webtrans
Internet-Draft
Intended status: Standards Track
Expires: 11 November 2026

M. Richter
Technische Universit辰t Berlin

10 May 2026

WebTransport over WebSocket
draft-richter-webtransport-websocket-05

Abstract

WebTransport [OVERVIEW], a protocol framework within the Web security model, empowers Web clients to initiate secure multiplexed transport for low-level client-server interactions with remote servers. This document outlines a protocol, based on WebSocket [WEBSOCKET], offering WebTransport capabilities similar to the HTTP/2 variant [WEBTRANSPORT-H2]. It serves as an alternative when UDP-based protocols are inaccessible, and the client environment exclusively supports WebSocket [WEBSOCKET].

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/martenrichter/draft-ietf-webtransport-websocket.

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 11 November 2026.

Copyright Notice

Table of Contents

1. Introduction

WebTransport [OVERVIEW] is designed to facilitate communication for Web clients over HTTP/3 [HTTP3], leveraging QUIC [QUIC] semantics with streams or datagrams [DATAGRAM]. In cases where UDP-based traffic is restricted, HTTP/2 protocol [WEBTRANSPORT-H2] serves as an alternative built solely on HTTP semantics.

Both [WEBTRANSPORT-H2] and [WEBTRANSPORT-H3] variants require a native WebClient implementation due to the usual unavailability of plain UDP and TCP/IP socket access for scripts within WebClients

This document defines a protocol that can be implemented on the WebClient using available scripting languages without altering the WebClient's native code. It uses the widespread WebSocket protocol as the base without modification. However, a direct implementation in a WebClient is possible.

The protocol utilizes capsule semantics derived from [WEBTRANSPORT-H2] and translates them into WebSocket frames. By relying on WebSockets, also intermediates such as proxies unaware of WebTransports can apply application-layer processing.

An implementation should support both WebSocket over http/1 and http/2. The server should incorporate WebTransport flow control constraints and capsule processing into its WebSocket parser code. Therefore, using unmodified existing WebSocket code is not recommended.

2. 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.

The document follows the terminology defined in Section 1.2 of [OVERVIEW].

3. Protocol Overview

WebTransport servers are identified by an HTTPS URI per Section 4.2.2 of [HTTP].

The protocol uses [WEBTRANSPORT-H2] semantics with the following modifications.

3.1. Connection, version negotiation and application level protocol

The WebSocket connection is established according to Section 4 of [WEBSOCKET] or [WEBSOCKET-H2].

When a WebSocket connection is established, both the client and server select the WebTransport-Websocket protocol by setting |Sec-WebSocket-Protocol| Section 1.9 of [WEBSOCKET] to the supported versions. The protocol names follow the scheme "webtransport_VERSIONNAME" or additionally "webtransport_VERSIONNAME_APPLICATIONLEVELPROTOCOL, where VERSIONNAME identifies the particular protocol version and APPLICATIONLEVELPROTOCOL an application-level protocol. For this protocol version, VERSIONNAME would be "kDraft3" and the |Sec-WebSocket-Protocol| field would include "webtransport_kDraft2" or additionally "webtransport_kDraft2_application_level_protocol" for an application-level protocol "application_level_protocol". The application level protocol is the same available handled via WT-

Available-Protocols and WT-Protocol described in Section 3.4 of [WEBTRANSPORT-H2] and Section 3.4 of [WEBTRANSPORT-H3]. The protocol negotiation follows the procedures as described in Section 4.1 of [WEBSOCKET] and Section 4.2.2 of [WEBSOCKET]. No protocol extensions MUST BE negotiated.

3.2. Data framing

The protocol uses the data frames as defined in Section 5 of [WEBSOCKET]. PING and PONG frame handling is not changed Section 5.5 of [WEBSOCKET].

For closing a session, a CLOSE_WEBTRANSPORT_SESSION capsule followed by the CLOSE frame Section 5.5.1 of [WEBSOCKET] is sent.

Data Frames containing Text are reserved for future use and MUST NOT be sent. Binary Data Frames transport CAPSULE content defined in [WEBTRANSPORT-H2] and [DATAGRAM]. For details, refer to the next section Section 3.3. Their length is limited by WebTransport flow control, and a violation SHOULD lead to connection termination. CONTINUATION frames are processed per [WEBSOCKET] specifications. Given the streaming nature of the content, partial DATA frames or CONTINUATION frames should be promptly forwarded to corresponding streams reducing latency.

3.3. Capsule frames

This protocol adopts the mechanisms and intrinsic elements outlined in [WEBTRANSPORT-H2], which itself is constructed upon the CAPSULE protocol originating from [DATAGRAM].

A CAPSULE has the form in [DATAGRAM]:

```
Capsule {
  Capsule Type (i),
  Capsule Length (i),
  Capsule Value (..),
}
```

where Capsule Type and Length are variable-length integers. The Capsule Value represents the payload of the capsule, and its semantics are determined by the payload type

In the context of WebTransport over WebSockets, CAPSULEs are substituted by binary DATA FRAMES of WebSockets, following the format:

```
WebSocketDataFrameCapsule {
  FrameHeader (..),
  PayloadData (..)
}
```

FrameHeader contains the first two bytes of the FRAME, and if present the extended payload length and masking key as defined in Section 5.2 of [WEBSOCKET]. PayloadData is defined as:

```
PayloadData {
  Capsule Type (i),
  Capsule Value (..)
}
```

with the variable length integer Capsule Type and Capsule Value as in the CAPSULE protocol.

Capsule length can be calculated from the Payload Length as set in Section 5.2 of [WEBSOCKET]:

```
  Capsule Length = Payload Length - sizeof(Capsule Type),
```

as no Extension Data is allowed.

3.4. Replacement for SETTINGS

Section 3.2 of [WEBTRANSPORT-H2] requires sending an SETTINGS_WEBTRANSPORT_MAX_SESSIONS settings parameter. This is not required here, as the protocol type is negotiated using the subprotocol mechanism of WebSockets and SETTINGS_WEBTRANSPORT_MAX_SESSIONS equal to 1 is assumed per WebSocket connection(HTTP1)/stream(HTTP2). Subsections of Section 4.3 of [WEBTRANSPORT-H2] require sending initial SETTINGS for flow control. As SETTINGS are not accessible for the WebSocket protocol using the existing WebSocket interfaces, a replacement is required.

Therefore client and server MUST send the initial flow control values using CAPSULES immediately before ANY other capsules such as WT_STREAM or DATAGRAM capsules have been sent.

4. Implementation Status

The protocol is implemented in a node.js package (https://github.com/fails-components/webtransport).

5. Security Considerations

The security considerations of Section 10 of [WEBSOCKET] also apply here. The last paragraph of Section 8 of [WEBTRANSPORT-H2] is equally applicable to this protocol.

6. IANA Considerations

6.1. WebSocket Subprotocol Name Registry

All possible subprotocol names following the format "webtransport_VERSION" and "webtransport_VERSION_SUBPROTOCOL," where VERSION is an alphanumeric string denoting the subprotocol version of this protocol and SUBPROTOCOL can be any application-level string, are added to the registry as domains for this protocol and its successors.

6.2. WebTransport WebSocket Protocol Version Registry

This specification establishes a new IANA registry for WebTransport Protocol Version names, intended for use with the WebSocket WebTransport Protocol, in alignment with the principles outlined in [RFC5226].

As part of this registry, IANA manages the following information (similar to [WEBSOCKET] versions):

Version String The version string name as part of the subprotocol defined in Section 6.1 and Section 3.1. The value must only include alphanumeric characters.

Reference The RFC requesting a new version number or a draft name with version number (see below).

Status Either "Interim" or "Standard". See below for a description.

A version string can be either "Interim" or "Standard".

A "Standard" version string is part of an RFC and identifies a major, stable version of the WebTransport-WebSocket protocol. The "IETF Review" IANA registration policy [RFC5226] applies to "Standard" version string.

An Internet-Draft documents an "Interim" version string. Internet-Drafts helps implementors to identify and interoperate with the WebTransport-WebSocket protocol, as this current draft. The "Expert Review" IANA registration policy [RFC5226] applies to the "Interim" version names. The initial Designated Experts need to be determined.

7. References

7.1. Normative References

[HTTP] Fielding, R. T., Nottingham, M., and J. Reschke, "HTTP Semantics", Work in Progress, Internet-Draft, draft-ietf-httpbis-semantics-19, 12 September 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-httpbis-semantics-19>.

[OVERVIEW] Kinnear, E. and V. Vasiliev, "The WebTransport Protocol Framework", Work in Progress, Internet-Draft, draft-ietf-webtrans-overview-12, 2 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-webtrans-overview-12>.

[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>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <https://www.rfc-editor.org/rfc/rfc5226>.

[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>.

[WEBSOCKET] Fette, I. and A. Melnikov, "The WebSocket Protocol", RFC 6455, DOI 10.17487/RFC6455, December 2011, <https://www.rfc-editor.org/rfc/rfc6455>.

[WEBSOCKET-H2] McManus, P., "Bootstrapping WebSockets with HTTP/2", RFC 8441, DOI 10.17487/RFC8441, September 2018, <https://www.rfc-editor.org/rfc/rfc8441>.

[WEBTRANSPORT-H2] Frindell, A., Kinnear, E., Pauly, T., Thomson, M., Vasiliev, V., and W. Xie, "WebTransport over HTTP/2", Work in Progress, Internet-Draft, draft-ietf-webtrans-http2-11, 16 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-webtrans-http2-11>.

[WEBTRANSPORT-H3]
Frindell, A., Kinnear, E., and V. Vasiliev, "WebTransport over HTTP/3", Work in Progress, Internet-Draft, draft-ietf-webtrans-http3-12, 3 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-webtrans-http3-12>.

7.2. Informative References

[DATAGRAM] Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable Datagram Extension to QUIC", RFC 9221, DOI 10.17487/RFC9221, March 2022, <https://www.rfc-editor.org/rfc/rfc9221>.

[HTTP3] Bishop, M., "HTTP/3", Work in Progress, Internet-Draft, draft-ietf-quic-http-34, 2 February 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-quic-http-34>.

[QUIC] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/rfc/rfc9000>.

Acknowledgments

Parts of the text were rephrased using ChatGPT. Portions of this document are based upon a modification of text parts from the underlying standards.

Author’s Address

Marten Richter
Technische Universit辰t Berlin
Email: marten.richter@tu-berlin.de