WebSockets

HTTP[S] Review

- HTTP[S]
  - Establish a TCP connection
  - Client sends HTTP[S] request
  - Server sends HTTP[S] response
  - Close the TCP connection
- HTTP[S] requires a new TCP connection for every request/response

WebSocket Overview

- WebSocket
  - Establish a TCP connection
  - Client sends HTTP[S] request to upgrade to the WebSocket protocol
  - Server responds confirming the upgrade request
  - Client and server keep the TCP connection open
  - Client and server send WebSocket messages/frames over the TCP connection until one side closes the connection

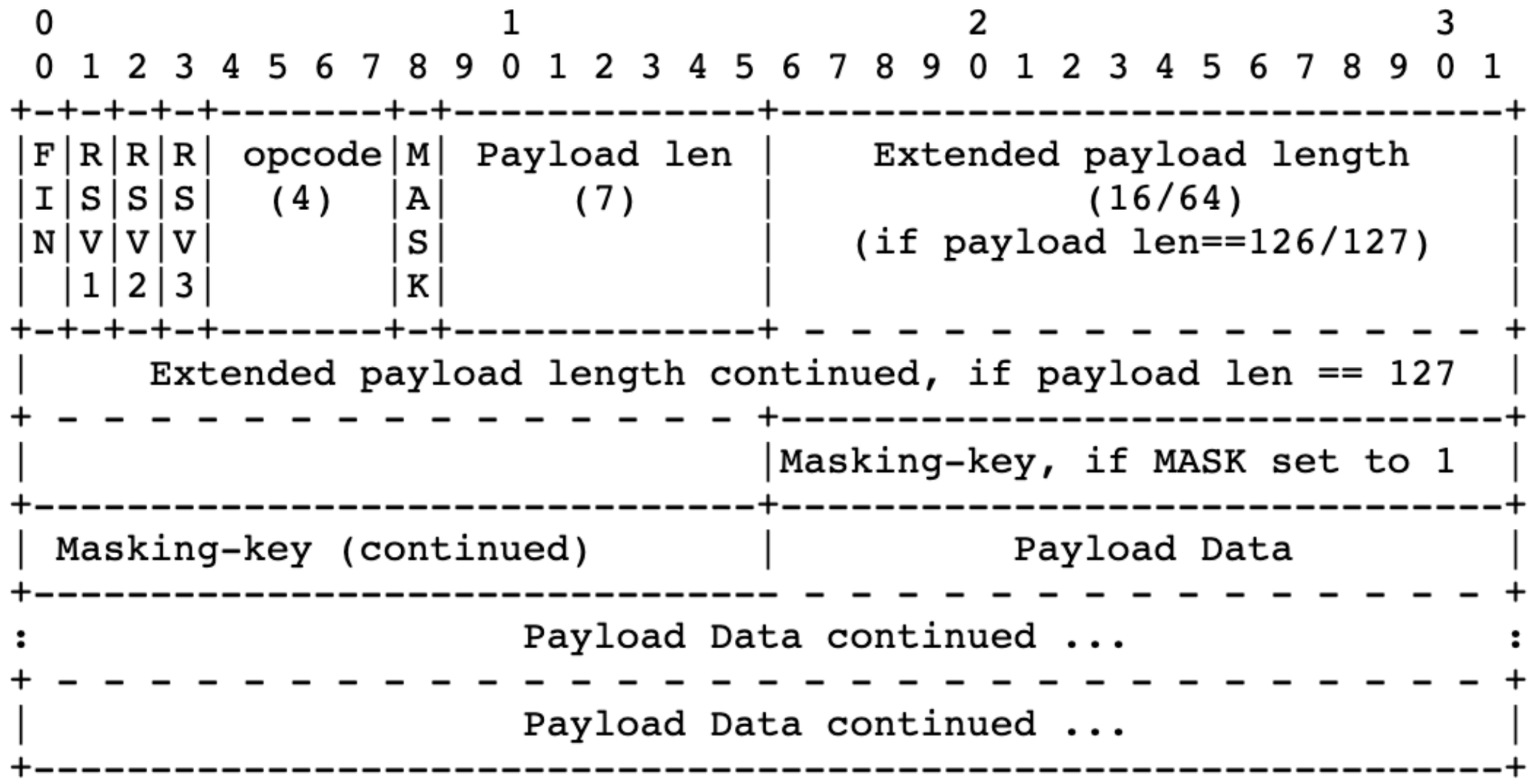
WebSocket Handshake

- Client sends an HTTP GET request to WebSocket path
- Client sets headers
  - Connection: Upgrade
  - Upgrade: websocket
  - Sec-WebSocket-Key: <random_key>
- Server responds with 101 Switching Protocols with headers
  - Connection: Upgrade
  - Upgrade: websocket
  - Sec-WebSocket-Accept: <accept_response>

WebSocket Handshake

- The client generates a random "Sec-WebSocket-Key" for each new WebSocket connection
- The server appends a specific GUID to this key
  - "258EAFA5-E914-47DA-95CA-C5AB0DC85B11"
- Computes the SHA-1 hash
- "Sec-WebSocket-Accept" is the base64 encoding of the hash
- Why?
  - Ensure client and server both implement the protocol
    - Highly unlikely this value would be returned by accident
  - Avoid caching

WebSocket Frame



https://tools.ietf.org/html/rfc6455#section-5.2

Masking

- Client-to-Server messages must be masked
- Mask is a 4 byte value that is XOR'ed with each 4 bytes of the message
- Client generates a random mask for every frame
- Why?
  - Prevent caching

Secure WebSockets

- WebSockets initiated with an HTTP request remain on port 80
  - Frames are **not** encrypted
  - The ws protocol
- If HTTPS was used for the handshake the connection remains on port 443
    - Frames are encrypted
    - The wss protocol

Socket.io

- **Adds more functionality to WebSockets**
- Reverts to long-polling if WebSockets are not supported
- Reconnects when the connection is lost
- Adds message types
- Has a middleware architecture
- Tracks all connections making it easy to broadcast to all users, or a subset of users (called a room)