WebSocket Multiplexing & Compression Extension

IETF 83 HyBi WG

Takeshi Yoshino
tyoshino at google dot com
Our Next Task
Make WebSocket More Efficient

• Connection multiplexing (mux)
  – Reduce # of TCP connections for scalability
    • Managing dozen of persistent TCP connections is burden especially for servers

• Data compression
  – Reduce bandwidth usage
    • Take in what’s done for HTTP
Mux Extension
Baseline Proposal (1)

• [draft-tamplin-hybi-google-mux-03](#)

• **Channel ID** tagging by extension data
  – Logical channel 0: Control channel

• **Flow control**
  – Have send quota for each logical channel

• **Latency**
  – Open 1 logical channel on handshake
    • 1 RTT for each logical channel
Mux Extension
Baseline Proposal (2)

• Optimization
  – Open more channels by sending diff of handshake
  – Keep an idle connection (only control channel) open for a while

• Using other extensions together
  – Extension token order = application order
    • compress, mux → compress mux-ed channels
    • mux, compress → compress physical channel
Mux Extension
Baseline Proposal (3)

• Mux commands
  – Sent as binary data frames with ID=0
  – WebSocket frame header
  – Channel ID of 0 (1 byte)
  – Multiplexing command(s)
    • Objective logical channel ID
    • Multiplex opcode
    • Additional data
Mux Extension
Baseline Proposal (4)

• List of **mux commands**
  – AddChannel request
  – AddChannel response
  – DropChannel request
    • To notify mux level errors and close logical channel abnormally
  – DropChannel response (TBA)
  – FlowControl
Frames of mux-ed connections
  – Sent as frames with their channel ID
    – WebSocket frame header
    – Non-zero Channel ID (1-4 byte variable size)
    – Application data
Mux Extension

Issues

• Control frames of mux-ed channels may confuse intermediaries
  – Convert control frames into mux commands
• Channel ID assignment by client or server?
• Nesting: allow/disallow?
• Have way to open multiple channels at once?
• Use of channel ID values as service identifier by non-browser app
Mux Extension
Things to Leave to Implementors

• Send algorithm
• Flow control algorithm
• Time to close idle connections
• Channel ID selection

• Just provide some notes to implementors
  – Fairness among logical channels
  – No starvation
Compression Extension
Baseline Proposal

• draft-tyoshino-hybi-websocket-perframe-deflate-06

• Defined general per-frame compression
  – More choices of algorithm in the future
  – Share precious RSV bit

• Deflate as default available algorithm
Compression Extension
Baseline Proposal (deflate)

• Less overhead by adopting RFC 1979

• Configurable sliding window size
  – For systems with limited memory
  – Ask the other peer to use small window

• Turn on/off compression context takeover
  – [Off] Load balancers can dispatch w/o decoding
  – [On] Utilize redundancy between messages
Compression Extension Issues (1)
Decoupling extension/algorithm

• Extensions for each algorithm
  – compress-alpha; foo=bar,
    compress-beta; foo=bar; bar=baz

• One extension with algorithm parameter
  – As an extension parameter
    • perframe-compress; mode=“
      alpha; foo=bar, beta; foo=bar; bar=baz”
  – As a new header
    • Sec-WebSocket-Compression: alpha; ...
Compression Extension Issues (2)

• Allocation of per-frame compressed bit
  – RSV1, first octet or dynamic allocation?

• For incompressible/compressed data
  – API to turn on/off compression dynamically
  – Heuristics