DPRIVE
Internet-Draft
Intended status: Experimental
Expires: February 17, 2017

T. Reddy
D. Wing
P. Patil
Cisco

August 16, 2016

Specification for DNS over Datagram Transport Layer Security (DTLS)
draft-ietf-dprive-dnsodtls-10

Abstract

DNS queries and responses are visible to network elements on the path between the DNS client and its server. These queries and responses can contain privacy-sensitive information which is valuable to protect. An active attacker can send bogus responses causing misdirection of the subsequent connection.

This document proposes the use of Datagram Transport Layer Security (DTLS) for DNS, to protect against passive listeners and certain active attacks. As latency is critical for DNS, this proposal also discusses mechanisms to reduce DTLS round trips and reduce DTLS handshake size. The proposed mechanism runs over port 853.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Domain Name System is specified in [RFC1034] and [RFC1035]. DNS queries and responses are normally exchanged unencrypted and are thus vulnerable to eavesdropping. Such eavesdropping can result in an undesired entity learning domains that a host wishes to access, thus resulting in privacy leakage. The DNS privacy problem is further discussed in [RFC7626].

Active attackers have long been successful at injecting bogus responses, causing cache poisoning and causing misdirection of the subsequent connection (if attacking A or AAAA records). A popular mitigation against that attack is to use ephemeral and random source ports for DNS queries [RFC5452].

This document defines DNS over DTLS (DNS-over-DTLS) which provides confidential DNS communication between stub resolvers and recursive resolvers, stub resolvers and forwarders, forwarders and recursive resolvers. DNS-over-DTLS puts an additional computational load on

servers. The largest gain for privacy is to protect the communication between the DNS client (the end user's machine) and its caching resolver. DNS-over-DTLS might work equally between recursive clients and authoritative servers, but this application of the protocol is out of scope for the DNS PRIVate Exchange (DPRIVE) Working Group per its current charter. This document does not change the format of DNS messages.

The motivations for proposing DNS-over-DTLS are that

- TCP suffers from network head-of-line blocking, where the loss of a packet causes all other TCP segments to not be delivered to the application until the lost packet is re-transmitted. DNS-over-DTLS, because it uses UDP, does not suffer from network head-of-line blocking.

- DTLS session resumption consumes 1 round trip whereas TLS session resumption can start only after TCP handshake is complete. However TCP Fast Open [RFC7413] can eliminate 1-RTT in the latter case.

1.1. Relationship to TCP Queries and to DNSSEC

DNS queries can be sent over UDP or TCP. The scope of this document, however, is only UDP. DNS over TCP can be protected with TLS, as described in [RFC7858]. DNS-over-DTLS alone cannot provide privacy for DNS messages in all circumstances, specifically when the DTLS record size is larger than the path MTU. In such situations the DNS server will respond with a truncated response (see Section 5). Therefore DNS clients and servers that implement DNS-over-DTLS MUST also implement DNS-over-TLS in order to provide privacy for clients that desire Strict Privacy as described in [I-D.ietf-dprive-dtls-and-tls-profiles].

DNS Security Extensions (DNSSEC [RFC4033]) provides object integrity of DNS resource records, allowing end-users (or their resolver) to verify legitimacy of responses. However, DNSSEC does not protect privacy of DNS requests or responses. DNS-over-DTLS works in conjunction with DNSSEC, but DNS-over-DTLS does not replace the need or value of DNSSEC.

2. Terminology

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

3. Establishing and Managing DNS-over-DTLS Sessions

3.1. Session Initiation

By default, DNS-over-DTLS MUST run over standard UDP port 853 as defined in Section 8, unless the DNS server has mutual agreement with its clients to use a port other than 853 for DNS-over-DTLS. In order to use a port other than 853, both clients and servers would need a configuration option in their software.

The DNS client should determine if the DNS server supports DNS-over-DTLS by sending a DTLS ClientHello message to port 853 on the server, unless it has mutual agreement with its server to use a port other than port 853 for DNS-over-DTLS. Such another port MUST NOT be port 53 but MAY be from the "first-come, first-served" port range. This recommendation against use of port 53 for DNS-over-DTLS is to avoid complication in selecting use or non-use of DTLS and to reduce risk of downgrade attacks.

A DNS server that does not support DNS-over-DTLS will not respond to ClientHello messages sent by the client. If no response is received from that server, and the client has no better round-trip estimate, the client MUST retransmit the DTLS ClientHello according to Section 4.2.4.1 of [RFC6347]. After 15 seconds, it MUST cease attempts to re-transmit its ClientHello. The client MAY repeat that procedure to discover if DNS-over-DTLS service becomes available from the DNS server, but such probing SHOULD NOT be done more frequently than every 24 hours and MUST NOT be done more frequently than every 15 minutes. This mechanism requires no additional signaling between the client and server.

DNS clients and servers MUST NOT use port 853 to transport cleartext DNS messages. DNS clients MUST NOT send and DNS servers MUST NOT respond to cleartext DNS messages on any port used for DNS-over-DTLS (including, for example, after a failed DTLS handshake). There are significant security issues in mixing protected and unprotected data, UDP connections on a port designated by a given server for DNS-over-DTLS are reserved purely for encrypted communications.

3.2. DTLS Handshake and Authentication

Once the DNS client succeeds in receiving HelloVerifyRequest from the server via UDP on the well-known port for DNS-over-DTLS, it proceeds with the DTLS handshake as described in [RFC6347], following the best practices specified in [RFC7525].

DNS privacy requires encrypting the query (and response) from passive attacks. Such encryption typically provides integrity protection as

a side-effect, which means on-path attackers cannot simply inject bogus DNS responses. However, to provide stronger protection from active attackers pretending to be the server, the server itself needs to be authenticated. To authenticate the server providing DNS privacy, DNS client MUST use the authenication mechanisms discussed in [I-D.ietf-dprive-dtls-and-tls-profiles]. This document does not propose new ideas for authentication.

After DTLS negotiation completes, the connection will be encrypted and is now protected from eavesdropping.

3.3. Established Sessions

In DTLS, all data is protected using the same record encoding and mechanisms. When the mechanism described in this document is in effect, DNS messages are encrypted using the standard DTLS record encoding. When a user of DTLS wishes to send a DNS message, the data is delivered to the DTLS implementation as an ordinary application data write (e.g., SSL_write()). A single DTLS session can be used to send multiple DNS requests and receive multiple DNS responses.

To mitigate the risk of unintentional server overload, DNS-over-DTLS clients MUST take care to minimize the number of concurrent DTLS sessions made to any individual server. It is RECOMMENDED that for any given client/server interaction there SHOULD be no more than one DTLS session. Similarly, servers MAY impose limits on the number of concurrent DTLS sessions being handled for any particular client IP address or subnet. These limits SHOULD be much looser than the client guidelines above, because the server does not know, for example, if a client IP address belongs to a single client, is multiple resolvers on a single machine, or is multiple clients behind a device performing Network Address Translation (NAT).

In between normal DNS traffic while the communication to the DNS server is quiescent, the DNS client may want to probe the server using DTLS heartbeat [RFC6520] to ensure it has maintained cryptographic state. Such probes can also keep alive firewall or NAT bindings. This probing reduces the frequency of needing a new handshake when a DNS query needs to be resolved, improving the user experience at the cost of bandwidth and processing time.

A DTLS session is terminated by the receipt of an authenticated message that closes the connection (e.g., a DTLS fatal alert). If the server has lost state, a DTLS handshake needs to be initiated with the server. For the client, state should be destroyed when disconnecting from the network (e.g., associated IP interface is brought down). For the server, to mitigate the risk of unintentional server overload, it is RECOMMENDED that the default DNS-over-DTLS

server application-level idle time out be on the order of several seconds, but no particular value is specified. When no DNS queries have been received from the client after idle time out, the server MUST send a DTLS fatal alert and then destroy its DTLS state. The DTLS fatal alert packet indicates the server has destroyed its state, signaling to the client if it wants to send a new DTLS message it will need to re-establish cryptographic context with the server (via full DTLS handshake or DTLS session resumption). In practice, the idle period can vary dynamically, and servers MAY allow idle connections to remain open for longer periods as resources permit.

Figure 1 shows DTLS handshake using cookie and issuing new session ticket for session resumption.

```
        Client                                          Server
        ------                                          ------

        ClientHello             -------->


                                <-------    HelloVerifyRequest
                                              (contains cookie)

        ClientHello             -------->
        (contains cookie)
        (empty SessionTicket extension)
                                                   ServerHello
                                       (empty SessionTicket extension)
                                                  Certificate*
                                            ServerKeyExchange*
                                           CertificateRequest*
                                <--------      ServerHelloDone

        Certificate*
        ClientKeyExchange
        CertificateVerify*
        (ChangeCipherSpec)
        Finished                -------->
                                              NewSessionTicket
                                            (ChangeCipherSpec)
                                <--------             Finished


        DNS Request             --------->

                                <---------  DNS Response
```

Figure 1: Message Flow for Full Handshake Issuing New Session Ticket

4. Performance Considerations

DTLS protocol profile for DNS-over-DTLS is discussed in Section 11 of [I-D.ietf-dprive-dtls-and-tls-profiles]. To reduce the number of octets of the DTLS handshake, especially the size of the certificate in the ServerHello (which can be several kilobytes), DNS clients and servers can use raw public keys [RFC7250] or Cached Information Extension [I-D.ietf-tls-cached-info]. Cached Information Extension avoids transmitting the server's certificate and certificate chain if the client has cached that information from a previous TLS handshake. TLS False Start [I-D.ietf-tls-falsestart] which reduces round-trips by allowing the TLS second flight of messages (ChangeCipherSpec) to also contain the (encrypted) DNS query.

It is highly advantageous to avoid server-side DTLS state and reduce the number of new DTLS sessions on the server which can be done with TLS Session Resumption without server state [RFC5077]. This also eliminates a round-trip for subsequent DNS-over-DTLS queries, because with [RFC5077] the DTLS session does not need to be re-established.

Since responses within a DTLS session can arrive out of order, clients MUST match responses to outstanding queries on the same DTLS connection using the DNS Message ID. If the response contains a question section, the client MUST match the QNAME, QCLASS, and QTYPE fields. Failure by clients to properly match responses to outstanding queries can have serious consequences for interoperability ([RFC7766], Section 7).

5. PMTU issues

Compared to normal DNS, DTLS adds at least 13 octets of header, plus cipher and authentication overhead to every query and every response. This reduces the size of the DNS payload that can be carried. DNS client and server MUST support the EDNS0 option defined in [RFC6891] so that the DNS client can indicate to the DNS server the maximum DNS response size it can reassemble and deliver in the DNS client's network stack. If the DNS client does set the EDNS0 option defined in [RFC6891] then the maximum DNS response size of 512 bytes plus DTLS overhead will be well within the Path MTU. If the Path MTU is not known, an IP MTU of 1280 bytes SHOULD be assumed. The client sets its EDNS0 value as if DTLS is not being used. The DNS server MUST ensure that the DNS response size does not exceed the Path MTU i.e. each DTLS record MUST fit within a single datagram, as required by [RFC6347]. The DNS server must consider the amount of record expansion expected by the DTLS processing when calculating the size of DNS response that fits within the path MTU. Path MTU MUST be greater than or equal to [DNS response size + DTLS overhead of 13 octets + padding size ([RFC7830]) + authentication overhead of the negotiated DTLS cipher suite + block padding (Section 4.1.1.1 of [RFC6347]]. If the DNS server's response were to exceed that calculated value, the server MUST send a response that does fit within that value and sets the TC (truncated) bit. Upon receiving a response with the TC bit set and wanting to receive the entire response, the client behaviour is governed by the current Usage profile [I-D.ietf-dprive-dtls-and-tls-profiles]. For Strict Privacy the client MUST only send a new DNS request for the same resource record over an encrypted transport (e.g. DNS-over-TLS [RFC7858]). Clients using Opportunistic Privacy SHOULD try for the best case (an encrypted and authenticated transport) but MAY fallback to intermediate cases and eventually the worst case scenario (clear text) in order to obtain a response.

6. Anycast

DNS servers are often configured with anycast addresses. While the network is stable, packets transmitted from a particular source to an anycast address will reach the same server that has the cryptographic context from the DNS-over-DTLS handshake. But when the network configuration changes, a DNS-over-DTLS packet can be received by a server that does not have the necessary cryptographic context. To encourage the client to initiate a new DTLS handshake, DNS servers SHOULD generate a DTLS Alert message in response to receiving a DTLS packet for which the server does not have any cryptographic context. Upon receipt of an un-authenicated DTLS alert, the DTLS client validates the Alert is within the replay window (Section 4.1.2.6 of [RFC6347]). It is difficult for the DTLS client to validate that the DTLS alert was generated by the DTLS server in response to a request or was generated by an on- or off-path attacker. Thus, upon receipt of an in-window DTLS Alert, the client SHOULD continue re-transmitting the DTLS packet (in the event the Alert was spoofed), and at the same time it SHOULD initiate DTLS session resumption. When the DTLS client receives an authenticated DNS response from one of those DTLS sessions, the other DTLS session should be terminated.

7. Usage

Two Usage Profiles, Strict and Opportunistic are explained in [I-D.ietf-dprive-dtls-and-tls-profiles]. Using encrypted DNS messages with an authenticated server is most preferred, encrypted DNS messages with an unauthenticated server is next preferred, and plain text DNS messages is least preferred.

8. IANA Considerations

This specification uses port 853 already allocated in the IANA port number registry as defined in Section 6 of [RFC7858].

9. Security Considerations

The interaction between a DNS client and DNS server requires Datagram Transport Layer Security (DTLS) with a ciphersuite offering confidentiality protection. The guidance given in [RFC7525] MUST be followed to avoid attacks on DTLS. DNS clients keeping track of servers known to support DTLS enables clients to detect downgrade attacks. To interfere with DNS-over-DTLS, an on- or off-path attacker might send an ICMP message towards the DTLS client or DTLS server. As these ICMP messages cannot be authenticated, all ICMP errors should be treated as soft errors [RFC1122]. If the DNS query was sent over DTLS then the corresponding DNS response MUST only be accepted if it is received over the same DTLS connection. This

behavior mitigates all possible attacks described in Measures for Making DNS More Resilient against Forged Answers [RFC5452]. Security considerations in [RFC6347] and [I-D.ietf-dprive-dtls-and-tls-profiles] are to be taken into account.

A malicious client might attempt to perform a high number of DTLS handshakes with a server. As the clients are not uniquely identified by the protocol and can be obfuscated with IPv4 address sharing and with IPv6 temporary addresses, a server needs to mitigate the impact of such an attack. Such mitigation might involve rate limiting handshakes from a certain subnet or more advanced DoS/DDoS techniques beyond the scope of this paper.

10. Acknowledgements

Thanks to Phil Hedrick for his review comments on TCP and to Josh Littlefield for pointing out DNS-over-DTLS load on busy servers (most notably root servers). The authors would like to thank Simon Josefsson, Daniel Kahn Gillmor, Bob Harold, Ilari Liusvaara, Sara Dickinson, Christian Huitema, Stephane Bortzmeyer, Alexander Mayrhofer and Geoff Huston for discussions and comments on the design of DNS-over-DTLS. The authors would like to give special thanks to Sara Dickinson for her help.

11. References

11.1. Normative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <http://www.rfc-editor.org/info/rfc1034>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <http://www.rfc-editor.org/info/rfc1035>.

[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>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <http://www.rfc-editor.org/info/rfc4033>.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, DOI 10.17487/RFC5077, January 2008, <http://www.rfc-editor.org/info/rfc5077>.

[RFC5452] Hubert, A. and R. van Mook, "Measures for Making DNS More Resilient against Forged Answers", RFC 5452, DOI 10.17487/RFC5452, January 2009, <http://www.rfc-editor.org/info/rfc5452>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <http://www.rfc-editor.org/info/rfc6347>.

[RFC6520] Seggelmann, R., Tuexen, M., and M. Williams, "Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension", RFC 6520, DOI 10.17487/RFC6520, February 2012, <http://www.rfc-editor.org/info/rfc6520>.

[RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <http://www.rfc-editor.org/info/rfc6891>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <http://www.rfc-editor.org/info/rfc7525>.

[RFC7830] Mayrhofer, A., "The EDNS(0) Padding Option", RFC 7830, DOI 10.17487/RFC7830, May 2016, <http://www.rfc-editor.org/info/rfc7830>.

11.2. Informative References

[I-D.ietf-dprive-dtls-and-tls-profiles]
Dickinson, S., Gillmor, D., and T. Reddy, "Authentication and (D)TLS Profile for DNS-over-(D)TLS", draft-ietf-dprive-dtls-and-tls-profiles-03 (work in progress), July 2016.

[I-D.ietf-tls-cached-info]
Santesson, S. and H. Tschofenig, "Transport Layer Security (TLS) Cached Information Extension", draft-ietf-tls-cached-info-23 (work in progress), May 2016.

[I-D.ietf-tls-falsestart]
Langley, A., Modadugu, N., and B. Moeller, "Transport Layer Security (TLS) False Start", draft-ietf-tls-falsestart-02 (work in progress), May 2016.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <http://www.rfc-editor.org/info/rfc1122>.

[RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <http://www.rfc-editor.org/info/rfc7250>.

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <http://www.rfc-editor.org/info/rfc7413>.

[RFC7626] Bortzmeyer, S., "DNS Privacy Considerations", RFC 7626, DOI 10.17487/RFC7626, August 2015, <http://www.rfc-editor.org/info/rfc7626>.

[RFC7766] Dickinson, J., Dickinson, S., Bellis, R., Mankin, A., and D. Wessels, "DNS Transport over TCP - Implementation Requirements", RFC 7766, DOI 10.17487/RFC7766, March 2016, <http://www.rfc-editor.org/info/rfc7766>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <http://www.rfc-editor.org/info/rfc7858>.

Authors' Addresses

Tirumaleswar Reddy
Cisco Systems, Inc.
Cessna Business Park, Varthur Hobli
Sarjapur Marathalli Outer Ring Road
Bangalore, Karnataka 560103
India

Email: tireddy@cisco.com

Dan Wing
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134
USA

Email: dwing@cisco.com

Prashanth Patil
Cisco Systems, Inc.
Bangalore
India

Email: praspati@cisco.com