Internet Engineering Task Force
Internet-Draft
Intended status: Experimental
Expires: August 20, 2016

L. Song
S. Kerr
R. Wan
Beijing Internet Institute
February 17, 2016

DNS wire-format over HTTP
draft-song-dns-wireformat-http-00

Abstract

This memo introduces a way to tunnel DNS data over HTTP. This may be useful in any situation where DNS is not working properly, such as when there is middlebox misbehavior.

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 August 20, 2016.

Copyright Notice

Table of Contents

1. Introduction

RFC 1035 [RFC1035] specifies the wire format for DNS messages. It also specifies DNS transport on UDP and TCP on port 53, which is still used today. However, there are other ways to access DNS database, for example in a different data format or via alternative DNS transport. These approaches are summarized in [draft-shane-review-DNS-over-http].

One of alternative way of DNS described in that document is to transport DNS binary data inside HTTP, with the goal of improving DNS service availability. The DNS traffic is simply sent as web traffic using port 80/443 over HTTP. It can bypass badly behaving middle boxes like firewalls, proxies or traffic shaping devices on path which might interfere with normal DNS traffic [RFC5625] [DOTSE] [SAC035].

This approach has the advantage that HTTP usually makes it through even the worst coffee shop or hotel room firewalls, as Internet users expect web browsing to always work. It also benefits from HTTP's support for persistent TCP connections (see section 6.3 in [RFC7230]) and multiplexing (see section 5 in [RFC7540]). Note that 5966bis [I-D.ietf-dnsop-5966bis] specifies the persistent and query pipelining features for DNS on TCP port 53; HTTP is easy and mature to deploy.Finally, developers can choose HTTPS to provides data integrity and privacy as well.

Unlike a REST DNS API using JSON [I-D.bortzmeyer-dns-json] or XML [I-D.mohan-dns-query-xml] encoding for DNS data, in this approach wire-format data is wrapped with a HTTP header and transmitted on port 80 or 443. This protocol is not designed for web developers or advanced usage in application development in mind. It simply serves as a sort of DNS VPN, and does not introduce another format of DNS data.

This memo aims to describe how the DNS binary over HTTP concept works. Hopefully implementations by different developers following this memo can speak with each other.

This mechanism is designed for DNS transaction between client stub resolver with recursive server. Interaction with authority server like normal queries, DNS zone transfer, DNS updates are out-of-scope for this document.

2. Methodology and Configuration

As mentioned in introduction, the basic methodology is wrapping the DNS wire-format data into an HTTP header and transmitting on port 80 or 443; however, there are two different scenarios for implementation.

Scenario 1:

The DNS server implementation handles DNS queries and responses via both UDP and TCP on port 53, and HTTP on port 80 or 443. It works as follows:

1. The client creates a DNS query message.

2. The client encapsulates the DNS message in a HTTP message body and assigns parameters with the HTTP header.

3. The client connects to the server and issues an HTTP POST request method. This may re-use an existing HTTP connection.

4. The server decapsulates the HTTP packet to get the DNS query, and resolves the DNS query.

5. The server encapsulates the DNS response in HTTP and sends it back via the HTTP session.

Scenario 2:

In this scenario there is a DNS-HTTP proxy sitting between stub-resolver and the recursive server. The stub uses a client proxy and the recursive server uses a server proxy. This works like a DNS VPN and transmits wire-format DNS messages over HTTP between the proxy client and a server, as follows:

1. The stub-resolver sends a DNS query (over UDP or TCP) to the proxy client.

2. The proxy client encapsulates the DNS message in an HTTP message body and assigns parameters with the HTTP header.

3. The proxy client connects to the proxy server and issues an HTTP POST request method. This may re-use an existing HTTP connection.

4. The proxy server decapsulates the HTTP packet to get the DNS query, and sends it to a real DNS server over UDP/TCP.

5. The proxy server encapsulates the DNS response in HTTP and sends it back via the HTTP session.

6. The proxy client decapsulates the DNS message from the HTTP response and sends it back to the stub-resolver via previous DNS session (either UDP or TCP).

Note that the proxy client can be implemented listening to a loop-back address in the same host with stub-resolver. The proxy server can be implemented as a caching server as well. It is also possible to use the proxy server as a regular web server at the same time that it is acting as a proxy server.

3. DNS-over-HTTP Message Format

Although this document simply proposes that HTTP be used for DNS massage exchange, it is informative to explore the context with HTTP-specific semantics, such as the HTTP request method, specific request-target, HTTP-version, and HTTP header field, all of which have impact on interoperability as well as performance.

In HTTP 1.1 specification section 3 in RFC 7230 [RFC7230], an HTTP message consists of three parts: start-line, header fields, and message body with an empty line between header fields and message body. The DNS over HTTP message also contains these parts.

3.1. Start-line

By definition the HTTP start-line is either a request-line (for requests) or a status-line (for responses). A request-line consists of the request method, request-target, and HTTP-version with a CRLF in the end.

For a DNS query over HTTP, the request is always POST [section 4.3.3 in RFC 7230 [RFC7230]]. Note if a GET is sent to the server, it optionally returns a human-readable page showing its web server environment.

Usually the target URI is provides explicitly by the DNS services provider. Derived from the target URI, the request-target in request-line identifies the target resource upon which to apply the request. In the case of DNS query over HTTP, one approach is to always use a fixed string for request-target by default, like "/dns-over-http". It works if there is no other resource sharing the same request-target name, otherwise there may be a conflict. Another approach is to allow the server to use a defined URI which contains the specific request-target. Note in the BII proxy implementation, we use "/proxy_dns" for this purpose.

DNS transaction over HTTP has no specific requirement for transport protocol; developers can use any version of HTTP to accomplish the transaction. But developers should be aware that HTTP/1.1 RFC 7230 [RFC7230] and HTTP/2 RFC 7540 [RFC7540] do have differences in performance regarding multiplexing. HTTP/2 is fully multiplexed, instead of ordered and blocking. Because there is a general desire to achieve similar performance with DNS over UDP, the modern HTTP/2 is preferred for DNS over HTTP implementation. Note that there should be no problem for advanced HTTP protocol in the future deployed for DNS over HTTP.

For example, given HTTP-version is "HTTP/1.1". So the request-line is :

POST /dns-over-http HTTP/1.1

The status-line returns the status-code [Section 6 of RFC 7231 [RFC7231]], which only reflects status of HTTP connection. If the request has succeeded, the status-line is:

HTTP/1.1 200 OK

If the request fails, the proxy server will supply an appropriate error code, typically 4xx (client error) if the client has provided a query that the server cannot understand for some reasons, or 5xx (server error) if some server-side problem prevented a query from succeeding.

3.2. Header Fields

By definition header fields are key:value pairs that can be used to communicate data about the message, its payload, the target resource, or the connection (for example control data). Considering the proxy configuration in scenario 2, we use a new header field:

Proxy-DNS-Transport: xyz

Where xyz is either UDP or TCP, which is the client's indication of how it received the underlying DNS query, and which the server will use when sending the query to the far-end DNS server. This means if a stub DNS client asks for TCP, then that's what the far-end DNS server will see, and likewise for UDP. This header field is used for both request and response, for all DNS over HTTP message.

3.3. Message Body

As mentioned, the message body is DNS wire-format data.

It is worth mentioning that DNS messages sent over TCP connections is prefixed with a two-byte length field which gives the message length[section 4.2.2 in RFC 1035 [RFC1035]], excluding the two-byte length field. This length field allows the low-level processing to assemble a complete message before beginning to parse it. In the context of HTTP, there is content-length header filed [section 3.3.2 in RFC 7230 [RFC7230]]in which the field-value is the same with two bytes length field in DNS over TCP.

In normal configuration, both client and server should assign a value for the DNS message length to HTTP content-length field. In proxy configuration, if Proxy-DNS-Transport is TCP, the proxy must aware the two-byte length field in DNS message over TCP before it handle HTTP request and response.

4. Considerations on DNS over HTTPS

HTTPS is recommended for communication. It provides privacy and integrity for HTTP sessions.

As specified in RFC 5246 [RFC5246], both the DNS server and client can be authenticated or not authenticated. The DNS service providers can decide authenticated pattern on both server and client sides based on their own requirement for security and privacy. For an open resolver, clients do not require authentication.

5. IANA considerations

Registration for a new HTTP header field: Proxy-DNS-Transport

6. References

[DOTSE] and , "DNSSEC Tests of Consumer Broadband Routers", February 2008, <http://www.iis.se/docs/Routertester_en.pdf>.

[I-D.bortzmeyer-dns-json]
Bortzmeyer, S., "JSON format to represent DNS data", draft-bortzmeyer-dns-json-01 (work in progress), February 2013.

[I-D.ietf-dnsop-5966bis]
Dickinson, J., Dickinson, S., Bellis, R., Mankin, A., and D. Wessels, "DNS Transport over TCP - Implementation Requirements", draft-ietf-dnsop-5966bis-04 (work in progress), November 2015.

[I-D.mohan-dns-query-xml]
Parthasarathy, M. and P. Vixie, "Representing DNS messages using XML", draft-mohan-dns-query-xml-00 (work in progress), September 2011.

[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>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC5625] Bellis, R., "DNS Proxy Implementation Guidelines", BCP 152, RFC 5625, DOI 10.17487/RFC5625, August 2009, <http://www.rfc-editor.org/info/rfc5625>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <http://www.rfc-editor.org/info/rfc7230>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <http://www.rfc-editor.org/info/rfc7231>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <http://www.rfc-editor.org/info/rfc7540>.

[SAC035] ICANN Security and Stability Advisory Committee, "DNSSEC Impact on Broadband Routers and Firewalls", 2008.

Authors' Addresses

Linjian Song
Beijing Internet Institute
2508 Room, 25th Floor, Tower A, Time Fortune
Beijing 100028
P. R. China

Email: songlinjian@gmail.com
URI: http://www.biigroup.com/

Shane Kerr
Beijing Internet Institute
2/F, Building 5, No.58 Jinghai Road, BDA
Beijing 100176
CN

Email: shane@biigroup.cn
URI: http://www.biigroup.com/

Runxia Wan
Beijing Internet Institute
2508 Room, 25th Floor, Tower A, Time Fortune
Beijing 100028
P. R. China

Email: rxwan@biigroup.cn
URI: http://www.biigroup.com/