, only clients using *only* the DLV trust anchors would have felt these five events.

B. Event duration

We analyzed the duration of each event to understand the responsiveness of DNS administrators in identifying and taking corrective action for a DNSSEC-related problem. In some cases events were corrected by removing their path to the trusted anchor (i.e., removing it from production status), and in some cases the problem was fixed only on some authoritative servers before it was resolved completely. Figure 4 plots the lifetime of each event as a cumulative distribution function (CDF), and the averages are show in Table I. Note that we exclude from our analysis of event duration 215 events which were not corrected prior to the conclusion of our polling period, the first occurrence of which ranged from the date of our first poll to the date of our final poll.

On average, events involving missing RRSIGs were corrected within the least amount of time, followed by RRSIGs with invalid dates and RRSIGs with bogus signatures. This is not surprising since the remedy for each of these is simply re-signing the of the zone which replaces the affected RRSIGs.

Corrections to DS RRs referencing invalid SEPs had the largest average correction time, and about 30% of such events took more than 10 days to be corrected.

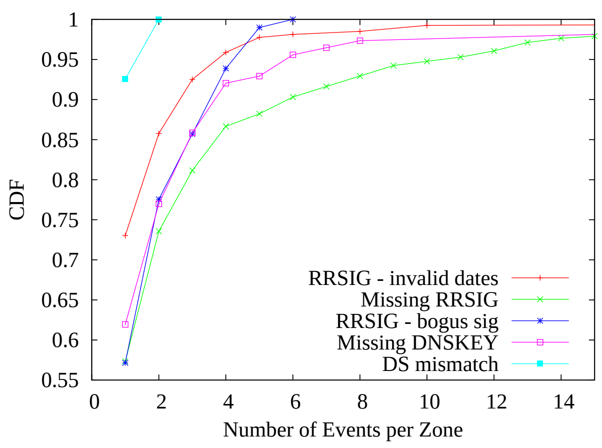

Fig. 5. Number of events per zone, by type.

C. *Repeated misconfiguration*

Figure 5 shows the number of events per zone, classified by type, and Table I includes the repetition rate for different misconfigurations.

DS mismatches were the misconfiguration repeated the least by DNS zones at 7%. This could be because KSK rollovers generally happen less frequently than RRSIG renewals [11], so there is less opportunity for error. Next to least was the repeated occurrence of RRSIGs with invalid dates, which happened on nearly 15% of affected zones. Almost 20% of zones affected by missing (completely or on a fraction of servers) RRSIGs experienced this misconfiguration at least three times.

V. DNSSEC Tools

Overall, the results from our survey are quite alarming. Events in each category totaled more than 100 for the polling period, and zones in nearly every TLD were represented. The facts that a fraction of 10 to 15% of events in different categories were not resolved within 20 days and misconfigurations as obvious as RRSIG expirations were repeated by nearly 15% of offenders are significant. These observations lead us to conclude that administrators are unaware of the implications of improper maintenance of DNSSEC deployments, an insignificant number of users are validating and creating a stir when validation fails, and/or insufficient tool functionality is available to assist administrators in identifying problems in their deployments.

We propose in this section tool functionality for assisting with effective DNSSEC deployment. We highlight the DNSViz visualization tool [16] as a work-in-progress tool which we've developed to achieve some of this functionality, and we additionally make mention of other tools.

A. *Analysis tools*

Because relationships in DNSSEC are both complex and critical, it is important for a DNS administrator—both on the validation end and the authoritative end—to have effective

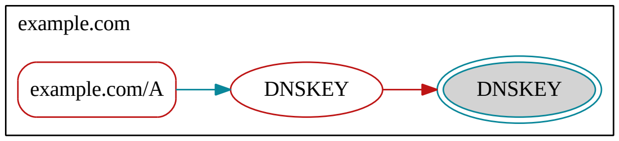

Fig. 6. DNSViz output for an RRSIG in the authentication chain for the A RRset of *example.com*.

tools to aid in analysis and troubleshooting. An effective tool automates the tedious and computationally complex portions of analysis, such as mapping the key tag of DS and RRSIG RRs to DNSKEY RRs and verifying RRSIG signatures. It also demonstrates relationships between various components, such as DNSKEYs, RRSIGs, and the RRsets they cover. This allows an administrator to isolate a problem area, even if it exists somewhere in the zone ancestry of an affected RRset. It should be comprehensive enough to identify problems with inconsistencies among authoritative servers.

DNSViz implements this functionality via a Web interface [16]. Relationships between DNSSEC components are represented as a directed graph, with edges representing RRSIGs and cryptographic digests in DS RRs. Attention is drawn to bogus and expired RRSIGs, which are identified by their color. Nodes (DNSSEC components, such as DNSKEYs and signed RRsets) below a break in the chain of trust are colored red to indicate their bogus status, just as nodes categorized as secure are colored blue. This allows a user to both identify the source of the problem and see its effects. An example is shown in Figure 6, which shows a break in the chain of trust between the KSK for *example.com* (trust anchor represented with a double border) and the A RRset for *example.com*. Because the RRSIG made by the KSK is bogus, the A RRset is also bogus, even though its RRSIG made by the ZSK is valid. In DNSViz, multiple edges extend from nodes to represent inconsistencies across authoritative servers.

What DNSViz currently lacks is the ability to analyze DNSSEC deployments from different vantage points, including querying different instances of anycast nodes. This is important to the administrator troubleshooting the resolution failure returned by his or her own validating resolvers. Especially useful would be the ability to analyze the authentication chain as contained in the cache of a validating resolver itself. Although its objectives differ, SecSpider [7], [20], another DNSSEC project, polls zones from different world-wide locations and verifies consistency of results from different vantage points.

Other online tools exist, some of which follow the authentication chain from RRset to trust anchor [21], while others examine only the zone itself [22], [23]. Relationships in all these are represented textually, as opposed to graphically.

B. *DNSSEC monitoring*

Also important for DNS administrators of validating resolvers and authoritative servers is monitoring functionality.

DNSViz regularly monitors DNSSEC-signed zones, but there currently is no interface to see an executive summary of DNSSEC deployment and problems, nor is there an automated alert mechanism. The former functionality could serve as a central "weather map" for administrators experiencing validation problems to get a sanity check; the latter could alert subscribed administrators when errors or anomalies are detected.

Other sites which monitor DNSSEC deployment are SecSpider [7], [20] and IKS Jena (Information, Communication, and Systems) [6], which both maintain an ongoing status of DNSSEC signed zones on their Web sites. The SecSpider site contains information about the pervasiveness of deployment in general, and the IKS Jena site summarizes deployment issues, such as the status of `DS` RRs. The DNS Operation, Analysis, and Research Center (DNS-OARC) also maintains a site which monitors the TLDs for RRSIG expiration and other general DNS configuration [24].

C. Interactive maintenance

While DNSSEC operational practices are outlined in several Internet standards [11], [12], measurements detailed in this paper have shown that errors due to misconfiguration are pervasive. One practice that may help is the integration of monitoring into signing and maintenance tools. For example, one of the variables associated with a key rollover is propagation time—the time for a zone to propagate to all servers. While this variable may be estimated, the only way to truly know if a change has propagated is with active probing. If an authoritative server reaches an unexpected state and is unreachable or is unable to obtain zone updates, then the propagation time allocated by the signer is completely arbitrary. If this happens, then a new `DNSKEY` introduced may not be available on all servers, which could result in a broken chain of trust. If the signer is able to probe servers to verify that propagation has completed, then it can with better assurance perform DNSSEC maintenance duties.

VI. Conclusion

DNS is an essential component of the Internet. DNSSEC was designed to protect the integrity of DNS responses, but its deployment has been wrought with challenges. In this paper we have presented an analysis of DNSSEC deployment based on a survey spanning five months. We identified and analyzed configuration errors affecting the validation of DNSSEC. We categorized the errors and analyzed the time taken to correct them. We found that misconfiguration affected DNSSEC deployment at even the highest levels, and on average took between five and ten days to correct, depending on the type of misconfiguration. Nearly 20% of zones in which `RRSIG`s were allowed to expire experienced such expirations three or more times.

If DNSSEC deployment is to be successful, then the challenges posed by its administrative and protocol complexity must be met. We have presented three suggestions for tool functionality to improve DNSSEC deployment experience: analysis, monitoring, and interactive maintenance. We believe that such tools will better enable administrators to understand, maintain, and troubleshoot DNSSEC deployments.

References

[1] P. Mockapetris, "RFC 1034: Domain names - concepts and facilities," 1987.
[2] ——, "RFC 1035: domain names - implementation and specification," 1987.
[3] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "RFC 4033: DNS security introduction and requirements," 2005.
[4] ——, "RFC 4034: Resource records for the DNS security extensions," 2005.
[5] ——, "RFC 4035: Protocol modifications for the DNS security extensions," 2005.
[6] IKS, "DNSSEC." [Online]. Available: https://www.iks-jena.de/leistungen/dnssec.php
[7] "SecSpider." [Online]. Available: http://secspider.cs.ucla.edu/
[8] 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.
[9] "Root DNSSEC." [Online]. Available: http://www.root-dnssec.org/
[10] B. Laurie, G. Sisson, R. Arends, and D. Blacka, "RFC 5155: DNS security (DNSSEC) hashed authenticated denial of existence," 2008.
[11] O. Kolkman and R. Gieben, "RFC 4641: DNSSEC operational practices," 2006.
[12] M. StJohns, "RFC 5011: Automated updates of DNS security (DNSSEC) trust anchors," 2007.
[13] ISC BIND. [Online]. Available: http://www.isc.org/products/BIND/
[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] Internet Systems Consortium, "DNSSEC look-aside validation registry." [Online]. Available: https://dlv.isc.org/
[18] S. Weiler, "RFC 5074: DNSSEC lookaside validation," 2007.
[19] National Institude of Standards and Technology, "Secure naming infrastructure pilot." [Online]. Available: http://www.dnsops.gov/
[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] VeriSign Labs, "DNSSEC Debugger." [Online]. Available: http://dnssec-debugger.verisignlabs.com/
[22] "DNSCheck by .SE." [Online]. Available: http://dnscheck.iis.se/
[23] "Zonecheck." [Online]. Available: http://www.zonecheck.fr/
[24] "DNS Operations, Analysis, and Research Center." [Online]. Available: http://www.dns-oarc.net/