

Discovering WebSocket over HTTP/2 and HTTP/3

IETF 116 – Yokohama – 2023-03

Lucas Pardue – lucaspardue.24.7@gmail.com

Momoka Yamamoto - momoka.my6@gmail.com

Dragana Damjanovic - dragana.damjano@gmail.com

Recap

- Bootstrapping WebSockets with
 - HTTP/2 - RFC 8441
 - HTTP/3 - RFC 9220
- WebSocket per request stream, converted via **extended** CONNECT

```
:method = CONNECT
:protocol = websocket
:scheme = https
:path = /chat
:authority = server.example.com
sec-websocket-protocol = chat, superchat
sec-websocket-extensions = permessage-deflate
sec-websocket-version = 13
origin = http://www.example.com
```

We extended CONNECT semantics and flowers bloomed

SETTINGS_ENABLE_CONNECT_PROTOCOL=1

Once a client knows, it can send extended CONNECT

Sending extended CONNECT at any other time == malformed request

:protocol pseudo-header

Value is an [HTTP Upgrade Token](#)

Registered: websocket, connect-udp

WIP: webtransport, connect-ip, ... connect-tcp

The setting makes client implementation difficult

A client probably discovers a WebSocket resource with the scheme wss://

SETTINGS_ENABLE_CONNECT_PROTOCOL is a **strong** signal that extended CONNECT is supported but a **weak** signal that WebSockets are supported

A client has to make several gambles when determining what connection to pick to open a WebSocket.

- New H1.1 conn + Upgrade: websocket probably will work
- New H2 or H3 conn => wait for SETTING
 - Send a request that might fail because :protocol is not supported
- Existing H2 or H3 => will already have SETTING
 - Send extended CONNECT that might fail because :protocol is not supported

So what're the perceived problems?

From past WG mailing list discussion, these are views are not shared by all:

- Availability of a resource at an authority is not tightly linked to the HTTP version features available when connecting to the authority.
- Latency risks from guessing wrong adds friction to uptake of “X over HTTP/Y”
- Dispatching requests based on state can be opaque and a bit non-deterministic
- More extended CONNECT on the way e.g., WebTransport
 - Supporting different protocols requires `SETTINGS_ENABLE_CONNECT_PROTOCOL`
 - But setting doesn't indicate list of supported protocols
 - Server operator may have reasons to support a protocol on a subset of HTTP versions

Fix, mitigate, avoid, or ignore?

1. Better advertisement could provide **stronger hints**, reducing risks:
 - 2 proposals, see next slides
2. Better semantic HTTP feature discovery could provide **stronger hints**:
 - E.g. OPTIONS for HTTP-version-specific features?
3. Better response status or error codes could provide better **failover hints**
 - currently defined but unsuitable(?):
421, 426, HTTP_1_1_REQUIRED, H3_VERSION_FALLBACK
4. Require deployments to support “all the things” to avoid client (user) pain?
5. Live with status quo, do nothing, etc.

Current Problem with just extended CONNECT knowledge

When there is an existing H2 or H3 connection and client discovers a WebSocket resource with the scheme wss://

	client sends a WebSockets request by extended CONNECT using the existing connection	client creates new HTTP/1.1 connection and does Upgrade
server supports WebSockets over HTTP/2	WebSockets over HTTP/2 connection successfully created :) 😊 😊 😊	WebSockets creation connected (Requires unnecessary RTTs to create new HTTP/1.1 connection) :(😞 😞 😞
server does not support WebSockets over HTTP/2	WebSockets over HTTP/2 connection fails :(😞 😞 😞	WebSockets creation connected :) 😊 😊 😊

The client does not know if the server supports WebSockets over the current connection, so it cannot make the right choice.

Proposal 1: SETTINGS_ENABLE_WEBSOCKETS

Create a SETTINGS_ENABLE_WEBSOCKET parameter

[draft-momoka-httpbis-settings-enable-websockets](#)

server **supports** WebSockets over H2 or H3:
SETTINGS_ENABLE_WEBSOCKETS = 1

server **does not support** WebSockets over H2 or H3:
SETTINGS_ENABLE_WEBSOCKETS = 0

WebTransport has a SETTINGS_ENABLE_WEBTRANSPORT
(or SETTINGS_WEBTRANSPORT_MAX_SESSIONS>0) parameter.

Proposal 1: SETTINGS_ENABLE_WEBSOCKETS

The Client behavior:

	There is an existing H2 or H3 connection
settings parameter is not sent from server (current behavior)	Behavior may vary by implementation.
SETTINGS_ENABLE_WEBSOCKETS = 0	New HTTP/1.1 connection + Upgrade
SETTINGS_ENABLE_WEBSOCKETS = 1	Use current connection for WebSockets over H2 or H3

Proposal 2: Advertising WebSocket support in HTTPS RR

Discover the WebSockets support before creating a connection

Extending HTTPS RR:

- Already use for discovering alpn, etc.

[draft-damjanovic-websockets-https-rr-01](#)

Proposal 2: Advertising WebSocket support in HTTPS RR

example.net IN HTTPS 1 . alpn=h2,h3 **wss=h2,h3**

- New "wss" SvcParamKey
- the SvcParamValue: a list of alpn-ids that support the WebSocket Protocol
- The alpn-ids must be present in the "alpn" key as well

Proposal 2: Advertising WebSocket support in HTTPS RR

- The Client behavior:
 - the "wss" key is present - strong indication of support, the client can attempt WebSockets over HTTP/2 or HTTP/3
 - the "wss" key is not present - the client should use WebSockets over HTTP/1.1
 - the "wss" key - no indication of the support for WebSockets over HTTP/1.1

Conclusion

This topic has been rumbling along for ~2 years

Let's determine **if there is consensus to address it with fixes**

...and is so, let's quickly align on **what fixes**

... two complimentary proposals on the table

- [draft-momoka-httpbis-settings-enable-websockets](#)
- [draft-damjanovic-websockets-https-rr-01](#)