Using HTTPS for Privacy Between DNS Stub and Recursive Resolvers
draft-hoffman-dprive-dns-tls-https-00

Abstract

   DNS queries and responses can contain information that reveals
   important information about the person who caused the queries, and it
   would be better if eavesdroppers were unable to see DNS traffic.
   This document describes how to use TLS for encrypting DNS traffic
   between a system acting as a DNS stub resolver and a system acting as
   a DNS recursive resolver. It defines how to easily wrap DNS queries
   in HTTP requests and interpret DNS responses in the HTTP respones;
   the HTTP here is always run under TLS on port 443.

   Discussion of this draft should take place in the DPRIVE WG.

Status of This Memo

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

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF). Note that other groups may also distribute
   working documents as Internet-Drafts. The list of current Internet-
   Drafts is at http://datatracker.ietf.org/drafts/current/.

   Internet-Drafts are draft documents valid for a maximum of six months
   and may be updated, replaced, or obsoleted by other documents at any
   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 25, 2015.

Table of Contents

1. Introduction

   As described in [I-D.bortzmeyer-dnsop-dns-privacy], there are many
   reasons why a user or system making a DNS query would like the query
   and the response to not be seen by others. The best way to make a
   query and response private is to use encryption, and TLS is a
   commonly-deployed protocol that provides encryption to clients and
   servers. This document describes how to use TLS for encrypting DNS
   traffic between a system acting as a stub resolver and a system
   acting as a recursive resolver.

   There is a desire for programs running in Javascript in browsers to
   be able to make DNS requests, particularly to get DNSSEC-protected
   responses such as for DANE [RFC6698] queries. The design in this
   document allows Javascript and other languages and environments that
   require connections to come from URLs to perform DNS requests.

   This document defines how to easily wrap DNS queries in HTTP requests
   and interpret DNS responses in the HTTP respones; the HTTP here is
   always run under TLS on port 443. Using HTTP-under-TLS as a

substrate was chosen for many of the reasons given in [RFC3205]. The
specification in this document follows the restrictions of RFC 3205,
including using generic HTTP clients and servers, not adding
restrictions on HTTP, and so on. It is expected that this protocol
would work just fine (maybe even better) under HTTP/2
[I-D.ietf-httpbis-http2].

Because there is currently no expectation of privacy for DNS queries,
this document defines the use of opportunistic security as described
in [I-D.dukhovni-opportunistic-security] for adding privacy for DNS
traffic between a stub resolver and a recursive resolver.

The protocol described in this document cannot be used by a stub
resolver to trust the DNSSEC validation status of responses from a
recursive server. Such trust might be described in a different
protocol that always uses authenticated TLS, but not the one here.

1.1. Other Designs

There have been many designs proposed for using TLS to protect DNS
traffic between a stub resolver and a recursive resolver. Among them
are:

o [draft-hzhwm-dprive-start-tls-for-dns] describes DNS over TCP
   begun on port 53 as normal, but there is an in-band signal to
   change the transport to TLS.

o [draft-hoffman-dprive-dns-tls-alpn] describes DNS over TCP begun
   on port 443, with ALPN [RFC7301] being used to specify that the
   DNS protocol will be run after TLS is set up.

o [draft-hoffman-dprive-dns-tls-newport] describes DNS over TCP is
   begun on a port specific to the protocol.

(Yet a different design, call DNSCrypt, has a fair amount of
deployment. A pointer will be added here for the technical
specification of that design if it becomes available.)

1.2. 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 RFC
2119, BCP 14 [RFC2119].

The roles of agents that make DNS requests, and those that give DNS
responses have been loosely named over time. Because this protocol
is meant to be used between specific types of agents, they need to be

defined here. [[Note: if these are adequately defined in existing
RFCs in ways that the community agrees on, it would be better to
simply repeat those definitions.]]

Stub resolver: A system that sends DNS queries with the intention of
   using the answers locally.

Authoritative server: A system that responds to DNS queries with
   information about zones for which it is authoritative.

Recursive resolver: A system that receives DNS queries and either
   responds to those queries from a local cache or sends queries to
   authoritative servers in order to get the answers to the original
   queries. These systems are also commonly called "recursive
   servers".

DNS forwarder: A system receives a DNS query from a stub resolver,
   possibly changes the query, sends the resulting query to a
   recursive resolver, receives the response from the recursive
   resolver, possibly changes the response, and sends the resulting
   response to the stub resolver. [RFC5625] does not give a specific
   definition for DNS forwarder, but describes in detail what
   features they need to support. The protocol interfaces for DNS
   forwarders are exactly the same as those for recursive resolvers
   (for interactions with DNS stubs) and as those for stub resolvers
   (for interactions with recursive resolvers).

2. Specification of Using HTTPS Between a DNS Stub Resolver and a Recursive Resolver

A stub resolver MAY attempt to communicate with a recursive resolver
using TLS [RFC5246] over port 443.

An https: URI [RFC3986] is resolved. The URI uses the "/.well-
known/" prefix defined in [RFC5785].

The URI is marshaled as follows:

1. The URI scheme MUST be "https:". (To restate the obvious, the
    URI scheme MUST NOT be "http:" or any other scheme.)

2. The authority MAY be a domain name, but is much more likely to be
    an IP address.

3. A port number MAY be specified, but if it is not present, port
    443 is assumed.

4. The path begins with ".well-known/dns-in-https/".

5. The octets in the DNS request (defined in [RFC1035] and all the
    relevant updates) are converted to base64url encoding from
    [RFC4648] and appended to the path.

The URI is resolved using a standard HTTP client, such as the "curl"
or "wget" tools or the libraries that support them.

If the HTTP request is successful, the server uses an HTTP 200
response and sends back a single part that is of type application/
dns-response. The body of the response is the octets of the DNS
response. Note that a DNS request that returns a DNS error is still
considered an HTTP request that is successful and should be served
with a 200 response.

If the request is not successful, the server might return HTTP
responses in the 400 or 500 ranges with empty bodies. Note that HTTP
response in the 300 range are also possible, such as if the DNS
server has moved.

For example, a request URI would look as follows (with a line break
due to publication limits):

https://8.8.8.8/.well-known/dns-in-https/
    TN4AAAABAAAAAAAAB2V4YW1wbGUDY29tAAABAAE=

This example is based on a request for the A record for example.com.
The set of octets in the query (expressed here in hex notation) is:

0x4CDE00000001000000000000076578616D706C6503636F6D0000010001

2.1. Design Rationale

A recursive resolver SHOULD offer authentication using one or more of
the many methods allowed by TLS, and the stub resolver SHOULD
authenticate the recursive resolver if it can. However, if the stub
resolver cannot authenticate the recursive resolver during TLS setup,
the stub resolver SHOULD still complete the handshake in order to
achieve encrypted communication.

A typical form of authentication for a recursive resolver would be a
PKIX [RFC5280] certificate that has a CommonName (CN) that is the IP
address that stub resolvers use to connect to it. Note that there
are many other standardized types of TLS authentication that can be
used, such as raw public keys keys [RFC7250].

The TLS connection is kept up for as long as each party is willing to
do so.

2.2. Stub Resolver Policy

A stub resolver MAY use policy to allow unauthenticated encryption (which can possibly be intercepted by an on-path adversary) or authenticated encryption (which might prevent all DNS resolution if the server does not have correct authentication credentials) when contacting a recursive resolver using this protocol.

It is expected that users will want one of the following policies available to them:

o The stub resolver MUST achieve authenticated TLS with a recursive
   server; if that can't be achieved, the stub resolver refuses to
   send out DNS queries

o The stub resolver tries to achieve authenticated TLS with a
   recursive server; if it cannot achieve authenticated TLS, it tries
   to achieve unauthenticated TLS; if that can't be achieved, the
   stub resolver refuses to send out DNS queries

o The stub resolver tries to achieve authenticated TLS with a
   recursive server; if it cannot achieve authenticated TLS, it tries
   to achieve unauthenticated TLS; if that can't be achieved, the
   stub resolver uses normal DNS cleartext on port 53

o The stub resolver doesn't want to try TLS at all, and uses normal
   DNS cleartext on port 53

2.3. Privacy Through DNS Forwarders

A stub resolver cannot tell whether it is sending queries to a recursive resolver or to a DNS forwarder. Therefore, a DNS forwarder that acts as a TLS server for DNS requests SHOULD attempt to use TLS with its upstream resolver(s) to maximize the confidentiality of its stub clients.

2.4. Use by Authoritative Servers

There is absolutely no expectation that any authoritative server will deploy this protocol. Thus, a DNS recursive resolver that tries to contact an authoritative server on TCP port 443 in hopes of keeping its communication private is probably wasting its time and delaying getting the actual answer over port 53.

[3](). Privacy Considerations

This entire document is about improving privacy for DNS requests and responses.

[4](). IANA Considerations

[4.1](). Well-Known URI

IANA is requested add the following value to the "Well-Known URIs" registry. That registry is populated by expert review, and such a review will be requested if this document progresses.

```
URI suffix:                dns-in-https
Change controller:         IETF
Specification document(s): This document
Related information:       None
```

[4.2](). Media Type

IANA is requested add the following value to the "Media Types" registry. That registry is populated by expert review, and such a review will be requested as this document progresses.

```
Type name: application
Subtype name: dns-response
Required parameters: N/A
Optional parameters: N/A
Encoding considerations: N/A
Security considerations: Given in this document
Interoperability considerations: N/A
Published specification: This document
Applications that use this media type: This document
Fragment identifier considerations: N/A
Additional information: None
Person & email address to contact for further information:
  Paul Hoffman, paul.hoffman@vpnc.org
Intended usage: COMMON
Restrictions on usage: N/A
Author: Paul Hoffman
Change controller: IESG
Provisional registration? (standards tree only): No
```

[5](). Security Considerations

An adversary who can observe encrypted queries from stub resolvers, and can simultaneously observe the cleartext queries from a recursive resolver to authoritative servers, might be able to associate those

two sets of queries and thus ascertain that a particular client asked
a particular query. Such observations can be prevented by the
recursive resolver already having the answer in its cache. If a
recursive resolver has ample room in its cache, it can make the
adversary's job harder by refreshing entries in its cache before the
TTL on those entries time out, thereby preventing the adversary's
ability to associate encrypted queries with cleartext ones.

6. Acknowledgements

Many people have thought about protecting DNS queries and responses,
and various discussions with those people resulted in this document.

The following have made significant contributions to this document:
Jacob Appelbaum, Carsten Bormann, Tatuya JINMEI, Warren Kumari, and
Paul Wouters.

7. References

7.1. Normative References

[I-D.ietf-httpbis-http2]
          Belshe, M., Peon, R., and M. Thomson, "Hypertext Transfer
          Protocol version 2", draft-ietf-httpbis-http2-14 (work in
          progress), July 2014.

[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.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform
          Resource Identifier (URI): Generic Syntax", STD 66, RFC
          3986, January 2005.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data
          Encodings", RFC 4648, October 2006.

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

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known
          Uniform Resource Identifiers (URIs)", RFC 5785, April
          2010.

7.2. Informative References

   [I-D.bortzmeyer-dnsop-dns-privacy]
              Bortzmeyer, S., "DNS privacy considerations", draft-
              bortzmeyer-dnsop-dns-privacy-02 (work in progress), April
              2014.

   [I-D.dukhovni-opportunistic-security]
              Dukhovni, V., "Opportunistic Security: Some Protection
              Most of the Time", draft-dukhovni-opportunistic-
              security-04 (work in progress), August 2014.

   [RFC3205] Moore, K., "On the use of HTTP as a Substrate", BCP 56,
              RFC 3205, February 2002.

   [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.

   [RFC5625] Bellis, R., "DNS Proxy Implementation Guidelines", BCP
              152, RFC 5625, August 2009.

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

   [RFC7250] Wouters, P., Tschofenig, H., Gilmore, J., Weiler, S., and
              T. Kivinen, "Using Raw Public Keys in Transport Layer
              Security (TLS) and Datagram Transport Layer Security
              (DTLS)", RFC 7250, June 2014.

   [RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan,
              "Transport Layer Security (TLS) Application-Layer Protocol
              Negotiation Extension", RFC 7301, July 2014.

   [draft-hoffman-dprive-dns-tls-alpn]
              Hoffman, P., "Using TLS and ALPN for Privacy Between DNS
              Stub and Recursive Resolvers", draft-hoffman-dns-tls-alpn
              , October 2014.

   [draft-hoffman-dprive-dns-tls-newport]
              Hoffman, P., "Using TLS on a New Port for Privacy Between
              DNS Stub and Recursive Resolvers", draft-hoffman-dns-tls-
              newport , October 2014.

   [draft-hzhwm-dprive-start-tls-for-dns]
                Hu, Z., "TLS for DNS: Initiation and Performance
                Considerations", draft-hzhwm-dprive-start-tls-for-dns ,
                October 2014.

Author's Address

   Paul Hoffman
   VPN Consortium

   Email: paul.hoffman@vpnc.org