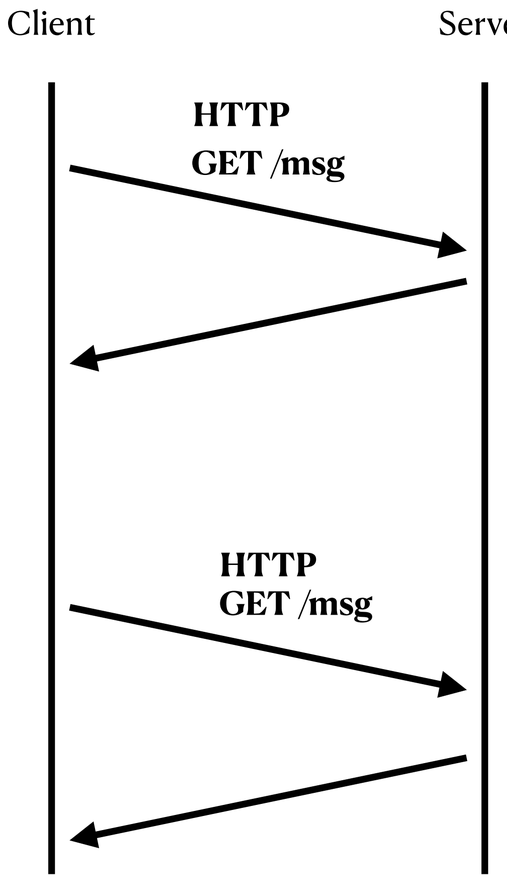
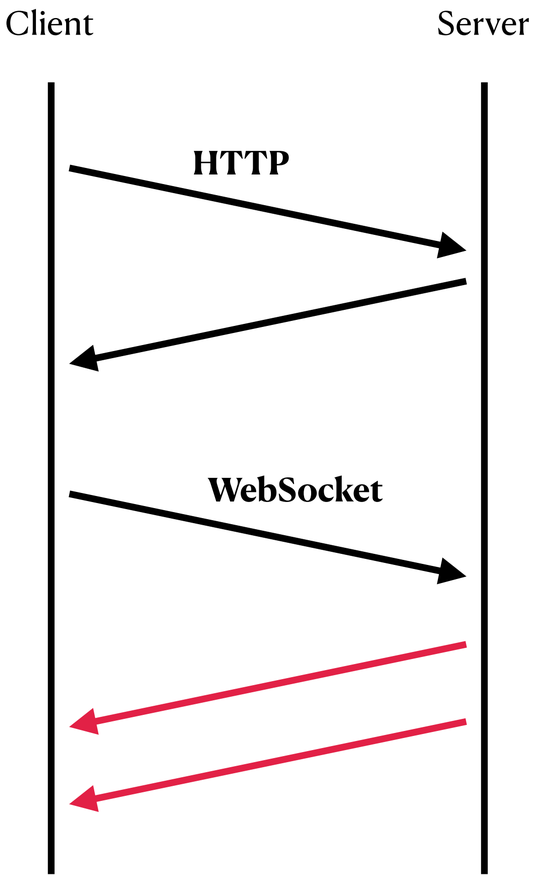
websocket 协议

如何及时的获得更新的内容

轮询到主动通知

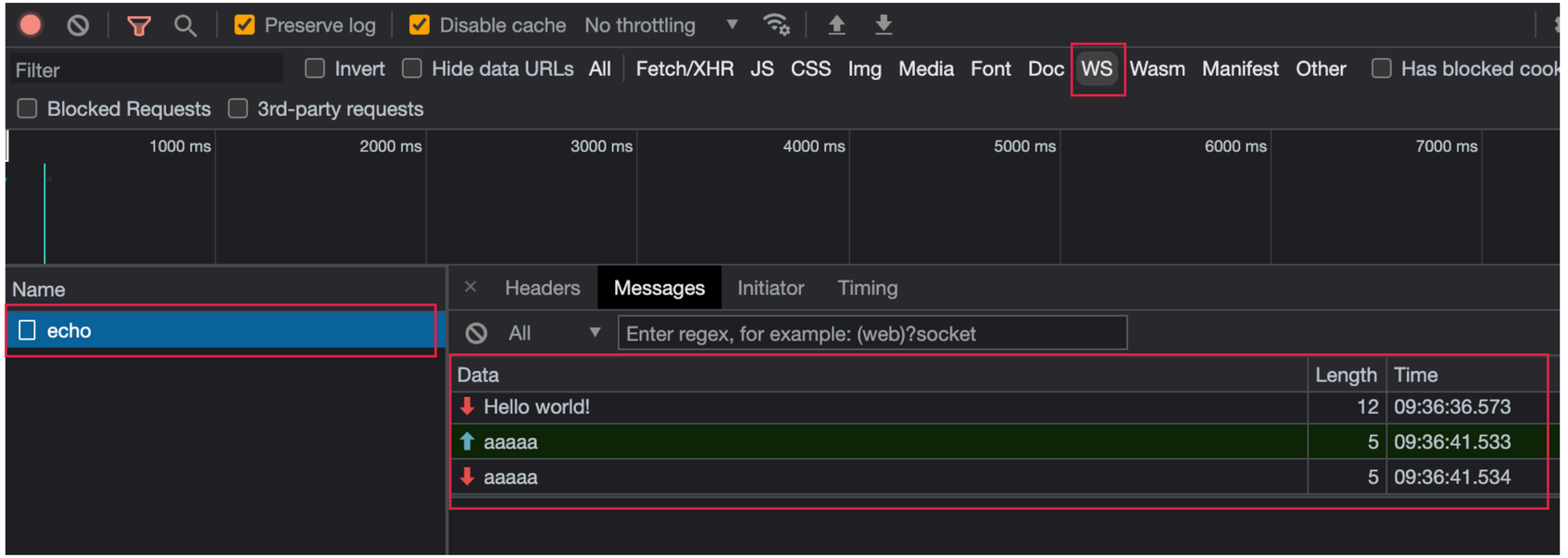
轮询间隔?





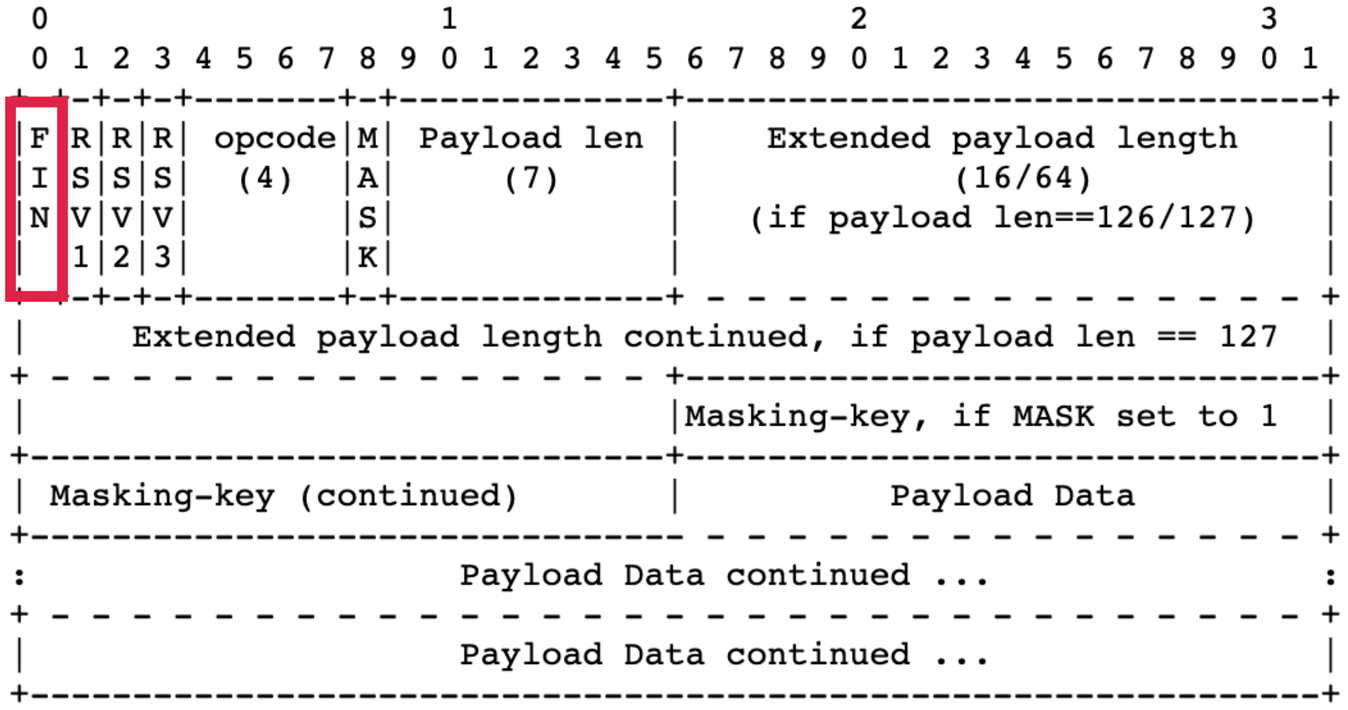
HTTP 协议的缺陷：通信只能由客户端发起

在 Chrome 中分析 WebSocket



帧类型

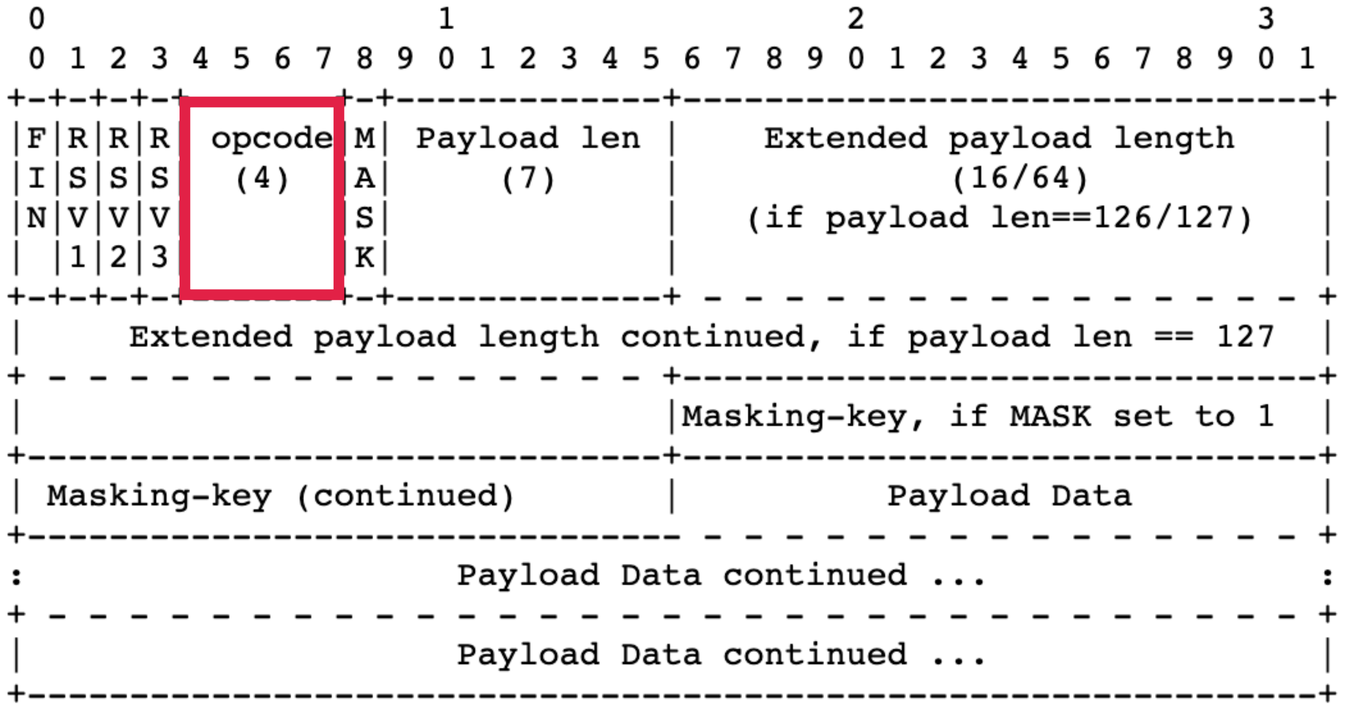
FIN 标记位



FIN 是消息结束的标志位，相当于 HTTP/2 里的“END_STREAM”，表示数据发送完毕。

一个消息可以拆成多个帧，接收方看到“FIN”后，就可以把前面的帧拼起来，组成完整的消息。

帧类型



持续帧

0：继续前一帧

非控制帧

1：文本帧

2：二进制帧

3-7：保留

控制帧

8：关闭帧

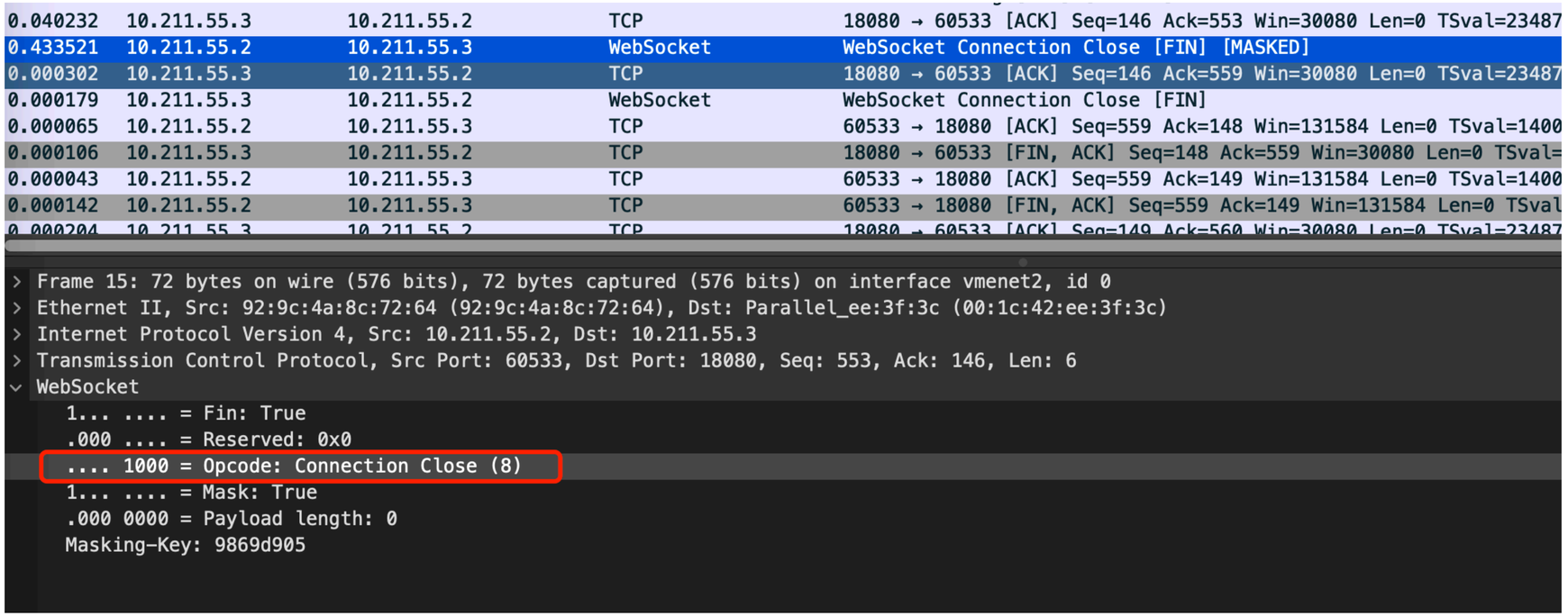
9：Ping 心跳帧

A：Pong 心跳帧

B-F：保留

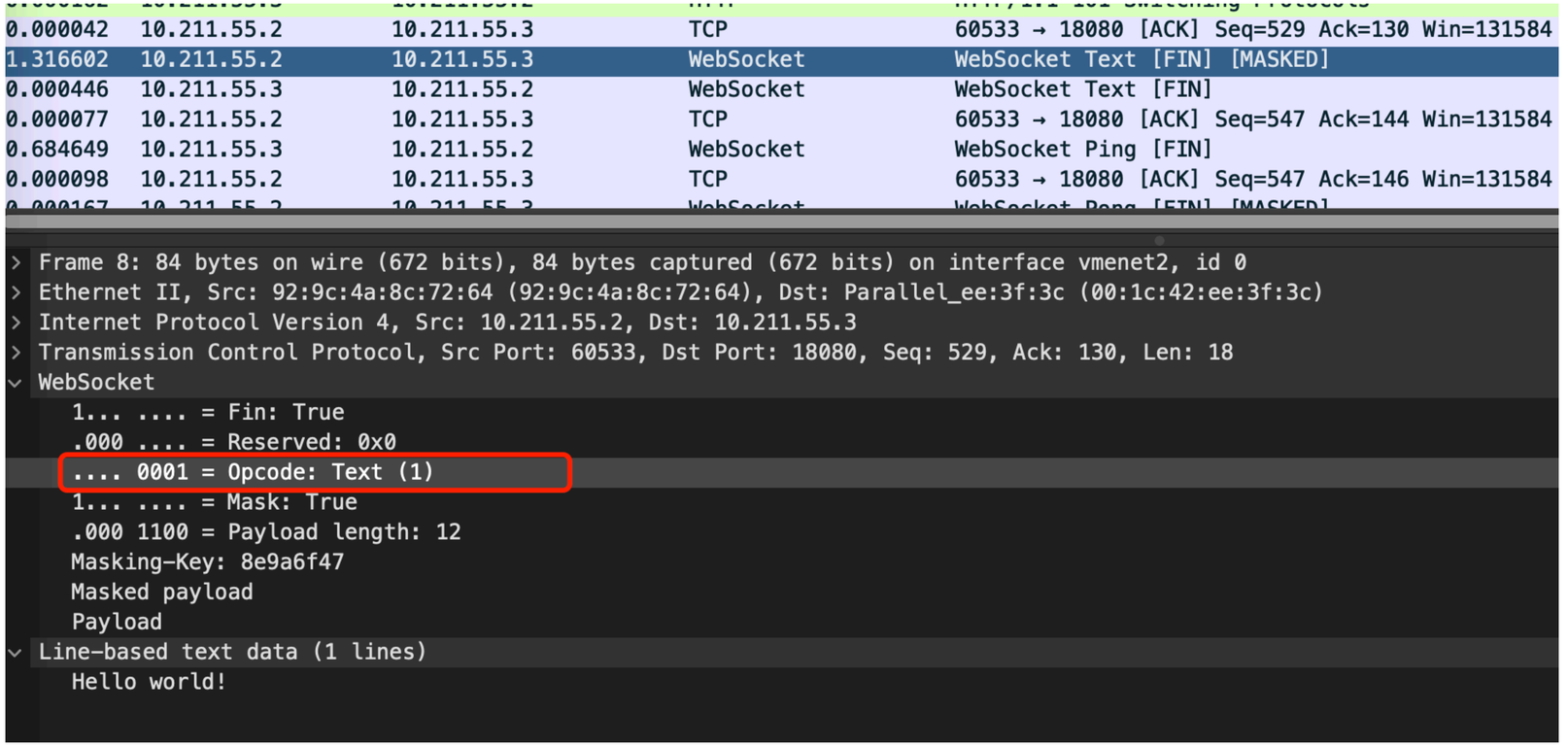
控制帧

8：关闭帧



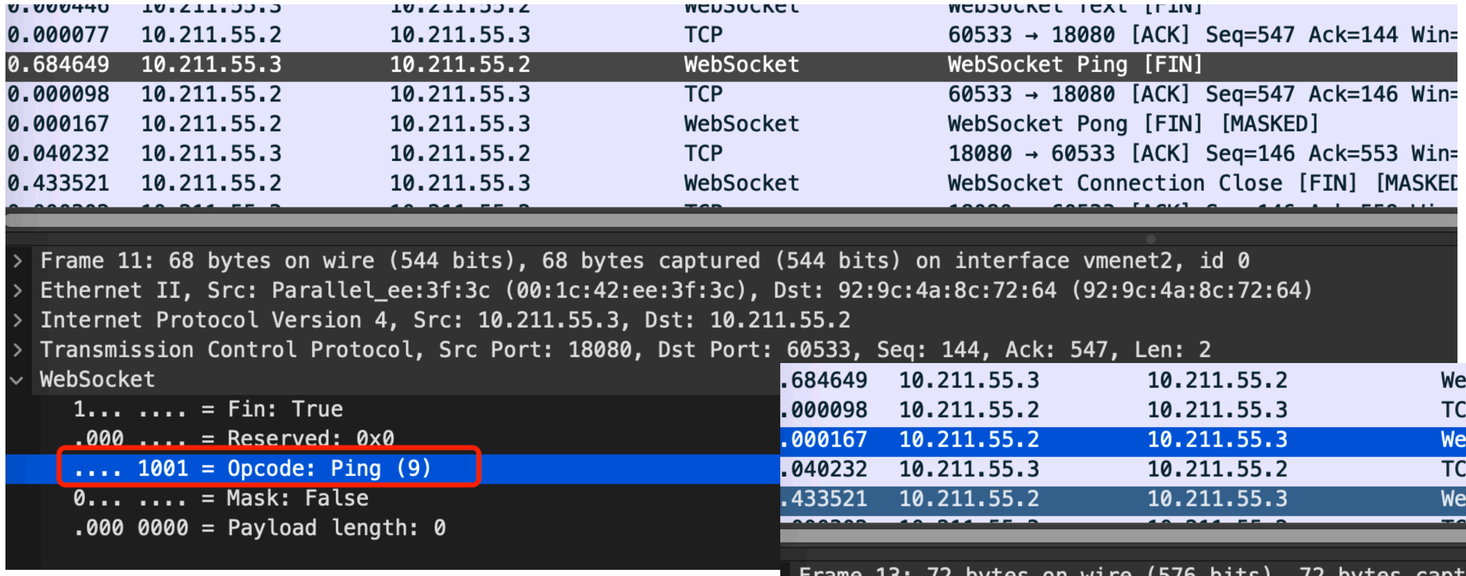
非控制帧

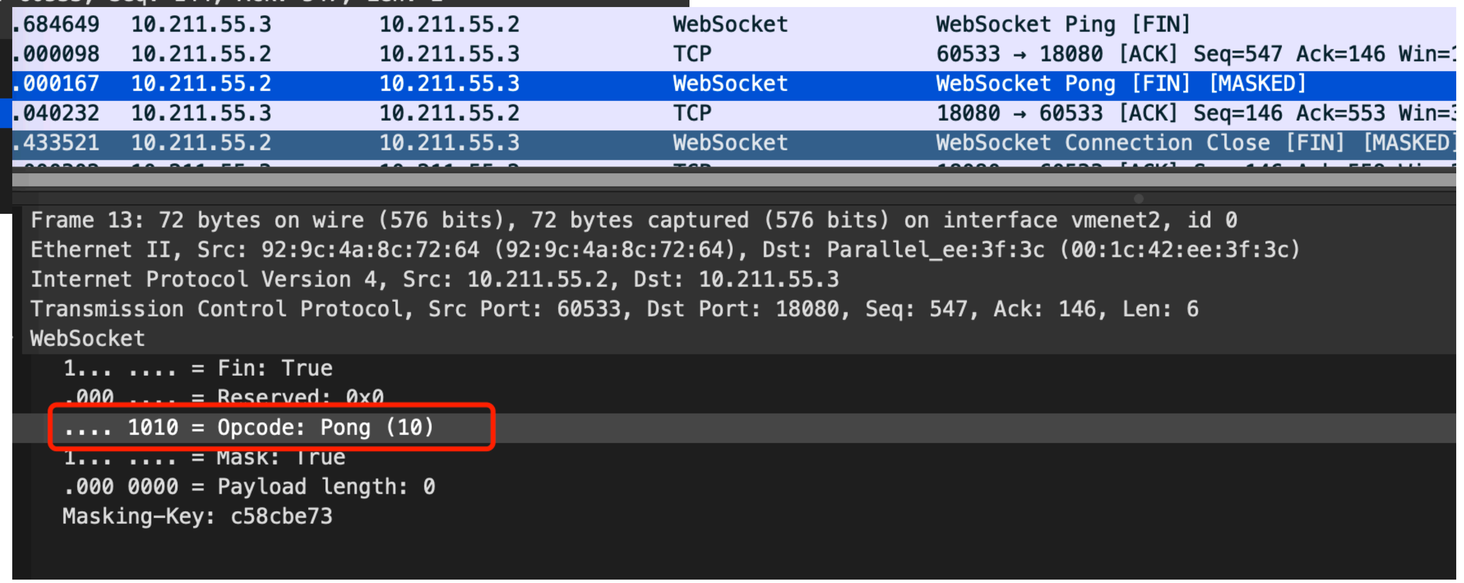
1：文本帧



控制帧

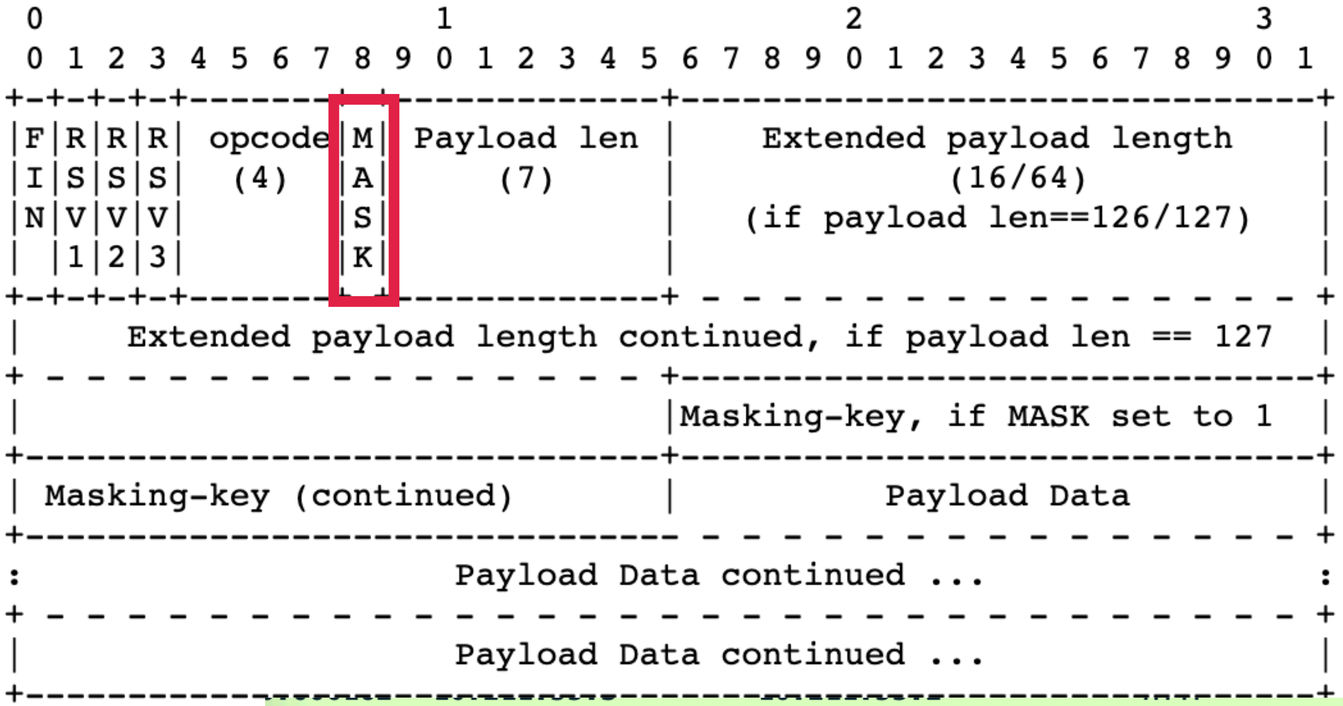
9：Ping 心跳帧、A：Pong 心跳帧





帧类型

Mask



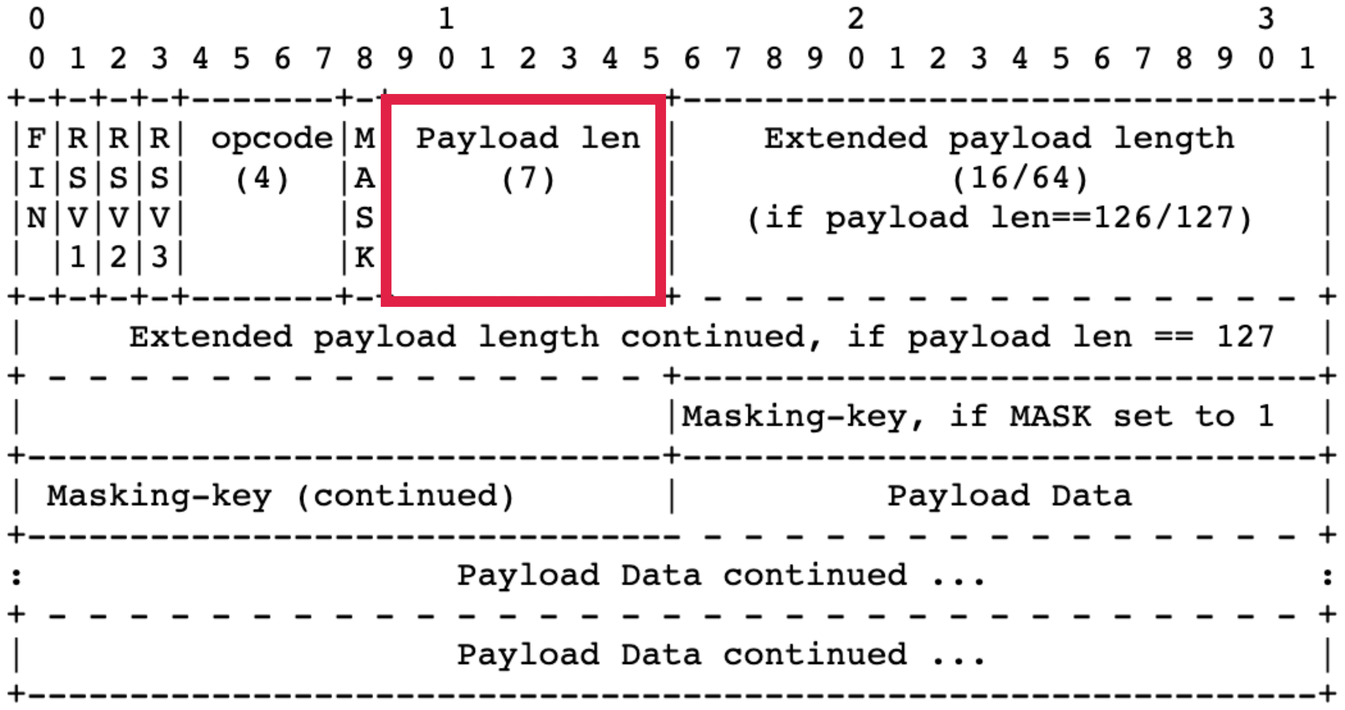
表示帧内容是否使用异或操作（xor）做简单的加密。

目前 **WebSocket** 标准规定，客户端发送数据必须使用掩码，而服务器发送则必须不使用掩码。

0.000042	10.211.55.2	10.211.55.3	TCP	60533 → 18080 [ACK] Seq=529 Ack=130 Win=131584
1.316602	10.211.55.2	10.211.55.3	WebSocket	WebSocket Text [FIN] [MASKED]
0.000446	10.211.55.3	10.211.55.2	WebSocket	WebSocket Text [FIN]
0.000077	10.211.55.2	10.211.55.3	TCP	60533 → 18080 [ACK] Seq=547 Ack=144 Win=131584
0.684649	10.211.55.3	10.211.55.2	WebSocket	WebSocket Ping [FIN]

帧类型

Payload len



“**Payload len**”，表示帧内容的长度。它是另一种变长编码

握手协议

WebSocket 的握手是一个标准的 HTTP GET 请求，但要带上两个协议升级的专用头字段

```
GET /echo HTTP/1.1
Host: 10.211.55.3:18080
Connection: Upgrade
Pragma: no-cache
Cache-Control: no-cache
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
like Gecko) Chrome/100.0.4896.88 Safari/537.36
Upgrade: websocket
Origin: http://10.211.55.3:18080
Sec-WebSocket-Version: 13
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7
Sec-WebSocket-Key: MKo/wnVQGDdDZWWhQQCPNA==
Sec-WebSocket-Extensions: permessage-deflate; client_max_wir

HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept: 4GO20hBX5OOpqC7QyVkXdHqYvdw=

....oG...+...(...f..Hello world!.......s...i....
```

- Connection: Upgrade：表示要升级协议
- Upgrade: websocket：表示要升级到websocket协议。
- Sec-WebSocket-Version: 13：表示websocket的版本。
- Sec-WebSocket-Key：与后面服务端响应首部的Sec-WebSocket-Accept是配套的，提供基本的防护，比如恶意的连接，或者无意的连接。

Sec-WebSocket-Accept计算

BASE64(SHA1(Sec-WebSocket-KeyGUID))

1.将Sec-WebSocket-Key跟258EAFA5-E914-47DA-95CA-C5AB0DC85B11拼接。

2.通过SHA1计算出摘要，并转成base64字符串。

```
public static String calc(String key) {
    String toSha = key + "258EAFA5-E914-47DA-95CA-C5AB0DC85B11";
    String sha1Hex = DigestUtils.sha1Hex(toSha);
    try {
        return Base64.encodeBase64String(Hex.decodeHex(sha1Hex));
    } catch (DecoderException e) {
    }
    return null;
}
```

小结

- HTTP 的“请求 - 应答”模式不适合开发“实时通信”应用，为了解决轮询的开销出现了 WebSocket
- WebSocket 使用兼容 HTTP 的 URI 来发现服务，但定义了新的协议名“ws”和“wss”，端口号也沿用了 80 和 443；
- WebSocket 利用 HTTP 协议实现连接握手，发送 GET 请求要求“协议升级”