Deribit Multicast

Client Development Guide

v1.6.2 - 2 December 2023

Table of Contents

1 Introduction

Introducing multicasting of public events on the Deribit colocation network provides a low latency interface to distribute information that also significantly reduces resource requirements on both client and server side. Migrating clients to use multicasted events instead of event subscriptions on the Websocket API does not only allow them to react faster due to the lower latency and more efficient encoding, but also improves the processing speed of trading actions on the API nodes.

2 Multicast packet encoding

The events are sent in UDP multicast packets using the SBE (Simple Binary Encoding) format.

The multicast system groups events per currency and product (perpetual, options, futures) combinations. These groups can be associated with channels (the actual channel associations are maintained and shared in a separate document) where each channel can be assigned a separate multicast group address and UDP destination port number. Each channel keeps a limited history of 100.000 packets for recovery.

After the UDP protocol header, multicast packets start with a framing header that contains the Packet Length, Channel ID and a Sequence Number in the channel (more about Sequence Number in 4.D). The Packet Length is the total length of all the SBE messages (after the Sequence Number) in bytes. The main purpose of this header is to allow clients to detect when they miss packets and retrieve them, using the regular websocket/REST API.

Packet Length (uint16)
Channel ID (uint16)
Sequence Number (uint32)
SBE message
SBE message
SBE message
…

Note that all sample messages provided below, as well as the sample packet capture are from a local development system, with instruments defined solely for the purpose of generating data for testing/developing the multicast decoder. The product/instrument parameters used don’t (and are not intended to) match the production products/instruments.

A - SBE message format intro

SBE is an efficient binary encoding with features that allow e.g. extending the protocol with new elements in a backward compatible way. Also it has a fairly widely used/known standard, with a community behind it providing e.g. tooling for code generation.

Each SBE message contains a header, a number of fixed length fields, groups (variable length lists of items) and optionally variable length fields in this strict sequence.

Header (Fixed size)
Fixed length fields (Fixed size)
Groups
Variable length fields

NOTE : The XML specification is provided in 2 formats from v1.6.1. It is due to discrepancy of how generated code behaves in different languages. Based on feedback from clients, it appears that generated code for some of the languages already assumes the `messageHeader` as part of each message and does not require it to be part of each message section (even requires manual fix). For these there is an XML file provided without the `messageHeader` being part of the message specifications (deribit_multicast_noheader.xml). The other (default) version is for the languages (e.g. CPP) where the code generator needs the header being specified in each message (`deribit_multicast.xml`). For clients using CPP with the code generator tool of Real-Logic or in-house developed tooling that depends on the `messageHeader` specified in the messages, the default XML is recommended.
The XML message snippets in this document are WITHOUT the header included. The same message with header included additionally contains messageHeader as the first field (and subsequent field id's will increase by one).

A.1 SBE Header

```
<composite name="messageHeader">
      <type name="blockLength" primitiveType="uint16" />
      <type name="templateId" primitiveType="uint16" />
      <type name="schemaId" primitiveType="uint16" />
      <type name="version" primitiveType="uint16" />
      <type name="numGroups" primitiveType="uint16" />
      <type name="numVarDataFields" primitiveType="uint16" />
</composite>
```

- `blockLength` is the total number of bytes of the fixed fields in the message. The purpose of this field is that if a new fixed field is introduced (always after the existing fields), the client can decode the fields it is aware of and, based on this field can skip

the additional items to find the beginning of the eventual group/variable fields, or the end of the message
- `templateId` refers to the ID of the corresponding message (e.g. 1001 for order book change)
- The `schemaId` and `version` fields are for indicating the ID of the XML specification and its version number
- `numGroups` is the number of groups/lists in the message. The purpose of this field is similar to the `blockLength` and allows introducing new groups in a backward compatible way
- `numVarDataFields` serve a similar purpose as `blockLength` and `numGroups`

A.2 Fixed length block

The list of single fixed length fields are in this block. Strictly in the sequence as defined in the XML specification. Optional fields are present with a default value, either explicitly mentioned in the XML message/field template, or as defined in the SBE specification for the particular primitive type.

A.3 Groups

Groups are a list of items. Each message can contain 0 or more groups. Each group can have 0 or more items. The items have a similar structure as SBE messages. They can consist of a fixed length block, (nested) groups and variable length fields. The information about the structure of the items, as well as the number of items (`numInGroup`) is described in the header of each group.

The group header definition:

```
<composite name="groupSizeEncoding">
      <type name="blockLength" primitiveType="uint16"/>
      <type name="numInGroup" primitiveType="uint16"/>
      <type name="numGroups" primitiveType="uint16"/>
      <type name="numVarDataFields" primitiveType="uint16"/>
</composite>
```

Note that if the the group header indicates that the entries do not have nested groups (`numGroups` = 0) or variable length data fields (`numVarDataFields` = 0) then the entries have a fixed length which is equal to blockLength. In this case the total number of bytes in the group can be calculated by : `blockLength * numInGroup`.

A.4 Variable length fields

Variable length fields begin with a length and a length number of bytes. In the current implementation variable fields are used for Instrument names in instrument events/messages.

```
<composite name="varString" description="Variable-length string">
      <type name="length" primitiveType="uint8" />
      <type name="varData" length="0" primitiveType="uint8" />
</composite>
```

A.5 Floats instead of decimals

The SBE specification recommends using decimals for price/amount representation, however since in this use case due to the broad variety of instruments, we chose double (64 bit float) to simplify the interface. This means that the inherent property of floats that not all numbers can be represented exactly should be taken in consideration when using those values for calculation. Since this inherent float “inaccuracy” happens after 15 decimals, it is recommended that the client rounds the received values e.g. to 9 decimals.

3 Multicast events

Since version 1.6 the platform multicasts 4 events

- instrument
- instrument v2
- order book change
- trades
- ticker
- snapshots
- snapshot start
- snapshot end
- combo legs
- price index
- rfq
- spots

The structure and content of the events is aimed to be the same or very similar to the current Websocket API subscription events.

A - Instrument

The purpose of this event is to enable the clients to get notified when a new instrument/book is created/closed and settled. Also to provide static instrument information in snapshots. Based on this event, the client can start tracking a new book, or stop the tracking of one.

This event can/should also be used to maintain the Instrument ID/Name mapping, since it contains both of this information.

When instrument changes happen in a batch (e.g. closing option books) and multiple messages fit in a packet then they can be combined, even for different instruments for a combination of currency/product. When books are closed, then book change events (deleting the remaining levels) and instrument state change (close) events may be combined.

The instrument message is also sent on the snapshot channels as part of the snapshot, providing static instrument information.

Combos

Combo instruments are new instrument kinds and since they can be deactivated/reactivated before closing, they have more states than other instruments.

The combos have their own multicast channels. Combo channel assignments are in the Deribit Multicast Channels document (v1.1 or above).

Changes in the instrument message (new instrument kinds and new states) only impact the combo channels.

When a combo is deactivated, an instrument message is sent with the state change “deactivated”, and the static state of the combo instrument (e.g. in ticker or instrument snapshot) changes to “inactive”.

When a combo book is reactivated, after deactivation, an instrument message is sent with the state change “started”, and the static state of the combo instrument changes to “open”.

Message specification

```
<message name="instrument" id="1000">
      <field name="instrumentId" id ="1" type="uint32" />
      <field name="instrumentState" id="2" type="instrumentState" />
      <field name="kind" id="3" type="instrumentKind" />
      <field name="instrumentType" id="4" type="instrumentType" />
      <field name="optionType" id="5" type="optionType" />
      <field name="rfq" id="6" type="yesNo" />
      <field name="settlementPeriod" id="7" type="period"
            presence="optional" />
      <field name="settlementPeriodCount" id="8" type="uint16" />
      <field name="baseCurrency" id="9" type="string8" />
      <field name="quoteCurrency" id="10" type="string8" />
      <field name="counterCurrency" id="11" type="string8" />
      <field name="settlementCurrency" id="12" type="string8" />
      <field name="sizeCurrency" id="13" type="string8" />
      <field name="creationTimestampMs" id="14" type="uint64" />
      <field name="expirationTimestampMs" id="15" type="uint64" />
```

```
      <field name="strikePrice" id="16" type="double"
            presence="optional" />
      <field name="contractSize" id="17" type="double" />
      <field name="minTradeAmount" id="18" type="double" />
      <field name="tickSize" id="19" type="double" />
      <field name="makerCommission" id="20" type="double" />
      <field name="takerCommission" id="21" type="double" />
      <field name="blockTradeCommission" id="22" type="double"
            presence="optional" />
      <field name="maxLiquidationCommission" id="23" type="double"
            presence="optional" />
      <field name="maxLeverage" id="24" type="double"
            presence="optional" />
      <data name="instrumentName" id="25" type="varString" />
</message>
```

Enum Types used in the message

```
<enum name="instrumentState" encodingType="uint8">
      <validValue name="created">0</validValue>
      <validValue name="open">1</validValue>
      <validValue name="closed">2</validValue>
      <validValue name="settled">3</validValue>
      <validValue name="deactivated">4</validValue>
      <validValue name="inactive">5</validValue>
      <validValue name="started">6</validValue>
</enum>
<enum name="instrumentKind" encodingType="uint8">
      <validValue name="future">0</validValue>
      <validValue name="option">1</validValue>
      <validValue name="future_combo">2</validValue>
      <validValue name="option_combo">3</validValue>
      <validValue name="spot">4</validValue>
</enum>
<enum name="optionType" encodingType="uint8">
      <validValue name="not_applicable">0</validValue>
      <validValue name="call">1</validValue>
      <validValue name="put">2</validValue>
</enum>
<enum name="instrumentType" encodingType="uint8">
      <validValue name="not_applicable">0</validValue>
      <validValue name="reversed">1</validValue>
      <validValue name="linear">2</validValue>
</enum>
<enum name="period" encodingType="uint8">
      <validValue name="perpetual">0</validValue>
      <validValue name="minute">1</validValue>
      <validValue name="hour">2</validValue>
      <validValue name="day">3</validValue>
      <validValue name="week">4</validValue>
      <validValue name="month">5</validValue>
      <validValue name="year">6</validValue>
```

```
</enum>
```

Example message

```
Frame 4096: 227 bytes on wire (1816 bits), 227 bytes captured (1816 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.2
User Datagram Protocol, Src Port: 48210, Dst Port: 6100
Deribit SBE
    Framing Header
        Packet Length: 177
        Channel Id: 2
        Channel Sequence: 1
    Instrument State
        Header
            Root Block Length: 140
            Type: instrument (1000)
            Schema Id: 1
            Version: 1
            Num Groups: 0
            Num Vars: 1
        Instrument Id: 618
        Instrument State: created (0)
        Instrument Kind: option (1)
        Instrument Type: not applicable (0)
        Option Type: call (1)
        RFQ: 0
        Settlement Period: minute (1)
        Settlement Period Count: 15
        Base Currency: BTC
        Quote Currency: BTC
        Counter Currency: USD
        Settlement Currency: BTC
        Size Currency: BTC
        Creation Timestamp: May 14, 2022 06:35:05.000000000 UTC
        Expiration Timestamp: May 14, 2022 06:45:00.000000000 UTC
        Strike Price: 29200
        Contract Size: 1
        Minimum Trade Amount: 0,01
        Tick Size: 0,0001
        Maker Commission: 0,0001
        Taker Commission: 0,0005
        Block Trade Commission: 0,00015
        Max Liquidation Commission: 0
        Max Leverage: 0
        Instrument Name: BTC-14MAY22_0645-29200-C

0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   00 d5 bc fc 40 00 20 11 bf a8 7f 00 00 01 ef 6f   ....@. ........o
0020   6f 02 bc 52 17 d4 00 c1 de 45 b1 00 02 00 01 00   o..R.....E......
0030   00 00 8c 00 e8 03 01 00 01 00 00 00 01 00 6a 02   ..............j.
```

```
0040   00 00 00 01 00 01 00 01 0f 00 42 54 43 00 00 00   ..........BTC...
0050   00 00 42 54 43 00 00 00 00 00 55 53 44 00 00 00   ..BTC.....USD...
0060   00 00 42 54 43 00 00 00 00 00 42 54 43 00 00 00   ..BTC.....BTC...
0070   00 00 a8 1d 47 c1 80 01 00 00 e0 31 50 c1 80 01   ....G......1P...
0080   00 00 00 00 00 00 00 84 dc 40 00 00 00 00 00 00   .........@......
0090   f0 3f 7b 14 ae 47 e1 7a 84 3f 2d 43 1c eb e2 36   .?{..G.z.?-C...6
00a0   1a 3f 2d 43 1c eb e2 36 1a 3f fc a9 f1 d2 4d 62   .?-C...6.?....Mb
00b0   40 3f 61 32 55 30 2a a9 23 3f 00 00 00 00 00 00   @?a2U0*.#?......
00c0   00 00 00 00 00 00 00 00 00 00 18 42 54 43 2d 31   ...........BTC-1
00d0   34 4d 41 59 32 32 5f 30 36 34 35 2d 32 39 32 30   4MAY22_0645-2920
00e0   30 2d 43                                          0-C
```

B - Instrument v2

With the introduction of multiple tick sizes, we need to add the information about the tick size steps, so clients can place orders with the correct tick sizes in the different price ranges. Although SBE provides a backward compatibility mechanism for adding new fields, the choice to introduce a new message was made based on the assumption that client implementations are more likely able to ignore unknown message types than implement the backward compatibility mechanism of SBE. A second reason for this is that the new message allows us to remove the deprecated “rfq” field. The recently introduced RFQ message provides a much more reliable and comprehensive mechanism for RFQ status tracking. SBE doesn’t support removing fields from a message in a backward compatible way, so a new message is the only way to do it.
The intention is that both messages will be sent out in instrument state changes and snapshots for a while. After sufficient time and timely announcement to allow clients to migrate to using this new message, the original instrument message will be removed from the multicast.
We intend to use this mechanism in the future to make changes in the messages.

Message specification

```
<message name="instrumentV2" id="1010" sinceVersion="3">
      <field name="instrumentId" id ="1" type="uint32" />
      <field name="instrumentState" id="2" type="instrumentState" />
      <field name="kind" id="3" type="instrumentKind" />
      <field name="instrumentType" id="4" type="instrumentType" />
      <field name="optionType" id="5" type="optionType" />
      <field name="settlementPeriod" id="6" type="period"
            presence="optional" />
      <field name="settlementPeriodCount" id="7" type="uint16" />
      <field name="baseCurrency" id="8" type="string8" />
      <field name="quoteCurrency" id="9" type="string8" />
      <field name="counterCurrency" id="10" type="string8" />
      <field name="settlementCurrency" id="11" type="string8" />
      <field name="sizeCurrency" id="12" type="string8" />
      <field name="creationTimestampMs" id="13" type="uint64" />
      <field name="expirationTimestampMs" id="14" type="uint64" />
      <field name="strikePrice" id="15" type="double"
            presence="optional" />
```

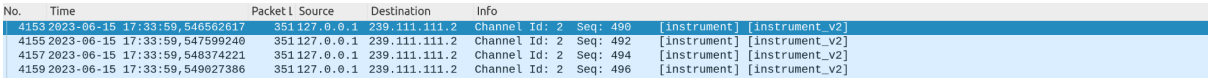
```
        <field name="contractSize" id="16" type="double" />
        <field name="minTradeAmount" id="17" type="double" />
        <field name="tickSize" id="18" type="double" />
        <field name="makerCommission" id="19" type="double" />
        <field name="takerCommission" id="20" type="double" />
        <field name="blockTradeCommission" id="21" type="double"
              presence="optional" />
        <field name="maxLiquidationCommission" id="22" type="double"
              presence="optional" />
        <field name="maxLeverage" id="23" type="double"
              presence="optional" />
        <group name="tickStepsList" id="24"
              dimensionType="groupSizeEncoding">
              <field name="abovePrice" id="1" type="double" />
              <field name="tickSize" id="2" type="double" />
        </group>
        <data name="instrumentName" id="25" type="varString" />
</message>
```

The same enum types are used in this message as in the original instrument message.

Example sequence

The instrumentV2 message is inserted right after the the instrument message in snapshot sequence and in instrument state change events.

No.	Time	Packet l	Source	Destination	Info
1419	2023-06-15 18:46:47,055054573	16	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 0 [snapshot_start]
1421	2023-06-15 18:46:47,256813576	1273	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 1 [instrument] [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument]
1422	2023-06-15 18:46:47,256823278	1425	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 2 [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker]
1423	2023-06-15 18:46:47,256827618	1315	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 3 [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument]
1424	2023-06-15 18:46:47,256831120	1425	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 4 [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker]
1425	2023-06-15 18:46:47,256834285	1315	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 5 [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument]
1426	2023-06-15 18:46:47,256837482	1425	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 6 [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker]
1427	2023-06-15 18:46:47,256840431	1315	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 7 [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument]
1428	2023-06-15 18:46:47,256843518	1425	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 8 [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker] [snapshot] [instrument] [instrument_v2] [ticker]
1429	2023-06-15 18:46:47,256846685	42	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 9 [snapshot]
1430	2023-06-15 18:46:47,256851904	12	127.0.0.1	239.111.111.11	Channel Id: 111 Seq: 10 [snapshot_end]



No.	Time	Packet l	Source	Destination	Info
4153	2023-06-15 17:33:59,546562617	351	127.0.0.1	239.111.111.2	Channel Id: 2 Seq: 490 [instrument] [instrument_v2]
4155	2023-06-15 17:33:59,547599240	351	127.0.0.1	239.111.111.2	Channel Id: 2 Seq: 492 [instrument] [instrument_v2]
4157	2023-06-15 17:33:59,548374221	351	127.0.0.1	239.111.111.2	Channel Id: 2 Seq: 494 [instrument] [instrument_v2]
4159	2023-06-15 17:33:59,549027386	351	127.0.0.1	239.111.111.2	Channel Id: 2 Seq: 496 [instrument] [instrument_v2]

Example message

```
Frame 12890: 417 bytes on wire (3336 bits), 417 bytes captured (3336 bits) on
interface lo, id 0
    Interface id: 0 (lo)
        Interface name: lo
    Encapsulation type: Ethernet (1)
    Arrival Time: Jun 15, 2023 20:51:00.018508968 CEST
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1686855060.018508968 seconds
    [Time delta from previous captured frame: 0.001292339 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 283.628602527 seconds]
    Frame Number: 12890
    Frame Length: 417 bytes (3336 bits)
    Capture Length: 417 bytes (3336 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:udp:deribit_sbe]
```

```
    [Coloring Rule Name: UDP]
    [Coloring Rule String: udp]
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
    Destination: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address
(factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Source: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address
(factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
        0000 00.. = Differentiated Services Codepoint: Default (0)
        .... ..00 = Explicit Congestion Notification: Not ECN-Capable
Transport (0)
    Total Length: 403
    Identification: 0x5ba3 (23459)
    Flags: 0x40, Don't fragment
        0... .... = Reserved bit: Not set
        .1.. .... = Don't fragment: Set
        ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 32
    Protocol: UDP (17)
    Header Checksum: 0x2044 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 127.0.0.1
    Destination Address: 239.111.111.2
User Datagram Protocol, Src Port: 32859, Dst Port: 6100
    Source Port: 32859
    Destination Port: 6100
    Length: 383
    Checksum: 0xdf03 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 2]
    [Timestamps]
        [Time since first frame: 283.625157436 seconds]
        [Time since previous frame: 0.001292339 seconds]
    UDP payload (375 bytes)
Deribit SBE
    Framing Header
        Packet Length: 367
        Channel Id: 2
        Channel Sequence: 3145
    Instrument State
        Header
```

```
        Root Block Length: 140
        Type: instrument (1000)
        Schema Id: 1
        Version: 2
        Num Groups: 0
        Num Vars: 1
    Instrument Id: 77
    Instrument State: created (0)
    Instrument Kind: option (1)
    Instrument Type: reversed (1)
    Option Type: call (1)
    RFQ: 0
    Settlement Period: week (4)
    Settlement Period Count: 1
    Base Currency: BTC
    Quote Currency: BTC
    Counter Currency: USD
    Settlement Currency: BTC
    Size Currency: BTC
    Creation Timestamp: Jun 15, 2023 18:51:00.000000000 UTC
    Expiration Timestamp: Jun 23, 2023 08:00:00.000000000 UTC
    Strike Price: 25500
    Contract Size: 1
    Minimum Trade Amount: 0,01
    Tick Size: 0,0001
    Maker Commission: 0
    Taker Commission: 0,0001
    Block Trade Commission: 0,00015
    Max Liquidation Commission: -nan
    Max Leverage: -nan
    Instrument Name: BTC-23JUN23-25500-C
Instrument State v2
    Header
        Root Block Length: 139
        Type: instrument_v2 (1010)
        Schema Id: 1
        Version: 3
        Num Groups: 1
        Num Vars: 1
    Instrument Id: 77
    Instrument State: created (0)
    Instrument Kind: option (1)
    Instrument Type: reversed (1)
    Option Type: call (1)
    Settlement Period: week (4)
    Settlement Period Count: 1
    Base Currency: BTC
    Quote Currency: BTC
    Counter Currency: USD
    Settlement Currency: BTC
    Size Currency: BTC
    Creation Timestamp: Jun 15, 2023 18:51:00.000000000 UTC
    Expiration Timestamp: Jun 23, 2023 08:00:00.000000000 UTC
```

```
Strike Price: 25500
Contract Size: 1
Minimum Trade Amount: 0,01
Tick Size: 0,0001
Maker Commission: 0
Taker Commission: 0,0001
Block Trade Commission: 0,00015
Max Liquidation Commission: -nan
Max Leverage: -nan
Tick Size Steps
    Group Header
        Group Block Length: 16
        Num In Group: 1
        Group Num Groups: 0
        Group Num Vars: 0
    Tick Step List
         above price : 0,001 tick_size : 0,0002
Instrument Name: BTC-23JUN23-25500-C
```

```
0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   01 93 5b a3 40 00 20 11 20 44 7f 00 00 01 ef 6f   ..[.@. . D.....o
0020   6f 02 80 5b 17 d4 01 7f df 03 6f 01 02 00 49 0c   o..[......o...I.
0030   00 00 8c 00 e8 03 01 00 02 00 00 00 01 00 4d 00   ..............M.
0040   00 00 00 01 01 01 00 04 01 00 42 54 43 00 00 00   ..........BTC...
0050   00 00 42 54 43 00 00 00 00 00 55 53 44 00 00 00   ..BTC.....USD...
0060   00 00 42 54 43 00 00 00 00 00 42 54 43 00 00 00   ..BTC.....BTC...
0070   00 00 20 8a 65 c0 88 01 00 00 00 68 44 e7 88 01   .. .e......hD...
0080   00 00 00 00 00 00 00 e7 d8 40 00 00 00 00 00 00   .........@......
0090   f0 3f 7b 14 ae 47 e1 7a 84 3f 2d 43 1c eb e2 36   .?{..G.z.?-C...6
00a0   1a 3f 00 00 00 00 00 00 00 00 2d 43 1c eb e2 36   .?........-C...6
00b0   1a 3f 61 32 55 30 2a a9 23 3f ff ff ff ff ff ff   .?a2U0*.#?......
00c0   ff ff ff ff ff ff ff ff ff ff 13 42 54 43 2d 32   ...........BTC-2
00d0   33 4a 55 4e 32 33 2d 32 35 35 30 30 2d 43 8b 00   3JUN23-25500-C..
00e0   f2 03 01 00 03 00 01 00 01 00 4d 00 00 00 00 01   ..........M.....
00f0   01 01 04 01 00 42 54 43 00 00 00 00 00 42 54 43   .....BTC.....BTC
0100   00 00 00 00 00 55 53 44 00 00 00 00 00 42 54 43   .....USD.....BTC
0110   00 00 00 00 00 42 54 43 00 00 00 00 00 20 8a 65   .....BTC..... .e
0120   c0 88 01 00 00 00 68 44 e7 88 01 00 00 00 00 00   ......hD........
0130   00 00 e7 d8 40 00 00 00 00 00 00 f0 3f 7b 14 ae   ....@.......?{..
0140   47 e1 7a 84 3f 2d 43 1c eb e2 36 1a 3f 00 00 00   G.z.?-C...6.?...
0150   00 00 00 00 00 2d 43 1c eb e2 36 1a 3f 61 32 55   .....-C...6.?a2U
0160   30 2a a9 23 3f ff ff ff ff ff ff ff ff ff ff ff   0*.#?...........
0170   ff ff ff ff ff 10 00 01 00 00 00 00 00 fc a9 f1   ................
0180   d2 4d 62 50 3f 2d 43 1c eb e2 36 2a 3f 13 42 54   .MbP?-C...6*?.BT
0190   43 2d 32 33 4a 55 4e 32 33 2d 32 35 35 30 30 2d   C-23JUN23-25500-
01a0   43                                                C
```

C - Order book change

This event contains changes of the order book levels. The changes can be new levels, change in the amount on the levels or deletion (no amount left), just as in the websocket subscription events.

Combos

Order book changes for combos are the same as for normal instruments. However, since trades in combos change the positions (but not the order book levels) on the individual instruments that are part of the combo (a.k.a. legs), combo trades will generate book change events with empty level changes on the legs.
Likewise, a position move initiated by an admin may also trigger a book change event with empty level changes.

Message Specification

```
<message name="book" id="1001">
      <field name="instrumentId" id="1" type="uint32" />
      <field name="timestampMs" id="2" type="uint64" />
      <field name="prevChangeId" id="3" type="uint64" />
      <field name="changeId" id="4" type="uint64" />
      <field name="isLast" id="5" type="yesNo" />
      <group name="changesList" id="6" dimensionType="groupSizeEncoding">
            <field name="side" id="1" type="bookSide" />
            <field name="change" id="2" type="bookChange" />
            <!-- Use double (64 bit float) encoding,
            SBE FIX price/amount decimal encoding makes sense when the
            decimal point is on a somewhat fixed position, for crypto,
            it can vary more by instrument -->
            <field name="price" id="3" type="double" />
            <field name="amount" id="4" type="double" />
      </group>
</message>
```

Enum types used in this message

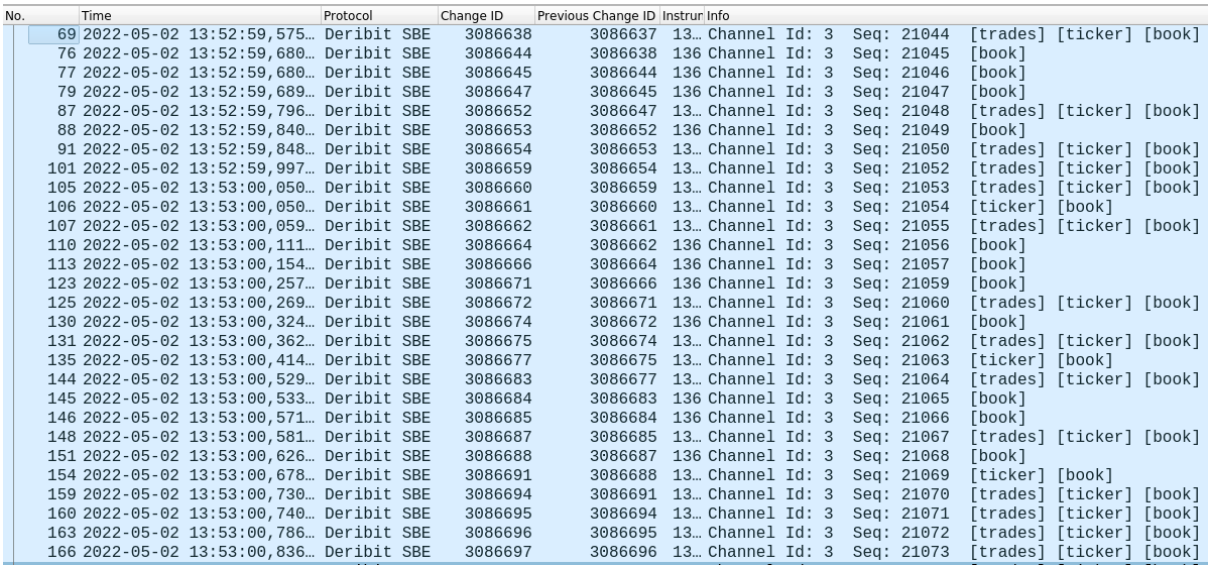
```
<enum name="bookChange" encodingType="uint8">
      <validValue name="created">0</validValue>
      <validValue name="changed">1</validValue>
      <validValue name="deleted">2</validValue>
</enum>

<enum name="bookSide" encodingType="uint8">
      <validValue name="ask">0</validValue>
      <validValue name="bid">1</validValue>
</enum>

<enum name="yesNo" encodingType="uint8">
      <validValue name="no">0</validValue>
      <validValue name="yes">1</validValue>
</enum>
```

The events use the same mechanism to ensure the consistency of the order book as the websocket events.

Each message contains a current `changeId` and `prevChangeId`. When an event is received, its `prevChangeId` should be compared with the `changeId` of the last seen event in the book. It should be assumed that the change id is just a not strictly consecutive monotonically increasing number.

The list of book changes may occasionally result in a message that exceeds the maximum packet size. In this case the platform splits the list of changes into multiple lists and generates a sequence of multiple complete order book change messages. The messages have the same fixed fields (`header`, `timestampMs`, `prevChangeId`, `changeId`) but the `isLast` field is set to the value 0 for all but the last message in the sequence, to indicate if the complete change list is sent.

Example sequence



No.	Time	Protocol	Change ID	Previous Change ID	Instrur	Info
69	2022-05-02 13:52:59,575…	Deribit SBE	3086638	3086637	13…	Channel Id: 3 Seq: 21044 [trades] [ticker] [book]
76	2022-05-02 13:52:59,680…	Deribit SBE	3086644	3086638	136	Channel Id: 3 Seq: 21045 [book]
77	2022-05-02 13:52:59,680…	Deribit SBE	3086645	3086644	136	Channel Id: 3 Seq: 21046 [book]
79	2022-05-02 13:52:59,689…	Deribit SBE	3086647	3086645	136	Channel Id: 3 Seq: 21047 [book]
87	2022-05-02 13:52:59,796…	Deribit SBE	3086652	3086647	13…	Channel Id: 3 Seq: 21048 [trades] [ticker] [book]
88	2022-05-02 13:52:59,840…	Deribit SBE	3086653	3086652	136	Channel Id: 3 Seq: 21049 [book]
91	2022-05-02 13:52:59,848…	Deribit SBE	3086654	3086653	13…	Channel Id: 3 Seq: 21050 [trades] [ticker] [book]
101	2022-05-02 13:52:59,997…	Deribit SBE	3086659	3086654	13…	Channel Id: 3 Seq: 21052 [trades] [ticker] [book]
105	2022-05-02 13:53:00,050…	Deribit SBE	3086660	3086659	13…	Channel Id: 3 Seq: 21053 [trades] [ticker] [book]
106	2022-05-02 13:53:00,050…	Deribit SBE	3086661	3086660	13…	Channel Id: 3 Seq: 21054 [ticker] [book]
107	2022-05-02 13:53:00,059…	Deribit SBE	3086662	3086661	13…	Channel Id: 3 Seq: 21055 [trades] [ticker] [book]
110	2022-05-02 13:53:00,111…	Deribit SBE	3086664	3086662	136	Channel Id: 3 Seq: 21056 [book]
113	2022-05-02 13:53:00,154…	Deribit SBE	3086666	3086664	136	Channel Id: 3 Seq: 21057 [book]
123	2022-05-02 13:53:00,257…	Deribit SBE	3086671	3086666	136	Channel Id: 3 Seq: 21059 [book]
125	2022-05-02 13:53:00,269…	Deribit SBE	3086672	3086671	13…	Channel Id: 3 Seq: 21060 [trades] [ticker] [book]
130	2022-05-02 13:53:00,324…	Deribit SBE	3086674	3086672	136	Channel Id: 3 Seq: 21061 [book]
131	2022-05-02 13:53:00,362…	Deribit SBE	3086675	3086674	13…	Channel Id: 3 Seq: 21062 [trades] [ticker] [book]
135	2022-05-02 13:53:00,414…	Deribit SBE	3086677	3086675	13…	Channel Id: 3 Seq: 21063 [ticker] [book]
144	2022-05-02 13:53:00,529…	Deribit SBE	3086683	3086677	13…	Channel Id: 3 Seq: 21064 [trades] [ticker] [book]
145	2022-05-02 13:53:00,533…	Deribit SBE	3086684	3086683	136	Channel Id: 3 Seq: 21065 [book]
146	2022-05-02 13:53:00,571…	Deribit SBE	3086685	3086684	136	Channel Id: 3 Seq: 21066 [book]
148	2022-05-02 13:53:00,581…	Deribit SBE	3086687	3086685	13…	Channel Id: 3 Seq: 21067 [trades] [ticker] [book]
151	2022-05-02 13:53:00,626…	Deribit SBE	3086688	3086687	136	Channel Id: 3 Seq: 21068 [book]
154	2022-05-02 13:53:00,678…	Deribit SBE	3086691	3086688	13…	Channel Id: 3 Seq: 21069 [ticker] [book]
159	2022-05-02 13:53:00,730…	Deribit SBE	3086694	3086691	13…	Channel Id: 3 Seq: 21070 [trades] [ticker] [book]
160	2022-05-02 13:53:00,740…	Deribit SBE	3086695	3086694	13…	Channel Id: 3 Seq: 21071 [trades] [ticker] [book]
163	2022-05-02 13:53:00,786…	Deribit SBE	3086696	3086695	13…	Channel Id: 3 Seq: 21072 [trades] [ticker] [book]
166	2022-05-02 13:53:00,836…	Deribit SBE	3086697	3086696	13…	Channel Id: 3 Seq: 21073 [trades] [ticker] [book]

Example message

```
Frame 220: 135 bytes on wire (1080 bits), 135 bytes captured (1080 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.3
User Datagram Protocol, Src Port: 51012, Dst Port: 6100
Deribit SBE
    Framing Header
        Packet Length: 85
        Channel Id: 3
        Channel Sequence: 21093
    Book Change
        Header
            Root Block Length: 29
            Type: book (1001)
            Schema Id: 1
            Version: 1
            Num Groups: 1
            Num Vars: 0
        Instrument Id: 136
        Timestamp: May  2, 2022 11:53:01.475000000 UTC
        Previous Change ID: 3086730
        Change ID: 3086733
        Is Last Part: last (1)
        Changes
            Group Header
                Group Block Length: 18
                Num In Group: 2
```

```
                Group Num Groups: 0
                Group Num Vars: 0
            Change List
                delete bid - price : 35171,99 amount : 0
                new bid - price : 36930,58 amount : 40

0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   00 79 7b 98 40 00 20 11 01 68 7f 00 00 01 ef 6f   .y{.@. ..h.....o
0020   6f 03 c7 44 17 d4 00 65 dd ea 55 00 03 00 65 52   o..D...e..U...eR
0030   00 00 1d 00 e9 03 01 00 01 00 01 00 00 00 88 00   ................
0040   00 00 23 e3 9d 84 80 01 00 00 8a 19 2f 00 00 00   ..#........./...
0050   00 00 8d 19 2f 00 00 00 00 00 01 12 00 02 00 00   ..../...........
0060   00 00 00 01 02 e1 7a 14 ae 7f 2c e1 40 00 00 00   ......z...,.@...
0070   00 00 00 00 00 01 00 f6 28 5c 8f 52 08 e2 40 00   ........(\.R..@.
0080   00 00 00 00 00 44 40                              .....D@
```

D - Trades

In case an order results in one or more trades, these are sent in a trades event containing one or more trades related to the same instrument and order.

The meaning/purpose of the fields are the same as the corresponding websocket API event.

Combos

Since combo trades change positions in the legs, trades of combos are also visible on the instrument (leg) streams that are part of the combo. The trades on the leg level will have their “comboTradeId” field set to the “tradeId” of the combo instrument trade.

Message specification

```
<message name="trades" id="1002">
      <field name="instrumentId" id="1" type="uint32" />
      <group name="tradesList" id="2" dimensionType="groupSizeEncoding">
            <field name="direction" id="1" type="direction" />
            <field name="price" id="2" type="double" />
            <field name="amount" id="3" type="double" />
            <field name="timestampMs" id="4" type="uint64" />
            <field name="markPrice" id="5" type="double" />
            <field name="indexPrice" id="6" type="double" />
            <field name="tradeSeq" id="7" type="uint64" />
            <field name="tradeId" id="8" type="uint64" />
            <field name="tickDirection" id="9" type="tickDirection" />
            <field name="liquidation" id="10" type="liquidation" />
            <field name="iv" id="11" type="double"
                  presence="optional" />
            <field name="blockTradeId" id="12" type="uint64"
```

```
                presence="optional" nullValue="0" />
            <field name="comboTradeId" id="13" type="uint64"
                presence="optional" nullValue="0" />
    </group>
</message>
```

Enum types used in the message

```
<enum name="tickDirection" encodingType="uint8">
      <validValue name="plus">0</validValue>
      <validValue name="zeroplus">1</validValue>
      <validValue name="minus">2</validValue>
      <validValue name="zerominus">3</validValue>
</enum>

<enum name="direction" encodingType="uint8">
      <validValue name="buy">0</validValue>
      <validValue name="sell">1</validValue>
</enum>

<enum name="liquidation" encodingType="uint8">
      <validValue name="none">0</validValue>
      <validValue name="maker">1</validValue>
      <validValue name="taker">2</validValue>
      <validValue name="both">3</validValue>
</enum>
```

The event system combines the events related to the same transaction on an instrument. If these events fit in a single packet, the related `book`, `trades` and `ticker` messages will be combined in a packet.

Example message

```
Frame 181: 470 bytes on wire (3760 bits), 470 bytes captured (3760 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.1
User Datagram Protocol, Src Port: 55022, Dst Port: 6100
Deribit SBE
    Framing Header
        Packet Length: 420
        Channel Id: 1
        Channel Sequence: 10477
    Trades
        Header
            Root Block Length: 4
            Type: trades (1002)
            Schema Id: 1
            Version: 1
            Num Groups: 1
```

```
        Num Vars: 0
    Instrument Id: 1
    Trades
        Group Header
            Group Block Length: 83
            Num In Group: 2
            Group Num Groups: 0
            Group Num Vars: 0
        Trade list
            Trade
                Trade Direction: sell (1)
                Price: 39344,25
                Amount: 10
                Timestamp: May  2, 2022 11:53:01.000000000 UTC
                Mark Price: 38815,96
                Index Price: 38603,64
                Trade Sequence: 392167
                Trade Id: 1297362
                Tick Direction: zerominus (3)
                Liquidation: no liquidation (0)
                Implied Volatility: 0
                Block Trade Id: 0
                Combo Trade Id: 0
            Trade
                Trade Direction: sell (1)
                Price: 39316,72
                Amount: 10
                Timestamp: May  2, 2022 11:53:01.000000000 UTC
                Mark Price: 38815,96
                Index Price: 38603,64
                Trade Sequence: 392168
                Trade Id: 1297363
                Tick Direction: minus (2)
                Liquidation: no liquidation (0)
                Implied Volatility: 0
                Block Trade Id: 0
                Combo Trade Id: 0
Ticker
    Header
        Root Block Length: 133
        Type: ticker (1003)
        Schema Id: 1
        Version: 1
        Num Groups: 0
        Num Vars: 0
    Instrument Id: 1
    Instrument State: open (1)
    Timestamp: May  2, 2022 11:53:01.000000000 UTC
    Open Interest: 60
    Min Price: 38980,56
    Max Price: 39375,71
    Last Price: 39316,72
    Index Price: 38603,64
```

```
        Mark Price: 38815,96
        Best Bid Price: 39316,31
        Best Bid Amount: 30
        Best Ask Price: 39375,72
        Best Ask Amount: 10
        Current Funding: 0,005
        Funding 8h: 0,00751341
        Estimated Delivery Price: 38603,64
        Delivery Price: 0
        Settlement Price: 39229,82
    Book Change
        Header
            Root Block Length: 29
            Type: book (1001)
            Schema Id: 1
            Version: 1
            Num Groups: 1
            Num Vars: 0
        Instrument Id: 1
        Timestamp: May  2, 2022 11:53:01.000000000 UTC
        Previous Change ID: 3086708
        Change ID: 3086709
        Is Last Part: last (1)
        Changes
            Group Header
                Group Block Length: 18
                Num In Group: 2
                Group Num Groups: 0
                Group Num Vars: 0
            Change List
                delete bid - price : 39316,72 amount : 0
                delete bid - price : 39344,25 amount : 0
```

```
0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   01 c8 bd 2f 40 00 20 11 be 83 7f 00 00 01 ef 6f   .../@. ........o
0020   6f 01 d6 ee 17 d4 01 b4 df 37 a4 01 01 00 ed 28   o........7.....(
0030   00 00 04 00 ea 03 01 00 01 00 01 00 00 00 01 00   ................
0040   00 00 53 00 02 00 00 00 00 00 01 00 00 00 00 08   ..S.............
0050   36 e3 40 00 00 00 00 00 00 24 40 48 e1 9d 84 80   6.@......$@H....
0060   01 00 00 85 eb 51 b8 fe f3 e2 40 ae 47 e1 7a 74   .....Q....@.G.zt
0070   d9 e2 40 e7 fb 05 00 00 00 00 00 d2 cb 13 00 00   ..@.............
0080   00 00 00 03 00 00 00 00 00 00 00 00 00 00 00 00   ................
0090   00 00 00 00 00 00 00 00 00 00 00 00 00 01 a4 70   ...............p
00a0   3d 0a 97 32 e3 40 00 00 00 00 00 00 24 40 48 e1   =..2.@......$@H.
00b0   9d 84 80 01 00 00 85 eb 51 b8 fe f3 e2 40 ae 47   ........Q....@.G
00c0   e1 7a 74 d9 e2 40 e8 fb 05 00 00 00 00 00 d3 cb   .zt..@..........
00d0   13 00 00 00 00 00 02 00 00 00 00 00 00 00 00 00   ................
00e0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00f0   85 00 eb 03 01 00 01 00 00 00 00 00 01 00 00 00   ................
0100   01 48 e1 9d 84 80 01 00 00 00 00 00 00 00 00 4e   .H.............N
0110   40 b8 1e 85 eb 91 08 e3 40 85 eb 51 b8 f6 39 e3   @.......@..Q..9.
0120   40 a4 70 3d 0a 97 32 e3 40 ae 47 e1 7a 74 d9 e2   @.p=..2.@.G.zt..
0130   40 85 eb 51 b8 fe f3 e2 40 b8 1e 85 eb 89 32 e3   @..Q....@.....2.
```

```
0140   40 00 00 00 00 00 00 3e 40 a4 70 3d 0a f7 39 e3   @......>@.p=..9.
0150   40 00 00 00 00 00 00 24 40 7b 14 ae 47 e1 7a 74   @......$@{..G.zt
0160   3f c2 f9 b3 a3 61 c6 7e 3f ae 47 e1 7a 74 d9 e2   ?....a.~?.G.zt..
0170   40 00 00 00 00 00 00 00 00 d7 a3 70 3d ba 27 e3   @..........p=.'.
0180   40 1d 00 e9 03 01 00 01 00 01 00 00 00 01 00 00   @...............
0190   00 48 e1 9d 84 80 01 00 00 74 19 2f 00 00 00 00   .H.......t./....
01a0   00 75 19 2f 00 00 00 00 00 01 12 00 02 00 00 00   .u./............
01b0   00 00 01 02 a4 70 3d 0a 97 32 e3 40 00 00 00 00   .....p=..2.@....
01c0   00 00 00 00 01 02 00 00 00 00 08 36 e3 40 00 00   ...........6.@..
01d0   00 00 00 00 00 00                                 ......
```

E - Ticker

The ticker event is sent periodically to indicate changes in the book data that are not strictly level (e.g. index price). The event can be related to a book level change (e.g. best bid/ask changes) in which case it is sent together with the corresponding book change. It can also be sent as a standalone event.

The ticker message structure is also used on the snapshot channels

Message specification

```
<message name="ticker" id="1003">
      <!-- according the the SBE spec, optional floats use the
      quietNaN null value 0xffffffffffffffff -->
      <field name="instrumentId" id="1" type="uint32" />
      <field name="instrumentState" id="2" type="instrumentState" />
      <field name="timestampMs" id="3" type="uint64" />
      <field name="openInterest" id="4" type="double"
            presence="optional" />
      <field name="minSellPrice" id="5" type="double" />
      <field name="maxBuyPrice" id="6" type="double" />
      <field name="lastPrice" id="7" type="double"
            presence="optional"/>
      <field name="indexPrice" id="8" type="double" />
      <field name="markPrice" id="9" type="double" />
      <field name="bestBidPrice" id="10" type="double" />
      <field name="bestBidAmount" id="11" type="double" />
      <field name="bestAskPrice" id="12" type="double" />
      <field name="bestAskAmount" id="13" type="double" />
      <field name="currentFunding" id="14" type="double"
            presence="optional" />
      <field name="funding8h" id="15" type="double"
            presence="optional" />
      <field name="estimatedDeliveryPrice" id="16" type="double"
            presence="optional"/>
      <field name="deliveryPrice" id="17" type="double"
            presence="optional" />
```

```
      <field name="settlementPrice" id="18" type="double"
            presence="optional" />
</message>
```

Example message

```
Frame 106: 195 bytes on wire (1560 bits), 195 bytes captured (1560 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.4
User Datagram Protocol, Src Port: 37630, Dst Port: 6100
Deribit SBE
    Framing Header
        Packet Length: 145
        Channel Id: 4
        Channel Sequence: 4218
    Ticker
        Header
            Root Block Length: 133
            Type: ticker (1003)
            Schema Id: 1
            Version: 1
            Num Groups: 0
            Num Vars: 0
        Instrument Id: 2
        Instrument State: open (1)
        Timestamp: May  3, 2022 13:17:24.827000000 UTC
        Open Interest: 10
        Min Price: 2837,56
        Max Price: 2866,08
        Last Price: 2865,61
        Index Price: 2834,85
        Mark Price: 2850,44
        Best Bid Price: 2866,09
        Best Bid Amount: 20
        Best Ask Price: 2866,1
        Best Ask Amount: 10
        Current Funding: 0,004999
        Funding 8h: -0,003006
        Estimated Delivery Price: 2834,85
        Delivery Price: 0
        Settlement Price: 2837,27
0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   00 b5 9e 37 40 00 20 11 de 8b 7f 00 00 01 ef 6f   ...7@. ........o
0020   6f 04 92 fe 17 d4 00 a1 de 27 91 00 04 00 7a 10   o........'....z.
0030   00 00 85 00 eb 03 01 00 01 00 00 00 00 00 02 00   ................
0040   00 00 01 db 81 11 8a 80 01 00 00 00 00 00 00 00   ................
0050   00 24 40 85 eb 51 b8 1e 2b a6 40 5c 8f c2 f5 28   .$@..Q..+.@\...(
0060   64 a6 40 1f 85 eb 51 38 63 a6 40 33 33 33 33 b3   d.@...Q8c.@3333.
0070   25 a6 40 7b 14 ae 47 e1 44 a6 40 48 e1 7a 14 2e   %.@{..G.D.@H.z..
0080   64 a6 40 00 00 00 00 00 00 34 40 33 33 33 33 33   d.@......4@33333
```

```
0090   64 a6 40 00 00 00 00 00 00 24 40 1c 09 34 d8 d4   d.@......$@..4..
00a0   79 74 3f 6c 07 23 f6 09 a0 68 bf 33 33 33 33 b3   yt?l.#...h.3333.
00b0   25 a6 40 00 00 00 00 00 00 00 00 d7 a3 70 3d 8a   %.@..........p=.
00c0   2a a6 40                                          *.@
```

F - Snapshots

Next to the events above, regular (each 1 minute by default) snapshots of order books are also multicasted. Snapshots are also grouped by currency/product (like events) and can be assigned to separate channels (see Deribit Multicast Channels document).

Snapshots are a sequence of `instrument/instrument_v2/ticker/snapshot` messages that contain the static instrument data, the ticker information in the current state of the book, the order book levels, as well as the last change ID with the last change timestamp.

The book levels on both sides (asks/bids) are combined in a single list using a zigzag method (bid1, ask1, bid2, ask2 ….) .

Snapshots are generally complete (i.e. containing all book levels), however if they have more than 10000 levels, they will be limited to 10000.

To indicate whether the snapshot is complete or not, the message contains an `isComplete` flag.

If a complete snapshot does not fit in a packet, the level list is split (similarly to order book changes). The `isLast` flag indicates whether the message contains a part of the level list and the next message contains follow up (value = 0) or the message is the last in the sequence and the sending of the levels is done, or the message has all the levels sent in the snapshot (value = 1).

If multiple instrument/instrument_v2/ticker/snapshot message sequences (for different instruments on a currency/product pair) fit in a single packet, they will be combined. Book level snapshot messages may be split across packets.

Message specification

```
<message name="snapshot" id="1004">
      <field name="instrumentId" id="1" type="uint32" />
      <field name="timestampMs" id="2" type="uint64" />
      <field name="changeId" id="3" type="uint64" />
      <field name="isBookComplete" id="4" type="yesNo" />
      <field name="isLastInBook" id="5" type="yesNo" />
      <group name="levelsList" id="6" dimensionType="groupSizeEncoding">
            <field name="side" id="1" type="bookSide" />
            <field name="price" id="2" type="double" />
```

```
            <field name="amount" id="3" type="double" />
      </group>
</message>
```

Enum types used in the message

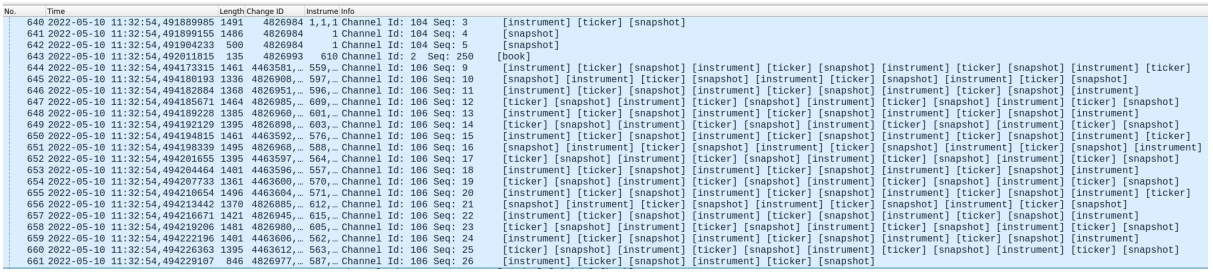
```
<enum name="yesNo" encodingType="uint8">
      <validValue name="no">0</validValue>
      <validValue name="yes">1</validValue>
</enum>

<enum name="bookSide" encodingType="uint8">
      <validValue name="ask">0</validValue>
      <validValue name="bid">1</validValue>
</enum>
```

Example message

```
Frame 13142: 347 bytes on wire (2776 bits), 347 bytes captured (2776 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.5
User Datagram Protocol, Src Port: 33646, Dst Port: 6101
Deribit SBE
    Framing Header
        Packet Length: 297
        Channel Id: 105
        Channel Sequence: 95
    Book Snapshot
        Header
            Root Block Length: 22
            Type: snapshot (1004)
            Schema Id: 1
            Version: 1
            Num Groups: 1
            Num Vars: 0
        Instrument Id: 486
        Timestamp: May  9, 2022 14:04:23.702000000 UTC
        Change ID: 4434022
        Is Book Complete: complete (1)
        Is Last Part: last (1)
        Levels
            Group Header
                Group Block Length: 17
                Num In Group: 15
                Group Num Groups: 0
                Group Num Vars: 0
            Level List
                bid - price : 32030,22 amount : 10
                bid - price : 32018,26 amount : 10
                bid - price : 32017,43 amount : 10
```

```
                bid - price : 32016,94 amount : 20
                bid - price : 32013,36 amount : 10
                bid - price : 32011,99 amount : 10
                bid - price : 31728,66 amount : 10
                bid - price : 31724,04 amount : 30
                bid - price : 31712,1 amount : 10
                bid - price : 31711,62 amount : 20
                bid - price : 31709,33 amount : 20
                bid - price : 31694,14 amount : 20
                bid - price : 31684,67 amount : 10
                bid - price : 31683,33 amount : 20
                bid - price : 31079,33 amount : 30
0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   01 4d c1 01 40 00 20 11 bb 28 7f 00 00 01 ef 6f   .M..@. ..(.....o
0020   6f 05 83 6e 17 d5 01 39 de c0 29 01 69 00 5f 00   o..n...9..).i._.
0030   00 00 16 00 ec 03 01 00 01 00 01 00 00 00 e6 01   ................
0040   00 00 16 ad 22 a9 80 01 00 00 66 a8 43 00 00 00   ....".....f.C...
0050   00 00 01 01 11 00 0f 00 00 00 00 00 01 48 e1 7a   .............H.z
0060   14 8e 47 df 40 00 00 00 00 00 00 24 40 01 3d 0a   ..G.@......$@.=.
0070   d7 a3 90 44 df 40 00 00 00 00 00 00 24 40 01 52   ...D.@......$@.R
0080   b8 1e 85 5b 44 df 40 00 00 00 00 00 00 24 40 01   ...[D.@......$@.
0090   8f c2 f5 28 3c 44 df 40 00 00 00 00 00 00 34 40   ...(<D.@......4@
00a0   01 a4 70 3d 0a 57 43 df 40 00 00 00 00 00 00 24   ..p=.WC.@......$
00b0   40 01 c3 f5 28 5c ff 42 df 40 00 00 00 00 00 00   @...(\.B.@......
00c0   24 40 01 d7 a3 70 3d 2a fc de 40 00 00 00 00 00   $@...p=*..@.....
00d0   00 24 40 01 f6 28 5c 8f 02 fb de 40 00 00 00 00   .$@..(\....@....
00e0   00 00 3e 40 01 66 66 66 66 06 f8 de 40 00 00 00   ..>@.ffff...@...
00f0   00 00 00 24 40 01 e1 7a 14 ae e7 f7 de 40 00 00   ...$@..z.....@..
0100   00 00 00 00 34 40 01 ec 51 b8 1e 55 f7 de 40 00   ....4@..Q..U..@.
0110   00 00 00 00 00 34 40 01 5c 8f c2 f5 88 f3 de 40   .....4@.\.......@
0120   00 00 00 00 00 00 34 40 01 14 ae 47 e1 2a f1 de   ......4@...G.*..
0130   40 00 00 00 00 00 00 00 24 40 01 ec 51 b8 1e d5 f0   @.......$@..Q....
0140   de 40 00 00 00 00 00 00 34 40 01 ec 51 b8 1e d5   .@......4@..Q...
0150   59 de 40 00 00 00 00 00 00 3e 40                  Y.@......>@
```

Example sequence of snapshots



No.	Time	Length	Change ID	Instrume	Info
640	2022-05-10 11:32:54,491889985	1491	4826984	1,1,1	Channel Id: 104 Seq: 3 [instrument] [ticker] [snapshot]
641	2022-05-10 11:32:54,491899155	1486	4826984	1	Channel Id: 104 Seq: 4 [snapshot]
642	2022-05-10 11:32:54,491904233	500	4826984	1	Channel Id: 104 Seq: 5 [snapshot]
643	2022-05-10 11:32:54,492011815	135	4826993	610	Channel Id: 2 Seq: 250 [book]
644	2022-05-10 11:32:54,494173315	1461	4463581,…	559,…	Channel Id: 106 Seq: 9 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker]
645	2022-05-10 11:32:54,494180193	1336	4826908,…	597,…	Channel Id: 106 Seq: 10 [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot]
646	2022-05-10 11:32:54,494182884	1368	4826951,…	596,…	Channel Id: 106 Seq: 11 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument]
647	2022-05-10 11:32:54,494185671	1464	4826985,…	609,…	Channel Id: 106 Seq: 12 [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot]
648	2022-05-10 11:32:54,494189228	1385	4826960,…	601,…	Channel Id: 106 Seq: 13 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument]
649	2022-05-10 11:32:54,494192129	1395	4826898,…	603,…	Channel Id: 106 Seq: 14 [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot]
650	2022-05-10 11:32:54,494194815	1461	4463592,…	576,…	Channel Id: 106 Seq: 15 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker]
651	2022-05-10 11:32:54,494198339	1495	4826968,…	588,…	Channel Id: 106 Seq: 16 [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument]
652	2022-05-10 11:32:54,494201655	1395	4463597,…	564,…	Channel Id: 106 Seq: 17 [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot]
653	2022-05-10 11:32:54,494204464	1401	4463596,…	557,…	Channel Id: 106 Seq: 18 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument]
654	2022-05-10 11:32:54,494207733	1361	4463600,…	570,…	Channel Id: 106 Seq: 19 [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot]
655	2022-05-10 11:32:54,494210654	1496	4463604,…	571,…	Channel Id: 106 Seq: 20 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker]
656	2022-05-10 11:32:54,494213442	1370	4826885,…	612,…	Channel Id: 106 Seq: 21 [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot]
657	2022-05-10 11:32:54,494216671	1421	4826945,…	615,…	Channel Id: 106 Seq: 22 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument]
658	2022-05-10 11:32:54,494219206	1481	4826980,…	605,…	Channel Id: 106 Seq: 23 [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot]
659	2022-05-10 11:32:54,494222196	1401	4463606,…	562,…	Channel Id: 106 Seq: 24 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument]
660	2022-05-10 11:32:54,494226363	1395	4463612,…	563,…	Channel Id: 106 Seq: 25 [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot]
661	2022-05-10 11:32:54,494229107	846	4826977,…	587,…	Channel Id: 106 Seq: 26 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot]

F.1 Snapshot Start

The purpose of the `snapshotStart` event is to indicate to the client that it can start preparing for the arrival of an upcoming snapshot, by buffering the events before receiving/processing the snapshot. Shortly (currently 200 ms) before each snapshot is sent, a message with type `snapshotStart` is sent, to indicate that a possible snapshot sequence will be on the way.

The field `snapshotDelay` indicates the delay in milliseconds.

Another message with type `snapshotEnd` is sent as soon as the snapshot sequence for the channel is completed

In the case where there is no snapshot available (no active books in the channel), only a pair of `snapshotStart` and `snapshotEnd` will be sent.

Message specification

```
<message name="snapshotStart" id="1005">
      <field name="snapshotDelay" id="1" type="uint32" />
</message>
```

Example message

```
Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.10
User Datagram Protocol, Src Port: 48141, Dst Port: 6101
Deribit SBE
      Framing Header
      Packet Length: 16
      Channel Id: 110
      Channel Sequence: 2
      Book Snapshot Start
      Header
            Root Block Length: 4
            Type: snapshot_start (1005)
            Schema Id: 1
            Version: 1
            Num Groups: 0
            Num Vars: 0
      Snapshot Delay: 200

0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   00 b5 54 33 40 00 20 11 28 8a 7f 00 00 01 ef 6f   ..T3@. .(......o
0020   6f 0a bc 0d 17 d5 00 20 91 9c 10 00 6e 00 02 00   o...... ....n...
0030   00 00 04 00 ed 03 01 00 01 00 00 00 00 00 c8 00   ................
0040   00 00                                             ..
```

F.2 Snapshot End

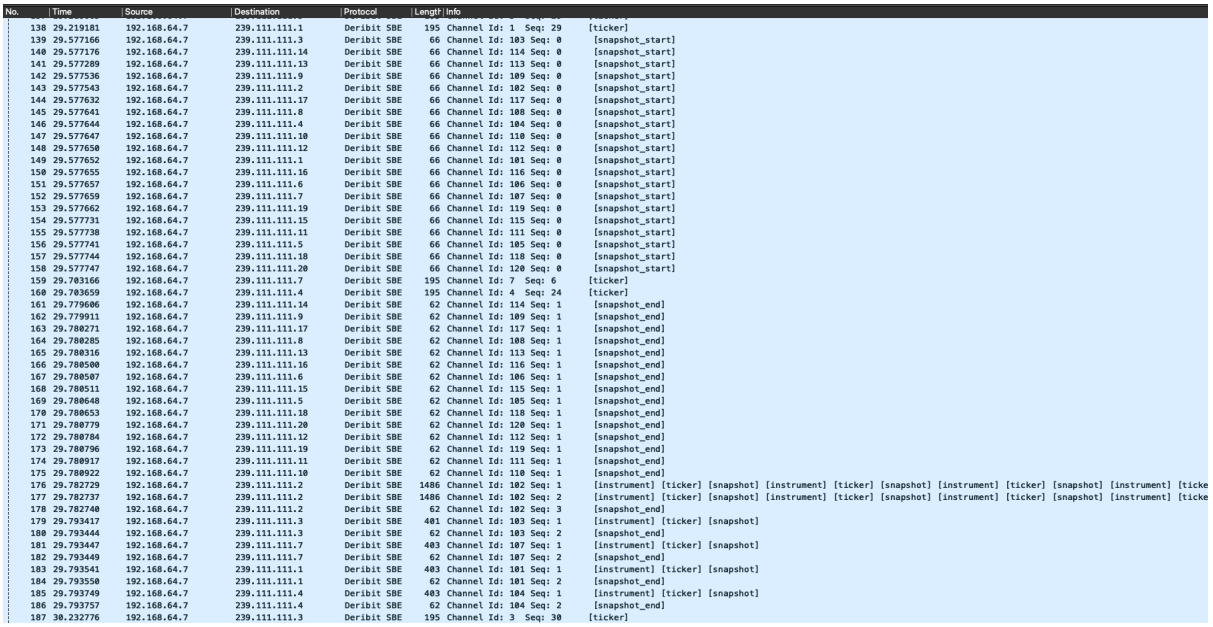
Message specification

```
<message name="snapshotEnd" id="1006">
</message>
```

Example message

```
Frame 1: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.10
User Datagram Protocol, Src Port: 48141, Dst Port: 6101
Deribit SBE
      Framing Header
      Packet Length: 12
      Channel Id: 110
      Channel Sequence: 3
      Book Snapshot End
      Header
            Root Block Length: 0
            Type: snapshot_end (1006)
            Schema Id: 1
            Version: 1
            Num Groups: 0
            Num Vars: 0

0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   00 b5 54 33 40 00 20 11 28 8a 7f 00 00 01 ef 6f   ..T3@. .(......o
0020   6f 0a bc 0d 17 d5 00 1c 5f a5 0c 00 6e 00 03 00   o......._...n...
0030   00 00 00 00 ee 03 01 00 01 00 00 00 00 00          ..............
```

Example sequence of snapshots



No.	Time	Source	Destination	Protocol	Length	Info
138	29.219181	192.168.64.7	239.111.111.1	Deribit SBE	195	Channel Id: 1 Seq: 29 [ticker]
139	29.577166	192.168.64.7	239.111.111.3	Deribit SBE	66	Channel Id: 103 Seq: 0 [snapshot_start]
140	29.577176	192.168.64.7	239.111.111.14	Deribit SBE	66	Channel Id: 114 Seq: 0 [snapshot_start]
141	29.577289	192.168.64.7	239.111.111.13	Deribit SBE	66	Channel Id: 113 Seq: 0 [snapshot_start]
142	29.577536	192.168.64.7	239.111.111.9	Deribit SBE	66	Channel Id: 109 Seq: 0 [snapshot_start]
143	29.577543	192.168.64.7	239.111.111.2	Deribit SBE	66	Channel Id: 102 Seq: 0 [snapshot_start]
144	29.577632	192.168.64.7	239.111.111.17	Deribit SBE	66	Channel Id: 117 Seq: 0 [snapshot_start]
145	29.577641	192.168.64.7	239.111.111.8	Deribit SBE	66	Channel Id: 108 Seq: 0 [snapshot_start]
146	29.577644	192.168.64.7	239.111.111.4	Deribit SBE	66	Channel Id: 104 Seq: 0 [snapshot_start]
147	29.577647	192.168.64.7	239.111.111.10	Deribit SBE	66	Channel Id: 110 Seq: 0 [snapshot_start]
148	29.577650	192.168.64.7	239.111.111.12	Deribit SBE	66	Channel Id: 112 Seq: 0 [snapshot_start]
149	29.577652	192.168.64.7	239.111.111.1	Deribit SBE	66	Channel Id: 101 Seq: 0 [snapshot_start]
150	29.577655	192.168.64.7	239.111.111.16	Deribit SBE	66	Channel Id: 116 Seq: 0 [snapshot_start]
151	29.577657	192.168.64.7	239.111.111.6	Deribit SBE	66	Channel Id: 106 Seq: 0 [snapshot_start]
152	29.577659	192.168.64.7	239.111.111.7	Deribit SBE	66	Channel Id: 107 Seq: 0 [snapshot_start]
153	29.577662	192.168.64.7	239.111.111.19	Deribit SBE	66	Channel Id: 119 Seq: 0 [snapshot_start]
154	29.577731	192.168.64.7	239.111.111.15	Deribit SBE	66	Channel Id: 115 Seq: 0 [snapshot_start]
155	29.577738	192.168.64.7	239.111.111.11	Deribit SBE	66	Channel Id: 111 Seq: 0 [snapshot_start]
156	29.577741	192.168.64.7	239.111.111.5	Deribit SBE	66	Channel Id: 105 Seq: 0 [snapshot_start]
157	29.577744	192.168.64.7	239.111.111.18	Deribit SBE	66	Channel Id: 118 Seq: 0 [snapshot_start]
158	29.577747	192.168.64.7	239.111.111.20	Deribit SBE	66	Channel Id: 120 Seq: 0 [snapshot_start]
159	29.703166	192.168.64.7	239.111.111.7	Deribit SBE	195	Channel Id: 7 Seq: 6 [ticker]
160	29.703659	192.168.64.7	239.111.111.4	Deribit SBE	195	Channel Id: 4 Seq: 24 [ticker]
161	29.779606	192.168.64.7	239.111.111.14	Deribit SBE	62	Channel Id: 114 Seq: 1 [snapshot_end]
162	29.779911	192.168.64.7	239.111.111.9	Deribit SBE	62	Channel Id: 109 Seq: 1 [snapshot_end]
163	29.780271	192.168.64.7	239.111.111.17	Deribit SBE	62	Channel Id: 117 Seq: 1 [snapshot_end]
164	29.780285	192.168.64.7	239.111.111.8	Deribit SBE	62	Channel Id: 108 Seq: 1 [snapshot_end]
165	29.780316	192.168.64.7	239.111.111.13	Deribit SBE	62	Channel Id: 113 Seq: 1 [snapshot_end]
166	29.780500	192.168.64.7	239.111.111.16	Deribit SBE	62	Channel Id: 116 Seq: 1 [snapshot_end]
167	29.780507	192.168.64.7	239.111.111.6	Deribit SBE	62	Channel Id: 106 Seq: 1 [snapshot_end]
168	29.780511	192.168.64.7	239.111.111.15	Deribit SBE	62	Channel Id: 115 Seq: 1 [snapshot_end]
169	29.780648	192.168.64.7	239.111.111.5	Deribit SBE	62	Channel Id: 105 Seq: 1 [snapshot_end]
170	29.780653	192.168.64.7	239.111.111.18	Deribit SBE	62	Channel Id: 118 Seq: 1 [snapshot_end]
171	29.780779	192.168.64.7	239.111.111.20	Deribit SBE	62	Channel Id: 120 Seq: 1 [snapshot_end]
172	29.780784	192.168.64.7	239.111.111.12	Deribit SBE	62	Channel Id: 112 Seq: 1 [snapshot_end]
173	29.780796	192.168.64.7	239.111.111.19	Deribit SBE	62	Channel Id: 119 Seq: 1 [snapshot_end]
174	29.780917	192.168.64.7	239.111.111.11	Deribit SBE	62	Channel Id: 111 Seq: 1 [snapshot_end]
175	29.780922	192.168.64.7	239.111.111.10	Deribit SBE	62	Channel Id: 110 Seq: 1 [snapshot_end]
176	29.782729	192.168.64.7	239.111.111.2	Deribit SBE	1486	Channel Id: 102 Seq: 1 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticke
177	29.782737	192.168.64.7	239.111.111.2	Deribit SBE	1486	Channel Id: 102 Seq: 2 [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticker] [snapshot] [instrument] [ticke
178	29.782740	192.168.64.7	239.111.111.2	Deribit SBE	62	Channel Id: 102 Seq: 3 [snapshot_end]
179	29.793417	192.168.64.7	239.111.111.3	Deribit SBE	401	Channel Id: 103 Seq: 1 [instrument] [ticker] [snapshot]
180	29.793444	192.168.64.7	239.111.111.3	Deribit SBE	62	Channel Id: 103 Seq: 2 [snapshot_end]
181	29.793447	192.168.64.7	239.111.111.7	Deribit SBE	403	Channel Id: 107 Seq: 1 [instrument] [ticker] [snapshot]
182	29.793449	192.168.64.7	239.111.111.7	Deribit SBE	62	Channel Id: 107 Seq: 2 [snapshot_end]
183	29.793541	192.168.64.7	239.111.111.1	Deribit SBE	403	Channel Id: 101 Seq: 1 [instrument] [ticker] [snapshot]
184	29.793550	192.168.64.7	239.111.111.1	Deribit SBE	62	Channel Id: 101 Seq: 2 [snapshot_end]
185	29.793749	192.168.64.7	239.111.111.4	Deribit SBE	403	Channel Id: 104 Seq: 1 [instrument] [ticker] [snapshot]
186	29.793757	192.168.64.7	239.111.111.4	Deribit SBE	62	Channel Id: 104 Seq: 2 [snapshot_end]
187	30.232776	192.168.64.7	239.111.111.3	Deribit SBE	195	Channel Id: 3 Seq: 30 [ticker]

G - Combo legs

The combo legs message is sent after the instrument message in the combo event and multicast channels on two occasions: 1) in the event channel when a combo is created 2) in the snapshot channel with each snapshot.

The event contains an instrument ID, and a list of combo legs. Each combo leg in the list contains an instrument ID for the combo leg, as well as the combo leg size (int32).

Message specification

```
<message name="comboLegs" id="1007">
	<field name="instrumentId" id="1" type="uint32" />
	<group name="legsList" id="2" dimensionType="groupSizeEncoding">
		<field name="legInstrumentId" id="1" type="uint32" />
		<field name="legSize" id="2" type="int32" />
	</group>
</message>
```

Example message

```
Frame 1865: 264 bytes on wire (2112 bits), 264 bytes captured (2112 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.14
User Datagram Protocol, Src Port: 54658, Dst Port: 6100
Deribit SBE
	Framing Header
```

```
Packet Length: 214
Channel Id: 14
Channel Sequence: 0
Instrument State
Combo Legs
Header
      Root Block Length: 4
      Type: combo_legs (1007)
      Schema Id: 1
      Version: 2
      Num Groups: 1
      Num Vars: 0
Instrument Id: 32
Legs
      Group Header
      Group Block Length: 8
      Num In Group: 2
      Group Num Groups: 0
      Group Num Vars: 0
      Leg List
      leg instrument id: 1 leg size : -1
      leg instrument id: 2 leg size : 1
```

```
0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   00 fa ed e2 40 00 20 11 8e 91 7f 00 00 01 ef 6f   ....@. ........o
0020   6f 0e d5 82 17 d4 00 e6 de 76 d6 00 0e 00 00 00   o........v......
0030   00 00 8c 00 e8 03 01 00 02 00 00 00 01 00 20 00   .............. .
0040   00 00 00 02 00 00 01 04 01 00 42 54 43 00 00 00   ..........BTC...
0050   00 00 55 53 44 00 00 00 00 00 55 53 44 00 00 00   ..USD.....USD...
0060   00 00 42 54 43 00 00 00 00 00 55 53 44 00 00 00   ..BTC.....USD...
0070   00 00 10 bd 10 3d 84 01 00 00 00 64 a7 41 84 01   .....=.....d.A..
0080   00 00 ff ff ff ff ff ff ff ff 00 00 00 00 00 00   ................
0090   24 40 00 00 00 00 00 00 24 40 7b 14 ae 47 e1 7a   $@......$@{..G.z
00a0   84 3f 00 00 00 00 00 00 00 00 00 00 00 00 00 00   .?..............
00b0   00 00 ff ff ff ff ff ff ff ff ff ff ff ff ff ff   ................
00c0   ff ff ff ff ff ff ff ff ff ff 15 42 54 43 2d 46   ...........BTC-F
00d0   53 2d 31 31 4e 4f 56 32 32 5f 34 4e 4f 56 32 32   S-11NOV22_4NOV22
00e0   04 00 ef 03 01 00 02 00 01 00 00 00 20 00 00 00   ............ ...
00f0   08 00 02 00 00 00 00 00 01 00 00 00 ff ff ff ff   ................
0100   02 00 00 00 01 00 00 00                           ........
```

H - Price Index

The price index event contains an index name of 16 bytes in the form of currency pairs (e.g. `eth_usdc`), followed by price and timestamp. The price index messages are not associated with any currency/product group, therefore sent on the generic multicast channel (0).

Message specification

```
<message name="priceIndex" id="1008">
      <field name="indexName" id="1" type="string16" />
      <field name="price" id="2" type="double" />
      <field name="timestampMs" id="3" type="uint64" />
</message>
```

Other types used in the message:

```
<type name="string16" primitiveType="char" length="16" />
```

Example message

```
Frame 1228: 94 bytes on wire (752 bits), 94 bytes captured (752 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.0
User Datagram Protocol, Src Port: 40820, Dst Port: 6100
Deribit SBE
      Framing Header
      Packet Length: 44
      Channel Id: 0
      Channel Sequence: 306
      Price Index
      Header
            Root Block Length: 32
            Type: price_index (1008)
            Schema Id: 1
            Version: 2
            Num Groups: 0
            Num Vars: 0
      Currency Pair: eth_usdc
      Price: 1271,1231
      Timestamp: Nov 11, 2022 10:10:45.315000000 UTC

0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   00 50 1e 49 40 00 20 11 5e e3 7f 00 00 01 ef 6f   .P.I@. .^......o
0020   6f 00 9f 74 17 d4 00 3c dd be 2c 00 00 00 32 01   o..t...<..,...2.
0030   00 00 20 00 f0 03 01 00 02 00 00 00 00 00 65 74   .. ..........et
0040   68 5f 75 73 64 63 00 00 00 00 00 00 00 00 8d 28   h_usdc.........(
0050   ed 0d 7e dc 93 40 c3 9d 2b 66 84 01 00 00          ..~..@..+f....
```

I - RFQ

The RFQ (Request For Quotation) event contains the state (active or inactive), side (buy, sell, or no_direction), followed by amount and timestamp. This message will be sent on the generic channel id: 0. Please refer to the channel documentation for its multicast address.

Message specification

```
<message name="rfq" id="1009">
      <field name="instrumentId" id="1" type="uint32" />
      <field name="state" id="2" type="yesNo" />
      <field name="side" id="3" type="rfqDirection" />
      <field name="amount" id="4" type="double" />
      <field name="timestampMs" id="5" type="uint64" />
</message>
```

Enum Types used in the message:

```
<enum name="yesNo" encodingType="uint8">
      <validValue name="no">0</validValue>
      <validValue name="yes">1</validValue>
</enum>

<enum name="rfqDirection" encodingType="uint8">
      <validValue name="buy">0</validValue>
      <validValue name="sell">1</validValue>
      <validValue name="no_direction">2</validValue>
</enum>
```

Example message

```
Frame 1229: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.1
User Datagram Protocol, Src Port: 37608, Dst Port: 6100
Deribit SBE
      Framing Header
      Packet Length: 34
      Channel Id: 1
      Channel Sequence: 59
      RFQ
      Header
            Root Block Length: 22
            Type: rfq (1009)
            Schema Id: 1
            Version: 2
            Num Groups: 0
            Num Vars: 0
      Instrument Id: 12
      RFQ State: active (1)
      RFQ Side: buy (0)
      Amount: 1
```

```
      Timestamp: Nov 11, 2022 10:10:46.250000000 UTC

0000   00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00   ..............E.
0010   00 46 29 55 40 00 20 11 53 e0 7f 00 00 01 ef 6f   .F)U@. .S......o
0020   6f 01 92 e8 17 d4 00 32 dd b5 22 00 01 00 3b 00   o......2.."...;.
0030   00 00 16 00 f1 03 01 00 02 00 00 00 00 00 0c 00   ................
0040   00 00 01 00 00 00 00 00 00 00 f0 3f 6a a1 2b 66   ...........?j.+f
0050   84 01 00 00                                       ....
```

J - Spots

Spot instruments generate the same messages as other instruments. Book events and ticker events are generated for changes in order book data. Spot snapshots include ticker and book data for all spot instruments.

Example message

```
Frame 19: 117 bytes on wire (936 bits), 117 bytes captured (936 bits) on
interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.21
User Datagram Protocol, Src Port: 52994, Dst Port: 6100
Deribit SBE
      Framing Header
      Packet Length: 67
      Channel Id: 21
      Channel Sequence: 184
      Book Change
      Header
            Root Block Length: 29
            Type: book (1001)
            Schema Id: 1
            Version: 2
            Num Groups: 1
            Num Vars: 0
      Instrument Id: 29
      Timestamp: Mar  2, 2023 13:43:14.929000000 UTC
      Previous Change ID: 80
      Change ID: 86
      Is Last Part: last (1)
      Changes
            Group Header
            Group Block Length: 18
            Num In Group: 1
            Group Num Groups: 0
            Group Num Vars: 0
            Change List
            new ask - price : 23308,9845 amount : 0,0002
```

```
0000   43 00 15 00 b8 00 00 00 1d 00 e9 03 01 00 02 00
0010   01 00 00 00 1d 00 00 00 f1 0c 90 a2 86 01 00 00
0020   50 00 00 00 00 00 00 00 56 00 00 00 00 00 00 00
0030   01 12 00 01 00 00 00 00 00 00 00 ba 49 0c 02 3f
0040   c3 d6 40 2d 43 1c eb e2 36 2a 3f
```

Example message

```
Frame 723: 1494 bytes on wire (11952 bits), 1494 bytes captured (11952 bits)
on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst:
00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 239.111.111.21
User Datagram Protocol, Src Port: 48500, Dst Port: 6101
Deribit SBE
      Framing Header
      Packet Length: 1444
      Channel Id: 121
      Channel Sequence: 4
      Instrument State
      Header
            Root Block Length: 140
            Type: instrument (1000)
            Schema Id: 1
            Version: 2
            Num Groups: 0
            Num Vars: 1
      Instrument Id: 29
      Instrument State: open (1)
      Instrument Kind: spot (4)
      Instrument Type: not applicable (0)
      Option Type: not applicable (0)
      RFQ: 0
      Settlement Period: Unknown (255)
      Settlement Period Count: 0
      Base Currency: BTC
      Quote Currency: USDC
      Counter Currency: USDC
      Settlement Currency:
      Size Currency: BTC
      Creation Timestamp: Mar  2, 2023 13:42:24.000000000 UTC
      Expiration Timestamp: Jan  1, 3000 08:00:00.000000000 UTC
      Strike Price: -nan
      Contract Size: 0,0001
      Minimum Trade Amount: 0,0001
      Tick Size: 0,0001
      Maker Commission: 0
      Taker Commission: 0,0001
      Block Trade Commission: -nan
      Max Liquidation Commission: -nan
      Max Leverage: -nan
      Instrument Name: BTC_USDC
      Ticker
```

```
Header
      Root Block Length: 133
      Type: ticker (1003)
      Schema Id: 1
      Version: 2
      Num Groups: 0
      Num Vars: 0
Instrument Id: 29
Instrument State: open (1)
Timestamp: Mar  2, 2023 13:43:26.970000000 UTC
Open Interest: -nan
Min Sell Price: 23226,7493
Max Buy Price: 23460,1839
Last Price: 23274,6191
Index Price: 23372,9729
Mark Price: 23296,2361
Best Bid Price: 23274,6191
Best Bid Amount: 0,0005
Best Ask Price: 23317,8531
Best Ask Amount: 0,0001
Current Funding: -nan
Funding 8h: -nan
Estimated Delivery Price: -nan
Delivery Price: -nan
Settlement Price: -nan
Book Snapshot
Header
      Root Block Length: 22
      Type: snapshot (1004)
      Schema Id: 1
      Version: 2
      Num Groups: 1
      Num Vars: 0
Instrument Id: 29
Timestamp: Mar  2, 2023 13:43:26.970000000 UTC
Change ID: 163
Is Book Complete: complete (1)
Is Last Part: last (1)
Levels
      Group Header
      Group Block Length: 17
      Num In Group: 27
      Group Num Groups: 0
      Group Num Vars: 0
      Level List
      ask - price : 23317,8531 amount : 0,0001
      bid - price : 23274,6191 amount : 0,0005
      ask - price : 23325,4224 amount : 0,0001
      bid - price : 23267,3293 amount : 0,0003
      ask - price : 23327,1627 amount : 0,0004
      ask - price : 23332,8221 amount : 0,0002
      ask - price : 23343,9179 amount : 0,0006
      ask - price : 23355,7196 amount : 0,0007
```

```
      ask - price : 23355,7235 amount : 0,0004
      ask - price : 23360,0454 amount : 0,0004
      ask - price : 23363,7177 amount : 0,0006
      ask - price : 23370,4559 amount : 0,0007
      ask - price : 23371,5373 amount : 0,0001
      ask - price : 23374,1273 amount : 0,0002
      ask - price : 23383,1583 amount : 0,0002
      ask - price : 23386,2295 amount : 0,0001
      ask - price : 23392,8084 amount : 0,0004
      ask - price : 23392,822 amount : 0,0002
      ask - price : 23409,2294 amount : 0,0004
      ask - price : 23410,6344 amount : 0,0001
      ask - price : 23417,449 amount : 0,0003
      ask - price : 23419,4629 amount : 0,0008
      ask - price : 23436,5183 amount : 0,0001
      ask - price : 23436,5464 amount : 0,0008
      ask - price : 23437,9561 amount : 0,0001
      ask - price : 23438,0801 amount : 0,0002
      ask - price : 23438,229 amount : 0,0003
Instrument State
Header
      Root Block Length: 140
      Type: instrument (1000)
      Schema Id: 1
      Version: 2
      Num Groups: 0
      Num Vars: 1
Instrument Id: 30
Instrument State: open (1)
Instrument Kind: spot (4)
Instrument Type: not applicable (0)
Option Type: not applicable (0)
RFQ: 0
Settlement Period: Unknown (255)
Settlement Period Count: 0
Base Currency: ETH
Quote Currency: USDC
Counter Currency: USDC
Settlement Currency:
Size Currency: ETH
Creation Timestamp: Mar  2, 2023 13:42:24.000000000 UTC
Expiration Timestamp: Jan  1, 3000 08:00:00.000000000 UTC
Strike Price: -nan
Contract Size: 0,0001
Minimum Trade Amount: 0,0001
Tick Size: 0,0001
Maker Commission: 0
Taker Commission: 0,0001
Block Trade Commission: -nan
Max Liquidation Commission: -nan
Max Leverage: -nan
Instrument Name: ETH_USDC
Ticker
```

```
Header
      Root Block Length: 133
      Type: ticker (1003)
      Schema Id: 1
      Version: 2
      Num Groups: 0
      Num Vars: 0
Instrument Id: 30
Instrument State: open (1)
Timestamp: Mar  2, 2023 13:43:26.973000000 UTC
Open Interest: -nan
Min Sell Price: 1624,8012
Max Buy Price: 1641,1309
Last Price: 1628,9218
Index Price: 1634,1202
Mark Price: 1629,0167
Best Bid Price: 0
Best Bid Amount: 0
Best Ask Price: 1624,9995
Best Ask Amount: 0,0025
Current Funding: -nan
Funding 8h: -nan
Estimated Delivery Price: -nan
Delivery Price: -nan
Settlement Price: -nan
Book Snapshot
Header
      Root Block Length: 22
      Type: snapshot (1004)
      Schema Id: 1
      Version: 2
      Num Groups: 1
      Num Vars: 0
Instrument Id: 30
Timestamp: Mar  2, 2023 13:43:26.973000000 UTC
Change ID: 162
Is Book Complete: complete (1)
Is Last Part: more to come (0)
Levels
      Group Header
      Group Block Length: 17
      Num In Group: 17
      Group Num Groups: 0
      Group Num Vars: 0
      Level List
      ask - price : 1624,9995 amount : 0,0025
      ask - price : 1630,9028 amount : 0,0104
      ask - price : 1630,9219 amount : 0,0098
      ask - price : 1631,4677 amount : 0,0012
      ask - price : 1632,6882 amount : 0,0043
      ask - price : 1634,0983 amount : 0,0052
      ask - price : 1634,7069 amount : 0,0061
      ask - price : 1635,2764 amount : 0,0098
```

```
            ask - price : 1635,3288 amount : 0,0098
            ask - price : 1635,583 amount : 0,0006
            ask - price : 1635,9182 amount : 0,0012
            ask - price : 1637,3179 amount : 0,0055
            ask - price : 1637,4283 amount : 0,0043
            ask - price : 1637,641 amount : 0,0061
            ask - price : 1637,8371 amount : 0,0061
            ask - price : 1638,5149 amount : 0,0117
            ask - price : 1639,1852 amount : 0,0043

0000   a4 05 79 00 04 00 00 00 8c 00 e8 03 01 00 02 00
0010   00 00 01 00 1d 00 00 00 01 04 00 00 00 ff 00 00
0020   42 54 43 00 00 00 00 00 55 53 44 43 00 00 00 00
0030   55 53 44 43 00 00 00 00 00 00 00 00 00 00 00 00
0040   42 54 43 00 00 00 00 00 00 46 8f a2 86 01 00 00
0050   00 54 04 dc 8f 1d 00 00 ff ff ff ff ff ff ff ff
0060   2d 43 1c eb e2 36 1a 3f 2d 43 1c eb e2 36 1a 3f
0070   2d 43 1c eb e2 36 1a 3f 00 00 00 00 00 00 00 00
0080   2d 43 1c eb e2 36 1a 3f ff ff ff ff ff ff ff ff
0090   ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
00a0   08 42 54 43 5f 55 53 44 43 85 00 eb 03 01 00 02
00b0   00 00 00 00 00 1d 00 00 00 01 fa 3b 90 a2 86 01
00c0   00 00 ff ff ff ff ff ff ff ff b9 fc 87 f4 af ae
00d0   d6 40 6f 81 04 c5 0b e9 d6 40 3d 9b 55 9f a7 ba
00e0   d6 40 92 5c fe 43 3e d3 d6 40 a5 2c 43 1c 0f c0
00f0   d6 40 3d 9b 55 9f a7 ba d6 40 fc a9 f1 d2 4d 62
0100   40 3f 0e be 30 99 76 c5 d6 40 2d 43 1c eb e2 36
0110   1a 3f ff ff ff ff ff ff ff ff ff ff ff ff ff ff
0120   ff ff 92 5c fe 43 3e d3 d6 40 ff ff ff ff ff ff
0130   ff ff ff ff ff ff ff ff ff ff 16 00 ec 03 01 00
0140   02 00 01 00 00 00 1d 00 00 00 fa 3b 90 a2 86 01
0150   00 00 a3 00 00 00 00 00 00 00 01 01 11 00 1b 00
0160   00 00 00 00 00 0e be 30 99 76 c5 d6 40 2d 43 1c
0170   eb e2 36 1a 3f 01 3d 9b 55 9f a7 ba d6 40 fc a9
0180   f1 d2 4d 62 40 3f 00 75 02 9a 08 5b c7 d6 40 2d
0190   43 1c eb e2 36 1a 3f 01 a5 4e 40 13 d5 b8 d6 40
01a0   61 32 55 30 2a a9 33 3f 00 c4 42 ad 69 ca c7 d6
01b0   40 2d 43 1c eb e2 36 3a 3f 00 83 51 49 9d 34 c9
01c0   d6 40 2d 43 1c eb e2 36 2a 3f 00 40 a4 df be fa
01d0   cb d6 40 61 32 55 30 2a a9 43 3f 00 8d 28 ed 0d
01e0   ee ce d6 40 c7 ba b8 8d 06 f0 46 3f 00 aa f1 d2
01f0   4d ee ce d6 40 2d 43 1c eb e2 36 3a 3f 00 cf 66
0200   d5 e7 02 d0 d6 40 2d 43 1c eb e2 36 3a 3f 00 16
0210   fb cb ee ed d0 d6 40 61 32 55 30 2a a9 43 3f 00
0220   90 31 77 2d 9d d2 d6 40 c7 ba b8 8d 06 f0 46 3f
0230   00 09 8a 1f 63 e2 d2 d6 40 2d 43 1c eb e2 36 1a
0240   3f 00 32 e6 ae 25 88 d3 d6 40 2d 43 1c eb e2 36
0250   2a 3f 00 bd 52 96 21 ca d5 d6 40 2d 43 1c eb e2
0260   36 2a 3f 00 9c c4 20 b0 8e d6 d6 40 2d 43 1c eb
0270   e2 36 1a 3f 00 86 5a d3 bc 33 d8 d6 40 2d 43 1c
0280   eb e2 36 3a 3f 00 54 e3 a5 9b 34 d8 d6 40 2d 43
0290   1c eb e2 36 2a 3f 00 6d 56 7d ae 4e dc d6 40 2d
02a0   43 1c eb e2 36 3a 3f 00 25 75 02 9a a8 dc d6 40
```

```
02b0   2d 43 1c eb e2 36 1a 3f 00 fa 7e 6a bc 5c de d6
02c0   40 61 32 55 30 2a a9 33 3f 00 54 52 27 a0 dd de
02d0   d6 40 2d 43 1c eb e2 36 4a 3f 00 61 c3 d3 2b 21
02e0   e3 d6 40 2d 43 1c eb e2 36 1a 3f 00 a2 b4 37 f8
02f0   22 e3 d6 40 2d 43 1c eb e2 36 4a 3f 00 ed 0d be
0300   30 7d e3 d6 40 2d 43 1c eb e2 36 1a 3f 00 1a c0
0310   5b 20 85 e3 d6 40 2d 43 1c eb e2 36 2a 3f 00 b2
0320   9d ef a7 8e e3 d6 40 61 32 55 30 2a a9 33 3f 8c
0330   00 e8 03 01 00 02 00 00 00 01 00 1e 00 00 00 01
0340   04 00 00 00 ff 00 00 45 54 48 00 00 00 00 00 55
0350   53 44 43 00 00 00 00 55 53 44 43 00 00 00 00 00
0360   00 00 00 00 00 00 00 45 54 48 00 00 00 00 00 00
0370   46 8f a2 86 01 00 00 00 54 04 dc 8f 1d 00 00 ff
0380   ff ff ff ff ff ff ff 2d 43 1c eb e2 36 1a 3f 2d
0390   43 1c eb e2 36 1a 3f 2d 43 1c eb e2 36 1a 3f 00
03a0   00 00 00 00 00 00 00 2d 43 1c eb e2 36 1a 3f ff
03b0   ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
03c0   ff ff ff ff ff ff ff 08 45 54 48 5f 55 53 44 43
03d0   85 00 eb 03 01 00 02 00 00 00 00 00 1e 00 00 00
03e0   01 fd 3b 90 a2 86 01 00 00 ff ff ff ff ff ff ff
03f0   ff 39 d6 c5 6d 34 63 99 40 30 4c a6 0a 86 a4 99
0400   40 d0 d5 56 ec af 73 99 40 eb 73 b5 15 7b 88 99
0410   40 5f 07 ce 19 11 74 99 40 00 00 00 00 00 00 00
0420   00 00 00 00 00 00 00 00 00 68 91 ed 7c ff 63 99
0430   40 7b 14 ae 47 e1 7a 64 3f ff ff ff ff ff ff ff
0440   ff ff ff ff ff ff ff ff ff eb 73 b5 15 7b 88 99
0450   40 ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
0460   ff 16 00 ec 03 01 00 02 00 01 00 00 00 1e 00 00
0470   00 fd 3b 90 a2 86 01 00 00 a2 00 00 00 00 00 00
0480   00 01 00 11 00 11 00 00 00 00 00 00 68 91 ed 7c
0490   ff 63 99 40 7b 14 ae 47 e1 7a 64 3f 00 51 6b 9a
04a0   77 9c 7b 99 40 94 f6 06 5f 98 4c 85 3f 00 bb b8
04b0   8d 06 b0 7b 99 40 6e a3 01 bc 05 12 84 3f 00 5b
04c0   b1 bf ec de 7d 99 40 61 32 55 30 2a a9 53 3f 00
04d0   6e 34 80 b7 c0 82 99 40 22 fd f6 75 e0 9c 71 3f
04e0   00 ca 54 c1 a8 64 88 99 40 94 f6 06 5f 98 4c 75
04f0   3f 00 2b f6 97 dd d3 8a 99 40 07 f0 16 48 50 fc
0500   78 3f 00 75 02 9a 08 1b 8d 99 40 6e a3 01 bc 05
0510   12 84 3f 00 b3 7b f2 b0 50 8d 99 40 6e a3 01 bc
0520   05 12 84 3f 00 46 b6 f3 fd 54 8e 99 40 61 32 55
0530   30 2a a9 43 3f 00 c0 ec 9e 3c ac 8f 99 40 61 32
0540   55 30 2a a9 53 3f 00 98 dd 93 87 45 95 99 40 ba
0550   49 0c 02 2b 87 76 3f 00 82 73 46 94 b6 95 99 40
0560   22 fd f6 75 e0 9c 71 3f 00 f2 d2 4d 62 90 96 99
0570   40 07 f0 16 48 50 fc 78 3f 00 ed 0d be 30 59 97
0580   99 40 07 f0 16 48 50 fc 78 3f 00 d7 12 f2 41 0f
0590   9a 99 40 67 d5 e7 6a 2b f6 87 3f 00 e0 9c 11 a5
05a0   bd 9c 99 40 22 fd f6 75 e0 9c 71 3f
```

4 Basic mechanisms

A - Client start using the API

When a client starts up, it should start listening to multicast events and queue the received messages to make sure that it does not miss changes while building initial data with the following steps.

1. Retrieve the current complete instrument dictionary for all the open instruments with the new API call `multicast/get_instrument_dictionary`

The result of the call is a mapping between instrument ID and Instrument name
e.g. :

```
{"jsonrpc":"2.0","result":{"ETH-PERPETUAL":2,"BTC-PERPETUAL":3,"BTC-15APR22":
4}}
```

The mapping can also be retrieved for the given currency/product from the regular API using the `public/get_instruments` call.
The instrument data now contains the numeric instrument ID. The client can use this to build an instrument ID/Name dictionary.

2. Retrieve the status of the order books for each instrument using the `public/get_order_book` (or `public/get_order_book_by_instrument_id`) API call.

Apply the eventually missed order book changes, based on the change ID in the retrieved books and in the received multicast events.

B - Client start using snapshot multicasts

The client can also build the initial state of the books based on the regular snapshots. See details of the snapshot messages (Instrument/instrument_v2/ticker/snapshot) are described above.

When using snapshots, it is still recommended to queue changes, as they may happen while the platform builds/sends the snapshots. From the queued book level changes, if there are any with higher Change ID than the one indicated in the snapshot, they should be applied. Obviously changes before the snapshot Change ID are not relevant and should be discarded. To handle the (unlikely) corner case when there are only events collected with higher change ID than the snapshot (or for that matter : API call result) the Prev Change ID of the event should also be validated against the change ID of the book snapshot.

Since snapshots contain both the instrument ID and Instrument name, as well as other static instrument data, probably there is no need to retrieve the dictionary from the API.

Note that while the `multicast/get_instrument_dictionary` API call contains only the list of active instruments, the snapshot stream includes closed/settled instruments until they are archived (usually after a day).

When joining a snapshot channel in the middle of a snapshot batch (i.e. packets received immediately when joining), it is recommended to ignore those and wait for the next batch, to make sure that a complete snapshot batch, with all instruments on the channel is received/processed.

C - New instruments or closed instruments

When a new instrument is added to the system, an `instrument` event (see above) is generated which contains the ID and the name of the new instrument. The new book will also be part of the snapshots.

When instruments are closed/settled there are also `instrument` events generated that can be used by the client to stop maintaining the book.

D - Channel packet recovery

A client can detect a missed packet, based on a jump in sequence numbers. Note that the sequence number is a 32 bit integer and it will roll over to 0 after reaching its max value (2^32-1), which the clients should be able to recognize as a normal sequence, and not treat it as an error.
Sequence numbers can also reset to 0 after a system maintenance which may require the clients to rebuild their initial data (e.g. as described above).

D.1 API call

When the client detects missed packets, it can retrieve them using the `multicast/get_packets` call.

The parameters of the call

channel_id	The integer ID of the multicast channel where the packet was missed
start	Start sequence number
end	End sequence number

The response contains the list of requested packets excluding the framing header (only the SBE messages) in base64 encoded binary format, that can be decoded the same way as a multicast packet (after base64 decoding).

API call limits

It is possible to request max 1000 consecutive packets on the channel with this call. The call also has a rate limit of 1 req/sec with a burst of 3 req/sec. If more than 1000 packets are lost on the channel at a time then it is preferred to do a full channel recovery using either of the client start mechanisms described in 4.A and 4.B rather than trying to recover with this API.

Example

Request

```
api/v2/multicast/get_packets?channel_id=15&end=2&start=1
```

Response


```
{
	"jsonrpc":"2.0",
	"result":[
		{
			"seq":1,
			"packet":"6QMBAAEAGQABAAAAAQAAABuxRACAAQAAAAAAAAAAAAAFAAAA
		AAAAAAESAAEAAAAAAAAAzczMzBxy5kAAAAAAAAAkQOsDAQABACgAAAAAAAEAAAAb
		sUQAgAEAAAAAAAAAAAAAAAAAAAAAADNzMzMHHLmQAAAAAAAACRA"
		},
		{
			"seq":2,
			"packet":"6QMBAAEAGQABAAAAAQAAAIWxRACAAQAABQAAAAAAAAAHAAAA
		AAAAAAESAAEAAAAAAAAEA7FG4Hr185UAAAAAAAAA0QOsDAQABACgAAAAAAAEAAACF
		sUQAgAEAAOxRuB69fOVAAAAAAAAAANEDNzMzMHHLmQAAAAAAAACRA"
		}
	],
	"usIn":1649273556304472,
	"usOut":1649273556304731,
	"usDiff":259,
	"testnet":false
}
```

5 Developer information

Next to this document, the following files are provided to aid client development.

A - Sample captures

These are taken from a development test setup and can be used as a reference to inspect with Wireshark (using the plugin we created), or replay packets with tools like tcpreplay. Additional traces with new information are provided separately e.g. for new messages that should be used to

Replay sample capture

tcpreplay is available on most linux distributions as a package. After installation the pcap file can be replayed e.g.

```
sudo tcpreplay -p 1 -i lo sample_capture_v1_6.pcapng
```

With this command the packets in the capture file are replayed on the local (loop) interface with 1 packet per second speed.

Note that tcpreplay can replay packets only to the network. While local replay is visible e.g. with tcpdump on the host itself, it is not possible to actually receive them with an application. In order to use it with a test client, tcpreplay needs to be given a real physical interface and the client will need to run on another host attached to the same network.

sample_capture_v1_6.pcapng	multicast snapshot and message samples

B - Wireshark dissector plugin

A LUA wireshark dissector plugin **deribit_sbe.lua** has been created that allows Wireshark packet capture/analysis tool to display the content of the multicast packets.

The dissector has been tested with Wireshark version 3.4.8 on linux (ubuntu 20.04).

To add the dissector to Wireshark, copy the file to the directory of **'Personal LUA plugins'** or **'Global LUA plugins'** which can be found in Wireshark under **'Help' -> 'About Wireshark' -> 'Folders'**.

The loading of the plugin can be verified by clicking on the **'Plugins'** tab in the same window.

The plugin is automatically assigned to UDP ports 6100 and 6101, however if decoding on other ports are required (e.g. test streams) then they should be explicitly assigned in

wireshark.

After the plugin is loaded, and the packets are not decoded then **Right click on any of the packets** and select **'Decode As…'** from the context menu. In the dialog select the UDP destination port number of the packet and assign the **'DERIBIT SBE'** decoder to it.

The LUA file can also be used as a sample for understanding the message decoding.

C - SBE XML definition

The **deribit_multicast.xml** XML file contains the definitions of messages in the SBE XML format (see SBE specification) and can be used as a reference for creating the decoder, or as input for the code generation for different languages.

D - Code generation tools

The tool that can be used for code generation (requires java to be installed) is created by Real Logic. It is only bundled for convenience, and it is not supported by us.

For any inquiry or issues (not related to eventual issues with the XML definition), the project github repository is the best resource (https://github.com/real-logic/simple-binary-encoding). Also newer versions of the tool can be found there. It can generate library code from the XML specification in different programming languages (c, c++, java, go, rust). The tool is open source under the Apache 2.0 license.

Please note that XML definition is only provided to assist client development, it is not used in the platform (and not extensively tested) so feedback about any issues found in XML is welcome.
Sample commands are provided as scripts (e.g. **generate_c.sh**) as an example to generate libraries.

E - Note about optional fields in SBE

In SBE the sequence and size of fields is fixed. If a field is marked as optional, it only means that the field takes a special "nullValue". The SBE specification specifies the default null values for different types of fields unless "nullValue" is specifically defined in a field. For integer fields the default null value is $2^{bitsize}$-1, for floats it is a NaN value.

6 Change history

Date	Version	Summary
6 April 2022	1.1	Initial published
11 April 2022	1.2	Change of message header structure (blockLength moved to the top of the header). Add a new API call to retrieve the instrument dictionary. Improve message descriptions with types and sample messages. Updated XML spec, sample trace and LUA dissector
13 April 2022	1.3	Fix the instrumentId field in the XML specification, update generated code libraries
2 May 2022	1.4	- Introduce Ticker message - Replace Quote event with Ticker - Extend the Instrument message with static instrument data - Include the Instrument message in the snapshots - Remove instrument name from snapshot message (included in the instrument message) - Include the Ticker message in the snapshots - Change the order of messages in the packet when a book changes (Trades/Ticker/Book) - Improve packet filling (e.g. large snapshot messages that are split to multiple packet don't start from a new packet, but fill the space after Instrument/Ticker messages) - Provide sample trace with production like multicast channel configuration - Update the wireshark LUA dissector with the changes in the messages
13 May 2022	1.4.1	- Fix root block length of the instrument message sample and provide new sample trace
23 May 2022	1.4.2	- Fix XML definition of instrumentKind enum (XML change) - Fix XML snippet in the document for float nullValues in the ticker message (doc only change) - Add note to snapshots about closed/settled instruments (doc only change) - Add comment about mid batch joining of snapshot channel (doc only change) - Change name of pcap sample file to match the doc version to avoid confusion
09 June 2022	1.4.3	- Fix XML definition of optionType enum (XML and result in generated client code change) - Fix XML snippet in the document for optionType in instrument messages (doc only change) - Add note about packet framing header sequence

		number range (doc only change) - Add note about the non sequential nature of the book change IDs (doc only change) - Update included sbe code generation tool from Real Logic to latest release (see https://github.com/real-logic/simple-binary-encoding/releases). See notes in 5.D regarding the tool. - Rename pcap sample file to match the doc version
12 July 2022	1.5	- Added combo specific new states and instrument kinds (XML change, but only impacts new combo channels) - Added notes about combo behavior in the message descriptions, where relevant (doc only change) - More clarification of packet sequence numbers in section 4.D (doc only change) - Clarification about using floats in section A.5 (doc only change) - Clarification about checking Prev Change ID in recovery (doc only change) - Renamed sample pcap to match the doc version number (doc only change) - Added note about tcpreplay tool (doc only change) - Added note about packet recovery API limits (doc only change)
14 October 2022	1.5.1	- Added new message types snapshotStart and snapshotEnd
3 November 2022	1.5.2	- Added new message type comboLegs
11 November 2022	1.5.3	- Added new message types priceIndex and rfq
31 January 2023	1.5.4	- Corrected enum used in rfq
6 March 2023	1.5.5	- Added spots
6 April 2023	1.5.6	- Rename all references to “Future Type” into “Instrument Type”
27 April 2023	1.5.7	- Add explanation about empty change list in Order book change
22 May 2023	1.5.8	- Added optional presence to open_interest ticker field. - Updated estimated_delivery_price field to -nan in example ticker message for spot.
15 June 2023	1.6	- Add new instrument v2 message to remove the deprecated rfq state field and add the tick size steps group to instruments - Removed messageHeader field from XML message specs, as it is implicitly added by the

		code generators - Added information about which channel price index messages are sent - Updated XML specs in this document from the XML spec file - Added information about optional fields in messages - Updated pcap sample
20 July 2023	1.6.1	- Include both (with and without messageHeaders in each message) XML definitions in the dev pack. - Add a note about this in Chapter A (on page 3, before A.1). - Update the code generator JAR in the dev pack to the latest (1.28.3) from Real-Logic (https://mvnrepository.com/artifact/uk.co.real-logic/sbe-all/1.28.3)
2 December 2023	1.6.2	- Update test and prod channel allocation docs only in the dev pack for new USDT channels (no change in this document, only updated version nr.)