IEX OPTIONS MARKET DATA TRANSPORT PROTOCOL SPECIFICATION

Version 1.05

Updated: August 14th, 2026

OVERVIEW

IEX Options LLC ("IEX Options") is a wholly owned subsidiary of IEX Group, Inc. that operates an options exchange trading system (the "System"). The System is designed to offer an electronic limit order book for transacting listed options.

The **IEX Options Market Data Transport Protocol** is a session-layer protocol based on UDP, designed for transmitting business messages. It does not guarantee message delivery. The IEX Options Market Data Transport Protocol operates as a broadcast-only protocol, meaning consumers do not send messages back to the source. Any missed data must be recovered through out-of-band mechanisms, such as the gap fill or snapshot service. The specific recovery method depends on the service and is detailed in its respective specification.

All IEX Options Market Data Transport Protocol fields are transmitted in little-endian byte order.

MESSAGE TYPES

Common Field Types

Channel ID

An identifier for a given stream of packets. Packets received from A/B side of the same Channel ID are guaranteed to be identical.

Sequence Number

Sequence number is a counter representing the sequence number of the first message in the packet. If there is more than one message in a packet, all subsequent messages are implicitly numbered sequentially. Sequence numbers are unique for a given multicast group/port pair.

Heartbeat Packet

Heartbeat packets are sent once a second to inform clients that a data session with the provided Channel ID is active and that connectivity has not been interrupted.

Heartbeat packets do not increase the sequence number. The sequence number in the **Heartbeat packet** is equal to the highest published **Sequenced Packet** sequence number for the provided Channel ID. **Heartbeat packets** sent before any **Sequenced packets** are published will contain a sequence number of 0.

Field	Offset	Length	Type	Notes
SBE header	0	8		*SchemaId: 10000, TemplateId: 300, BlockLength: 12, Version: 0*
channelId	8	4	uint32	Channel identifier
sequenceNumber	12	8	uint64	Highest sequence number for this channel id

Sequenced Packet

Each **Sequenced Packet** contains one or more messages for the client to process.

The sequence number provided in the Packet Header is the sequence number of the first message contained within that packet. Each subsequent message in the list increments this value by one. As a

result, after processing all messages within a given **Sequenced Packet**, consumers should expect the first message in the next **Sequenced Packet** to have a sequence number equal to the sequence number field of the preceding Packet Header, plus the value of the preceding packet's Message Count field.

Field	Offset	Length	Type	Notes
SBE header	0	8		*SchemaId: 10000, TemplateId: 301, BlockLength: 20, Version: 0*
channelId	8	4	uint32	Channel identifier
sequenceNumber	12	8	uint64	Sequence number of the first message
sequenceTime	20	8	uint64	Time at which events included in packet were sequenced
messages	28	Variable	Message Group List Encoding	Variable data

Message Group List Encoding

Length	Type	Name	Description
1	uint8	blockLength	Always 0
1	uint8	numInGroup	Number of messages in group

VarData Encoding

The messages field contains exactly numInGroup messages, each is individually encoded as:

Length	Type	Name	Description
2	uint16	length	Length in bytes of the following payload
Variable	Bytes	varData	Payload of the message

Session Shutdown

The **Session Shutdown packet** replaces the **Heartbeat packet** when no further **Sequenced Packets** will be sent for a given Channel ID. To ensure all clients receive this notification, it is transmitted repeatedly.

The sequence number in the **Session Shutdown packet** matches the highest sequence number of any previously published **Sequenced Packet** for the specified Channel ID.

Field	Offset	Length	Type	Notes
SBE header	0	8		*SchemaId: 10000, TemplateId: 302, BlockLength: 12, Version: 0*
channelId	8	4	uint32	Channel identifier
sequenceNumber	12	8	uint64	Highest sequence number for this channel id

Sequence Number Reset

The **Sequenced Number Reset** message is published at start of day or in the event of an intraday system restart. This message will always contain a sequenceNumber value of 1 to denote the reset.

Field	Offset	Length	Type	Notes
SBE header	0	8		*SchemaId: 10000, TemplateId: 303, BlockLength: 20, Version: 0*
channelId	8	4	uint32	Channel identifier
sequenceNumber	12	8	uint64	Always 1
sequenceTime	20	8	uint64	Time at which the Sequence Number Reset message was sequenced.

Handling Data Loss

When data is transmitted via the IEX Options Market Data Transport Protocol, consumers may occasionally miss messages due to the inherent unreliability of UDP. To address this, two separate Multicast groups are provided for standard A/B arbitration and to facilitate rapid recovery of messages lost on a single channel. However, in situations such as client-side system failures or when a message is lost on both channels, clients may need to request a retransmission from IEX Options. For these cases, two recovery methods are available:

- **Retransmission Service:** This mechanism recovers a finite number of messages, specified by the client via a beginning and ending Sequence Number, and is typically used during brief episodes of packet loss that affect the same messages on both A/B sides.
- **Snapshot Service:** This method sends the current business state of the feed as of the real-time sequence number that allows the client to resume processing real-time channels.

RETRANSMISSION SERVICE

Purpose & Overview

The Retransmission Service allows consumers to recover a specific range of missed real-time messages when real-time multicast recovery mechanisms (e.g., A/B arbitration) are insufficient. Retransmission is intended for targeted recovery of finite message ranges and is not a substitute for snapshot-based state recovery.

Retransmission is an explicit, on-demand service initiated by the consumer and is subject to acceptance by IEX Options.

Retransmission Architecture

The Retransmission Service uses a split control/data model:

- Control Plane:
  Retransmission requests are submitted by the consumer over a reliable TCP connection to a designated retransmission request endpoint.
- Data Plane:
  Approved retransmission data is delivered asynchronously over a dedicated retransmission multicast line that is separate from:
  - The real-time feed
  - The snapshot feed

Retransmitted messages are encoded using the same transport and business message formats and sequence numbers as the real-time feed.

Server Heartbeat

Upon establishing a TCP connection with the control plane, a **Server Heartbeat message** is sent to the client every minute. The client must respond within 5 seconds (see **Client Heartbeat**).

Field	Offset	Length	Type	Notes
Packet length	0	2	uint16	Length is 8
SBE header	2	8		*SchemaId: 10000, TemplateId: 400, BlockLength: 0, Version: 0*

Client Heartbeat

A **Client Heartbeat message** must be sent by the client whenever a **Server Heartbeat message** is received via the established TCP connection from the control plane. The response must be sent back within 5 seconds, or the control plane will force disconnect.

In the event where the message is incorrect (eg: incorrect LogonId or malformed), the connection will be closed.

Field	Offset	Length	Type	Notes
Packet length	0	2	uint16	Length is 8
SBE header	2	8		*SchemaId: 10000, TemplateId: 401, BlockLength:0, Version: 0*

Retransmission Request

A **Retransmission Request** specifies a contiguous range of real-time messages to be recovered.

Field	Offset	Length	Type	Notes
Packet length	0	2	uint16	Length is 49
SBE header	2	8		*SchemaId: 10000, TemplateId: 402, BlockLength: 41, Version: 0*
beginSequence	10	8	uint64	Inclusive beginning sequence number
endSequence	18	8	uint64	Inclusive ending sequence number
logonId	26	16	STRING(16)	Logon identifier, right padded with NULL character
requestId	42	4	uint32	Monotonically increasing identifier to correlate the request and response
channelId	46	4	uint32	Channel identifier
feed	50	1	FeedType enum	0: DEEP 1: TOPS

Request Validation

Upon receipt of a **Retransmission Request**, IEX Options performs validation checks, which may include but are not limited to:

- Validity of the Channel ID
- Validity of the Logon ID
- Availability of the requested sequence number range
- Size of the requested range
- Current system capacity
- Compliance with rate and usage limits

Validation checks are performed at various levels, which means some may be enforced during the TCP request flow, or over multicast with a reference ID to the original request. If the request is malformed, the TCP connection will be disconnected.

IEX Options responds to the TCP request with an acknowledgement which contains the original request, a reference session ID to correlate the request to a retransmission, and a status code.

The response to the retransmission request will be delivered over the multicast A/B line, whether it is the retransmission itself or one of the validation failures that are not captured in the TCP request flow.

Retransmission Requests are evaluated independently and may be accepted or rejected by IEX Options.

Retransmission Response

Retransmission Status

The Status enumeration (uint8) conveys the result status of a request.

Value	Meaning	Notes
0	Success	Accepted by the control plane, observe for a response on multicast.
1	InvalidChannelId	Channel ID does not exist.
2	InvalidFeed	Feed is invalid
3	InvalidLogonId	Logon ID is invalid. Make sure it is well formed.

4	InvalidRequestId	Request ID must be a monotonically increasing numerical value.
5	InvalidPacketLength	The advertised packet length is invalid.
6		Reserved (not in use)
7	InvalidSeqNumRange	The sequence number range is not valid
8	DenialOfService	Too many requests were made within a timeframe. The client must throttle their requests.
9	MaxDailyRequestsExceeded	Maximum daily **Retransmission Request** limit exceeded
10	MaxSeqNumRangeExceeded	Maximum sequence number range per request exceeded
11	SeqNumTTLExpired	Sequence number range is no longer available
12	InternalError	Internal system error

Retransmission Response

Field	Offset	Length	Type	Notes
Packet length	0	2	uint16	Length is 34
SBE header	2	8		*SchemaId: 10000, TemplateId: 403, BlockLength: 26, Version: 0*
logonId	10	16	STRING(16)	Logon identifier, right padded with NULL character
requestId	26	4	uint32	Monotonically increasing identifier to correlate the request and response
channelId	30	4	uint32	Channel identifier
feed	34	1	FeedType enum	0: DEEP 1: TOPS
status	35	1	Status	Status of the **Retransmission Request**

Usage Constraints

To ensure fair access and protect system stability, retransmission usage is subject to operational limits. These limits may include, but are not limited to:

- A maximum number of **Retransmission Request** per client per trading day
- A maximum sequence number range per request

Consumers are expected to design systems that minimize reliance on gap fill through appropriate use of A/B channels, retransmission, and snapshot services.

Retransmission Data Delivery

Data Encoding

Retransmitted data is delivered over UDP using the standard **IEX Options Market Data Transport Protocol** packet format.

- Retransmitted packets use the same Message Types as real-time **Sequenced Packets**
- Original Channel IDs and sequence numbers are preserved
- **Business Message** payloads are identical to those originally published

Retransmission packets may contain one or more **Business Messages**, subject to the same encoding rules as real-time packets. Note that, while ordering is maintained, retransmission packets may group messages differently than was originally sent over live transmission.

Ordering and Completeness

Retransmitted messages are transmitted in ascending sequence number order. However, because delivery occurs over UDP:

- Packet loss is possible
- Delivery is not guaranteed
- Consumers must apply the same sequencing and gap-detection logic used for real-time processing

Retransmission Consumer Processing Rules

Consumers must integrate retransmitted messages into their real-time processing pipeline using the following rules:

Sequence Validation

- Retransmitted messages must be validated using Channel ID and sequence number.
- Duplicate messages must be safely ignored.

Gap Handling

- If gaps remain after retransmission delivery, the consumer may:
  - Issue an additional retransmission request, or
  - Fall back to snapshot recovery if state cannot be reliably reconstructed.

State Consistency

- Retransmitted messages must be applied in strict sequence number order.
- Consumers must ensure retransmitted messages do not violate ordering relative to already-processed real-time messages.

Relationship to Other Recovery Mechanisms

Retransmission is complementary to other recovery mechanisms:

- **A/B Arbitration:** First-line defense against packet loss
- **Snapshot:** Full state recovery when message-based reconstruction is impractical

If a consumer is unable to reconcile feed state after applying retransmitted messages, it must discard partial state and perform snapshot recovery.

Error Handling

Retransmission data must be discarded if any of the following occur:

- Retransmission packets are received for an unknown or invalid Channel ID
- Sequence numbers fall outside the requested range
- Packet loss prevents reconstruction of the requested message range

The Retransmission Service does not provide retransmission-of-retransmission. If recovery fails, consumers must rely on subsequent retransmission requests or use Snapshot Service.

SNAPSHOT SERVICE

Purpose and Overview

The Snapshot Service provides a complete refresh of the current business state for a given market data feed. It is intended for recovery scenarios where consumers are unable to reliably reconstruct state using retransmission alone, including late joiner, intra-day restarts, extended outages, or large-scale packet loss.

The Snapshot Service operates independently of the real-time multicast feed and is delivered over dedicated snapshot multicast lines. Snapshot packets are delivered on both A and B lines. Packets published on the snapshot feed are encoded using the same transport and business message formats as the real-time feed.

Transmission Model

Snapshot messaging is transmitted on a consistent, periodic timer. Each snapshot transmission represents the most recent complete view of the real-time feed state at the time of publication. The snapshot publication cadence is fixed by IEX Options and may be adjusted without prior notice.

Each snapshot cycle consists of Sequenced Packets containing:

- Snapshot Header message (always the 1st message of each packet)
- One or more snapshot business messages

The sequence number of the Sequenced Packet published on the snapshot feed will restart at 1 at the start of each new snapshot cycle. Similar to the real-time feed, Heartbeat packets will be published during periods of inactivity.

Snapshot Header

Field	Offset	Length	Type	Notes
SBE header	0	8		*SchemaId: 10000, TemplateId: 601, BlockLength: 20, Version: 0*
snapshotId	8	4	uint32	32-bit identifier of the current snapshot cycle
currentPacketNumber	12	4	uint32	Number of the current packet within the snapshot cycle
totalPacketCount	16	4	uint32	Total number of packets in the snapshot cycle
asOfSequenceNumber	20	8	uint64	Real-time sequence reflected for all snapshot business messages in the current packet

- All packets within a given snapshot cycle will share the same snapshot ID.
- At the beginning of a snapshot cycle, the first packet will have a sequence number of 1, and its Snapshot Header will set currentPacketNumber to 1 and totalPacketCount to the total number of packets in the cycle.

- At the end of the snapshot cycle, the currentPacketNumber will be equivalent to totalPacketCount.

All snapshot business messages represent feed state after applying all real-time messages up to and including the asOfSequenceNumber in the packet's SnapshotHeader. Its possible for each packet in the same snapshot cycle to have a different asOfSequenceNumber. For a given instrument, all snapshot business messages will have the same asOfSequenceNumber, even if they span multiple packets.

Snapshot Business Messages

Purpose

Snapshot business messages convey the complete business state of the feed at the snapshot point-in-time, including the recent state of all reference data and order book state-related messages.

For DEEP, snapshot business messages include:

- Underlying RefData (template ID 1)
- Symbol Mapping (template ID 2)
- Trading Status (template ID 4)
- Add Order – Non-Customer (template ID 100)
- Add Order – Customer (template ID 101)

For TOPS, snapshot business messages include:

- Underlying RefData (template ID 1)
- Symbol Mapping (template ID 2)
- Trading Status (template ID 4)
- Quote Update – No Customer Interest (template ID 200)
- Quote Update – Customer Interest (template ID 201)

These messages reuse existing business message encodings and semantics defined in the DEEP and TOPS specifications.

Snapshot Consumer Processing Rules

Before consuming the snapshot feed for a channel, consumers must first join that channel's real-time feed and begin queueing real-time data. After joining the snapshot feed, consumers should wait for the next snapshot cycle with an asOfSequenceNumber that is equal or greater than the first queued real-time packet.

For each packet in the snapshot cycle:

- Process the snapshot business messages.
- Record the latest asOfSequenceNumber for instrument-specific snapshot business messages received

At the start of a snapshot cycle (Sequenced Packet with sequenceNumber 1, SnapshotHeader with currentPacketNumber 1):

- Before processing the packet, clear any previously stored book state for all instruments on the channel.

At the end of a snapshot cycle (SnapshotHeader with currentPacketNumber equal to totalPacketCount):

- After processing the packet, resume processing of real-time feed for each instrument starting with the first sequence number after the asOfSequenceNumber observed for each instrument.

Error Handling

Snapshot messaging must be discarded if **Sequenced Packets** contain gaps or out-of-order sequence numbers. If a snapshot is discarded, consumers must rely on the next scheduled snapshot cycle

REVISION HISTORY

Version	Date	Change
1.00	February 5, 2026	Initial publication of document.
1.01	June 4, 2026	Addition of updated details for Retransmission Service and Snapshot Service
1.02	June 22, 2026	Removed “Packet length” field from Heartbeat Packet, Sequenced Packet, and Session Shutdown message types.
1.03	July 27, 2026	1. SBE header added to Snapshot Header description. 2. sequenceTime added to Sequenced Packet description. SBE Header “BlockLength” in Sequenced Packet description updated from 12 to 20. 3. “Sequence numbers published on the snapshot feed restart at 1 during each new snapshot cycle.” added to description of Snapshot Service. 4. For Retransmission Request, updated packet length from 51 to 49 and SBE block length from 40 to 41. For Retransmission Response, updated packet length from 36 to 34 and SBE block length from 9 to 26. 5. For Retransmission Server Heartbeat and Client Heartbeat, updated packet length from 10 to 8. 6. Clarified Retransmission Request validation behavior for malformed requests and marked retransmission status code 6 as reserved.
1.04	August 14, 2026	1. Added Sequence Number Reset message type.