Privacy Implications of DNSSEC Look-aside Validation

Aziz Mohaisen
University at Buffalo
mohaisen@buffalo.edu

Zhongshu Gu
IBM T.J. Watson Research Center
zgu@us.ibm.com

Kui Ren
University at Buffalo
kuiren@buffalo.edu

***Abstract*—To complement DNSSEC operations, DNSSEC Look-aside Validation (DLV) is designed for alternative off-path validation. While DNS privacy attracts a lot of attention, the privacy implications of DLV are not fully investigated and understood. In this paper, we take a first in-depth look into DLV, highlighting its lax specifications and privacy implications. By performing extensive experiments over datasets of domain names under comprehensive experimental settings, our findings firmly confirm the privacy leakages caused by DLV. We discover that a large number of domains that should not be sent to DLV servers are being leaked. We explore the root causes, including the lax specifications of DLV. We also propose two approaches to fix the privacy leakages. Our approaches require trivial modifications to the existing DNS standards and we demonstrate their cost in terms of latency and communication.**

I. Introduction

Domain Name System Security Extensions (DNSSEC) leverage cryptographic methods to secure origin authentication, data integrity, and denial of existence of DNS reponses. Digital signatures ensure the authenticity of DNS responses by validating against a public key of the signer. Given the hierarchical nature of DNS [22], certain trust chains that include involved parties need to be validated up to a trust anchor. Such trust chains end with the root, which has a key pre-distributed and hardcoded in the operating system for validation. Accordingly, users only need to trust the root to successfully validate a DNSSEC response: a signature of a record is accepted and consumed by a user if the signature is validated using the signing public key. The authenticity of the signing keys beneath the root is ensured using key signing keys (KSKs). The validation of a record fails when any link in the chain of trust fails.

DNSSEC's deployment is incomplete and only a small proportion of domains have a complete chain of trust up to the root. While 85% of TLDs are signed, only $\approx 3\%$ of SLDs are signed as of early 2016 [14]. Thus, while some domain names have the capability of signing their zones, they cannot be validated up to the root because there is no delegation signer (DS) in the parent zone to validate the authenticity of the signing key. To address this issue, Weiler [26] proposed DNSSEC look-aside validation (DLV) to allow publishing of trust anchors outside of the delegation chain. By publishing DLV records in DLV repositories, domains are validated with the trust anchors in DLV records. Recursive resolvers are configured to use the DLV repositories for validation. The configurations of the DLV options in the recursive vary from one operating system and installation to another.

While RFC 5074 specifies DLV's use and the general validator's behavior [26], it leaves out a lot of details to implementations, including guidelines on when a DLV server is queried for the various use cases of DLV (see §II-A). Whether such lax specifications affect users' privacy and expose unintended queries to DLV servers was not tested before. Indeed, the prior art on DNS privacy treats unsecured DNS. Thus, we analyze the privacy leakage of DLV by highlighting the potential of *unintentional* DLV queries sent to the DLV servers while providing no validation benefits. We perform extensive analyses on 16 configurations of the Berkeley Internet Name Domain (BIND) and Unbound, two popular recursive resolvers, running on various operating systems. We find the rules of referring to DLV servers for validation are lax, resulting in DNS query leakages. The privacy leakage is highlighted as a third party (DLV server) can observe most queries a user has sent, while providing little validation utility.

Contributions. Our contributions are as follows. 1) We formulate the privacy leakage in DNS caused by unintentional queries. We analyze DNSSEC and DLV in light of this privacy risk in various settings. We anticipate that such partial deployment of DNSSEC and lax rules of DLV would amplify such risks. 2) We validate the privacy risks in various settings (resolvers, operating systems, installation tools, etc.) and use a large number of domains. We find that the amount of unintentional leaked queries is an order of magnitude larger than the number it is supposed to be. We discuss root causes of leakage and provide fixes. To the best of our knowledge, this is the first systematic treatment of privacy leakage in DLV.

Organization. We outline background and preliminaries in §II, and the threat model in §III. We review the main results in §IV and provide explanations of root causes and remedies in §V. The related work is in §VI and the concluding remarks are in §VII.

II. Background and Preliminaries

A. DNSSEC Look-aside Validation (DLV)

The deployment of DNSSEC to root was completed in July 2010. As of February 2016, more than 85% of delegated TLDs are signed in the root [14]. Nevertheless, the number of secured SLDs that are both signed and have DS registered in the parent zones is quite small compared to the total number of SLDs [25], [19], [24], [8] (see §V-A).

DNSSEC adds authentication on records in every zone. The partial deployment of DNSSEC results in "islands of security", where all nodes of a *subtree* in the domain namespace implement DNSSEC [17]. For example, having records in `example`, `com`, and `root` signed, while having no DS record in `com` to validate the signing public key used in `example` would result in an island of security: records in `example` cannot be validated, since the signing key is not trusted by the resolver. DLV addresses this problem [26]. With DLV, an owner of a zone can submit the signing public keys as DS records to a DLV registry, delegating a trust anchor. The DLV record is used when the normal operation of DNSSEC fails.

When DNSSEC fails for `example.com`, a security-aware resolver would generate a DLV query by appending the DLV domain after the queried domain. An example of a DLV is run by Internet Systems Consortium (ISC) and the DLV domain is `dlv.isc.org`. The type bit is set to `DLV` as 32769 in the DNS query. The DLV server, `dlv.isc.org`, searches its depository for the corresponding DS records: if there is no deposited DS record, the validator removes the leading label from the query and tries again [26]. This process is repeated until a DLV record is found or it is determined that there are no (enclosing) records applicable to the query in the repository. The enclosing record is helpful for queries containing multiple levels of domains, such as `bbs.sub1.example.com`. If there is no corresponding DS record, "No such name" will be returned. If the DLV records are deposited for the queried domain, the resolver can expect the zone to be securely signed and "No error" is returned. Figure 1 shows the workflow of DNSSEC and DLV when `example.com` is queried.

Aggressive negative caching. For efficiency, aggressive negative caching is implemented at the resolver, where failed queries are also cached [3]. The aggressive negative caching implemented in the validator checks whether any cached and validated NSEC record provides a denial of existence proof for records. In this way, a queried nonexistent domain, which is cached or is proved to be nonexistent by NSEC records, will not be sent to the DLV server for validation [26]. Negative caching, in general, is useful as it reduces the response time for negative answers and limits queries sent to name servers.

B. Privacy in DNSSEC and DLV

We note that most privacy concerns associated with DNS discussed in the related work (§VI) are also applicable to DNSSEC. Furthermore, given that DNSSEC involves a third party, certain extra privacy risks may arise. To the best of our knowledge, this risk was not previously studied at any level and deserves a treatment of measurement and analysis.

Hypothesis. DLV is used as an alternative off-path validation method. We hypothesize that there is a privacy risk of relying on DLV. We examine this hypothesis in the rest of this paper by measuring DLV in two most adopted DNS resolvers with different installation settings and configurations, running on various operating systems.

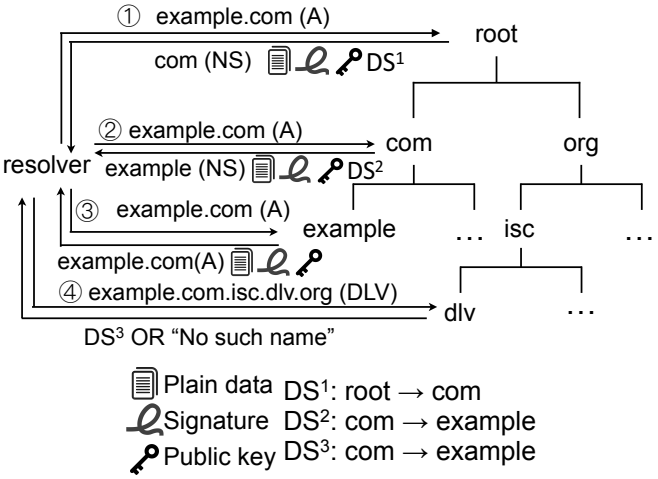


Fig. 1: A Workflow of DNSSEC and DLV

III. Threat Model

We now define our threat model, excluding cases that are out of its scope. For any given query, any entity in DNS is considered either an involved or an uninvolved party.

First, we exclude all directly involved name servers, viz. authoritative name servers that are involved in the given DNS queries. Second, we expect a DLV server, as a secondary validation server, to *be queried only for domain names that have records deposited in such server*. As such, we exclude the *intentional leakage* of domain names to DLV servers for domain names that are verifiable through it. As such, we consider such leakage to be no worse than the leakage of DNS queries to root or TLD name servers.

Defining Leakage. Informally, we state that a DNS query is leaked if an uninvolved party can *observe* that query during the DNS resolution [18]. Such party observes DNS queries without the corresponding record configured by the registrar or the consent of the user. As such, a DLV server is treated as uninvolved party when the queried domains do not have a DLV record deposited in the DLV server.

Under this model, we expect a DNS query not be sent to an uninvolved party; e.g., it is unexpected `nxdomain.com` to be sent to `.net` if `com` responds with an NXDOMAIN response. Conversely, if an authority server *A* signals the corresponding record of interest is deposited in another server *B* through a referral, the resolver is ought to query *B*. The actual implementations of DNS resolution comply with this rule. However, whether DLV does that or not is untested.

With DLV's main design goal to serve off-path validation only if DNSSEC fails, it is unclear if additional signaling for whether the corresponding DLV record is deposited in the DLV server. To this end, we define two cases of leakage with DLV:

- **Case-1:** happens when a domain has a DLV record deposited and the recursive resolver queries the DLV server for validation. As such, the DLV server may know what domain is queried, and may also serve them.

- **Case-2:** happens when a domain does not have DLV record, while the recursive resolver still queries the DLV server, thus allowing the DLV server to know about the queried domains without providing any validation benefit.

We notice that only the second case qualifies as a privacy issue in our model, since the first case of leakage is no worse than today's primary DNS resolution.

IV. Measurement Results

First, we measure the leakage of popular domain names (§IV-A), DNSSEC-secured domains (§IV-B) and the validation utility of DLV (§IV-C). We first quantify the DLV queries for both the top 1 million domains and the 45 DNSSEC-secured domains, and then quantify the unintentional ones by inspecting the payload of the query and response of a DNS resolution (case-2 in §III). The number of DLV queries where a domain name is not associated with DLV records highlights the privacy leaked. We note that, unless otherwise specified, the measurements, results and findings are the same for both resolver software packages, BIND and Unbound.

A. DLV Leakage for Popular Domains

The Alexa's top 100 and 1000,000 domains are used for testing popular domains. All DLV queries are extracted from the network traffic by filtering the query type. The number of leaked domains, is counted and the proportion of them over the total queried domains is calculated. Results show a large number of DLV queries, *e.g.,* 84% domains are sent to the DLV server when the Alexa's top 100 domain names are queried.

On the other hand, 67,838 domains are queried for DLV when the top million domains are used (a smaller percent). We notice an decreasing trend on the proportion of the leaked domains as the sample increases. After analyzing the design and implementation of DLV, we found the reason to be the use of aggressive negative caching (see §II-A). The validator collects more NSEC records by sending DLV queries. As such, by utilizing the knowledge of NSEC records, the resolver will not query a domain for which it has a proof of non-existence.

Order matters. We conduct a measurement where the same set of queried domain names is shuffled. Shuffling the domain names would result in different outcomes; for the top 100 domains used in the previous measurements we obtain 82%, 84%, and 77%, for three trials. The different results highlight the effect of the aggressive negative caching. If there are two domains which can be proved to be non-existent by the same NSEC record, only the first domain will be queried with DLV.

B. DLV Leakage for DNSSEC Domains

To examine the compliance of DLV with the standard operation of DNSSEC, we performed the following. First, 45 DNSSEC-secured domains are queried to see if their queries would result in additional DLV queries. When DNSSEC and DLV are correctly configured with trust anchor included in the configuration, five of the DNSSEC secured domains are sent to the DLV server. We found that at the time of the experiment all of those five domain names could not be validated through the on-path DNSSEC validation mechanism: we found that there is no `DS` record for these five domains in their parent zone, thus making them islands of security.

TABLE I: DLV for Secured Domains

	apt-get	apt-get†	yum	manual
DLV	No	Yes	No	Yes

DNSSEC-secured domains leaked to DLV. We set `dnssec-validation` to `yes`, while the trust anchor is not included, corresponding to the case where BIND is installed using `apt-get` and `dnssec-validation` is modified to `yes` according to the manual of BIND, or where BIND is installed manually where the trust anchor is not included.

In such cases, the DNSSEC-secured domains cannot be validated since the trust anchor is not included, thus making it impossible to complete the chain of trust. The DNSSEC secured domains are then sent to the DLV server for validation. Table I shows whether the secured domains will be queried for DLV when the trust anchor is not manually included. Each column stands for the default configuration of BIND where *apt-get*† means a user will change the `dnssec-validation` to `yes` in accordance with the manual configuration of BIND.

• **Unbound.** Since Unbound utilizes a different configuration style where the DNSSEC and DLV are enabled by including the trust anchors, domains do not leak with Unbound.

Practical Implications. We notice that BIND installed manually or by `apt-get` does not contain the statement for including the trust anchor, and only BIND installed by `yum` does by default. A user without the practical expertise or careful understanding of the operation of DNSSEC validation, including the knowledge of the used cryptographic schemes and required configuration is unlikely to correctly make the configuration that would not result in leakage.

To understand if the practical implications highlighted above are real or not, we performed the following survey. During a DNS-OARC 2015 Workshop, a gathering of DNS operators, administrators, and researchers, we surveyed attendees who use their own recursive for the prevalence of the problem. Out of 56 responses we obtained, 17 respondents (30.35%) indicated that they use defaults with package installer (apt-get or yum), 5 (8.9%) indicated that they use default settings with manual installation and the rest (60.7%) indicated that they use their own configuration. Of the 56 respondents, 35 (62.5%) indicated that they use ISC's DLV server, while the rest (37.5%) indicated that they use other trust anchors.

C. Validation Utility by DLV

The validation utility provided by DLV is measured by inspecting DLV responses, where "No such name" indicates the non-existence of DLV records, and thus the DLV provides no validation utility to the queried domains despite being exposed to the DLV server. According to our threat model, such unintended DLV queries leak privacy of users. DLV responses containing "No error" mean the queried domain is validated by DLV records deposited in the DLV server.

By further inspecting the responses (at the packet-level), we conclude that those are the only two types of messages returned by the DLV server. By running this experiment for Alexa's top 10,000 domain names, we found that less than 1.2% DLV queries receiving "No error" (1168 domains). As a result, about 98.8% of DLV queries are a result of leakage.

V. Root Causes and Remedies

A. Root Causes of Privacy Leakage

DNSSEC Deployment. DNSSEC is partially deployed despite many years since its creation. Only 0.88% zones are signed as of November, 2015 [19] and the proportion of DNSSEC secured domains (SLD-level) is quite low: 0.43% for `com`, 0.61% for `net` and 0.89% for `edu` based on a survey we conducted in November 2015. While not particularly a root cause of the leakage of domains to DLV, the current level of deployment of DNSSEC actually contributes to the utility of DLV highlighted in §IV-C. We anticipate such utility to be higher if DNSSEC is more widely deployed.

When to use DLV. Although intended for the use as an off-path validation mechanism when the main DNSSEC validation fails for a reason or another, the rules used to determine when a DLV server is queried are lax. The configurations associated with various distribution are ambiguous, inconsistent, or poorly documented, depending on the distribution itself and the mechanism being used for the resolver installation. Such lax conditions and rules lack any signaling of when DLV should be used. In general, we believe that not every domain name issued by a stub resolver should be sent to a DLV server, even when the DLV and DNSSEC options are enabled at the recursive resolver, especially that many domains are not DNSSEC-enabled in the first place.

B. Possible Remedies of Privacy Leakage

Here we propose two possible remedies, **DLV-Aware DNS** and **Privacy-Preserving DLV**.

1) DLV-Aware DNS: The main idea of the remedy is to signal and inform the resolver to only issue DLV queries for those domains that have deposited their DLV records in the DLV server. We assume that the recursive is trusted, and would comply with such signaling on behalf of the stub resolver. We suggest two possible methods to achieve the goal.

- **Using TXT record.** In this method, we add a description (*e.g.,* `DLV=1`) in the DNS TXT record, indicating the existence of DLV records, where the resolver has to query for them in case the main validation method fails. The resolver will be informed by querying and checking the TXT record.
- **Using Z bit.** Another similar way is by setting the spare "Z" bit [1] in the DNS response header to signal the existence of DLV records. Note that using the "Z" bit requires IANA allocating the bit for a special use, although it can easily fit in the current DNS implementation.

Potential Attacks. While not requiring minimal modifications to DNS, the proposed fixes are vulnerable to zone poisoning and man-in-the-middle. An attacker can also mislead the recursive by modifying the TXT record, or by flipping the "Z" bit. A potential remedy to such attack is to sign the response.

Evaluation of TXT record. We measure the overhead when using `TXT` queries with three evaluation metrics: the response time (in seconds), traffic volume (in Megabytes) and the number of issued queries. The experiment is performed under four datasets containing different numbers of domains (to account for caching behavior). We use 100, 1K, 10K and 100K domains, respectively. Note that we take the record `TXT` as a "spare" record, where a domain registrant configures the information indicating the existence of the corresponding DLV records. Alternatively, the domain registrant can configure such information in other unused DNS records. For signaling, we TXT record includes either `dlv=1` or `dlv=0`.

Overhead measurement. Four datasets containing 100, 1K, 10K, and 100K domains are used, where we insert the `TXT` queries after each original query, to measure the overhead. We compare the overhead with DLV alone. Statistics are extracted under the original traffic without `TXT` queries and responses, the actual overhead, and the total.

There are six types of DNS queries including `A`, `AAAA`, `DS`, `DNSKEY`, `NS` and `PTR` aside from the `DLV` and `TXT`. We show the statistics for the number of issued queries to give an overview of the total traffic in Table II.

TABLE II: Number of Different Types of DNS Queries

# domains	A	AAAA	DNSKEY	DS	NS	PTR
100	467	243	32	221	36	2
1k	4032	1881	96	1963	285	13
10k	30972	10566	390	18582	2701	43
100k	283949	66498	3264	203683	33402	331

The statistics under the three metrics are shown in Table III. When 100 domains are queried, the fix increases the response time by 18.68%, the traffic volume by 6.67% and the number of issued queries by 10.79%. Similar results are extrapolated for the different datasets, as shown in Table III.

Using the three metrics, the response time for the added `TXT` records represents the largest overhead. By further inspection, we find the average response time for `TXT` queries are higher than other types such as `A`, mainly because not all domains are configured with the `TXT` record. We argue that the actual overhead will be reduced with a wide deployment. For the rest of the datasets, the ratio of the overhead increases from about 10% to 20% when 100K domains are queried. We interpret the result as more queries on average for the referrals of SLDs.

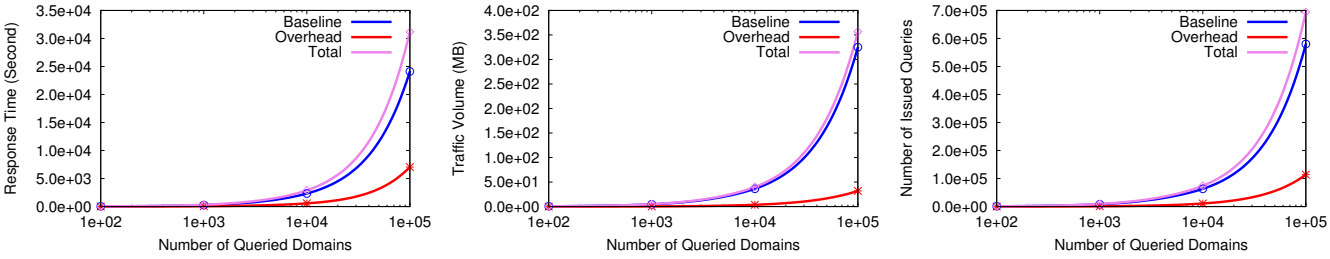
We compare the overhead with the baseline (without `TXT`) and the total overhead in Figure 2.

Evaluation using Z bit. Similarly, we compare the overhead of `TXT` to the overhead of "Z" bit as fixes. The results are shown in Figure 3 (baseline), showing the benefit the "Z" bit approach, which requires nominal overhead.

2) Privacy-Preserving DLV: The second remedy involves changing the data format provided for both DLV registration and query. On DLV record registration, instead of depositing `(domain_name, DNSKEY)`, we compute `$digest = crypto_hash(domain_name)` and

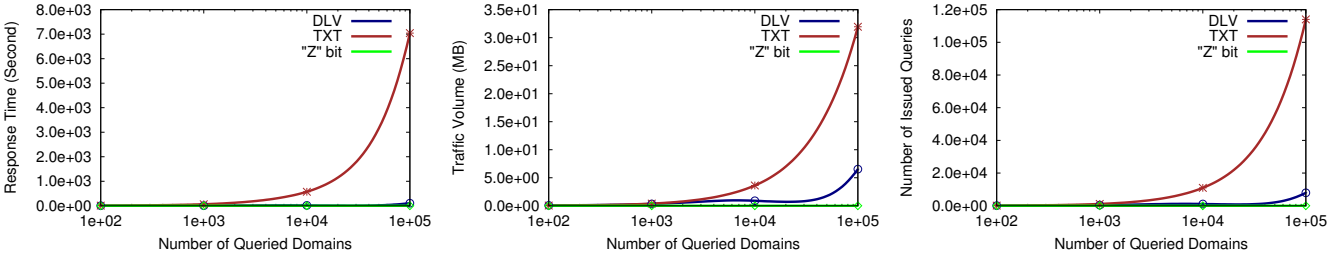
TABLE III: Overhead of the Original and Increased in Three Metrics

	Response Time (Seconds)			Traffic Volume (MB)			# Issued Queries		
#. Domains	Baseline	Overhead	Ratio	Baseline	Overhead	Ratio	Baseline	Overhead	Ratio
100	38.16	7.13	18.68%	0.60	0.04	6.67%	1001	108	10.79%
1K	270.278	63.28	23.41%	4.61	0.39	8.46%	8270	1120	13.54%
10K	2324.45	571.69	24.59%	36.31	3.62	9.97%	63254	10960	17.33%
100K	24119.23	7043.17	29.20%	324.90	31.95	9.83%	580127	114043	19.66%



(a) Response Time (b) Traffic Volume (c) Number of Queries

Fig. 2: Baseline, overhead, and total performance.



(a) Response Time (b) Traffic Volume (c) Number of Queries

Fig. 3: DLV, TXT, and "Z" bit performance.

deposit `(domain_name, DNSKEY, $digest)` to the DLV server. On DLV query, the resolver only sends the computed hash instead of the `domain_name` to the DLV server. For example, in step 4 of Figure 1, assuming `$hash = crypto_hash("example.com")`, we send `$hash.isc.dlv.org` to the DLV server. At DLV server, it compares `$hash` with the stored `$digest`. If they match, it is in the same situation as the Case-1 (§III), thus leading to no additional leakage. If they do not match, DLV server is not able to reverse engineer the domain name from the `$hash` (except if it computes exhaustively for all the digests of domain names that are not on its DLV server, which we consider impractical), thus no privacy leakage if compared to Case-2 (§III).

VI. Related Work

Here we review the related work on DNS privacy and neighboring topics, including the theoretical and experimental literature quantifying privacy risks, prior designs for DNS privacy, measurements, and evaluations of DNSSEC as well as prior work and discussions on DLV.

Privacy Risks with DNS. With the rise of pervasive surveillance as a threat [5], DNS privacy has attracted some attention recently [28], [18], [11], [21], [16], [7], [27]. A monitor performing pervasive surveillance is able to gather artifacts from which he can breach the privacy of users [9]. DNS traffic is highly valuable for two reasons, 1) it is metadata, thus it is easy to analyze and use, and 2) it often includes explicit information about user's behavior. To highlight this risk, the literature provides various studies in various contexts. Banse *et al.* [4] performed a behavior-based tracking to analyze DNS query logs for a large user group. They show that more than 2000 users are correctly linked to 88.2% of all sessions with their DNS query log. Konings *et al.* [15] performed device identification based on DNS traffic: using a one-week network traffic dataset from a university public Wi-Fi network, they showed that 59% of the device names include personal information where 17.6% of the information contain both first and last name.

Privacy risks of DNS prefetching are explored by Krishnan *et al.* [16]. Shulman [21] performed a meta-study of existing proposals that implemented encryption in DNS requests for privacy. She highlighted that a straightforward application of encryption alone may not suffice to protect the privacy in DNS.

Designs for DNS Privacy. Zhao *et al.* [27] propose to ensure DNS privacy by concealing actual queries using noisy

traffic. Castillo-Perez *et al.* [7] evaluated this approach and demonstrated that the privacy ensured by added queries is somewhat difficult to analyze, and that the technique introduces additional latency and overhead, making it less practical. Hermann *et al.* [11] proposed EncDNS, a lightweight privacy-preserving implementation that replaces third-party resolvers and provides a client software to forward queries to it through conventional DNS forwarders. EncDNS provides an end-to-end encryption, thus queries are not exposed to forwarders.

The IETF has recently established a working group for addressing DNS privacy concerns (called DNS PRIVate Exchange, DPRIVE). The group proposed various techniques that are currently being under consideration [18]. Zhu *et al.* [28], [13] proposed a connection-oriented DNS transport over TCP and using TLS for privacy. Reddy *et al.* [23] proposed to use the Datagram Transport Layer Security (DTLS) for DNS exchange. They add a protection layer for the sensitive information in DNS queries, which would withstand a passive listener and certain active attackers, and argue that their mechanism reduces the round trip time of DTLS and the handshake phase.

DNSSEC Measurements and Evaluations. Bau *et al.* [6] formally modeled the cryptographic operations in DNSSEC and discovered a vulnerability that allows an attacker to add a forged name into a signed zone. Herzberg *et al.* [12] presented a comprehensive overview of challenges and potential pitfalls of DNSSEC, including vulnerable configurations, increased vulnerabilities due to incremental deployment, and interoperability challenges in large DNSSEC-enabled DNS responses. Goldberg *et al.* [10] demonstrated zone-enumeration vulnerabilities on the NSEC and NSEC3. They show that the security against attackers tampering with DNS messages and protection against zone enumeration cannot be satisfied simultaneously. They also proposed NSEC5, a provably secure DNSSEC denial of existence mechanism.

DNSSEC Look-aside Validation. The operation of DLV has been a topic of debate in the IETF community [2], [20]. Discussions have been mainly focused on trustworthiness of the third party running the DLV servers. For example, it is argued that a DLV server should be continuously running in order for the DLV to serve its intended purpose. However, this is not always guaranteed, given several reported outages. Finally, Osterweil [20] argued, although did not measure, that DLV presents a privacy risk, by allowing a third party to observe queries initiated by users.

VII. Conclusion

In this paper, we analyzed privacy leakage in the DNSSEC look-aside validation (DLV). Experiments are performed on two popular DNS resolver software under extensive experimental environments and various settings. Results show that DLV's rules are lax and result in privacy leakage of unintended queries to third parties. We also propose fixes to the problem and evaluate their scalability. While highlighting privacy risks of this particular protocol, this study also aims to call for further efforts on understanding the privacy risks in DNS.

Acknowledgement

We would like to thank Duane Wessels for pointing out the issue with DLV default behavior in Ubuntu, which motivated this work. This work is supported by NSF grant CNS-1643207 and Global Research Lab. (GRL) Program of the National Research Foundation (NRF) funded by Ministry of Science, ICT (Information and Communication Technologies) and Future Planning (NRF-2016K1A1A2912757).

References

[1] D. E. 3rd, "Domain name system (DNS) IANA Considerations," IETF RFC 5395, 2008.
[2] M. Andrews, "dnssec with freebsd's resolver," Online.
[3] ——, "Negative caching of DNS queries (DNS NCACHE)," IEFT RFC 2308.
[4] C. Banse, D. Herrmann, and H. Federrath, "Tracking users on the internet with behavioral patterns: Evaluation of its practical feasibility." in *Proc. of AICT*, 2012.
[5] R. Barnes and et al., "Confidentiality in the face of pervasive surveillance: A threat model and problem statement," IETF RFC 7624, 2015.
[6] J. Bau and J. C. Mitchell, "A security evaluation of DNSSEC with NSEC3," in *Proceedings of NDSS*, 2010.
[7] S. Castillo-Perez and J. Garcia-Alfaro, "Evaluation of two privacy-preserving protocols for the DNS," in *Proceedings of IEEE ITNG*, 2009, pp. 411–416.
[8] Educause, "Average Counts of .EDU Domains," Online, 2015.
[9] S. Farrell and H. Tschofenig, "Pervasive monitoring is an attack," IETF RFC 7258, 2014.
[10] S. Goldberg, M. Naor, D. Papadopoulos, L. Reyzin, S. Vasant, and A. Ziv, "NSEC5: provably preventing DNSSEC zone enumeration," in *Proceedings of NDSS*, 2015.
[11] D. Herrmann, K. Fuchs, J. Lindemann, and H. Federrath, "Encdns: A lightweight privacy-preserving name resolution service," in *Proceedings of ESORICS*, 2014, pp. 37–55.
[12] A. Herzberg and H. Shulman, "DNSSEC: Security and availability challenges," in *Proceedings of IEEE CNS*, 2013, pp. 365–366.
[13] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman, "DNS over TLS: Initiation and performance considerations," IETF Internet Draft, 2015.
[14] ICANN, "TLD DNSSEC Report," Online, 05 2016.
[15] B. Konings, C. Bachmaier, F. Schaub, and M. Weber, "Device names in the wild: Investigating privacy risks of zero configuration networking," in *Proceedings of the 2013 IEEE MDM*, 2013, pp. 51–56.
[16] S. Krishnan and F. Monrose, "DNS prefetching and its privacy implications: When good things go bad," in *Proceedings of USENIX LEET*, 2010, pp. 1–9.
[17] E. Lewis, "DNS security extension clarification on zone status," IETF RFC 3090, 2001.
[18] A. Mohaisen and A. Mankin, "Evaluation of privacy for DNS private exchange," IETF Internet Draft, 2015.
[19] nTLDStats, "DNSSEC Breakdown," Online, 2015.
[20] E. Osterweil, "Unplanned DLV zone outage on 2009-Apr-06," Online, 2009.
[21] H. Shulman, "Pretty bad privacy: Pitfalls of DNS encryption," in *Proceedings of WPES*, 2014, pp. 191–200.
[22] J. Spaulding, S. J. Upadhyaya, and A. Mohaisen, "The landscape of domain name typosquatting: Techniques and countermeasures," in *11th International Conference on Availability, Reliability and Security, ARES*, 2016, pp. 284–289.
[23] D. W. T. Reddy and P. Patil, "DNS over DTLS (DNSoD)," IETF Internet Draft, 2015.
[24] VeriSign, "The domain name industry brief," Online, 2015.
[25] ——, "Domains Secured with DNSSEC," Online, 2015.
[26] S. Weiler, "DNSSEC lookaside validation (DLV)," IETF RFC 5074.
[27] F. Zhao, Y. Hori, and K. Sakurai, "Analysis of privacy disclosure in dns query," in *Proceedings of MUE*, 2007, pp. 952–957.
[28] L. Zhu, Z. Hu, H. J., W. D., M. A., and S. N., "Connection-oriented DNS to improve privacy and security," in *Proceedings of 2015 IEEE S&P*, 2015, pp. 171–186.