# EDX Markets™ FIX Specifications – Binary Market Data (UAT)

## EDX Markets

v1.0.10

*Change Log* ..... *4*
v1.0.2 - Added Streaming Session Start Message Type 8 ..... 4
v1.0.3 - Added Heartbeat Message Type 0 to UDP Broadcast Protocol ..... 4
v1.0.4 - Corrected Order Executed from 64-bit integer to Fixed Point Decimal ..... 4
v1.0.5 - Removed session identifier message table. ..... 4
v1.0.6 - Corrected heartbeat message datagram 1. ..... 4
*Order and Snapshot Messages* ..... *4*
**Introduction** ..... **4**
**Common SBE Header** ..... **4**
Versioning ..... 5
**Instrument Directory** ..... **5**
Instrument Type Values ..... 6
**Instrument Trading Status** ..... **6**
Instrument Trading Status Field Values ..... 6
Instrument Trading Status Reason Field Values ..... 7
**Trading Session Status** ..... **7**
Trading Session Field Values ..... 7
**Snapshot Complete** ..... **7**
**Order Added** ..... **8**
**Order Deleted** ..... **9**
**Order Reduced** ..... **9**
**Order Executed** ..... **10**
**Common Field Types** ..... **10**
**Incremental Trading Metric** ..... **11**
MdEntryType Values ..... 12
*UDP Broadcast Protocol* ..... *12*
**Introduction** ..... **12**
**Protocol Definition** ..... **12**
Header Definition ..... 12
Heartbeat Message (Message Type 0) ..... 14
Market Data Message (Message Type 2) ..... 14
*TCP Snapshot Protocol* ..... *16*
**Introduction** ..... **16**
**Protocol Definition** ..... **16**
Header Definition ..... 16
Snapshot Request ..... 16
Snapshot Request Accepted Response ..... 16

Snapshot Request Rejected Response....16
Snapshot Header Message....17
Snapshot Message....17
Snapshot Footer Message....17
Snapshot Session Start Message....17
**Interaction Model....17**
***TCP Streaming Protocol....18***
**Introduction....18**
**Protocol Definition....18**
Header Definition....18
Login Request....18
Login Request Accepted Response....18
Login Request Rejected Response....18
Snapshot Header Message....18
Snapshot Message....19
Snapshot Footer Message....19
Stream Data Message....19
Streaming Session Start Message....19
**Interaction Model....19**

# Change Log

| Date | Section / Message(s) | Description |
|---|---|---|
| 3/4/2024 | N/A | Initial v1.0 release of Binary Market Data specifications for EDX Markets match engine. |
| 3/6/2024 | Snapshot Session Start Message | v1.0.1 – Added Snapshot Session Start Message Type 8 |
| 4/5/2024 | Streaming Session Start Message | v1.0.2 - Added Streaming Session Start Message Type 8 |
| 4/22/2024 | Heartbeat Message | v1.0.3 - Added Heartbeat Message Type 0 to UDP Broadcast Protocol |
| 4/30/2024 | Order Executed | v1.0.4 - Corrected Order Executed from 64-bit integer to Fixed Point Decimal |
| 5/1/2024 | Session Identifier Message | v1.0.5 - Removed session identifier message table. |
| 5/6/2024 | Heartbeat Message | v1.0.6 - Corrected heartbeat message datagram 1. |
| 5/21/2024 | OrderReduced Message | v1.0.7 - Added OrderReduced Message |
| 6/7/2024 | Current SessionID | v1.0.8 - Clarified unscheduled disconnect description |
| 9/11/2024 | Instrument Directory Message | v1.0.9 - Added Instrument Type |
| 9/26/2024 | Incremental Trading Metric Message | v1.0.9 - Added Incremental Trading Metric Message |
| 10/10/2024 | All messages | v1.0.10 - Increased length of Token ID and Base Currency and Quote Currency |

# Order and Snapshot Messages

## Introduction

This section describes the implementation and protocol for Order messages transmitted from the Binary Market Data Gateway. These messages are encoded using the Simple Binary Encoding (SBE) format, with fields encoded in big-endian byte order.

Common SBE Header

Every message begins with the SBE Header, which describes the message payload.

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 0 | 2 | Block Length | The number of fixed sized bytes in the message body. | Unsigned 16-bit integer |
| 2 | 1 | Template ID | Identifier of the message type. | Unsigned 8-bit integer |
| 3 | 1 | Schema ID | Identifier of the message schema. | Unsigned 8-bit integer |
| 4 | 2 | Version | Identifier of the message version. | Unsigned 16-bit integer |

Versioning
Versioning of the Order Message schema is encoded within the Common SBE Header. The first 8 bits of the version field represents the major version of the schema, and the subsequent 8 bits represents the minor version of the schema. A version of 0300 in hex therefore represents the semantic version 2.0.

## Instrument Directory

This message will be emitted as part of a Snapshot to inform the client of available instruments.
The header is sent using the values

| Name | Value |
|---|---|
| Block Length | 46 |
| Template ID | 1 |
| Schema ID | 6 |
| Version | 514 |

The payload is structured using the specification

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 0 | 6 | Header | Common SBE Header | - |
| 6 | 8 | Timestamp | The timestamp when the event occurred, in nanoseconds | Signed 64-bit integer |
| 14 | 16 | Token ID | Unique identifier of the Order instrument | 16-byte character sequence |
| 30 | 5 | Base Currency | Base Currency Symbol | 5-byte character sequence |
| 35 | 5 | Quote Currency | Quote Currency Symbol | 5-byte character sequence |
| 40 | 2 | Unit Multiplier | The Unit Multiplier of the Instrument | Signed 16-bit integer |
| 42 | 1 | Is Test Symbol | If the Instrument is a Test Instrument | Boolean Type |
| 43 | 8 | MPV | The price increment of the instrument | FixedPointDecimal |

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 51 | 1 | Instrument Type | Type of Instrument | 1 byte character |

Instrument Type Values

| ASCII Value | Name |
|---|---|
| '1' | Spot |
| '2' | Perpetual Futures |

## Instrument Trading Status

This message will be emitted as part of a Snapshot to inform the client of the status of available instruments
The header is sent using the values

| Name | Value |
|---|---|
| Block Length | 26 |
| Template ID | 2 |
| Schema ID | 6 |
| Version | 514 |

The payload is structured using the specification

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 0 | 6 | Header | Common SBE Header | - |
| 6 | 8 | Timestamp | The timestamp when the event occurred, in nanoseconds | Signed 64-bit integer |
| 14 | 16 | Token ID | Unique identifier of the Order instrument | 16-byte character sequence |
| 30 | 1 | Instrument Trading Status | The Trading Status of the Instrument | 1 byte character sequence |
| 31 | 1 | Instrument Trading Status Reason | The Reason for this Status | 1 byte character sequence |

Instrument Trading Status Field Values

| ASCII Value | Name |
|---|---|
| 'H' | Halted |
| 'Q' | Quoting |

| ASCII Value | Name |
|---|---|
| ‘L’ | Limit Only Trading |
| ‘T’ | Trading |

Instrument Trading Status Reason Field Values

| ASCII Value | Name |
|---|---|
| ‘X’ | None |
| ‘A’ | Administrative |

## Trading Session Status

This message will be emitted as part of a Snapshot to inform the client of the status of the trading session
The header is sent using the values

| Name | Value |
|---|---|
| Block Length | 9 |
| Template ID | 3 |
| Schema ID | 6 |
| Version | 514 |

The payload is structured using the specification

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 0 | 6 | Header | Common SBE Header | - |
| 6 | 8 | Timestamp | The timestamp when the event occurred, in nanoseconds | Signed 64-bit integer |
| 14 | 1 | Trading Session | Status representing if the Exchange is open or closed | 1 byte character sequence |

Trading Session Field Values

| ASCII Value | Name |
|---|---|
| ‘1’ | Trading |
| ‘2' | Closed |

## Snapshot Complete

This message will be emitted as part of a Snapshot to indicate the end of the snapshot
The header is sent using the values

| Name | Value |
|---|---|
| Block Length | 16 |

| Name | Value |
|---|---|
| Template ID | 4 |
| Schema ID | 6 |
| Version | 514 |

The payload is structured using the specification

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 0 | 6 | Header | Common SBE Header | - |
| 6 | 8 | Timestamp | The timestamp when the event occurred, in nanoseconds | Signed 64-bit integer |
| 14 | 8 | Sequence Number | The current broadcast Sequence number | Signed 64-bit integer |

## Order Added

This message will be emitted when an order has been added to the book or as part of a snapshot.

The header is sent using the values

| Name | Value |
|---|---|
| Block Length | 58 |
| Template ID | 10 |
| Schema ID | 6 |
| Version | 514 |

The payload is structured using the specification

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 0 | 6 | Header | Common SBE Header | - |
| 6 | 8 | Timestamp | The timestamp when the event occurred, in nanoseconds | Signed 64-bit integer |
| 14 | 16 | Token ID | Unique identifier of the Order instrument | 16-byte character sequence |
| 30 | 8 | Order ID | Unique identifier of the Order | Signed 64-bit integer |
| 38 | 8 | Correlation ID | Correlating ID between this message and the Order feed. This value will always be equal to the Order ID | Signed 64-bit integer |
| 46 | 1 | Side | The side of the order being added | 1 byte character |
| 47 | 8 | Quantity | The total quantity being added | Order Quantity |
| 55 | 8 | Price | The price of the new order | FixedPointDecimal |
| 63 | 1 | Retail Indicator | The retail disposition of the order | 1 byte character |

## Order Deleted

This message will be emitted when an order has been removed from the book.
The header is sent using the values

| Name | Value |
|---|---|
| Block Length | 32 |
| Template ID | 11 |
| Schema ID | 6 |
| Version | 514 |

The payload is structured using the specification

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 0 | 6 | Header | Common SBE Header | - |
| 6 | 8 | Timestamp | The timestamp when the event occurred, in nanoseconds | Signed 64-bit integer |
| 14 | 16 | Token ID | Unique identifier of the Order instrument | 16-byte character sequence |
| 30 | 8 | Order ID | Unique identifier of the Order | Signed 64-bit integer |

## Order Reduced

This message will be emitted when an order has been reduced in quantity.
The header is sent using the values

| Name | Value |
|---|---|
| Block Length | 40 |
| Template ID | 12 |
| Schema ID | 6 |
| Version | 514 |

The payload is structured using the specification

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 0 | 6 | Header | Common SBE Header | - |
| 6 | 8 | Timestamp | The timestamp when the event occurred, in nanoseconds | Signed 64-bit integer |
| 14 | 16 | Token ID | Unique identifier of the Order instrument | 16-byte character sequence |
| 30 | 8 | Order ID | Unique identifier of the Order | Signed 64-bit integer |
| 38 | 8 | Quantity | The new quantity | Order quantity |

## Order Executed

This message will be emitted when an order on the book has been executed against. When the quantity executed reduces the remaining quantity to zero, consumers of this message should consider the order to be removed from the book. Otherwise, any outstanding quantity remains on the book with the same priority as the original order.

The header is sent using the values

| Name | Value |
|---|---|
| Block Length | 64 |
| Template ID | 13 |
| Schema ID | 6 |
| Version | 514 |

The payload is structured using the specification

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 0 | 6 | Header | Common SBE Header | - |
| 6 | 8 | Timestamp | The timestamp when the event occurred, in nanoseconds | Signed 64-bit integer |
| 14 | 16 | Token ID | Unique identifier of the Order instrument | 16-byte character sequence |
| 30 | 8 | Order ID | Unique identifier of the Order | Signed 64-bit integer |
| 38 | 16 | Trade ID | Unique identifier of the Trade | 2 Signed 64-bit integers |
| 54 | 8 | Quantity | The total quantity executed | Order quantity |
| 62 | 8 | Price | The executed price of the order | Fixed Point Decimal |

## Common Field Types

*Timestamp*

Represents a timestamp with nanosecond precision starting from the UNIX epoch.

| Length | Type | Name | Description |
|---|---|---|---|
| 8 | Signed 64-bit integer | Timestamp | Time represented as the number of nanoseconds since the UNIX epoch. |

*Token ID*

Unique identifier of an Instrument. Generally, this will take the form of Base Currency/Quote Currency combination.

| Length | Type | Name | Description |
|---|---|---|---|
| 16 | 16-byte character sequence | Token ID | Unique identifier of an Instrument |

*Order ID*

Unique identifier of an Order.

| Length | Type | Name | Description |
|---|---|---|---|
| 8 | Signed 64-bit integer | Order ID | Unique identifier of an Order |

*Trade ID*

Trades are identified using a 128-bit globally unique value. They are expressed as two signed 64-bit integers.

| Offset | Length | Type | Name | Description |
|---|---|---|---|---|
| 0 | 8 | Signed 64-bit integer | Upper Bits | The most significant bits of the ID |
| 8 | 8 | Signed 64-bit integer | Lower Bits | The lease significant bits of the ID |

*Side*

This field type describes the Side which an order is on. This value will is either BID/BUY or OFFER/SELL. The value is encoded as a single character value.

| ASCII Value | Name |
|---|---|
| 'B' | Buy |
| 'S' | Sell |

*Order Quantity*

This field represents the quantity that is present on an order or has been executed against an order as part of a trade. It is encoded as a signed 64-bit integer. To obtain the true quantity value, consumers of this data need to multiply the quantity by 10^(unit multiplier). The unit multiplier is available on the instrument directory.

| Offset | Length | Type | Name |
|---|---|---|---|
| 0 | 8 | Signed 64-bit integer | Quantity |

For example, an order quantity of 123456789 with a corresponding instrument unit multiplier of -8 would represent the decimal number 1.23456789. This was calculated by 123456789 * 10^(-8).

*FixedPointDecimal (Used in Price and MPV)*

This field represents a fixed point decimal, encoded as a signed 64-bit integer with a constant exponent of -8.

| Offset | Length | Type | Name |
|---|---|---|---|
| 0 | 8 | Signed 64-bit integer | Price |

For example, a value 123456789 would represent the decimal number 1.23456789.

*Retail Indicator Type*

The Retail Indicator Type describes the retail disposition for an Order when it is added. It is encoded as a single character.

| ASCII Value | Name |
|---|---|
| '1' | Normal |
| '2' | Designated Retail |
| '3' | Retail Liquidity Provider |

## Incremental Trading Metric

| Name | Value |
|---|---|
| Block Length | 33 |
| Template ID | 14 |
| Schema ID | 6 |
| Version | 514 |

| Offset | Length | Name | Description | Type |
|---|---|---|---|---|
| 0 | 6 | Header | Common SBE Header | - |
| 6 | 8 | Timestamp | The timestamp when the event occurred, in nanoseconds | Signed 64-bit integer |
| 14 | 16 | Token ID | Unique identifier of the Order instrument | 16-byte character sequence |
| 30 | 1 | MdEntryType | Type of trading metric | 1-byte character |
| 31 | 8 | MDEntryValue | Value of trading metric | FixedPointDecimal |

MdEntryType Values

| ASCII Value | Name |
|---|---|
| '3' | IndexValue |
| 'm' | PreliminaryMarkPrice |
| 'n' | FinalMarkPrice |
| 'p' | PreliminaryFundingRate |
| 'f' | FinalFundingRate |
| 'C' | OpenInterest |

# UDP Broadcast Protocol

## Introduction

This section describes the implementation and protocol of the UDP Broadcast functionality present in the Binary Market Data Gateway.

The Binary Market Data Gateway can be configured to broadcast Market Data updates to a UDP Endpoint. This Endpoint can be configured to be either a Standard UDP Address or a UDP Multicast Address. Market Data updates received from the Cluster will be sent to this configured endpoint.

## Protocol Definition

### Header Definition

All messages broadcast from the Binary Market Data Gateway use a shared header. This header is defined using a pure binary format and is encoded in BIG_ENDIAN form. The current size of the header is in total 20 bytes, reading implementations are encouraged to always read the version to determine the size of the header and following content.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |

| Byte Offset | Length | Description |
|---|---|---|
| 1 | 1 | Version and Flags |
| 2 | 8 | Current Session Id |
| 10 | 8 | Sequence Number |
| 18 | 2 | Message Count |

The Binary Market Data Gateway will ensure that a single Header is broadcast per UDP Datagram emitted to the network from its host address. The underlying Data Link Layer may split this packet into multiple fragments, but they will be reassembled before being delivered to a receiving application by this layer.

*Version and Flags*

The second byte of the header encodes information about the version and optional bits for flags presented by the broadcast endpoint.

| Bit Offset | Length (in bits) | Description |
|---|---|---|
| 0 | 4 | Protocol Version |
| 4 | 1 | Reserved Flag |
| 5 | 1 | Reserved Flag |
| 6 | 1 | Reserved Flag |
| 7 | 1 | Reserved Flag |

*Message Types*

The Binary Market Data Gateway supports the following types of messages.

| Value | Message Type |
|---|---|
| 0 | Heartbeat |
| 1 | Reserved |
| 2 | Market Data Message |
| 3-7 | Not Used |

In its current implementation only the values 0 and 2 will be broadcast from the Binary Market Data Gateway. Value 1 is reserved as it was used by earlier versions of the application. The remainder are currently available for future use.

*Current Session Id*

The Current Session Id is an identifier for the current session of the Binary Market Data Gateway. This Session Id changes whenever the Binary Market Data Gateway loses connectivity from upstream components or restarts.

In the event of an unscheduled broadcast Session ID change when the gateway goes down, a receiving client must request a new snapshot of Market Data from the Gateway, to ensure it has the correct view of the current market data.

When SessionID changes as scheduled, receiving clients will carry on as normal.

*Sequence Number*

The Header Value contains the current Sequence Number that the Binary Market Data Gateway has sent up to. Each Market Data message that is sent will increase the Sequence Number by the number of messages contained within the Datagram.

As part of consumption of UDP Market Data it is expected that some loss may occur on the network. In situations where a client detects loss through the Sequence Number incrementing by a larger than expected value between messages it is required to request a new snapshot of Market Data from the Gateway to ensure that it has a correct view of the current market data.

Heartbeat Message (Message Type 0)

In the absence of other messages, a heartbeat message will be sent every 15 seconds, indicating the current Session ID and Sequence Number.

A heartbeat will not increment the sequence number.

The Header portion of a Sequenced Message is encoded using a pure binary format in BIG_ENDIAN format.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 20 | Common Header |
| 20 | 0 | Payload |

*Heartbeat Message Example*

Datagram 1

| Byte Offset | Length | Description | Example Value | Notes |
|---|---|---|---|---|
| 0 | 1 | Message Type | 0 (Heartbeat Message) | |
| 1 | 1 | Protocol Version and Framing | 16 | Protocol Version 1, No Flags Set |
| 2 | 8 | Session Id | 17065462840000000 | The current Session ID |
| 10 | 8 | Sequence Number | 5 | The current Sequence Number |
| 18 | 2 | Message Count | 0 | |

Market Data Message (Message Type 2)

Each Market Data Message datagram contains one or more messages to be processed by listening clients. The first message within the datagram has a sequence number equal to the field found in the Header. Subsequent messages increment this value by one, but these sequence numbers are not encoded as part of the datagram itself and are instead implicit. Clients should expect the Sequence Number in a subsequent message to be the previous message Sequence Number plus the number of messages within the Datagram.

The Header portion of a Sequenced Message is encoded using a pure binary format in BIG_ENDIAN format.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 20 | Common Header |
| 20 | - | Payload |

The Payload portion of the Market Data Message is then divided into sections for each Message containing the length of message and payload bytes. This is again encoded using a pure binary format in BIG_ENDIAN.

| Length | Description |
|---|---|
| 2 | Payload Length |
| - | Payload Bytes |

*Market Data Message Example*

Datagram 1

| Byte Offset | Length | Description | Example Value | Notes |
|---|---|---|---|---|
| 0 | 1 | Message Type | 2 (Market Data Message) | |
| 1 | 1 | Protocol Version and Framing | 16 | Protocol Version 1, No Flags Set |
| 2 | 8 | Session Id | 17065462840000000 | |
| 10 | 8 | Sequence Number | 1 | |
| 18 | 2 | Message Count | 2 | |
| 20 | 2 | Message 1 Length | 16 | |
| 22 | 16 | Message 1 Payload | USD/BTC;1.1;1.2 | (Message Sequence Number 1) |
| 38 | 2 | Message 2 Length | 16 | |
| 40 | 16 | Message 2 Payload | USD/ETH;1.1;1.2 | (Message Sequence Number 2) |

Datagram 2

| Byte Offset | Length | Description | Example Value | Notes |
|---|---|---|---|---|
| 0 | 1 | Message Type | 2 (Market Data Message) | |
| 1 | 1 | Protocol Version and Framing | 16 | Protocol Version 1, No Flags Set |
| 2 | 8 | Session Id | 17065462840000000 | |
| 10 | 8 | Sequence Number | 3 | |
| 18 | 2 | Message Count | 2 | |
| 20 | 2 | Message 1 Length | 16 | |
| 22 | 16 | Message 1 Payload | USD/BTC;1.1;1.2 | (Message Sequence Number 3) |
| 38 | 2 | Message 2 Length | 16 | |
| 40 | 16 | Message 2 Payload | USD/ETH;1.1;1.2 | (Message Sequence Number 4) |

# TCP Snapshot Protocol

## Introduction

This section describes the implementation and protocol of the TCP Snapshot functionality present in the Binary Market Data Gateway.
The Binary Market Data Gateway can be configured to listen for Snapshot requests from a TCP Endpoint. Market Data updates received from the Cluster will be made available to clients which request snapshots from this endpoint.
Once a Snapshot has been received from a client the Binary Market Data Gateway will then perform a disconnect.

## Protocol Definition

Header Definition

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |

Snapshot Request

This message is encoded as Message Type 1.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |
| 3 | n | Login Token |

Snapshot Request Accepted Response

This message is encoded as Message Type 2.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |

Snapshot Request Rejected Response

This message is encoded as Message Type 3.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |
| 3 | 1 | Rejection Reason |

*Rejection Reasons*

| Value | Description |
|---|---|
| T | Bad Token |
| A | Authentication Failure |

Snapshot Header Message

This message is encoded as Message Type 4.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |

Snapshot Message

This message is encoded as Message Type 5.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |
| 3 | n | Snapshot Message Payload |

Snapshot Footer Message

This message is encoded as Message Type 6.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |

Snapshot Session Start Message

This message is encoded as Message Type 8.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |
| 3 | 8 | Session Identifier |

## Interaction Model

- Upon connecting to the Snapshot Gateway the client sends a Login request to the Snapshot Gateway.
- The Snapshot Gateway validates the Login Request and issues a Login Accepted Response upon successful validation.
- The Snapshot Gateway then begins the process of sending a Snapshot by issuing a Snapshot Session Start message.
- The Snapshot Gateway will then send a number of Snapshot Messages
  - The Snapshot begins by sending Instrument Directory messages framed by the Snapshot Message header.
  - The Snapshot then contains Instrument Trading Status messages for each instrument
  - The Snapshot then contains the Trading Session status
  - The Snapshot then contains an OrderAdded entry for each order present on the Exchange
- The Snapshot Gateway will finally issue a Snapshot Footer Message to denote the completion of the Snapshot.
- The Snapshot Gateway will then disconnect the client.

# TCP Streaming Protocol

## Introduction

This section describes the implementation and protocol of the TCP Unicast functionality present in the Binary Market Data Gateway.

The Binary Market Data Gateway can be configured to listen for Streaming requests from a TCP Endpoint. Market Data updates received from the Cluster will be made available to clients which request streams from this endpoint.

When a client connects to the streaming endpoint it will receive a Snapshot, followed by a number of streamed updates. The connection remains active until the client decides to disconnect or the Gateway disconnects from the Cluster.

## Protocol Definition

Header Definition

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |

Login Request

This message is encoded as Message Type 1.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |
| 3 | n | Login Token |

Login Request Accepted Response

This message is encoded as Message Type 2.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |

Login Request Rejected Response

This message is encoded as Message Type 3.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |
| 3 | 1 | Rejection Reason |

*Rejection Reasons*

| Value | Description |
|---|---|
| T | Bad Token |
| A | Authentication Failure |

Snapshot Header Message

This message is encoded as Message Type 4.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |

Snapshot Message
This message is encoded as Message Type 5.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |
| 3 | n | Snapshot Message Payload |

Snapshot Footer Message
This message is encoded as Message Type 6.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |

Stream Data Message
This message is encoded as Message Type 7.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |
| 3 | n | Stream Message Payload |

Streaming Session Start Message
This message is encoded as Message Type 8.

| Byte Offset | Length | Description |
|---|---|---|
| 0 | 1 | Message Type |
| 1 | 2 | Message Length |
| 3 | 8 | Session Identifier |

## Interaction Model

- Upon connecting to the Unicast Gateway the client sends a Login request to the Unicast Gateway.
- The Unicast Gateway validates the Login Request and issues a Login Accepted Response upon successful validation.
- The Unicast Gateway then begins the process of sending a Snapshot by issuing a Snapshot Header message.
- The Unicast Gateway will then send a number of Snapshot Messages
  - The Snapshot begins by sending Instrument Directory messages framed by the Snapshot Message header.
  - The Snapshot then contains Instrument Trading Status messages for each instrument
  - The Snapshot then contains the Trading Session status

- The Snapshot then contains an OrderAdded entry for each order present on the Exchange

- The Unicast Gateway will finally issue a Snapshot Footer Message to denote the completion of the Snapshot.
- The Unicast Gateway will then begin streaming data to the Client through Stream Data messages, these are framed using the Stream Data message.