Workgroup: MASQUE
Internet-Draft:
draft-duke-masque-other-transport-00
Published: 25 January 2021
Intended Status: Experimental
Expires: 29 July 2021
Authors: M. Duke
F5, Inc.

The Other-Transport Extension: Arbitrary Transports over CONNECT-UDP

Abstract

This document describes an extension to the HTTP CONNECT-UDP method [CONNECTUDP] that supports tunneling of other transport protocols, as long as the first four octets of those protocols encode the source and destination ports.

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 29 July 2021.

Table of Contents

1. Introduction

The HTTP CONNECT method (section 4.3.6 of [RFC7231]) has long provided a means of tunneling a TCP connection over an HTTP stream. The CONNECT-UDP method [CONNECTUDP] extends this capability to include UDP datagrams over a stream.

As CONNECT-UDP delivers discrete datagrams to each endpoint, it can extend conceptually to any packetized protocol. The Other-Transport extension allows a CONNECT-UDP proxy to tunnel packets with non-TCP, non-UDP protocol numbers, as long as the corresponding protocol meets minimal formatting requirements.

Specifically, any protocol header where the first four octets encode the source and destination ports can be tunneled using this framework. The client and proxy include all other protocol header information in the datagrams delivered over the tunnel. For example, 33 (DCCP, [RFC4330]); 132 (SCTP, [RFC4960]); and 136 (UDPLite, [RFC3828]) would all be valid candidates for Other-Transport.

In principle, TCP can be proxied using this extension as well. This might provide advantages over traditional HTTP CONNECT if the client's TCP implementation has features lacking at the proxy.

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.

In this document, we use the term "proxy" to refer to the HTTP server that opens the UDP socket and responds to the CONNECT-UDP request. If there are HTTP intermediaries (as defined in Section 2.3 of [RFC7230]) between the client and the proxy, those are referred to as "intermediaries" in this document.

2. Other-Transport Header Definition

"Other-Transport" is an Item Structured Header [HDRSTRUCT]. Its ABNF is:

```
Other-Transport = sf-integer
```

The value MUST be between 0 and 255, inclusive. Any other value is malformed. This value indicates the value of the Protocol Number (IPv4) or Next Header (IPv6) in IP headers corresponding to the CONNECT-UDP stream.

An Other-Transport header is ignored in any method other than CONNECT-UDP.

A client that sends this header MUST include the entire transport header, with the exception of the first four octets, in each HTTP/3 DATAGRAM or Stream Chunk payload it sends. It MUST process incoming datagrams with the same assumption.

When a client sends the Other-Transport header field, it MUST use a value that corresponds to a protocol whose first four octets of each packet correspond to the source and destination ports. For example, 33 (DCCP, [RFC4330]); 132 (SCTP, [RFC4960]); and 136 (UDPLite, [RFC3828]) would all be valid choices.

A proxy MUST NOT include this header unless it will prepend and strip port numbers instead of entire UDP headers, and use the Protocol Number in IP headers for both packets to the server and for demultiplexing incoming server packets.

A proxy MAY choose not to send the header due to policy regarding specific protocol numbers.

This extension is said to have been negotiated when both client and proxy indicate support for it in their CONNECT-UDP request and response using the same value.

If the server responds without the Other-Transport header, the client may either proceed with UDP datagrams or close the stream.

A response with a Other-Transport value other than that provided by the client is malformed.

3. Datagram Encoding of Proxied Packets

All DATAGRAM frames corresponding to the negotiated Datagram-Flow-Id headers are processed in accordance with the Other-Transport extension, if negotiated.

All DATAGRAM frames MUST include the entire IP payload with the exception of the first four octets.

If a client sent an Other-Transport header, it MUST NOT send DATAGRAM frames until it confirms this extension has been negotiated. If the proxy does not support Other-Transport, it will interpret DATAGRAM frames as UDP payloads, with unpredictable results.

4. Stream Encoding of Proxied Packets

Endpoints use the 0x10 Stream Chunk Type to encode datagrams.

Clients MAY send payloads using Stream Chunks before negotiation is complete. Proxies that do not support the extension will simply ignore these chunks.

5. HTTP Intermediaries

HTTP Intermediaries that discover that an upstream proxy does not support the Other-Transport header MUST abort the stream in the direction of the client.

6. Security Considerations

CONNECT-UDP streams that use the Other-Transport header have similar security properties to other CONNECT-UDP streams, as described in [CONNECTUDP].

However, as more of the transport header originates at the server, and the tunneled protocols are less ubiquitous than UDP, these headers may serve to fingerprint the protocol implementation that generated them.

Furthermore, additional control over packet headers enhances the ability of clients to induce the proxy to generate certain packets, which might have undesirable effects in the network while being less traceable to the client.

7. IANA Considerations

7.1. HTTP Header

This document requests that IANA registers the "Other-Transport" header in the "Permanent Message Header Field Names" registry maintained at <https://www.iana.org/assignments/message-headers>.

Header Field Name	Protocol	Status	Reference
Other-Transport	http	std	This document

7.2. Stream Chunk Type Registration

This document will request IANA to register the following entry in the "CONNECT-UDP Stream Chunk Type" registry [CONNECTUDP]:

Value	Type	Description	Reference
0x10	OTHER_TRANSPORT	Other Transport Protocol	This document

8. References

8.1. Normative References

[CONNECTUDP] Schinazi, D., "The CONNECT-UDP HTTP Method", Work in Progress, Internet-Draft, draft-ietf-masque-connect-udp-03, 5 January 2021, <http://www.ietf.org/internet-drafts/draft-ietf-masque-connect-udp-03.txt>.

[HDRSTRUCT] 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>.

[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>.

[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>.

8.2. Informative References

[RFC3828] Larzon, L-A., Degermark, M., Pink, S., Jonsson, L-E., Ed., and G. Fairhurst, Ed., "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, DOI 10.17487/RFC3828, July 2004, <https://www.rfc-editor.org/info/rfc3828>.

[RFC4330] Mills, D., "Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI", RFC 4330, DOI 10.17487/RFC4330, January 2006, <https://www.rfc-editor.org/info/rfc4330>.

[RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <https://www.rfc-editor.org/info/rfc4960>.

[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>.

Acknowledgments

Author's Address

Martin Duke
F5, Inc.
801 5th Ave
Seattle, Washington, 98104,
United States of America

Email: martin.h.duke@gmail.com