MIAX Pearl Equities Exchange

Extended TCP Session Management (ESesM) Protocol Specification

Revision Date: 06/26/2020
Version 1.0.a

Table of Contents

1. Overview

In an attempt to provide a robust and uniform session management protocol across all interfaces that require guaranteed delivery of certain messages, MIAX has created the Extended TCP Session Management (ESesM) protocol. It is a lightweight point-to-point protocol that is built on top of TCP/IP protocol in order to facilitate client/server communication. ESesM extends MIAX SesM protocol; whereas SesM supports client interaction with only one Matching Engine per connection, ESesM can support client interaction with all Matching Engines on the same connection.

ESesM features:

Session level authentication: ESesM includes a simple protocol that allows the server to authenticate the client on login. This allows the right user to connect without service disruptions that could be caused due to wrong connections allowed by a protocol without authentication.

Quick link failure detection: Uses bi-directional heartbeats to quickly and proactively detect link failures.

Failure recovery: The additional benefit that it provides over raw TCP/IP protocol is that it ensures the delivery of sequenced messages in correct order across TCP/IP sessions even in the event of a reconnection after a connection failure.

Flexibility in sequencing: It provides the flexibility to the application layer to deliver messages as sequenced or unsequenced. Application layer can choose the messages that need to be sequenced to ensure quick recovery. ESesM delivers unsequenced messages on a best-effort basis only, meaning that they may be lost in the case of a TCP/IP socket failure.

Clear delineation from Application layer: ESesM is designed to be used in conjunction with higher level protocols that specify the contents of the messages that ESesM delivers. The ESesM protocol layer is opaque to the higher-level messages permitting it to encapsulate and transport an application message consisting of ASCII or binary data.

Interaction with all available Matching Engines: On one connection between a client and server, the client can interact with all available Matching Engines. Messages originated from different Matching Engines are sent to the client as separate sequenced streams, messages on each sequenced stream identify the Matching Engine they originated from. ESesM also enables clients to stay connected with the server and interact with all available Matching Engines even when one of the Matching Engines is not available. This isolates the fault domain and reduces the impact on clients due to a failed Matching Engine.

2. ESesM Protocol

2.1 Packets

ESesM client and server communicate by exchanging ESesM packets. ESesM packets do not necessarily map directly to physical packets on the underlying network socket; they may be broken apart or aggregated by the TCP/IP stack.

ESesM protocol limits the maximum payload length to 65,534 bytes. ESesM payload is exactly 1 application message encapsulated in an ESesM protocol packet. The payload may contain binary data including the line feed character.

2.2 Matching Engine Trading Session

A Matching Engine trading session is identified by a Trading Session ID and is defined by a sequenced stream of messages. The Trading Session ID of a Matching Engine starts with 1 and will typically remain the same for the trading day. Only in case of a Matching Engine recovery, a new trading session will start with a Trading Session ID incremented by 1. In each new trading session, the sequenced stream of messages starts with Sequence Number = 1. Clients interact with the currently active trading session of the Matching Engines over ESesM.

2.3 Messaging

2.3.1 Establishing a Connection

An ESesM connection begins when a client opens a TCP/IP socket to the server and sends a Login packet. If the login request is valid, the server responds with a login response packet following which, the client and the server can begin sending data packets. The connection continues until the TCP/IP socket is broken or closed.

On successful login, client can interact with currently active trading session of the Matching Engines. Typically, when initially logging into a server, the client will set the *Requested Sequence Number* field to 1 and set the *Requested Trading Session ID* of a Matching Engine to zero in the login request. The client will then inspect the login response to determine the currently active trading sessions of the Matching Engines. If the TCP/IP connection is ever broken, the client can then re-log into the server indicating the current Trading Session ID and its next expected Sequence Number from the Matching Engines. By doing this, the client is guaranteed to always receive every sequenced packet in order from the Matching Engines, despite TCP/IP connection failures. If the client sends a zero for the next expected Sequence Number from a Matching Engine, server will only send new packets generated from the Matching Engine after login and not send any old packets to the client.

2.3.2 Sending and Receiving Application Messages

Once the client has successfully established a connection with the server, the client and the server can start communicating using application messages. ESesM encapsulates each application message into an ESesM packet. Each data packet carries a single higher-level protocol message. Sequenced data packets contain an explicit sequence number and the ID of the Matching Engine that originated the message. The sequence number of the data packet increases monotonically per Matching Engine in its current trading session. Unsequenced data packets do not contain a sequence number and are not counted in the sequence numbering used in the sequenced data packets.

2.3.3 Gap Fill

ESesM clients can request a gap-fill in one of the following available methods.

- *Login Request*: ESesM client sends in the expected sequence number of the next packet from each Matching Engine, in the *Login request*. ESesM server sends to the client all the Sequenced packets with a sequence number that is greater than or equal to the expected sequence number from each Matching Engine. ESesM server will keep the connection live after this login synchronization is complete for all Matching Engines. Some application interfaces (e.g. Top of Market Retransmission) will not support this method if they have limited number of gap-fill servicing servers that have to be shared by many clients. This is the preferred method for application interfaces (e.g. MIAX Express Orders (MEO)) that have dedicated client/server connections.
- *Retransmission Request*: ESesM clients that have a need for a gap-fill with a specific start and end sequence number must use the *Retransmission request*. ESesM server sends all the packets as per the request and disconnects the client connection. This is the preferred method for retransmission of packets that the client application may have missed from a data feed of a Matching Engine. For ESesM clients that must use the Retransmission request, Login message must be sent with a Requested Sequence Number of zero (0).

Note: For retransmission interfaces such as the Top of Market Retransmission Services, ESesM clients must use the *Retransmission request* instead of the *Login request* in order to fill gaps.

2.3.4 Heartbeats

ESesM uses heartbeat packets to quickly and proactively detect link failures. The server must send a *Server heartbeat* packet anytime more than 1 second has passed since the server last sent any data. This ensures that the client will receive data on a regular basis. If the client does not receive anything (neither data nor heartbeats) for an extended period of time (typically 3 heartbeats), it can assume that the link is down and attempt to reconnect using a new TCP/IP socket.

Similarly, once logged in, the client must send a *Client heartbeat* packet anytime more than 1 second has passed since the client last sent anything. If the server does not receive anything from the client for an extended period of time (typically 3 heartbeats), it can close the existing socket and listen for a new connection.

2.3.5 Trading Session Update

On initial login, the server indicates the current trading session of each Matching Engine. In case a Matching Engine fails over to a new trading session, ESesM will send a Trading Session Update message for the failed over Matching Engine. In response to the Trading Session Update message, client should expect a Sequence Number of 1 in the next sequenced message from the Matching Engine.

If the client needs to reconnect with the server during the day, it will need to login with the new Trading Session ID for the Matching Engine or a Trading Session ID of zero to begin receiving messages for the currently available Matching Engine trading session.

2.4 Data Types

The following table describes the data types used in ESesM protocol:

Data Type	Description
Binary	Unsigned, little-endian byte-ordered, binary encoded numbers
Alphanumeric	Each byte can contain characters or numbers. Left justified and space-padded on to the right

2.5 Configuration

ESesM client/server setup will require the following configuration:

1) *Username & Computer ID*: Client will use this in the Login packet and server will only allow connections from valid clients
2) *ESesM version*: Client will use this in the Login packet and server will only allow connections from a client if the client is using the ESesM version that the server is expecting based on configuration.

3. Packet Types and Packet Structure

3.1 Application Data Packets

3.1.1 Sequenced Data Packet

The Server application can choose to use Sequenced data packets to send all messages that must be recoverable by the client in case of a connection loss. Each Sequenced data packet carries one sequenced message from the higher-level protocol. The Sequenced data packet structure incorporates a sequence number field and the ID of the Matching Engine that originated the message. The sequence number is a monotonically increasing number per Matching Engine in its current trading session. ESesM protocol also retains all Sequenced data packets at the Server end until the end of each Server session. These factors allow the server application to guarantee messages and the message sequencing across TCP/IP sessions.

Since ESesM packets are carried via TCP/IP sockets, the only way logical packets can be lost is in the event of a TCP/IP socket connection failure. In this case, the client can reconnect to the server and request the next expected sequence number per Matching Engine and pick up from where it left off.

Sequenced Packet Structure

Packet Length	Packet Type	Sequence Number	Matching Engine ID	Application Data

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet (meaning the total length of the payload, packet type, sequence number, Matching Engine ID)
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "s"** in this field indicates that this is a sequenced data packet
Sequence Number	8	Binary	Sequence number of this sequenced data packet
Matching Engine ID	1	Binary	Matching Engine that originated the packet
Application Data	Variable	Any	1 Application Message

3.1.2 Unsequenced Data Packet

Server can choose to send some data in the form of unsequenced data packets; Client can send data to the server only in the form of unsequenced data packets. Each unsequenced data packet carries one unsequenced message from the higher-lever protocol. These messages are unsequenced, and must not update the client's or server's sequence numbers.

Unsequenced data packets are delivered on a best-effort basis only. They will be delivered at most once in the same order that they are generated by the higher level protocol. However, some messages may be lost due to TCP/IP connection trouble including full buffers. If message loss occurs, no portion of the lost message will be re-transmitted and ESesM does not support the recovery of such lost messages. ESesM does not define the higher level protocol behavior when unsequenced messages are lost. These semantics are defined by the higher level protocol. The higher-level protocol must be able to handle these lost messages in the case of a TCP/IP socket connection failure.

Unsequenced Packet Structure

Packet Length	Packet Type	Application Data

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet (meaning the length of the payload plus the length of the packet type)
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "U"** in this field indicates that this is an unsequenced data packet
Application Data	Variable	Any	1 Application Message

3.1.3 Packet Length

The Packet Length in the header of a sequenced or unsequenced message may be used by recipients to determine the beginning of the next message. This allows for smoother migration when the application protocol is expanded.

- If new messages are introduced that the client does not need, client may hop over the unused message type using the packet length.
- If new fields are added to the end of a message, clients that do not need these fields do not need to be modified.

3.2 Session Management Protocol Packets

3.2.1 Login Request

The ESesM client uses this packet to allow the server to authenticate the client and to convey to the server the last sequenced packet that the client received from each Matching Engine. The ESesM client must send a Login request packet immediately upon establishing a new TCP/IP socket connection to the server. If the login request is not received within a reasonable period of time, the server will terminate the connection. Server will not accept any message on a connection until a successful client login.

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "l"** (lower case L) in this field indicates that this is a Login request packet
ESesM Version	5	Alphanumeric	1.0 (right padded with spaces)
Username	5	Alphanumeric	Username issued by MIAX during initial setup
Computer ID	8	Alphanumeric	ID issued by MIAX during initial setup
Application Protocol	8	Alphanumeric	Eg: MEO1.0 (right padded with spaces)
Number of Matching Engines	1	Binary	See notes below for the value to be specified Following group of fields must be repeated for each Matching Engine
Requested Trading Session ID	1	Binary	Specifies the Matching Engine trading session the client would like to log into, or zero to log into the currently active trading session.
Requested Sequence Number	8	Binary	Specifies client requested sequence number for the Matching Engine • next sequence number the client wants to receive upon connection, or • Zero to start receiving only new messages without any replay of old messages

Notes:

- Client must use the Username and Computer ID agreed upon at the time of setup. These fields provide simple authentication to prevent a client from inadvertently connecting to the wrong server. Both Username and Computer ID are case-insensitive and should be padded on the right with spaces.
- Server will also validate the ESesM version number sent in the login request with preconfigured version number and reject the login if the client and server are configured to use different versions of the ESesM protocol.
- The server can send a GoodBye packet and terminate a client TCP/IP socket if it does not receive a Login request packet within a reasonable period of time (typically 30 seconds).
- Number of Matching Engines must be specified as below
    - For gateway interfaces such as MIAX Express Orders (MEO), Number of Matching Engines = Count of all Matching Engines for the exchange. Contact MIAX Trading Operations at TradingOperations@miaxglobal.com or (609) 897-7302 to obtain this information.
    - For retransmission interfaces such as Top of Market Retransmission that service client requests for a specific Matching Engine, Number of Matching Engines = 1
    - If an invalid value is specified for Number of Matching Engines, the login request will be rejected
- The number of repeated Requested Trading Session/Sequence Number groups must be equal to Number of Matching Engines specified. This group must be repeated sequentially for the Matching Engines. E.g. the 1st group is for Matching Engine#1, 2nd group is for Matching Engine#2 etc.

- If the value supplied in the field *Requested Sequence Number* is higher than the current sequence number from the Matching Engine, the Server rejects the login request.
- If the value supplied in the field *Requested Sequence Number* is less than the current sequence number the server has for the client, the server will replay all packets for the client from the Matching Engine starting with the specified *Requested Sequence Number.*
- For retransmission interfaces such as the Top of Market Retransmission Services, ESesM clients must use the *Retransmission request* instead of the *Login request* in order to fill gaps.

3.2.2 Login Response

The ESesM server sends a Login response packet in response to a Login request packet from the client. This packet will always be the first packet sent by the server after a successful login request.

If the status in the Login Response from all Matching Engines is Successful, client will receive the current Trading Session ID and the Highest Sequence Number for the client from all the Matching Engines and can start sending messages to the server to interact with any of the Matching Engines.

In case of a *Recoverable* error status from a Matching Engine and a Successful status from all other Matching Engines, the server will not close the socket connection. The client will receive the current Trading Session ID and the Highest Sequence Number for the client from all available Matching Engines. Client can choose to use this information, recover from the error and start sending messages to the server or disconnect and re-login with correct login information. In case a Matching Engine is unavailable, client can choose to stay connected, send messages to the server and interact with the available Matching Engines. Any messages sent to the server and targeted to the Matching Engine not available will be rejected. Once this Matching Engine becomes available, client will receive a Trading Session Update message. Following this, client can start interacting with this Matching Engine as well.

In case of a *Non-Recoverable* error status (e.g. Invalid Username/Computer ID) from any of the Matching Engines, the server closes the socket connection and the login request fails. A re-login request with correct login information is required for the client to be successfully connected to the server.

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "r"** in this field indicates that this is a Login response packet
Number of Matching Engines	1	Binary	Same as Number of Matching Engines specified in the Login Request Following group of fields is repeated for each Matching Engine sequentially
Status	1	Alphanumeric	" " – Successful *Recoverable Error status:*

Field Name	Length	Data Type	Notes
			"S" – Invalid trading session requested for the Matching Engine "N" – Invalid start sequence number requested for the Matching Engine "U" – No active trading session exists for the Matching Engine, Matching Engine unavailable *Non-Recoverable Error status:* "X" – Rejected: Invalid Username/Computer ID combination "I" – Incompatible Session protocol version "A" – Incompatible Application protocol version "C" – Invalid Number of Matching Engines specified in Login Request "L" – Request rejected because client already logged in
Trading Session ID	1	Binary	The current trading session ID of the corresponding Matching Engine.
Highest Sequence Number	8	Binary	The highest sequence number the server currently has for the client from the Matching Engine.

Notes:

- Clients can use the *Highest sequence number* from the Matching Engine just as an indication as to when the replay of old messages ends and client is current with the server stream. In case new messages arrive at the Server during the replay of the old messages, Server will have more messages from the Matching Engine than the *Highest sequence number*. Client will have to wait for the Synchronization Complete packet as a confirmation of end of replay for the Matching Engine.

3.2.3 Synchronization Complete

In case the client sends in a login request with a sequence number for a the Matching Engine that is less than the current sequence number the server has for the client, the server will send a Synchronization complete packet when it is done replaying all the old packets and is ready to start sending new packets from the Matching Engine. This indicates to the client that the server and hence the client is current with the stream of the specified Matching Engine.

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "c"** in this field indicates that this is a Synchronization complete packet

Field Name	Length	Data Type	Notes
Matching Engine ID	1	Binary	Matching Engine for which this Synchronization Complete message is sent

Notes:

- This packet will not be sent if the server does not have to replay any old messages.

3.2.4 Retransmission Request

The ESesM client uses this packet to request a retransmission server to retransmit a range of packets. A retransmission server is connected to a specific Matching Engine and retransmits packets for only that Matching Engine.

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "a"** in this field indicates that this is a Retransmission request packet
Start Sequence Number	8	Binary	Sequence number of the first packet to be retransmitted
End Sequence Number	8	Binary	Sequence number of the last packet to be retransmitted

Notes:

- For retransmission interfaces such as the Top of Market Retransmission Services, ESesM clients must use the *Retransmission request* instead of the *Login request* in order to fill gaps.
- At the completion of the retransmission, server will disconnect the ESesM client. If the End Sequence number is greater than the last packet that the server has, server will send up to the last packet it has at the time of receipt of the request and then the server will disconnect.
- Only Sequenced application messages can be retransmitted.
- ESesM clients must not send heartbeats during gapfill. Otherwise, upon receipt of a heartbeat, the server will reset the connection if it is done sending all the data. This may make the client not read the rest of the data on the connection.

3.2.5 Logout Request

The **client may send** a Logout request packet to request the connection be terminated. Upon receiving a Logout request packet, the server will immediately terminate the connection and close the associated TCP/IP socket.

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet

Field Name	Length	Data Type	Notes
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "X"** in this field indicates that this is a Logout request packet
Reason	1	Alphanumeric	" " – Graceful Logout (Done for now) "B" – Bad ESesM Packet "L" – Timed out waiting for Login Packet "A" – Application terminating connection
Text	Variable	Alphanumeric	Free form human readable text to provide more details beyond the reasons mentioned above.

3.2.6 GoodBye Packet

The **server may send** a GoodBye packet to inform the client a reason for termination of the connection just before it terminates the connection. Upon sending a GoodBye packet, the server will immediately terminate the connection and close the associated TCP/IP socket. The client can also immediately close the associated TCP/IP socket.

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "G"** in this field indicates that this is a GoodBye packet
Reason	1	Alphanumeric	"B" – Bad ESesM Packet "L" – Timed out waiting for Login Packet "A" – Application terminating connection
Text	Variable	Alphanumeric	Free form human readable text to provide more details beyond the reasons mentioned above.

3.2.7 Trading Session Update

The server will send a Trading Session Update packet to inform the client that a new trading session is starting on the specified Matching Engine. This packet is sent when the Matching Engine fails over to a new trading session during the day.

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "u"** in this field indicates that this is a Trading Session Update packet
Matching Engine ID	1	Binary	
Trading Session ID	1	Binary	ID of the new Matching Engine trading session

Notes:

- A new Matching Engine trading session starts with Sequence Number of 1.
- After start of a new trading session of a Matching Engine, messages from the previous session are not available for replay.

3.2.8 Heartbeat

3.2.8.1 Server Heartbeat Packet

The server should send a Server Heartbeat packet anytime more than 1 second passes where no data has been sent to the client. The client can then assume that the link is lost if it does not receive anything for an extended period of time (3 Heartbeat intervals).

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "0"** (zero) in this field indicates that this is a Server Heartbeat packet

3.2.8.2 Client Heartbeat Packet

The client should send a Client Heartbeat packet anytime more than 1 second passes where no data has been sent to the server. The server can then assume that the link is lost if it does not receive anything for an extended period of time (3 Heartbeat intervals).

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "1"** in this field indicates that this is a Client Heartbeat packet

3.2.9 Test Packet

A Test packet can be sent by either side of an ESesM connection at any time. Test packets are intended to provide human readable text that may aid in debugging problems. Test packets should be ignored by both client and server application software.

Field Description (length is in bytes):

Field Name	Length	Data Type	Notes
Packet Length	2	Binary	The length of rest of the packet
Packet Type	1	Alphanumeric	ESesM protocol packet type. A **value of "T"** in this field indicates that this is a Test packet
Text	Variable	Alphanumeric	Free form human readable text

Appendix A: Contact List

Please visit MIAX website for obtaining most up-to-date contact list and other such information.

Appendix B: Revision History

Revision Date	Version	Description
Jan 10, 2020	1.0	First official release
Jun 26, 2020	1.0.a	Included the Matching Engine ID in the Sequenced Packet Structure diagram

miaxglobal.com