NYSE GLOBAL INDEX FEED

EXCHANGE DATA PUBLISHER (XDP) CLIENT SPECIFICATION

Version	Date
1.9c	14 Jun 2016

© 2016 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 June 14, 2016. NYSE disclaims any duty to update this information

PREFACE

DOCUMENT HISTORY

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

VERSION NO.	DATE	CHANGE DESCRIPTION
1.0	04/21/10	Initial version.
1.1	02/11/10	Feed renamed from 'NSYE Arca Index Service' to 'NYSE Euronext Global Index Feed'.
1.2	02/18/11	The following changes were made: ■ Added SymbolSeqNum data item to MSG TYPE '34' ■ Updated Table 16 Control Message Types ■ Deleted Sourcetime Reference Message (MSG TYPE '2') ■ Added Section 3.8 Refresh Request (MSG TYPE '15') ■ Added Section 3.13 Message Unavailable Message (MSG TYPE '31') ■ Added Section 3.15 Refresh Header (MSG TYPE '35') ■ Added Production Channel Configuration Table ■ Added Test Channel Configuration Table
1.3	04/13/11	The following changes were made: ■ Updated the description of the 'Value' field in Table 13 (ETP Update Message Fields - Message 131) and changed the Size (Byte) value to 8. ■ Updated Table 24 (Symbol Index Mapping Message Fields - Message Type 3) with high-lighted descriptions specific to GIF: – Exchange Code 'P' (NYSE Arca) – Security Type 'E' (EFT) – Security Type 'X' (Index) ■ Replaced references to GMT with EST
1.4	06/29/11	The following changes were made: ■ Updated descriptions of EstCash and TotalCash in Table 14 (ETP Value Message Fields) ■ Added Section 2.1.4 (Notes on Symbol Dissemination) ■ Added Section 1.2.2 ('Interval' and 'Tick-by-Tick' Dissemination) ■ Added section in Preface called "Important Notes about this Specification" ■ Added publication time (12.30 EST) to the description of Msg Type 3. ■ Made numerous edits throughout the document to clarify sections on

VERSION NO.	DATE	CHANGE DESCRIPTION
		Refresh and Retransmission ■ Updated descriptions of DeliveryFlag fields with note that "17, 18, 19 and 20 are for the interval service only and indicate the total number of refresh packets being sent for that update". ■ Updated Table 11 (Message Publication Times) and Table 30 (Production Event Schedule) ■ Changed references to "ETF" to "ETP" or "ETP IOPV".
1.5	08/17/11	Clarified Production hours in Sections 2.1.2 and 4.2 Replaced references to "UTP-MD" with "XDP"
1.6	05/18/2012	Included references to NYSE MKT throughout
	08/09/2012	Rebranded with new NYSE Technologies template
1.7	10/22/2012	Updated the times in Table 30 (Production Event Schedule)
1.8	11/10/2014	Rebranded to ICE / NYSE template and remove Euronext from name
1.9	07/09/2015	Extend feed production time from 20:15 ET to 23:00 ET; Service Desk contact information update
1.9a	12/01/2015	Change feed production time to 7:45pm ET (19:45) to 7:00pm ET (19:00) next day EST; effective February 7, 2016.
1.9b	01/11/2016	Change feed production time to 7:45pm ET (19:45) to 7:15pm ET (19:15) next day EST; effective February 7, 2016.
1.9c	06/14/2016	Change feed production time to an earlier open of 6:00pm ET (18:00) on Sunday start up; existing production time of 7:45pm ET (19:45) to 7:15pm ET (19:15) next day EST for the remainder of the week; effective August 7, 2016.

REFERENCE MATERIAL

The following lists the associated documents, which either should be read in conjunction with this document or which provide other relevant information for the user:

- SFTI US Technical Specification
- SFTI US Customer Guide

FURTHER INFORMATION

- For additional product information, visit: http://www.nyxdata.com/nysedata/Default.aspx?tabid=1000
- For updated capacity figures, visit our capacity pages at: http://www.nyxdata.com/capacity
- For details of IP addresses, visit our IP address pages at: http://www.nyxdata.com/ipaddresses
- For a full glossary, visit: http://www.nyxdata.com/glossary/

CONTACT INFORMATION

Service Desk

- Telephone: +1 212 896 2830 (Option 4)
- Email: service.desk@nyse.com

IMPORTANT NOTES ABOUT THIS SPECIFICATION

Refresh and Retransmission

Information on Retransmission and Refresh functionality is provided in this specification that denotes future implementations only. *TCP/IP functionality is currently not supported*. This release of GIF supports:

- Packet retransmissions for Symbol Mapping data *only*
- Interval dissemination of data which includes refresh as each message update is sent

Although not supported currently, in a future release, clients will be able to connect to a TCP/IP server to request retransmission and refresh data.

Interval and Real-Time Tick-by-Tick Dissemination Frequencies

The GIF Client Specification is common to interval and real-time tick-by-tick dissemination frequencies of the GIF service channels. There are additional features specific to supporting real-time tick-by-tick that are not utilized by the interval service. When clients are programming to the feed, they should keep in mind that both interval and real-time tick-by-tick are possible when interpreting the areas of the specification applicable to each of these service channels.

Areas Highlighted in Yellow

In this specification, areas that are highlighted in yellow denote areas of the XDP feed that are specific to the NYSE Global Index Feed product.

CONTENTS

1. NYSE MARKET DATA PROCESSING INFORMATION

1.1 INTRODUCTION

The NYSE Global Index Feed (GIF) supports advanced trading methodologies and dissemination frequency required for today's real-time index and exchange-traded product calculations. GIF is available via the Exchange Data Publisher (XDP) feed. XDP is a high-speed real-time multicast data feed, offering ultra low-latency publishing.

Data and features of GIF include:

- Real-time publication of NYSE index data, including:
  - NYSE, NYSE Arca, NYSE MKT and other indices (for example, fixed income, strategy)
  - Third-party index values (for example, DB Liquid Commodity, Wisdom Tree, Wilder Hill New Energy)
- Exchange-traded product (ETP) data, including:
  - Segmented ETF Indicative Optimized Price Value (IOPV) values and static portfolio data
  - Intraday Portfolio Values (IPVs)
  - Daily valuation information, including: Net Asset Value (NAV), shares outstanding, estimated cash per-unit creation, total cash per-unit creation, and dividends
- Data distribution for ETF IOPV and other exchange-traded products
- Real-time sub-second tick index data for select index and exchange-traded product valuations
- Broader distribution framework for multipurpose real-time valuation use
- Direct access to index and ETP data from the source via NYSE Technologies' Secure Financial Transaction Infrastructure® (SFTI®)

Data on GIF is generated by the NYSE Global Index Group which is a full-service index provider dedicated to the design and calculation of market and custom indices that meet the demands of increasingly changing markets.

The NYSE Global Index Group creates custom indices as the basis of ETP and other exchange-traded products fully supported with real-time index, IOPV and portfolio-dissemination services. In some cases, data on the GIF feed is generated by third-parties and routed to NYSE for distribution on GIF.

GIF is based on the same technical platform as the NSYE Arca OpenBook. The differentiating message structure of GIF is as follows:

- **Message 130** Provides index intraday real-time updates; see Index Update Message (Msg Type '130')
- **Message 131** Provides IOPV intraday real-time updates; see ETP Update Message (Msg Type '131')
- **Message 132** Provides start-of-day static data, and less often intraday updates, for net asset value, shares outstanding, estimated cash, total cash and dividends; see ETP Value Message (Msg Type '132')
- **Message 133 (Not currently disseminating on GIF)** Provides index start-of-day information and dividend information; see Index Value Update Message (Msg Type '133')

GIF supports both 'Interval' and 'Tick-by-Tick' dissemination frequencies – see 'Interval' and 'Tick-by-Tick' Dissemination.

1.2 NYSE MARKET DATA OVERVIEW

The XDP feed provides high-speed real-time market data to the NYSE Global Index Feed for U.S.-traded indices and exchange-traded products.

The data feed has the following high-level features:

- Multicast technology
- High Availability
- Ultra-low latency
- Reliable network solution
- High level of scalability

This chapter provides detailed information about the features of the feed, to support the development of client applications by Traders, Independent Software Vendors and Quote Vendors.

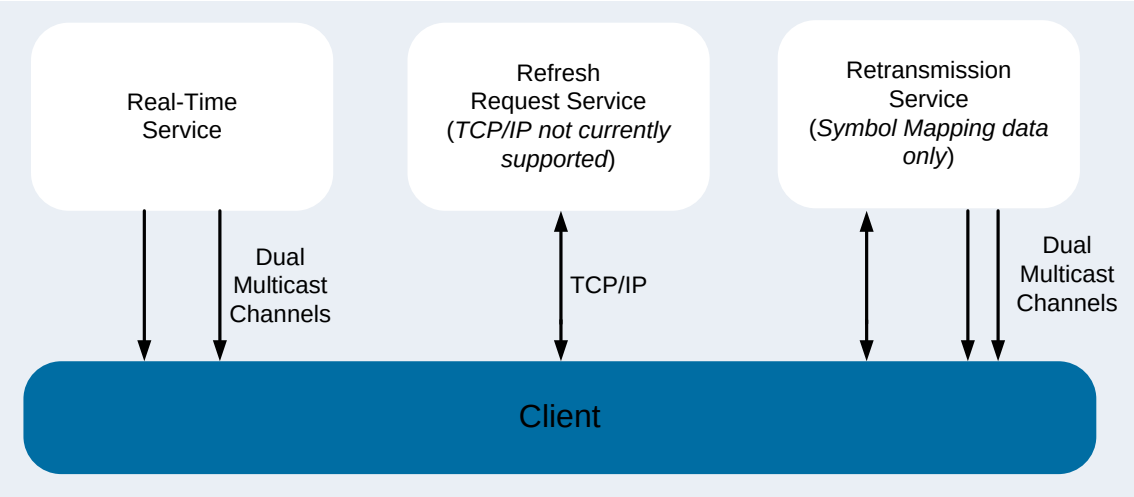
The following chapters of this document provide details that are specific to each of the NYSE Global Index market data sets, including formats for each message type.

1.3 ACCESS TO MARKET DATA

Clients connect to multicast addresses for the real-time market data messages. GIF currently supports interval dissemination of data which includes refresh as each message update is sent. GIF also supports packet retransmissions for Symbol Mapping data *only*.

Note: Although not supported currently, in a future release, clients will be able to connect to a TCP/IP server to request retransmission and refresh data.

Figure 1 Access to Market Data



1.3.1 Real-Time Market Data

Real-time market data is message-based over the UDP IP protocol with fixed length binary and ASCII fields.

It uses the push-based publishing model. This means that data will be published based on its availability. Once an update is available, it will be published to the appropriate multicast group.

For capacity reasons, market data can be split across a number of multicast groups organized into predefined data sets (channels). The number of channels can be changed in the future with notice.

Each multicast group will deliver a set of data for a certain market segment.

The client application will be responsible for issuing multicast subscriptions to one or more of the multicast groups assigned to each product.

The process of subscribing to a multicast group ID is also known as 'joining' a multicast group. Upon session termination, the client's host system should issue an "unjoin" message. This will terminate delivery of data to that host's local network. If a client application terminates without issuing an "unjoin" message, the network will eventually issue a 'timeout' for the multicast group subscription that will automatically terminate delivery of the multicast packets to the host's local network.

The 'join' and 'unjoin' processes are standard functions. No specific instructions are provided here, as they are specific to the user's operating system and programming language.

1.3.2 'Interval' and 'Tick-by-Tick' Dissemination

Both 'Interval' and 'Tick-by-Tick' designations are considered real-time dissemination:

- **Interval** This service operates at consistent 15-second intervals, which is current industry standard.
- **Tick-by-Tick** This is an advanced service that disseminates an update value each time that an underlying price changes within the underlying portfolio.

1.3.3 Retransmission Functionality (For Future Implementation)

Note: GIF supports packet retransmissions for Symbol Mapping data *only*. Although not supported currently, in a future release, clients will be able to connect to a TCP/IP server to request retransmission data.

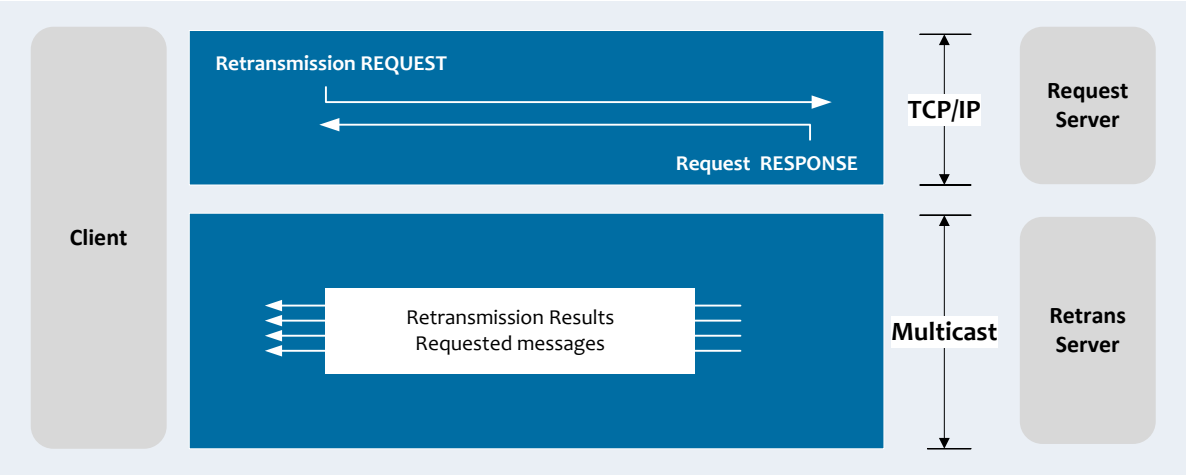
The retransmission functionality is designed to allow the user to recapture a small number of missed messages.

It is not intended that clients use the retransmission functionality to recover data after long outages or on late start up. Accordingly, the number of messages that the user can request by each Source ID is limited. The number of retransmission requests permitted per user is also limited per day.

The client makes a TCP/IP connection with the Retransmission Server, and receives the requested messages also via the multicast channel.

The following diagram shows the sequence of messages and the transport protocols employed when making a retransmission request.

Figure 2 Retransmission Request



The retransmission request will include a Source ID (username), which will be validated by the Retransmission Server. It is important to note that only one Source ID can be used per application session.

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

- Invalid Source ID (username)
- Invalid requested sequence numbers
- Incorrectly formatted request packet
- Maximum number of packets by request exceeded
- Messages are no longer in cache
- Total number of messages requested in the current day exceeds the predefined system limit
- Number of retransmission requests in the current day exceeds the predefined system limit

In the case of such a failure, the user will receive an error message to advise of the reason for failure.

If the reason for failure is exceeding a predefined system limit, clients are asked to not make any further requests. If further retransmissions are required, the client should contact the Service Desk.

1.4 PROCESSING GUIDELINES

1.4.1 General Processing Notes

The following processing notes apply to all messages:

- All fields will be sent for every message
- Only field values will appear in the published messages (for example, no names or 'tags' will appear in the message)
- 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 and constant
- The Message sizes may vary
- Binary fields are provided in **Little-Endian format**

- ASCII string fields are left aligned and null padded
- Segmentation of messages across packets will not be supported. This means a message will never straddle a packet boundary.

1.4.2 FAST Optimization

FAST optimization will **not** be used for all products (i.e. NYSE Global Index Feed). If a data feed is offered in FAST optimized format, it will be provided over a different set of multicast IP addresses/ports.

1.4.3 Packet Structure

All packets of data sent on the XDP feed will have a common packet header followed by one or more messages (with the exception of some technical packets that do not contain any messages).

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

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

The maximum length of a packet is 1500 bytes.

A packet will only ever contain complete messages. A single message will never straddle multiple packets.

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

PACKET HEADER	MESSAGE 1	MESSAGE 2	...	MESSAGE N

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

Table 1 Packet Header Fields

FIELD	OFFSETS	SIZE (BYTES)	FORMAT	DESCRIPTION
PktSize	0	2	Binary Integer	This field indicates the size of the packet including the 16 -byte packet header in bytes
DeliveryFlag	2	1	Binary Integer	A flag that indicates whether this is an original, retransmitted, or 'replayed' message. Valid values are provided below. **Note**: Some of these values are associated with the tick-by-tick data (1, 11 and 12). Also, 17, 18, 19 and 20 are for the interval service only and indicate the total number of refresh packets being sent for that update. ■ '1' – Heartbeat Message ■ '11' – Original Message ■ '12' – Sequence Number Reset Message

FIELD	OFFSETS	SIZE (BYTES)	FORMAT	DESCRIPTION
				■ '13' – Only one packet in retransmission update Uncompressed ■ '14' – Start of retrans Update Uncompressed ■ '15' – Part of a retransmission sequence Uncompressed ■ '16' – End of retrans Update Uncompressed ■ '17' – Only one packet in Refresh update Uncompressed ■ '18 – Start of Refresh Update Uncompressed ■ '19' – Part of a Refresh sequence Uncompressed ■ '20' – End of Refresh Update Uncompressed ■ '21' – Message Unavailable ■ '31' – Original Message compressed (Fast) ■ '32' – Sequence Number Message (Fast) ■ '33' - Only one packet in retransmission update compressed (Fast) ■ '34' – Start of retrans Update compressed (Fast) ■ '35' – Part of a retransmission sequence compressed (Fast) ■ '36' – End of retrans Update compressed (Fast) ■ '37' – Only one packet in Refresh update compressed (Fast) ■ '38' – Start of Refresh Update compressed (Fast) ■ '39' – Part of a Refresh sequence compressed (Fast) ■ '40' – End of Refresh Update

FIELD	OFFSETS	SIZE (BYTES)	FORMAT	DESCRIPTION
				compressed (Fast) ■ '41' – Message Unavailable
NumberMsgs	3	1	Binary Integer	This field contains the number of messages in the packet and also used to determine the next sequence number, see Sequence Numbers for more information.
SeqNum	4	4	Binary Integer	This field contains the message sequence number assigned by XDP for each product. It is used for gap detection, unique for each broadcast stream (except if reset during the day), see Sequence Numbers for more information.
SendTime	8	4	Binary Integer	This field specifies the time when the message was generated in the order book. The number represents the number of seconds in UTC time (EPOCH).
SendTimeNS	12	4	Binary Integer	This field specifies the number represents the nanosecond portion of UTC time (since EPOCH).

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

However, regardless of the message type, each message will start with a message header consisting of 2 fields: a 2-byte message length, followed by a 2-byte message type.

Table 2 Message Header Fields

FIELD	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary Integer	This field indicates the size of the message body in bytes including this field.
MsgType	2	2	Binary Integer	This field identifies the type of message

1.4.4 Message Size Field Processing

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

For example: The following table demonstrates a message type used by NYSE, NYSE MKT and NYSE Arca. Each market shares the same fields until byte 28. If you were taking only the NYSE Arca version of the feed, you would read "28 bytes" in the Msg Size field, and the next message will begin on "Byte 29". If you were reading an NYSE message, then the Msg Size field will indicate that the next message begin after the 35th byte.

Table 3 Message Size Field Processing

FIELD	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary Integer	Size of the message. **NYSE – 34 Bytes** **NYSE MKT – 34 bytes** **NYSE Arca - 28 bytes**
Msg Type	2	2	Binary Integer	This field identifies the type of message. Message '100' – Add Order Message
SourceTimeNS	4	4	Binary Integer	This field represents the nanosecond offset from the time reference second in UTC time (EPOCH).
SymbolIndex	8	4	Binary Integer	This field identifies the numerical representation of the symbol.
OrderID	12	4	Binary Integer	The Order ID identifies a unique order.
Price	16	4	Binary Integer	This field specifies the price of the order, see Price Formats. Use the Price scale from the symbol mapping index.
Volume	20	4	Binary Integer	This field contains the size of the order.
Side	24	1	ASCII Character	This field indicates the side of the order Buy/Sell. Valid values: ■ 'B' – Buy ■ 'S' – Sell
OrderIDGTCIndicator	25	1	Binary Integer	This field specifies if Trade Order ID is a GTC order: ■ '0' – Day Order ■ '1'- GTC Order
TradeSession	26	1	Binary Integer	Bit Shift values:

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

FIELD	OFFSET	SIZE	FORMAT	DESCRIPTION
				0x01 Ok for morning hours 0x02 Ok for national hours (core) 0x04 Ok for late hours
QuoteCondition	27	1	Binary Integer	The current quote condition for the symbol. The quote condition shall be blank if no quote condition exists (example when the Book is fast).
AggregatedVolume	28	4	Binary Integer	This field is the Total Volume at the Price Point after the event has been applied.
NumOrders	32	2	Binary Integer	This field contains the number of orders at the current price point.

Market-specific content

The variable message size allows that the feed handler can also insulate your code from any future field additions that you may not want.

For example, the original format had the following 16 byte message:

Table 4 Example 1

FIELD	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary Integer	Size of the message. NYSE – 16 Bytes NYSE MKT– 16 bytes NYSE Arca - 16 bytes
Msg Type	2	2	Binary Integer	This field identifies the type of message Message '999' – Price message example
SourceTimeNS	4	4	Binary Integer	This field represents the nanosecond offset from the time reference second in UTC time (EPOCH).
SymbolIndex	8	4	Binary Integer	This field identifies the numerical representation

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

FIELD	OFFSET	SIZE	FORMAT	DESCRIPTION
				of the symbol.
Price	12	4	Binary Integer	This field specifies the price of the order see Price Formats. Use the Price scale from the symbol mapping index.

Now the new format adds a new 4 byte volume field adjusting the Msg Size to 20 bytes.

The feed handler code automatically is prepared for the 20 byte format without any work, allowing for the receiving application to either continue to read on the first 16 bytes passed to it or develop to read the new field at your own pace.

Table 5 Example 2

FIELD	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary Integer	Size of the message. NYSE – 20 Bytes NYSE MKT– 20 bytes NYSE Arca – 20 bytes
Msg Type	2	2	Binary Integer	This field identifies the type of message Message '999' – Price message example
SourceTimeNS	4	4	Binary Integer	This field represents the nanosecond offset from the time reference second in UTC time (EPOCH).
SymbolIndex	8	4	Binary Integer	This field identifies the numerical representation of the symbol.
Price	12	4	Binary Integer	This field specifies the price of the order, see Price Formats. Use the Price scale from the symbol mapping index.
Volume	16	4	Binary Integer	This field contains the size of the order.

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

1.4.5 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.

XDP uses the concept of a time reference that identifies the whole number of seconds in UTC time, and each data message contains the nanosecond offset from that time reference value.

Each matching engine will send a Time Reference Message (Msg Type '2') every x number of seconds containing the matching engine system ID to identify the source timestamp.

The XDP System itself will send only a packet header Send Time that is in UTC time that does not require a time reference message.

Note: Nanosecond offsets are not applicable to the Index Service Feed data messages; all index service messages include a source timestamp that does not require a time reference message.

1.4.6 Product IDs

Table 6 Product IDs

EXCHANGE	PRODUCTID	DESCRIPTION
NYSE Arca	156	Index Service feed

1.4.7 Sequence Numbers

All messages conform to the message level sequencing. Each channel A, B, C, D and so forth shall have its own sequence number. This will allow recipients to identify "gaps" or duplicates in each message sequence number and, if appropriate, reconcile them (line arbitrage) with the primary/secondary multicast groups or request retransmission of the missing/corrupted data packets.

Clients can use sequence numbers to determine the following:

- Missing (gapped) packets
- Unordered packets
- Duplicate packets

Table 7 Calculating Sequence Numbers

The next SeqNumbers are calculated by adding the current SeqNum+NumMsgs:

FIELD NAME	SEQNUM	NUMMSGS
Message 1	1	4
Message 2	5	2
Message 3	7	1
Message 4	8	3
Message 5	11	1

If the client drops the first five packets they would request a gap fill for messages 1-11. Instead of sending the original five packets we would send one packet with 11 messages (assuming the total number of messages fit within the 1500 byte packet size, otherwise the delivery flag will indicate that the update will span multiple packets).

1.4.8 Line Arbitration

Client applications should check the Sequence Number (SN) for every packet received.

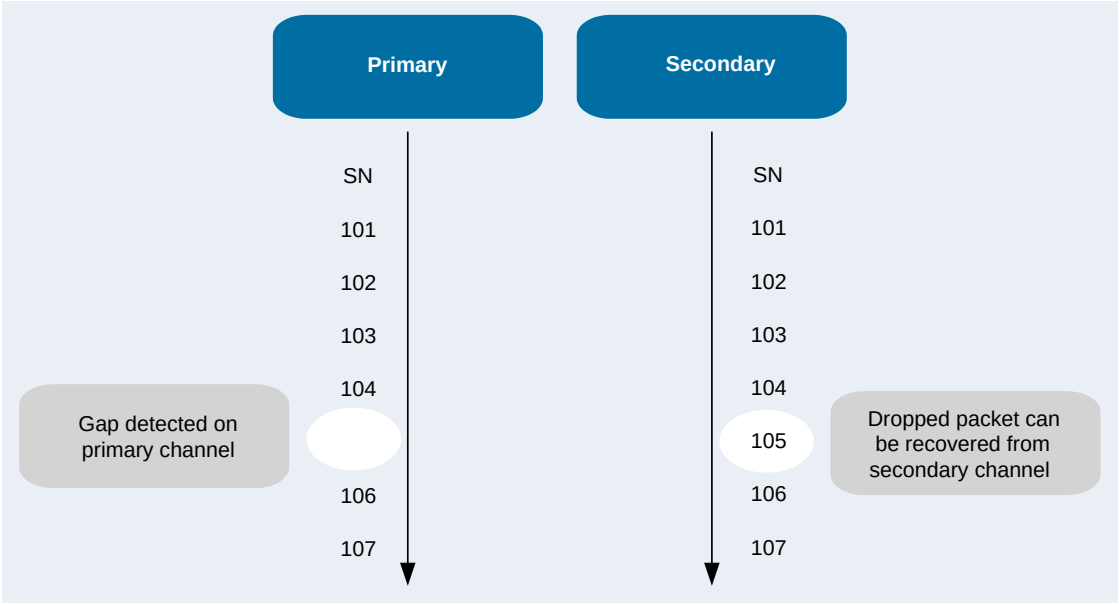
SNs are unique for each channel however they do not increase monotonically. The sequence numbers increase as shown in the example above.

Line A and line B are identical in terms of:

- Packet contents
- SNs
- Sequence in which packets are sent

Client applications should listen to both channels in real-time. Clients should look at packets coming from both lines and process the ones that arrive first, regardless of whether they came from line A or line B. It is advisable to apply the 'first come – first served' rule.

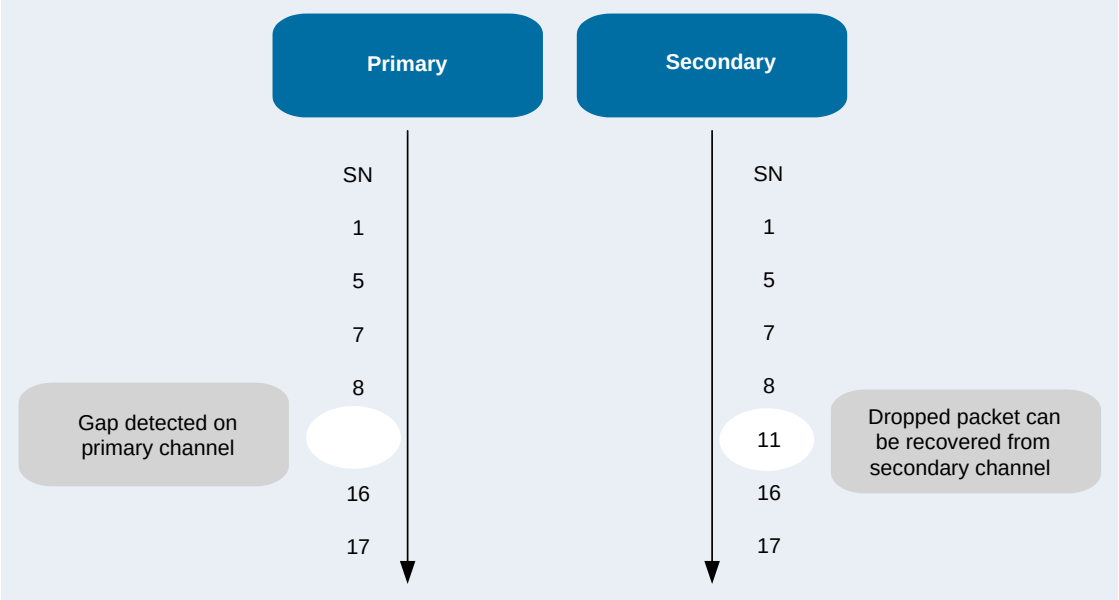
Figure 3 Packet Header Sequence Numbers (assuming one message per packet)



Assuming the message sequence below:

FIELD NAME	SEQNUM	NUMMSGS	NEXT EXPECTED SEQNUM
Message 1	1	4	5
Message 2	5	2	7
Message 3	7	1	8
Message 4	8	3	11
Message 5	11	1	12
Message 6	12	4	16
Message 7	16	1	17

Figure 4 Detecting Missed Packets



1.4.9 Detecting and Recovering Missed Data

UDP is an 'unreliable' protocol and therefore may drop packets. All multicast data is provided over dual channels (line A and line B).

The GIF feed provides the following mechanisms for recovering missed data:

- Line arbitration – Using dual multicast channels
- Retransmission Server – Recovery of a limited number of packets (future enhancement); Currently GIF supports packet retransmissions for Symbol Mapping data *only*.
- Refresh Server – Snapshot of the current market state. Currently GIF supports interval dissemination of data which includes refresh as each message update is sent.

Note: Although not supported currently, in a future release, clients will be able to connect to a TCP/IP server to request retransmission and refresh data.

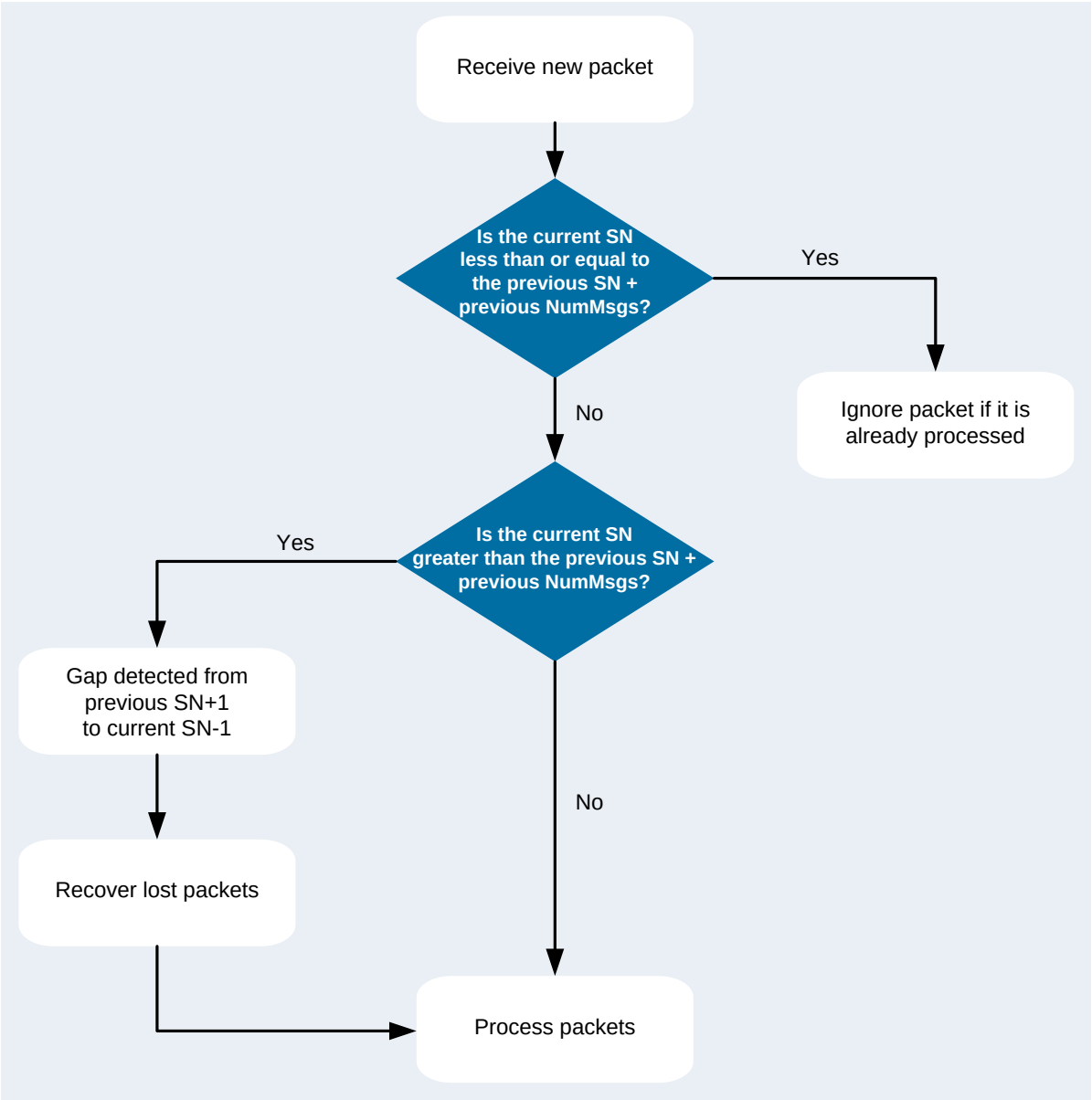
These mechanisms should be used as follows:

Table 8 Recovery Mechanisms

EVENT	ACTION
Packet lost on one of the two lines	Try to recover data from the other line with a configurable timeout
Dropped packet(s) on both line A and line B	Recover dropped packet(s) from Retransmission Server
Late start up or extended intraday outage	Request a refresh of the current market state and then continue with real time messages

The following diagram illustrates how the SN should be used to detect gaps in the feed:

Figure 5 Gap Detection



1.4.10 Retransmission Server (Future Implementation)

If a packet is lost from both line A and line B, clients then make a TCP/IP request to have the packets resent. Packets are resent from the Retransmission Server.

After a client establishes a TCP/IP connection, the Retransmission Server will periodically send heartbeat request messages to the client. Clients must respond to this request with a heartbeat response within a specific timeframe – otherwise, the Retransmission Server will close the connection.

The client makes a TCP/IP connection to the Retransmission Server for both requesting and receiving retransmitted packets (only applies to Symbol Index mapping – to be available in a future release).

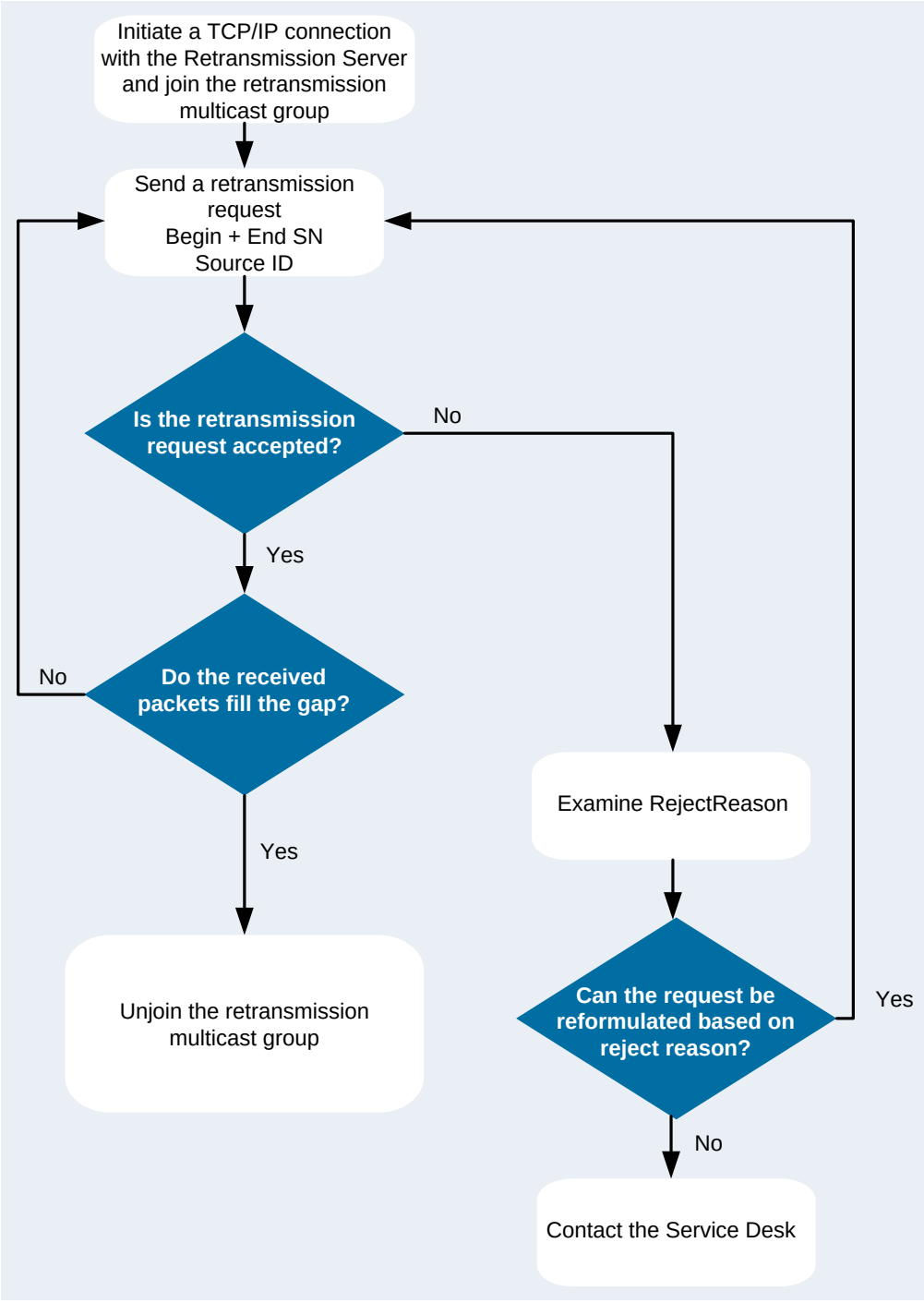
Retransmission requests should contain a Start SN, an End SN and a Source ID. The Source ID identifies the client application, and will be supplied by the exchange. The request can be rejected for a number of reasons, see Request Response Message (Msg Type '11').

The number of retransmissions allowed per client per day is limited see Retransmission/Refresh Request Limitations for detailed content of Retransmission Server limitations.

The length of each retransmission is limited to a pre-defined number of messages.

The following diagram illustrates the process of requesting dropped packets from the Retransmission Server:

Figure 6 Requesting Retransmission of Dropped Packets



1.4.11 Price Formats

All price fields are sent in integer format.

Prices in this feed are represented by two fields, separating the denominator and the numerator. All prices in the feed share a common denominator (unless otherwise specified), which is represented in the PriceScaleCode.

The PriceScaleCode field value represents the common denominator for the following formula:

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

For example, a price of 27.56 is represented by a Numerator of 2756 and a PriceScaleCode equals to 2.

1.4.12 OrderIDs

The OrderID will consist of multiple data values; the OrderID, MarketID, SessionID, and OrderBit. Depending upon the host bytes alignment the data structure will be different.

The OrderID will be a value of a 64 bit long. To convert it in a 32 bit processing environment the following example can be used. Note compaction will use the below structure. To byte align binary fields in a little-endian system all 8 bytes must be aligned as a 64 bit long.

Table 9 Big-Endian System

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
OrderID	0	4	Binary Integer	Order ID
Market ID	4	2	Binary Integer	ID of the Originating market in the Symbol Index Mapping
System ID	6	1	Binary Integer	ID of the Originating System in the Symbol Index Mapping
GTCIndicator	7	1	Binary Integer	This field specifies if Order ID is a GTC order: ■ '0' – Day Order ■ '1' – GTC Order

Table 10 Little-Endian System

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
GTCIndicator	7	1	Binary Integer	This field specifies if Trade Order ID is a GTC order: ■ '0' – Day Order ■ '1' – GTC Order
System ID	6	1	Binary Integer	ID of the Originating System in the Symbol Index Mapping
Market ID	4	2	Binary Integer	ID of the Originating market in the

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
				Symbol Index Mapping
OrderID	0	4	Binary Integer	Order ID

1.4.13 Symbol Mapping

To ensure high throughput and low latency, symbols are identified using a Symbol Index message (Msg Type '3'). This is an ordered list from 1 to N of all symbols per multicast group. Symbol Indices are unique for every symbol and do not change each trading day. New symbols are appended to the end of the symbol-mapping index and symbols that are removed do not have their index number reused.

1.5 OPERATIONAL INFORMATION

The following measures are in place to safeguard against unexpected system failures.

1.5.1 Exchange System Failure

1.5.1.1 Dual Multicast Lines

Under normal operating conditions, the exchange system will send real-time messages to two unique multicast addresses. This provides clients with two redundant data feeds. The client application should be designed to handle the loss of one of the two multicast channels without any interruption to service.

1.5.1.2 TCP/IP Channels (Future Implementation)

TCP/IP channels are made available for retransmission and refresh requests and responses.

The user can choose to disconnect/reconnect in between requests. However if choosing to remain connected, the user will need to respond to heartbeat requests from the exchange.

1.5.1.3 High Availability

The High Availability (HA) functionality of the market data publisher is set up to ensure there is no loss of service for clients if there is any kind of outage in the exchange on the primary publisher, for example a hardware failure. The HA failover has been designed to be as transparent as possible for clients, as the connectivity in terms of multicast groups and ports will not change. However, clients should note that there are specific technical details that should be considered.

For details of retransmissions and refresh behavior that should be included as part of application logic, see Retransmission Message Processing.

1.5.2 Disaster Recovery Site

In order to mitigate any serious outage in the primary data center, a secondary data center is online in standby mode, in case of a serious incident.

Clients should ensure all configurations surrounding the secondary data center are included as detailed in Retransmission/Refresh Request Limitations.

1.5.3 Client System Failure

Real-time market data will be made available on two different multicast groups. This offers clients the possibility to set up more than one receiving system processing the same data. In the event of a client system failure, the backup client system should continue to process the real-time data sent on the second multicast group.

1.5.4 Gap Detection

The XDP feed provides a unique, sequential packet sequence number for each multicast channel. This will allow recipients to identify 'gaps' in the message sequence and, if appropriate, reconcile them 'locally' with an alternate channel or request retransmission of the missing/corrupted data packet. Refer to Detecting and Recovering Missed Data for more details on gap detection.

1.5.5 System Behavior on Start and Restart

At the start of the day, the feed will send the following messages:

- Ten heartbeat messages with Delivery Flag set to 1, sequence number is set to 1 (next expected seqnum)
- Sequence Number Reset message (Msg Type '1'), sequence number is set to 1
- Symbol Mapping messages for securities traded on the market; (Msg Type '3')

Note that this sequence will also be followed on system recovery following a failure. Therefore, in exceptional circumstances a user may see this during the trading day.

2. NYSE GLOBAL INDEX FEED MESSAGE SPECIFICATIONS

2.1 DATA FEED INFORMATION

2.1.1 Overview

The NYSE Global Index Feed is a proprietary index feed which publishes IOPVs and indices from internal calculation systems as well as those passed through from market data vendors and service providers.

2.1.2 Publication Times

Table 11 Message Publication Times

MSGTYPE	DESCRIPTION	NYSE ARCA
130	Index update message	7:45pm – 7:15pm EST
131	ETP update message	7:45pm – 7:15pm EST
132	ETP value message	7:45pm – 7:15pm EST
133	Index value message	7:45pm – 7:15pm EST

Note: Refer to **Table 30 Production Event Schedule** for details pertaining to system behavior prior to 7:45pm and on Sunday (ET) start up time.

2.1.3 Control Message Types

For details, see Control Message Specifications.

2.1.4 Notes on Symbol Dissemination

On GIF, the core symbol along with a numerical SymbolIndex field is the only code with associated data delivered within designated fields within the real-time messages.

For example, currently for the IOPV iShares S&P Growth Allocation Fund, five symbols represent the data as follows:

- AOR.IV – Real-time IOPV value (message 131)
- AOR.EU – Estimated cash per-unit creation value for the ETF IOPV (message 132)
- AOR.NV – Net asset value of the ETF IOPV (message 132)
- AOR.SO – Shares Outstanding of the ETF IOPV (message 132)
- AOR.TC – Total Cash of the ETF IOPV (message 132)

On GIF, only 'AOR' is presented as the core symbol with the numerical SymbolIndex and data values populated within specific fields in the message 132 structure.

Note the following points:

- In general, index symbols publish to GIF as they are – including, or not including, the suffix.
- IOPV symbols publish to GIF but drop the suffix. For example, the IOPV for AFK publishes without the ".IV" suffix and the data items publish only against AFK, with the associated populating the other fields within the 132 message.
- Suffixes are dropped for the 131 and 132 messages.
- There are some non-standard ETF IOPV-related symbols:

  - Roots different from product
  - Suffixes other than .IV, .NV, .SO, .EU or .TC

- Some 130 index messages contain ETF IOPV-related data. However the symbol in a 130 message will be in its entirety; no suffix needs to be added. Additionally, the symbol in 131 and 132 messages will be the root and the requisite suffix (.IV, .NV, .SO, .EU, .TC) should be added based on the position of the data in the message.

2.1.5 Index Update Message (Msg Type '130')

2.1.5.1 Message Overview

An Index Update message provides information about the value updates for Index.

2.1.5.2 Message Sending Rules

An Index Update message 130 message is sent as a result of one of the following events:

- When the value of the Index message changes due to a change in Bid/Ask or last sale of an underlying index component

2.1.5.3 Message Structure

The table below describes the body fields of an Index Update message (MsgType '130'). (See also Notes on Symbol Dissemination.)

Table 12 Index Update Message Fields

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
Msg Size	0	2	Binary Integer	Size of the message. 24 Bytes
Msg Type	2	2	Binary Integer	This field identifies the type of message '130 – Index Update Message
SourceTime	4	4	Binary Integer	This field specifies the time when the message was generated in the order book. The number represents the number of seconds in UTC time (EPOCH).
SourceTimeNS	8	4	Binary Integer	This field specifies the number represents the nanosecond portion of UTC time (since EPOC)
SymbolIndex	12	4	Binary Integer	This field identifies the numerical representation of the symbol. See Symbol Mapping
Value	16	8	Binary Integer	This field contains the corresponding value of the symbol

2.1.6 ETP Update Message (Msg Type '131')

2.1.6.1 Message Overview

An ETP Update message provides information about the value updates for ETP.

2.1.6.2 Message Sending Rules

An ETP Update message 131 is sent as a result of one of the following events:

- When the value of the ETP message changes due to a change in Bid/Ask or last sale
- When a dividend, estimated unit of cash creation, total cash amount and shares outstanding values are published

2.1.6.3 Message Structure

The table below describes the body fields of an ETP Update message (MsgType '131'). (See also Notes on Symbol Dissemination.)

Table 13 ETP Update Message Fields

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
Msg Size	0	2	Binary Integer	Size of the message. 24 Bytes
Msg Type	2	2	Binary Integer	This field identifies the type of message '131 – ETP Update Message
SourceTime	4	4	Binary Integer	This field specifies the time when the message was generated in the order book. The number represents the number of seconds in UTC time (EPOCH).
SourceTimeNS	8	4	Binary Integer	This field specifies the number represents the nanosecond portion of UTC time (since EPOCH).
SymbolIndex	12	4	Binary Integer	This field identifies the numerical representation of the symbol. See Symbol Mapping
Value	16	8	Binary Integer	This field contains the corresponding value of the symbol.

2.1.7 ETP Value Message (Msg Type '132')

2.1.7.1 Message Overview

An ETP Value message provides information about the ETP reference data.

2.1.7.2 Message Sending Rules

An ETP Value message 132 is sent as a result of one of the following events:

- At the start of day
- If the reference data changes intraday

2.1.7.3 Message Structure

The table below describes the body fields of an ETP Update message (MsgType '132'). The SymbolIndex maps to the actual ETP symbol (for example, SPY) and the suffix (for example, .NV) can be derived from the NAV, SharesOut, EstCash, TotalCash and Dividend fields. (See also Notes on Symbol Dissemination.)

Table 14 ETP Value Message Fields

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
Msg Size	0	2	Binary Integer	Size of the message. 56 Bytes
Msg Type	2	2	Binary Integer	This field identifies the type of message '132 – ETP Value Message
SourceTime	4	4	Binary Integer	This field specifies the time when the message was generated in the order book. The number represents the number of seconds in UTC time (EPOCH).
SourceTimeNS	8	4	Binary Integer	This field specifies the number represents the nanosecond portion of UTC time (since EPOCH).
SymbolIndex	12	4	Binary Integer	This field identifies the numerical representation of the symbol. See Symbol Mapping.
NAV	16	8	Binary Integer	This field represents the Net Asset value for the ETP.
SharesOut	24	8	Binary Integer	This field represents the shares outstanding value for the ETP. Please note the PriceScaleCode for this field is zero
EstCash	32	8	Binary Integer (Signed)	This field represents the Estimated Cash per unit creation value for the ETP. This can be a negative number.
TotalCash	40	8	Binary Integer (Signed)	This field represents the Total Cash per unit creation value for the ETP. This can be a negative number.
Dividend	48	8	Binary Integer	This field represents the dividend value for the ETP.

2.1.8 Index Value Update Message (Msg Type '133')

2.1.8.1 Message Overview

An Index Value message provides information about Index reference data.

2.1.8.2 Message Sending Rules

An Index Value message 133 is sent as a result of one of the following events:

- At the start of day
- If the reference data changes intraday

2.1.8.3 Message Structure

The table below describes the body fields of an Index Value message (MsgType '133'). (See also Notes on Symbol Dissemination.)

Table 15 Index Value Message Fields

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
Msg Size	0	2	Binary Integer	Size of the message. 24 Bytes
Msg Type	2	2	Binary Integer	This field identifies the type of message '133' – Index Value Message
SourceTime	4	4	Binary Integer	This field specifies the time when the message was generated in the order book. The number represents the number of seconds in UTC time (EPOCH).
SourceTimeNS	8	4	Binary Integer	This field specifies the number represents the nanosecond portion of UTC time (since EPOCH).
SymbolIndex	12	4	Binary Integer	This field identifies the numerical representation of the symbol. See Symbol Mapping
Dividend	16	8	Binary Integer	This field represents the dividend value for the Index

3. CONTROL MESSAGE SPECIFICATIONS

3.1 INTRODUCTION

There are two types of messages transmitted as part of this protocol:

- **Control messages** These do not contain data, they allow conversing parties to exchange session-specific information (for example 'reset sequence number').
- **Data messages** These are product-specific and control messages apply to all products.

3.2 PACKET HEADER FORMAT

All messages will contain a common packet header. See Packet Header for product-specific headers. The design is intended to minimize the development burden on behalf of clients. This means that all clients may implement line-level protocol processing once, and then need only develop parsing algorithms for their choice of message.

3.3 CONTROL MESSAGE TYPES

The table below shows all of the message types used by the XDP system. To determine which message types are contained in the specific product, see Control Message Types in the Index Service Message Specifications section.

- Msgtypes 1-9 are reserved for control messages published on both TCP and MC
- Msgtypes 10-30 are reserved for control messages published on TCP only
- Msgtypes 31-50 are reserved for control messages published on MC only

Table 16 Control Message Types

MSGTYPE	DESCRIPTION	REAL-TIME INDEX FEED	INTERVAL BASED INDEX FEED	INTERVAL BASED INDEX FEED (BASIC)	DELIVERY
1	Sequence Number Reset	x	x	x	Multicast
2	SourceTime Reference Message				Multicast
3	Symbol Index Mapping	x	x	x	Multicast
5	Option Series Index Mapping				Multicast
10	Retransmission Request Message	x	x		TCP/IP*
11	Request Response Message	x	x		TCP/IP*
12	Heartbeat Response Message				TCP/IP*

MSGTYPE	DESCRIPTION	REAL-TIME INDEX FEED	INTERVAL BASED INDEX FEED	INTERVAL BASED INDEX FEED (BASIC)	DELIVERY
13	Symbol Index Mapping Request Message	x	x		TCP/IP*
15	Refresh Request Message				TCP/IP*
31	Message Unavailable				Multicast
32	Symbol Clear				Multicast
33	Trading Session Change				Multicast
34	Security Status Message				Multicast
35	Refresh Header Message				Multicast

*** Note:** TCP/IP capability is not included in the initial release of this product but will be added in a future release.

3.4 SEQUENCE NUMBER RESET (MSG TYPE '1')

This message is sent to 'reset' the Packet Sequence Number at start of day, in response to failures, and so forth. Note that this message will contain a valid sequence number.

Table 17 Sequence Number Reset Message Fields

Header

FIELD	VALUE
PktSize	'30 Bytes'
DeliveryFlag	Valid values include: '12' – Sequence Number Reset
NumberMsgs	1
SeqNum	1
SendTime	
SendTimeNS	

Message Body

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary Integer	The size of the message body in bytes.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
				14 Bytes
MsgType	2	2	Binary Integer	The type of message. '1' – Sequence Number Reset message
SourceTime	4	4	Binary Integer	The time when the message was generated in the order book. The number represents the number of seconds in UTC time (EPOCH).
SourceTimeNS	8	4	Binary Integer	The number represents the nanosecond portion of UTC time (since EPOCH).
ProductID	12	1	Binary Integer	The product ID used in the XDP header to identify the NYSE feed. See Product IDs
ChannelID	13	1	Binary Integer	The multicast channel ID over which the packet was sent.

3.4.1 Sequence Number Reset Processing Notes

Sequence numbers normally begin at one (1) with a sequence number reset message and increase by calculating SeqNum+NumberMsgs with each subsequent packet. Sequence numbers are reset:

- At start of day a sequence number reset message is sent
- If the value should exceed the maximum value that the SeqNum field may contain, it will be reset to one (1)
- If XDP has a failure and it recovers, it sends a sequence number reset message. The SeqNum field of that message will be set to one (1).

3.5 GENERAL HEARTBEAT PROCESSING NOTES (TCP AND MULTICAST)

Note: TCP/IP capability is not included in the initial release of this product but will be added in a future release.

The following applies to the TCP channels for retransmissions and refresh, and also the multicast channels for real-time and refresh data:

- Heartbeat messages will only contain the packet header (with a DeliveryFlag = '1')
- Heartbeats will sent over on the TCP/IP Retrans Request connection and over the production multicast
- Heartbeat frequency since the last packet, is:
  - 60 seconds in the active TCP/IP retransmission sessions
  - Heartbeat messages will be sent with the next expected sequence number, see Sequence Numbers

3.5.1 Retransmission and Refresh Heartbeat Processing Notes (TCP) (Future Implementation)

- Clients may receive a heartbeat message if they have an active TCP/IP session with the retransmission or Refresh Server.

- To determine the health of the user connection on the TCP/IP channel, the retransmission or Refresh Server will send regular heartbeat messages to the user. The heartbeat frequency is 60 seconds. The time out for this heartbeat response message is set at 5 seconds. If no response is received by the server within this timeframe, the TCP/IP session will be disconnected.
- Clients that choose to establish and remain connected to the retransmission or Refresh Server intraday must respond to a heartbeat message with a heartbeat response message. Users can choose to either disconnect following each retransmission or refresh request, or remain connected to the Retransmission or Refresh Server.

Figure 7 Retransmission/Refresh Server Heartbeats

3.6 HEARTBEAT RESPONSE (MSG TYPE '12')

Clients that choose to establish and remain connected to the Retransmission Server intraday must respond to a heartbeat message with a heartbeat response message.

Table 18 Heartbeat Response Message Fields

Header

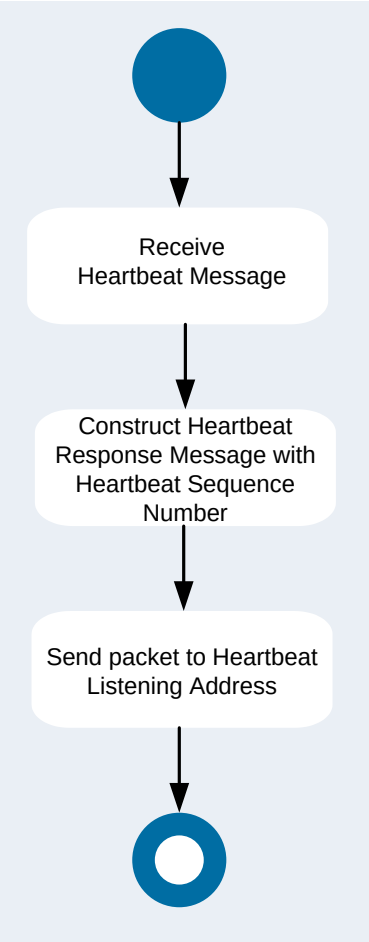
FIELD	VALUE
PktSize	30 Bytes
DeliveryFlag	Valid values include: '11' – Original Message Only
NumberMsgs	
SeqNum	
SendTime	
SendTimeNS	

Message Body

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary Integer	The size of the message body in bytes. 14 Bytes
MsgType	2	2	Binary Integer	The type of message. '12' – Heartbeat Response message
SourceID	4	10	ASCII String	The name of the source requesting retransmission. This field is 9 characters, null terminated.

3.6.1 Heartbeat Message Processing

Figure 8 Processing of Heartbeat Messages



3.7 RETRANSMISSION REQUEST (MSG TYPE '10') (FUTURE IMPLEMENTATION)

Note: This functionality is not included in the initial release of this product but will be added in a future release.

This message is sent by clients requesting missing messages identified by a sequence number gap. Upon receipt of a valid retransmission request message, the requested message(s) will be sent. The requested message(s) have the same message format and content as the original sent by the system.

Table 19 Retransmission Request Message Fields

Header

FIELD	VALUE
PktSize	40 Bytes
DeliveryFlag	Valid values include: '11' – Original Message uncompressed '31' – Original Message compressed (FAST)
NumberMsgs	
SeqNum	
SendTime	
SendTimeNS	

Message Body

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary Integer	This field indicates the size of the message body in bytes. 24 Bytes
MsgType	2	2	Binary Integer	This field identifies the type of message. '10' – Retransmission Request message
BeginSeqNum	4	4	Binary Integer	The beginning sequence number of the requested range of messages to be retransmitted.
EndSeqNum	8	4	Binary Integer	The end sequence number of the requested range of messages to be retransmitted.
SourceID	12	10	ASCII String	This field represents the name of the source requesting retransmission. This field is nine characters, null terminated.
ProductID	22	1	Binary Integer	The product ID used to identify the NYSE feed. See Product IDs
ChannelID	23	1	Binary Integer	This field contains the multicast channel ID where you would like to request data from

3.8 REFRESH REQUEST (MSG TYPE '15') (FUTURE IMPLEMENTATION)

Note: This functionality is not included in the initial release of this product but will be added in a future release.

This message is sent by clients requesting a refresh. The system will provide the appropriate message(s) in response.

Table 20 Refresh Request Message Field

Header

FIELD	VALUE
PktSize	'36 Bytes'
DeliveryFlag	Valid values include: • '11' – Original Message uncompressed • '31' – Original Message compressed (FAST)
NumberMsgs	1
SeqNum	
SendTime	
SendTimeNS	

Message Body

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary Integer	This field indicates the size of the message body in bytes. '20 Bytes'
MsgType	2	2	Binary Integer	This field identifies the type of message. '15' – Refresh Request Message
SymbolIndex	4	4	Binary Integer	This field identifies the numerical representation of the symbol. **Note**: If this field is set to zero, XDP will generate a refresh for all symbols.
SourceID	8	10	ASCII String	This field represents the name of the source requesting retransmission. This field is nine characters, null terminated.
ProductID	18	1	Binary Integer	The product ID used to identify the NYSE feed. See Product IDs.
ChannelID	19	1	Binary Integer	This field contains the multicast channel ID where you would like to request data from

3.9 REQUEST RESPONSE MESSAGE (MSG TYPE '11') (FUTURE IMPLEMENTATION)

Note: This functionality is not included in the initial release of this product but will be added in a future release.

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

Table 21 Request Response Message Fields

Header

FIELD	VALUE
PktSize	
DeliveryFlag	Valid values include: '11' – Original Message uncompressed '31' – Original Message compressed (FAST)
NumberMsgs	1
SeqNum	
SendTime	
SendTimeNS	

Message Body

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary Integer	This field indicates the size of the message body in bytes. 21Bytes
MsgType	2	2	Binary Integer	This field identifies the type of message. '11' – Request Response Message
RequestSeqNum	4	4	Binary Integer	This field contains the request message sequence number assigned by the client. It is used by the client to couple the request with the response message.
SourceID	8	10	ASCII String	This field represents the name of the source requesting retransmission. This field is 10 characters, null terminated.
ProductID	18	1	Binary Integer	The product ID used in the XDP header to identify the NYSE feed. See Product IDs
ChannelID	19	1	Binary Integer	This field contains the multicast channel ID from which you requested data.
Status	20	1	ASCII Character	This is a flag that indicates the reason why the request was rejected.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
				Valid values are: ■ '0' – Message was accepted ■ '1' – Rejected due to an Invalid source ID ■ '2' – Rejected due to invalid sequence range ■ '3' – Rejected due to maximum sequence range (see threshold limits) ■ '4' – Rejected due to maximum request in a day ■ '5' – Rejected due to maximum number of refresh requests in a day ■ '6' – Rejected. Request message seqnum TTL (Time to live) is too old. Use refresh to recover current state if necessary. ■ '7' – Rejected due to an Invalid Channel ID ■ '8' – Rejected due to an Invalid Product ID

3.10 RETRANSMISSION MESSAGES(FUTURE IMPLEMENTATION)

Note: This functionality is not included in the initial release of this product but will be added in a future release.

Upon receipt of a valid retransmission request message, the requested message(s) will be sent. This message(s) has the same message format and content as the original messages sent by XDP but may be grouped in a single packet for maximum packet efficiency (see Sequence Numbers). These messages will flow through the retransmission IP address and channel

Table 22 Retransmission Message Packet Header

FIELD	DESCRIPTION
PktSize	
DeliveryFlag	'40' – End of Refresh Update compressed (Fast)
NumberMsgs	
SeqNum	
SendTime	
SendTimeNS	

3.10.1 Retransmission Message Processing

All Subscribers will receive retransmission messages through the retransmission channel.

Due to the multicast nature, subscribers will receive 'all' retransmission messages, including messages that were not requested by them.

Note that when a message for a particular symbol is retransmitted, a new message **for the same symbol** may be sent through the regular channel. This scenario is very likely to occur with busy symbols and may cause confusion as to which message contains the latest information on that symbol.

In order to resolve the conflict, the following qualification method should be applied:

- Check the SeqNum field. A retransmitted message retains the same sequence number as the original message.
- The most current sequence number (SEQNUM) contains the latest information.
- If the SeqNum fields are the same: messages are the same, any of the two messages contains the same information.

3.11 SYMBOL INDEX MAPPING REQUEST MESSAGE (MSG TYPE '13')

This message is sent (in Little-Endian format) by Subscribers via TCP/IP requesting the Symbol index mapping.

Table 23 Symbol Index Mapping Request Message Fields

Header

FIELD	DESCRIPTION
PktSize	
DeliveryFlag	Valid values include: '11' – Original Message uncompressed '31' – Original Message compressed (FAST)
NumberMsgs	
SeqNum	
SendTime	
SendTimeNS	

Message Body

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary Integer	This field indicates the size of the message body in bytes. 21 Bytes
MsgType	2	2	Binary Integer	This field identifies the type of message. '13' – Symbol Index Mapping Request Message

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
SymbolIndex	4	4	Binary Integer	This field identifies the numerical representation of the symbol. SymbolIndex value can be zero, which is to request all symbol mapping for the multicast group.*
SourceID	8	10	ASCII String	This field represents the name of the source requesting retransmission. This field is nine characters, null terminated.
ProductID	18	1	Binary Integer	The product ID used in the XDP header to identify the NYSE feed. See Product IDs
ChannelID	19	1	Binary Integer	This field contains the multicast channel ID where you requesting the index map from.
RetransmitMethod	20	1	Binary Integer	This field identifies the Retransmission method for the symbol index mapping. Valid values are: '0' – retransmit via UDP

* To request all symbols for a specific multicast group, specify '0' (zero) in the SymbolIndex.

** RetransmitMethod is a field that gives customers the ability to specify if they want Symbol Index Mapping sent to them via TCP/IP or UDP. *This feature is not yet available.*

3.12 SYMBOL INDEX MAPPING MESSAGE (MSG TYPE '3')

This message is sent to Subscribers via TCP/IP or multicast upon request of the Symbol index mapping. It is also sent as part of a refresh packet. The message is sent at approximately 12.30 EST.

Note: For NYSE / NYSE MKT this message is available ONLY over multicast at this time.

Table 24 Symbol Index Mapping Message Fields

Header

FIELD	DESCRIPTION
PktSize	This field indicates the size of the message body in bytes.
DeliveryFlag	Note: 17, 18, 19 and 20 are for the interval service only and indicate the total number of refresh packets being sent for that update. Valid values include: ■ '11' – Original Message Uncompressed ■ '31' – Original Message Compressed ■ 13' – Only one packet in retransmission update Uncompressed ■ '14' – Start of retrans Update Uncompressed ■ '15' – Part of a retransmission sequence Uncompressed

FIELD	DESCRIPTION
	■ '16' – End of retrans Update Uncompressed ■ '17' – Only one packet in Refresh update Uncompressed ■ '18 – Start of Refresh Update Uncompressed ■ '19' – Part of a Refresh sequence Uncompressed ■ '20' – End of Refresh Update Uncompressed ■ '33' - Only one packet in retransmission update compressed (Fast) ■ '34' – Start of retrans Update compressed (Fast) ■ '35' – Part of a retransmission sequence compressed (Fast) ■ '36' – End of retrans Update compressed (Fast) ■ '37' – Only one packet in Refresh update compressed (Fast) ■ '38' – Start of Refresh Update compressed (Fast) ■ '39' – Part of a Refresh sequence compressed (Fast) ■ '40' – End of Refresh Update compressed (Fast)
NumberMsgs	
SeqNum	
SendTime	
SendTimeNS	

Message Body

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary Integer	Size of the message. NYSE Arca – 26 Bytes
Msg Type	2	2	Binary Integer	This field identifies the type of message '3' – Symbol Index Map Message
SymbolIndex	4	4	Binary Integer	This field identifies the numerical representation of the symbol.
Symbol	8	11	ASCII String	This field contains the full symbol in NYSE Symbology. A sequence of characters representing the symbol, padded with NULLs.
Flag	19	1	Binary Integer	This field indicates whether the value was calculated from bid, ask or last sale prices: ■ '0' – NA ■ '1' – Calculated from the Bid Price ■ '2' – Calculated from the Ask Price

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
				■ '3' – Calculated from the last sale
Market ID	20	2	Binary Integer	ID of the Originating Market. This field is not yet applicable to the NYSE & NYSE MKT market.
System ID	22	1	Binary Integer	ID of the Originating System. This field is not yet applicable to the NYSE & NYSE MKT market.
Exchange Code	23	1	ASCII String	Exchanges where it is listed: ■ 'N' – NYSE ■ 'P' – NYSE Arca (GIF set to all 'P') ■ 'Q' – NASDAQ ■ 'A' – NYSE MKT
PriceScaleCode	24	1	Binary Integer	Price scale for price conversion of the symbol. See Price Formats.
Security Type	25	1	ASCII String	Type of Security: ■ 'A – ADR ■ 'C' – Common Stock ■ 'D' – Debentures ■ 'E' – ETP ■ 'F' – Foreign ■ 'H' – Depository Shares ■ 'I' – Units ■ 'L' – Index Linked Notes ■ 'M' - Misc/Liquid Trust ■ 'P' – Preferred Stock ■ 'R' – Rights ■ 'S' – Shares of Beneficiary Interest ■ 'T' – Test ■ 'U' – Closed-End Fund ■ 'X' – Index ■ 'W' – Warrant

3.13 MESSAGE UNAVAILABLE MESSAGE (MSG TYPE '31') (FUTURE IMPLEMENTATION)

Note: This functionality is not included in the initial release of this product but will be added in a future release.

This message will be sent to inform subscribers that a range of messages are unavailable which they may have requested retransmission of via the Retransmission multicast channels.

Table 25 Message Unavailable Message Fields

Header

FIELD	DESCRIPTION
PktSize	
DeliveryFlag	Valid values include: ■ '21' – Message Unavailable ■ '41' – Message Unavailable
NumberMsgs	
SeqNum	
SendTime	
SendTimeNS	

Message Body

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary Integer	This field indicates the size of the message body in bytes. '14 Bytes'
MsgType	2	2	Binary Integer	This field identifies the type of message. '31' – Message Unavailable
BeginSeqNum	4	4	Binary Integer	The beginning sequence number of the requested range of messages to be retransmitted.
EndSeqNum	8	4	Binary Integer	The end sequence number of the requested range of messages to be retransmitted.
ProductID	12	1	Binary Integer	The product ID used in the XDP header to identify the NYSE feed. See Product IDs.
ChannelID	13	1	Binary Integer	This field contains the multicast channel ID where you requested the index map from.

3.14 TRADING HALT MESSAGE (MSG TYPE '34')

This message will be sent to inform the subscribers of Trading Halts on the securities traded.

Table 26 Trading Halt Message Fields

Header

FIELD	DESCRIPTION
PktSize	
DeliveryFlag	Valid values include: – '11' – Original Message Uncompressed – '31' – Original Message Compressed
NumberMsgs	
SeqNum	
SendTime	
SendTimeNS	

Message Body

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary Integer	This field indicates the size of the message body in bytes. 18 bytes
MsgType	2	2	Binary Integer	This field identifies the type of message. '34' – Trading Halt Message
SourceTime	4	4	Binary Integer	This field specifies the time when the message was generated in the order book. The number represents the number of seconds in UTC time (EPOCH).
SourceTimeNS	8	4	Binary Integer	This field specifies the number represents the nanosecond portion of UTC time (since EPOCH).
SymbolIndex	12	4	Binary Integer	This field identifies the numerical representation of the symbol.
SymbolSeqNum	16	4	Binary Integer	This field contains the symbol sequence number.
Security Status	20	1	Binary Integer	■ '3' - Opening Delay ■ '4' - Trading Halt ■ '5' - Resume ■ '6' - No open/no resume

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Halt Condition	21	1	ASCII String	Null – not used

3.15 REFRESH HEADER (MSG TYPE '35') (FUTURE IMPLEMENTATION)

Note: This functionality is not included in the initial release of this product but will be added in a future release.

This message is the first message type sent in each refresh packet and will never been sent out as its own packet.

Table 27 Refresh Header Message Fields

Header

FIELD	DESCRIPTION
PktSize	
DeliveryFlag	**Note**: 17, 18, 19 and 20 are for the interval service only and indicate the total number of refresh packets being sent for that update. Valid values include: ■ '17' – Only one packet in Refresh Update Uncompressed ■ '18 – Start of Refresh Update Uncompressed ■ '19' – Part of a Refresh sequence Uncompressed ■ '20' – End of Refresh Update Uncompressed ■ '37' – Only one packet in Refresh Update compressed (Fast) ■ '38' – Start of Refresh Update compressed (Fast) ■ '39' – Part of a Refresh sequence compressed (Fast) ■ '40' – End of Refresh Update compressed (Fast)
NumberMsgs	
SeqNum	
SendTime	
SendTimeNS	

Message Body

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary Integer	The size of the message body in bytes. '12 Bytes'
MsgType	2	2	Binary Integer	This field identifies the type of message. '35' – Refresh Header Message
CurrentRefreshPkt	4	2	Binary	This field represents the current refresh packet in

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
			Integer	the update.
TotalRefreshPkts	6	2	Binary Integer	This field represents the total number of refresh packets that you should expect in the update.
LastSeqNum	8	4	Binary Integer	This field contains the last sequence number sent on the feed. The refresh is the state of the book as of this sequence number.

3.15.1.1 Refresh Example

Assuming the refresh of symbol xyz requires three packets, the first Packet structure will look as follows:

PACKET HEADER	REFRESH HEADER	MESSAGE 1	MESSAGE 2	...	MESSAGE N

Table 28 Packet Header

FIELD	DESCRIPTION
PktSize	
DeliveryFlag	**Note**: 17, 18, 19 and 20 are for the interval service only and indicate the total number of refresh packets being sent for that update. Valid values include: ■ '17' – Only one packet in Refresh Update Uncompressed ■ '18 – Start of Refresh Update Uncompressed ■ '19' – Part of a Refresh sequence Uncompressed ■ '20' – End of Refresh Update Uncompressed ■ '37' – Only one packet in Refresh Update compressed (Fast) ■ '38' – Start of Refresh Update compressed (Fast) ■ '39' – Part of a Refresh sequence compressed (Fast) ■ '40' – End of Refresh Update compressed (Fast)
NumberMsgs	
SeqNum	
SendTime	
SendTimeNS	

The first message in the body will be the refresh header:

Table 29 Refresh Header

FIELD NAME	OFFSET	SIZE (BYTES)	VALUE
MsgSize	0	2	12 Bytes
MsgType	2	2	35' – Refresh Header Message
CurrentRefreshPkt	4	2	1
TotalRefreshPkts	6	2	3
LastSeqNum	8	4	5000

In the case of the Integrated Feed, the Refresh header will be followed by the following message types:

1. Symbol Index Mapping Message (Msg Type 3)
2. Security Status Message (Msg Type 34), if there is a trading status notification

4. PRODUCTION CONFIGURATION

4.1 INTRODUCTION

4.1.1 Data Content

The information supplied in this chapter applies only to the XDP feed for the NYSE Arca production environment on which the NYSE Global Index Feed is built.

4.1.2 Data Delivery

NYSE market data is available only via the SFTI® network.

4.2 PRODUCTION HOURS

The following table describes the main daily event–generating activity and indicative time on the NYSE Global Index Feed.

Note: See **Table 11 Message Publication Times** and below for details of when MsgTypes -130,131,132 and133 are published.

Table 30 Production Event Schedule

EVENT	TIME (EST)*	COMMENT
Sequence Number reset	~7:45pm	*Note: The startup cycle begins at 6:00pm on Sundays
Symbol Mapping	~7:45pm	System opens at approximately 19:45pm.
Pre-Open	~7:45pm	Heartbeats.
Open	~7:45pm	Open time refers to the start of the GIF feed production hours. (see Sunday start up note above)
Close	7:15pm	Close time refers to the close of the GIF feed production hours.

4.3 MULTICAST/TCP SETUP

4.3.1 Joining Multicast Groups

Recipients' applications/hosts are responsible for issuing multicast subscriptions to one or more of the multicast groups assigned to the product. In response to each authorized subscription request, the SFTI® network will complete the tasks associated with delivering the multicast packets from the NYSE data source to the recipient's network.

The process of subscribing to a multicast group ID is also known as 'joining' a multicast group. Upon session termination, the subscriber's host system should issue an 'unjoin' message. This will terminate delivery of data to that host's local network. If an application/host terminates without issuing an 'unjoin' message, the network will eventually issue a 'timeout' for the multicast group subscription that will automatically terminate delivery of the multicast packets to the host's local network.

4.3.2 Feeds

All data is published using two sets of unique IP multicast group IDs ("primary" and "secondary") – the two will originate from the same distribution site. The feeds are redundant to each other, that is, they are synchronized with each other. Each message from each feed contains the same packet sequence number.

4.3.3 Heartbeat Mechanism

The heartbeat message frequency is set to 60 seconds on the TCP/IP connection. A heartbeat message response has to be sent within 5 seconds to stay connected to the server.

4.3.4 Source ID

The Source ID allows clients to perform retransmission and refresh requests. Please note that the Source IDs for retransmissions and refresh are identical. NYSE will provide each client with a default of one Source ID for Production with a limit of up to four Source IDs per client.

Each Source ID may be logged only on to a server once at a given time.

4.3.5 Number of Source IDs

Clients are allowed a max of four Source IDs. Contact NYSE Technical support services to set up a Source ID.

4.3.6 Parallel Sessions

Clients may file several concurrent requests on the server at the same time with the same Source ID; there is no need to wait for the active retransmission to be closed to ask for another one. Responses to these requests are sent in the same order as the initial requests.

4.3.7 Retransmission/Refresh Request Limitations (Future Implementation)

The recommendations below apply to production and test environments.

Table 31 Retransmission/Refresh Request Limitations

CAPABILITY	DESCRIPTION	THRESHOLD	ACTION
Prevention of invalid subscribers	Incoming requests from subscribers that are not in the enabled subscriber's Source ID list will not be honored. PDP subscribers will need a Source ID, which is a string that uniquely identifies the subscriber of the retransmission requests. Please contact the NYSE Service Desk to get a unique Source ID.	N/A	Request will not be processed.
Limitation of Requests for a large number of packets	Only retransmission requests for 1000 packets or less will be honored.	1000	Request will not be processed.
Limitation of Generic Requests Time Interval	If the generic request on a message which is not within this threshold, the request will not be honored.	75000	Request will not be honored.
Limitation of Generic Requests	Generic requests for messages not within the threshold number of requests per day, will not be honored during that particular	500	Subsequent retransmissions requests from

CAPABILITY	DESCRIPTION	THRESHOLD	ACTION
	day.		that subscriber will be blocked.
Limitation of requests for refresh messages	Up to 5000 refresh requests will be honored.	5000	Request will not be honored.

5. PRODUCTION CHANNEL CONFIGURATION

For details, see: http://www.nyxdata.com/ipaddresses