dnsop P. Wouters
Internet-Draft Red Hat
Intended status: Standards Track October 27, 2014
Expires: April 30, 2015

Chain Query requests in DNS
draft-ietf-dnsop-edns-chain-query-01

Abstract

This document defines an EDNS0 extension that can be used by a DNSSEC enabled Recursive Nameserver configured as a forwarder to send a single DNS query requesting to receive a complete validation path along with the regular DNS answer, without the need to rapid-fire many UDP requests in an attempt to attain a low latency.

Table of Contents

1. Introduction

Traditionally, clients operate in stub-mode for DNS. For each DNS question the client needs to resolve, it sends a single query to an upstream DNS resolver to obtain a single DNS answer. When DNSSEC [RFC4033] is deployed on such clients, validation requires that the client obtains all the (intermediate) information from the DNS root down to the queried-for hostname so it can perform DNSSEC validation on the complete chain of trust. This process increases the number of DNS queries and answers required, and thus increases the latency before a validated DNS answer has been obtained.

Currently, applications use a rapid-fire approach to send out many UDP requests concurrently. This requires more resources on the DNS client with respect to state (cpu, memory, battery) and bandwidth. There is also no guarantee that the initial burst of UDP questions will result in all the records required for DNSSEC validation, and

more round trips could be required depending on the resulting DNS answers. This especially affects high-latency links.

This document specifies an EDNS0 extension that allows a validating recursive name server running as a forwarder to request another recursive name server for a DNS chain answer using one DNS query/ answer pair. This reduces the number of round-trip times ("RTT") to two. If combined with [TCP-KEEPALIVE] there is only 1 RTT. While the upstream DNS resolver still needs to perform all the individual queries required for the complete answer, it usually has a much bigger cache and does not experience significant slowdown from last-mile latency.

This EDNS0 extension allows the Forwarder to indicate which part of the DNS hierarchy it already contains in its cache. This reduces the amount of data required to be transferred and reduces the work the upstream Resolving Nameserver has to perform.

This EDNS0 extension is only intended for Forwarders. It can (and should be) ignored by Authoritative Nameservers and by Recursive Nameservers that do not support this EDNS0 option.

1.1. Requirements Notation

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 [RFC2119].

2. Terminology

Stub Resolver: A simple DNS protocol implementation on the client side as described in [RFC1034] section 5.3.1.

Authoritative Nameserver: A nameserver that has authority over one or more DNS zones. These are normally not contacted by clients directly but by Recursive Resolvers. Described in [RFC1035] chapter 6.

Recursive Resolver: A nameserver that is responsible for resolving domain names for clients by following the domain's delegation chain, starting at the root. Recursive Resolvers frequently use caches to be able to respond to client queries quickly. Described in [RFC1035] chapter 7.

Validating Resolver: A recursive nameserver that also performs DNSSEC [RFC4033] validation.

Forwarder: A Recursive Resolver that is using another (upstream) Recursive Resolver instead of querying Authoritative Nameservers directly. It still performs validation.

3. Overview

When DNSSEC is deployed on the client, it can no longer delegate all DNS work to the upstream Resolving Nameserer. Obtaining just the DNS answer itself is not enough to validate that answer using DNSSEC. For DNSSEC validation, the client requires a locally running validating DNS server configured as Resolving Nameserver so it can confirm DNSSEC validation of all intermediary DNS answers. It can configure itself as a Forwarder if the DHCP server has indicated that one or more Resolving Nameservers are available. Regardless, generating the required queries for validation adds a significant delay in answering the DNS question of the locally running application. The application has to wait while the Forwarder on the client is querying for all the intermediate work. Each round-trip adds to the total time waiting on DNS resolving to complete. This makes DNSSEC resolving impractical on networks with a high latency.

The edns-chain-query option allows the client to request all intermediate DNS data it requires to resolve and validate a particular DNS answer in a single round-trip DNS query and answer.

Servers answering with chain query data exceeding 512 bytes should ensure that the transport is TCP or source IP address verified UDP. See Section 8. This avoids abuse in DNS amplification attacks.

The format of this option is described in Section 4.

As described in Section 5.3, a recursive nameserver could use this EDNS0 option to include additional data required by the client in the Authority Section of the DNS answer packet when using a source IP verified transport. The Answer Section remains unchanged from a traditional DNS answer and contains the answer and related DNSSEC entries.

An empty edns-chain-query EDNS0 option MAY be sent over any transport as a discovery method. A DNS server receiving such an empty edns-chain-query option SHOULD add an empty edns-chain-query option in its answer to indicate that it supports edns-chain-query for source IP address verified transports.

The mechanisms provided by edns-chain-query raise various security related concerns, related to the additional work, bandwidth, amplification attacks as well as privacy issues with the cache. These concerns are described in Section 8.

4. Option Format

This draft uses an EDNS0 ([RFC2671]) option to include client IP information in DNS messages. The option is structured as follows:

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-------------------------------+-------------------------------+
!         OPTION-CODE           !         OPTION-LENGTH         !
+-------------------------------+-------------------------------+
~                Last Known Query Name (FQDN)                   ~
+---------------------------------------------------------------+
```

- (Defined in [RFC2671]) OPTION-CODE, 2 octets, for edns-chain-query is [TBD].

- (Defined in [RFC2671]) OPTION-LENGTH, 2 octets, contains the length of the payload (everything after Option-length) in octets.

- Last Known Query Name, a variable length FDQN of the requested start point of the chain. This entry is the 'lowest' known entry in the DNS chain known by the recursive server seeking a edns-chain-query answer. The end point of the chain is obtained from the DNS Query Section itself. No compression is allowed for this value.

- Assigned by IANA in IANA-AFI [1].

5. Protocol Description

5.1. Discovery of Support

A Forwarder may include a zero-length edns-chain-query option in queries over UDP or TCP to discover the DNS server capability for edns-chain-query. DNS Servers that support and are willing to accept chain queries over TCP SHOULD respond to a zero-length edns-chain-query received over UDP or TCP queries by including a zero-length edns-chain-query option in the answer. A Forwarder MAY then switch to the TCP transport and sent a non-zero edns-chain-query value to request a chain-query response from the DNS server.

5.2. Generating a Query

The edns-chain-query option should generally be deployed by Forwarders, as described in Section 5.4.

In this option value, the Forwarder sets the last known entry point in the chain - furthest from the root - that it already has a DNSSEC validated (secure or not) answer for in its cache. The upstream Recursive Resolver does not need to include any part of the chain from the root down to this option's FQDN. A complete example is described in Section 9.

Depending on the size of the labels of the last known entry point value, a DNS Query packet could be arbitrarily large. If using the last known entry point would result in a query size of more then 512 bytes, the last known entry point should be replaced with its parent entry until the query size would be 512 bytes or less.

5.3. Generating a Response

When a query containing a non-zero edns-chain-query option is received over a TCP connection from a Forwarder, the upstream Recursive Resolver supporting edns-chain-query MAY respond by confirming that it is returning a DNS Query Chain. To do so, it MUST set the edns-chain-query option with an OPTION-LENGTH of zero to indicate the DNS answer contains a Chain Query. It extends the Authority Section for the DNS answer packet with the required DNS RRSets resulting in an Authority Section that contains a complete chain of DNS RRsets that start with the first chain element below the received Last Known Query Name upto and including the NS and DS RRsets that represent the zone cut (authoritative servers) of the QNAME. The actual DNS answer to the question in the Query Section is placed in the DNS Answer Section identical to traditional DNS answers. If the received query has the DNSSEC OK flag set, all required DNSSEC related records must be added to their appropriate sections. This includes records required for proof of non-existence of regular and/or wildcard records, such as NSEC or NSEC3 records.

Recursive Resolvers that have not implemented or enabled support for the edns-chain-query option, or are otherwise unwilling to perform the additional work for a Chain Query due to work load, may safely ignore the option in the incoming queries. Such a server MUST NOT include an edns-chain-query option when sending DNS answer replies back, thus indicating it is not able to support Chain Queries at this time.

Requests with wrongly formatted options (i.e. bogus FQDN) MUST be rejected and a FORMERR response must be returned to the sender, as described by [RFC2671], Transport Considerations.

Requests resulting in chains that the receiving resolver is unwilling to serve can be rejected by sending a REFUSED response to the sender, as described by [RFC2671], Transport Considerations. This refusal

can be used for chains that would be too big or chains that would reveal too much information considered private.

At any time, a DNS server that has determined that it is running low on resources can refuse to acknowledge a Chain Query by omitting the edns-chain-query option. It may do so even if it conveyed support to a DNS client previously. If [TCP-KEEPALIVE] is used, it may even change its support for edns-chain-query within the same TCP session.

If the DNS request results in an CNAME or DNAME for the Answer Section, the DNS server MUST return these records in the Answer Section similar to regular DNS processing. The CNAME or DNAME target MAY be placed in the Additional Section only if all supporting records for DNSSEC validation of the CNAME or DNAME target is also added to the Authority Section.

In any case, the response from the receiving resolver to the client resolver MUST NOT contain the edns-chain-query option if none was present in the client's resolver original request.

5.4. Sending the Option

When edns-chain-query is available, the downstream Resolving Nameserver can adjust its query strategy based on the desired queries and its cache contents.

A Forwarder can request the edns-chain-query option with every outgoing DNS query. However, it is RECOMMENDED that Forwarders remember which upstream Resolving Nameservers did not return the option (and additional data) with their response. The Forwarder SHOULD fallback to regular DNS for subsequent queries to those Recursive Nameservers. It MAY switch to another Resolving Nameserver that does support the edns-chain-query option or try again later to see if the server has become less loaded and is now willing to answer with Query Chains.

6. Protocol Considerations

6.1. DNSSEC Considerations

The presence or absence of an OPT resource record containing an edns-chain-query option in a DNS query does not change the usage of those resource records and mechanisms used to provide data origin authentication and data integrity to the DNS, as described in [RFC4033], [RFC4034] and [RFC4035].

6.2. NS record Considerations

edns-chain-query responses MUST include the NS RRset from the child zone, which includes DNSSEC RRSIG records required for validation.

When a DNSSEC chain is supplied via edns-chain-query, the Forwarder no longer requires to use the NS RRset, as it can construct the validation path via the DNSKEY and DS RRsets without using the NS RRset. However, it is prefered that the Forwarder can populate its cache with this information regardless, to avoid requiring queries in the future just to obtain the missing NS records. This can happen on a roaming device that needs to swich from using a DHCP obtained DNS server as forwarder to running in full autonomous resolver mode, for example when the DHCP obtained DNS server is broken in some way.

6.3. TCP Session Management

It is recommended that TCP Chain Queries are used in combination with [TCP-KEEPALIVE].

Both DNS clients and servers are subject to resource constraints which will limit the extent to which TCP Chain Queries can be executed. Effective limits for the number of active sessions that can be maintained on individual clients and servers should be established, either as configuration options or by interrogation of process limits imposed by the operating system.

In the event that there is greater demand for TCP Chain Queries than can be accommodated, DNS servers may stop advertising the edns-query-chain option in successive DNS messages. This allows, for example, clients with other candidate servers to query to establish new TCP sessions with different servers in expectation that those servers might still allow TCP Chain Queries.

6.4. Non-Clean Paths

Many paths between DNS clients and servers suffer from poor hygiene, limiting the free flow of DNS messages that include particular EDNS0 options, or messages that exceed a particular size. A fallback strategy similar to that described in [RFC6891] section 6.2.2 SHOULD be employed to avoid persistent interference due to non-clean paths.

6.5. Anycast Considerations

DNS servers of various types are commonly deployed using anycast [RFC4786].

Successive DNS transactions between a client and server using UDP transport may involve responses generated by different anycast nodes, and the use of anycast in the implementation of a DNS server is

effectively undetectable by the client. The edns-chain-query option SHOULD NOT be included in responses using UDP transport from servers provisioned using anycast unless all anycast server nodes are capable of processing the edns-query-chain option.

Changes in network topology between clients and anycast servers may cause disruption to TCP sessions making use of edns-chain-query more often than with TCP sessions that omit it, since the TCP sessions are expected to be longer-lived. Anycast servers MAY make use of TCP multipath [RFC6824] to anchor the server side of the TCP connection to an unambiguously-unicast address in order to avoid disruption due to topology changes.

7. Implementation Status

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC6982]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC6982], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

[While there is some interest, no work has started yet]

8. Security Considerations

8.1. Amplification Attacks

Chain Queries can potentially send very large DNS answers. Attackers could abuse this using spoofed source IP addresses to inflict large Distributed Denial of Service attacks using query-chains as an amplification vector in their attack. While TCP is not vulnerable for this type of abuse, the UDP protocol is vulnerable to this.

A recursive nameserver MUST NOT return Query Chain answers to clients over UDP without source IP address verification, for instance using [EASTLAKE-COOKIES]. A recursive nameserver SHOULD signal support in response to a Query Chain request over UDP by responding using a zero-length edns-chain-query option over UDO even without source IP address verification.

9. Examples

9.1. Simple Query for example.com

1. A web browser on a client machine asks the Forwarder running on localhost to resolve the A record of "www.example.com." by sending a regular DNS UDP query on port 53 to 127.0.0.1.

2. The Forwarder on the client machine checks its cache, and notices it already has a DNSSEC validated entry of "com." in its cache. This includes the DNSKEY RRset with its RRSIG records. In other words, according to its cache, ".com" is DNSSEC validated as "secure" and can be used to continue a DNSSEC validated chain on.

3. The Forwarder on the client opens a TCP connection to its upstream Recursive Resolver on port 53. It adds the edns-chain-query option as follows:

   * Option-code, set to [TBD]

   * Option-length, set to 0x00 0x04

   * Last Known Query Name set to "com."

4. The upstream Recursive Resolver receives a DNS query over TCP with the edns-chain-query Last Known Query Name set to "com.". After accepting the query it starts constructing a DNS reply packet.

5. The upstream Recursive Resolver performs all the regular work to ensure it has all the answers to the query for the A record of "www.example.com.". It does so without using the edns-chain-query option - unless it is also configured as a Forwarder. The answer to the original DNS question could be the actual A record, the DNSSEC proof of non-existence, or an insecure NXDOMAIN response.

6. The upstream Recursive Resolver adds the edns-chain-query option to the DNS answer reply as follows:

   * Option-code, set to [TBD]

   * Option-length, set to 0x00 0x00

   * The Last Known Query Name is ommited (zero length)

7. The upstream Recursive Resolver constructs the DNS Authority Section and fills it with:

   * The DS RRset for "example.com." and its corresponding RRSIGs (made by the "com." DNSKEY(s))

   * The DNSKEY RRset for "example.com." and its corresponding RRSIGs (made by the "example.com" DNSKEY(s))

   * The authoritative NS RRset for "example.com." and its corresponding RRSIGs (from the child zone)

   If the answer does not exist, and the zone uses DNSSEC, it also adds the proof of non-existance, such as NSEC or NSEC3 records, to the Authority Section.

8. The upstream Recursive Resolver constructs the DNS Answer Section and fills it with:

   * The A record of "www.example.com." and its corresponding RRSIGs

   If the answer does not exist (no-data or NXDOMAIN), the Answer Section remains empty. For the NXDOMAIN case, the RCode of the DNS answer packet is set to NXDOMAIN. Otherwise it remains NOERROR.

9. The upstream Recursive Resolver returns the DNS answer over the existing TCP connection. When all data is sent, it SHOULD keep the TCP connection open to allow for additional incoming DNS queries - provided it has enough resources to do so.

10. The Forwarder receives the DNS answer. It processes the Authority Section and the Answer Section and places the information in its local cache. It ensures that no data is accepted into the cache without having proper DNSSEC validation. It MAY do so by looping over the entries in the Authority and Answer Sections. When an entry is validated for its cache, it is removed from the processing list. If an entry cannot be validated it is left in the process list. When the end of the list is reached, the list is processed again until either all entries are placed in the cache, or the remaining items cannot

be placed in the cache due to lack of validation. Those entries are then disgarded.

11. If the cache contains a valid answer to the application's query, this answer is returned to the application via a regular DNS answer packet. This packet MUST NOT contain an edns-chain-query option. If no valid answer can be returned, normal error processing is done. For example, an NXDOMAIN or an empty Answer Section could be returned depending on the error condition.

9.2. Out-of-path query for example.com

A Recursive Resolver receives a query for the A record for example.com. It includes the edns-chain-query option with the following parameters:

- Option-code, set to [TBD]
- Option-length, set to 0x00 0x0D
- The Last Known Query Name set to 'unrelated.ca.'

As there is no chain that leads from "unrelated.ca." to "example.com", the Resolving Nameserver answers with RCODE "FormErr". It includes the edns-chain-query with the following parameters:

- Option-code, set to [TBD]
- Option-length, set to 0x00 0x00
- The Last Known Query Name is ommited (zero length)

9.3. non-existent data

A Recursive Resolver receives a query for the A record for "ipv6.toronto.redhat.ca". It includes the edns-chain-query option with the following parameters:

- Option-code, set to [TBD]
- Option-length, set to 0x00 0x03
- The Last Known Query Name set to 'ca.'

Using regular UDP queries towards Authoritative Nameservers, it locates the NS RRset for "toronto.redhat.ca.". When querying for the A record it receives a reply with RCODE "NoError" and an empty Answer Section. The Authority Section contains NSEC3 and RRSIG records proving there is no A RRtype for the QNAME "ipv6.toronto.redhat.ca".

The Recursive Resolver constructs a DNS reply with the following edns-chain-query option parameters:

- Option-code, set to [TBD]
- Option-length, set to 0x00 0x00
- The Last Known Query Name is ommited (zero length)

The RCODE is set to "NoError". The Authority Section is filled in with:

- The DS RRset for "redhat.ca." plus RRSIGs
- The DNSKEY RRset for "redhat.ca." plus RRSIGs
- The NS RRset for "redhat.ca." plus RRSIGs (eg ns[01].redhat.ca)
- The A RRset for "ns0.redhat.ca." and "ns1.redhat.ca." plus RRSIGs
- The DS RRset for "toronto.redhat.ca." plus RRSIGs
- The NS RRset for "toronto.redhat.ca." plus RRSIGs (eg ns[01].toronto.redhat.ca)
- The DNSKEY RRset for "toronto.redhat.ca." plus RRSIGs
- The A RRset and/or AAAA RRset for "ns0.toronto.redhat.ca." and "ns1.toronto.redhat.ca." plus RRSIGs
- The NSEC record for "ipv6.toronto.redhat.ca." (proves what RRTYPEs do exist, does not include A)
- The NSEC record for "toronto.redhat.ca." (proves no wildcard exists)

The Answer Section is empty. The RCode is set to NOERROR.

10. IANA Considerations

10.1. EDNS0 option code for edns-chain-query

IANA has assigned option code [TBD] in the "DNS EDNS0 Option Codes (OPT)" registry to edns-chain-query.

11. Acknowledgements

Andrew Sullivan pointed out that we do not need any new data formats to support DNS chains. Olafur Gudmundsson ensured the RRsets are returned in the proper Sections.

12. Normative References

[EASTLAKE-COOKIES] Eastlake, Donald., "Domain Name System (DNS) Cookies", draft-eastlake-dnsext-cookies (work in progress), October 2014.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

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

[RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, August 1999.

[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.

[RFC4786] Abley, J. and K. Lindqvist, "Operation of Anycast Services", BCP 126, RFC 4786, December 2006.

[RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, January 2013.

[RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, April 2013.

[RFC6982] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", RFC 6982, July 2013.

[TCP-KEEPALIVE] Wouters, P., "The edns-tcp-keepalive EDNS0 Option", draft-ietf-dnsop-edns-tcp-keeaplive (work in progress), October 2014.

Author's Address

Paul Wouters
Red Hat

Email: pwouters@redhat.com