IEX OPTIONS MARKET DATA TRANSPORT PROTOCOL SPECIFICATION

Version 1.00

Updated: February 5th, 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.

This document is offered for informational purposes at this time, and the contents are subject to change. As of February 5, 2026, IEX Options LLC is not yet an operating trading venue. Some of the functionality described may not have received regulatory approval.

MESSAGE TYPES

Common Field Types

Channel ID

An identifier for a given stream of messages. Messages received from multiple sources which use 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 each Channel ID.

Heartbeat Message

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

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

Field	Offset	Length	Type	Notes
Packet length	0	2	uint16	Length is 22
SBE header	2	8		*SchemaId: 10000, TemplateId: 300, BlockLength: 12, Version: 0*
ChannelId	10	4	uint32	Channel identifier
SequenceNumber	14	8	uint64	Highest sequence number for this channel id

Sequenced Message

Each **Sequenced Message** 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 Message**, consumers should expect the first message in the next **Sequenced Message** 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
Packet length	0	2	uint16	A **Sequenced Message** with no payload has a length of 22.
SBE header	2	8		*SchemaId: 10000, TemplateId: 301, BlockLength: 12, Version: 0*
ChannelId	10	4	uint32	Channel identifier
SequenceNumber	14	8	uint64	Highest sequence number for this channel id
Messages	22	Variable	Message List Group Encoding	Variable data

Message List Group Encoding

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

VarData Encoding

The Payload field contains exactly Message Count messages, each is individually encoded as:

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

Session Shutdown Message

The **Session Shutdown Message** replaces the **Heartbeat Message** when no further **Sequenced Messages** 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 Message** matches the highest sequence number of any previously published **Sequenced Message** for the specified Channel ID.

Field	Offset	Length	Type	Notes

Packet length	0	2	uint16	Length is 22
SBE header	2	8		*SchemaId: 10000, TemplateId: 302, BlockLength: 12, Version: 0*
ChannelId	10	4	uint32	Channel identifier
SequenceNumber	14	8	uint64	Highest sequence number for this channel id

Handling Data Loss

When data is transmitted via the IEX Options 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:** 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 channels.
- **Snapshot:** This method sends the current business state of the feed, followed by a "live" 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 UDP multicast line that is separate from:
  - The real-time feed
  - The snapshot service

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

Heartbeat Message

Upon establishing a TCP connection with the control plane, **Heartbeat Messages** are sent to the client every minute. The client must respond within 5 seconds (see **Heartbeat Response**).

Field	Offset	Length	Type	Notes
Packet length	0	2	uint16	Length is 22
SBE header	2	8		*SchemaId: 10000, TemplateId: 400, BlockLength: 12, Version: 0*
ChannelId	10	4	uint32	Channel identifier
SequenceNumber	14	8	uint64	Highest sequence number for this channel id

Heartbeat Response

A **Heartbeat Response** must be sent by the client whenever a **Heartbeat Request** 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 LoginId or malformed), the connection will be closed.

Field	Offset	Length	Type	Notes

Packet length	0	2	uint16	Length is 30
SBE header	2	8		*SchemaId: 10000, TemplateId: 401, BlockLength:20, Version: 0*
ChannelId	10	4	uint32	Channel identifier
LoginId	14	16	STRING(16)	Login identifier, right padded with NULL character

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 50
SBE header	2	8		*SchemaId: 10000, TemplateId: 402, BlockLength: 40, Version: 0*
SessionId	10	4	uint32	Monotonically increasing session identifier to correlate the request with its response
ChannelId	14	4	uint32	Channel identifier
LoginId	18	16	STRING(16)	Login identifier, right padded with NULL character
BeginSequence	34	8	uint64	Inclusive beginning sequence number
EndSequence	42	8	uint64	Inclusive ending sequence number

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 user token
- 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.

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 (u8) 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	InvalidLoginId	Login id is invalid. Make sure it is well formed.
3	InvalidSessionId	Session id must be a monotonically increasing numerical value.
4	InvalidPacketLength	The advertised packet length is invalid.
5	InvalidMessage	The message is invalid / malformed.
6	InvalidRange	The range does not logically make sense. Further validation for a given range is done at the retransmission server.
7	DenialOfService	Too many requests were made within a timeframe. The client must throttle their requests.
8	RateLimitExceeded	Maximum daily limit **Retransmission Request** hit.
254	Unknown	Catch all if an unexpected error is encountered.

Retransmission Response

Field	Offset	Length	Type	Notes
Packet length	0	2	uint16	Length is 19
SBE header	2	8		*SchemaId: 10000, TemplateId: 403, BlockLength: 9, Version: 0*
ChannelId	10	4	uint32	Channel identifier
ReferenceID	14	4	uint32	Generated Session ID to correlate with response
Status	18	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 Messages**
- 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 data does not include a **Snapshot Start Message** or **Snapshot End Message** and must not be interpreted as a state refresh.

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
- **Gap Fill:** Efficient recovery of small, contiguous message gaps
- **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 snapshot services.

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 client restarts, extended outages, or large-scale packet loss.

The Snapshot Service operates independently of the real-time multicast channels and is delivered over dedicated UDP multicast lines. Snapshots are delivered on both A and B lines.

Transmission Model

Snapshots are transmitted on a consistent, periodic timer. Each snapshot transmission represents the most recent complete view of the 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 transmission consists of:

- A **Snapshot Start Message**
- Zero or more **Snapshot Business Messages**
- A **Snapshot End Message**

The **Snapshot Start Message** and **Snapshot End Message** delimit a single snapshot instance and define its sequencing and reconciliation semantics.

Snapshot Control Messages

Snapshot Control Messages are transmitted using the **IEX Options Market Data Transport Protocol** packet header and are distinguished by a Message Type value reserved for snapshot control.

Snapshot Control Messages contain no business payloads and do not participate in real-time sequence numbering.

Snapshot Start Message

The **Snapshot Start Message** marks the beginning of a snapshot instance and establishes the point-in-time relationship between the snapshot and the real-time feed.

Snapshot Start Message Semantics

The **Snapshot Start Message** defines the As-Of Sequence Number, which identifies the highest real-time sequence number fully reflected in the snapshot state.

All snapshot business messages represent feed state after applying all real-time messages up to and including the As-Of sequence number.

Upon receipt of a **Snapshot Start Message**, consumers must:

- Treat the snapshot as a new, independent state refresh
- Discard any in-progress snapshot for the same Channel ID

- Suspend application of real-time updates for the associated Channel ID
- Initialize snapshot sequencing state for the provided Snapshot ID
- Record the As-Of Sequence Number for use when rejoining the real-time feed

Snapshot Start Message Fields

Field	Offset	Length	Type	Notes
Packet length	0	2	uint16	Length is 26
SBE header	2	8		*SchemaId: 10000, TemplateId: 601, BlockLength: 16, Version: 0*
Snapshot ID	10	4	uint32	32-bit snapshot identifier
Channel ID	14	4	uint32	Channel identifier
As-Of Sequence Number	18	8	uint64	Highest real-time sequence reflected

Name	Description
Snapshot ID	Identifier that uniquely distinguishes this snapshot instance for the given Channel ID
Channel ID	Channel identifier to which the snapshot applies
As-Of Sequence Number	Highest real-time sequence number fully reflected in the snapshot

Snapshot End Message

The **Snapshot End Message** marks the successful completion of a snapshot instance and provides the information required to resume real-time processing.

Snapshot End Message Semantics

Upon receipt of a **Snapshot End Message**, consumers must determine whether the snapshot instance is complete and valid.

A snapshot is considered complete only if:

- The Snapshot ID matches the active snapshot instance
- No gaps were detected in snapshot sequence numbers
- The Snapshot End message is received after **Snapshot Start Message**

If the snapshot is complete and valid, consumers may finalize snapshot state and resume real-time processing beginning with the next expected sequence number following the As-Of sequence number.

Snapshot End Message Fields

Field	Offset	Length	Type	Notes
Packet length	0	2	uint16	Length is 26

SBE header	2	8		*SchemaId: 10000, TemplateId: 602, BlockLength: 16, Version: 0*
Snapshot ID	10	4	uint32	32-bit snapshot identifier
Channel ID	14	4	uint32	Channel identifier
As-Of Sequence Number	18	8	uint64	Highest snapshot sequence sent

Name	Description
Snapshot ID	Identifier matching the Snapshot Start message
Channel ID	Channel identifier to which the snapshot applies
As-Of Sequence Number	Highest Snapshot sequence number transmitted for the snapshot instance

Snapshot Business Messages

Purpose

Snapshot Business Messages convey the complete business state of the feed at the snapshot point-in-time, including a full spin of the most recent state of all reference data, followed by order book state-related messages. These messages reuse existing business message encodings and semantics defined elsewhere in the specification, with additional snapshot-specific metadata to support ordering and loss detection over UDP.

Snapshot Business Message Fields

Field	Offset	Length	Type	Notes
Packet length	0	2	uint16	Length is 22, excluding the message payload (variable)
SBE header	2	8		*SchemaId: 10000, TemplateId: 603, BlockLength: 12, Version: 0*
Snapshot ID	10	4	uint32	32-bit snapshot identifier
Snapshot Sequence Number	14	8	uint64	64-bit monotonic snapshot sequence
Business Message Payload	22	Variable	Bytes	Encoded business message

Name	Description
Snapshot ID	Identifier of the snapshot instance; must match Snapshot Start and Snapshot End

Snapshot Sequence Number	Monotonic sequence number identifying message order within the snapshot
Business Message Payload	Business message encoded per the feed's business specification

Retransmission Consumer Processing Rules

Consumers must maintain snapshot state keyed by (Channel ID, Snapshot ID) and apply the following logic:

On Snapshot Start

- Discard any in-progress snapshot for the Channel ID
- Initialize expected_snapshot_sequence = 1
- Suspend processing and begin queueing
- Record the As-Of Sequence Number

On Snapshot Business Message

- If Snapshot ID does not match the active snapshot, ignore the message
- If Snapshot sequence number != expected_snapshot_sequence, mark the snapshot as invalid
- If valid, apply the business payload and increment expected_snapshot_sequence

On Snapshot End

- If Snapshot ID does not match the active snapshot, ignore the message
- Verify that:
  - No sequence gaps were detected, and
  - As-Of sequence number == expected_snapshot_sequence - 1
- If valid:
  - Finalize snapshot state
  - Resume real-time processing beginning with the next expected sequence number immediately following the As-Of sequence number
- If invalid:
  - Discard the snapshot and wait for the next snapshot cycle

Error Handling

A snapshot instance must be discarded if any of the following occur:

- **Snapshot Start Message** is received without a matching Snapshot End
- **Snapshot End Message** is received with a mismatched Snapshot ID
- **Snapshot Business Messages** contain gaps or out-of-order Snapshot sequence numbers
- **Snapshot Business Messages** are received before **Snapshot Start Message**
- **Snapshot Business Messages** are received after **Snapshot End Message**

If a snapshot is discarded, consumers must rely on the next scheduled snapshot transmission or use gap fill or retransmission as appropriate.

Sequenced Message Example

The table below is an example of a **Sequenced Message** containing two messages, starting at sequence number 42. Sequence number 42's message payload contains "Hello", and sequence number 43's payload contains "World!!!".

Offset	Length	Type	Name	Value
0	2	uint16	Packet Length	40
2	2	uint16	Block Length	30
4	2	uint16	Template ID	10000
6	2	uint16	Schema ID	301
8	2	uint16	Version	0
12	4	uint32	Channel ID	1
16	8	uint64	Sequence Number	234092384
24	1	uint8	blockLength	0
25	1	uint8	Number In Group	1
26	2	uint16	Message Length	12
Message Payload				
28	8		SBE Header	SchemaID set as specified in SBE Header definition, TemplateID=1
36	4	uint32	Message Payload	1770301800

REVISION HISTORY

Version	Date	Change
1.00	February 5, 2026	Initial publication of document.