NYSE Pillar Stream Protocol Specification

NYSE Equities

NYSE MKT Equities

NYSE Arca Equities

October 28, 2016

SPEC VERSION #1.1.1

PROTOCOL VERSION 1.1

Pillar Stream Protocol

Version 1.1

Intercontinental Exchange | NYSE

October 28, 2016

1 Architecture

The Pillar platform is a messaging system. All communications are implemented using messages; and each message has a 4-byte header with type and a length (See **MsgHeader**). This is a common header for all messages.

One particular message type, the **SeqMsg** is reserved for persisted application layer messages. Each **SeqMsg** has a **SeqMsgId**, a globally unique 128-bit identifier consisting of a 64-bit "stream ID" and a sequence number. The first message on a stream has sequence number 1. A stream is an append-only file consisting of a sequence of **SeqMsg**s. Once a message is added to a stream and assigned its unique ID, this action cannot be undone.

Clients use Pillar Client Gateways to read and write streams. Once a client authenticates with the gateway, the gateway continually informs the client of availability of various streams using the **StreamAvail** message (see section Connection below).

The same stream can be read from multiple gateways simultaneously. Only one connection is allowed to write a given stream at any given time. One gateway connection supports multiple open streams.

2 Connection/Reconnection

A client connects to the gateway using TCP/IP and authenticates by sending **Login** message. Gateway responds with **LoginResponse** message. Additionally, gateway may send unsolicited **LoginResponse** with an appropriate status code (see **Status**) to indicate client logout due to violation of protocol, heartbeat timeout or if there is a new login to the same destination by the user. As long as the connection is open, client and gateway exchange heartbeats. Client sends one **Heartbeat** per second. Gateway sends one **StreamAvail** per second for each stream that's available.

To read or write a stream, client sends **Open** message, specifying a stream id, message range and delivery options. For writing, `start_seq` of message range should be the `next_seq` provided by **StreamAvail**. Gateway responds with **OpenResponse** message. Additionally gateway may send unsolicited **OpenResponse** to indicate change in access to the stream, which may happen when there is access request on the same stream from a different connection. While satisfying the read request, gateway delivers requested messages via **SeqMsg**. Client may specify a large `end_seq` (e.g. `1ULL<<63`) to subscribe to future messages.

When writing to a stream, client posts new messages with **SeqMsg**, starting with the sequence number the client specified in the **Open** request that was accepted, and incrementing it after each messages. If the client attempts to write an out-of-sequence message to a stream, the gateway will close the stream by sending an unsolicited **CloseResponse** with an appropriate error code (see **Status**).

To close a stream, the client sends **Close** message, and gateway responds with a **CloseResponse**. The gateway will automatically close a stream by sending an unsolicited **CloseResponse** once the message range specified in the **Open** message has been satisfied. If the client sends an unknown or malformed session-level message, the gateway will drop the connection.

Note: When cancel-on-disconnect is enabled, it is automatically triggered when a **TG** (trader-to-gateway) stream is closed for writing. One use case is when client closes the **TG** stream while continuing to read from the **GT** stream for cancel messages. When a connection is closed, any open streams associated with the connection are automatically closed as well.

3 Data Formats

All binary fields are *little-endian*. All alphanumeric fields are left-justified and padded on the right with ascii NULs (0 byte value).

3.1 MsgHeader

Name	***Type***	***Offset***	***Size***	***Comment***
type	u16	0	2	Message type
length	u16	2	2	Total message length, including this header

3.2 StreamId

Name	***Type***	***Offset***	***Size***	***Comment***
sess	u32	0	4	32-bit session Id
value	u32	4	4	Id of stream within session

Bit Field Name	*Source*	*Offset*	*Bits*	*Comment*
env_id	sess	24	8	Environment id. e.g. `(sess_id >> 24) & 0xff`
sess_num	sess	0	24	Session number. e.g. `sess_id & 0xffffff`
stream_type	value	24	8	Type of stream. e.g. `(id >> 24) & 0xff`
user_id	value	8	16	User id. e.g. `(id >> 8) & 0xffff`
sub_id	value	0	8	Stream sub id. e.g. `id & 0xff`

StreamType defines all the possible stream types.

3.3 SeqMsgId

Name	***Type***	***Offset***	***Size***	***Comment***
stream_id	StreamId	0	8	Target stream
seq	u64	8	8	Sequence number, starting from 1

3.4 StreamType

Name	***Type***	***Offset***	***Size***	***Comment***
value	u8	0	1	

3.4.1 Stream Type Values

Value	*Comment*
15	TG: Trader to Gateway
13	GT: Gateway to Trader
33	REF: Reference data from gateway to trader
27	XDP: Market Data (currently unavailable)

3.5 Status

Name	***Type***	***Offset***	***Size***	***Comment***
value	u8	0	1	Status code

3.5.1 Status Code Values

Value	*Comment*
0	Request processed successfully
18	Not logged in
24	Invalid login details
27	Already logged in
28	Heartbeat timeout
29	Login timed out
33	Invalid message
54	No stream permission
81	Invalid protocol version
82	Message out of sequence
84	Invalid stream
85	Stream not open
86	Invalid timestamp

4 Message Layouts

4.1 Login

Direction: client-to-gateway. Client must send Login before any other message.

Name	***Type***	***Offset***	***Size***	***Comment***
msghdr	MsgHeader	0	4	type:0x0201, length:76
username	char(16)	4	16	User name
password	char(32)	20	32	User password (plain text)
mic	char(4)	52	4	Market to login
version	char(20)	56	20	Protocol version, should be "1.1"

4.2 LoginResponse

Direction: gateway-to-client.

Name	***Type***	***Offset***	***Size***	***Comment***
msghdr	MsgHeader	0	4	type:0x0202, length:21
username	char(16)	4	16	User name
status	Status	20	1	Status of login attempt. Sucess, failure etc.

4.3 StreamAvail

Direction: gateway-to-client. Pillar gateway sends this message immediately following **LoginResponse** and once per second for each of the streams that client can interact with. The message contains stream ID and sequence of next message on stream. Once the stream is opened, no action is required by the client on receiving this message. This message provides heartbeat for the stream.

Name	***Type***	***Offset***	***Size***	***Comment***
msghdr	MsgHeader	0	4	type:0x0203, length:21
stream_id	StreamId	4	8	Target stream
next_seq	u64	12	8	Next sequence number. First message is 1.
access	u8	20	1	Available access on the stream, bit 0: Read, bit 1: Write, bit 2:Throttle Reject

4.4 Heartbeat

Direction: client-to-gateway. Message must be sent once a second (whether other data has been sent or not). If no heartbeat is received within 5 seconds, Pillar gateway will close the connection.

Name	***Type***	***Offset***	***Size***	***Comment***
msghdr	MsgHeader	0	4	type:`0x0204`, length:`4`

4.5 Open

Direction: client-to-gateway. Request open a stream for reading or writing. **Open** can be called on an already open stream to upgrade the *access* on the stream, in which case the new set of access flags will be applied.

"Lossy" mode is an optional configuration that allows the gateway to drop messages whenever the client-facing TCP buffer is full. In addition, the gateway will not attempt to retrieve any messages from disk. This results in only recently-created messages being passed through to the client. Lossy mode is only available for GT streams on drop copy gateways.

If "Throttle Reject" is set, when the input throttle is hit, instead of default behavior to queue messages until throttle is released, the New Orders are rejected with throttle reject code, Cancels are permitted and Cancel-Replaces are decomposed into Cancel and New Order and handled accordingly.

Name	***Type***	***Offset***	***Size***	***Comment***
msghdr	MsgHeader	0	4	type:`0x0205`, length:`30`
stream_id	StreamId	4	8	Target stream
start_seq	u64	12	8	Start sequence
end_seq	u64	20	8	End sequence (ignored for write request)
access	u8	28	1	Access requested, bit 0: Read, bit 1: Write, bit 2:Throttle Reject
mode	u8	29	1	Mode requested, bit 0: Lossy

4.6 OpenResponse

Direction: gateway-to-client. Response to **Open**

Name	***Type***	***Offset***	***Size***	***Comment***
msghdr	MsgHeader	0	4	type:`0x0206`, length:`14`
stream_id	StreamId	4	8	Target stream
status	Status	12	1	Response status
access	u8	13	1	Access granted

4.7 Close

Direction: client-to-gateway. Request close stream.

Name	***Type***	***Offset***	***Size***	***Comment***
msghdr	MsgHeader	0	4	type:`0x0207`, length:`12`
stream_id	StreamId	4	8	Target stream

4.8 CloseResponse

Direction: gateway-to-client. Response to **Close**

Name	***Type***	***Offset***	***Size***	***Comment***
msghdr	MsgHeader	0	4	type:`0x0208`, length:`13`
stream_id	StreamId	4	8	Target stream
status	Status	12	1	Response status

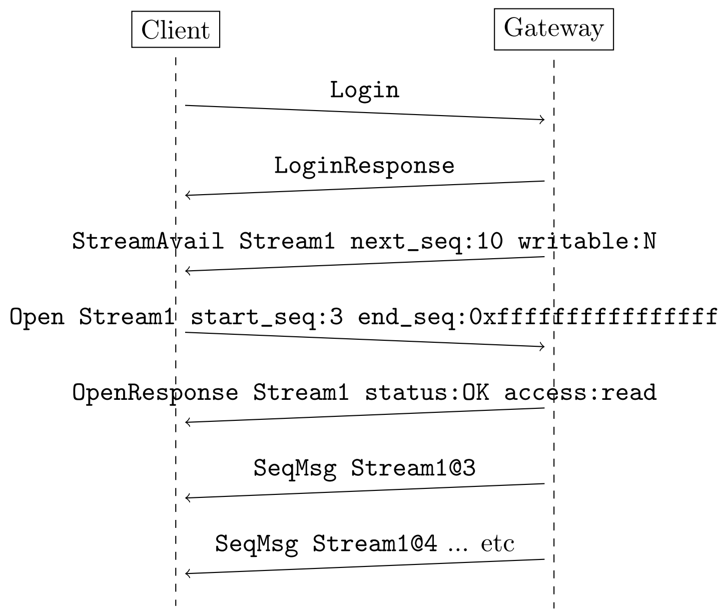
4.9 SeqMsg

Direction: both. Used to transmit a stream message.

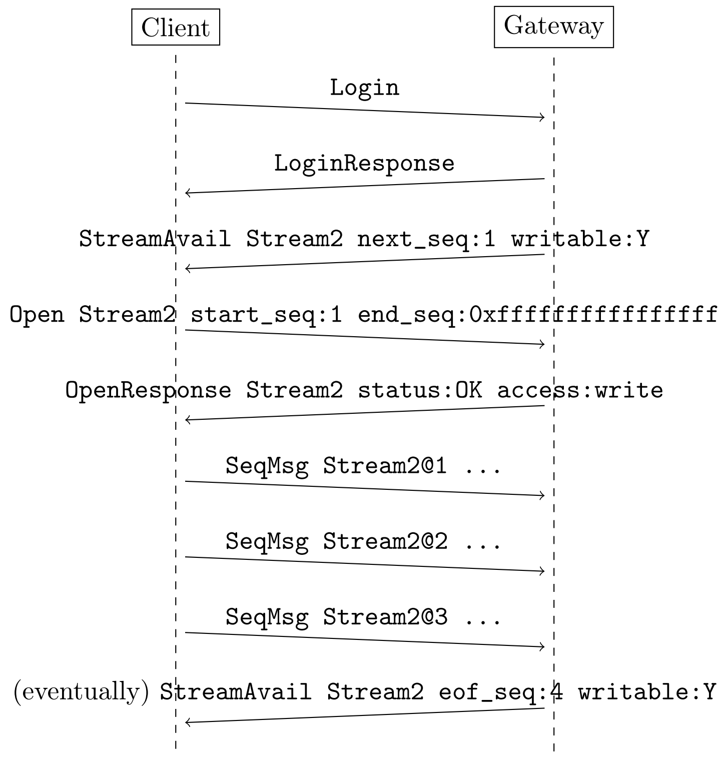
Name	***Type***	***Offset***	***Size***	***Comment***
msghdr	MsgHeader	0	4	type:`0x0905`, minimum length:`32`
seqmsg	SeqMsgId	4	16	Globally unique message id
reserved	u32	20	4	Reserved field
timestamp	u64	24	8	Message timestamp
payload	MsgHeader	32	4	Message header for the payload, present when $length - sizeof(SeqMsg) >= sizeof(MsgHeader)$

5 Examples

5.0.1 Stream Read


Client
Gateway
Login
LoginResponse
StreamAvail Stream1 next_seq:10 writable:N
Open Stream1 start_seq:3 end_seq:0xffffffffffffffff
OpenResponse Stream1 status:OK access:read
SeqMsg Stream1@3
SeqMsg Stream1@4 ... etc

5.0.2 Stream Write


Client
Gateway
Login
LoginResponse
StreamAvail Stream2 next_seq:1 writable:Y
Open Stream2 start_seq:1 end_seq:0xffffffffffffffff
OpenResponse Stream2 status:OK access:write
SeqMsg Stream2@1 ...
SeqMsg Stream2@2 ...
SeqMsg Stream2@3 ...
(eventually) StreamAvail Stream2 eof_seq:4 writable:Y

6 Document History

Date	Spec Version #	Change Summary
August 12, 2016	1.1.0	Initial version of the specification.
October 28, 2016	1.1.1	- Removed error code: Permission denied - Added error codes: Not logged in Invalid message No stream permission Invalid stream Stream not open Invalid timestamp - Added mic field to Login message - Removed mic field from LoginResponse message - username field of Login/LoginResponse message changed from 32 to 16 bytes - msghdr type for Heartbeat message changed from 0x0e01 to 0x0204 - writable field replaced by access in StreamAvail message - timestamp in SeqMsg message changed from optional to non-optional for writing

Built on October 28, 2016