Using HTTP/2 as a Transport for Arbitrary Bytestreams

draft-kinnear-httpbis-http2-transport

Eric Kinnear (ekinnear@apple.com)

Tommy Pauly (tpauly@apple.com)

WEBTRANS

IETF 106, November 2019, Singapore

Motivation

Generic transport for secure, multiplexed bytestreams

- These can be unidirectional or bidirectional
- Low setup cost for new streams
- Single congestion and recovery context

Peer-to-peer communication

- Example: Remote IPC

Share underlying transport with existing infrastructure

Why HTTP/2?

HTTP/2 provides framing layer with many desired transport features

- Configuration exchange
- Multiplexed streams
- Shared congestion control and loss recovery state
- Flow control
- Stream relationships and priorities
- Traverses the internet

Some of these properties are really coming from TLS/TCP

Potential Solution

CONNECT allows tunneling to another endpoint

Extended CONNECT allows connecting to server itself

HTTP headers enable additional negotiation

Coexists with standard HTTP request/response streams

Can also enable tunneling of UDP, with additional framing

New `:protocol` Values

Extended CONNECT defines `:protocol` value for use with WebSocket

Make generic by defining common base not specific to WebSocket

Define additional `:protocol` value

- `“bytestream”`
  - Direct stream mapping for arbitrary bytestreams to remote server

Individual applications can use specific `:protocol` values for negotiation

Motivation

- Generic transport for secure, multiplexed bytestreams
  - These can be unidirectional or bidirectional
  - Low setup cost for new streams
  - Single congestion and recovery context
- Peer-to-peer communication
  - Example: Remote IPC
- Share underlying transport with existing infrastructure

Motivation

- Generic transport for secure, multiplexed bytestreams
  - These can be unidirectional or bidirectional
  - Low setup cost for new streams
  - Single congestion and recovery context
- Peer-to-peer communication
  - Example: Remote IPC, QUIC
- Share underlying transport with existing infrastructure

Why QUIC Transport?

HTTP/3 over QUIC Transport falls back to HTTP/2 over TLS/TCP

What transport abstraction does QUIC Transport alone use over TCP?

HTTP/2 provides framing layer with many desired transport features

- Configuration exchange
- Multiplexed streams
- Flow Control
- Stream relationships and priorities

TLS/TCP provides shared congestion control and loss recovery state

Solution

Extended CONNECT defines `:protocol` value for use with WebSocket

Make generic by defining common base not specific to WebSocket

Define additional `:protocol` value

- `“bytestream”`
  - Direct stream mapping for arbitrary bytestreams to remote server

Individual applications can use specific `:protocol` values for negotiation

Define new SETTING to allow bidirectional use of (Extended) CONNECT

Summary

Add new `:protocol` values to Extended CONNECT handshake

- Sharing multiple connections to server over single underlying transport
- Ability to proxy UDP traffic more effectively to (and through) the server
- Built in security with low setup cost for new streams

Add new SETTING to allow using Extended CONNECT in both directions

- Enables the benefits above for peer-to-peer communications
- Provides fallback mechanism for QUIC Transport over HTTP/2 framing

Underlying Concepts

- Multiplex multiple protocols over a single transport connection
  - Method for negotiating use of stream for different protocol
  - Built in security with minimal setup cost for new streams
- Bidirectional establishment of streams
  - Must traverse intermediaries in both directions
- Can be extended to support unreliable delivery and datagrams

Questions?