Bootstrapping WebSockets with HTTP/2

Patrick McManus, November 2017

The Problem

RFC 6455 uses HTTP/1 specific protocol mechanisms such as Connection, Host, and Upgrade to initiate the WebSocket Protocol

These mechanisms do not exist in RFC 7540

The Websocket Community has pushed back at maintaining a HTTP/1 stack for the sole purpose of initiating WebSockets

Lack of HTTP/2 features such as multiplexing and priority were **not** cited as top level concerns and therefore are **non-goals** of this work.

The Solution Space

This was a discussion item at the third HTTP Workshop in London last June.

Native HTTP/2 integrations have so far failed to gain traction.

My proposal was a **minimal solution to tunnel 6455 an HTTP/2 stream**. These streams are multiplexed in the usual way with other HTTP/2 streams. Emphasis is on simplicity.

The scope of this work is limited to HTTP/2's initiation of 6455 - suitable for HTTPbis. Changes to 6455 (e.g. when masking is apropos) are imo not best done in this forum.

Some Details - We Need a Tunnel.

draft-mcmanus-httpbis-h2-websockets-02.txt

Use an HTTP/2 Opt-In Extension to create an **extended CONNECT**.

CONNECT already builds tunnels, but is defined to make them in proxies across TCP to a different host and port.

This **extends CONNECT (via an opt-in) to the origin server**. It still makes a tunnel but speaking a new protocol (i.e. websocket) instead of a connection to a client specified host,

CONNECT is already a special snowflake - it is hop to hop, builds bi-directional tunnels, etc. IMO its better than a new method which would typically be end to end. GET/Upgrade is an opportunistic upgrade of version, not really a tunnel.

Example

Early feedback expressed a strong preference for keeping 6455 metadata in envelope instead of inside tunnel.

The scheme of 6455 is actually a little unclear (wss or https?) as generally unstated in HTTP/1. This draft clarifies it as https.

```
[[ From Client ]]                       [[ From Server ]]

                                        SETTINGS
                                        ENABLE_CONNECT_PROTOCOL = 1

HEADERS + END_HEADERS
:method = CONNECT
:protocol = websocket
:scheme = https
:path = /chat
:authority = server.example.com:443
sec-websocket-protocol = chat, superchat
sec-websocket-extensions = permessage-deflate
sec-websocket-version = 13
origin = http://www.example.com

                                        HEADERS + END_HEADERS
                                        :status = 200
                                        sec-websocket-protocol = chat

DATA
WebSocket Data

                                        DATA + END_STREAM
                                        WebSocket Data

DATA + END_STREAM
WebSocket Data
```

Next Steps

Document has received “a lot” of feedback for an individual draft. Thank you.

If we keep the focus narrow I believe this can break through as a solution to a long standing nagging problem. Stakeholders have expressed a willingness to implement.

Suggest adoption by working group as standards track.