Workgroup: MASQUE
Internet-Draft: draft-cms-masque-connect-ip-02
Published: 27 August 2021
Intended Status: Standards Track
Expires: 28 February 2022
Authors: A. Chernyakhovsky D. McCall D. Schinazi
Google LLC Google LLC Google LLC

The CONNECT-IP HTTP Method

Abstract

This document describes the CONNECT-IP HTTP method. CONNECT-IP is similar to CONNECT-UDP, but allows transmitting IP packets, without being limited to just TCP like CONNECT or UDP like CONNECT-UDP.

Discussion Venues

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

Discussion of this document takes place on the Multiplexed Application Substrate over QUIC Encryption Working Group mailing list (masque@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/masque/.

Source for this draft and an issue tracker can be found at https://github.com/DavidSchinazi/draft-cms-masque-connect-ip.

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 2022.

Copyright Notice

Table of Contents

1. Introduction

This document describes the CONNECT-IP HTTP method. CONNECT-IP is similar to CONNECT-UDP, but allows transmitting IP packets, without being limited to just TCP like CONNECT or UDP like CONNECT-UDP.

CONNECT-IP allows endpoints to set up an IP tunnel between one another. This can be used to implement a consumer VPN, point-to-point and point-to-network capabilities as described in [REQS].

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 responds to the CONNECT-IP 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. The CONNECT-IP Method

The CONNECT-IP method establishes a stream to an endpoint server that then permits the exchange of control data, such as IP address information and other relevant information for successfully transmitting IP datagrams between hosts.

The request-target of a CONNECT-IP request is a URI [URI] which uses the "https" scheme and a client-specified path. When using HTTP/2 [H2] or later, CONNECT-IP requests use HTTP pseudo-headers with the following requirements:

- The ":method" pseudo-header field is set to "CONNECT-IP".
- The ":scheme" pseudo-header field is set to "https".
- The ":path" pseudo-header field is set to the value provided by the client. That value MUST NOT be empty.
- The ":authority" pseudo-header field contains the host and port of the proxy. The target of a CONNECT-IP request is the server providing the CONNECT-IP featureset, not an individual endpoint with which a connection is desired.

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

Any 2xx (Successful) response indicates that the proxy is willing to open an IP tunnel between it and the client. Any response other than a successful response indicates that the tunnel has not yet been formed.

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

A payload within a CONNECT-IP request message has no defined semantics; a CONNECT-IP request with a non-empty payload is malformed. Note that the CONNECT-IP stream is used to convey control

messages, but they are not semantically part of the request or response themselves.

Responses to the CONNECT-IP method are not cacheable.

The lifetime of the tunnel is tied to the CONNECT-IP stream. Closing the stream (via the FIN bit on a QUIC STREAM frame, or a QUIC RESET_STREAM frame) closes the associated tunnel.

3. Transmitting IP Packets using HTTP Datagrams

IP packets are sent using HTTP Datagrams [HTTP-DGRAM]. The HTTP Datagram Payload contains a full IP packet, from the IP Version field until the last byte of the IP Payload. In order to use HTTP Datagrams, the CONNECT-IP client will first decide whether or not to use HTTP Datagram Contexts and then register its context ID (or lack thereof) using the corresponding registration capsule, see [HTTP-DGRAM].

Since HTTP Datagrams require prior negotiation (for example, in HTTP/3 it is necessary to both send and receive the H3_DATAGRAM SETTINGS Parameter), clients MUST NOT send any HTTP Datagrams until they have established support on a given connection. If negotiation of HTTP Datagrams fails (for example if an HTTP/3 SETTINGS frame was received without the H3_DATAGRAM SETTINGS Parameter), the client MUST consider its CONNECT-IP request as failed.

4. Forwarding of IP Packets

Since CONNECT-IP allows the transmission of IP packets over HTTP, CONNECT-IP endpoints will most often forward these packets to and from traditional IP interfaces. As such, CONNECT-IP endpoints act as IP routers. When a CONNECT-IP endpoint receives an HTTP Datagram containing an IP packet, it will parse the packet's IP header, perform any local policy checks (e.g., source address validation), check their routing table to pick an outbound interface, and then use an implementation-specific mechanism (such as raw sockets) to send the IP packet on that interface.

Conversely, when a CONNECT-IP endpoint receives an IP packet whose destination address does not match any local addresses, it consults its routing table to pick a forwarding destination, and if the table points to a CONNECT-IP tunnel, the endpoint performs the same forwarding checks before transmitting the packet inside the tunnel.

Note that CONNECT-IP endpoints will decrement the IP Hop Count (or TTL) upon encapsulation but not decapsulation. In other words, the Hop Count is decremented right before an IP packet is transmitted in an HTTP Datagram. This prevents infinite loops in the presence of routing loops, and matches the choices in IPsec [IPSEC].

Endpoints MAY implement additional filtering policies on the IP packets they forward.

5. Capsules

5.1. ADDRESS_ASSIGN Capsule

The ADDRESS_ASSIGN capsule allows an endpoint to inform its peer that it has assigned an IP address to it. It allows assigning a prefix which can contain multiple addresses. This capsule uses a Capsule Type of 0xfff100. Its value uses the following format:

```
ADDRESS_ASSIGN Capsule {
  IP Version (8),
  IP Address (32..128),
  IP Prefix Length (8),
}
```

Figure 1: ADDRESS_ASSIGN Capsule Format

IP Version: IP Version of this address assignment. MUST be either 4 or 6.

IP Address: Assigned IP address. If the IP Version field has value 4, the IP Address field SHALL have a length of 32 bits. If the IP Version field has value 6, the IP Address field SHALL have a length of 128 bits.

IP Prefix Length: Length of the IP Prefix assigned, in bits. MUST be lesser or equal to the length of the IP Address field, in bits.

5.2. ADDRESS_REQUEST Capsule

The ADDRESS_REQUEST capsule allows an endpoint to request assignment of an IP address from its peer. It allows the endpoint to optionally indicate a preference for which address it would get assigned. This capsule uses a Capsule Type of 0xfff101. Its value uses the following format:

```
ADDRESS_REQUEST Capsule {
  IP Version (8),
  IP Address (32..128),
  IP Prefix Length (8),
}
```

Figure 2: ADDRESS_REQUEST Capsule Format

IP Version:

IP Version of this address request. MUST be either 4 or 6.

IP Address: Requested IP address. If the IP Version field has value 4, the IP Address field SHALL have a length of 32 bits. If the IP Version field has value 6, the IP Address field SHALL have a length of 128 bits.

IP Prefix Length: Length of the IP Prefix requested, in bits. MUST be lesser or equal to the length of the IP Address field, in bits.

Upon receiving the ADDRESS_REQUEST capsule, an endpoint SHOULD assign an IP address to its peer, and then respond with an ADDRESS_ASSIGN capsule to inform the peer of the assignment.

5.3. SHUTDOWN Capsule

The SHUTDOWN capsule allows an endpoint to communicate to its peer that it is about to close the CONNECT-IP stream, with a string explaining the reason for the shutdown. This capsule uses a Capsule Type of 0xfff105. Its value uses the following format:

```
SHUTDOWN Capsule {
  Reason Phrase (..),
}
```

Figure 3: SHUTDOWN Capsule Format

Reason Phrase: Additional diagnostic information for the shutdown. This SHOULD be a UTF-8 encoded string [UTF8], though the frame does not carry information, such as language tags, that would aid comprehension by any entity other than the one that created the text.

Note that the SHUTDOWN capsule is informational, the tunnel is only closed when its corresponding CONNECT-IP stream is closed. Endpoints MAY close the tunnel with a reason phrase by sending the SHUTDOWN capsule with the FIN bit set on the underlying QUIC STREAM frame that carried it.

6. Extensibility Considerations

CONNECT-IP can be extended via multiple mechanisms to increase functionality. There are three main ways to extend CONNECT-IP: HTTP headers, Capsule Types, and HTTP Datagram Registration Extension Data. For example, an authentication extension could define an HTTP header that allows endpoints to send authentication credentials to their peer during the creation of the tunnel. Alternatively, one could specify an extension that defines a new Capsule Type which allows exchanging DNS configuration between endpoints. Additionally,

an extension to CONNECT-IP can use multiple HTTP Datagram Contexts [HTTP-DGRAM] simultaneously to compress some IP packets by associating the compression context with an HTTP Datagram Context ID.

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-IP SHOULD restrict its use to authenticated users. The HTTP Authorization header [AUTH] MAY be used to authenticate clients. More complex authentication schemes are out of scope for this document but can be implemented using CONNECT-IP extensions.

Since CONNECT-IP endpoints can proxy IP packets send by their peer, they SHOULD follow the guidance in [BCP38] to help prevent denial of service attacks.

8. IANA Considerations

8.1. HTTP Method

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

Method Name	Safe	Idempotent	Reference
CONNECT-IP	no	no	This document

8.2. Capsule Type Registrations

This document will request IANA to add the following values to the "HTTP Capsule Types" registry created by [HTTP-DGRAM]:

Value	Type	Description	Reference
0xfff100	ADDRESS_ASSIGN	Address Assignment	This document
0xfff101	ADDRESS_REQUEST	Address Request	This document
0xfff105	SHUTDOWN	Shutdown Reason	This document

9. References

9.1. Normative References

[BCP38] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <https://www.rfc-editor.org/rfc/rfc2827>.

[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/rfc/rfc7540>.

[HTTP-DGRAM] Schinazi, D. and L. Pardue, "Using Datagrams with HTTP", Work in Progress, Internet-Draft, draft-ietf-masque-h3-datagram-03, 12 July 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-masque-h3-datagram-03>.

[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/rfc/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/rfc/rfc7230>.

[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/rfc/rfc8174>.

[URI] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/rfc/rfc3986>.

[UTF8] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <https://www.rfc-editor.org/rfc/rfc3629>.

9.2. Informative References

[AUTH] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Authentication", RFC 7235, DOI 10.17487/RFC7235, June 2014, <https://www.rfc-editor.org/rfc/rfc7235>.

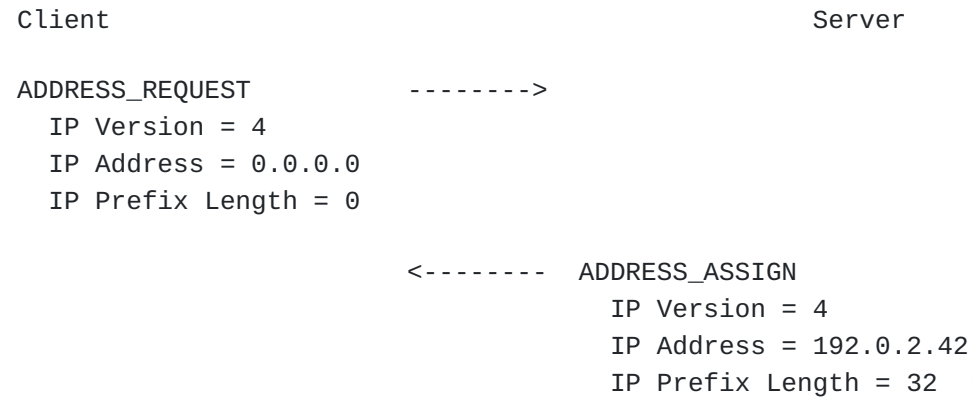
[IPSEC] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/rfc/rfc4301>.

[REQS]
Chernyakhovsky, A., McCall, D., and D. Schinazi, "Requirements for a MASQUE Protocol to Proxy IP Traffic", Work in Progress, Internet-Draft, draft-ietf-masque-ip-proxy-reqs-03, 27 August 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-masque-ip-proxy-reqs-03>.

Appendix A. Examples

A.1. Consumer VPN

In this scenario, the client will typically receive a single IP address that the proxy has picked from a pool of addresses it maintains. The client will route all traffic through the tunnel. The exchange could look as follows:



Acknowledgments

The design of CONNECT-IP was inspired by discussions in the MASQUE working group around [REQS]. The authors would like to thank participants in those discussions for their feedback.

Authors' Addresses

Alex Chernyakhovsky
Google LLC
1600 Amphitheatre Parkway
Mountain View, California 94043,
United States of America

Email: achernya@google.com

Dallas McCall
Google LLC
1600 Amphitheatre Parkway
Mountain View, California 94043,
United States of America

Email: dallasmccall@google.com

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

Email: dschinazi.ietf@gmail.com