

Yahoo Messenger Protocol v 9

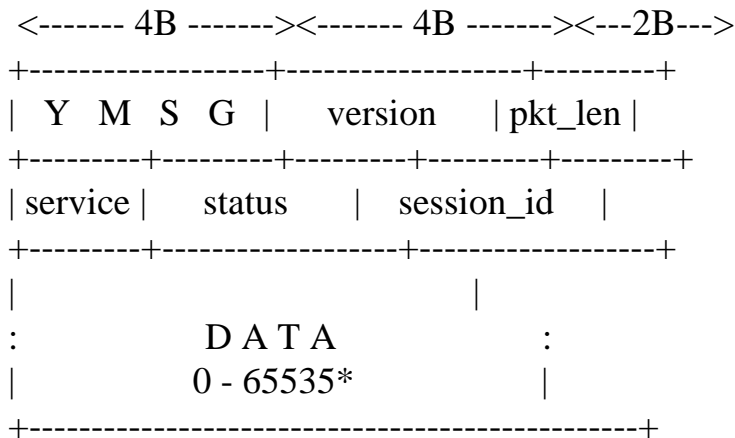
The Yahoo Messenger Protocol is an application layer protocol running most of the time over TCP, but in some cases over HTTP as well. Throughout this document, we will speak about the YMSG packets, after stripping out any other protocol data, but will mention this other data if it is of relevance.

This document is incomplete. For anything not mentioned here, refer to the source of libyahoo2.

1. The YMSG packet structure

The YMSG packet structure is as follows:

(each byte is represented by 5 spaces in the following diagram, including the | at the end)



* 65535 is the theoretical limit, since the length field is two bytes long. Practically though, the data section does not exceed about 1000 bytes.

All numeric fields are stored in network byte order. i.e. Most significant byte first.

YMSG - The first four bytes of all packets are always **YMSG** - the protocol name.

version - The next four bytes are for the protocol version number.

For version 9, these are 0x09 0x00 0x00 0x00

NOTE: The last three bytes of this may just be padding bytes.

pkt_len - A two byte value, in network byte order, stating how many bytes are in the `_data_` section of the packet. In practice, this value does not exceed about 1000.

service - This is an opcode that tells the client/server what kind of service is requested/being responded to. There are 45 known services. See the services section of this document for a full listing.

status - In case of a response from the server, indicates the status of the request (success/failure/etc.). For a request, it is 0 in most cases, except for packets that set the user's status (set status, typing notify, etc.)

session id - The session id is used primarily when connecting through a HTTP proxy. It is set in all cases, but has no effect in a direct connection. When the client sends the first packet, it is 0, the server responds with a session id that is used by the client and the server in all further packets. The server may change the session id, in which case the client must use the new session id henceforth.

DATA - The data section is `pkt_len` bytes long and consists of a series of key/value pairs. All keys are numeric strings. The packet contains their numeric values in the ASCII character set. e.g. 1 == 0x31, 21 == 0x32 0x31

The maximum number of digits in a key is unknown, although keys of up to three digits have been seen.

Every key and value is terminated by a two byte sequence of 0xc0 0x80. Some keys may have empty values.

The actual keys sent, and their meanings depend on the service in use.

e.g. The packet data to send an instant message looks like this:

0x30 0xc080 yahoo_id 0xc080 0x31 0xc080 active_id 0xc080 0x35

0xc080 recipient_id 0xc080 0x3134 0xc080 message_text 0xc080

The 0xc080 byte sequence is a separator. The values 0x30, 0x31, 0x35 and 0x3134 are the keys. Convert them to their ASCII equivalents and you get 0, 1, 5, 14 (0x3134 == 0x31 0x34)

2. Services

There are 45 known services at the moment, although more may exist. All known services are listed below along with the hex values that they correspond to. Any service without a hex value is one more than the previous value. i.e. YAHOO_SERVICE_LOGOFF=0x02 and YAHOO_SERVICE_ISBACK=0x04.

YAHOO_SERVICE_LOGON	= 0x01
YAHOO_SERVICE_LOGOFF	
YAHOO_SERVICE_ISAWAY	
YAHOO_SERVICE_ISBACK	
YAHOO_SERVICE_IDLE	= 0x05
YAHOO_SERVICE_MESSAGE	
YAHOO_SERVICE_IDACT	
YAHOO_SERVICE_IDDEACT	
YAHOO_SERVICE_MAILSTAT	
YAHOO_SERVICE_USERSTAT	= 0x0a
YAHOO_SERVICE_NEWMAIL	
YAHOO_SERVICE_CHATINVITE	
YAHOO_SERVICE_CALENDAR	
YAHOO_SERVICE_NEWPERSONALMAIL	
YAHOO_SERVICE_NEWCONTACT	= 0x0f
YAHOO_SERVICE_ADDIDENT	= 0x10
YAHOO_SERVICE_ADDIGNORE	
YAHOO_SERVICE_PING	
YAHOO_SERVICE_GROUPRENAME	
YAHOO_SERVICE_SYSMESSAGE	= 0x14
YAHOO_SERVICE_PASSTHROUGH2	= 0x16
YAHOO_SERVICE_CONFINVITE	= 0x18
YAHOO_SERVICE_CONFLOGON	
YAHOO_SERVICE_CONFDECLINE	= 0x1a
YAHOO_SERVICE_CONFLOGOFF	
YAHOO_SERVICE_CONFADDINVITE	
YAHOO_SERVICE_CONFMSG	
YAHOO_SERVICE_CHATLOGON	

YAHOO_SERVICE_CHATLOGOFF = 0x1f
 YAHOO_SERVICE_CHATMSG = 0x20
 YAHOO_SERVICE_GAMELOGON = 0x28
 YAHOO_SERVICE_GAMELOGOFF
 YAHOO_SERVICE_GAMEMSG = 0x2a
 YAHOO_SERVICE_FILETRANSFER = 0x46
 YAHOO_SERVICE_VOICECHAT = 0x4a
 YAHOO_SERVICE_NOTIFY = 0x4b
 YAHOO_SERVICE_P2PFILEXFER = 0x4d
 YAHOO_SERVICE_PEERTOPEER = 0x4f
 YAHOO_SERVICE_AUTHRESP = 0x54
 YAHOO_SERVICE_LIST = 0x55
 YAHOO_SERVICE_AUTH = 0x57
 YAHOO_SERVICE_ADDBUDDY = 0x83
 YAHOO_SERVICE_REMBUDDY = 0x84
 YAHOO_SERVICE_IGNORECONTACT = 0x85
 YAHOO_SERVICE_REJECTCONTACT = 0x86

Most of the service codes should be self explanatory. Those that aren't are listed here:

IDACT/IDDEACT - activate/deactivate an identity

NOTIFY - typing/game notification

FILETRASNFER - transfer a file using the yahoo filetransfer server as an intermediate

P2PFILEXFER - transfer a file between two peers, yahoo server not used

PEERTOPEER - check if peer to peer connections are possible

AUTH - Send initial login packet (username), response contains challenge string

AUTHRESP - Send response to challenge string, or, if received from server, contains reason for login failure

LOGON/LOGOFF - a buddy logged in/out

3. Status codes

The status code is a four byte value. Most status codes are two bytes long. The status codes (in decimal except for offline and typing) are:

YAHOO_STATUS_AVAILABLE = 0
 YAHOO_STATUS_BRB
 YAHOO_STATUS_BUSY
 YAHOO_STATUS_NOTATHOME

YAHOO_STATUS_NOTATDESK
YAHOO_STATUS_NOTINOFFICE = 5
YAHOO_STATUS_ONPHONE
YAHOO_STATUS_ONVACATION
YAHOO_STATUS_OUTTOLUNCH
YAHOO_STATUS_STEPPEDOUT = 9
YAHOO_STATUS_INVISIBLE = 12
YAHOO_STATUS_CUSTOM = 99
YAHOO_STATUS_IDLE = 999
YAHOO_STATUS_OFFLINE = 0x5a55aa56
YAHOO_STATUS_TYPING = 0x16

You may choose either AVAILABLE or INVISIBLE as your initial login status. TYPING is used only when sending a TYPING notification packet.

4. Session states

A Yahoo session has two states, Authentication and Messaging.

4.1. Authentication

The session starts in the authentication state. The client sends the username to the server. The server responds with a challenge string. The client responds to this challenge with two response strings. If authentication is successful, the connection goes into the messaging state, else, an error response is sent back.

4.2. Messaging state

After successful authentication, the session goes into the messaging state. The server sends the buddy list, ignore list, identity list and a list of cookies to the client. These might all be sent in a single packet. It then sends the list of online buddies along with their status codes. All this is sent without the client requesting anything.

At this time, any offline messages are also delivered to the client.

In the messaging state, a client may send/receive messages, join conferences, send/receive files, change state, etc.

Messaging state is terminated when the user goes offline by sending a LOGOFF packet.

5. Requests

5.1. Authentication

The first packet sent from the client is the authentication request packet. This consists of the user's yahoo id, or any identity corresponding to that yahoo id. The AUTH packet has one key/value pair.

```
service: YAHOO_SERVICE_AUTH
status: YAHOO_STATUS_AVAILABLE
```

```
1: yahoo_id
```

The server responds with a Challenge string. The client then hashes the username and password with this challenge string, and sends it back as an AUTH_RESP packet.

```
service: YAHOO_SERVICE_AUTHRESP
status: initial login status
```

```
0: yahoo_id
6: response_string_1
96: response_string_2
1: active_id
```

5.2. Sending a message

```
service: YAHOO_SERVICE_MESSAGE
status: 0
```

```
0: yahoo_id
1: active_id
5: recipient_id
14: message to send
```

5.3. Send typing start/stop notification

```
service: YAHOO_SERVICE_NOTIFY
status: YAHOO_STATUS_TYPING
```

4: active_id
5: recipient_id
13: 1 or 0 depending on whether this is a typing start or stop packet
14: <space>
49: TYPING /* The literal string */

5.4. Set status

service: YAHOO_SERVICE_ISBACK or YAHOO_SERVICE_ISAWAY
status: the status to set to

10: status_code
if custom_status:
 19: custom away message
 47: 0 or 1 depending on whether it is away or not

5.5. Logoff

service: YAHOO_SERVICE_LOGOFF
status: YAHOO_STATUS_AVAILABLE

no key value pairs

5.6. Keep alive - sent every 20 minutes

service: YAHOO_SERVICE_PING
status: YAHOO_STATUS_AVAILABLE

no key value pairs

5.7. Add buddy

service: YAHOO_SERVICE_ADDBUDDY
status: YAHOO_STATUS_AVAILABLE

1: yahoo_id
7: buddy_to_add
65: group to add to

5.8. Remove buddy

service: YAHOO_SERVICE_REMBUDDY
status: YAHOO_STATUS_AVAILABLE

1: yahoo_id
7: buddy_to_remove
65: group to remove from

5.9. Reject buddy add

service: YAHOO_SERVICE_REJECTCONTACT
status: YAHOO_STATUS_AVAILABLE

1: yahoo_id
7: buddy_to_reject
14: reject message

What? Is that all?

Use the source Luke!
