Network Working Group
Internet-Draft
Updates: 4035 (if approved)
Intended status: Standards Track
Expires: March 17, 2017

K. Fujiwara
JPRS
A. Kato
Keio/WIDE
W. Kumari
Google

September 13, 2016

Aggressive use of NSEC/NSEC3
draft-ietf-dnsop-nsec-aggressiveuse-02

Abstract

The DNS relies upon caching to scale; however, the cache lookup generally requires an exact match. This document specifies the use of NSEC/NSEC3 resource records to generate negative answers within a range. This increases performance / decreases latency, decreases resource utilization on both authoritative and recursive servers, and also increases privacy. It may also help increase resilience to certain DoS attacks in some circumstances.

This document updates RFC4035 by allowing resolvers to generate negative answers based upon NSEC/NSEC3 records.

[Ed note: Text inside square brackets ([]) is additional background information, answers to frequently asked questions, general musings, etc. They will be removed before publication.This document is being collaborated on in Github at: https://github.com/wkumari/draft-ietf-dnsop-nsec-aggressiveuse. The most recent version of the document, open issues, etc should all be available here. The authors (gratefully) accept pull requests.

Known / open issues [To be moved to Github issue tracker]:

1. We say things like: "Currently the DNS does ..." - this will not be true after this is deployed, but I'm having a hard time rewording this. "Without the techniques described in this document..." seems klunky. Perhaps "historically?!"

]

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Copyright Notice

Table of Contents

1. Introduction

A DNS negative cache currently exists, and is used to cache the fact that a name does not exist. This method of negative caching requires exact matching; this leads to unnecessary additional lookups, increases latency, leads to extra resource utilization on both authoritative and recursive servers, and decreases privacy by leaking queries.

This document updates RFC 4035 to allow recursive resolvers to use NSEC/NSEC3 resource records to aggressively cache negative answers. This would allow such resolvers to respond with NXDOMAIN immediately if the name in question falls into a range expressed by a NSEC/NSEC3 resource record already in the cache.

Aggressive Negative Caching was first proposed in Section 6 of DNSSEC Lookaside Validation (DLV) [RFC5074] in order to find covering NSEC records efficiently.

Section 3 of [I-D.vixie-dnsext-resimprove] "Stopping Downward Cache Search on NXDOMAIN" and [I-D.ietf-dnsop-nxdomain-cut] proposed another approach to use NXDOMAIN information effectively.

2. Terminology

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].

Many of the specialized terms used in this document are defined in DNS Terminology [RFC7719]. In this document we are using the terms "recursive resolver" or "recursive server" as a more readable alternative to the more formal[RFC7719] "full-service resolver"

The key words "Closest Encloser" and "Source of Synthesis" in this document are to be interpreted as described in[RFC4592].

"Closest Encloser" is also defined in NSEC3 [RFC5155], as is "Next closer name".

3. Problem Statement

The current DNS negative cache caches negative (non-existent) information, and requires an exact match in most instances [RFC2308].

Assume that the (DNSSEC signed) "example.com" zone contains:

apple.example.com IN A 192.0.2.1

elephant.example.com IN A 192.0.2.2

zebra.example.com IN A 192.0.2.3

If a recursive resolver gets a query for cat.example.com, it will query the example.com authoritative servers and will get back an NSEC (or NSEC3) record starting that there are no records between apple and elephant. The recursive resolver then knows that cat.example.com does not exist; however, it (currently) does not use the fact that the proof covers a range (apple to elephant) to suppress queries for other labels that fall within this range. This means that if the recursive resolvers gets a query for ball.example.com (or dog.example.com) it will once again go off and query the example.com servers for these names.

Apart from wasting bandwidth, this also wastes resources on the recursive server (it needs to keep state for outstanding queries), wastes resources on the authoritative server (it has to answer additional questions), increases latency (the end user has to wait longer than necessary to get back an NXDOMAIN answer), can be used by attackers to cause a DoS (see additional resources), and also has privacy implications (e.g: typos leak out further than necessary).

4. Background

DNSSEC [RFC4035] and [RFC5155] both provide "authenticated denial of existence"; this is a cryptographic proof that the queried for name does not exist, accomplished by providing a (DNSSEC secured) record containing the names which appear alphabetically before and after the queried for name. In the example above, if the (DNSSEC validating) recursive server were to query for lion.example.com it would receive a (signed) NSEC/NSEC3 record stating that there are no labels between "elephant" and "zebra". This is a signed, cryptographic proof that these names are the ones before and after the queried for label. As lion.example.com falls within this range, the recursive server knows that lion.example.com really does not exist. This document specifies that this NSEC/NSEC3 record should be used to generate negative answers for any queries that the recursive server receives that fall within the range covered by the record (for the TTL for the record).

[RFC4035]; Section 4.5 states:

For a zone signed with NSEC, it would be possible to use the information carried in NSEC resource records to indicate the non-existence of a range of names. However, such use is discouraged by Section 4.5 of RFC4035. It is recommended that readers read RFC4035 in its entirety for a better understanding. At the root of the concern is that new records could have been added to the zone during the TTL of the NSEC record, and that generating negative responses from the NSEC record would hide these. We believe this recommendation can be relaxed because lookups for the specific name could have come in during the normal negative cache time and so operators should have no expectation that an added name would work immediately. We think that the TTL of the NSEC record is the authoritative statement of how quickly a name can start working within a zone.

5. Proposed Solution

5.1. Aggressive Negative Caching

Section 4.5 of [RFC4035] shows that "In theory, a resolver could use wildcards or NSEC RRs to generate positive and negative responses (respectively) until the TTL or signatures on the records in question expire. However, it seems prudent for resolvers to avoid blocking new authoritative data or synthesizing new data on their own. Resolvers that follow this recommendation will have a more consistent view of the namespace".

This document relaxes this this restriction, as follows:

```
+-------------------------------------------------------------+
|  Once the records are validated, DNSSEC enabled validating  |
|  resolvers MAY use NSEC/NSEC3 resource records to generate  |
|  negative responses until their effective TTLs or signatures |
|  for those records expire.                                  |
+-------------------------------------------------------------+
```

If the validating resolver's cache has sufficient information to validate the query, the resolver SHOULD use NSEC/NSEC3/wildcard records aggressively. Otherwise, it MUST fall back to send the query to the authoritative DNS servers.

If the query name has the matching NSEC/NSEC3 RR proving the information requested does not exist, the resolver may respond with a NODATA (empty) answer.

5.2. NSEC

Implementations SHOULD enable aggressive use of NSEC by default. Implementations SHOULD provide a configuration switch to disable aggressive use of NSEC and allow it to be enabled or disabled per domain.

The validating resolver needs to check the existence of an NSEC RR matching/covering the source of synthesis and an NSEC RR covering the query name.

If the validating resolver's cache contains an NSEC RR covering the source of synthesis and the covering NSEC RR of the query name, the resolver may respond with NXDOMAIN error immediately.

5.3. NSEC3

NSEC3 aggressive negative caching is more difficult. If the zone is signed with NSEC3, the validating resolver needs to check the existence of non-terminals and wildcards which derive from query names.

If the validating resolver's cache contains an NSEC3 RR matching the closest encloser, an NSEC3 RR covering the next closer name, and an NSEC3 RR covering the source of synthesis, it is possible for the resolver to respond with NXDOMAIN immediately.

If a covering NSEC3 RR has Opt-Out flag, the covering NSEC3 RR does not prove the non-existence of the domain name and the aggressive negative caching is not possible for the domain name.

A validating resolver implementation MAY support aggressive use of NSEC3. If it does aggressive use of NSEC3, it SHOULD provide a configuration switch to disable aggressive use of NSEC3 and allow it to be enabled or disabled for specific zones.

5.4. Wildcard

The last paragraph of RFC 4035 Section 4.5 discusses aggressive use of a cached deduced wildcard (as well as aggressive use of NSEC) and recommends that it is not relied upon.

Just like the case for the aggressive use of NSEC discussed in this draft, we revise this recommendation. As long as the resolver knows a name would not exist without the wildcard match, it can answer a query for that name using the cached deduced wildcard, and it may be justified for performance and other benefits.

Such aggressive use of cached deduced wildcard can be employed independently from aggressive use of NSEC. But, it will be more effective when both are enabled since the resolver can determine the name subject to wildcard would not otherwise exist more efficiently.

Furthermore, when aggressive use of NSEC is enabled, the aggressive use of cached deduced wildcard will be more effective.

An implementation MAY support aggressive use of wildcards. It SHOULD provide a configuration switch to disable aggressive use of wildcards.

5.5. Consideration on TTL

The TTL value of negative information is especially important, because newly added domain names cannot be used while the negative information is effective. Section 5 of RFC 2308 states that the maximum number of negative cache TTL value is 3 hours (10800). It is RECOMMENDED that resolvers limit the maximum effective TTL value of negative responses (NSEC/NSEC3 RRs) to this same value.

6. Benefits

The techniques described in this document provide a number of benefits, including (in no specific order):

Latency By answering directly from cache, recursive resolvers can immediately inform clients that the name they are looking for does not exist, improving the user experience.

Decreased recursive server load By answering negative queries from the cache, recursive servers avoid having send a query and wait for a response. In addition to decreasing the bandwidth used, it also means that the server does not need to allocate and maintain state, thereby decreasing memory and CPU load.

Decreased authorative server load Because recursive servers can answer (negative) queries without asking the authoritative server, the authoritative servers receive less queries. This decreases the authoritative server bandwidth, queries per second and CPU utilization.

The scale of the benefit depends upon multiple factors, including the query distribution. For example, currently around 65% of queries to Root Name servers result in NXDOMAIN responses; this technique will eliminate a sizable quantity of these.

[Editor note: There has been some discussion on if this document should discuss this attack and mitigation. The authors think that this is useful / important, but some participants feel that it oversells the DoS mitigation benefit. Please let us know if the below is helpful. Also, the below description is not as clear as it could be - it's been tricky to balance readability, correctness and conciseness. Text gratefully accepted...]

The technique described in this document may also mitigate so-called "random QNAME attacks", in which attackers send many queries for random sub-domains to recursive resolvers. As the recursive server will not have the answers cached it has to ask the authoritative servers for each random query, leading to a DoS on the authoritative (and often recursive) servers. Aggressive NSEC may help mitigate these attacks by allowing the recursive to answer directly from cache for any random queries which fall within already requested ranges. The effectiveness of this depends upon a number of factors, including if the attacker is making his queries through recursive resolvers (e.g to hide his source), the number of entries in the zone, the TTL, if the zone is using NSEC, if the attacker is setting the CD bit, etc. In the ideal case, authoritative servers under attack will need to answer somewhere between number_of_entries_in_zone queries and 2 * number_of_entries_in_zone queries from each recursive server. This is because there are as many "holes" between labels as there are labels in a zone. If the random query falls in range for which recursive server does not have an NSEC record cached, it will send a query to the authoritative server, and so it will send approximately the same number of queries as there are "holes" between entries. If the random queries happen to be for names which exist in the zone, the recursive will send those as well.

7. Update to RFC 4035

Section 4.5 of [RFC4035] shows that "In theory, a resolver could use wildcards or NSEC RRs to generate positive and negative responses (respectively) until the TTL or signatures on the records in question expire. However, it seems prudent for resolvers to avoid blocking new authoritative data or synthesizing new data on their own. Resolvers that follow this recommendation will have a more consistent view of the namespace".

The paragraph is updated as follows:

```
+--------------------------------------------------------------+
|  Once the records are validated, DNSSEC enabled recursive    |
|  resolvers MAY use wildcards and NSEC/NSEC3 resource records |
|  to generate (positive and) negative responses until their   |
|  effective TTLs or signatures for those records expire.      |
+--------------------------------------------------------------+
```

8. IANA Considerations

This document has no IANA actions.

9. Security Considerations

Newly registered resource records may not be used immediately. However, choosing suitable TTL value and negative cache TTL value (SOA MINIMUM field) will mitigate the delay concern, and it is not a security problem.

It is also suggested to limit the maximum TTL value of NSEC / NSEC3 resource records in the negative cache to, for example, 10800 seconds (3hrs), to mitigate this issue. Implementations which comply with this proposal are recommended to have a configurable maximum value of NSEC RRs in the negative cache.

Aggressive use of NSEC / NSEC3 resource records without DNSSEC validation may cause security problems. It is highly recommended to apply DNSSEC validation.

10. Implementation Status

Unbound supports aggressive negative caching.

11. Acknowledgments

The authors gratefully acknowledge DLV [RFC5074] author Samuel Weiler and the Unbound developers.

The authors would like to specifically thank Tatuya JINMEI for extensive review and comments, and also Mark Andrews, Stephane Bortzmeyer, Casey Deccio, Alexander Dupuy, Olafur Gudmundsson, Bob Harold, Shumon Huque, Pieter Lexis and Matthijs Mekking.

12. Change History

RFC Editor: Please remove this section prior to publication.

-01 to -02:

- Added Section 6 - Benefits (as suggested by Jinmei).
- Removed Appendix B (Jinmei)
- Replaced "full-service" with "validating" (where applicable)
- Integrated other comments from Jinmei from https://www.ietf.org/mail-archive/web/dnsop/current/msg17875.html
- Integrated comment from co-authors, including re-adding parts of Appendix B, terminology, typos.
- Tried to explain under what conditions this may actually mitigate attacks.

-00 to -01:

- Comments from DNSOP meeting in Berlin.
- Changed intended status to Standards Track (updates RFC 4035)
- Added a section "Updates to RFC 4035"
- Some language clarification / typo / cleanup
- Cleaned up the TTL section a bit.
- Removed Effects section, Additional proposal section, and pseudo code.
- Moved "mitigation of random subdomain attacks" to Appendix.

From draft-fujiwara-dnsop-nsec-aggressiveuse-03 -> draft-ietf-dnsop-nsec-aggressiveuse

- Document adopted by DNSOP WG.
- Adoption comments
- Changed main purpose to performance
- Use NSEC3/Wildcard keywords
- Improved wordings (from good comments)
- Simplified pseudo code for NSEC3
- Added Warren as co-author.

- Reworded much of the problem statement
- Reworked examples to better explain the problem / solution.

12.1. Version draft-fujiwara-dnsop-nsec-aggressiveuse-01

- Added reference to DLV [RFC5074] and imported some sentences.
- Added Aggressive Negative Caching Flag idea.
- Added detailed algorithms.

12.2. Version draft-fujiwara-dnsop-nsec-aggressiveuse-02

- Added reference to [I-D.vixie-dnsext-resimprove]
- Added considerations for the CD bit
- Updated detailed algorithms.
- Moved Aggressive Negative Caching Flag idea into Additional Proposals

12.3. Version draft-fujiwara-dnsop-nsec-aggressiveuse-03

- Added "Partial implementation"
- Section 4,5,6 reorganized for better representation
- Added NODATA answer in Section 4
- Trivial updates
- Updated pseudo code

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2308] Andrews, M., "Negative Caching of DNS Queries (DNS NCACHE)", RFC 2308, DOI 10.17487/RFC2308, March 1998, <http://www.rfc-editor.org/info/rfc2308>.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <http://www.rfc-editor.org/info/rfc4035>.

[RFC4592] Lewis, E., "The Role of Wildcards in the Domain Name System", RFC 4592, DOI 10.17487/RFC4592, July 2006, <http://www.rfc-editor.org/info/rfc4592>.

[RFC5074] Weiler, S., "DNSSEC Lookaside Validation (DLV)", RFC 5074, DOI 10.17487/RFC5074, November 2007, <http://www.rfc-editor.org/info/rfc5074>.

[RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, DOI 10.17487/RFC5155, March 2008, <http://www.rfc-editor.org/info/rfc5155>.

[RFC7719] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", RFC 7719, DOI 10.17487/RFC7719, December 2015, <http://www.rfc-editor.org/info/rfc7719>.

13.2. Informative References

[I-D.ietf-dnsop-nxdomain-cut]
Bortzmeyer, S. and S. Huque, "NXDOMAIN really means there is nothing underneath", draft-ietf-dnsop-nxdomain-cut-03 (work in progress), May 2016.

[I-D.vixie-dnsext-resimprove]
Vixie, P., Joffe, R., and F. Neves, "Improvements to DNS Resolvers for Resiliency, Robustness, and Responsiveness", draft-vixie-dnsext-resimprove-00 (work in progress), June 2010.

Appendix A. Detailed implementation notes

- 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. (Imported from Section 6 of [RFC5074]).

- Implementing aggressive negative caching suggests that a validator will need to build an ordered data structure of NSEC and NSEC3 records for each signer domain name of NSEC / NSEC3 records in order to efficiently find covering NSEC / NSEC3 records. Call the table as NSEC_TABLE. (Imported from Section 6.1 of [RFC5074] and expanded.)

- The aggressive negative caching may be inserted at the cache lookup part of the recursive resolvers.

- If errors happen in aggressive negative caching algorithm, resolvers MUST fall back to resolve the query as usual. "Resolve the query as usual" means that the resolver must process the query as though it does not implement aggressive negative caching.

Authors' Addresses

Kazunori Fujiwara
Japan Registry Services Co., Ltd.
Chiyoda First Bldg. East 13F, 3-8-1 Nishi-Kanda
Chiyoda-ku, Tokyo 101-0065
Japan

Phone: +81 3 5215 8451
Email: fujiwara@jprs.co.jp

Akira Kato
Keio University/WIDE Project
Graduate School of Media Design, 4-1-1 Hiyoshi
Kohoku, Yokohama 223-8526
Japan

Phone: +81 45 564 2490
Email: kato@wide.ad.jp

Warren Kumari
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
US

Email: warren@kumari.net