



---

## **XDP OPTIONS CLIENT SPECIFICATION**

**NYSE ARCA OPTIONS**

**NYSE AMEX OPTIONS**

**Version**

1.0o

**Date**

November 10, 2015

© 2015 NYSE. All rights reserved. No part of this material may be copied, photocopied or duplicated in any form by any means or redistributed without the prior written consent of NYSE. All third party trademarks are owned by their respective owners and are used with permission. NYSE and its affiliates do not recommend or make any representation as to possible benefits from any securities or investments, or third-party products or services. Investors should undertake their own due diligence regarding securities and investment practices. This material may contain forward-looking statements regarding NYSE and its affiliates that are based on the current beliefs and expectations of management, are subject to significant risks and uncertainties, and which may differ from actual results. NYSE does not guarantee that its products or services will result in any savings or specific outcome. All data is as of November 10, 2015. NYSE disclaims any duty to update this information.

## PREFACE

---

### DOCUMENT HISTORY

The following table provides a description of all changes to this document.

VERSION	DATE	CHANGE DESCRIPTION
1.0g	11/14/14	Rebranded the document to ICE standard
1.0h	7/24/2015	Revisions for the 2015 launch <ul style="list-style-type: none"> <li>▪ Removed LZ4 compression</li> <li>▪ Removed Complex Imbalance and Complex EOD Summary messages</li> <li>▪ Removed Trade Correction Refresh message</li> <li>▪ Removed Beginning of Auction indicator in Series Status message</li> <li>▪ Changed levels 2 and 3 of Market Depth messages to be full prices</li> <li>▪ Changed Volume of Trade, Correction, Trade Refresh msgs to 4 bytes</li> <li>▪ Added delivery flag 3</li> <li>▪ Imbalance msg: removed AuctionTime, added Market Imbalance Side</li> <li>▪ Complex Status message: revised list of values for SecurityStatus field</li> <li>▪ Outright Summary msg: expanded total volume field to 4 bytes</li> <li>▪ Renamed SeriesSeqNum to SymbolSeqNum</li> <li>▪ Refresh msgs don't increment SymbolSeqNum, contain original timestamp</li> <li>▪ Documented request server quotas section 1.7</li> <li>▪ Clarified, streamlined explanatory text in section 1</li> <li>▪ Grouped msg structures by feed</li> </ul>
1.0i	8/5/2015	<ul style="list-style-type: none"> <li>▪ Corrected msg sizes for msg types 407, 507, 411, 425, 513</li> <li>▪ Corrected Refresh Imbalance msg (509) to match Imbalance message (413)</li> </ul>
1.0j	8/10/2015	<ul style="list-style-type: none"> <li>▪ Added Quote Condition 4 to msg types 423, 501, 511</li> <li>▪ Minor clean-ups to descriptive text</li> </ul>
1.0k	9/28/2015	<ul style="list-style-type: none"> <li>▪ Corrected prices to be signed in all cases, added explanation of rationale</li> <li>▪ Added notes that complex prices are used with the underlying price scale code.</li> </ul>
1.0L	10/28/2015	<ul style="list-style-type: none"> <li>▪ Clarified uniqueness of series and complex indexes vs. stream</li> <li>▪ Corrected the default value for the price field in the Complex RFQ message</li> <li>▪ Clarified that the Product ID field in the Sequence Reset message is unused</li> </ul>
1.0m	10/30/2015	<ul style="list-style-type: none"> <li>▪ Correction: Series indexes are unique across the market, not just per stream</li> </ul>
1.0n	11/03/2015	<ul style="list-style-type: none"> <li>▪ Added description of start of day symbol mapping file</li> </ul>
1.0o	11/10/2015	<ul style="list-style-type: none"> <li>▪ Clarified uniqueness of messages within streams, channels</li> </ul>

# CONTENTS

---

PREFACE .....	2
Document History .....	2
CONTENTS .....	3
1.    XDP OPTIONS FUNDAMENTALS .....	5
1.1 Access to Market Data .....	5
1.2 Refresh Functionality .....	6
1.3 Symbol Index Mapping.....	8
1.4 Packet, Message and Field Structure .....	9
1.5 Heartbeats .....	13
1.6 Operational Information .....	15
1.7 Request Server Per-Client Quotas.....	15
2.    TOP FEED MESSAGE FORMATS .....	16
2.1 Outright Quote Message – Msg Type 401 .....	16
2.2 Outright Trade Message – Msg 407 .....	17
2.3 Outright Trade Cancel Message – Msg Type 409 .....	18
2.4 Outright Trade Correction Message – Msg Type 411 .....	19
2.5 Outright Imbalance Message – Msg Type 413 .....	20
2.6 Outright Crossing RFQ Message – Msg 415 .....	21
2.7 Outright Summary Message – Msg 417 .....	22
2.8 Underlying Status Message – Msg 419 .....	23
2.9 Outright Series Status Message – Msg 421 .....	24
2.10 Refresh Outright Quote Message – Msg Type 501.....	25
2.11 Refresh Outright Trade Message – Msg 507.....	26
2.12 Refresh Outright Imbalance Message – Msg Type 509 .....	27
3.    DEEP FEED MESSAGE TYPES.....	28
3.1 Outright Market Depth Message, Buy – Msg Type 403 .....	28
3.2 Outright Market Depth Message, Sell – Msg Type 405.....	29
3.3 Refresh Outright Market Depth Message, Buy – Msg Type 503.....	30
3.4 Refresh Outright Market Depth Message, Sell – Msg Type 505 .....	31
4.    COMPLEX FEED MESSAGE FORMATS .....	32
4.1 Complex Quote Message – Msg Type 423.....	32
4.2 Complex Trade Message – Msg 425.....	33
4.1 Complex Crossing RFQ Message– Msg 429 .....	34
4.2 Complex Status Message – Msg 433 .....	35
4.3 Refresh Complex Quote Message – Msg Type 511 .....	36
4.5 Refresh Complex Trade Message– Msg 513 .....	37
5.    INDEX MAPPING MESSAGE FORMATS.....	38
5.1 Underlying Index Mapping Message – Msg 435.....	38
5.2 Series Index Mapping Message– Msg 437.....	40
5.3 Complex Symbol Definition Message – Msg 439 .....	42
6.    CONTROL MESSAGE FORMATS .....	44
6.1 Stream ID Message – Msg 455 .....	44
6.2 TCP Login Request Message – Msg 456 .....	44

<b>6.3</b>	<b>TCP Login Request Response Message – Msg 457</b>	<b>45</b>
<b>6.4</b>	<b>TCP Heartbeat Message - Msg 12</b>	<b>45</b>
<b>6.5</b>	<b>Test Request Message – Msg 458</b>	<b>46</b>
<b>6.6</b>	<b>Test Request Response Message – Msg 459</b>	<b>46</b>
<b>6.7</b>	<b>Underlying Index Mapping Request Message - Msg Type 441</b>	<b>47</b>
<b>6.8</b>	<b>Series Index Mapping Request Message - Msg Type 443</b>	<b>48</b>
<b>6.9</b>	<b>Complex Symbol Definition Request Message - Msg Type 445</b>	<b>49</b>
<b>6.10</b>	<b>Request Response Message - Msg Type 447</b>	<b>50</b>
<b>6.11</b>	<b>TCP Logout Request message – Msg Type 460</b>	<b>51</b>
<b>6.12</b>	<b>Sequence Number Reset - Msg Type 1</b>	<b>51</b>
<b>7.</b>	<b>SYMBOL MAPPING DOWNLOAD FILE FORMAT</b>	<b>52</b>
<b>7.1</b>	<b>FTP File Locations</b>	<b>52</b>
<b>7.2</b>	<b>FTP File Format</b>	<b>52</b>

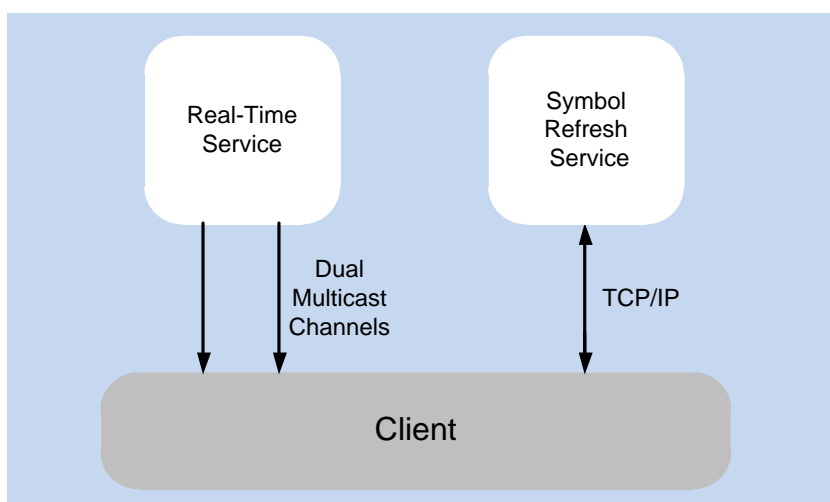
# 1. XDP Options Fundamentals

---

NOTE: Due to the performance required to publish options data, some basic XDP Options features such as channelization and error-handling are different from the XDP feeds in other NYSE markets. For this reason, this document does not make reference to the XDP Common Client Specification. All information needed to develop options client applications is contained in this one specification.

## 1.1 ACCESS TO MARKET DATA

In XDP Options, clients subscribe to multicast addresses to receive real-time market data and refresh messages as well as start-of-day referential symbol data. If clients start late or experience a failure during the trading day, they may connect via TCP/IP to a Symbol Refresh Service to request referential symbol data. Since refresh messages are published in the normal multicast channels, there is no need for retransmissions or refreshes of real-time data.



XDP Options messages are packaged into three distinct Feeds.

<b>XDP Options Top Feed</b>	Top of book information for outright options symbols
<b>XDP Options Deep Feed</b>	Depth of Book information for outright options symbols
<b>XDP Options Complex Feed</b>	Top of book information for complex options symbols

### 1.1.1 Data Fundamentals

Real-time XDP Options data is message-based with fixed length fields (all fields are binary except a very small number of ASCII fields). In order to make efficient use of the network, the market data messages are bundled into larger application packets. The packets are published via multicast.

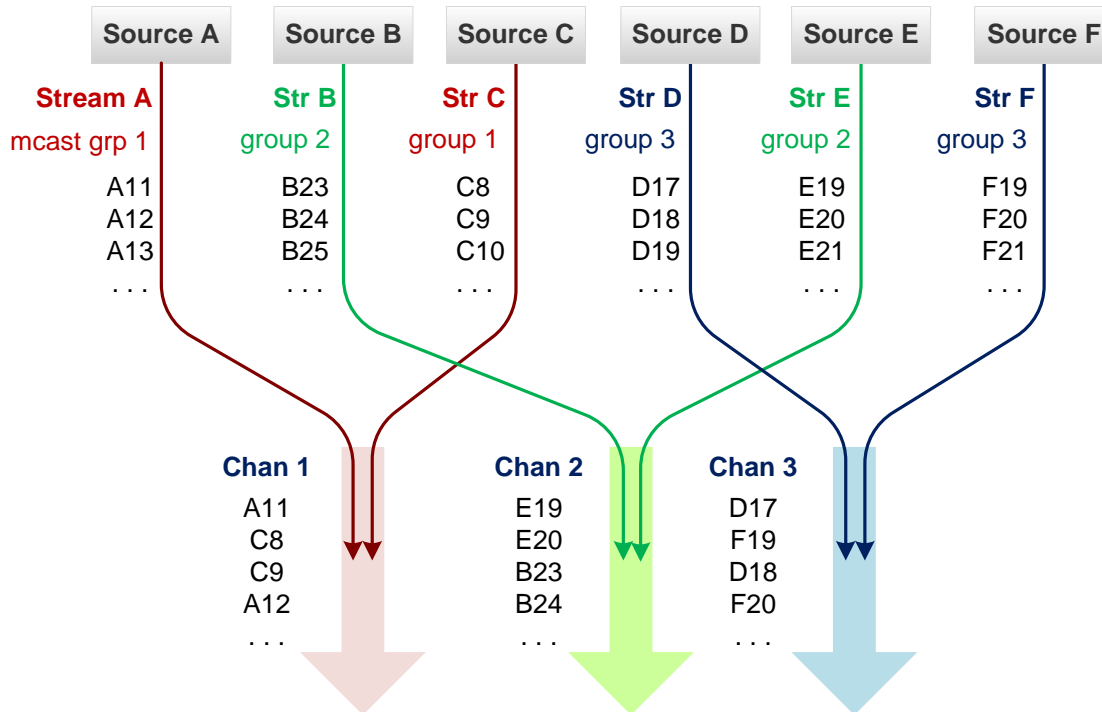
For capacity reasons, the market data packets are routed to clients via a number of predefined data sets called channels. Each channel is duplicated and published to two distinct multicast groups for redundancy. The two redundant multicast groups per channel (known as lines) are referred to as line A and line B.

#### 1.1.1.1 Streams and Sequence Numbers

Due to the very high volume of data in the options markets, a given channel's data typically originates from several different Matching Engine servers. The subset of a channel's data that originates from a given server is referred to as a **stream**. Each packet in a channel is marked with a Stream ID identifying which stream the packet belongs to, and therefore which server the packet came from.

Unlike typical market data feeds in which each channel has a single sequence number series, in XDP Options feeds, each stream within a channel has its own sequence number series. And unlike typical feeds in which a sequence number uniquely identifies a message, in XDP Options feeds, a Stream ID, a Channel ID, and a sequence number taken together uniquely identify a message. (Note that the same stream ID can appear in multiple channels.)

The benefit of this modest increase in complexity is that market data can be published to the client directly from the Matching Engine servers, with no extra tier of publishing applications in between. This results in significantly lower latency to the client.



## 1.2 REFRESH FUNCTIONALITY

If a client experiences a loss of data, either because his application fails and restarts intraday, or because he experiences a sequence number gap on one or more streams, he needs to receive a snapshot of the current state of the market for all affected symbols in order to be in sync with the market again. XDP Options provides this snapshot functionality by periodically publishing Refresh messages directly in the feed channels.

For any individual symbol, as soon as the first order or quote of the day is received, publication of Refresh messages begins.

On client restart or detection of a gap, the client recovers simply by continuing to listen to the feed channels and processing all messages received, including the Refresh messages. The client is guaranteed to be fully in sync with the market once again within at most two minutes.

This inline snapshot publication makes packet request functionality and a dedicated refresh feed unnecessary.

Refresh messages have different message types than their standard counterparts. Furthermore, a specific value displayed in the Delivery flag of the packet header indicates that the packet contains at least one Refresh message.

For the Top and Complex feeds, data points that automatically refresh per symbol are:

Refreshable data	Original message	Refresh message
Current quote	Quote	Refresh Quote
Last Trade	Trade, Correction, Cancel	Refresh Trade, Refresh Correction
Current imbalance, if any (Top only)	Imbalance	Refresh Imbalance

For the Deep feed, data points that automatically refresh per symbol are:

Refreshable data	Original message	Refresh message
Current 3 best price points with aggregated volumes (bid side and ask side)	Market Depth	Refresh Market Depth

**Symbol-status and Crossing RFQ messages are time-sensitive, and therefore are only published as they occur.**

### 1.2.1 Top and Complex feeds: refreshes of current Quote data

The Top feed and the Complex feed are fairly equivalent technically, so their refresh behaviors are the same.

Whenever a symbol publishes a new Quote a 2-minute quote refresh timer is set. If another new Quote is published within the next two minutes, the quote refresh timer is set again, so it starts over again.

As long as the symbol keeps publishing quotes within 2 minutes of each other, the quote refresh timer never expires, and no Refresh Quote message is ever sent. If 2 minutes pass and no new Quote has been published, the timer expires and the information published in the current Quote message is sent again as a Refresh Quote message. The timer is then set again, and if another 2 minutes passes with no new Quote, the same Refresh Quote is sent again. This continues until a new Quote is published.

If a symbol has no current quote because it is before start of day, no Refresh Quotes are sent. If the symbol is halted, Refresh Quote messages with defaulted values are sent every 2 minutes.

Note that although there are no Symbol Status Refresh messages, it is possible to determine that a symbol is halted based on the Quote Condition field in the Quote messages.

### 1.2.2 Top feed: refreshes of current Imbalance data

The Refresh Imbalance algorithm involves a dedicated Imbalance Refresh timer and is exactly the same as the Refresh Quote algorithm, except that when there is no current imbalance, no Refresh Imbalance messages are sent.

### 1.2.3 Top feed: refreshes of last Trade data

The Refresh Trade algorithm involves a dedicated Trade Refresh timer and is exactly the same as the Refresh Quote algorithm, except that there is added complexity when Corrections and Cancels to the last Trade come into the picture.

If a Correction to the last Trade is published, the Trade Refresh timer is set. If 2 minutes pass and no new Trade or Correction/Cancel to the last Trade is published, a Trade Refresh message is published with the corrections applied. This corrected Trade Refresh message is republished every 2 minutes as long as there is no new Trade or Correction/Cancel to the last Trade.

If a Cancel to the last Trade is published, the Trade Refresh timer is set. If 2 minutes pass and no new Trade or Correction/Cancel to the last Trade is published, the new last Trade (previously the second-to-last Trade) is republished as a Refresh Trade, since it is now the new last Trade. Note that any corrections that may have been made to the new last Trade are applied and reflected in the Refresh Trade. This continues every 2 minutes as long as there is no new Trade or Correction/Cancel to the last Trade.

### 1.2.4 Deep feed: refreshes of current book data

The Refresh Depth algorithm involves two dedicated Depth Refresh timers: one for the buy side and one for the sell side. The algorithm for each timer is exactly the same as the Refresh Quote algorithm.

Note that although there are no Symbol Status Refresh messages, it is possible to determine that a symbol is halted based on the non-significant price values in the Depth messages.

### 1.3 SYMBOL INDEX MAPPING

To promote compact messages and therefore high throughput and low latency, underlying symbols, series symbols and complex series symbols are not published as full names in market data messages, but as symbol index codes. Mappings between all symbol index codes and the corresponding full symbol names are provided by a set of symbol index mapping messages. These mapping messages also contain key referential data about each symbol.

Symbol Index Mapping messages come in three different message types corresponding to symbol type: Underlying, Outright, and Complex.

These messages are published over multicast channels, by request via the Request Server, and via a file which the customer may download via ftp every morning.

#### 1.3.1 Symbol Mapping Via Multicast

At the start of the trading day, three sets of symbol index mapping messages (Underlying, Outright, and Complex Index Mapping messages) are published over the multicast channels. The Series feeds, Top and Deep, publish Underlying and Outright mapping messages. The Complex feed publishes Underlying and Outright mapping messages as well as Complex leg definition messages.

In the event a symbol is created during the trading day, individual Index mapping messages can also be published intraday. This is especially applicable to the Complex feed, whose symbols come and go as a matter of course.

#### 1.3.2 Symbol Download via the Request Server

The Request Server can be used to request symbol mapping downloads intraday. Clients must connect to the Request Server via a TCP/IP connection first before sending and receiving the symbol download request and response messages (ack/nack).

To establish a connection, Clients must send a TCP Login Request message and wait for a response. As long as a TCP connection is established, the server generates a Heartbeat every minute to indicate that the session is active. Clients who wish to remain connected throughout the day must respond with a Heartbeat Response message within 5 seconds, or the server will close the connection. Clients can also send a Test message at any time to verify the state of their connections.

The Symbol download request contains a Source ID. The Source ID identifies the client application, and will be supplied by the exchange. It is important to note that only one Source ID can be used per application session. Requests are made by message type (Underlying, Outright, and Complex), and can be defined to request: all symbols of a given type, all symbols of a given type for a given channel, or a specific symbol of a given type.

The Request Server responds to each download request with a Refresh Response message. If the request is accepted, the symbol download is provided over the same TCP/IP connection. When the download is complete, the Request Server generates a second Request Response to indicate that the transmission is done.

The number of refreshes allowed per client per day is limited to a total of 100 requests.

The retransmission request may be rejected for any of the following reasons:

- Invalid Source ID (username)
- Invalid Channel ID
- Incorrectly formatted request packet
- Number of requests in the current day exceeds the predefined system limit

In case of a rejection, the Request Response message indicates the reason for failure.

If the reason for failure is exceeding a predefined system limit, all subsequent requests that continue to violate the limit will be rejected. If further requests are required, the client should contact NYSE.



### 1.3.3 Symbol Download via FTP

For customers who would prefer to download the symbol index mapping information from an FTP server, 2 files containing the information is made available via FTP at 6:00am ET every trading day, one for Amex and one for Arca. Note that options symbols can be added during the trading day. The download files are not updated intraday to keep up with these changes.

These files are in the a pipe-delimited format. The format and FTP locations can be found in Section 7.

## 1.4 PACKET, MESSAGE AND FIELD STRUCTURE

### 1.4.1 General Format Notes

The following processing notes apply to all messages:

- All fields are sent for every message
- Only field values appear in the published messages (no names or 'tags' appear in the messages)
- The field names that appear in the message format documents are for reference purposes only
- All the fields are contiguous, with reserved fields for alignment issues
- All field sizes are fixed
- Message sizes may vary due to optional trailing fields
- Binary fields are published in Little-Endian format
- ASCII fields are left-aligned and null padded
- Segmentation of messages across packets is not supported, so a message will never straddle a packet boundary.
- Reserved filler fields defined as ASCII are filled with a space, and those defined as Binary are populated with '0'.
- No compression is applied

### 1.4.2 Packet Structure

All packets of data sent on the multicast XDP feeds will have a common packet header followed by one or more messages (including the Heartbeat which consists of a packet header followed by a Stream ID message, type 455). For TCP connections to the Request Server, messages are not packetized either from the client or from the server.

The packet header format is the same for all packets, and contains packet length, number of messages within the packet, sequence number for the first message in the packet, and so on.

The format of each message in the packet depends on message type, but each message will start with a message size and a message type.

The maximum length of a decompressed packet is 1,400 bytes.

The message size will never exceed the maximum packet length (less the packet header size).



The packet header provides information including the total packet length, the sequence number of the first message in the packet, and the number of messages within the packet. The format is as follows:

FIELD	OFFSETS	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>PktSize</b>	0	2	Binary	Size of the packet including this 16 -byte packet header in bytes
<b>DeliveryFlag</b>	2	1	Binary	A flag that indicates whether this is an original, retransmitted, or 'replayed' message. Valid values are: <ul style="list-style-type: none"> <li>▪ 1 – Heartbeat</li> <li>▪ 2 – Contains original and Refresh msgs</li> <li>▪ 3 – Contains only Refresh messages</li> <li>▪ 10 – XDP Failover</li> <li>▪ 11 – Contains only original messages</li> <li>▪ 12 – Sequence Number Reset Message</li> </ul>
<b>NumberMsgs</b>	3	1	Binary	The number of messages in this packet
<b>SeqNum</b>	4	4	Binary	The message sequence number of the first message in this packet. See <a href="#">Sequence Numbers</a> .
<b>SendTime</b>	8	4	Binary	The time when the packet was published to the multicast channel, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SendTimeNS</b>	12	4	Binary	The nanosecond offset from the Send Time

The format of each message within a packet varies according to message type.

However, regardless of the message type, each message will start with a message header consisting of two fields: a two-byte message length, followed by a two-byte message type.

**Table 1 Message Header Fields**

FIELD	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	-	2	Binary	Size of the message body in bytes including this field.
<b>MsgType</b>	-	2	Binary	Type of message.

**1.4.3 Msg Size Field Processing**

Customers should not hard code message sizes in feed handlers; instead the feed handler should use the Msg Size field to determine where the next message in the packet begins. This allows the XDP format to accommodate different market needs for data content and allow the format to be more agile.

The variable message size allows the feed handler to insulate client code from any future field additions that may not be desired.

For example, if an original format had the following 20-byte message:

FIELD	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary	Size of the message. 20 bytes
Msg Type	2	2	Binary	Type of message Message '999' – Price message example
SourceTime	4	4		Seconds portion of the ME timestamp.
SourceTimeNS	8	4	Binary	NanoSeconds portion of the ME timestamp.
SymbolIndex	12	4	Binary	Numerical representation of the symbol.
Price	16	4	Binary	Price of the order see Price Formats. Use the Price Scale Code from the symbol index mapping message.

Look at the Msg Size field to know where the next record will be.

Now the new format adds a new four-byte volume field adjusting the Msg Size to 20 bytes.

The feed handler code automatically is prepared for the 24-byte format without any work, allowing for the receiving application to either continue to read on the first 20 bytes passed to it or develop to read the new field.

FIELD	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary	Size of the message. 24 Bytes
Msg Type	2	2	Binary	Type of message Message '999' – Price message example
SourceTime	4	4		Seconds portion of the ME timestamp.
SourceTimeNS	8	4	Binary	NanoSeconds portion of the ME timestamp.
SymbolIndex	12	4	Binary	Numerical representation of

Look at the Msg Size field to know where the next record will be.

				the symbol.
<b>Price</b>	16	4	Binary	Price of the order, see <a href="#">Price Formats</a> . Use the Price Scale Code from the symbol index mapping message.
<b>Volume</b>	20	4	Binary	Size of the order.

#### 1.4.4 Stream ID Message

In all XDP Options multicast channels, each Packet Header is immediately followed by a Stream ID Message. This message's Stream ID field identifies which stream in the channel this packet belongs to.

By combining the Stream ID with the Sequence Number in the Packet Header, the client can uniquely identify each packet and can detect gaps (lost packets) within the stream.

**NOTE: All complex symbol indexes are unique per stream only, not across the entire market.**

See [Streams and Sequence Numbers](#) for more information.

Note that streams are not applicable to TCP connections to the Request Server, so Stream ID messages are not sent by either the client or by the Request Server.

#### 1.4.5 Sequence Numbers

When reading the multicast channels, client applications should check the Stream ID and Sequence Number of every packet received.

Sequence numbers are unique for each stream, although they do not increase monotonically. Each new sequence number is incremented not by 1, but by the number of messages in the previous packet. So for example if a packet has Stream ID A, Sequence Number 60, and Num Msgs 5, then the next packet to arrive in Stream A will have Sequence Number 65.

Sequence numbers in the packet header are not packet sequence numbers, but message sequence numbers. For lower bandwidth, message sequence number fields are not explicitly expressed in each message, but once only in the packet header.

#### 1.4.6 Symbol Sequence Numbers

In addition to the sequence number, many message types explicitly include a field called SymbolSeqNum, which identifies the message's position in the sequence of all messages published by the feed for an individual symbol.

The SymbolSeqNum increments every time a new non-refresh message for the same symbol is published. Refresh messages do not increment the SymbolSeqNum.

Clients who are tracking only a small number of symbols may opt to ignore message sequence numbers and track only Symbol Sequence Numbers for each symbol of interest.

If packets are dropped which do not contain updates to any of the symbols the client is tracking, he doesn't know about the gap and he doesn't need to know. If such a client experiences a Symbol Sequence Number gap for a symbol he's tracking, he has to invalidate all current state for that symbol and listen to both current and refresh messages until the symbol's state is fully restored to current values.

#### 1.4.7 Line Arbitration

Each channel is duplicated and published to two distinct multicast groups for redundancy. The redundant groups are referred to as line A and line B.

Client applications are advised to read both lines in real-time and process the packets that arrive first, regardless of whether they come from line A or line B. Packets that have lower-than-expected sequence numbers should be discarded.

Using this algorithm, if a packet has been dropped from line A, it can still be read from line B, and vice versa.

In the rare case where a packet is dropped from both lines, it is necessary to use the refresh mechanism to re-sync the market in the stream that gapped. (see [Refresh Functionality](#))

### 1.4.8 Date and Time Conventions

Dates and times use UTC (Universal Time, Coordinated) EPOCH. For example Wednesday 12/1/09 22:05:17.000 UTC is indicated as 1259791537.

Time stamps are provided over two fields: one that identifies the whole number of seconds in UTC time, and another that contains the nanosecond offset.

NOTE: Refresh messages contain the same timestamp as the original message that the refresh represents derived from.

### 1.4.9 Prices

All price fields are published as signed binary integers (prices of complex instruments can be negative). To interpret a price correctly, the client must use the published price value as a numerator along with the Price Scale Code in the symbol's Index Mapping Message as follows:

$$Price = \frac{Numerator}{10^{PriceScaleCode}}$$

For example, a price of \$27.56 is represented as a published price field of 2756 and a PriceScaleCode of 2.

## 1.5 HEARTBEATS

### 1.5.1 Heartbeat Packets in Multicast Feeds

Heartbeat packets are sent over the multicast channels during the startup sequence and in periods of total inactivity (see [System Behavior on Start and Restart](#) for more information). Heartbeat packets are sent per stream, and therefore consist of a packet header followed by a Stream ID message.

**Heartbeat packet header layout**

FIELD	VALUE
PktSize	16 Bytes
DeliveryFlag	1 (Heartbeat Message Only)
NumberMsgs	1, due to the Stream ID msg (type 455) that follows
SeqNum	Next expected sequence number
SendTime	
SendTimeNS	

### 1.5.2 TCP Heartbeat Processing Notes

The following applies to TCP connections to the Request Server only.

Clients receive TCP Heartbeat messages (Msg Type 12) periodically (nominally every 60 seconds) whenever they have a TCP/IP connection with the Request Server.

Clients must respond to any heartbeat received by sending back a TCP Heartbeat message within 5 seconds. If no response is received by the Request Server within this timeframe, the server will close the TCP/IP connection.

The Request Server uses this mechanism to close improperly dropped connections, which can arise as a result of client or network hardware failures.

### Request Server Heartbeats



## 1.6 OPERATIONAL INFORMATION

### 1.6.1 System Behavior on Start and Restart

At the start of the day, the following messages will be published over each stream:

- 10 Heartbeats for network multicast priming
- Sequence Number Reset message (Msg Type 1), sequence number is set to 1

This means that if channel X is composed of 3 streams, subscribers to channel X will see a total of 30 Heartbeats and 3 Sequence Number Reset messages.

### 1.6.2 Client System Failure

In case of client system failure, the client should restart, re-subscribe to the multicast channels, and rely on the inline Refresh messages to re-sync with the current state of the market.

For more information, see [Refresh Functionality](#).

## 1.7 REQUEST SERVER PER-CLIENT QUOTAS

Requests made to the Request Server for refreshes of symbol information will not be accepted until the symbol index messages are published on the main data channels. This occurs at approximately 3:00 AM EST.

Requests are fulfilled in the order in which they were received.

The following limitations apply to all clients a given day.

1,000	Maximum number of requests allowed	Further requests will be rejected
50	Maximum connection attempts within 30 seconds	The client IP will be locked out
3	Maximum number of simultaneous login sessions	Further sessions will be closed
30 seconds	Maximum seconds between a TCP connection and a login request	Session will be closed

## 2. Top Feed Message Formats

In addition to the message types detailed here, the Top feed also publishes Underlying Index Mapping messages (435), Series Index Mapping messages (437), and various control messages. These are described in later sections.

### 2.1 OUTRIGHT QUOTE MESSAGE – MSG TYPE 401

The Outright Quote message provides the price and aggregated volume for the best bid and best offer. Volumes aggregate quote and order interest. The message also indicates aggregated Customer volumes within the overall volumes.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>40 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>401 – Outright Quote Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the outright options symbol.
<b>AskPrice</b>	20	4	Signed Binary	Best Ask price. To be used with the Price Scale Code from the Outright Index Mapping message.
<b>BidPrice</b>	24	4	Signed Binary	Best Bid price. To be used with the Price Scale Code from the Outright Index Mapping message.
<b>AskShares</b>	28	2	Binary	Total quantity available at the above Ask price.
<b>BidShares</b>	30	2	Binary	Total quantity available at the above Bid price.
<b>AskCustomerShares</b>	32	2	Binary	Total quantity of 'Customer' orders available at the Ask price.
<b>Bid CustomerShares</b>	34	2	Binary	Total quantity of 'Customer' orders available at the Bid price.
<b>QuoteCondition</b>	36	1	ASCII	<ul style="list-style-type: none"> <li>1 (Regular Trading)</li> <li>2 (Rotation)</li> <li>3 (Trading Halted)</li> <li>4 (Pre-open)</li> </ul>
<b>Reserved</b>	37	1	Binary	Filler
<b>Reserved</b>	38	2	Binary	Filler



## 2.2 OUTRIGHT TRADE MESSAGE – MSG 407

The Outright Trade message is used to publish all Last Sales for outright symbols.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>36 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>407 – Trade Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>TradeID</b>	20	4	Binary	Unique Trade execution ID.
<b>Price</b>	24	4	Signed Binary	Price of the trade. Use the Price Scale Code from the symbol index mapping message.symbol index mapping message
<b>Volume</b>	28	4	Binary	Volume of the trade in number of contracts
<b>TradeCond1</b>	32	1	ASCII	<ul style="list-style-type: none"> <li>Blank = regular trade</li> <li>I = Late report</li> <li>R = Floor trade</li> <li>S = SO sweep trade</li> </ul>
<b>TradeCond2</b>	33	1	ASCII	Complex indicator: <ul style="list-style-type: none"> <li>P = Complex trade with equity trade</li> <li>L = Complex trade</li> </ul>
<b>Reserved</b>	34	2	Binary	Filler

## 2.3 OUTRIGHT TRADE CANCEL MESSAGE – MSG TYPE 409

The Trade Cancel message is used to cancel or bust a trade.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>24 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>409 – Outright Trade Cancel Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>OriginalTradeID</b>	20	4	Binary	Original TradeID of the trade.

## 2.4 OUTRIGHT TRADE CORRECTION MESSAGE – MSG TYPE 411

The Trade Correction message is used to correct a trade.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>Msg Size</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>40 bytes</li> </ul>
<b>Msg Type</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>411 – Outright Trade Correction Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>OriginalTradeID</b>	20	4	Binary	Original TradeID of the trade.
<b>TradeID</b>	24	4	Binary	New unique TradeID used to identify the corrected trade.
<b>Price</b>	28	4	Signed Binary	Price of the order. Use the Price Scale Code from the symbol index mapping message.
<b>Volume</b>	32	4	Binary	volume of the trade in number of contracts.
<b>TradeCond1</b>	36	1	ASCII	<ul style="list-style-type: none"> <li>Blank = regular trade</li> <li>I = Late report</li> <li>R = Floor trade</li> <li>S = SO sweep trade</li> </ul>
<b>TradeCond2</b>	37	1	ASCII	Complex indicator: <ul style="list-style-type: none"> <li>P = Complex trade with equity trade</li> <li>L = Complex trade</li> </ul>
<b>Reserved</b>	38	2	Binary	Filler

## 2.5 OUTRIGHT IMBALANCE MESSAGE – MSG TYPE 413

The Outright Imbalance message is sent during the Pre-opening phase or a symbol Halt to provide indicative opening price and net imbalance of all orders at that price which are orders eligible for next Auction. It also indicates the type of Auction to follow. Imbalance values are calculated using both orders and quotes.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>Msg Size</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>36 bytes</li> </ul>
<b>Msg Type</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>413 – Outright Imbalance</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>ReferencePrice</b>	20	4	Signed Binary	Indicative opening price.
<b>PairedQty</b>	24	2	Binary	Paired off quantity at the indicative opening price (Indicative matching quantity)
<b>TotalImbalanceQty</b>	26	2	Binary	Total imbalance quantity at the indicative opening price.
<b>MarketImbalanceQty</b>	28	2	Binary	Total market order imbalance at the indicative opening price.
<b>AuctionType</b>	30	1	ASCII	<ul style="list-style-type: none"> <li>O = Opening</li> <li>H = Halt</li> </ul>
<b>ImbalanceSide</b>	31	1	ASCII	Side of the imbalance Buy/sell. Valid Values: <ul style="list-style-type: none"> <li>B = Buy</li> <li>S = Sell</li> <li>Blank = No imbalance</li> </ul>
<b>MarketImbalanceSide</b>	32	1	ASCII	Side of the market imbalance Buy/sell. Valid Values: <ul style="list-style-type: none"> <li>B = Buy</li> <li>S = Sell</li> <li>Blank = No imbalance</li> </ul>
<b>Reserved</b>	33	3	Binary	Filler

## 2.6 OUTRIGHT CROSSING RFQ MESSAGE – MSG 415

The Outright Crossing RFQ (Request for Quote) message is sent out in the event of an auction on an outright order.

FIELD	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>28 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>415 - Outright Crossing RFQ</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>Side</b>	20	1	ASCII	Side of the RFQ <ul style="list-style-type: none"> <li>B = Buy</li> <li>S = Sell</li> </ul>
<b>Reserved</b>	21	1	Binary	Filler
<b>Shares</b>	22	2	Binary	Total quantity
<b>Price</b>	24	4	Signed Binary	Price of crossing transaction

## 2.7 OUTRIGHT SUMMARY MESSAGE – MSG 417

The Outright summary message provides trading highlights of the day. It is sent out once at the end of the trading day as soon as a symbol closes.

Note: If no trades occurred on a symbol for the whole trading day, no End of Day Summary is generated.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>Msg Size</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>40 bytes</li> </ul>
<b>Msg Type</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>417 – Outright Summary message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>HighPrice</b>	20	4	Signed Binary	Exchange high price of the symbol for the day. Use the Price Scale Code from the symbol index mapping message.
<b>LowPrice</b>	24	4	Signed Binary	Exchange Low price of the symbol for the day. Use the Price Scale Code from the symbol index mapping message.
<b>Open</b>	28	4	Signed Binary	Exchange Opening price of the symbol for the day. Use the Price Scale Code from the symbol index mapping message.
<b>Close</b>	32	4	Signed Binary	Exchange Closing price of the symbol for the day. Use the Price Scale Code from the symbol index mapping message.
<b>TotalVolume</b>	36	4	Binary	Exchange cumulative volume for the symbol throughout the day.

## 2.8 UNDERLYING STATUS MESSAGE – MSG 419

The Underlying Status message is used to inform the subscribers of changes in the status of an Underlying symbol.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Message size: <ul style="list-style-type: none"> <li>24 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>419 – Underlying Status Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>UnderlyingIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>UnderlyingSeqNum</b>	16	4	Binary	Sequence number of messages for the underlying symbol.
<b>SecurityStatus</b>	20	1	ASCII	<ul style="list-style-type: none"> <li>S Halt</li> <li>U Unhalt</li> <li>O Open indication</li> <li>X Close indication</li> </ul>
<b>Halt Condition</b>	21	1	ASCII	Not applicable
<b>Reserved</b>	22	2	Binary	Filler

## 2.9 OUTRIGHT SERIES STATUS MESSAGE – MSG 421

The Series Status message is used to inform the subscribers of changes in symbol status of an outright symbol.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>24 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>421 – Outright Series Status Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>SecurityStatus</b>	20	1	ASCII	<ul style="list-style-type: none"> <li>L Light up a dark series</li> <li>N Open a dark series</li> <li>O Open</li> <li>X Close</li> <li>S Halt</li> <li>U Unhalt</li> <li>T Unhalt a dark series</li> <li>Q End of RFQ auction</li> </ul>
<b>HaltCondition</b>	21	1	ASCII	Not applicable
<b>Reserved</b>	22	2	Binary	Filler



## 2.10 REFRESH OUTRIGHT QUOTE MESSAGE – MSG TYPE 501

The Refresh Quote message is sent in the event no quote messages are published for the symbol for 2 consecutive minutes. It has the same field content as the last published Outright Quote message, with the following exceptions:

- MsgType is 501
- SymbolSeqNum increments on every publication

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>▪ 40 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>▪ 501 – Refresh Outright Quote message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this data was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	Numerical representation of the outright options symbol.
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>AskPrice</b>	20	4	Signed Binary	Best Ask price. Should be used with the Price Scale Code from the symbol index mapping message.
<b>BidPrice</b>	24	4	Signed Binary	Best Bid price. Should be used with the Price Scale Code from the symbol index mapping message.
<b>AskShares</b>	28	2	Binary	total quantity available at the above Ask price.
<b>BidShares</b>	30	2	Binary	total quantity available at the above Bid price.
<b>AskCustomerShares</b>	32	2	Binary	total quantity of 'Customer' orders available at the above Ask price.
<b>BidCustomerShares</b>	34	2	Binary	total quantity of 'Customer' orders available at the above Bid price.
<b>QuoteCondition</b>	36	1	ASCII	<ul style="list-style-type: none"> <li>▪ 1 (Regular Trading)</li> <li>▪ 2 (Rotation)</li> <li>▪ 3 (Trading Halted)</li> <li>▪ 4 (Pre-open)</li> </ul>
<b>Reserved</b>	37	1	Binary	Filler
<b>Reserved</b>	37	2	Binary	Filler

## 2.11 REFRESH OUTRIGHT TRADE MESSAGE – MSG 507

The Refresh Outright Trade message is sent in the event no Outright Trade messages are published for the symbol for 2 consecutive minutes. It has the same field content as the last published Outright Trade message (assuming no intervening corrections), with the following exceptions:

- MsgType is 507
- SymbolSeqNum increments on every publication

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>▪ 36 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>▪ 507 – Refresh Outright Trade Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this data was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	Numerical representation of the outright options symbol.
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the outright options symbol.
<b>TradeID</b>	20	4	Binary	The TradeID identifies a unique Trade execution.
<b>Price</b>	24	4	Signed Binary	Price of the trade.
<b>Volume</b>	28	4	Binary	Volume of the trade in number of contracts
<b>TradeCond1</b>	32	1	ASCII	Blank (regular trade) <ul style="list-style-type: none"> <li>▪ I (Late report)</li> <li>▪ R (Floor trade)</li> <li>▪ S (ISO sweep trade)</li> </ul>
<b>TradeCond2</b>	33	1	ASCII	Complex indicator: <ul style="list-style-type: none"> <li>▪ P (Complex trade with equity trade)</li> <li>▪ L (Complex trade)</li> </ul>
<b>Reserved</b>	34	2	Binary	Filler

## 2.12 REFRESH OUTRIGHT IMBALANCE MESSAGE – MSG TYPE 509

The Refresh Outright Imbalance message is sent in the event no Outright Imbalance messages are published for the symbol for 2 consecutive minutes during Pre-opening or a symbol Halt. It has the same field content as the last published Outright Imbalance message, with the following exceptions:

- MsgType is 509
- SymbolSeqNum increments on every publication

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>Msg Size</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>▪ 36 bytes</li> </ul>
<b>Msg Type</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>▪ 509 – Refresh Trade message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this data was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	Numerical representation of the symbol.
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>ReferencePrice</b>	20	4	Signed Binary	Indicative matching price
<b>PairedQty</b>	24	2	Binary	Paired off quantity at the indicative matching price (Indicative matching quantity)
<b>TotalImbalanceQty</b>	26	2	Binary	Total imbalance quantity at the indicative matching price.
<b>MarketImbalanceQty</b>	28	2	Binary	Total market order imbalance at the indicative matching price
<b>AuctionType</b>	30	1	ASCII	<ul style="list-style-type: none"> <li>▪ O (Opening)</li> <li>▪ H (Halt)</li> </ul>
<b>ImbalanceSide</b>	33	1	ASCII	Side of the imbalance Buy/sell. Valid Values: <ul style="list-style-type: none"> <li>▪ 'B' – Buy</li> <li>▪ 'S' – Sell</li> <li>▪ Space – No imbalance</li> </ul>
<b>MarketImbalanceSide</b>	32	1	ASCII	Side of the market imbalance Buy/sell. Valid Values: <ul style="list-style-type: none"> <li>▪ B = Buy</li> <li>▪ S = Sell</li> <li>▪ Blank = No imbalance</li> </ul>
<b>Reserved</b>	33	3	Binary	Filler

### 3. Deep Feed Message Types

In addition to the message types detailed here, the Deep feed also publishes Underlying Index Mapping messages (435), Series Index Mapping messages (437), and various control messages. These are described in later sections.

#### 3.1 OUTRIGHT MARKET DEPTH MESSAGE, BUY – MSG TYPE 403

The Buy Market Depth message provides prices and aggregated volumes for each of the best three Buy levels. Volumes include orders and quotes.

There is a separate message for the best three Sell levels. No add/delete messages are used.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>40 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>403 – Buy Market Depth</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>First Level Price</b>	20	4	Signed Binary	Price of first buy level
<b>Second Level Price</b>	24	4	Signed Binary	Price of second buy level
<b>Third Level Price</b>	28	4	Signed Binary	Price of third buy level
<b>First Level Volume</b>	32	2	Binary	Total volume at first price level
<b>Second Level Volume</b>	34	2	Binary	Total volume at second price level
<b>Third Level Volume</b>	36	2	Binary	Total volume at third price level
<b>Reserved</b>	38	2	Binary	Filler

### 3.2 OUTRIGHT MARKET DEPTH MESSAGE, SELL – MSG TYPE 405

The Sell Market Depth message provides prices and aggregated volumes for each of the best three Sell levels. Aggregated volumes include orders and quotes.

There is a separate message for the best three Buy levels. No add/delete messages are used.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>40 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>405 – Sell Market Depth</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	The unique ID of the symbol in the Series Index message
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>First Level Price</b>	20	4	Signed Binary	Price of First Sell level
<b>Second Level Price</b>	24	4	Signed Binary	Price of Second Sell level
<b>Third Level Price</b>	28	4	Signed Binary	Price of Third Sell level
<b>First Volume</b>	32	2	Binary	Total volume at First price level
<b>Second Volume</b>	34	2	Binary	Total volume at Second price level
<b>Third Volume</b>	36	2	Binary	Total volume at Third price level
<b>Reserved</b>	38	2	Binary	Filler

### 3.3 REFRESH OUTRIGHT MARKET DEPTH MESSAGE, BUY – MSG TYPE 503

The Refresh Market Depth Buy message is sent in the event no Buy Market Depth messages are published for the symbol for 2 consecutive minutes. It has the same field content as the last published Outright Quote message, with the following exceptions:

- MsgType is 503
- SymbolSeqNum increments on every publication

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>▪ 40 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>▪ 503 – Refresh Buy Market Depth</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this data was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	Numerical representation of the outright options symbol.
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>First Level Price</b>	20	4	Signed Binary	Price of first buy level
<b>Second Level Price</b>	24	4	Signed Binary	Price of second buy level
<b>Third Level Price</b>	28	4	Signed Binary	Price of third buy level
<b>First Volume</b>	32	2	Binary	Total volume at first price level
<b>Second Volume</b>	34	2	Binary	Total volume at second price level
<b>Third Volume</b>	36	2	Binary	Total volume at third price level
<b>Reserved</b>	38	2	Binary	Filler

### 3.4 REFRESH OUTRIGHT MARKET DEPTH MESSAGE, SELL – MSG TYPE 505

The Refresh Market Depth Sell message is sent in the event no Sell Market Depth messages are published for the symbol for 2 consecutive minutes. It has the same field content as the last published Sell Market Depth message, with the following exceptions:

- MsgType is 505
- SymbolSeqNum increments on every publication

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>▪ 40 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>▪ 505 – Refresh Sell Market Depth</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this data was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>SeriesIndex</b>	12	4	Binary	Numerical representation of the outright options symbol.
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the outright options symbol.
<b>First Level Price</b>	20	4	Signed Binary	Price of first sell level
<b>Second Level Price</b>	24	4	Signed Binary	Price of second sell level
<b>Third Level Price</b>	28	4	Signed Binary	Price of third sell level
<b>First Volume</b>	32	2	Binary	Total volume at first price level
<b>Second Volume</b>	34	2	Binary	Total volume at second price level
<b>Third Volume</b>	36	2	Binary	Total volume at third price level
<b>Reserved</b>	38	2	Binary	Filler

## 4. Complex Feed Message Formats

In addition to the message types detailed here, the Complex feed also publishes Complex Symbol Definition messages (439) and various control messages. These are described in later sections.

### 4.1 COMPLEX QUOTE MESSAGE – MSG TYPE 423

The Complex Quote message is a two-sided message providing best bid and offer limits of orders on complex symbols with aggregated volumes at each limit. The message also indicates aggregated Customer volumes within the overall volumes.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>40 Bytes</li> </ul>
MsgType	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>423 – Complex Quote</li> </ul>
SourceTime	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
SourceTimeNS	8	4	Binary	The nanosecond offset from the SourceTime.
ComplexIndex	12	4	Binary	The unique ID of the symbol in the Complex Index msg
SymbolSeqNum	16	4	Binary	Sequence number of messages for the complex symbol.
AskPrice	20	4	Signed Binary	Best Ask price. Should be used with the Price Scale Code from the Underlying Index Mapping message.
BidPrice	24	4	Signed Binary	Best Bid price. Should be used with the Price Scale Code from the Underlying Index Mapping message.
AskShares	28	2	Binary	total quantity available at the above Ask price.
BidShares	30	2	Binary	total quantity available at the above Bid price.
AskCustomerShares	32	2	Binary	total quantity of 'Customer' orders available at the above Ask price.
BidCustomerShares	34	2	Binary	total quantity of 'Customer' orders available at the above Bid price.
QuoteCondition	36	1	ASCII	<ul style="list-style-type: none"> <li>1 (Regular Trading)</li> <li>2 (Rotation)</li> <li>3 (Trading Halted)</li> <li>4 (Pre-open)</li> </ul>
Reserved1	37	1	Binary	Filler
Reserved2	38	2	Binary	Filler



## 4.2 COMPLEX TRADE MESSAGE – MSG 425

The Complex Trade message is used to publish all complex Last Sales.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>36 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>425 – Trade Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>ComplexIndex</b>	12	4	Binary	The unique ID of the symbol in the Complex Index msg
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the complex symbol.
<b>TradeID</b>	20	4	Binary	(blank) n/a for complex symbols
<b>Price</b>	24	4	Signed Binary	Price of the trade. Use the Price Scale Code from the Underlying Index Mapping message.
<b>Volume</b>	28	4	Binary	Volume of the trade in actual number of contracts
<b>TradeCond1</b>	32	1	ASCII	<ul style="list-style-type: none"> <li>Blank (regular trade)</li> <li>I (Late report)</li> <li>R (Floor trade)</li> <li>S (ISO sweep trade)</li> </ul>
<b>TradeCond2</b>	33	1	ASCII	(blank) n/a for complex symbols
<b>Reserved</b>	34	2	Binary	Filler

#### 4.1 COMPLEX CROSSING RFQ MESSAGE– MSG 429

The Complex Crossing RFQ (Request for Quote) message is sent out in the event of an auction on a complex order.

FIELD	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message body in bytes. <ul style="list-style-type: none"> <li>28 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>429 – Complex Crossing</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>ComplexIndex</b>	12	4	Binary	The unique ID of the symbol in the Complex Index msg
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the complex symbol.
<b>Side</b>	20	1	ASCII	Side of the RFQ <ul style="list-style-type: none"> <li>B (Buy)</li> <li>S (Sell)</li> </ul>
<b>Reserved</b>	21	1	Binary	Filler
<b>Shares</b>	22	2	Binary	Total quantity
<b>Price</b>	24	4	Signed Binary	Price of crossing transaction. Use the Price Scale Code from the Underlying Index Mapping message. 999999999 = The price is not displayed.

## 4.2 COMPLEX STATUS MESSAGE – MSG 433

The Complex Status message is used to inform the subscribers of changes in symbol status of a complex symbol.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Message size: <ul style="list-style-type: none"> <li>24 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>433 – Complex Status Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	Second portion of the ME timestamp.
<b>SourceTimeNS</b>	8	4	Binary	Nanosecond portion of the ME timestamp.
<b>ComplexIndex</b>	12	4	Binary	Numerical representation of the complex symbol.
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the complex symbol.
<b>SecurityStatus</b>	20	1	ASCII	<ul style="list-style-type: none"> <li>O Open</li> <li>X Close</li> <li>S Halt</li> <li>Q End of RFQ auction</li> </ul>
<b>Halt Condition</b>	21	1	ASCII	Not applicable
<b>Reserved</b>	22	2	Binary	Filler

### 4.3 REFRESH COMPLEX QUOTE MESSAGE – MSG TYPE 511

The Refresh Complex Quote message is sent in the event no quote messages are published for the symbol for 2 consecutive minutes. It has the same field content as the last published Complex Quote message, with the following exceptions:

- MsgType is 511
- SymbolSeqNum increments on every publication

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>▪ 40 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>▪ 511 – Refresh Complex Quote Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this data was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>ComplexIndex</b>	12	4	Binary	Numerical representation of the complex symbol.
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the complex symbol.
<b>AskPrice</b>	20	4	Signed Binary	Best Ask price. Use the Price Scale Code from the Underlying Index Mapping message.
<b>BidPrice</b>	24	4	Signed Binary	Best Bid price. Use the Price Scale Code from the Underlying Index Mapping message.
<b>AskShares</b>	28	2	Binary	Total quantity available at the Ask price.
<b>BidShares</b>	30	2	Binary	Total quantity available at the Bid price.
<b>AskCustomerShares</b>	32	2	Binary	Total quantity of 'Customer' orders available at the above Ask price.
<b>BidCustomerShares</b>	34	2	Binary	Total quantity of 'Customer' orders available at the above Bid price.
<b>QuoteCondition</b>	36	1	ASCII	<ul style="list-style-type: none"> <li>▪ 1 (Regular Trading)</li> <li>▪ 2 (Rotation)</li> <li>▪ 3 (Trading Halted)</li> <li>▪ 4 (Pre-open)</li> </ul>
<b>Reserved1</b>	37	1	Binary	Filler
<b>Reserved2</b>	38	2	Binary	Filler

#### 4.5 REFRESH COMPLEX TRADE MESSAGE– MSG 513

The Refresh Complex Trade message is sent in the event no Complex Trade messages are published for the symbol for 2 consecutive minutes. It has the same field content as the last published Complex Trade message, with the following exceptions:

- MsgType is 513
- SymbolSeqNum increments on every publication

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>▪ 36 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>▪ 513 – Refresh Complex Trade Message</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this data was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>ComplexIndex</b>	12	4	Binary	Numerical representation of the symbol.
<b>SymbolSeqNum</b>	16	4	Binary	Sequence number of messages for the Outright options symbol.
<b>TradeID</b>	20	4	Binary	(blank) n/a for complex symbols
<b>Price</b>	24	4	Signed Binary	Price of the trade. Use the Price Scale Code from the Underlying Index Mapping message.
<b>Volume</b>	28	4	Binary	Volume of the trade in actual number of contracts
<b>TradeCond1</b>	32	1	ASCII	<ul style="list-style-type: none"> <li>▪ Blank (regular trade)</li> <li>▪ I (Late report)</li> <li>▪ R (Floor trade)</li> <li>▪ S (ISO sweep trade)</li> </ul>
<b>TradeCond2</b>	33	1	ASCII	n/a
<b>Reserved</b>	34	2	Binary	Filler

## 5. Index Mapping Message Formats

### 5.1 UNDERLYING INDEX MAPPING MESSAGE – MSG 435

On system startup, each multicast channel sends all the symbols on its channel. Underlying Index Mapping messages are published first, followed by all Series Index Mapping messages, and for the Complex feed, Complex Symbol Definitions last of all.

If individual series for the same underlying symbol appear in multiple streams, the Underlying Index Mapping message is sent down all relevant channels.

This message is published in Top, Deep and Complex feeds. It is also available via a symbol refresh request.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>Msg Size</b>	0	2	Binary	Size of the message: 28 bytes
<b>Msg Type</b>	2	2	Binary	Type of message 435 – Underlying Index Mapping
<b>UnderlyingIndex</b>	4	4	Binary	The unique ID of this underlying symbol in this market. This ID cannot be used to cross reference a security between markets.
<b>UnderlyingSymbol</b>	8	11	ASCII	Full symbol in NYSE Symbology.
<b>ChannelID</b>	19	1	Binary	Multicast channel ID of the symbols being provided.
<b>Market ID</b>	20	2	Binary	Identifies originating market: <ul style="list-style-type: none"> <li>▪ 1 (NYSE Cash)</li> <li>▪ 2 (Europe Cash)</li> <li>▪ 3 (NYSE Arca Cash)</li> <li>▪ 4 (NYSE/Arca Options)</li> <li>▪ 5 (NYSE/Arca Bonds)</li> <li>▪ 6 (ArcaEdge)</li> <li>▪ 7 (LIFFE)</li> <li>▪ 8 (NYSE Amex Options)</li> <li>▪ 9 (NYSE MKT Cash)</li> </ul>
<b>System ID</b>	22	1	Binary	ID of the Originating System
<b>Exchange Code</b>	23	1	ASCII	Exchanges where it is listed: <ul style="list-style-type: none"> <li>▪ N (NYSE)</li> <li>▪ P (NYSE Arca)</li> <li>▪ Q (NASDAQ)</li> <li>▪ A (NYSE MKT)</li> </ul>
<b>PriceScaleCode</b>	24	1	Binary	Price Scale Code for price conversion of the symbol. See Price Formats.
<b>SecurityType</b>	25	1	ASCII	Type of Security: <ul style="list-style-type: none"> <li>▪ A (ADR)</li> <li>▪ C (COMMON STOCK)</li> <li>▪ D (DEBENTURES)</li> <li>▪ E (ETF)</li> </ul>

				<ul style="list-style-type: none"> <li>▪ F (FOREIGN)</li> <li>▪ I (UNITS)</li> <li>▪ M (MISC/LIQUID TRUST)</li> <li>▪ P (PREFERRED STOCK)</li> <li>▪ R (RIGHTS)</li> <li>▪ S (SHARES OF BENEFICIARY INTEREST)</li> <li>▪ T (TEST)</li> <li>▪ U (UNITS)</li> <li>▪ W (WARRANT)</li> </ul>
<b>Price Resolution</b>	26	1	Binary	<ul style="list-style-type: none"> <li>▪ 0 (All Penny)</li> <li>▪ 1 (Penny/Nickel)</li> <li>▪ 5 (Nickel/Dime)</li> </ul>
<b>Reserved</b>	27	1	Binary	Filler

## 5.2 SERIES INDEX MAPPING MESSAGE– MSG 437

Series symbols are used to identify options instruments. They are unique across the entire market.

On system startup, each multicast channel sends all the symbols on its channel. Underlying Index Mapping messages are published first, followed by all Series Index Mapping messages, and for the Complex feed, Complex Symbol Definitions last of all.

A Series Index message is also sent intraday whenever a new outright symbol is created (or was erroneously omitted from the initial spin), and is available via the symbol refresh request.

This message is published in Top, Deep and Complex feeds. It is also available via a symbol refresh request.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Message size: <ul style="list-style-type: none"> <li>60 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>437 - Series Index Mapping</li> </ul>
<b>SeriesIndex</b>	4	4	Binary	The unique ID of this series in this stream. This ID cannot be used to cross reference a security between markets.
<b>ChannelID</b>	8	1	Binary	Multicast channel ID of the symbols being provided.
<b>Reserved</b>	9	1	Binary	Filler
<b>MarketID</b>	10	2	Binary	Identifies originating market: <ul style="list-style-type: none"> <li>1 (NYSE Cash)</li> <li>2 (Europe Cash)</li> <li>3 (NYSE Arca Cash)</li> <li>4 (NYSE/Arca Options)</li> <li>5 (NYSE/Arca Bonds)</li> <li>6 (Global OTC)</li> <li>7 (LIFFE)</li> <li>8 (NYSE Amex Options)</li> <li>9 (NYSE MKT Cash)</li> </ul>
<b>SystemID</b>	12	1	Binary	Identifies Trading unit (TU)
<b>Reserved</b>	13	1	Binary	Filler
<b>StreamID</b>	14	2	Binary	Identifies Stream on which this symbol will be updated
<b>UnderlyingIndex</b>	16	4	Binary	Underlying Stock Mapping Index
<b>ContractMultiplier</b>	20	2	Binary	Contract quantity
<b>MaturityDate</b>	22	6	ASCII	YY MM DD
<b>PutOrCall</b>	28	1	Binary	<ul style="list-style-type: none"> <li>0 (Put)</li> <li>1 (Call)</li> </ul>
<b>StrikePrice</b>	29	10	ASCII	Strike price. ASCII 0-9 with optional decimal point. EG: 51.75, 123
<b>PriceScaleCode</b>	39	1	Binary	Decimal places on price



<b>UnderlyingSymbol</b>	40	11	ASCII	Full underlying symbol in NYSE Symbology
<b>OptionSymbolRoot</b>	51	5	ASCII	OCC root of option symbol
<b>GroupID</b>	56	4	Binary	Used by Market Makers. Predefined group of series within a given underlying symbol.

### 5.3 COMPLEX SYMBOL DEFINITION MESSAGE – MSG 439

The Complex Symbol Definition message provides a description of the legs in a Complex symbol, and specifies its Complex symbol ID, which is unique for its stream. If the same symbol ID appears in a different stream, it represents a different Complex symbol.

A leg can be an option or an equity; the leg type is indicated in the LegSecurityType field. The number of leg definitions is provided in the message. The order of legs is deterministic. The length of the message is variable depending on the number of legs in the complex symbol.

On system startup, each Complex feed multicast channel sends all the symbols on its channel. Underlying Index Mapping messages are published first, followed by all Series Index Mapping messages, and lastly Complex feed, Complex Symbol Definition messages.

This message is also sent intraday whenever a new complex symbol is created.

It is published in the Complex feed and is also available via a symbol refresh request.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>Variable, 80 bytes maximum</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>439 – Complex Symbol Definition</li> </ul>
<b>ComplexIndex</b>	4	4	Binary	The unique ID of this complex symbol in this stream. This ID cannot be used to cross reference a security between streams or between markets.
<b>ComplexSymbol</b>	8	21	ASCII	Complex Symbol. This is an alpha numeric symbol which can be used for a unique strategy until one of the legs expire.
<b>ChannelID</b>	29	1	Binary	Multicast channel ID of the symbols being provided.
<b>MarketID</b>	30	2	Binary	Identifies originating market: <ul style="list-style-type: none"> <li>1 (NYSE Cash)</li> <li>2 (Europe Cash)</li> <li>3 (NYSE Arca Cash)</li> <li>4 (NYSE/Arca Options)</li> <li>5 (NYSE/Arca Bonds)</li> <li>6 (ArcaEdge)</li> <li>7 (LIFFE)</li> <li>8 (NYSE Amex Options)</li> <li>9 (NYSE MKT Cash)</li> </ul>
<b>SystemID</b>	32	1	Binary	Identifies Trading unit (TU)
<b>Reserved</b>	33	1	Binary	Filler
<b>StreamID</b>	34	2	Binary	Identifies Stream on which this symbol will be updated
<b>NoOfLegs</b>	36	2	Binary	Number of legs in complex symbol 1 – 5
<b>Reserved</b>	38	2	Binary	Filler

**Leg definition – max of 5**

SymbolIndex	40	4	Binary	Series index if Security type is Option Underlying index if Security type is Equity
LegRatioQty	44	2	Binary	Leg ratio
Side	46	1	ASCII	Leg side <ul style="list-style-type: none"> <li>▪ B (Buy)</li> <li>▪ S (Sell)</li> </ul>
SecurityType	47	1	ASCII	Leg Security Type <ul style="list-style-type: none"> <li>▪ O (Options Series leg)</li> <li>▪ E (Equity stock leg)</li> </ul>

## 6. Control Message Formats

### 6.1 STREAM ID MESSAGE – MSG 455

In all multicast channels, a Stream ID Message is sent immediately after each packet header to uniquely identify the source or Stream ID of the data in the packet. Stream ID Messages are not applicable in TCP request/response connections, and so are not used by either the client or the Request Server.

This message is published in the Top, Deep and Complex feeds.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>8 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>455 – Stream ID Header Message</li> </ul>
<b>StreamID</b>	4	2	Binary	Represents the Stream ID for this packet update. The Stream ID represents a unique source (specific thread within a publisher task).
<b>Reserved</b>	6	2	Binary	Filler

### 6.2 TCP LOGIN REQUEST MESSAGE – MSG 456

This message is sent by clients to log into the Request Server. The server responds with a Login Request Response message, msg type 457.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>28 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>456 – Login Request Message</li> </ul>
<b>SourceID</b>	4	10	ASCII	Name of the source requesting login. This field is up to 9 characters, null terminated.
<b>Password</b>	14	12	ASCII	Password of the requestor
<b>Reserved</b>	26	2	Binary	Filler

### 6.3 TCP LOGIN REQUEST RESPONSE MESSAGE – MSG 457

This message is sent in response to a client request to log into the Symbol Refresh Server. It indicates whether or not the login request has been accepted. In the event the request is rejected, the message indicates the reason for the rejection. A request may be rejected for the following reasons:

- The requestor is not authorized or has an invalid password
- The server has exceeded the maximum connection limit for this port
- The connection has timed out (client connects and does not log in within 30 seconds).

In the event the request is rejected, the Symbol Refresh Server closes the socket connection after sending this message.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>▪ 8 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>▪ 457 – Login Request Response Message</li> </ul>
<b>ResponseCode</b>	4	1	ASCII	<ul style="list-style-type: none"> <li>▪ A (Accepted)</li> <li>▪ B (Rejected)</li> </ul>
<b>RejectCode</b>	5	1	ASCII	<ul style="list-style-type: none"> <li>▪ blank (Accepted)</li> <li>▪ A (Not authorized)</li> <li>▪ M (Maximum server connections reached)</li> <li>▪ T (Timeout)</li> </ul>
<b>Reserved</b>	6	2	Binary	Filler

### 6.4 TCP HEARTBEAT MESSAGE - MSG 12

This message is sent by the Request Server to the client as soon as a TCP connection is established, and is republished approximately every 60 seconds to indicate the session is active.

A responding TCP Heartbeat message from the client system must be received by the server within 5 seconds of sending the heartbeat, or else the session will be terminated.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>▪ 16 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>▪ 12 – Heartbeat</li> </ul>
<b>SourceID</b>	4	10	ASCII	Name of the connected client. This field is up to 9 characters, null terminated.
<b>Reserved</b>	14	2	Binary	

## 6.5 TEST REQUEST MESSAGE – MSG 458

This message can be sent by clients to request a response from the Request Server during periods of inactivity. The client can specify a text message for the server to echo back its response.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>12 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>458 – Test Request Message</li> </ul>
<b>TestMessage</b>	4	8	ASCII	Text to be echoed back

## 6.6 TEST REQUEST RESPONSE MESSAGE – MSG 459

This message is sent back from the Request Server to a client in response to a Test Request message.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>12 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>459 – Test Request Response Message</li> </ul>
<b>TestMessage</b>	4	8	ASCII	Client text to echo from the Test Request message

## 6.7 UNDERLYING INDEX MAPPING REQUEST MESSAGE - MSG TYPE 441

This message is sent by Subscribers via TCP/IP to request Underlying index mapping messages.

If the UnderlyingIndex field in the message is populated with a specific value, the response will contain the Underlying index requested. In this case, any value in the ChannelID field is ignored.

If the UnderlyingIndex field in the message is populated with the value '0', the response will contain all the Underlying indices on the channel defined in the ChannelID field.

If the ChannelID field is populated with the value '0', the request will contain all the Underlying symbols on the Exchange. In this case, any value in the UnderlyingIndex field is ignored.

The Request Server responds to this message with a Request Response Message, to indicate the outcome of the request. If the request is accepted, the Request Response Message is followed by the index mapping messages requested. When the download is complete, a second Request Response Message is sent to indicate that the transmission is finished.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>20 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message: <ul style="list-style-type: none"> <li>441 – Underlying Index Mapping Request Message</li> </ul>
<b>UnderlyingIndex</b>	4	4	Binary	The ID from the Underlying Index message of the requested symbol. To request a refresh for all symbols in the channel, set this field to 0.
<b>SourceID</b>	8	10	ASCII	Name of the requestor.
<b>ChannelID</b>	18	1	Binary	Multicast channel ID of the symbol(s) being requested. To request all underlying symbols in all channels, set this field to 0.
<b>Reserved</b>	19	1	Binary	Filler

## 6.8 SERIES INDEX MAPPING REQUEST MESSAGE - MSG TYPE 443

This message is sent by Subscribers via TCP/IP to request Series index mapping messages.

If the SeriesIndex field in the message is populated with a specific value, the response will contain the Series index requested. In this case, any value in the ChannelID field is ignored.

If the SeriesIndex field in the message is populated with the value '0', the response will contain all the Series indices on the channel defined in the ChannelID field.

If the ChannelID field is populated with the value '0', the request will contain all the Outright options symbol on the Exchange. In this case, any value in the SeriesIndex field is ignored.

The Request Server responds to this message with a Request Response Message, to indicate the outcome of the request. If the request is accepted, the Request Response Message is followed by the index mapping messages requested. When the download is complete, a second Request Response Message is sent to indicate that the transmission is finished.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>20 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>443 – Series Index Mapping Request Message</li> </ul>
<b>SeriesIndex</b>	4	4	Binary	The ID from the Series Index msg of the requested symbol. To request a refresh for all symbols in the channel, set this field to 0.
<b>SourceID</b>	8	10	ASCII	Name of the requestor.
<b>ChannelID</b>	18	1	Binary	Multicast channel ID of the symbol(s) being requested. To request all series symbols in all channels, set this field to 0.
<b>Reserved</b>	19	1	Binary	Filler



## 6.9 COMPLEX SYMBOL DEFINITION REQUEST MESSAGE - MSG TYPE 445

This message is sent by Subscribers via TCP/IP to request Complex symbol definition messages.

If the ComplexSymbolIndex field in the message is populated with a specific value, the response will contain the Complex symbol definition index requested. In this case, any value in the ChannelID field is ignored.

If the ComplexSymbolIndex field in the message is populated with the value '0', the response will contain all the Series indices on the channel defined in the ChannelID field.

If the ChannelID field is populated with the value '0', the request will contain all the Complex Symbols on the Exchange. In this case, any value in the ComplexSymbolIndex field is ignored.

The Request Server responds to this message with a Request Response Message, to indicate the outcome of the request. If the request is accepted, the Request Response Message is followed by the index mapping messages requested. When the download is complete, a second Request Response Message is sent to indicate that the transmission is finished.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>20 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>445 – Complex Index Mapping Request Message</li> </ul>
<b>ComplexSymbolIndex</b>	4	4	Binary	The ID from the Complex Index msg of the requested symbol. To request a refresh for all symbols in the channel, set this field to 0.
<b>SourceID</b>	8	10	ASCII	Name of the requestor.
<b>ChannelID</b>	18	1	Binary	Multicast channel ID of the symbol(s) being requested. To request all complex symbols in all channels, set this field to 0.
<b>Reserved</b>	19	1	Binary	Filler

## 6.10 REQUEST RESPONSE MESSAGE - MSG TYPE 447

This message will be sent via TCP/IP in response to the client's request for symbol mapping messages.

If the request is accepted and the mapping messages provided, this message is sent again immediately at the end of the transmission to indicate that the download is complete.

If the request is rejected, the reason for the rejection is indicated in this message.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Message size: <ul style="list-style-type: none"> <li>16 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>447 –Request Response Message</li> </ul>
<b>SourceID</b>	4	10	ASCII	Name of the source requesting retransmission.
<b>ChannelID</b>	14	1	Binary	Multicast channel ID
<b>Status</b>	15	1	Binary	Outcome of the request. Valid values: <ul style="list-style-type: none"> <li>0 – Request was accepted</li> <li>1 – Rejected due to an Invalid Source ID</li> <li>4 – Rejected due to maximum number of refresh requests in a day</li> <li>6 – Rejected due to an Invalid Channel ID</li> <li>8 – Underlying download complete</li> <li>9 – Series download complete</li> <li>10 – Complex download complete</li> </ul>

## 6.11 TCP LOGOUT REQUEST MESSAGE – MSG TYPE 460

The TCP Logout Request message can be sent by the client to the Request Server as an optional means of disconnecting. The Request Server responds by disconnecting the client.

The other alternative is for the client to simply disconnect when all needed information is received.

Note that if the client fails to respond to any TCP Heartbeat message within five seconds, the server will close the socket.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>4 bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>460 – Logout Request</li> </ul>

## 6.12 SEQUENCE NUMBER RESET - MSG TYPE 1

This message is sent to 'reset' the Sequence Number at start of day or following to a failover. Note that this message will contain a valid sequence number.

The table below describes the body fields of a Sequence Number Reset message, Msg Type 1.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
<b>MsgSize</b>	0	2	Binary	Size of the message: <ul style="list-style-type: none"> <li>16 Bytes</li> </ul>
<b>MsgType</b>	2	2	Binary	Type of message. <ul style="list-style-type: none"> <li>1 – Sequence Number Reset</li> </ul>
<b>SourceTime</b>	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
<b>SourceTimeNS</b>	8	4	Binary	The nanosecond offset from the SourceTime.
<b>ProductID</b>	12	1	Binary	This field is unused for Options. Ignore any content.
<b>ChannelID</b>	13	1	Binary	Multicast channel ID over which the packet was sent.
<b>Reserved</b>	14	2	Binary	Filler

## 7. Symbol Mapping Download File

---

For customers who would prefer to download the symbol index mapping information from an FTP server, a file containing the information is made available via FTP at 6:00am ET every trading day at the following locations. Note that options symbols can be added during the trading day. The download file is not updated intraday to keep up with these changes.

### 7.1 FTP FILE LOCATION

FTP files are available every trading day starting November 9, 2015 at

<ftp://ftp.nyxdata.com/ArcaAmexOptionsSymbolMapping>

This directory contains four files, for:

Amex production, Amex certification environment

Arca production, Arca certification environment

### 7.2 FTP FILE FORMAT

All FTP files are in the same pipe-delimited format. The file contains 3 different row types, corresponding to the 3 types of mapping messages:

Symbol mapping	MsgType field = 435
Series mapping	MsgType field = 437
Complex series mapping	MsgType field = 439

#### 7.2.1 Underlying mapping record format (MsgType = 435)

```
MsgType|UnderlyingIndex|UnderlyingSymbol|MarketID|SystemID|
ExchangeCode|PriceScaleCode|SecurityType|PriceResolution|
TopFeedChannelID|DeepFeedChannelID|ComplexFeedChannelID
```

##### Sample

```
435|2872|YANG|4|14|P|4|E|5|31|67|121
435|2873|YCS|4|14|P|4|E|5|31|67|121
435|2874|YELP|4|14|N|4|C|5|31|67|121
```

#### 7.2.2 Series mapping records format (MsgType = 437)

```
MsgType|StreamID|SeriesIndex|MarketID|SystemID|UnderlyingIndex|
ContractMultiplier|MaturityDate|PutOrCall|StrikePrice|PriceScaleCode|
UnderlyingSymbol|OptionSymbolRoot|GroupID
```

##### Sample

```
437|225|31717725|4|14|2872|100|160115|1|30|4|YANG|YANG|143601
437|225|31717726|4|14|2872|100|160115|1|35|4|YANG|YANG|143601
437|225|31717727|4|14|2872|100|160115|1|40|4|YANG|YANG|143604
```

### 7.2.3 Complex series mapping record format (MsgType = 439):

MsgType|StreamID|ComplexIndex|ComplexSymbol|MarketID|SystemID|NoOfLegs

|SymbolIndex|LegRatioQty|Side|SecurityType <repeat for each leg>

The last four fields in red contain information for a single leg and repeat as many times as the number of legs.

#### Sample

439|227|31731777|4YOKU15289247|4|14|2|31722253|1|S|O|31722254|1|B|O

439|227|31731778|4YOKU15289373|4|14|2|31722643|1|S|O|31722644|1|B|O

439|227|31731779|4YHOO153041|4|14|2|31720592|1|S|O|31721006|1|S|O