Network Working Group
Internet-Draft
Updates: 4035 (if approved)
Intended status: Standards Track
Expires: April 10, 2017

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

October 7, 2016

Aggressive use of NSEC/NSEC3
draft-ietf-dnsop-nsec-aggressiveuse-04

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 allow DNSSEC validating resolvers to generate negative answers within a range, and positive answers from wildcards. 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 validating resolvers to generate negative based upon NSEC/NSEC3 records (and positive answers in the presence of wildcards).

[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.]

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 10, 2017.

Copyright Notice

Table of Contents

1. Introduction 3
2. Terminology 3
3. Problem Statement 3
4. Background 4
5. Aggressive Caching 5
5.1. NSEC 6
5.2. NSEC3 6
5.3. Wildcards 7
5.4. Consideration on TTL 7
6. Benefits 8
7. Update to RFC 4035 8
8. IANA Considerations 9
9. Security Considerations 9
10. Implementation Status 9
11. Acknowledgments 10
11.1. Change History 10
11.1.1. Version draft-fujiwara-dnsop-nsec-aggressiveuse-01 . 12
11.1.2. Version draft-fujiwara-dnsop-nsec-aggressiveuse-02 . 12
11.1.3. Version draft-fujiwara-dnsop-nsec-aggressiveuse-03 . 13
11.2. new section 13
12. References 13
12.1. Normative References 13
12.2. Informative References 14
Appendix A. Detailed implementation notes 14
Authors' Addresses 15

1. Introduction

A DNS negative cache 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 synthesize negative answers from the information they have in the cache. This allows validating 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. It also allows the synthesis of positive answers in the presence of wildcard records.

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].

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 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
*.example.com        IN A 192.0.2.3
zebra.example.com    IN A 192.0.2.4
```

If a validating resolver receives a query for cat.example.com, it contacts its resolver (which may be itself) to query the example.com servers and will get back an NSEC record starting that there are no records (alphabetically) between apple and elephant, or an NSEC3 record stating there is nothing between two hashed names. The resolver then knows that cat.example.com does not exist; however, it 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 validating resolver 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.

Further, if a query is received for lion.example.com, it contacts its resolver (which may be itself) to query the example.com servers and will get back an NSEC record stating that there are no records (alphabetically) between elephant and zebra (or an NSEC3 record stating there is nothing between two hashed names), as well as an answer for lion.example.com, with the label count of the signature set to two (see [RFC7129], section 5.3 for more details).

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 dog.example.com it would receive a (signed) NSEC record stating that there are no labels between "apple" and "elephant" (or, for NSEC3, a similar pair of hashed names). This is a signed, cryptographic proof that these names are the ones before and after the queried for label. As dog.example.com falls within this range, the recursive server knows that dog.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 validating server receives that fall within the range covered by the record (for the TTL for the record). This document also specifies that a positive answer should be generated for any queries that the validating server receives that are proven to be covered by a wildcard record.

Section 4.5 of [RFC4035] says:

"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." and "The reason for these recommendations is that, between the initial query and the expiration of the data from the cache, the authoritative data might have been changed (for example, via dynamic update).". In other words, if a resolver generates negative answers from an NSEC record, it will not send any queries for names within that NSEC range (for the TTL). If a new name is added to the zone during this interval the resolver will not know this. Similarly, if the resolver is generating responses from a wildcard record, it will continue to do so (for the

We believe this recommendation can be relaxed because, in the absense of this technique, a lookup for the exact name could have come in during this interval, and so a negative answer could already be cached (see [RFC2308] for more background). This means that zone operators should have no expectation that an added name would work immediately. With DNSSEC and Aggressive NSEC, the TTL of the NSEC record is the authoritative statement of how quickly a name can start working within a zone.

5. Aggressive Caching

Section 4.5 of [RFC4035] says 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 wildcards and NSEC/NSEC3 resource records |
|  to generate positive and negative responses until the       |
|  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.

5.1. NSEC

Implementations which support aggressive use of NSEC SHOULD enable this by default. Implementations MAY 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 denial of existence can be determined according to the rules set out in Section 5.4 of [RFC4035], using NSEC records in the cache, then the resolver can immediately return an NXDOMAIN or NODATA (as appropriate) response.

5.2. NSEC3

NSEC3 aggressive negative caching is more difficult than NSEC aggressive caching. 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.

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

If denial of existence can be determined according to the rules set out in [RFC5155] sections 8.4, 8.5, 8.6, 8.7,using NSEC3 records in the cache, then the resolver can immediately return an NXDOMAIN or NODATA response (as appropriate).

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.

5.3. Wildcards

The last paragraph of [RFC4035] Section 4.5 also discusses the use of wildcards and NSEC RRs to generate positive responses and recommends that it not be relied upon. Just like the case for the aggressive use of NSEC/NSEC3 for negative answers, we revise this recommendation.

As long as the validating resolver can determine that a name would not exist without the wildcard match, it MAY synthesize an answer for that name using the cached deduced wildcard. If the corresponding wildcard record is not in the cache, it MUST fall back to send the query to the authoritative DNS servers.

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

5.4. 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 [RFC2308] states that the maximum number of negative cache TTL value is 3 hours (10800). It is RECOMMENDED that validating resolvers limit the maximum effective TTL value of negative responses (NSEC/NSEC3 RRs) to this same value.

Section 5 of [RFC2308]also states that a negative cache entry TTL is taken from the minimum of the SOA.MINIMUM field and SOA's TTL. This can be less than the TTL of an NSEC or NSEC3 record, since their TTL is equal to the SOA.MINIMUM field (see [RFC4035]section 2.3 and [RFC5155] section 3.)

A resolver that supports aggressive use of NSEC and NSEC3 should reduce the TTL of NSEC and NSEC3 records to match the TTL of the SOA record in the authority section of a negative response, if the SOA TTL is smaller.

6. Benefits

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

Reduced latency: By answering directly from cache, validating 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, validating servers avoid having to 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 fewer 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, at the time of this writing, around 65% of queries to Root Name servers result in NXDOMAIN responses (see statis [root-servers.org]); this technique will eliminate a sizable quantity of these.

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 resolvers. As the resolver will not have the answers cached, it has to ask external servers for each random query, leading to a DoS on the authoritative servers (and often resolvers). Aggressive NSEC may help mitigate these attacks by allowing the resolver to answer directly from cache for any random queries which fall within already requested ranges. It will not always work as an effective defense, not least because not many zones are DNSSEC signed at all -- but it will still provide an additional layer of defense.

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 validating   |
|  resolvers MAY use wildcards and NSEC/NSEC3 resource records |
|  to generate positive and negative responses until the       |
|  effective TTLs or signatures for those records expire.      |
+--------------------------------------------------------------+
```

8. IANA Considerations

This document has no IANA actions.

9. Security Considerations

Use of NSEC / NSEC3 resource records without DNSSEC validation may create serious security issues, and so this technique requires DNSSEC validation.

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.

Although the TTL of NSEC/NSEC3 records is typically fairly short (minutes or hours), their RRSIG expiration time can be much further in the future (weeks). An attacker who is able to successfully spoof responses might poison a cache with old NSEC/NSEC3 records. If the resolver is NOT making aggressive use of NSEC/NSEC3, the attacker has to repeat the attack for every query. If the resolver IS making aggressive use of NSEC/NSEC3, one successful attack would be able to suppress many queries for new names, up to the negative TTL.

10. Implementation Status

Unbound currenty implements aggressive negative caching, as does Google Public DNS.

11. Acknowledgments

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

The authors would like to specifically thank Stephane Bortzmeyer, Tony Finch, Tatuya JINMEI for extensive review and comments, and also Mark Andrews, Casey Deccio, Alexander Dupuy, Olafur Gudmundsson, Bob Harold, Shumon Huque, John Levine, Pieter Lexis and Matthijs Mekking (who even sent pull requests!).

11.1. Change History

RFC Editor: Please remove this section prior to publication.

-03 to -04:

- o Working group does want the "positive" answers, not just negative ones. This requires readding what used to be Section 7, and a bunch of cleanup, including:

  * Additional text in the Problem Statement

  * Added a wildcard record to the zone.

  * Added "or positive answers from wildcards" type text (where appropriate) to explain that this isn't just for negative answers.

  * Reworded much of the Wildcard text.

- o Incorporated pull request from Tony Finch (thanks!): https://github.com/wkumari/draft-ietf-dnsop-nsec-aggressiveuse/pull/1

- o More fixups from Tony (including text): https://www.ietf.org/mail-archive/web/dnsop/current/msg18271.html. This included much clearer text on TTL, refernces to the NSEC / NSEC3 RFCs (instead of my clumsy summary), good text on replays, etc.

- o Converted the "zone file" to a figure to make it more readable.

- o Text from Tim W: "If a validating resolver receives a query for cat.example.com, it contacts its resolver (which may be itself) to query..." - which satisfies Jinmei's concern (which I was too dense to grock).

- o Fixup of the "validation required" in security considerations.

-02 to -03:

o Integrated a bunch of comments from Matthijs Mekking - details in: https://github.com/wkumari/draft-ietf-dnsop-nsec-aggressiveuse/pull/1. I decided to keep "Aggressive Negative Caching" instead of "Aggressive USE OF Negative Caching" for readability.

o Attempted to address Bob Harold's comment on the readability issues with "But, it will be more effective when both are enabled..." in Section 5.4 - https://www.ietf.org/mail-archive/web/dnsop/current/msg17997.html

o MAYs and SHOULD drifted in the text block. Fixed - thanks to https://mailarchive.ietf.org/arch/msg/dnsop/2ljmmzxtIMCFMLOZmWcSbTYVOy4

o A number of good edits from Stephane in: https://www.ietf.org/mail-archive/web/dnsop/current/msg18109.html

o A bunch more edits from Jinmei, as in: https://www.ietf.org/mail-archive/web/dnsop/current/msg18206.html

-01 to -02:

o Added Section 6 - Benefits (as suggested by Jinmei).

o Removed Appendix B (Jinmei)

o Replaced "full-service" with "validating" (where applicable)

o Integrated other comments from Jinmei from https://www.ietf.org/mail-archive/web/dnsop/current/msg17875.html

o Integrated comment from co-authors, including re-adding parts of Appendix B, terminology, typos.

o Tried to explain under what conditions this may actually mitigate attacks.

-00 to -01:

o Comments from DNSOP meeting in Berlin.

o Changed intended status to Standards Track (updates RFC 4035)

o Added a section "Updates to RFC 4035"

o 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.

11.1.1. 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.

11.1.2. 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

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

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

11.2. new section

12. References

12.1. 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>.

[RFC7129] Gieben, R. and W. Mekking, "Authenticated Denial of Existence in the DNS", RFC 7129, DOI 10.17487/RFC7129, February 2014, <http://www.rfc-editor.org/info/rfc7129>.

[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>.

12.2. 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.

[root-servers.org]
IANA, "Root Server Technical Operations Assn", <http://www.root-servers.org/>.

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