Network Working Group
Request for Comments: 5074
Category: Informational

S. Weiler
SPARTA, Inc.

November 2007

DNSSEC Lookaside Validation (DLV)

Status of This Memo

Abstract

DNSSEC Lookaside Validation (DLV) is a mechanism for publishing DNS Security (DNSSEC) trust anchors outside of the DNS delegation chain. It allows validating resolvers to validate DNSSEC-signed data from zones whose ancestors either aren't signed or don't publish Delegation Signer (DS) records for their children.

Table of Contents

1. Introduction

DNSSEC [RFC4033] [RFC4034] [RFC4035] authenticates DNS data by building public-key signature chains along the DNS delegation chain from a trust anchor.

In the present world, with the DNS root and many key top level domains unsigned, the only way for a validating resolver ("validator") to validate the many DNSSEC-signed zones is to maintain a sizable collection of preconfigured trust anchors. Maintaining multiple preconfigured trust anchors in each DNSSEC-aware validator presents a significant management challenge.

This document describes a way to publish trust anchors in the DNS outside of the normal delegation chain, as a way to easily configure many validators within an organization or to "outsource" trust anchor management.

Some design trade-offs leading to the mechanism presented here are described in [INI1999-19].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Architecture

DNSSEC Lookaside Validation allows a set of domains, called "DLV domains", to publish secure entry points for zones that are not their own children.

With DNSSEC, validators may expect a zone to be secure when validators have one of two things: a preconfigured trust anchor for the zone or a validated Delegation Signer (DS) record for the zone in the zone's parent (which presumes a preconfigured trust anchor for the parent or another ancestor). DLV adds a third mechanism: a validated entry in a DLV domain (which presumes a preconfigured trust anchor for the DLV domain). Whenever a DLV domain contains a DLV RRset for a zone, a validator may expect the named zone to be signed. Absence of a DLV RRset for a zone does not necessarily mean that the zone should be expected to be insecure; if the validator has another reason to believe the zone should be secured, validation of that zone's data should still be attempted.

3. DLV Domains

A DLV domain includes trust statements about descendants of a single zone, called the 'target' zone. For example, the DLV domain trustbroker.example.com could target the org zone and the DLV domain bar.example.com could target the root.

A DLV domain contains one or more DLV records [RFC4431] for each of the target's descendant zones that have registered security information with it. For a given zone, the corresponding name in the DLV domain is formed by replacing the target zone name with the DLV domain name.

For example, assuming the DLV domain trustbroker.example.com targets the org zone, any DLV records corresponding to the zone example.org can be found at example.trustbroker.example.com. DLV records corresponding to the org zone can be found at the apex of trustbroker.example.com.

As another example, assuming the DLV domain bar.example.com targets the root zone, DLV records corresponding to the zone example.org can be found at example.org.bar.example.com. DLV records corresponding to the org zone can be found at org.bar.example.com, and DLV records corresponding to the root zone itself can be found at the apex of bar.example.com.

A DLV domain need not contain data other than DLV records, appropriate DNSSEC records validating that data, the apex NS and SOA records, and, optionally, delegations. In most cases, the operator of a DLV domain will probably not want to include any other RR types in the DLV domain.

To gain full benefit from aggressive negative caching, described in Section 6, a DLV domain SHOULD NOT use minimally-covering NSEC records, as described in [RFC4470], and it SHOULD NOT use NSEC3 records, as described in [NSEC3].

4. Overview of Validator Behavior

To minimize the load on the DLV domain's authoritative servers as well as query response time, a validator SHOULD first attempt validation using any applicable (non-DLV) trust anchors. If the validation succeeds (with a result of Secure), DLV processing need not occur.

When configured with a trust anchor for a DLV domain, a validator SHOULD attempt to validate all responses at and below the target of that DLV domain.

To do validation using DLV, a validator looks for a (validated) DLV RRset applicable to the query, as described in the following section, and uses it as though it were a DS RRset to validate the answer using the normal procedures in Section 5 of RFC 4035.

For each response, the validator attempts validation using the "closest enclosing" DLV RRset in the DLV domain, which is the DLV RRset with the longest name that matches the query or could be an ancestor of the QNAME. For example, assuming the DLV domain trustbroker.example.com targets the org zone, and there exist DLV RRsets named trustbroker.example.com (applicable to org), example.trustbroker.example.com (applicable to example.org), and sub.example.trustbroker.example.com (applicable to sub.example.org), a validator would use the sub.example.trustbroker.example.com DLV RRset for validating responses to a query for sub.example.org.

The choice of which DLV record(s) to use has a significant impact on the query load seen at DLV domains' authoritative servers. The particular DLV selection rule described in this document results in a higher query load than some other selection rules, but it has some advantages in terms of the security policies that it can implement. More detailed discussion of this DLV selection rule as well as several alternatives that were considered along the way can be found in [INI1999-19].

5. Details of Validator Behavior

As above, to minimize the load on the DLV domain's authoritative servers as well as query response time, a validator SHOULD first attempt validation using any applicable (non-DLV) trust anchors. If the validation succeeds (with a result of Secure), DLV processing need not occur.

To find the closest enclosing DLV RRset for a given query, the validator starts by looking for a DLV RRset corresponding to the QNAME. If it doesn't find a DLV RRset for that name (as confirmed by the presence of a validated NSEC record) and that name is not the apex of the DLV domain, the validator removes the leading label from the name and tries again. This process is repeated until a DLV RRset is found or it is proved that there is no enclosing DLV RRset applicable to the QNAME. In all cases, a validator SHOULD check its cache for the desired DLV RRset before issuing a query. Section 8 discusses a slight optimization to this strategy.

Having found the closest enclosing DLV RRset or received proof that no applicable DLV RRset exists, the validator MUST validate the RRset or non-existence proof using the normal procedures in Section 5 of RFC 4035. In particular, any delegations within the DLV domain need

to be followed, with normal DNSSEC validation applied. If validation of the DLV RRset leads to a result of Bogus, then it MUST NOT be used and the validation result for the original response SHOULD be Bogus, also. If validation of the DLV RRset leads to a result of Insecure (i.e., the DLV record is in an unsecured portion of the DLV domain), then it MUST NOT be used and the validation result for the original response SHOULD be Insecure, also. (It should be very odd, indeed, to find part of a DLV domain marked as Insecure: this is likely to happen only when there are delegations within the DLV domain and some portions of that domain use different cryptographic signing algorithms.) If the validation of the DLV RRset leads to a result of Secure, the validator then treats that DLV RRset as though it were a DS RRset for the applicable zone and attempts validation using the procedures described in Section 5 of RFC 4035.

In the interest of limiting complexity, validators SHOULD NOT attempt to use DLV to validate data from another DLV domain.

6. Aggressive Negative Caching

To minimize load on authoritative servers for DLV domains, particularly those with few entries, DLV validators SHOULD implement aggressive negative caching, as defined in this section.

Previously, cached negative responses were indexed by QNAME, QCLASS, QTYPE, and the setting of the CD bit (see RFC 4035, Section 4.7), and only queries matching the index key would be answered from the cache. With aggressive negative caching, the validator, in addition to checking to see if the answer is in its cache before sending a query, checks to see whether any cached and validated NSEC record denies the existence of the sought record(s).

Using aggressive negative caching, a validator will not make queries for any name covered by a cached and validated NSEC record. Furthermore, a validator answering queries from clients will synthesize a negative answer whenever it has an applicable validated NSEC in its cache unless the CD bit was set on the incoming query.

6.1. Implementation Notes

Implementing aggressive negative caching suggests that a validator will need to build an ordered data structure of NSEC records in order to efficiently find covering NSEC records. Only NSEC records from DLV domains need to be included in this data structure.

Also note that some DLV validator implementations do not synthesize negative answers to insert into outgoing responses -- they only use aggressive negative caching when looking up DLV RRs as part of their internal DLV validation.

7. Overlapping DLV Domains

It is possible to have multiple DLV domains targeting overlapping portions of the DNS hierarchy. For example, two DLV domains, perhaps operated by different parties, might target the org zone, or one DLV domain might target the root while another targets org.

If a validator supports multiple DLV domains, the choice of precedence in case of overlap is left up to the implementation and SHOULD be exposed as a configuration option to the user (as compared to the choice of DLV records within each domain, a precedence for which is clearly specified in this document). As a very simple default, a validator could give precedence to the most specific DLV domain.

Some other reasonable options include:

1. Searching all applicable DLV domains until an applicable DLV record is found that results in a successful validation of the response. In the case where no applicable DLV record is found in any DLV domain, the answer will be treated as Unsecure.

2. Applying some sort of precedence to the DLV domains based on their perceived trustworthiness.

3. Searching all applicable DLV domains for applicable DLV records and using only the most specific of those DLV records.

4. If multiple DLV domains provide applicable DLV records, use a threshold or scoring system (e.g., "best 2 out of 3") to determine the validation result.

The above list is surely not complete, and it's possible for validators to have different precedence rules and configuration options for these cases. [INI1999-19] discusses different policies for selecting from multiple DLV records within the same DLV domain. That discussion may also be applicable to the question of which DLV domain to use and may be of interest to implementers of validators that support multiple DLV domains.

8. Optimization

This section documents an optimization to further reduce query load on DLV servers and improve validator response time.

Authoritative servers, when processing a query for a DLV RRset, SHOULD include all DLV RRsets potentially applicable to a query (specifically, all DLV RRsets applicable to the QNAME and any of its ancestors) in the Additional section of the response as well as NSEC records proving the non-existence of any other applicable DLV records in the DLV domain. Authoritative servers need only include DLV RRsets they're aware of -- RRsets in sub-zones may be omitted.

Validators still seek out of the closest enclosing DLV RRset first. If they receive any data about other DLV RRsets in the zone, they MAY cache and use it (assuming that it validates), thus avoiding further round-trips to the DLV domain's authoritative servers.

9. Security Considerations

Validators MUST NOT use a DLV record unless it has been successfully authenticated. Normally, validators will have a trust anchor for the DLV domain and use DNSSEC to validate the data in it.

Aggressive negative caching increases the need for validators to do some basic validation of incoming NSEC records before caching them. In particular, the 'next name' field in the NSEC record MUST be within the zone that generated (and signed) the NSEC. Otherwise, a malicious zone operator could generate an NSEC that reaches out of its zone -- into its ancestor zones, even up into the root zone -- and use that NSEC to spoof away any name that sorts after the name of the NSEC. We call these overreaching NSECs. More insidiously, an attacker could use an overreaching NSEC in combination with a signed wildcard record to substitute a signed positive answer in place of the real data. This checking is not a new requirement -- these attacks are a risk even without aggressive negative caching. However, aggressive negative caching makes the checking more important. Before aggressive negative caching, NSECs were cached only as metadata associated with a particular query. An overreaching NSEC that resulted from a broken zone signing tool or some misconfiguration would only be used by a cache for those queries that it had specifically made before. Only an overreaching NSEC actively served by an attacker could cause misbehavior. With aggressive negative caching, an overreaching NSEC can cause broader problems even in the absence of an active attacker. This threat can be easily mitigated by checking the bounds on the NSEC.

As a reminder, validators MUST NOT use the mere presence of an RRSIG or apex DNSKEY RRset as a trigger for doing validation, whether through the normal DNSSEC hierarchy or DLV. Otherwise, an attacker might perpetrate a downgrade attack by stripping off those RRSIGs or DNSKEYs.

Section 8 of RFC 4034 describes security considerations specific to the DS RR. Those considerations are equally applicable to DLV RRs. Of particular note, the key tag field is used to help select DNSKEY RRs efficiently, but it does not uniquely identify a single DNSKEY RR. It is possible for two distinct DNSKEY RRs to have the same owner name, the same algorithm type, and the same key tag. An implementation that uses only the key tag to select a DNSKEY RR might select the wrong public key in some circumstances.

For further discussion of the security implications of DNSSEC, see RFCs 4033, 4034, and 4035.

10. IANA Considerations

DLV makes use of the DLV resource record (RR type 32769) previously assigned in [RFC4431].

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.

[RFC4431] Andrews, M. and S. Weiler, "The DNSSEC Lookaside Validation (DLV) DNS Resource Record", RFC 4431, February 2006.

11.2. Informative References

[INI1999-19] Weiler, S., "Deploying DNSSEC Without a Signed Root", Technical Report 1999-19, Information Networking Institute, Carnegie Mellon University, April 2004.

[NSEC3] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNSSEC Hashed Authenticated Denial of Existence", Work in Progress, July 2007.

[RFC4470] Weiler, S. and J. Ihren, "Minimally Covering NSEC Records and DNSSEC On-line Signing", RFC 4470, April 2006.

Appendix A. Acknowledgments

Johan Ihren, Paul Vixie, and Suzanne Woolf contributed significantly to the exploration of possible validator algorithms that led to this design. More about those explorations is documented in [INI1999-19].

Johan Ihren and the editor share the blame for aggressive negative caching.

Thanks to David B. Johnson and Marvin Sirbu for their patient review of [INI1999-19] which led to this specification being far more complete.

Thanks to Mark Andrews, Rob Austein, David Blacka, Stephane Bortzmeyer, Steve Crocker, Wes Hardaker, Alfred Hoenes, Russ Housley, Peter Koch, Olaf Kolkman, Juergen Quittek, and Suzanne Woolf for their valuable comments on this document.

Author's Address

Samuel Weiler
SPARTA, Inc.
7110 Samuel Morse Drive
Columbia, Maryland 21046
US

EMail: weiler@tislabs.com

Full Copyright Statement

Intellectual Property