WebSocket数据封包解包

qq\_28710983 2018-10-18 19:30:25 4139 收藏 2

展开

websocket和TCP/IP多了一个握手协议（ws协议不用2字节长度加数据体的协议）

1、ws\_protocol类

class ws\_protocol{

public:

static bool ws\_shake\_hand(session\* s, char\* body, int len);

static bool read\_ws\_header(unsigned char\* recv\_data, int recv\_len, int\* pkg\_size, int\* out\_header\_size);

static void parser\_ws\_recv\_data(unsigned char\* raw\_data, unsigned char\* mask, int raw\_len);

static unsigned char\* package\_ws\_send\_data(const unsigned char\* raw\_data, int len, int\* ws\_data\_len);

static void free\_package\_data(unsigned char\* ws\_pkg);

};

2、ws\_shake\_hand 握手协议，客户端向服务器发送握手请求报文

GET /chat HTTP/1.1

Host: server.example.com

Upgrade: websocket

Connection: Upgrade

Sec-WebSocket-Key: dGhlIHNhbXBsZSBub25jZQ==

Origin: http://example.com

Sec-WebSocket-Protocol: chat, superchat

Sec-WebSocket-Version: 13

3、当服务器收到报文请求时，解析报文，把Sec-WebSocket-Key加上migic(s3pPLMBiTxaQ9kYGzzhZRbK+xOo=258EAFA5-E914-47DA-95CA-C5AB0DC85B11)进行sha1加密，然后再base64编码后,发送给客户端

HTTP/1.1 101 Switching Protocols

Upgrade: websocket

Connection: Upgrade

Sec-WebSocket-Accept: s3pPLMBiTxaQ9kYGzzhZRbK+xOo=

Sec-WebSocket-Protocol: chat

s3pPLMBiTxaQ9kYGzzhZRbK+xOo=：这段报文就是Keymigic加密后编码而成的，当客户端收到上面报文websocket握手成功即可发送数据

4、read\_ws\_header读取头信息websocket数据收发协议

Websocket协议第一个字节是0x81或0x82，第二个字节是长度head\_size = 2

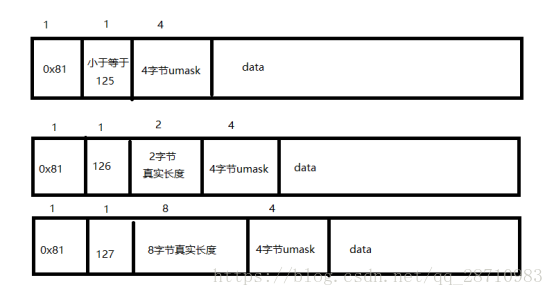
如果第二个字节小于125，表示后面就有多少个字节的数据 head\_size = 2 + 4

如果第二个字节等于126，表示后面2字节是后面数据的长度 head\_size = 2+2+4

如果第二个字节等于127，表示后面8字节是后面的长度 head\_size=2+8+4

长度后面紧跟4个字节的umask,

umask后紧跟真实数据



5、ws数据发送协议打包(发送给客户端的不需要umask即可)

unsigned char\* ws\_protocol::package\_ws\_send\_data(const unsigned char\* raw\_data, int len, int\* ws\_data\_len){

int head\_size = 2;

if (len > 125 && len < 65536) {

head\_size += 2;

}

else if (len >= 65536) {

head\_size += 8;

return NULL;

}

unsigned char\* data\_buf = (unsigned char\*)cache\_alloc(wbuf\_allocer, head\_size + len);

data\_buf[0] = 0x81;

if (len <= 125) {

data\_buf[1] = len;

}

else if (len > 125 && len < 65536) {

data\_buf[1] = 126;

data\_buf[2] = (len & 0x0000ff00) >> 8;

data\_buf[3] = (len & 0x000000ff);

}

memcpy(data\_buf + head\_size, raw\_data, len);

\*ws\_data\_len = (head\_size + len);

return data\_buf;

}

