The CONNECT-UDP HTTP Method

Abstract

   This document describes the CONNECT-UDP HTTP method. CONNECT-UDP is
   similar to the HTTP CONNECT method, but it uses UDP instead of TCP.

   Discussion of this work is encouraged to happen on the MASQUE IETF
   mailing list masque@ietf.org or on the GitHub repository which
   contains the draft: https://github.com/DavidSchinazi/masque-drafts.

Discussion Venues

   This note is to be removed before publishing as an RFC.

   Source for this draft and an issue tracker can be found at https://
   github.com/ietf-wg-masque/draft-ietf-masque-connect-udp.

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 https://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 28 February 2021.

Copyright Notice

Table of Contents

1. Introduction

This document describes the CONNECT-UDP HTTP method. CONNECT-UDP is
similar to the HTTP CONNECT method (see section 4.3.6 of [RFC7231]),
but it uses UDP [UDP] instead of TCP [TCP].

Discussion of this work is encouraged to happen on the MASQUE IETF
mailing list masque@ietf.org or on the GitHub repository which
contains the draft: https://github.com/DavidSchinazi/masque-drafts.

1.1. Conventions and Definitions

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
BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all
capitals, as shown here.

2. Supported HTTP Versions

The CONNECT-UDP method is defined for all versions of HTTP. When the
HTTP version used runs over QUIC [QUIC], UDP payloads can be sent
over QUIC DATAGRAM frames [DGRAM]. Otherwise they are sent on the

stream where the CONNECT-UDP request was made. Note that when
multiple proxies are involved in a CONNECT-UDP request, all the HTTP
connections along the path need to be using HTTP/3 [H3] or later in
order for UDP payloads to be sent over QUIC DATAGRAM frames.
Additionally, when the HTTP version in use does not support
multiplexing streams (such as HTTP/1.1), then any reference to
"stream" in this document is meant to represent the entire
connection.

3. The CONNECT-UDP Method

The CONNECT-UDP method requests that the recipient establish a
tunnel over a single HTTP stream to the destination origin server
identified by the request-target and, if successful, thereafter
restrict its behavior to blind forwarding of packets, in both
directions, until the tunnel is closed. Tunnels are commonly used to
create an end-to-end virtual connection, through one or more
proxies, which can then be secured using QUIC or another protocol
running over UDP.

A client sending a CONNECT-UDP request MUST send the authority form
of request-target (Section 5.3 of [RFC7230]); i.e., the request-
target consists of only the host name and port number of the tunnel
destination, separated by a colon. For example,

```
CONNECT-UDP server.example.com:443 HTTP/1.1
Host: server.example.com:443
```

When using HTTP/2 [H2] or later, CONNECT-UDP requests use HTTP
pseudo-headers with the following requirements:

  *The ":method" pseudo-header field is set to "CONNECT-UDP".

  *The ":scheme" and ":path" pseudo-header fields MUST be omitted.

  *The ":authority" pseudo-header field contains the host and port
   to connect to (equivalent to the authority-form of the request-
   target of CONNECT-UDP requests (see [RFC7230], Section 5.3)).

A CONNECT-UDP request that does not conform to these restrictions is
malformed (see [H2], Section 8.1.2.6).

The recipient proxy can establish a tunnel either by directly
opening a UDP socket to the request-target or, if configured to use
another proxy, by forwarding the CONNECT-UDP request to the next
inbound proxy. Any 2xx (Successful) response indicates that the
sender (and all inbound proxies) will switch to tunnel mode
immediately after the blank line that concludes the successful
response's header section; data received after that blank line is
from the server identified by the request-target. Any response other

than a successful response indicates that the tunnel has not yet
been formed and that the connection remains governed by HTTP.

A tunnel is closed when a tunnel intermediary detects that either
side has closed its connection: the intermediary MUST attempt to
send any outstanding data that came from the closed side to the
other side, close both connections, and then discard any remaining
data left undelivered.

A server MUST NOT send any Transfer-Encoding or Content-Length
header fields in a 2xx (Successful) response to CONNECT. A client
MUST treat a response to CONNECT-UDP containing any Content-Length
or Transfer-Encoding header fields as malformed.

A payload within a CONNECT-UDP request message has no defined
semantics; a CONNECT-UDP request with a non-empty payload is
malformed.

Responses to the CONNECT-UDP method are not cacheable.

4. Encoding of Proxied UDP Packets

When the HTTP connection between client and proxy supports HTTP/3
datagrams [H3DGRAM], UDP packets can be encoded using QUIC DATAGRAM
frames. This support is ascertained by checking receipt of the
H3_DATAGRAM SETTINGS Parameter. Note that when there are multiple
proxies involved, this support needs to be ascertained on all the
HTTP connections that will carry proxied UDP packets.

If the client supports HTTP/3 datagrams and has received the
H3_DATAGRAM SETTINGS Parameter on this connection, it SHOULD attempt
to use HTTP/3 datagrams. This is accomplished by requesting a
datagram flow identifier from the flow identifier allocation service
[H3DGRAM]. That service generates an even flow identifier, and the
client sends it to the server by using the "Datagram-Flow-Id" header
(see Section 5).

If there are multiple proxies involved, proxies along the chain MUST
check whether their upstream connection supports HTTP/3 datagrams.
If it does not, that proxy MUST remove the "Datagram-Flow-Id" header
before forwarding the CONNECT-UDP request.

The proxy that is creating the UDP socket to the destination
responds to the CONNECT-UDP request with a 2xx (Successful)
response, and MUST echo the "Datagram-Flow-Id" header. Once the
client has received the "Datagram-Flow-Id" header on the successful
response, it knows that it can use the HTTP/3 datagram encoding to
send proxied UDP packets for this particular destination. It then
encodes the payload of UDP datagrams into the payload of HTTP/3
datagrams.

Clients MAY optimistically start sending proxied UDP packets before receiving the response to its CONNECT-UDP request, noting however that those may not be processed by the proxy if it responds to the CONNECT-UDP request with a failure, or if they arrive before the CONNECT-UDP request.

If HTTP/3 datagrams are not supported, the stream is used to convey UDP payloads, by prefixing them with a 16-bit length.

5. Datagram-Flow-Id Header Definition

"Datagram-Flow-Id" is a Item Structured Header [STRUCT-HDR]. Its value MUST be an Integer. Its ABNF is:

```
Datagram-Flow-Id = sh-integer
```

6. Server Handling

Unlike TCP, UDP is connection-less. The HTTP server that opens the UDP socket has no way of knowing whether the destination is reachable. Therefore it needs to respond to the CONNECT-UDP request without waiting for a TCP SYN-ACK.

Servers can use connected UDP sockets if their operating system supports them, as that allows the HTTP server to rely on the kernel to only send it UDP packets that match the correct 5-tuple. If the server uses a non-connected socket, it MUST validate the IP source address and UDP source port on received packets to ensure they match the client's CONNECT-UDP request. Packets that do not match MUST be discarded by the server.

7. Security Considerations

There are significant risks in allowing arbitrary clients to establish a tunnel to arbitrary servers, as that could allow bad actors to send traffic and have it attributed to the proxy. Proxies that support CONNECT-UDP SHOULD restrict its use to authenticated users.

8. IANA Considerations

8.1. HTTP Method

This document will request IANA to register "CONNECT-UDP" in the HTTP Method Registry (IETF review) maintained at <https://www.iana.org/assignments/http-methods>.

```
+-------------+------+-----------+---------------+
| Method Name | Safe | Idempotent |   Reference   |
+-------------+------+-----------+---------------+
| CONNECT-UDP |  no  |     no    | This document |
+-------------+------+-----------+---------------+
```

8.2. HTTP Header

This document will request IANA to register the "Datagram-Flow-Id"
header in the "Permanent Message Header Field Names" registry
maintained at <https://www.iana.org/assignments/message-headers>.

```
+-------------------+----------+--------+---------------+
| Header Field Name | Protocol | Status |   Reference   |
+-------------------+----------+--------+---------------+
| Datagram-Flow-Id  |   http   |  exp   | This document |
+-------------------+----------+--------+---------------+
```

9. Normative References

[DGRAM]   Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable
           Datagram Extension to QUIC", Work in Progress, Internet-
           Draft, draft-ietf-quic-datagram-01, 24 August 2020,
           <http://www.ietf.org/internet-drafts/draft-ietf-quic-
           datagram-01.txt>.

[H2]   Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext
           Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI
           10.17487/RFC7540, May 2015, <https://www.rfc-editor.org/
           info/rfc7540>.

[H3]   Bishop, M., "Hypertext Transfer Protocol Version 3 (HTTP/
           3)", Work in Progress, Internet-Draft, draft-ietf-quic-
           http-29, 9 June 2020, <http://www.ietf.org/internet-
           drafts/draft-ietf-quic-http-29.txt>.

[H3DGRAM] Schinazi, D., "Using QUIC Datagrams with HTTP/3", Work in
           Progress, Internet-Draft, draft-schinazi-quic-h3-
           datagram-04, 16 April 2020, <http://www.ietf.org/
           internet-drafts/draft-schinazi-quic-h3-datagram-04.txt>.

[QUIC]   Iyengar, J. and M. Thomson, "QUIC: A UDP-Based
           Multiplexed and Secure Transport", Work in Progress,
           Internet-Draft, draft-ietf-quic-transport-29, 9 June
           2020, <http://www.ietf.org/internet-drafts/draft-ietf-
           quic-transport-29.txt>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
           Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/
```

                    RFC2119, March 1997, <https://www.rfc-editor.org/info/
                    rfc2119>.

     [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,
                    <https://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, <https://www.rfc-
                    editor.org/info/rfc7231>.

     [RFC8174]      Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
                    2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
                    May 2017, <https://www.rfc-editor.org/info/rfc8174>.

     [STRUCT-HDR]   Nottingham, M. and P. Kamp, "Structured Field Values
                    for HTTP", Work in Progress, Internet-Draft, draft-ietf-
                    httpbis-header-structure-19, 3 June 2020, <http://
                    www.ietf.org/internet-drafts/draft-ietf-httpbis-header-
                    structure-19.txt>.

     [TCP]          Postel, J., "Transmission Control Protocol", STD 7, RFC
                    793, DOI 10.17487/RFC0793, September 1981, <https://
                    www.rfc-editor.org/info/rfc793>.

     [UDP]          Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI
                    10.17487/RFC0768, August 1980, <https://www.rfc-
                    editor.org/info/rfc768>.

## Acknowledgments

## Author's Address

   David Schinazi
   Google LLC
   1600 Amphitheatre Parkway
   Mountain View, California 94043,
   United States of America

   Email: dschinazi.ietf@gmail.com