Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 17, 2015

Z. Hu
L. Zhu
J. Heidemann
USC/Information Sciences Institute
A. Mankin
D. Wessels
Verisign Labs
P. Hoffman
VPN Consortium
April 15, 2015

TLS for DNS: Initiation and Performance Considerations
draft-hzhwm-dprive-start-tls-for-dns-02

Abstract

This document offers an approach to initiating TLS for DNS: use of a dedicated DNS-over-TLS port, and fallback to a mechanism for upgrading a DNS-over-TCP connection over the standard port (TCP/53) to a DNS-over-TLS connection. Encryption provided by TLS eliminates opportunities for eavesdropping on DNS queries in the network, such as discussed in RFC 7258. In addition it specifies two usage profiles for DNS-over-TLS. Finally, it provides advice on performance considerations to minimize overheads from using TCP and TLS with DNS, pertaining to both approaches.

Status of this Memo

Copyright Notice

1. Introduction

Today, nearly all DNS queries ([RFC1034] and [RFC1035]) are sent unencrypted, which makes them vulnerable to eavesdropping by an attacker that has access to the network channel, reducing the privacy of the querier. Recent news reports have elevated these concerns, and ongoing efforts are beginning to identify privacy concerns about DNS ([I-D.ietf-dprive-problem-statement]).

Prior work has addressed some aspects of DNS security, but until recently there has been little work on privacy between a DNS client and server. DNS Security Extensions (DNSSEC, [RFC4033]) provide _response integrity_ by defining mechanisms to cryptographically sign zones, allowing end-users (or their first-hop resolver) to verify replies are correct. By intention, DNSSEC does not protect request and response privacy. Traditionally, either privacy was not considered a requirement for DNS traffic, or it was assumed that network traffic was sufficiently private, however these perceptions are evolving due to recent events [RFC7258].

DNSCurve [draft-dempsky-dnscurve] defines a method to add confidentiality to the link between DNS clients and servers; however, it does so with a new cryptographic protocol and does not take advantage of an existing standard protocol such as TLS. ConfidentialDNS [draft-wijngaards-confidentialdns] and IPSECA [draft-osterweil-dane-ipsec] use opportunistic encryption to offer privacy for DNS queries and responses. Finally, others have suggested DNS-over-TLS. Unbound DNS software [unbound] includes a DNS-over-TLS implementation. The present document goes beyond past DNS-over-TLS discussions by providing two modes of initiation for DNS-over-TLS: use of a well-known port, and use of a negotiation mechanism in an established connection.

Protocol changes proposed here must consider potential interactions with middle boxes. The port-based initiation of TLS is very

straightforward, but might be blocked by firewalls or be unwelcome to some DNS client or server implementations. If port-based initiation of TLS fails, the negotiation mechanism allows DNS clients and servers to upgrade an existing DNS-over-TCP connection to a DNS-over-TLS connection, analogous to upgrade mechanisms in other uses of TLS, such as STARTTLS [RFC2595] used in SMTP [RFC3207], IMAP [RFC3501] and POP [RFC1939], to name just a few of many. Adding TLS to DNS-over-TCP avoids port blocking, but maybe interact poorly with middle boxes that inspect DNS traffic. As is generally the case with TLS, both approaches are subject to downgrade attacks, as discussed in Section 2.2.

The protocol described here works for any DNS client to server communication using DNS-over-TCP. There can be different profiles providing different levels of privacy, as discussed in Section 3. The protocol may be used for any DNS communication both from stub to recursive, and from recursive to authoritative servers, but different protocols may be preferable for different environments.

This document describes two profiles in Section 3 providing different levels of assurance of privacy: an opportunistic privacy profile and a pre-deployed profile.

1.1. Reserved Words

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. Protocol Changes

The only changes required for port-based DNS-over-TLS are those optimizing TCP and TLS performance discussed in the following. The DNS protocol itself is unchanged.

Clients and servers negotiate upgrade-based DNS-over-TLS by setting a bit in the Flags field of the EDNS0 [RFC6891] OPT meta-RR. The "TLS OK" (TO) bit is defined as the second bit of the third and fourth bytes of the "extended RCODE and flags" portion of the EDNS0 OPT meta-RR, immediately adjacent to the "DNSSEC OK" (DO) bit [RFC4033]:

```
                 +0 (MSB)                +1 (LSB)
            +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
         0: |   EXTENDED-RCODE      |       VERSION         |
            +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
         2: |DO|TO|                  Z                      |
            +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

2.1. Use by DNS Clients

DNS clients first try port-based DNS-over-TLS. If that connection fails, they try upgrade-based DNS-over-TLS.

2.1.1. Port-Based DNS-over-TLS for Clients

DNS clients SHOULD first try using port-based DNS-over-TLS by establishing the TCP connection to the dedicated port TBD (number to be defined in Section 5). Clients MAY try STARTTLS upgrade before the dedicated port if there is information that this ordering is preferred. It SHOULD be an implementation and/or local determination as to whether to attempt TLS via the dedicated port first and then fall back to STARTTLS use, or to choose some other order of attempts and fallbacks.

2.1.2. Sending Queries for Upgrade-Based DNS-over-TLS

Setting the TO bit in queries sent using UDP transport has no protocol meaning. However, the client MAY set the TO bit when using UDP transport. The server MUST ignore the TO bit when receiving UDP transport.

DISCUSSION: community advice is sought on this. The advantage of allowing a client to send UDP on TO is that servers can collect information on deployment (as happened with the DO bit). The disadvantage is that a meaningless bit (TO over UDP) might cause confusion, and some middleboxes might not pass a UDP query with the TO bit set.

DNS clients set the TO bit in the initial query sent to a server using TCP transport to signal their desire that the TCP connection be upgraded to TLS. DNS clients SHOULD NOT set the TO bit on queries when using TLS transport because doing so has no meaning in this protocol.

Since the motivation for upgrade-based DNS-over-TLS is to preserve privacy, DNS clients SHOULD use an initial (unprotected) query that reveals no private information in the initial TO=1 query to a server. To provide a standard "dummy" query, it is RECOMMENDED to send the initial query with RD=0, QNAME="STARTTLS", QCLASS=CH, and QTYPE=TXT ("STARTTLS/CH/TXT") analogous to administrative queries already in widespread use [RFC4892]. (For some profiles, the client MUST use a dummy query for the initial query.)

After sending the initial TO=1 query using TCP transport, DNS clients MUST wait for the initial response before sending any subsequent queries over the same TCP connection.

2.1.3. Receiving Responses for Upgrade-Based DNS-over-TLS

A DNS client that receives a response using UDP transport that has the TO bit set handles that response as usual. It MAY record the server's support for DNS-over-TLS and use that information as part of its server selection algorithm in the case where multiple servers are available to service a particular query.

A DNS client that has sent the TO bit using TCP transport and receives a response to its initial query that has the TO bit set MUST immediately initiate a TLS handshake using the procedure described in [RFC5246]. (Note that this document does not yet deal with what happens when the TLS handshake does not succeed.)

DISCUSSION: are there any cases in which a DNS client that sent TO on DNS-over-TCP and receives TO in the initial response from the server would not initiate the TLS handshake? Is there any reason for this to be SHOULD rather than MUST?

A DNS client that receives a response to its initial query using TCP transport that has the TO bit clear MUST not initiate a TLS handshake and SHOULD utilize the existing TCP connection for subsequent queries. DNS clients SHOULD remember server IP addresses that don't support upgrade-based DNS-over-TLS, including TLS handshake failures, and not request DNS-over-TLS from them for reasonable period (such as one hour per server).

2.1.4. Use by DNS Servers

A DNS server that supports DNS-over-TLS SHOULD support port-based DNS-over-TLS, and SHOULD support upgrade-based DNS-over-TLS.

2.1.4.1. Receiving Queries for Upgrade-Based DNS-over-TLS

A DNS server receiving a query over UDP with the TO bit ignores that bit. A DNS server receiving a query over an existing TLS connection with the TO bit ignores that bit.

A DNS server receiving an initial query over TCP that has the TO bit set MAY inform the client it is willing to establish a TLS session, as described in the next section.

A DNS server receiving subsequent queries over TCP MUST ignore the TO bit. (A client wishing to start TLS after the initial query MUST open a new TCP connection to do so.)

2.1.4.2. Sending Responses

A DNS server sending a response over UDP to a query that had an OPT meta-RR SHOULD set the TO bit to indicate its general support for DNS-over-TLS, as long as it is willing and able to support a TLS connection with the particular client.

A DNS server receiving an initial query over TCP that has the TO bit set MAY set the TO bit in its response. The server MUST then proceed with the TLS handshake protocol.

A DNS server receiving a "dummy" STARTTLS/CH/TXT query over TCP MUST respond with RCODE=0 and a TXT RR in the Answer section. Contents of the TXT RR are strictly informative (for humans) and MUST NOT be interpreted by the client software. Recommended TXT RDATA values are "STARTTLS" or "NO_TLS".

2.1.5. Established Sessions

After TLS negotiation completes, the connection will be encrypted and is now protected from eavesdropping and normal DNS queries SHOULD take place, following DNS-over-TCP framing ([RFC1035], section 4.2.2).

It is expected that multiple DNS queries will be made over the same TLS connection instead of tearing down the TLS connection after each response. A user of DNS-over-TLS SHOULD follow best practices for DNS-over-TCP, as described in [I-D.ietf-dnsop-5966bis]. (For DNS clients that use library functions such as "gethostbyname()", current clients may open and close UDP connections each DNS call. We recommend they reuse a single TCP connection to the recursive resolver or use UDP to a caching resolver that uses a system-wide TCP connection to the recursive resolver.)

Both clients and servers SHOULD follow existing DNS-over-TCP timeout rules, which are often implementation- and situation-dependent. In the absence of any other advice, the RECOMMENDED timeout values are 30 seconds for recursive name servers, 60 seconds for clients of recursive name servers, 10 seconds for authoritative name servers, and 20 seconds for clients of authoritative name servers. Current work in this area may assist DNS-over-TLS clients and servers select useful timeout values [draft-wouters-edns-tcp-keepalive] [tdns].

As with current DNS-over-TCP, DNS servers MAY close the connection at any time (e.g., due to resource constraints). As with current DNS-over-TCP, clients MUST handle abrupt closes and be prepared to reestablish connections and/or retry queries. DNS servers SHOULD use the TLS close-notify request to shift TCP TIME-WAIT state to the

clients. Additional requirements and guidance for optimizing DNS-over-TCP are provided by [RFC5966], [I-D.ietf-dnsop-5966bis]. As discussed in [I-D.ietf-dnsop-5966bis], TCP Fast Open [RFC7413] is of benefit.

DNS servers SHOULD enable fast TLS session resumption [RFC5077] to avoid keeping per-client session state.

2.2. Downgrade Attacks and Middleboxes

Middleboxes [RFC3234] may be present in some networks and have been known to interfere with normal DNS resolution and create problems for DNS-over-TLS. Remarkably, downgrade attacks can affect plaintext protocols that utilize "STARTTLS" signaling in a similar way. A DNS client attempting upgrade-based DNS-over-TLS through a middlebox, or in the presence of a downgrade attack, could have one of the following outcomes. (These outcomes are similar to those discussed in prior RFCs, such as [RFC3207].)

- o The DNS client sends a TO=1 query and receives a TO=0 response. In this case there is no upgrade to TLS and DNS resolution occurs normally, without encryption.

- o The DNS client sends a TO=1 query and receives a TO=1 response, but the middlebox does not understand the TLS negotiation and does not allow those packets to pass through. Clients SHOULD retry DNS without TO set if negotiation fails, and then retry with TLS after a reasonable period (see Section 2.1.3).

- o The DNS client sends a TO=1 query but receives no response at all. The middlebox might be silently dropping the query due to the presence of the TO bit, when it should, in fact, ignore and pass through unknown flag bits [RFC6891]. The client SHOULD fall back to normal (unencrypted) DNS for a reasonable period (as discussed in Section 2.1.3).

In general, clients that attempt TLS and fail can either fall back on unencrypted DNS, or wait and retry later, depending on their privacy requirements.

3. Usage Profiles

This protocol provides flexibility to accommodate several different use cases. Two usage profiles are defined here to identify specific design points in performance and privacy. Other profiles are possible but are outside the scope of this document.

3.1. Opportunistic Privacy Profile

For opportunistic privacy, analogous to SMTP opportunistic encryption [RFC7435] one desires privacy when possible, but does not require it.

With opportunistic privacy, a client might acquire a recursive DNS resolver from an untrusted source (such as DHCP while roaming), it might or might not validate the TLS certificate, and it might not use a dummy value for the initial query. These choices maximize availability and performance, but they are vulnerable to on-path attacks.

Opportunistic privacy can be used by any current client, but it only provides privacy when there are no on-path attackers.

3.2. Pre-Deployed Profile

For pre-deployed privacy, the DNS client has one or more trusted recursive DNS providers. This profile provides strong privacy guarantees to the user.

With pre-deployed privacy, a client retains a copy of the TLS certificate and IP address of each provider. The client will only use one of those DNS providers. Because it has a pre-deployed TLS certificate, it may detect person-in-the-middle and downgrade attacks.

With pre-deployed privacy, the DNS client MUST signal to the user when none of the designated DNS servers are available, and MUST NOT provide DNS service until one of the designated DNS servers becomes available.

The designated DNS provider may be temporarily unavailable when configurating a network. For example, for clients on networks that require authentication through web-based login, such authentication may require DNS interception and spoofing. Techniques such as those used by DNSSEC-trigger MAY be used during network configuration, with the intent to transition to the designated DNS provider after authentication. The user MUST be alerted that the DNS is not private during such bootstrap.

Methods for pre-deployment of the designated DNS provider are outside the scope of this document. In corporate settings, such information may be provided at system installation. Use of multiple public DNS providers suggests that end users are able to configure DNS by hand.

4. Performance Considerations

DNS-over-TLS incurs additional latency at session startup. It also requires additional state (memory) and increased processing (CPU).

1. Latency: Compared to UDP, DNS-over-TCP requires an additional round-trip-time (RTT) of latency to establish the connection. The TLS handshake adds another two RTTs of latency. Clients and servers should support connection keepalive (reuse) and out-of-order processing to amortize connection setup costs. Moreover, TLS connection resumption can further reduce the setup delay.

2. State: The use of connection-oriented TCP requires keeping additional state in both kernels and applications. TLS has marginal increases in state over TCP alone. The state requirements are of particular concerns on servers with many clients. Smaller timeout values will reduce the number of concurrent connections, and servers can preemptively close connections when resources limits are exceeded.

3. Processing: Use of TLS encryption algorithms results in slightly higher CPU usage. Servers can choose to refuse new DNS-over-TCP clients if processing limits are exceeded.

4. Number of connections: To minimize state on DNS servers and connection startup time, clients SHOULD minimize creation of new TCP connections. Use of a local DNS forwarder allows a single active DNS-over-TLS connection allows a single active TCP connection for DNS per client computer. Additional guidance can be found in [I-D.ietf-dnsop-5966bis].

A full performance evaluation is outside the scope of this specification. A more detailed analysis of the performance implications of DNS-over-TLS (and DNS-over-TCP) is discussed in a technical report [tdns] and [I-D.ietf-dnsop-5966bis].

5. IANA Considerations

This document defines a new bit ("TO") in the Flags field of the EDNS0 OPT meta-RR. At the time of approval of this draft in the standards track, as per the IANA Considerations of RFC 6891, IANA is requested to reserve the second leftmost bit of the flags as the TO bit, immediately adjacent to the DNSSEC DO bit, as shown in Section 2.

IANA is requested add the following value to the "Service Name and Transport Protocol Port Number Registry" registry. That registry is

populated by expert review [RFC6335], and such a review will be requested if this document progresses.

Service Name	DNS-over-TLS
Transport Protocol(s)	TCP
Assignee	IESG
Contact	TBD
Description	DNS query-response protocol run over TLS
Reference	This document

6. Security Considerations

The goal of this proposal is to address the security risks that arise because DNS queries may be eavesdropped upon, as described above. There are a number of residual risks that may impact this goal.

1. There are known attacks on TLS, such as person-in-the-middle and protocol downgrade. These are general attacks on TLS and not specific to DNS-over-TLS; please refer to the TLS RFCs for discussion of these security issues.

2. Any protocol interactions prior to the TLS handshake are performed in the clear and can be modified by a man-in-the-middle attacker. For this reason, clients MAY discard cached information about server capabilities advertised prior to the start of the TLS handshake.

3. As with other uses of STARTTLS-upgrade to TLS, the mechanism specified here is susceptible to downgrade attacks, where a person-in-the-middle prevents a successful TLS upgrade. Keeping track of servers known to support TLS (i.e., "pinning") enables clients to detect downgrade attacks. For servers with no connection history, clients may choose to refuse non-TLS DNS, or they may continue without TLS, depending on their privacy requirements.

4. This document does not propose new ideas for certificate authentication for TLS in the context of DNS. Several external methods are possible, although each has weaknesses. The current Certificate Authority infrastructure [RFC5280] is used by HTTP/TLS [RFC2818]. With many trusted CAs, this approach has recognized weaknesses [CA_Compromise]. Some work is underway to partially address these concerns (for example, with certificate pinning [certificate_pinning], but more work is needed. DANE [RFC6698] provides mechanisms to root certificate trust with DNSSEC. That use here must be carefully evaluated to address potential issues in trust recursion. For stub-to-recursive

resolver use, certificate authentication is sometimes either easy or nearly impossible. If the recursive resolver is manually configured, its certificate can be authenticated when it is configured. If the recursive resolver is automatically configured (such as with DHCP [RFC2131]), it could use DHCP authentication mechanisms [RFC3118]).

Ongoing discussion and development of opportunistic TLS (connections without CA validation, [RFC7435]) may be relevant to DNS-over-TLS.

7. Acknowledgments

The authors would like to thank Stephane Bortzmeyer, Brian Haberman, Kim-Minh Kaplan, Bill Manning, George Michaelson, Eric Osterweil, Glen Wiley, John Dickinson, and Sara Dickinson for reviewing this Internet-draft, and Nikita Somaiya for early work on this idea.

Work by Zi Hu, Liang Zhu, and John Heidemann in this paper is partially sponsored by the U.S. Dept. of Homeland Security (DHS) Science and Technology Directorate, HSARPA, Cyber Security Division, BAA 11-01-RIKA and Air Force Research Laboratory, Information Directorate under agreement number FA8750-12-2-0344, and contract number D08PC75599.

8. References

8.1. Normative References

[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.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, January 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5966] Bellis, R., "DNS Transport over TCP - Implementation Requirements", RFC 5966, August 2010.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, August 2011.

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

8.2. Informative References

[CA_Compromise]
Infosec Island Admin, "CA Compromise", January 2012, <http://www.infosecisland.com/blogview/19782-Web-Authentication-A-Broken-Trust-with-No-Easy-Fix.html>.

[I-D.ietf-dnsop-5966bis]
Dickinson, J., Bellis, R., Mankin, A., and D. Wessels, "DNS Transport over TCP - Implementation Requirements", draft-ietf-dnsop-5966bis-00 (work in progress), December 2014.

[I-D.ietf-dprive-problem-statement]
Bortzmeyer, S., "DNS privacy considerations", draft-ietf-dprive-problem-statement-01 (work in progress), October 2014.

[RFC1939] Myers, J. and M. Rose, "Post Office Protocol - Version 3", STD 53, RFC 1939, May 1996.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[RFC2595] Newman, C., "Using TLS with IMAP, POP3 and ACAP", RFC 2595, June 1999.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC3118] Droms, R. and W. Arbaugh, "Authentication for DHCP Messages", RFC 3118, June 2001.

[RFC3207] Hoffman, P., "SMTP Service Extension for Secure SMTP over Transport Layer Security", RFC 3207, February 2002.

[RFC3234] Carpenter, B. and S. Brim, "Middleboxes: Taxonomy and Issues", RFC 3234, February 2002.

[RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, March 2003.

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

[RFC4892] Woolf, S. and D. Conrad, "Requirements for a Mechanism Identifying a Name Server Instance", RFC 4892, June 2007.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, August 2012.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, May 2014.

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, December 2014.

[RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, December 2014.

[certificate_pinning]
OWASP, "Certificate and Public Key Pinning", 2014, <https://www.owasp.org/index.php/Certificate_and_Public_Key_Pinning>.

[draft-dempsky-dnscurve]
Dempsky, M., "DNSCurve", draft-dempsky-dnscurve-01 (work in progress), August 2010, <http://tools.ietf.org/html/draft-dempsky-dnscurve-01>.

[draft-osterweil-dane-ipsec]
Osterweil, E., Wiley, G., Mitchell, D., and A. Newton, "Opportunistic Encryption with DANE Semantics and IPsec: IPSECA", draft-osterweil-dane-ipsec-00 (work in progress), February 2014, <http://tools.ietf.org/html/draft-osterweil-dane-ipsec-00>.

[draft-wijngaards-confidentialdns]
Wijngaards, W., "Confidential DNS",

draft-wijngaards-dnsop-confidentialdns-03 (work in progress), November 2013, <http://tools.ietf.org/html/draft-wijngaards-dnsop-confidentialdns-03>.

[draft-wouters-edns-tcp-keepalive] Wouters, P. and J. Abley, "The edns-tcp-keepalive EDNS0 Option", draft-wouters-edns-tcp-keepalive-00 (work in progress), October 2013, <http://tools.ietf.org/html/draft-wouters-edns-tcp-keepalive-00>.

[tdns] Zhu, L., Hu, Z., Heidemann, J., Wessels, D., Mankin, A., and N. Somaiya, "T-DNS: Connection-Oriented DNS to Improve Privacy and Security", Technical report ISI-TR-688, February 2014, <Technical report, ISI-TR-688, ftp://ftp.isi.edu/isi-pubs/tr-688.pdf>.

[unbound] NLnet Labs, Verisign labs, "Unbound", December 2013, <http://unbound.net/>.

Authors' Addresses

Zi Hu
USC/Information Sciences Institute
4676 Admiralty Way, Suite 1133
Marina del Rey, CA 90292
USA

Phone: +1 213 587-1057
Email: zihu@usc.edu

Liang Zhu
USC/Information Sciences Institute
4676 Admiralty Way, Suite 1133
Marina del Rey, CA 90292
USA

Phone: +1 310 448-8323
Email: liangzhu@usc.edu

John Heidemann
USC/Information Sciences Institute
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
USA

Phone: +1 310 822-1511
Email: johnh@isi.edu

Allison Mankin
Verisign Labs
12061 Bluemont Way
Reston, VA 20190

Phone: +1 703 948-3200
Email: amankin@verisign.com

Duane Wessels
Verisign Labs
12061 Bluemont Way
Reston, VA 20190

Phone: +1 703 948-3200
Email: dwessels@verisign.com

Paul Hoffman
VPN Consortium

Email: paul.hoffman@vpnc.org

DNSOP Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 7, 2015

W. Wijngaards
NLnet Labs
G. Wiley
VeriSign, Inc.

March 6, 2015

Confidential DNS
draft-wijngaards-dnsop-confidentialdns-03

Abstract

This document offers opportunistic encryption to provide privacy for DNS queries and responses.

Requirements Language

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].

Status of This Memo

Copyright Notice

1. Introduction

The privacy of the Question, Answer, Authority and Additional sections in DNS queries and responses is protected by the confidential DNS protocol by encrypting the contents of each section. The goal of this change to the DNS protocol is to make large scale monitoring more expensive, see [draft-bortzmeyer-dnsop-dns-privacy] and [draft-koch-perpass-dns-confidentiality]. Authenticity and integrity may be provided by DNSSEC, this protocol does not change DNSSEC and does not offer the means to authenticate responses.

Confidential communication between any pair of DNS servers is supported, both between iterative resolvers and authoritative servers and between stub resolvers and recursive resolvers.

The confidential DNS protocol has minimal impact on the number of packets involved in a typical DNS query/response exchange by leveraging a cacheable ENCRYPT Resource Record and an optionally cacheable shared secret. The protocol supports selectable cryptographic suites and parameters (such as key sizes).

The client fetches an ENCRYPT RR from the server that it wants to contact. The public key retrieved in the ENCRYPT RR is used to encrypt a shared secret or public key that the client uses to encrypt the sections in the DNS query and which the name server uses to encrypt the DNS response.

As this is opportunistic encryption, the key is (re-)fetched when the exchange fails or after the TTL expires. If the key fetch fails or the encrypted query fails, communication in the clear is performed.

The server advertises which crypto suites and key lengths may be used in the ENCRYPT RR, the client then chooses a crypto suite from this list and includes that selection in subsequent DNS queries.

The key from the server can be cached by the client, using the TTL specified in the ENCRYPT RR, the IP address of the server distinguishes keys in the cache. The server may also cache shared secrets and keys from clients.

The optional authenticated mode of operation uses two mechanisms, one for authoritative and one for recursive servers, that fetch the public key for the server and sign it with DNSSEC. For authoritative

servers, the key is included in an extra DS record in the parent's delegation. For recursive servers the key is at the reverse IP address location.

2. ENCRYPT RR Type

The RR type for confidential DNS is ENCRYPT, type TBD (decimal). The presentation format is:

. ENCRYPT [flags] [algo] [id] [data]

The flags, algo and id are unsigned numbers in decimal and the data is in base-64. The wireformat is: one octet flags, one octet algo, one octet id and the remainder of the rdata is for the data. The type is class independent. The domain name of the ENCRYPT record is '.' (the root label) for hop-by-hop exchanges.

In the flags the least two bits are the usage value. The other flag bits MUST be sent as zeroes, and the receiver MUST ignore RRs that have other flag bits set.

- o PAD (usage=0): the ENCRYPT contains padding material. Algo and id are set to 0. Its data length varies (0-63 octets), and may contain any value. It is used to pad packets to obscure the packet length. Append such records to make the DNS message for queries and answers a whole multiple of 64 bytes.

- o KEY (usage=1): the ENCRYPT contains a public or symmetric key. The algo field gives the algorithm. The id identifies the key, this id is copied to ENCRYPT type RRS to identify which key to use to decrypt the data. The data contains the key bits.

- o RRS (usage=2): encrypted data. The data contains encrypted resource records. The data is encrypted with the selected algorithm and key id. The data contains resource records in DNS wireformat [RFC1034], with a domain name, type, class, ttl, rdatalength and rdata.

- o SYM (usage=3): the ENCRYPT contains an encrypted symmetric key. The contained, encrypted data is rdata of an ENCRYPT of type KEY and has the symmetric key. The data is encrypted with the algorithm and id indicated. The encrypted data encompasses the flags, algo, id, data for the symmetric key.

The ENCRYPT RR type can contain keys. It uses the same format as the DNSKEY record [RFC4034] for public keys. algo=0 is reserved for future expansion of the algorithm number above 255. algo=1 is RSA, the rdata determines the key size. algo=2 is AES, aes-cbc, size of

the rdata determines the size of the key.

3. Server and Client Algorithm

If a clients wants to fetch the keys for the server from the server, it performs a query with query type ENCRYPT and query name '.' (root label). The reply contains the ENCRYPT (or multiple if a choice is offered) in the answer section. These ENCRYPTs have the KEY usage.

If a client wants to perform an encrypted query, it sends an unencrypted outer packet, with query type ENCRYPT and query name '.' (root label). In the authority section it includes an ENCRYPT record of type RRS. This encrypts a number of records, the first is a query-section style query record, and then zero or more ENCRYPTs of type KEY that the server uses to encrypt the reply. If the client wants to use a symmetric key, it omits the KEYs, and instead includes an ENCRYPT of type SYM in the authority section. The ENCRYPT of type RRs then follows after the SYM and can be encrypted with the key from that SYM.

If a server wants to encrypt a reply, it also uses the ENCRYPT type. The reply looks like a normal DNS packet, i.e. it has a normal unencrypted outer DNS packet. Because the query name and query type have been encrypted, the outer packet has a query name of '.' and query type of ENCRYPT and the reply has an ENCRYPT type RRS in the answer section. The reply RRs have been encrypted into the data of the ENCRYPT record. The RRS data starts with 10 bytes of header; the flags and section counts.

The client may lookup keys whenever it wants to. It may cache the keys for the server, using the TTL of those ENCRYPT records. It should also cache failures to lookup the ENCRYPT record for some time. If the client fails to look up the ENCRYPT records it MUST fall back to unencrypted communication (this is the opportunistic encryption case). The result of an encrypted query may also be timeouts, errors or replies with mangled contents, in that case the client MUST fall back to unencrypted communication (this is the opportunistic encryption case).

If some middlebox removes the ENCRYPT from the authority section of an encrypted query, the query looks like a . ENCRYPT lookup and likely a reply with ENCRYPTs of type KEY is returned instead of the encrypted reply with an ENCRYPT of type RRS, and again the client does the unencrypted fallback (this is the opportunistic encryption case). If the server has changed its keys and does not recognize the keys in an encrypted query, it should return an ENCRYPT record of type PAD with no data. A server may decide it does not (any longer) have the resources for encryption and reply with SERVFAIL to

encrypted queries, forcing unencrypted fallback (this is the opportunistic encryption case). Keys for unknown algorithms should be ignored by the client, if no usable keys remain, fallback to insecure (this is for both opportunistic and authenticated).

The client may cache the ENCRYPT of type SYM for a server together with the symmetric secret, this is better for performance, as public-key operations can be avoided for repeated queries. The server may also cache the ENCRYPTs of type SYM with the decoded secret, associating a lookup for the rdata of the SYM record with the decoded secret, avoiding public-key operations for repeated queries. This is why the SYM record is sent separately in the authority section in queries (it is identical and can be used for cache lookups).

Key rollover is possible, support the old key for its TTL, while advertising the new key, for the servers. For clients, generate a new public or symmetric key and use it.

4. Authenticated Operation

The previous documented the opportunistic operation, where deployment is easier, but security is weaker. This documents options for authenticated operation. The client selects if encryption is authenticated, opportunistic, or disabled in its local policy (configuration).

The authentication happens with a DNSSEC signed DS record that carries the key for confidential DNS. This removes a full roundtrip from the connection setup cost. The DS has hash type TBDhashtype, that is specific for confidential DNS. The DS record carries a flag byte and the public key (in DNSKEY’s wireformat) in its rdata. This means that the confidential DNS keys are acquired with a referral to the zone and are secured with DNSSEC.

Because the key itself is carried, the probe sequence can be omitted and an encrypted query can be sent to the delegated server straight away. The nameservers for that zone then MUST support using that key for encrypting packets. The servers have the same key with authenticated mode, where with the opportunistic mode, every server could have its own key.

Validators do not know or support the DS with ENCRYPT hash type, those validators ignore them and continue to DNSSEC validate the zone. Validators that support the new hash type should use them to encrypt messages and use the remaining DS records to DNSSEC validate the zone.

This changes the opportunistic encryption to authenticated

encryption. The fallback to insecure is still possible and this may make deployment easier. The one byte at the start of the base64 data, in its least significant bit, signals if fallback to insecure is allowed (value 0x01). That gives the zone owner the option to enable fallback to insecure or if it should be disabled. The remainder of the DS base64 data contains a public key in the same format as when sent in the rdata of ENCRYPT KEY. The type of the key is in the key type field of this DS record. With fallback to insecure disabled and the keys authenticated the confidential DNS query and response should be fully secure (i.e. not 'Opportunistically' secure).

With fallback to insecure disabled, queries fail instead of falling back to insecure. This means no answer is acquired, and DNS lookups for that zone fail because the security failed.

The DS method works for authority servers. Recursors need another method. The client looks up reverse-of-recursors-IP.arpa ENCRYPT and gets the keys signed with DNSSEC from there (type ENCRYPT KEY lookup). If there is no dnssec secure answer with a key, the opportunistic key exchange is attempted. Do this for DNSSEC-insecure answers, if there is no trust anchor, or when no such name and ENCRYPT are present. If it is dnssec bogus, then authentication failed and it is not possible to communicate with the server (with the authenticated communication mode selected by the client).

5. IANA Considerations

An RR type registration for type ENCRYPT with number TBD and it references this document [[to be done when this becomes RFC]].

A DS record hash type is registered TBDhashtype that references this document. It is for the confidential DNS public key, acronym ENCRYPT.

6. Security Considerations

Opportunistic encryption can be configured. Opportunistic encryption has many drawbacks against active intrusion, but it works against pervasive passive surveillance, and thus it improves privacy.

With authentication (if selected by the client) the key is secured with DNSSEC.

This technique encrypts DNS queries and answers, but other data sources, such as timing, IP addresses, and the packet size can be observed. These could provide almost all the information that was encrypted.

7. Acknowledgments

Roy Arends

8. Normative References

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

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

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

Authors' Addresses

Wouter Wijngaards
NLnet Labs
Science Park 140
Amsterdam 1098 XH
The Netherlands

EMail: wouter@nlnetlabs.nl

Glen Wiley
VeriSign, Inc.
Reston, VA
USA

EMail: gwiley@verisign.com