

CLIENT MIGRATION GUIDE · US PERPETUAL FUTURES

US Perps is moving to the Kraken matching engine

One guide for the whole migration. What's changing, why, what you need to do, and who to contact, followed by the full connectivity, FIX and SBE reference.

PART A · MIGRATION OVERVIEW

01	At a glance
02	Why we're making this change
03	What it means for you
04	Contract specifications
05	Migration timeline
06	Readiness checklist
07	Common pitfalls
08	Contacts & support
09	FAQ

PART B · TECHNICAL REFERENCE

10	Connectivity
11	Colocation
12	FIX order entry
13	FIX drop copy
14	SBE market data
APPENDIX	
A	Source documents
B	Glossary

PART A

Migration overview

What's changing, why, and what you need to do.

01 At a glance

Kraken US Perpetual Futures is moving from the legacy Bitnomial matching engine to Kraken’s new matching engine. Kraken spot and offshore perpetuals run on separate engines today and will move onto this same engine later. US Equity perps launch on it first and US crypto perps follow about a month later. It’s built on what Kraken has learned running those engines.

Before cutover you get a UAT environment to connect, build and certify against. Your production trading stays on the current Bitnomial engine the whole time. Nothing changes in production until the cutover date.

WHAT New matching engine, new endpoints & protocols	UAT OPENS Date TBC	CUTOVER Date TBC	YOUR ACTION Connect, build & certify in UAT before cutover
---	------------------------------	----------------------------	--

Start here

Trading & operations What changes, timeline, contacts	Network & infrastructure Routes, cross-connects, endpoints
FIX developers Order entry & drop copy	Market data developers SBE feed, snapshots, recovery

02 Why we’re making this change

US Perps is joining the infrastructure Kraken already runs at scale. For you, that means:

Built for high load	Designed from Kraken’s experience running spot and offshore perpetuals through peak volatility. Both will move onto this engine later.
Lower latency	Direct colocation and Beeks deliver under 1 ms to the engine. AWS London is about 2 ms.
Richer market data	SBE with a full order-by-order (L3) book, sequenced per instrument. It’s built to move to UDP multicast as a transport change, not a rewrite.
One Kraken integration	The FIX protocol, API keys and auth model are shared with Kraken’s other venues. If you already trade Kraken over FIX, most of your work carries over.

03 What it means for you

What changes

- **New endpoints**, hosted in Equinix London. Reach them by cross-connect, Beeks or AWS eu-west-2.
- **New credentials**. A FIX API key from Kraken Pro, HMAC-SHA512 logon, and new CompIDs issued at onboarding. The order entry TargetCompID is `KRAKEN-PERPS-TRD`.
- SBE market data authenticates separately, with its own Negotiate handshake: a raw (not base64) HMAC-SHA512 signature and a nanosecond nonce.
- Kraken's FIX 4.4 order entry protocol, plus a separate drop copy session.
- Market data moves to SBE, split across an incremental feed and a snapshot connection.
- Market data identifies instruments by an opaque uint64 Instrument Registry ID. Don't derive it from an asset ID.
- **ClOrdID must be a positive integer** of at most 19 digits, strictly increasing within the session, for example a microsecond timestamp. A UUID is not accepted. With `RelaxedClOrdID=Y` (tag 5033) it need not increase, but you must never reuse one in the session.
- **Account (1) is required on every order**: a non-zero numeric end-client id. It also identifies the end client for self-trade prevention.
- **Engine rejects arrive as Execution Reports** with `ExecType` and `OrdStatus` 8 and an `OrdRejReason`, not only as `BusinessMessageReject`.
- **Cancel and amend rejects arrive differently**. Cancel rejects come as `OrderCancelReject` (35=9) where enabled (UAT today) and as `BusinessMessageReject` (35=j) otherwise; amend rejects are 35=j or 35=3, never 35=9.
- Several order features are not supported on US Perps: `DisplayQty`, `EffectiveTime` and `Margin=1` are rejected, reduce-only is not supported, and `ValidUntilTime` is only range-checked.
- Cancel-on-disconnect is on by default, done by the FIX gateway.
- The daily 22:00 UTC sequence reset is still to be confirmed for 24/5 trading.

What stays the same

- Production trading runs on the Bitnomial engine until cutover. Your current strategies are unaffected.
- You still reach the venue through your FCM. Clearing is unchanged.
- Conditional orders (stops, take-profits) are still held and triggered by your FCM.
- Order entry stays on FIX 4.4.

REVIEW NOTE · INTERNAL

Confirm the "stays the same" list against the Bitnomial legacy spec: product set and symbols, FIX 4.4 on legacy, FCM trigger model. Also decide how we message London hosting to US clients, and whether open orders or GTCs migrate at cutover.

Who on your team needs to act

TEAM	WHAT THEY DO	READ
Network & infra	Choose a route. Order cross-connects or set up AWS peering. Supply source IPs.	10 · 11
FIX / OMS developers	Implement logon, ClOrdID, Account, order types and reject handling. Certify in UAT.	12
Market data developers	Build the SBE decoder and the snapshot and recovery logic.	14
Ops, risk & compliance	Connect drop copy for reconciliation. Update runbooks for the new rollover and cancel-on-disconnect.	13
Clearing / FCM liaison	Confirm your FCM arrangement covers the new engine. This gates production access separately from the build.	5

04 Contract specifications

Ten equity perpetual futures, each covering 100 shares of a single US stock. Each stock's CF Hybrid Equity Index drives the mark price and funding. There is no routine settlement: contracts are cash settled in USD on termination.

Common contract terms

These apply to all ten products.

TERM	SPECIFICATION
Minimum order notional	Varies with price: 0.0001 contract (0.01 shares) × the current share price
Price band	2%
Price limit	10%
Reportable level	25 contracts
Margin and leverage	15% of position value under the security-futures margin rule (17 CFR part 41 subpart E; SEC Rules 242.400-406), so leverage up to 6.67x. The house margin level above this floor is still to be set.
Listing standards	To list: \$20B market cap, 20M+ shares of deliverable supply and \$100M average daily traded value over six months. To stay listed: \$10B market cap, 20M shares and \$100M daily traded value.
Settlement	No routine settlement. Contracts are cash settled in USD only on termination, for example a delisting or a cash acquisition.
Reference index	Each stock's CF Hybrid Equity Index (e.g. AAPL CF Hybrid Equity Index): national stock exchange quotes 4:00am–8:00pm ET; Blue Ocean ATS (BOATS) 8:00pm–4:00am ET.
Order quantity (FIX tag 38)	In contracts. Minimum 0.0001 contract (0.01 shares).
Order types / time in force	Natively: market and limit orders; GTC, IOC and FOK. No Day or GTD. Post-only flag and self-trade prevention supported. Stops and take-profits are available only through an FCM, which holds and triggers them.
Trading hours	24/5: Sunday 8:00pm ET to Friday 8:00pm ET (Sunday 19:00 CT to Friday 19:00 CT), with no daily break. Closed on market holidays, with early closes, as set by the approved session calendar. Launch-window closures (6 Dec 2026 to 31 Mar 2027): Dec 24 early close; Dec 25; Jan 1; Jan 18; Feb 15; Mar 26 (Good Friday). Trading halts whenever the underlying stock is in a regulatory halt.
Halt states	Closed: scheduled close (weekend, holiday or early close), shown with the reopen time. Halted (corporate action): from 8:00 PM ET until the reopen gate passes, shown with the reopen time. Trading halted: the index stopped the contract, or the reason is unknown. No reason or reopen time is shown. Halted (primary market): the underlying stock is in a regulatory halt. Trading reopens when that halt lifts.
Funding rate interval and methodology	Every 8 hours at 03:00, 11:00 and 19:00 CT, every calendar day including weekends and holidays. The rate is the time-weighted average premium of the contract's impact bid/ask (book depth up to \$50,000 notional per side) over the CF Hybrid Equity Index, sampled every 15 seconds. The interest component is 0% with a ±0.001% adjustment, and the total rate has no cap. When the rate is positive, longs pay shorts. $\text{Payment} = \text{rate} \times \text{net contracts} \times 100 \times \text{Settlement Price}$, applied through the next variation margin cycle. Funding continues while the market is closed, at the last calculated rate. Full calculation: Bitnomial's filed Securities Perpetual Pricing method.

TERM	SPECIFICATION
Corporate-action treatment	Event notice: Corporate actions apply only to the affected contract; contract terms do not change. Scheduled events, halt windows, and reopening status are published through the corporate-action calendar and matching-engine status. Position and cash treatment: Forward/reverse splits and stock dividends adjust open positions to preserve economic value. Cash dividends, special dividends and spin-offs produce a separate per-share cash adjustment, paid by shorts to longs through margin/clearing. All resting orders in the affected name are cancelled for a corporate-action halt. Halt and reopen: The affected name halts at 8:00 PM ET before the effective date. Ordinary cash dividends, special dividends under 25% and stock dividends under 25% reopen as soon as the reopen gate passes, after a short minimum halt (default 5 minutes after 8:00 PM ET). Splits, reverse splits, spin-offs, mergers, symbol changes, cash or special dividends of 25% or more of the prior close (launch scope to be confirmed) and stock dividends of 25% or more reopen from 8:00 AM ET. Reopening is post-only; market makers should follow the published status for the live reopen time. Reopen gate: A name reopens only when the DevExperts event is confirmed, the CF Benchmarks index shows the post-event price, the index price is valid and the DCO has the switchover notice. It does not reopen on a timeout. Acquisitions and delistings (under review): In a stock acquisition, positions switch to the acquirer at the deal ratio, rounded down, with any fraction paid in cash. A cash acquisition goes reduce-only 3 days before the effective date, then cash-settles at the deal price. A delisting or bankruptcy terminates the contract at a final price taken from the VWAP of the last trade, then a top-of-book TWAP, then the index.

REVIEW NOTE · INTERNAL

The 19 Oct FIX review found that GTD (59=6 with `ExpireTime`) is accepted on NewOrderSingle. That conflicts with “No Day or GTD” in the order types row above (Confluence v72). Confirm which is correct before this guide is published.

Product schedule

SYMBOL	UNDERLYING	CONTRACT SIZE	MIN TRADABLE UNIT (CONTRACTS)	TICK SIZE (\$/SHARE)	TICK VALUE (\$/CONTRACT)	POS. LIMIT (CONTRACTS)
AAPL-PERP	Apple Inc.	100 shares	0.0001	\$0.01	\$1.00	200,000
MSFT-PERP	Microsoft Corporation	100 shares	0.0001	\$0.01	\$1.00	200,000
NVDA-PERP	NVIDIA Corporation	100 shares	0.0001	\$0.01	\$1.00	200,000
TSLA-PERP	Tesla, Inc.	100 shares	0.0001	\$0.01	\$1.00	200,000
SPCX-PERP	Space Exploration Technologies Corp. (SpaceX)	100 shares	0.0001	\$0.01	\$1.00	200,000
AMZN-PERP	Amazon.com, Inc.	100 shares	0.0001	\$0.01	\$1.00	200,000
AVGO-PERP	Broadcom Inc.	100 shares	0.0001	\$0.01	\$1.00	200,000
MU-PERP	Micron Technology, Inc.	100 shares	0.0001	\$0.01	\$1.00	200,000
GOOG-PERP	Alphabet Inc.	100 shares	0.0001	\$0.01	\$1.00	200,000

SYMBOL	UNDERLYING	CONTRACT SIZE	MIN TRADABLE UNIT (CONTRACTS)	TICK SIZE (\$/SHARE)	TICK VALUE (\$/CONTRACT)	POS. LIMIT (CONTRACTS)
PLTR-PERP	Palantir Technologies Inc.	100 shares	0.0001	\$0.01	\$1.00	200,000

05 Migration timeline

Both engines run side by side until cutover. You build and test on the new engine while trading on the old one.

	1 · ANNOUNCE	2 · UAT OPENS	3 · CERTIFY	4 · PROD READY	5 · CUTOVER
Bitnomial engine	Production trading continues				Retired
Kraken engine UAT	Connect, build, test, certify				
Kraken engine prod	No-trade check				Live trading

- Announcement** Now
 This guide is published. Talk to your account manager about entitlement and your connectivity route. Start the FCM conversation in parallel.
- UAT opens** Date TBC
 Network routes provisioned, UAT credentials and ComplIDs issued. Connect to all four UAT endpoints.
- Build & certify** Date TBC
 Integrate order entry, drop copy and market data. Complete certification against UAT with Kraken.
- Production readiness** Date TBC
 Production credentials issued. Connectivity and access confirmed during a short no-trading window.
- Cutover** Date TBC
 Trading moves to the Kraken matching engine. Legacy Bitnomial engine connectivity is retired.

06 Readiness checklist

Send your account manager the items in the first group to start onboarding. Tick the rest off as you go.

Send to your account manager

- Account ID and first five characters of your FIX API public key, per session
- Source IPs for UAT and production (AWS London only)
- Number of sessions and their purpose: order entry, drop copy, market data
- Expected message rates and instrument counts
- Chosen connectivity route, plus Equinix entity details for a cross-connect
- Technical and operational contacts

Build

- FIX logon with HMAC-SHA512 and NTP-synced clock
- Numeric ClOrdID generation (ever-increasing positive integer) and a numeric Account (1) on every order
- Handle rejects on 35=3, 35=9 and 35=j, and Execution Reports with 150=8
- Drop copy session with ExecID dedup
- SBE Negotiate signing: raw 64-byte HMAC-SHA512 over the encoded Negotiate message
- Buffer-then-snapshot book recovery

Before cutover

- UAT certification complete
- FCM clearing arrangement confirmed
- Production credentials received and tested
- Cutover-day contacts confirmed

07 Common pitfalls

These cause most failed connections and certification delays.

Clock drift at logon

LOGON

UAT accepts a nonce within 60 s of Kraken server time; the production window is to be confirmed. Sync with NTP.

Wrong CIOrdID format

NEWORDERSINGLE

US Perps accepts only unique numeric CIOrdIDs: a positive integer of at most 19 digits, ever-increasing, such as a microsecond timestamp. A non-integer gets a session-level Reject (35=3, reason 5). Without `RelaxedCIOrdID=Y`, a valid COMB UUIDv4 passes session validation and is rejected only on NewOrderSingle (35=j, “numeric CIOrdID required on this venue”).

Missing or non-numeric Account

NEWORDERSINGLE

Account (1) is required on every order and must be a non-zero numeric end-client id. Orders without it are rejected with “Account is required on this venue”, and non-numeric values with “numeric account id required”.

Sending unsupported order fields

NEWORDERSINGLE

`DisplayQty`, `EffectiveTime` and `Margin=1` are rejected with 35=j. `ValidUntilTime` is only range-checked and is rejected on an amend. Reduce-only (`ExecInst=E`) is not supported. On an amend only `ExecInst=P` is accepted, and leaving it out clears post-only.

Cancels reach further than your session

CANCEL

A single cancel reaches any order cached for your SenderCompID, earlier connections included, and Side and Symbol must match. Mass cancel types 1 and 7 sweep every order of the Account on each instrument where you have a cached open order, whoever placed it.

Status requests for finished orders

ORDERSTATUSREQUEST

Only live orders cached for your SenderCompID are answered. A finished or unknown order comes back as 35=j with `BusinessRejectReason 101`.

Snapshot before subscribing

RECOVERY

Subscribe and buffer on 4500 first, then snapshot on 4501.

Expecting to recover missed trades

SBE RECOVERY

There is no trade snapshot. A gap in `tradeSeq` can be detected, but missed trades cannot be recovered. Book, instrument and trading-session gaps are recovered from a snapshot.

Base64-encoding the SBE signature

SBE NEGOTIATE

The SBE signature is the raw 64-byte HMAC-SHA512 output over SHA256 of the encoded Negotiate message. Unlike FIX tag 554, it is not base64.

Deriving instrument IDs, or using the wrong connection

SBE

InstrumentId is an opaque Instrument Registry ID: take it from InstrumentDefinition, never from an asset ID. A Snapshot request on 4500, or Subscribe/Unsubscribe on 4501, returns `UnsupportedFeed`.

Mixing hostnames

ENDPOINTS

`colo-london.` and `eu-west-2-1.` resolve to different ingress. Use the one that matches your path.

Pinning IPs

RULES

DNS has a short TTL. Resolve at connect time.

Missing the BGP secret

CROSS-CONNECT

Confirm you have it before scheduling cross-connect turn-up.

Late allow-list requests

ALLOW-LISTING

AWS allow-listing isn't instant and applies per network and per API key.

Only handling one reject type

REJECTS

Cancel rejects arrive as OrderCancelReject (35=9) where enabled (UAT today) and as BusinessMessageReject (35=j) otherwise. Amend rejects are 35=j or 35=3, never 35=9. Orders the matching engine rejects after acknowledging them arrive as Execution Reports with 150=8 / 39=8.

Duplicate drop copy fills

DROP COPY

Delivery is at-least-once. On reconnect the feed resumes where you stopped. Deduplicate on ExecID.

Leaving clearing to the end

TIMELINE

Your FCM arrangement gates production access separately. Start it now.

08 Contacts & support

Your account manager is your first contact for everything. They'll bring in the right team.

PRIMARY CONTACT

Your account manager

Entitlements, onboarding, credentials, IP allow-list changes, LOAs, scheduling.

TECHNICAL ONBOARDING

Integration support

FIX and SBE questions, UAT issues, certification. Telegram chat.

NETWORK

Colocation & connectivity

Cross-connects, BGP, VLANs, AWS peering and PrivateLink. Email / channel TBC.

For clearing and margin questions, contact your FCM.

REVIEW NOTE · INTERNAL

Add real support addresses or Slack Connect channels, support hours, and an escalation path for cutover weekend.

09 FAQ

Can I keep trading while I build?

Yes. Production stays on the Bitnomial engine until cutover. UAT runs in parallel and doesn't affect your live strategies.

Do I need to colocate in London?

No. AWS London (eu-west-2) gets you about 2 ms. Beeks gives colocation latency without your own hardware. From outside London, Kraken partners with BSO and Avelacom.

When will production endpoints be published?

Ahead of the production readiness phase. They'll follow the same naming pattern as UAT.

Do I need separate credentials for UAT and production?

Yes. UAT credentials and CompIDs are issued first. Production credentials follow once you certify.

Does UAT testing use my production cross-connect?

Yes. The same physical pair carries both the Prod and UAT VLANs.

Will market data move to multicast?

UDP multicast is the eventual transport. The two-connection design means that move is a transport change, not a rewrite.

What happens to my open orders at cutover?

To be confirmed. Your account manager will share cutover-day procedures ahead of the date.

PART B

Technical reference

Everything from the connectivity guide, FIX specs and SBE reference, in one place. The FIX 4.4 XML data dictionary and SBE schema XML are available from your account manager.

10 Connectivity

Kraken's core trading infrastructure is in Equinix London. The exact facility is disclosed under NDA. There are three ways in. The hostname you connect to decides which path you're on. All connections require TLS.

Connectivity options

OPTION	LATENCY	WHAT YOU NEED
Direct colocation cross-connect	< 1 ms	Signed NDA and an Equinix London presence. A dual cross-connect (two per client) against Kraken-issued LOAs.
Beeks Financial Cloud	< 1 ms	A Beeks VPS or bare-metal subscription. Same colocation route. No IP allow-listing.
AWS London (eu-west-2)	~2 ms	An AWS presence in eu-west-2, via VPC peering, PrivateLink or allow-listed public ingress. CIDR and account details are exchanged during onboarding.

Colocation or Beeks? Same latency. All cables terminate in the same meet-me room. Colocation gives you control of your own hardware, not extra speed. Beeks is the faster route to market. Beeks and Kraken are in different IBXs, so the inter-site link adds a small latency cost. A direct cross-connect from a Beeks-hosted server is supported on approval.

Outside London? Kraken partners with BSO and Avelacom for inter-region links. Indicative round trip: Europe to US about 60 ms, Europe to Tokyo about 140 ms. Inside London, a carrier circuit adds 1 to 2 ms, so a cross-connect or Beeks is faster.

Endpoints

Four services, each on its own TLS-terminated TCP endpoint. Entitlement comes as a set: a firm enabled for US Perps gets all four. Use `colo-london.` for cross-connect or Beeks, and `eu-west-2-1.` for AWS. Ports are the same on both.

SERVICE	COLOCATION & BEEKS (UAT)	AWS LONDON (UAT)
FIX order entry	<code>colo-london.vip-fix.uat.kraken.com:4010</code>	<code>eu-west-2-1.vip-fix.uat.kraken.com:4010</code>
FIX drop copy	<code>colo-london.vip-fix.uat.kraken.com:4011</code>	<code>eu-west-2-1.vip-fix.uat.kraken.com:4011</code>
SBE incremental feed	<code>colo-london.vip-sbe.uat.kraken.com:4500</code>	<code>eu-west-2-1.vip-sbe.uat.kraken.com:4500</code>
SBE snapshots	<code>colo-london.vip-sbe.uat.kraken.com:4501</code>	<code>eu-west-2-1.vip-sbe.uat.kraken.com:4501</code>

ENDPOINT	WHAT IT DOES
4010 · order entry	Submit, amend, cancel and track orders. Execution reports and rejects. Instrument definitions and trading status come from the SBE feed, not FIX. Section 12
4011 · drop copy	Read-only execution reports for reconciliation and compliance. At-least-once delivery: on reconnect the feed resumes where you stopped, so deduplicate on ExecID. Section 13
4500 · incremental feed	Sequenced live market data. Subscribe starts incremental delivery with no initial snapshot. Hold it open. Section 14
4501 · snapshots	One snapshot cycle per request. Use at startup and on recovery only. Recovery

Endpoint rules

- Use the hostname that matches your access path. The two resolve to different ingress. Mixing them won't work.
- TLS is required on every endpoint, with public DigiCert wildcard certificates.
- DNS has a short TTL. Resolve at connect time. Never bind to a static IP.
- Production endpoints aren't published yet. They'll follow the same naming pattern.
- Your account manager confirms which endpoints are enabled for your firm.

IP allow-listing

Required only over AWS London. Cross-connect and Beeks traffic doesn't need it; the cross-connect is the trust boundary.

What we need	Static source IPs for your sessions, separately for UAT and production.
Where it applies	At network level and per API key. Both must be correct.
Lead time	Not instant. Submit changes through your account manager ahead of time.
Capacity	Finite. Use as few source addresses as your architecture allows.

Onboarding sequence

01 Eligibility and entitlement confirmed with your account manager and your FCM.

02 Network route provisioned. IPs allow-listed if you're on AWS London.

03 UAT credentials and ComplIDs issued.

04 Integration and certification against UAT.

05 Production credentials issued. Connectivity and access confirmed before go-live.

06 A short no-trading window in production, then live trading opens.

Market makers reach the venue through their FCM. Run the clearing arrangement in parallel with the technical build. It gates production access on its own.

11 Colocation

US Perps is reachable over the same colocation cross-connects that already serve Kraken Spot and Futures. If you already have a cross-connect to Kraken, it will work for US Perps — no new physical connectivity is needed.

If you don't have a cross-connect yet, see the Colocation and connectivity guide on the Kraken docs site for the options and requirements, and contact your account manager to begin onboarding. Network design details are shared under NDA.

12 FIX order entry

Unified FIX v2 is the FIX 4.4 order-entry interface for **Kraken US Perps**. It covers the session layer, order entry (new, cancel, amend, mass cancel and status requests) and the execution and reject messages Kraken sends back.

Protocol

Protocol	FIX 4.4 (BeginString = FIX.4.4)
Messages	15, plus the standard header (9 fields) and trailer (1 field)
Transport	TLS required on all connections
Authentication	API key, HMAC-SHA512 password signature and nonce on Logon
Session rollover	Daily at 22:00 UTC; sequence numbers reset
Cancel-on-disconnect	On by default; done by the FIX gateway; controlled by <code>CancelOrdersOnDisconnect</code> (8674)

Messages

Session layer

MESSAGE	MSGTYPE	DIRECTION	DESCRIPTION
Logon	35=A	Client → Kraken, then Kraken → client	Authenticates the session and sets its parameters. Must be the first message after the TCP connection is established.
Logout	35=5	Either direction	Initiates or confirms the end of the session, and carries the reason for a failed logon.
Heartbeat	35=0	Either direction	Keeps the session alive during inactivity, and answers a Test Request.
TestRequest	35=1	Either direction	Probes whether the counterparty is still alive; must be answered with a Heartbeat.
ResendRequest	35=2	Either direction	Asks the counterparty to retransmit a range of messages after a sequence gap.
SequenceReset	35=4	Either direction	Resets the counterparty's expected sequence number, or stands in for messages that will not be resent.

Order entry (Client → Kraken)

MESSAGE	MSGTYPE	DESCRIPTION
NewOrderSingle	35=D	Submits a new limit or market order.
OrderCancelRequest	35=F	Cancels a single working order.
OrderCancelReplaceRequest	35=G	Amends the price, quantity or expiry of a working order.
OrderMassCancelRequest	35=q	Cancels open orders in bulk.
OrderStatusRequest	35=H	Asks for the current state of a single order.

Execution and rejects (Kraken → client)

MESSAGE	MSGTYPE	DESCRIPTION
ExecutionReport	35=8	Reports every change to an order: acknowledgement, fill, amendment, cancellation, expiry, restatement, or an answer to a status request.

MESSAGE	MSGTYPE	DESCRIPTION
OrderCancelReject	35=9	Rejects a single-order cancel. On US Perps cancel business rejects arrive as 35=9 where it is enabled (UAT today; production to be confirmed) and as 35=j otherwise, so handle both. Amend rejects arrive as 35=j or 35=3 , never 35=9 .
Reject	35=3	Session-level rejection of a message that is malformed or fails field validation.
BusinessMessageReject	35=j	Business-level rejection of a message that fails a business rule before it reaches the trading engine.

Every message starts with the [standard header](#) and ends with the [standard trailer](#).

Cancel scope

A single-order cancel ([OrderCancelRequest](#)) looks the order up among the orders cached for this SenderCompID, earlier connections included, not only the orders of the current session. [Side](#) and [Symbol](#) (or [InstrumentID](#)) must match the order.

[OrderMassCancelRequest](#) (35=q) is an engine sweep per instrument and Account on US Perps for [MassCancelRequestType](#) (530) = 1 and 7 : it cancels every order of that Account on each instrument where this SenderCompID has a cached open order, whoever placed it. Type 6 cancels this connection's orders one by one. If nothing is cached for the scope, Kraken sends no request and no reply. See [Order Mass Cancel Request](#).

How US Perps differs from Spot and Futures

Some behaviours differ by venue. The table summarises where US Perps differs from Spot and Futures; the message pages give the details. Spot and Futures are shown for comparison only. Items marked "to be confirmed" have not been verified end to end.

AREA	US PERPS	SPOT	FUTURES
ClOrdID	Positive integer, up to 19 digits	UUIDs (36 characters) accepted	Timestamp-first COMB UUIDv4 required
Cancel business reject	35=9 where enabled (UAT; production to be confirmed), otherwise 35=j	35=j	35=j
Amend reject	35=j or 35=3 , never 35=9	35=j	35=j ; amend is off in production, on in UAT
Matching-engine reject	Execution Report 150=8 / 39=8 with OrdRejReason	Not sent; 35=j	Not sent; 35=j
Single-order cancel	Any order cached for the SenderCompID; Side and Symbol or InstrumentID must match	Same cache scope; no Side or Symbol rule	Same as Spot
Mass cancel type 7	Engine sweep per instrument and Account	Batch of the cached order ids; REST and WebSocket orders not reached	Same as Spot
Order status request	Live cached orders only; others 35=j , 380=101	Orders outside the cache are found; 380=101 for unknown orders	Off in production and UAT
ExecInst on amend	Only P ; leaving it out clears post-only	Only P ; effect of leaving it out differs by route	Same rule in UAT; amend off in production
GTD	Accepted (59=6 with ExpireTime)	Supported	Supported
DisplayQty	Rejected	Supported	Not read as a size; outcome to be confirmed
EffectiveTime , Margin	Rejected; Margin=0 accepted with no effect	Supported	Forwarded; to be confirmed

AREA	US PERPS	SPOT	FUTURES
ValidUntilTime	Range-checked only; rejected on amend	Forwarded as a deadline; minimum varies	Forwarded as a deadline; minimum varies
Logon nonce	60 s in UAT; production to be confirmed	Freshness not checked	Freshness not checked
Cancel-on-disconnect	FIX gateway only	Gateway, plus a back-end backstop timer by default	Gateway only (one production deployment also arms the back-end timer)
ExecID	Opaque	Base32 of a ClOrdID-derived value plus a timestamp	Same as Spot
LastLiquidityInd	Tag 851, values 1 and 2	Tag 5050	Same as Spot
InstrumentID (48)	Accepted on orders; sent on reports when cached	Never sent; ignored outside US Perps	Same as Spot

Message reference

Logon

FIX session: admin 35=A

Authenticates the session and sets session-level parameters: the heartbeat interval, whether sequence numbers reset, and what happens to open orders on disconnect. Cancel-on-disconnect is done by the FIX gateway, which cancels the session's orders when it sees the disconnect; the matching engine's cancel-all-after timer is not used on US Perps. Trading sessions carry the API key, the HMAC-SHA512 password signature and a nonce close to Kraken server time (UAT accepts up to 60 seconds of drift; the production window is to be confirmed); keep your clock NTP-synced, because clock drift is the commonest cause of logon failure. All connections require TLS.

Direction: Client → Kraken, then Kraken → client

STANDARD HEADER 35= A

98 EncryptMethod integer **required**

Encryption method for the session. Always none, since TLS 1.3 encrypts at the transport layer.

Possible values:

- 0 : No encryption. The only value Kraken accepts - TLS 1.3 does the encrypting at the transport layer.

108 HeartBtInt integer **required**

Heartbeat interval in seconds. Kraken recommends 60.

Format: Seconds. Kraken recommends 60.

141 ResetSeqNumFlag boolean

Whether both sides reset their sequence numbers to 1 on this logon.

Possible values:

- N : Continue the existing sequence. Default - expect a ResendRequest if there is a gap.
- Y : Both sides reset MsgSeqNum to 1. Use on a first connect or after a session reset.

553 Username string

Your FIX API key, created in Kraken Pro settings with FIX as the key type. Trading sessions only.

Required: Conditional

554 Password string

Base64 HMAC-SHA512 signature over the logon fields and the nonce, keyed with the base64-decoded API secret. Trading sessions only.

Format: base64(HMAC-SHA512(base64decode(secret), SHA256(message input + nonce)))

Required: Conditional

5025 Nonce string

Current time in milliseconds since the Unix epoch, as a string. The same value must appear in the password input. UAT accepts a nonce within 60 seconds of Kraken server time; the production window is to be confirmed, so keep your clock NTP-synced. Trading sessions only.

Format: Milliseconds since the Unix epoch, as a string, within 60 s of Kraken server time (UAT).

Required: Conditional

Nonce freshness: venue differences

The nonce is still part of the signed password input on every venue.

	US PERPS	SPOT	FUTURES
Allowed drift	60 seconds in UAT; production to be confirmed	Not checked in any deployment	Not checked in any deployment

5030 ForceResetCIOrdID boolean

Whether to reset the session CIOrdID sequence on re-logon. Emergency use only.

Possible values:

- N** : Keep enforcing CIOrdID uniqueness. Default.
- Y** : Wipe the session's CIOrdID uniqueness history so previously used ids can be reused. Emergency use only.

5033 RelaxedCIOrdID boolean

Whether the strictly increasing CIOrdID requirement is relaxed. On US Perps a CIOrdID must always be a positive integer of at most 19 digits, whatever this value is. With **Y** the CIOrdID no longer has to be greater than the previous one, and you are responsible for never reusing a CIOrdID within the session. See [New Order Single](#) for how invalid values are rejected.

Possible values:

- N** : CIOrdIDs must be strictly increasing within the session. Default.
- Y** : Any positive integer (positive int64) is accepted, provided you do not reuse it within the session. Anything else is rejected with `35=3` and reason `5`.

8674 CancelOrdersOnDisconnect integer

What happens to open orders if the session disconnects. Cancel-on-disconnect is on by default and needs no onboarding step. The FIX gateway cancels the orders; on US Perps the matching engine's cancel-all-after timer is off.

Possible values:

- 0** : Cancel open orders on disconnect. Default.
- 1** : Leave orders working after disconnect.

Cancel-on-disconnect: venue differences

	US PERPS	SPOT	FUTURES
Who cancels	The FIX gateway only	The gateway cancels the orders and, by default, also arms the back end's account-wide cancel-all-after timer as a backstop	The gateway only; one production deployment also arms the back-end timer

STANDARD TRAILER

Logout

FIX

session: admin

35=5

Initiates or confirms the end of the session, and carries the reason for a failed logon.

Direction: Either direction

STANDARD HEADER 35= 5

58 Text string

Free-text description: the cancellation reason, reject reason, or order status detail.

Format: Free text set by Kraken; carries the reject or cancellation reason.

STANDARD TRAILER

Heartbeat

FIX

session: admin

35=0

Sent when no other traffic has passed within the negotiated heartbeat interval, so that both sides can tell a quiet session from a dead one. When it answers a Test Request it must echo that request's TestReqID. The interval is negotiated through HeartBtInt (108) on Logon.

Direction: Either direction

STANDARD HEADER 35= 0

112 TestReqID string

Token identifying the test request. A heartbeat answering a test request must echo the value it carried.

Required: Conditional

STANDARD TRAILER

Test Request

FIX

session: admin

35=1

Sent when nothing has arrived within the heartbeat interval and the session's liveness is in doubt. The counterparty must respond with a Heartbeat echoing TestReqID, which is how the probe is matched to its answer.

Direction: Either direction

STANDARD HEADER 35= 1

112 TestReqID string **required**

Token identifying the test request. A heartbeat answering a test request must echo the value it carried.

STANDARD TRAILER

Resend Request

FIX

session: admin

35=2

Sent by either side when an incoming MsgSeqNum is higher than expected, asking for the messages in the gap. The counterparty answers either with the original messages carrying PossDupFlag=Y, or with a Sequence Reset carrying

GapFillFlag=Y for administrative messages it will not replay. Expect one whenever a client reconnects to an existing session without resetting sequence numbers.

Direction: Either direction

STANDARD HEADER 35= 2

7 BeginSeqNo SeqNum required

First message sequence number to retransmit.

Format: Sequence number of the first message to retransmit.

16 EndSeqNo SeqNum required

Last message sequence number to retransmit. Zero requests everything from BeginSeqNo onwards.

Format: Sequence number, or 0 to request everything from BeginSeqNo onwards.

STANDARD TRAILER

Sequence Reset

FIX

session: admin

35=4

With GapFillFlag=Y this stands in for messages that will not be resent and MsgSeqNum remains meaningful. With GapFillFlag=N it is a hard reset: the counterparty jumps to NewSeqNo and ignores MsgSeqNum. Note that sequence numbers also reset at the daily 22:00 UTC session rollover.

Direction: Either direction

STANDARD HEADER 35= 4

123 GapFillFlag boolean

Whether this is a gap fill, in which case MsgSeqNum is valid, or a hard sequence reset.

Possible values:

- N** : Hard sequence reset - ignore MsgSeqNum and jump to NewSeqNo.
- Y** : Gap fill - this SequenceReset stands in for messages that will not be resent; MsgSeqNum is valid.

36 NewSeqNo SeqNum required

Sequence number the counterparty should expect next.

STANDARD TRAILER

Header & Trailer

FIX

session: admin

35=*

The baseline specification for Unified FIX v2 is FIX 4.4. A standard header must be present at the start of every message in both directions, and a standard trailer closes every message. SenderCompID and TargetCompID are communicated by Kraken during onboarding.

Standard Header

Present at the start of every message in both directions. Nine header fields are used, with the same tags, types and required flags as Unified FIX.

8 BeginString string required

FIX protocol version. Always FIX.4.4.

9 BodyLength Length **required**

Number of characters between this field and the checksum, counting the delimiter before the checksum.

Format: Character count from after this field up to and including the delimiter before tag 10.

35 MsgType string **required**

Message type of this message.

Possible values:

- 0 : Heartbeat.
- 1 : TestRequest.
- 2 : ResendRequest.
- 3 : Reject.
- 4 : SequenceReset.
- 5 : Logout.
- 8 : ExecutionReport.
- 9 : OrderCancelReject.
- A : Logon.
- D : NewOrderSingle.
- F : OrderCancelRequest.
- G : OrderCancelReplaceRequest.
- H : OrderStatusRequest.
- j : BusinessMessageReject.
- q : OrderMassCancelRequest.

49 SenderCompID string **required**

Identifier of the party sending the message. Provided by Kraken during onboarding.

Format: Assigned by Kraken at onboarding.

56 TargetCompID string **required**

Identifier of the party receiving the message. KRAKEN-PERPS-TRD for US Perps order entry sessions.

Format: KRAKEN-PERPS-TRD

34 MsgSeqNum SeqNum **required**

Sequence number of this message on the session.

43 PossDupFlag boolean

Whether the message may be a retransmission of one already sent with this sequence number.

Possible values:

- Y : Possible duplicate — this sequence number may have been sent before.
- N : Not a duplicate.

52 SendingTime UTCTimestamp **required**

Time the sender transmitted the message, in UTC, formatted YYYYMMDD-HH:MM:SS.uuu.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

122 OrigSendingTime UTCTimestamp

Original sending time of a retransmitted message; set to SendingTime when no original is available.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

Required: Conditional

Standard Trailer

Closes every message. CheckSum is always the final field.

10 CheckSum string **required**

Modulo-256 checksum of the message. Always the last field.

Format: Three digits, modulo-256 sum of every preceding byte.

New Order Single

FIX

session: trading

35=D

Submits a new order onto the perps order book.

Direction: Client → Kraken

STANDARD HEADER 35= D

11 ClOrdID string **required**

Client-assigned order identifier, unique within the trading session. On US Perps it must be a positive integer of at most 19 digits (a positive int64). Use an ever-increasing number, for example a microsecond timestamp. With `RelaxedClOrdID=Y` on Logon the value no longer has to increase, but it must still be a positive integer and must never be reused within the session.

Format: Positive integer, up to 19 digits, e.g. a microsecond timestamp.

How invalid values are rejected:

- A value that is not an integer is rejected with a Reject (35=3) and `SessionRejectReason` (373) = 5 .
- Without `RelaxedClOrdID=Y` , a valid timestamp-first COMB UUIDv4 passes session validation and is rejected only on NewOrderSingle, with a BusinessMessageReject (35=j): "numeric ClOrdID required on this venue".
- With `RelaxedClOrdID=Y` , anything other than a positive int64 is rejected with 35=3 and reason 5 .

ClOrdID: venue differences

	US PERPS	SPOT	FUTURES
Accepted format	Positive integer, up to 19 digits	UUIDs (36 characters) are accepted	Timestamp-first COMB UUIDv4 required

1 Account string **required**

Numeric end-client id identifying the end client the order is placed for. Required on every order and must be a non-zero numeric value. Orders without it are rejected with "Account is required on this venue", and non-numeric values with "numeric account id required". It also identifies the end client for self-trade prevention.

Format: Non-zero numeric end-client id.

18 ExecInst MultipleStringValue

One or more execution instructions, space-separated.

Possible values:

- `P` : Post-only. Limit orders only. The order is rejected instead of taking liquidity.
- `n` : No market price protection. Market orders only.

55 Symbol string

Perpetual futures contract, in the venue symbol form. Ten contracts are listed at launch. Either Symbol or InstrumentID must be sent. Ignored when InstrumentID is also sent: the instrument is then recognised from InstrumentID.

Format: Venue symbol.

Required: Conditional

48 InstrumentID string

Identifier of the instrument within the identification scheme named by InstrumentIDSource. Either Symbol or InstrumentID must be sent.

Where both Symbol and InstrumentID are sent, InstrumentID is what the instrument is recognised from and Symbol is ignored.

Format: Venue instrument identifier. Takes precedence over Symbol when both are sent.

Required: Conditional

22 InstrumentIDSource string

Identification scheme that InstrumentID is expressed in.

Possible values:

- 8 : Exchange symbol.

54 Side char **required**

Side of the order.

Possible values:

- 1 : Buy.
- 2 : Sell.

60 TransactTime UTCTimestamp **required**

Time of the business event in UTC, formatted YYYYMMDD-HH:MM:SS.uuu.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

38 OrderQty Qty **required**

Order quantity in contracts.

40 OrdType char **required**

Execution model of the order.

Possible values:

- 1 : Market - fill the whole quantity immediately at the best available prices.
- 2 : Limit - rest in the book, trading only at the limit price or better.

44 Price Price

Limit price of the order, denominated in the quote currency. Required for limit orders.

Format: Limit price, denominated in the quote currency.

Required: Conditional

59 TimeInForce char required

How long the order stays in effect. GTC, IOC, FOK and GTD are accepted; a change of TimeInForce on an amend is rejected.

Possible values:

- 1 : GTC - good till cancelled. Rests indefinitely.
- 3 : IOC - fill what you can immediately, cancel the rest.
- 4 : FOK - fill the entire quantity immediately or cancel the whole order.
- 6 : GTD - good till date. Rests until the expiry in `ExpireTime` (126) , which is then required.

126 ExpireTime UTCTimestamp

Expiry of a good-till-date order, in UTC. Required when TimeInForce (59) is 6 . A GTC order (59=1) that carries ExpireTime is not rejected: it silently becomes good-till-date, so leave ExpireTime out of GTC orders.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

Required: Conditional

1138 DisplayQty Qty

Not supported on US Perps. An order that carries it is rejected with a BusinessMessageReject (35=j): "Order type iceberg is not supported on this venue".

GTD and DisplayQty: venue differences

	US PERPS	SPOT	FUTURES
GTD (59=6 with <code>ExpireTime</code>)	Accepted	Supported	Supported
DisplayQty (1138)	Rejected (iceberg orders are not supported)	Supported	Not read as a size, but its presence turns the order into an iceberg order with no display size. The back-end outcome is to be confirmed, so do not send it

168 EffectiveTime UTCTimestamp

Scheduled start time. Not supported on US Perps: an order that carries it is rejected with a BusinessMessageReject (35=j).

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

62 ValidUntilTime UTCTimestamp

On US Perps the value is only range-checked: it must be between 2 and 60 seconds in the future, formatted YYYYMMDD-HH:MM:SS.uuu. It is not sent to the matching engine, so it does not reject an order that arrives late. It is rejected on an amend.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu, and between 2 and 60 seconds in the future.

ValidUntilTime: venue differences

	US PERPS	SPOT	FUTURES
Behaviour	Range-checked only; never sent to the engine; rejected on amend	Forwarded to the back end as a deadline	Forwarded to the back end as a deadline
Minimum distance	2 seconds	Varies by deployment, from 1 ms to 2 s	Varies by deployment, 50 ms or 2 s

5001 Margin char

Whether the order draws on the margin account. Margin= 1 is rejected on US Perps with a BusinessMessageReject (35=j); Margin= 0 is accepted and has no effect.

Possible values:

- 0 : Fund the order from the cash account. Default. Accepted on US Perps and has no effect.
- 1 : Fund the order from the margin account. Rejected on US Perps.

EffectiveTime and Margin: venue differences

	US PERPS	SPOT	FUTURES
EffectiveTime (168)	Rejected (35=j)	Supported	Forwarded by the gateway; whether the back end honours it is to be confirmed
Margin (5001)	1 rejected (35=j); 0 accepted, no effect	Supported	Forwarded by the gateway; whether the back end honours it is to be confirmed

7928 SelfTradePrevention char

Self-trade prevention mode, deciding which order is expired when your own orders would match. Defaults to cancelling the newest.

Possible values:

- 0 : Cancel both - the arriving and the resting order are both cancelled.
- 1 : Cancel newest - the arriving order is cancelled. Default.
- 2 : Cancel oldest - the resting order is cancelled.

STANDARD TRAILER

Order Cancel Request

FIX

session: trading

35=F

Cancels one working order. At least one of OrderID (37) or OrigClOrdID (41) must identify the target; where both are present, OrderID wins. ClOrdID (11) is the identifier of the cancel request itself. On US Perps the order is looked up among the orders cached for this SenderCompID, earlier connections included, not only the orders of the current session, and Side and Symbol (or InstrumentID) must match the order. Success returns an Execution Report with OrdStatus=4. A cancel that Kraken will not act on is answered with an OrderCancelReject (35=9) or, where that is not enabled, a BusinessMessageReject (35=j); a field-level error is answered with a Reject (35=3). Clients should handle all three until the production setting is confirmed.

Cancel: venue differences

	US PERPS	SPOT	FUTURES
Which orders a single cancel reaches	Any order cached for this SenderCompID, earlier connections included; Side and Symbol or InstrumentID must match	Same cache scope; no Side or Symbol match rule	Same as Spot
Cancel business reject	35=9 (enabled in UAT; production to be confirmed)	35=j	35=j

Direction: Client → Kraken

STANDARD HEADER 35= F

41 OrigClOrdID string

ClOrdID of the previous action on the order. On a cancel or amend it identifies the order being acted on; if OrderID is also present, OrderID takes precedence. Optional on this message: Kraken relaxes the FIX 4.4 requirement so a client may cancel by OrderID alone.

Required: Conditional

37 OrderID string

Exchange-assigned order identifier generated by Kraken; the value to quote back on cancels, amends and status requests.

Format: Assigned by Kraken; quote it back on cancels, amends and status requests.

Required: Conditional

11 ClOrdID string **required**

Client-assigned identifier of the cancel request. On US Perps it must be a positive integer of at most 19 digits: use an ever-increasing number, for example a microsecond timestamp. See [New Order Single](#) for how invalid values are rejected.

Format: Positive integer, up to 19 digits, e.g. a microsecond timestamp.

1 Account string

Numeric end-client id of the end client the order belongs to. Optional on this message and not used: the end client is taken from the original order.

Format: Non-zero numeric end-client id.

55 Symbol string

Perpetual futures contract, in the venue symbol form. Ten contracts are listed at launch. Either Symbol or InstrumentID must be sent. Must match the instrument of the order being cancelled. Ignored when InstrumentID is also sent: the instrument is then recognised from InstrumentID.

Format: Venue symbol.

Required: Conditional

48 InstrumentID string

Identifier of the instrument within the identification scheme named by InstrumentIDSource. Either Symbol or InstrumentID must be sent.

Where both Symbol and InstrumentID are sent, InstrumentID is what the instrument is recognised from and Symbol is ignored.

Format: Venue instrument identifier. Takes precedence over Symbol when both are sent.

Required: Conditional

22 InstrumentIDSource string

Identification scheme that InstrumentID is expressed in.

Possible values:

- **8** : Exchange symbol.

54 Side char **required**

Side of the order. Must match the side of the order being cancelled.

Possible values:

- **1** : Buy.
- **2** : Sell.

60 TransactTime UTCTimestamp **required**

Time of the business event in UTC, formatted YYYYMMDD-HH:MM:SS.uuu.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

STANDARD TRAILER**Order Cancel Replace Request****FIX**

session: trading

35=G

Amends a working order. Price, quantity and, on a good-till-date order, the expiry are the amendable terms; Side, OrdType and Symbol must match the original, and a change of TimeInForce is rejected because there is no field for it in the engine schema. Both OrderID (37) and OrigClOrdID (41) are required, and at least one of OrderQty (38) and Price (44) must be present. An amend that Kraken refuses is answered with a BusinessMessageReject (35=j) or, for a field-level error, a Reject (35=3). It is never answered with an OrderCancelReject (35=9).

Amend: venue differences

	US PERPS	SPOT	FUTURES
Rejected amend	35=j or 35=3 , never 35=9	35=j	35=j
Availability	Available	Available	Off in production; on in UAT with the same rules
Order lookup	OrigClOrdID finds the order and OrderID must match it	Same rule	Same rule (UAT)

Direction: Client → Kraken

STANDARD HEADER 35= G**37 OrderID** string **required**

Exchange-assigned order identifier generated by Kraken; the value to quote back on cancels, amends and status requests. Required on an amend, together with OrigClOrdID: OrigClOrdID finds the order and OrderID must match it.

Format: Assigned by Kraken; quote it back on cancels, amends and status requests.

41 OrigClOrdID string **required**

ClOrdID of the previous action on the order. On an amend it finds the order, and OrderID (37) must match that order. Unlike on a cancel, OrderID does not take precedence over OrigClOrdID.

11 ClOrdID string **required**

Client-assigned identifier of the amend request. On US Perps it must be a positive integer of at most 19 digits: use an ever-increasing number, for example a microsecond timestamp. See [New Order Single](#) for how invalid values are rejected.

Format: Positive integer, up to 19 digits, e.g. a microsecond timestamp.

1 Account string

Numeric end-client id of the end client the order belongs to. Optional on this message and not used: the end client is taken from the original order.

Format: Non-zero numeric end-client id.

18 ExecInst MultipleStringValue

Execution instructions. The only value accepted on an amend is `P` (post-only). An amend restates post-only: leaving ExecInst out removes post-only from the resting order, so send `18=P` on every amend of an order that should stay post-only. `E` or `s` is rejected with a Reject (`35=3`) and `SessionRejectReason` (`373`) = `5` . Reduce-only is not supported on US Perps.

Possible values:

- `P` : Post-only. Limit orders only. An amend restates it: leaving `18=P` out removes post-only from the resting order.

Amend ExecInst: venue differences

	US PERPS	SPOT	FUTURES
Values accepted on an amend	Only <code>P</code>	Only <code>P</code>	Only <code>P</code> (amend is off in production)
<code>P</code> left out	Post-only is cleared	No flags are sent on the ACG route, so post-only is probably kept; post-only is cleared on DirectTE (effect in the engine unconfirmed)	Same rule (UAT)
Reduce-only (<code>E</code>)	Not supported	Not accepted on amend	Not accepted on amend

55 Symbol string

Perpetual futures contract, in the venue symbol form. Ten contracts are listed at launch. Either Symbol or InstrumentID must be sent. Ignored when InstrumentID is also sent: the instrument is then recognised from InstrumentID.

Format: Venue symbol.

Required: Conditional

48 InstrumentID string

Identifier of the instrument within the identification scheme named by InstrumentIDSource. Either Symbol or InstrumentID must be sent.

Where both Symbol and InstrumentID are sent, InstrumentID is what the instrument is recognised from and Symbol is ignored.

Format: Venue instrument identifier. Takes precedence over Symbol when both are sent.

Required: Conditional

22 InstrumentIDSource string

Identification scheme that InstrumentID is expressed in.

Possible values:

- `8` : Exchange symbol.

54 Side char **required**

Side of the order.

Possible values:

- `1` : Buy.
- `2` : Sell.

60 TransactTime UTCTimestamp **required**

Time of the business event in UTC, formatted YYYYMMDD-HH:MM:SS.uuu.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

38 OrderQty Qty

New order quantity in contracts. Optional: if left out the resting quantity stands. At least one of OrderQty and Price must be present.

Required: Conditional

40 OrdType char required

Execution model of the order.

Possible values:

- 1 : Market - fill the whole quantity immediately at the best available prices.
- 2 : Limit - rest in the book, trading only at the limit price or better.

44 Price Price

New limit price of the order, denominated in the quote currency. Optional: if left out the resting price stands. At least one of OrderQty and Price must be present. Not permitted on a market order.

Format: Limit price, denominated in the quote currency.

Required: Conditional

1138 DisplayQty Qty

Not supported on US Perps. NewOrderSingle rejects iceberg orders ("Order type iceberg is not supported on this venue"), so there is no display quantity to amend.

DisplayQty: venue differences

	US PERPS	SPOT	FUTURES
DisplayQty (1138)	Not supported	Supported (iceberg orders)	Not read as a size; its presence turns the order into an iceberg order with no display size (back-end outcome unconfirmed)

126 ExpireTime UTCTimestamp

New expiry of a good-till-date order, in UTC. Only accepted on an order resting as GTD (TimelnForce 6). If left out the expiry is unchanged.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

59 TimelnForce char

Optional. If sent it must equal the order's resting TimelnForce: a change of TimelnForce on an amend is rejected.

Possible values:

- 1 : GTC - good till cancelled. Rests indefinitely.
- 3 : IOC - fill what you can immediately, cancel the rest.
- 4 : FOK - fill the entire quantity immediately or cancel the whole order.
- 6 : GTD - good till date. The expiry is amended with ExpireTime (126).

62 ValidUntilTime UTCTimestamp

Not accepted on an amend: an amend that carries ValidUntilTime is rejected.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

STANDARD TRAILER

Order Mass Cancel Request

FIX

session: trading

35=q

Cancels open orders in bulk, including orders resting in the book. MassCancelRequestType sets the scope. On US Perps types 1 and 7 are engine sweeps per instrument and Account, not per session: they cancel every order of that Account on each instrument where this SenderCompID has a cached open order, earlier connections included, whoever placed the order. If nothing is cached for the scope, Kraken sends no request to the engine and sends no reply. Each cancellation is confirmed by its own Execution Report; there is no mass cancel report.

Mass cancel: venue differences

	US PERPS	SPOT	FUTURES
Type 1 (one symbol)	Engine sweep on that instrument and Account; also accepts InstrumentID (48)	As documented: orders created or replaced on this session	Same as Spot
Type 6 (this session)	Cancels this connection's orders one by one	As documented	Same as Spot
Type 7	Engine sweep per instrument and Account for every instrument with a cached open order; misses instruments with nothing cached	Sends a batch of the order ids cached for this SenderCompID, earlier connections included. There is no cancel-all call, so REST and WebSocket orders are not reached	Same as Spot

Direction: Client → Kraken

STANDARD HEADER 35= q

11 ClOrdID string **required**

Client-assigned identifier of the mass cancel request. On US Perps it must be a positive integer of at most 19 digits: use an ever-increasing number, for example a microsecond timestamp. See [New Order Single](#) for how invalid values are rejected.

Format: Positive integer, up to 19 digits, e.g. a microsecond timestamp.

530 MassCancelRequestType char **required**

Scope of the mass cancel.

Possible values:

- 1 : Cancel by instrument. Requires Symbol (55) or InstrumentID (48). Engine sweep: for each Account that has a cached open order of this SenderCompID on that instrument, every order of that Account on the instrument is cancelled, whoever placed it. Nothing cached for the instrument means no request and no reply.
- 6 : Cancel the orders created on this connection, one by one.
- 7 : Cancel by SenderCompID. Engine sweep for every instrument where this SenderCompID has a cached open order, earlier connections included: for each such instrument and Account, every order of that Account is cancelled, whoever placed it. An instrument with nothing cached is not reached; nothing cached at all means no request and no reply.

336 TradingSessionID string

Trading session the status applies to.

55 Symbol string

Perpetual futures contract, in the venue symbol form. Ten contracts are listed at launch. Required for type 1 unless InstrumentID is sent. Ignored when InstrumentID is also sent: the instrument is then recognised from InstrumentID.

Format: Venue symbol.

Required: Conditional

48 InstrumentID string

Identifier of the instrument within the identification scheme named by InstrumentIDSource.

Where both Symbol and InstrumentID are sent, InstrumentID is what the instrument is recognised from and Symbol is ignored.

Format: Venue instrument identifier. Takes precedence over Symbol when both are sent.

22 InstrumentIDSource string

Identification scheme that InstrumentID is expressed in.

Possible values:

- 8 : Exchange symbol.

60 TransactTime UTCTimestamp **required**

Time of the business event in UTC, formatted YYYYMMDD-HH:MM:SS.uuu.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

STANDARD TRAILER**Order Status Request****FIX**

session: trading

35=H

Asks for the current state of one order, identified by ClOrdID and OrderID. The answer arrives as an Execution Report with ExecType=I, where OrdStatus carries the state. On US Perps only live orders cached for this SenderCompID are answered this way. A finished order, or any order that is not in the cache, is not found and comes back as a BusinessMessageReject (35=j) with BusinessRejectReason (380) 101 (Unknown order), or 0 when the lookup itself fails.

Order status: venue differences

	US PERPS	SPOT	FUTURES
Which orders are answered	Live orders cached for this SenderCompID only	Orders outside the cache are still found, so there is no per-session limit	Status requests are off in production and UAT
Unknown order	35=j , 380=101 (380=0 if the lookup fails)	35=j , 380=101	n/a

Direction: Client → Kraken

STANDARD HEADER 35= H**37 OrderID** string **required**

Exchange-assigned order identifier generated by Kraken; the value to quote back on cancels, amends and status requests.

Format: Assigned by Kraken; quote it back on cancels, amends and status requests.

11 ClOrdID string **required**

Client-assigned identifier of the status request. On US Perps it must be a positive integer of at most 19 digits: use an ever-increasing number, for example a microsecond timestamp. See [New Order Single](#) for how invalid values are rejected.

Format: Positive integer, up to 19 digits, e.g. a microsecond timestamp.

1 Account string

Numeric end-client id of the end client the order belongs to. Optional on this message and not used: the end client is taken from the original order.

Format: Non-zero numeric end-client id.

55 Symbol string

Perpetual futures contract, in the venue symbol form. Ten contracts are listed at launch. Either Symbol or InstrumentID must be sent. Ignored when InstrumentID is also sent: the instrument is then recognised from InstrumentID.

Format: Venue symbol.

Required: Conditional

48 InstrumentID string

Identifier of the instrument within the identification scheme named by InstrumentIDSource. Either Symbol or InstrumentID must be sent.

Where both Symbol and InstrumentID are sent, InstrumentID is what the instrument is recognised from and Symbol is ignored.

Format: Venue instrument identifier. Takes precedence over Symbol when both are sent.

Required: Conditional

22 InstrumentIDSource string

Identification scheme that InstrumentID is expressed in.

Possible values:

- 8 : Exchange symbol.

54 Side char

Side of the order.

Possible values:

- 1 : Buy.
- 2 : Sell.

STANDARD TRAILER

Execution Report

FIX session: trading 35=8

Reports every state change on an order once the engine has accepted it. ExecType says what happened — new, replaced, cancelled, expired, traded, restated, or an answer to a status request — and OrdStatus says what state the order is now in. Fills carry LastPx, LastQty, the running CumQty and AvgPx, the trade identifier, and the maker/taker flag in LastLiquidityInd.

Direction: Kraken → client

STANDARD HEADER 35= 8

37 OrderID string

Exchange-assigned order identifier generated by Kraken; the value to quote back on cancels, amends and status requests.

Format: Assigned by Kraken; quote it back on cancels, amends and status requests.

11 ClOrdID string **required**

Client-assigned order identifier, as sent by the client. On US Perps it is a positive integer of at most 19 digits.

Format: Positive integer, up to 19 digits, e.g. a microsecond timestamp.

41 OrigClOrdID string

ClOrdID of the previous action on the order. Sent on cancel and amend reports.

Required: Conditional

17 ExecID string **required**

Unique identifier of this execution report. Treat it as opaque: do not parse it or derive it from ClOrdID. On US Perps the New (150=0) acknowledgement uses a ClOrdID-based form, reports driven by the engine carry an engine-generated identifier, and an Order Status answer (150=I) repeats the ExecID of the cached report.

ExecID: venue differences

	US PERPS	SPOT	FUTURES
Format	Opaque; three forms, see above	Base32 of a ClOrdID-derived value plus a timestamp	Same as Spot

150 ExecType char **required**

Nature of the event this execution report describes.

Possible values:

- **0** : New - the order is live in the engine.
- **4** : Canceled - the order was cancelled.
- **5** : Replace - the amend was applied.
- **8** : Rejected - the matching engine refused an order it had already acknowledged, for example a post-only order that would cross, a fill-or-kill order that cannot be filled in full or an order stopped by market price protection. The report carries OrdRejReason (103) and Text (58). A request refused before it is acknowledged (an unsupported feature, an invalid field, a risk limit) arrives as a BusinessMessageReject (35=j) instead. Engine limit checks (quantity cap, price bounds and bands) map to OrdRejReason **3**, so they may arrive as 150=8 rather than 35=j; this is to be confirmed in UAT.
- **C** : Expired - the order reached its expiry.
- **D** : Restated - the engine changed the order itself, for example by re-displaying an iceberg slice.
- **F** : Trade - the order got a fill. LastPx/LastQty carry that fill.
- **I** : Order Status - this report answers an OrderStatusRequest rather than reporting an event.

39 OrdStatus char **required**

Current state of the order.

Possible values:

- **0** : New - live, nothing filled.
- **1** : Partially filled - live with fills against it.
- **2** : Filled - fully filled and closed.
- **4** : Canceled - cancelled and closed.
- **8** : Rejected - the matching engine refused the order after acknowledging it. See OrdRejReason (103) and Text (58) for the reason.
- **C** : Expired - expired and closed.

103 OrdRejReason integer

Code identifying the reason why an order was rejected by the matching engine. Present only on Execution Reports with ExecType (150) = 8 and OrdStatus (39) = 8. The gateway maps only the values below; match-time rejects use 3, 11 and 99. Which engine rejections reach 150=8 is decided by the engine.

Possible values:

- 1 : Unknown symbol.
- 2 : Exchange closed.
- 3 : Order exceeds a limit (quantity cap, price bounds and bands).
- 5 : Unknown order.
- 6 : Duplicate order.
- 11 : Unsupported order characteristic.
- 13 : Incorrect quantity.
- 99 : Other.

The other FIX 4.4 values are not sent.

Required: Conditional

Engine rejects: venue differences

	US PERPS	SPOT	FUTURES
Matching-engine reject	Execution Report 150=8 / 39=8 with OrdRejReason (103)	Not sent; arrives as 35=j	Not sent; arrives as 35=j

1 Account string

Numeric end-client id of the end client the order was placed for, as sent in Account (1) on the order.

Format: Non-zero numeric end-client id.

55 Symbol string

Perpetual futures contract, in the venue symbol form. Ten contracts are listed at launch.

Format: Venue symbol.

48 InstrumentID string

Identifier of the instrument within the Kraken instrument scheme. Present when it is cached for the order.

Format: Venue instrument identifier.

InstrumentID: venue differences

	US PERPS	SPOT	FUTURES
InstrumentID (48) on reports	Sent when cached	Never sent; InstrumentID is ignored outside US Perps	Same as Spot

54 Side char **required**

Side of the order.

Possible values:

- 1 : Buy.
- 2 : Sell.

38 OrderQty Qty

Order quantity in contracts.

40 OrdType char required

Execution model of the order.

Possible values:

- 1 : Market - fill the whole quantity immediately at the best available prices.
- 2 : Limit - rest in the book, trading only at the limit price or better.

44 Price Price

Limit price of the order, denominated in the quote currency. Required for limit orders.

Format: Limit price, denominated in the quote currency.

Required: Conditional

59 TimeInForce char

How long the order stays in effect. GTC, IOC, FOK and GTD are the supported durations; a change of TimeInForce on an amend is rejected.

Possible values:

- 1 : GTC - good till cancelled. Rests indefinitely.
- 3 : IOC - fill what you can immediately, cancel the rest.
- 4 : FOK - fill the entire quantity immediately or cancel the whole order.
- 6 : GTD - good till date. The expiry is in ExpireTime (126).

32 LastQty Qty

Quantity bought or sold on this fill.

Required: Conditional

31 LastPx Price

Price of this fill.

Required: Conditional

151 LeavesQty Qty required

Quantity still working in the book.

14 CumQty Qty required

Total quantity filled on the order so far.

381 GrossTradeAmt Amt

Gross amount traded on the order. The New (150=0) acknowledgement carries 0 .

1138 DisplayQty Qty

May be present on reports. Display quantity cannot be set on US Perps orders: NewOrderSingle rejects iceberg orders.

6 AvgPx Price required

Average price across all fills on the order.

60 TransactTime UTCTimestamp

Time of the business event in UTC, formatted YYYYMMDD-HH:MM:SS.uuu.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

58 Text string

Free-text description: the cancellation reason, reject reason, or order status detail.

Format: Free text set by Kraken; carries the reject or cancellation reason.

851 LastLiquidityInd integer

Code indicating whether this fill added liquidity to or removed liquidity from the market. Only **1** and **2** are sent.

Possible values:

- 1** : The fill added liquidity to the order book.
- 2** : The fill took liquidity from the order book.

Required: Conditional

Liquidity indicator: venue differences

	US PERPS	SPOT	FUTURES
Where it is carried	Tag 851 (1 or 2)	Tag 5050, not 851	Same as Spot

1003 TradeID string

Exchange-assigned identifier of the trade.

Required: Conditional

STANDARD TRAILER**Order Cancel Reject****FIX**session: trading35=9

Rejects a single-order cancel that Kraken will not act on. The FIX gateway sends it itself in some cases, for example for an unknown order, so it is not only a rejection by the matching engine. It is sent only where it is enabled (UAT today; the production setting is to be confirmed); otherwise a cancel business reject arrives as a BusinessMessageReject (35=j), so handle both.

Amend rejections never arrive as 35=9. They arrive as a BusinessMessageReject (35=j) or, for a field-level error, a Reject (35=3), so CxlRejResponseTo (434) is always **1**. Field-level errors on a cancel are also answered with 35=3.

Cancel rejects: venue differences

	US PERPS	SPOT	FUTURES
OrderCancelReject (35=9)	Sent for cancels (UAT; production to be confirmed)	Not sent	Not sent
Cancel business reject	35=9 , or 35=j where 35=9 is not enabled	35=j	35=j
Amend reject	35=j or 35=3	35=j	35=j (amend is off in production, on in UAT)
Unknown order on a cancel	35=9 with CxlRejReason (102) 1	35=j with BusinessRejectReason (380) 1	Same as Spot

Direction: Kraken → client

STANDARD HEADER 35= 9

37 OrderID string **required**

Exchange-assigned order identifier generated by Kraken; the value to quote back on cancels, amends and status requests. `NONE` when the order is unknown.

Format: Assigned by Kraken; quote it back on cancels, amends and status requests. `NONE` for an unknown order.

11 ClOrdID string **required**

Client-assigned identifier of the cancel request this rejection answers. On US Perps it must be a positive integer of at most 19 digits: use an ever-increasing number, for example a microsecond timestamp. See [New Order Single](#) for how invalid values are rejected.

Format: Positive integer, up to 19 digits, e.g. a microsecond timestamp.

41 OrigClOrdID string

ClOrdID of the order that was being cancelled. Optional on this message: FIX 4.4 requires it, but because a cancel may quote OrderID alone, a reject for an order Kraken never knew has no ClOrdID to echo.

Required: Conditional

39 OrdStatus char **required**

Current state of the order. `8` accompanies OrderID `NONE` : the order is unknown.

Possible values:

- `0` : New - live, nothing filled.
- `1` : Partially filled - live with fills against it.
- `2` : Filled - fully filled and closed.
- `4` : Canceled - cancelled and closed.
- `8` : Rejected - used with OrderID `NONE` when the order is unknown.
- `C` : Expired - expired and closed.

Pending New (`A`), Replaced (`5`) and Pending Replace (`E`) are defined by FIX but are never sent on US Perps.

434 CxlRejResponseTo char **required**

Whether the rejection answers a cancel request or a cancel/replace request. Always `1` on US Perps: an amend is never rejected with OrderCancelReject.

Possible values:

- `1` : This rejects an OrderCancelRequest.
- `2` : This rejects an OrderCancelReplaceRequest. Defined by FIX but never sent on US Perps: amend rejects arrive as `35=j` or `35=3` .

102 CxlRejReason integer

Why the cancel was refused. Only `1` and `99` are sent on US Perps.

Possible values:

- `1` : Unknown order.
- `99` : Other.

The other FIX values (`0` , `2` , `3` , `4` , `5` , `6`) are not sent. For example `3` (order already pending cancel) would need an empty error, which the engine route never produces.

58 Text string

Free-text description: the cancellation reason, reject reason, or order status detail.

Format: Free text set by Kraken; carries the reject or cancellation reason.

STANDARD TRAILER

Reject (Session)

FIX

session: admin

35=3

Session-level rejection. Kraken discards any message that is garbled, cannot be parsed, or fails a data integrity or field-level check, and reports it here with the sequence number and message type of the offending message. RefTagID names the offending tag where one can be identified. A FIX field-level validation failure on an order arrives here rather than as a business reject.

Direction: Kraken → client

STANDARD HEADER 35= 3

45 RefSeqNum SeqNum required

MsgSeqNum of the message being rejected.

371 RefTagID integer

Tag that caused the rejection.

Required: Conditional

372 RefMsgType string

MsgType of the message being rejected.

373 SessionRejectReason integer

Why the message was rejected at session level.

Possible values:

- 0 : Invalid tag number.
- 1 : Required tag missing.
- 2 : Tag not defined for this message type.
- 3 : Undefined tag.
- 4 : Tag specified without a value.
- 5 : Value is incorrect or out of range for this tag. For example, a ClOrdID that is not a positive integer (with RelaxedClOrdID=Y , anything other than a positive int64), or an ExecInst of E or S on an amend.
- 6 : Incorrect data format for value.
- 7 : Decryption problem.
- 8 : Signature problem.
- 9 : CompID problem.
- 10 : SendingTime accuracy problem.
- 11 : Invalid MsgType.
- 12 : XML validation error.
- 13 : Tag appears more than once.
- 14 : Tag out of required order.
- 15 : Repeating group fields out of order.
- 16 : Incorrect NumInGroup count.
- 17 : Field delimiter inside a non-data value.
- 99 : Other.

58 Text string

Free-text description: the cancellation reason, reject reason, or order status detail.

Format: Free text set by Kraken; carries the reject or cancellation reason.

STANDARD TRAILER

Business Message Reject

FIX session: trading 35=j

Business-level rejection, used when a message is well formed but fails a business rule, either at the gateway or when the matching engine admits it. This is where unsupported order features, an invalid Account, unknown symbols and orders, a ClOrdID that is not increasing or is already in use, rate limit breaches, risk limits and requests that time out surface. BusinessRejectRefID echoes the ClOrdID of the rejected message and BusinessRejectReason gives the code, with the detail in Text.

Direction: Kraken → client

STANDARD HEADER 35= j

45 RefSeqNum SeqNum

MsgSeqNum of the message being rejected.

372 RefMsgType string **required**

MsgType of the message being rejected.

379 BusinessRejectRefID string

ClOrdID of the message being rejected.

Required: Conditional

380 BusinessRejectReason integer **required**

Why the message was rejected.

Possible values:

- **0** : Other - used for nearly every business reject: unsupported order features, an invalid Account, a ClOrdID that is not increasing or already in use, rate limit breaches, risk limits and engine refusals. Also used when the lookup behind an OrderStatusRequest fails. Read Text (58).
- **1** : Unknown id - an amend names an order the session does not hold. A cancel for an unknown order is answered with an OrderCancelReject (35=9 , CxlRejReason **1**) where that is enabled.
- **2** : Unknown instrument - Symbol or InstrumentID matches no tradable contract.
- **101** : Unknown order - an OrderStatusRequest for an order that is finished or not in the cache.
- **104** : Order too old - defined, but effectively never sent on US Perps: a finished order is answered with **101** .

Values **3** , **4** , **5** , **6** , **8** , **100** , **102** and **103** are defined by the Unified FIX dictionary but are not sent on US Perps order entry: a rate limit breach, a reused ClOrdID and an engine refusal all arrive as **0** , with the cause in Text (58).

Business rejects: venue differences

	US PERPS	SPOT	FUTURES
Unknown order on a cancel	35=9 with CxlRejReason 1 (where enabled)	35=j , 380=1	35=j , 380=1
Unknown order on an amend	35=j , 380=1	35=j , 380=1	35=j , 380=1
Unknown order on a status request	35=j , 380=101	35=j , 380=101	Status requests are off

58 Text string

Free-text description: the cancellation reason, reject reason, or order status detail.

Format: Free text set by Kraken; carries the reject or cancellation reason.

STANDARD TRAILER

13 FIX drop copy

The **Kraken US Perps drop copy** is a FIX 4.4 session that delivers a copy of every Execution Report on your account: acknowledgements, fills, amendments, cancellations, expiries, restatements and status answers. Business messages flow one way, from Kraken to the client.

On the drop copy, the Execution Report is identical to the one sent on the [Unified FIX v2](#) order-entry session. `InstrumentID (48)` carries the venue instrument identifier.

Protocol

Protocol	FIX 4.4 (<code>BeginString = FIX.4.4</code>)
Scope	Session layer, standard header and trailer, and the Execution Report

Messages

Session layer (either direction)

MESSAGE	MSGTYPE	DESCRIPTION
Logon	<code>35=A</code>	Opens the session and agrees the heartbeat interval, sequence-number reset and credentials. Must be the first message after the connection is established.
Logout	<code>35=5</code>	Initiates or confirms the end of the session, and carries the reason for a failed logon.
Heartbeat	<code>35=0</code>	Keeps the session alive during inactivity, and answers a Test Request.
TestRequest	<code>35=1</code>	Probes whether the counterparty is still alive; must be answered with a Heartbeat.
ResendRequest	<code>35=2</code>	Asks the counterparty to retransmit a range of messages after a sequence gap.
SequenceReset	<code>35=4</code>	Resets the counterparty's expected sequence number, or stands in for messages that will not be resent.
Reject	<code>35=3</code>	Session-level rejection of a message that is malformed or fails field validation.

Execution (Kraken → client)

MESSAGE	MSGTYPE	DESCRIPTION
ExecutionReport	<code>35=8</code>	The drop copy payload. Reports every change to an order: acknowledgement, fill, amendment, cancellation, expiry, restatement, trade correction, or an answer to a status request.

Every message starts with the [standard header](#) and ends with the [standard trailer](#).

Recovery

On reconnect, the feed resumes from where the client stopped, including after a Logon with `ResetSeqNumFlag (141)=Y`. Execution Reports that are delivered again after a reconnect carry `PossResend (97)=Y`.

What the client should do:

- De-duplicate Execution Reports by `ExecID (17)`. A report with `PossResend (97)=Y` may be one you have already processed.
- Do not rely on a sequence-number reset to skip missed reports. The feed resumes from where you stopped even when `ResetSeqNumFlag (141)=Y`.

Message reference

Logon

FIX

session: drop copy

35=A

Opens the session and agrees its parameters: the heartbeat interval, whether both sides reset sequence numbers, and the credentials. Credentials are required: the logon is authenticated the same way as the Unified FIX v2 trading session, with the API key, the HMAC-SHA512 password signature and a nonce within five seconds of Kraken server time. Without them the logon fails. Must be the first message after the transport connection is established. Alongside the standard FIX 4.4 Logon fields it carries a NoMsgTypes group and venue-defined tags in the 5025 and 9000 ranges: Nonce, user and publication preferences, and ApplicationName / Version / Vendor identification.

Direction: Either direction

STANDARD HEADER 35= A

98 EncryptMethod integer **required**

Encryption method for the session. Where TLS protects the transport the only meaningful value is 0 (None).

Possible values:

- 0 : None.
- 1 : PKCS.
- 2 : DES.
- 3 : PKCS/DES.
- 4 : PGP/DES.
- 5 : PGP/DES-MD5.
- 6 : PEM/DES-MD5.

108 HeartbeatInt integer **required**

Heartbeat interval in seconds. This dialect names the field HeartbeatInt rather than the FIX 4.4 standard HeartBtInt.

Format: Interval in seconds.

95 RawDataLength Length

Length in bytes of the RawData field that follows.

Format: Byte count.

Required: Conditional

96 RawData data

Free-form binary payload. Not used for credentials; send them in Username (553) and Password (554).

Format: Raw data; length given by RawDataLength (95).

Required: Conditional

141 ResetSeqNumFlag boolean

Whether both sides reset their sequence numbers to 1 on this logon.

Possible values:

- Y : Both sides reset MsgSeqNum to 1. Use on a first connect or after a session reset.
- N : Continue the existing sequence. Expect a ResendRequest if there is a gap.

789 NextExpectedMsgSeqNum SeqNum

Next MsgSeqNum the sender expects to receive. An alternative to a Resend Request for resynchronising at logon.

Format: Sequence number.

383 MaxMessageSize Length

Largest message in bytes the sender is willing to receive.

Format: Byte count.

384 NoMsgTypes NumInGroup

Number of message-type entries that follow, declaring which message types the session will send or receive.

Required: Conditional

REPEATING GROUP**372 RefMsgType** string

MsgType being referred to.

Format: Any MsgType value.

385 MsgDirection char

Whether the referenced message type is sent or received by the party declaring it.

Possible values:

- **S** : The session sends this message type.
- **R** : The session receives this message type.

464 TestMessageIndicator boolean

Whether this session is a test session rather than production.

Possible values:

- **Y** : Yes.
- **N** : No.

553 Username string **required**

Your FIX API key, created in Kraken Pro settings with FIX as the key type.

554 Password string **required**

Base64 HMAC-SHA512 signature over the logon fields and the nonce, keyed with the base64-decoded API secret.

Format: base64(HMAC-SHA512(base64decode(secret), SHA256(message input + nonce)))

5025 Nonce string **required**

Current time in milliseconds since the Unix epoch, as a string, within five seconds of Kraken server time. The same value must appear in the password input. Clock drift is the commonest cause of logon failure.

Format: Milliseconds since the Unix epoch, as a string, within 5 s of Kraken server time.

9001 UserType integer

Venue-defined. Category of user establishing the session.

Format: Venue-defined integer.

9006 StrategyPreference integer

Venue-defined. Strategy handling preference for the session.

Format: Venue-defined integer.

9007 TimeStampPreference integer

Venue-defined. Timestamp granularity or format preference for the session.

Format: Venue-defined integer.

9008 PublishClearingAllocations integer

Venue-defined. Whether clearing allocations are published to this session.

Format: Venue-defined integer.

9009 PublishMiFIDRegisteredDetails integer

Venue-defined. Whether MiFID registered details are published to this session.

Format: Venue-defined integer.

9010 PublishMktCreationRealtime integer

Venue-defined. Whether market creation events are published to this session in real time.

Format: Venue-defined integer.

9450 ApplicationName string

Venue-defined. Name of the client application connecting.

9451 ApplicationVersion string

Venue-defined. Version of the client application.

9452 ApplicationVendor string

Venue-defined. Vendor of the client application.

STANDARD TRAILER

Logout

FIX

session: drop copy

35=5

Initiates or confirms the end of the session. On a failed logon the reason arrives here in Text. EncodedTextLen and EncodedText are the standard FIX 4.4 pair for carrying the same text in a non-ASCII encoding.

Direction: Either direction

STANDARD HEADER 35= 5

58 Text string

Free-text description. Carries the reject reason, the logout reason, or other detail.

Format: Free text set by the sender.

354 EncodedTextLen Length

Length in bytes of the EncodedText field that follows.

Format: Byte count.

Required: Conditional

355 EncodedText data

Text repeated in a non-ASCII encoding, as named by MessageEncoding (347).

Format: Raw data; length given by EncodedTextLen (354).

Required: Conditional

STANDARD TRAILER

Heartbeat

FIX

session: drop copy

35=0

Sent when no other traffic has passed within the negotiated heartbeat interval, so that each side can tell a quiet session from a dead one. When it answers a Test Request it must echo that request's TestReqID.

Direction: Either direction

STANDARD HEADER 35= 0

112 TestReqID string

Token identifying a test request. A Heartbeat answering a Test Request must echo the value the request carried.

Required: Conditional

STANDARD TRAILER

Test Request

FIX

session: drop copy

35=1

Probes whether the counterparty is still alive when nothing has arrived within the heartbeat interval. The counterparty must answer with a Heartbeat echoing TestReqID, which is how the probe is matched to its answer.

Direction: Either direction

STANDARD HEADER 35= 1

112 TestReqID string **required**

Token identifying a test request. A Heartbeat answering a Test Request must echo the value the request carried.

STANDARD TRAILER

Resend Request

FIX

session: drop copy

35=2

Asks the counterparty to retransmit a range of messages after a sequence gap is detected. The counterparty answers either with the original messages carrying PossDupFlag=Y, or with a Sequence Reset carrying GapFillFlag=Y for administrative messages it will not replay.

Direction: Either direction

STANDARD HEADER 35= 2

7 BeginSeqNo SeqNum **required**

First message sequence number to retransmit.

Format: Sequence number of the first message to retransmit.

16 EndSeqNo SeqNum **required**

Last message sequence number to retransmit. Zero requests everything from BeginSeqNo onwards.

Format: Sequence number, or 0 for everything from BeginSeqNo onwards.

STANDARD TRAILER

Sequence Reset

FIX

session: drop copy

35=4

With GapFillFlag=Y this stands in for messages that will not be resent and MsgSeqNum remains meaningful. With GapFillFlag=N it is a hard reset: the counterparty jumps to NewSeqNo and ignores MsgSeqNum.

Direction: Either direction

STANDARD HEADER 35= 4

123 GapFillFlag boolean

Whether this is a gap fill, in which case MsgSeqNum stays valid, or a hard sequence reset.

Possible values:

- **Y** : Gap fill - this SequenceReset stands in for messages that will not be resent; MsgSeqNum stays valid.
- **N** : Hard sequence reset - ignore MsgSeqNum and jump to NewSeqNo.

36 NewSeqNo SeqNum **required**

Sequence number the counterparty should expect next.

STANDARD TRAILER

Reject

FIX

session: drop copy

35=3

Session-level rejection of a message that is malformed, cannot be parsed, or fails a field-level validation check. RefSeqNum identifies the offending message and RefTagID names the offending tag where one can be identified. A session-level reject is not the same thing as a business rejection.

Direction: Either direction

STANDARD HEADER 35= 3

45 RefSeqNum SeqNum **required**

MsgSeqNum of the message being rejected.

Format: Sequence number.

371 RefTagID integer

Tag that caused the rejection, where one can be identified.

Format: FIX tag number.

Required: Conditional

372 RefMsgType string

MsgType being referred to.

Format: Any MsgType value.

Required: Conditional

373 SessionRejectReason integer

Why the message was rejected at session level.

Possible values:

- 0 : Invalid tag number.
- 1 : Required tag missing.
- 2 : Tag not defined for this message type.
- 3 : Undefined tag.
- 4 : Tag specified without a value.
- 5 : Value is incorrect or out of range for this tag.
- 6 : Incorrect data format for value.
- 7 : Decryption problem.
- 8 : Signature problem.
- 9 : CompID problem.
- 10 : SendingTime accuracy problem.
- 11 : Invalid MsgType.
- 12 : XML validation error.
- 13 : Tag appears more than once.
- 14 : Tag out of required order.
- 15 : Repeating group fields out of order.
- 16 : Incorrect NumInGroup count.
- 17 : Field delimiter inside a non-data value.
- 99 : Other.

58 Text string

Free-text description. Carries the reject reason, the logout reason, or other detail.

Format: Free text set by the sender.

354 EncodedTextLen Length

Length in bytes of the EncodedText field that follows.

Format: Byte count.

Required: Conditional

355 EncodedText data

Text repeated in a non-ASCII encoding, as named by MessageEncoding (347).

Format: Raw data; length given by EncodedTextLen (354).

Required: Conditional

STANDARD TRAILER

Header & Trailer

FIX session: drop copy 35=*

The baseline specification for the US Perps drop copy is FIX 4.4. A standard header must be present at the start of every message in both directions, and a standard trailer closes every message. SenderCompID and TargetCompID are communicated by Kraken during onboarding.

Standard Header

Present at the start of every message in both directions. The full standard FIX 4.4 header is supported: thirty fields, including the SecureData, XmlData and NoHops constructs that most drop-copy sessions never populate. Six fields are required; the

remaining twenty-four are optional and may be left unused.

8 BeginString string **required**

FIX protocol version. Always FIX.4.4.

9 BodyLength Length **required**

Number of characters between this field and the checksum, counting the delimiter before the checksum.

Format: Character count from after this field up to and including the delimiter before tag 10.

35 MsgType string **required**

Message type of this message.

Possible values:

- **0** : Heartbeat.
- **1** : TestRequest.
- **2** : ResendRequest.
- **3** : Reject.
- **4** : SequenceReset.
- **5** : Logout.
- **8** : ExecutionReport.
- **A** : Logon.

49 SenderCompID string **required**

Identifier of the party sending the message. Assigned during onboarding.

Format: Assigned by Kraken at onboarding.

56 TargetCompID string **required**

Identifier of the party receiving the message. Assigned during onboarding.

Format: Assigned by Kraken at onboarding.

115 OnBehalfOfCompID string

CompID of the firm the message is sent on behalf of, when a third party is relaying.

128 DeliverToCompID string

CompID of the firm the message is ultimately to be delivered to, when a third party is relaying.

90 SecureDataLen Length

Length in bytes of the SecureData field that follows.

Format: Byte count.

Required: Conditional

91 SecureData data

Encrypted portion of the message. Unused where TLS protects the transport.

Format: Raw data; length given by SecureDataLen (90).

Required: Conditional

34 MsgSeqNum SeqNum **required**

Sequence number of this message on the session.

Format: Sequence number of this message on the session, per direction.

50 SenderSubID string

Sub-identifier within SenderCompID, e.g. a desk or trader.

142 SenderLocationID string

Location of the sending party, e.g. a country or state code.

57 TargetSubID string

Sub-identifier within TargetCompID.

143 TargetLocationID string

Location of the receiving party.

116 OnBehalfOfSubID string

Sub-identifier within OnBehalfOfCompID.

144 OnBehalfOfLocationID string

Location of the OnBehalfOfCompID party.

129 DeliverToSubID string

Sub-identifier within DeliverToCompID.

145 DeliverToLocationID string

Location of the DeliverToCompID party.

43 PossDupFlag boolean

Whether the message may be a retransmission of one already sent with this sequence number.

Possible values:

- **Y** : Possible duplicate — this sequence number may have been sent before.
- **N** : Not a duplicate.

97 PossResend boolean

Whether the message was resent from the application layer rather than the session layer, so the receiver should check for a duplicate at the business level.

Possible values:

- **Y** : Yes.
- **N** : No.

52 SendingTime UTCTimestamp **required**

Time the sender transmitted the message, in UTC.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

122 OrigSendingTime UTCTimestamp

Original sending time of a retransmitted message; set to SendingTime when no original is available. Required on a retransmission.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

Required: Conditional

212 XmlDataLen Length

Length in bytes of the XmlData field that follows.

Format: Byte count.

Required: Conditional

213 XmlData data

XML payload carried inside the FIX message.

Format: Raw data; length given by XmlDataLen (212).

Required: Conditional

347 MessageEncoding string

Character encoding of the Encoded* fields in this message.

Format: EUC-JP, ISO-2022-JP, Shift_JIS or UTF-8.

369 LastMsgSeqNumProcessed SeqNum

Highest MsgSeqNum the sender has processed from the counterparty. Lets each side confirm its position without a Test Request.

Format: Sequence number.

627 NoHops NumInGroup

Number of hop entries that follow, recording each third party the message passed through.

REPEATING GROUP**628 HopCompID** string

CompID of the third party that forwarded this message.

629 HopSendingTime UTCTimestamp

Time that third party sent the message.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

630 HopRefID SeqNum

MsgSeqNum the message carried on that hop.

Format: Sequence number.

Standard Trailer

Closes every message. CheckSum is always the final field. SignatureLength and Signature are the standard FIX 4.4 message-signing pair and are optional; they are rarely used where the transport is already protected by TLS.

93 SignatureLength Length

Length in bytes of the Signature field that follows.

Format: Byte count.

89 Signature data

Message signature. Optional, and rarely used where TLS already protects the transport.

Format: Raw data; length given by SignatureLength (93).

10 CheckSum string **required**

Modulo-256 checksum of the message. Always the last field.

Format: Three digits, modulo-256 sum of every preceding byte.

Execution Report

FIX

session: trading

35=8

Reports every state change on an order once the engine has accepted it. ExecType says what happened — new, replaced, cancelled, expired, traded, restated, or an answer to a status request — and OrdStatus says what state the order is now in. Fills carry LastPx, LastQty, the running CumQty and AvgPx, the trade identifier, the maker/taker flag in LastLiquidityInd and a fee entry in the NoMiscFees group.

Direction: Kraken → client

STANDARD HEADER 35= 8**37 OrderID** string

Exchange-assigned order identifier generated by Kraken; the value to quote back on cancels, amends and status requests.

Format: Assigned by Kraken; quote it back on cancels, amends and status requests.

11 ClOrdID string **required**

Client-assigned order identifier, as sent by the client. On US Perps it is a positive integer of at most 19 digits.

Format: Positive integer, up to 19 digits, e.g. a microsecond timestamp.

41 OrigClOrdID string

ClOrdID of the previous action on the order. Sent on cancel and amend reports.

Required: Conditional

17 ExecID string **required**

Unique identifier of this execution report. Treat it as opaque: do not parse it or derive it from ClOrdID. On US Perps the New (150=0) acknowledgement uses a ClOrdID-based form, reports driven by the engine carry an engine-generated identifier, and an Order Status answer (150=I) repeats the ExecID of the cached report.

150 ExecType char **required**

Nature of the event this execution report describes.

Possible values:

- **0** : New - the order is live in the engine.
- **4** : Canceled - the order was cancelled.
- **5** : Replace - the amend was applied.
- **8** : Rejected - the matching engine refused an order it had already acknowledged, for example a post-only order that would cross, a fill-or-kill order that cannot be filled in full or an order stopped by market price protection. The report carries OrdRejReason (103) and Text (58). Engine limit checks (quantity cap, price bounds and bands) map to OrdRejReason **3**, so they may be reported this way as well; this is to be confirmed in UAT.
- **A** : Pending New - not sent on US Perps, because EffectiveTime is rejected on order entry.
- **C** : Expired - the order reached its expiry.
- **D** : Restated - the engine changed the order itself, e.g. a trigger firing or a reprice. ExecRestatementReason (378) says why.
- **F** : Trade - the order got a fill. LastPx/LastQty carry that fill.
- **I** : Order Status - this report answers an OrderStatusRequest rather than reporting an event.

39 OrdStatus char required

Current state of the order.

Possible values:

- 0 : New - live, nothing filled.
- 1 : Partially filled - live with fills against it.
- 2 : Filled - fully filled and closed.
- 4 : Canceled - cancelled and closed.
- 5 : Replaced - the amend succeeded.
- 8 : Rejected - the matching engine refused the order after acknowledging it. See OrdRejReason (103) and Text (58) for the reason.
- A : Pending New - not live yet.
- C : Expired - expired and closed.
- E : Pending Replace - the amend is accepted but the new parameters are not live yet, e.g. a trade was in flight when the amend arrived.

103 OrdRejReason integer

Code identifying the reason why an order was rejected by the matching engine. Present only on Execution Reports with ExecType (150) = 8 and OrdStatus (39) = 8. The gateway maps only the values below; match-time rejects use 3, 11 and 99. Which engine rejections reach 150=8 is decided by the engine.

Possible values:

- 1 : Unknown symbol.
- 2 : Exchange closed.
- 3 : Order exceeds a limit (quantity cap, price bounds and bands).
- 5 : Unknown order.
- 6 : Duplicate order.
- 11 : Unsupported order characteristic.
- 13 : Incorrect quantity.
- 99 : Other.

The other FIX 4.4 values are not sent.

Required: Conditional

1 Account string

Numeric end-client id of the end client the order was placed for, as sent in Account (1) on the order.

Format: Non-zero numeric end-client id.

582 CustOrderCapacity integer

Capacity in which the customer traded, as reported to futures exchanges.

Possible values:

- 1 : Exchange member trading for their own account.
- 2 : Clearing firm trading for its own proprietary account.
- 3 : Member trading on behalf of another exchange member.
- 4 : Any other customer capacity not listed above.

55 Symbol string

Perpetual futures contract, in the venue symbol form. Ten contracts are listed at launch.

Format: Venue symbol.

48 InstrumentID string

Identifier of the instrument within the Kraken instrument scheme. Present when it is cached for the order.

Format: Venue instrument identifier.

54 Side char **required**

Side of the order.

Possible values:

- **1** : Buy.
- **2** : Sell.

38 OrderQty Qty

Order quantity in contracts. A reduce-only order is clamped to the remaining position at submission; clamped to zero is a reject.

40 OrdType char **required**

Execution model of the order. On a triggered order this reflects the resulting model (market or limit) once the trigger has fired.

Possible values:

- **1** : Market - fill the whole quantity immediately at the best available prices.
- **2** : Limit - rest in the book, trading only at the limit price or better.

423 PriceType integer

Code indicating the format or units in which a price is expressed.

Possible values:

- **1** : Price expressed as a percentage of par or nominal.
- **2** : Price quoted per unit or per share.
- **3** : Absolute fixed price amount.
- **4** : Price quoted as a discount, a percentage below par.
- **5** : Price quoted as a premium, a percentage above par.
- **6** : Price quoted as a spread over a benchmark.
- **7** : Price expressed as a TED spread price.
- **8** : Price expressed as a TED spread yield.
- **9** : Price expressed as a yield rather than a price.
- **10** : Fixed price used for cabinet trades of near-worthless options.
- **11** : Variable price used for cabinet trades of near-worthless options.

44 Price Price

Limit price of the order, denominated in the quote currency. Required for limit orders.

Format: Limit price, denominated in the quote currency.

Required: Conditional

388 DiscretionInst char

Reference price the FCM trigger watches. Defaults to the last trade price.

Possible values:

- **1** : Track the index price as the trigger reference.
- **5** : Track the last trade price as the trigger reference. Default.

59 TimeInForce char

How long the order stays in effect. GTC, IOC, FOK and GTD are the supported durations; a change of TimeInForce on an amend is rejected.

Possible values:

- **1** : GTC - good till cancelled. Rests indefinitely.
- **3** : IOC - fill what you can immediately, cancel the rest.
- **4** : FOK - fill the entire quantity immediately or cancel the whole order.
- **6** : GTD - good till date. The expiry is in ExpireTime (126).

168 EffectiveTime UTCimestamp

Scheduled start time. Not used on US Perps: EffectiveTime is rejected on order entry.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

18 ExecInst MultipleStringValue

One or more execution instructions, space-separated. Reduce-only is clamped by the FCM against the live position at submission, not by the engine, which holds no position or margin context.

Possible values:

- **E** : Reduce-only - may shrink an existing margin position but never open or extend an opposite one.
- **s** : Single fee - fee reported per trade on the execution report. Mandatory on Derivatives, and the only fee option there.

32 LastQty Qty

Quantity bought or sold on this fill.

Required: Conditional

31 LastPx Price

Price of this fill.

Required: Conditional

336 TradingSessionID string

Trading session the status applies to.

151 LeavesQty Qty **required**

Quantity still working in the book.

14 CumQty Qty **required**

Total quantity filled on the order so far.

381 GrossTradeAmt Amt

Gross amount traded on the order. The New (150=0) acknowledgement carries **0** .

1138 DisplayQty Qty

May be present on reports. Display quantity cannot be set on US Perps orders: NewOrderSingle rejects iceberg orders.

6 AvgPx Price **required**

Average price across all fills on the order.

60 TransactTime UTCTimestamp

Time of the business event in UTC, formatted YYYYMMDD-HH:MM:SS.uuu.

Format: UTC timestamp, YYYYMMDD-HH:MM:SS.uuu

58 Text string

Free-text description: the cancellation reason, reject reason, or order status detail.

Format: Free text set by Kraken; carries the reject or cancellation reason.

494 Designation string

Additional free-text designation or reference attached to the allocation or order.

851 LastLiquidityInd integer

Code indicating whether this fill added liquidity to or removed liquidity from the market. Only **1** and **2** are sent.

Possible values:

- **1** : The fill added liquidity to the order book.
- **2** : The fill took liquidity from the order book.

Required: Conditional

136 NoMiscFees NumInGroup

Number of fee entries that follow.

Required: Conditional

REPEATING GROUP**137 MiscFeeAmt** Amt

Absolute fee value on a fill. Fees are cumulative across the previous trades on the order.

Required: Conditional

138 MiscFeeCurr Currency

Asset the fee is charged in.

Required: Conditional

139 MiscFeeType char

Category of fee.

Possible values:

- **4** : Exchange fee. The only value Kraken sends.

Required: Conditional

891 MiscFeeBasis integer

How the fee is charged.

Possible values:

- **2** : Percentage - the fee is a percentage rate.

Required: Conditional

1003 TradeID string

Exchange-assigned identifier of the trade.

Required: Conditional

78 NoAllocs NumInGroup

Number of subaccount allocations that follow. Always 1 for broker accounts.

REPEATING GROUP

79 AllocAccount string

Account identifier of the subaccount the order is directed to. Broker accounts only.

Required: Conditional

STANDARD TRAILER

14 SBE market data

Kraken US Perps publishes public market data over **Simple Binary Encoding (SBE)**: a compact, fixed-layout binary format designed for low-latency decoding. The protocol is defined by two schemas — a **session** schema for TLS session negotiation, authentication, subscriptions and keepalive, and a **market data** schema for instrument definitions, order books, trades and trading status.

Schemas

SCHEMA	PACKAGE	SCHEMA ID	WIRE VERSION	SEMANTIC VERSION	BYTE ORDER	ROLE
Session	Kraken.MarketDataAPIProto	33	3	5.0.0	littleEndiann	TLS session negotiation, authentication, subscriptions and keepalive
Market data	Kraken.MarketDataProto	34	11	10.0.0	littleEndiann	Instrument definitions, books, trades and trading status

Every SBE message starts with the standard SBE `messageHeader` (`blockLength:uint16` , `templateId:uint16` , `schemaId:uint16` , `version:uint16`). Use `schemaId` and `templateId` to select the decoder. On the wire, each SBE message is carried inside a TCP frame that adds a framing header in front of the SBE `messageHeader` — see [Framing](#).

Compatibility rule: new optional or versioned fields require a schema version increase. Existing field and template IDs are never reused or reordered.

Connections

The session uses two connections:

- **Snapshot connection** — send a `SubscriptionRequest` with `subscriptionRequestType` = `Snapshot` to receive one snapshot (instrument definitions, order book or trading session status).
- **Live connection** — send `Subscribe` or `Unsubscribe` to start or stop incremental delivery for a feed.

Feeds are selected with `MarketDataFeed` : `InstrumentDefinition` , `OrderBookL3` , `Trade` , `InstrumentStatus` and `TradingSessionStatus` .

Framing

Each TCP frame carries exactly one SBE message, in both directions. TLS protects the TCP stream but does not change the frame bytes.

bytes 0..5	SOFH
bytes 6..13	Kraken TCP header
bytes 14..21	SBE message header
bytes 22..end	SBE message body

OFFSET	SIZE	FIELD	TYPE	BYTE ORDER	DESCRIPTION
0	4	messageLength	uint32	big endian	Total frame size in bytes, including the SOFH and the Kraken TCP header
4	2	encodingType	uint16	big endian	0xEB50 : Kraken TCP header followed by an SBE 1.0 message in little endian byte order
6	8	SendingTime	int64	little endian	Unix epoch nanoseconds (Kraken TCP header)
14	8	SBE messageHeader	comPOSITE	little endian	blockLength , templateId , schemaId , version

OFFSET	SIZE	FIELD	TYPE	BYTE ORDER	DESCRIPTION
22	variable	SBE message body		little endian	Root block, repeating groups and variable-length data

The first six bytes are the Simple Open Framing Header (SOFH). The eight-byte Kraken TCP header is a Kraken extension to the usual SOFH and SBE layout: a generic SOFH reader must skip those eight bytes before decoding the SBE `messageHeader`.

- `SendingTime` is diagnostic only. The gateway stamps a shared market data frame once, when it enters the shared fanout, so every client sees the same time for that frame. It is not a kernel transmit timestamp, and it must not be used for authentication or replay checks.
- **Reading frames.** TCP may split one frame across several reads or return several frames in one read. Read the full SOFH, check its fields, then wait until `messageLength` bytes have arrived.
- **Protocol errors.** An unsupported `encodingType`, a `messageLength` below 22 (the minimum frame size), or a `messageLength` above the service limit is a protocol error. Kraken closes the connection instead of skipping the frame.
- **Repeating groups.** Before reading a group, verify that its block length and entry count fit inside the enclosing frame. Check the size arithmetic for overflow.

Frame counting

Frame counting is implicit: there is no sequence number on the wire for session control.

- Each direction has its own counter. Both start at 1 on a new connection, never wrap, and have no daily reset.
- A sender advances its counter after it successfully sends a complete frame. A receiver advances its counter after it validates the complete SOFH and SBE frame.
- Every frame counts, including negotiation, subscription, heartbeat, termination and market data messages. Zero means no frame has been received.
- `AdminHeartbeat.lastReceivedClientSequence` reports the latest client frame Kraken accepted, and `SubscriberHeartbeat.lastReceivedServerSequence` reports the latest server frame the client processed.
- An acknowledgement greater than the number of frames its peer has sent is a protocol violation. A lower value is allowed while frames are in flight or waiting to be processed; persistent lag is handled by the service's slow consumer limits.
- Heartbeat acknowledgements never request a replay. After a broken connection, start a new session and request a fresh snapshot.

Session schema (ID 33)

TEMPLATE	MESSAGE	DESCRIPTION
1	Negotiate	Opens and authenticates a session.
2	NegotiationResponse	Confirms a successful negotiation.
3	NegotiationReject	Rejects a negotiation.
4	AdminHeartbeat	Keepalive from Kraken.
5	SubscriberHeartbeat	Keepalive from the client.
6	Terminate	Ends the session.
7	SubscriptionRequest	Requests a snapshot, or starts or stops incremental delivery.
8	SubscriptionResponse	Answers a SubscriptionRequest.

Enumerations and encodings: [Session enums & encodings](#).

Market data schema (ID 34)

TEMPLATE	MESSAGE	DESCRIPTION
1	InstrumentDefinitionSnapshot	Instrument definitions, one per message, within a snapshot cycle.

TEMPLATE	MESSAGE	DESCRIPTION
2	OrderBookIncremental	Order-by-order (L3) book updates.
3	TradeIncremental	Trades.
4	OrderBookSnapshot	Full L3 book snapshot, in chunks.
5	InstrumentStatusIncremental	Live instrument trading-status changes.
6	TradingSessionStatusIncremental	Trading session status changes.
7	InstrumentDefinitionIncremental	Instrument additions, modifications and deletions.
10	TradingSessionStatusSnapshot	Current trading session status.

Enumerations and encodings: [Market data enums & encodings](#).

Sequencing

Incremental feeds carry contiguous per-instrument (or per-feed) sequence numbers starting at one — `orderBookSeq`, `tradeSeq`, `instrumentStatusSeq`, `instrumentDefinitionSeq` and `tradingSessionSeq`. Snapshots carry the last sequence number they include (for example `lastOrderBookSeqProcessed`), so you can apply buffered incrementals with a higher sequence number on top of the snapshot.

Gap recovery

There is no message replay. When you detect a gap, request a new snapshot on the snapshot connection and rebuild from it:

FEED	GAP RECOVERED BY
<code>OrderBookL3</code>	<code>OrderBookSnapshot</code>
<code>InstrumentDefinition</code>	<code>InstrumentDefinitionSnapshot</code>
<code>InstrumentStatus</code>	<code>InstrumentDefinitionSnapshot</code> , which carries each instrument's current status and its <code>lastInstrumentStatusSeqProcessed</code>
<code>TradingSessionStatus</code>	<code>TradingSessionStatusSnapshot</code>
<code>Trade</code>	Not recoverable

There is no trade snapshot. A gap in `tradeSeq` can be detected, but trades missed during the gap, or while you were disconnected, cannot be recovered from a book snapshot or any other feed.

Connection roles and startup

Market data uses two connections with enforced roles. A full snapshot is large and chunked; on a shared connection it would block the live updates behind it. The split also mirrors the UDP multicast channels to come. Each connection negotiates separately.

REQUEST	4500 · LIVE (INCREMENTAL) FEED	4501 · SNAPSHOTS
<code>Subscribe</code>	Accepted · incremental only, no initial snapshot	<code>UnsupportedFeed</code>
<code>Unsubscribe</code>	Accepted	<code>UnsupportedFeed</code>
<code>Snapshot</code>	<code>UnsupportedFeed</code>	Accepted · one cycle

`InvalidRequest` means the request type and feed combination is invalid. `UnsupportedFeed` means the combination is valid but is not offered on that endpoint or service version. `SpecificInstruments` requires a nonempty instruments group; `AllSecurities` requires an empty group. Unsubscribe uses `subscriptionId` and ignores scope fields.

Order book startup and recovery

- 01 Subscribe to `OrderBookL3` on 4500 and start buffering incremental updates.
- 02 Request a snapshot on 4501.
- 03 Receive every chunk. The snapshot carries `lastOrderBookSeqProcessed`.
- 04 Discard buffered updates with `orderBookSeq` $\leq$ `lastOrderBookSeqProcessed`.
- 05 Apply the rest in `orderBookSeq` order on top of the snapshot.
- 06 Book is live. The snapshot connection sits idle until needed.

Steady state uses 4500 only. On a gap or a stale book, go back to step 2.

Buffer first. Always subscribe on 4500 and start buffering before you request the snapshot. Requesting the snapshot first leaves a gap you can't recover.

Session schema messages (ID 33)

Negotiate

SBE

schema 33 · session

template 1

Opens and authenticates a session. `requestTimestampNs` is the authentication nonce and replay check.

Direction: Client → Kraken

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	<code>sessionId</code>	<code>SessionId</code>	field	required	
2	<code>requestTimestampNs</code>	<code>TimestampNs</code>	field	required	Client-generated UTC timestamp used as the ACG authentication nonce and replay check
3	<code>keepaliveIntervalMs</code>	<code>uint32</code>	field	required	Keepalive interval in milliseconds. Kraken returns the agreed interval in <code>NegotiationResponse.keepaliveIntervalMs</code>
4	<code>credentialType</code>	<code>CredentialType</code>	field	required	Signing scheme. The only value is <code>KrakenHmacSha512</code> (0)
5	<code>compId</code>	<code>VarStringEncoding</code>	var data	required	Registered logical client identity; 1 to 128 printable ASCII bytes
6	<code>accessKey</code>	<code>VarStringEncoding</code>	var data	required	Kraken API key. Its API secret signs the message
7	<code>signature</code>	<code>VarBufferEncoding</code>	var data	required	Raw 64-byte HMAC-SHA512 output, not base64 encoded. See Authentication

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

Authentication

`KrakenHmacSha512` authenticates `Negotiate` with the API secret that belongs to `accessKey` , using the same HMAC-SHA512 model as the Kraken FIX API.

Signed payload

The signed payload is the contiguous encoded SBE message from the first byte of the 8-byte `messageHeader` through the last byte of `accessKey` . It includes the encoded `compId` and `accessKey` length prefixes and values. The SOFH, the Kraken TCP header and `signature` (including its length prefix) are not part of the payload.

Read `blockLength` from the message header you are encoding, not from a constant in your generated codec: new fixed fields extend `blockLength` and are therefore signed.

```

payloadLength = 8 + messageHeader.blockLength
               + 2 + compIdLength
               + 2 + accessKeyLength
payload       = sbeMessage[0:payloadLength]
data          = SHA256(payload)
signature     = HMAC-SHA512(base64_decode(apiSecret), data)

```

The two-byte terms are the `uint16` length prefixes of `compId` and `accessKey`. Because the SBE header, session ID, timestamp, keepalive interval, credential type, `compId` and `accessKey` are all signed, none of them can be changed without invalidating the signature.

`signature` must remain the last variable-length field of the message. Put the raw 64 bytes of the HMAC output in `signature`; do not base64 encode them.

Fields

- `requestTimestampNs` is a UTC timestamp in nanoseconds that you generate for each attempt. It is the authentication nonce and replay check. Kraken rejects values outside its allowed clock window (see `NegotiationReject`); there is no separate nonce field.
- `sessionId` identifies one negotiation attempt. Generate a new `sessionId` for every attempt, including when you reconnect.
- `compId` is your registered logical client identity. It is case-sensitive, 1 to 128 printable ASCII bytes, and must be authorized for the account that owns `accessKey`. Use the same `compId` every time you reconnect. An account can have more than one registered `compId`.
- `accessKey` is the Kraken API key whose secret signs the message.

The service endpoint identifies the target, so there is no FIX-style `TargetCompID`. Credentials are provided during onboarding.

NegotiationResponse

SBE

schema 33 · session

template 2

Confirms a successful negotiation, with the agreed keepalive interval and the session expiry time.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	<code>sessionId</code>	<code>SessionId</code>	field	required	
2	<code>requestTimestampNs</code>	<code>TimestampNs</code>	field	required	
3	<code>serverTimestampNs</code>	<code>TimestampNs</code>	field	required	
4	<code>sessionExpiryTimestampNs</code>	<code>TimestampNs</code>	field	required	
5	<code>keepaliveIntervalMs</code>	<code>uint32</code>	field	required	

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

NegotiationReject

SBE

schema 33 · session

template 3

Rejects a negotiation. `code` gives the reason and `reason` the detail.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	<code>sessionId</code>	<code>SessionId</code>	field	required	

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
2	requestTimestampNs	TimestampNs	field	required	
3	serverTimestampNs	TimestampNs	field	required	
4	code	NegotiationRejectCode	field	required	
5	reason	VarStringEncoding	var data	required	

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

AdminHeartbeat

SBE

schema 33 · session

template 4

Keepalive from Kraken. Reports the latest valid client frame accepted on this connection.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	sessionId	SessionId	field	required	
2	serverTimestampNs	TimestampNs	field	required	
3	lastReceivedClientSequence	SequenceNumber	field	required	Latest valid client frame accepted on this connection

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

SubscriberHeartbeat

SBE

schema 33 · session

template 5

Keepalive from the client. Reports the latest server frame processed on this connection.

Direction: Client → Kraken

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	sessionId	SessionId	field	required	
2	clientTimestampNs	TimestampNs	field	required	
3	lastReceivedServerSequence	SequenceNumber	field	required	Latest server frame processed on this connection

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

Terminate

SBE

schema 33 · session

template 6

Ends the session. `code` gives the reason and `reason` the detail.

Direction: Either direction

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	sessionId	SessionId	field	required	
2	timestampNs	TimestampNs	field	required	

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
3	code	TerminationCode	field	required	
4	reason	VarStringEncoding	var data	required	

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

SubscriptionRequest

SBE schema 33 · session **template 7**

Requests a snapshot on the snapshot connection, or starts or stops incremental delivery on the live connection, for one market data feed.

Direction: Client → Kraken

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	sessionId	SessionId	field	required	
2	subscriptionId	uint64	field	required	
3	requestTimestampNs	TimestampNs	field	required	
4	subscriptionRequestType	SubscriptionRequestType	field	required	Snapshot is valid on the snapshot connection; Subscribe and Unsubscribe are valid on the live connection
5	marketDataFeed	MarketDataFeed	field	required	
7	instrumentListRequestType	InstrumentListRequestType	field	required	
6	instruments	group	group	required	
1	↳ instrumentId	InstrumentId	group field	required	

Rows marked ↳ are fields of the repeating group above them. Each group is preceded by a `groupSizeEncoding` header (`blockLength` , `numInGroup`).

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

SubscriptionResponse

SBE schema 33 · session **template 8**

Answers a `SubscriptionRequest`. `subscriptionResult` says whether it was accepted.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	sessionId	SessionId	field	required	
2	subscriptionId	uint64	field	required	
3	serverTimestampNs	TimestampNs	field	required	
4	subscriptionResult	SubscriptionResult	field	required	
5	reason	VarStringEncoding	var data	required	

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

Session enums & encodings

[SBE](#)[schema 33 · session](#)[types](#)

Enumerations

CredentialType

Encoding: `uint8`

NAME	VALUE	DESCRIPTION
KrakenHmacSha512	0	

NegotiationRejectCode

Encoding: `uint8`

NAME	VALUE	DESCRIPTION
InvalidCredentials	0	
NotAuthorized	1	
InvalidTimestamp	2	
AlreadyNegotiated	3	
UnsupportedVersion	4	
TemporaryFailure	5	

TerminationCode

Encoding: `uint8`

NAME	VALUE	DESCRIPTION
Finished	0	
NotAuthorized	1	
KeepaliveLapsed	2	
SlowConsumer	3	
ProtocolViolation	4	
ServerShutdown	5	

SubscriptionRequestType

Encoding: `uint8`

NAME	VALUE	DESCRIPTION
Snapshot	0	Request one snapshot on the snapshot connection
Subscribe	1	Start incremental delivery on the live connection
Unsubscribe	2	Stop an active subscription on the live connection

InstrumentListRequestType

Encoding: `uint8`

NAME	VALUE	DESCRIPTION
SpecificInstruments	0	
AllSecurities	1	

MarketDataFeed

Encoding: uint8

NAME	VALUE	DESCRIPTION
InstrumentDefinition	0	
OrderBookL3	1	
Trade	2	
InstrumentStatus	4	Live instrument-status changes. Initial status is included in InstrumentDefinitionSnapshot
TradingSessionStatus	5	

SubscriptionResult

Encoding: uint8

NAME	VALUE	DESCRIPTION
Accepted	0	
InvalidRequest	1	
UnsupportedFeed	2	
UnknownInstrument	3	
AlreadySubscribed	4	
NotAuthorized	5	
TemporaryFailure	6	

Primitive and composite encodings

messageHeader

Composite of: blockLength:uint16 , templateId:uint16 , schemaId:uint16 , version:uint16 .

SessionId

Composite of: low:uint64 , high:uint64 .

groupSizeEncoding

Composite of: blockLength:uint16 , numInGroup:uint16 .

InstrumentId

Primitive uint64 , length 1, required.

Opaque, globally unique instrument identifier assigned by Instrument Registry.

SequenceNumber

Primitive uint64 , length 1, required.

Connection-local frame count, with zero meaning no frame received.

TimestampNs

Primitive int64 , length 1, required.

Positive UTC timestamp in nanoseconds since the Unix epoch.

VarStringEncoding

Composite of: length:uint16 , varData:char .

Variable-length UTF-8 string. length is the number of bytes in varData , up to 65534. Used by compId , accessKey and the reason fields.

VarBufferEncoding

Composite of: length:uint32 , varData:uint8 .

Variable-length byte buffer. `length` is the number of bytes in `varData` , up to 2147483647. Used by the `Negotiate` signature .

Market data schema messages (ID 34)

InstrumentDefinitionSnapshot

SBE

schema 34 · market data

template 1

One instrument definition within a snapshot cycle, including the instrument's trading status.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	<code>instrumentId</code>	<code>InstrumentId</code>	field	required	
2	<code>lastInstrumentDefinitionSeqProcessed</code>	<code>uint64</code>	field	required	Last <code>InstrumentDefinitionIncremental</code> <code>instrumentDefinitionSeq</code> included, or zero if no update has been published
3	<code>lastInstrumentStatusSeqProcessed</code>	<code>uint64</code>	field	required	Last <code>InstrumentStatusIncremental</code> <code>instrumentStatusSeq</code> reflected in <code>instrumentStatus</code> for this instrument, or zero if no update has been published
4	<code>instrumentNumber</code>	<code>uint32</code>	field	required	One-based position of this instrument in the serialized snapshot response
5	<code>totalInstruments</code>	<code>uint32</code>	field	required	Total number of instruments in this snapshot cycle
6	<code>priceScale</code>	<code>Scale</code>	field	required	
7	<code>qtyScale</code>	<code>Scale</code>	field	required	
8	<code>tickSize</code>	<code>Price</code>	field	required	Minimum valid price increment
9	<code>qtyIncrement</code>	<code>Quantity</code>	field	required	Smallest valid quantity step
10	<code>maxTradeVol</code>	<code>Quantity</code>	field	required	Largest quantity accepted for one order, or zero when no public limit is configured
11	<code>contractMultiplier</code>	<code>Decimal</code>	field	required	Amount of underlying represented by one contract, or one for non-contract instruments
12	<code>productType</code>	<code>ProductType</code>	field	required	
13	<code>marketSector</code>	<code>MarketSector</code>	field	required	Sector of the instrument's underlying exposure

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1 4	maturityType	MaturityType	field	optional	Maturity schedule for futures and options; null for products without a maturity
1 5	instrumentStatus	InstrumentTradingStatus	field	required	Trading status at lastInstrumentStatusSeq Processed
1 6	symbol	SymbolEncoding	var data	required	
1 7	currency	CurrencyEncoding	var data	required	Currency in which prices are expressed
1 8	underlyingSymbol	SymbolEncoding	var data	required	Client-facing symbol of the underlying asset, empty when not applicable
1 9	description	DescriptionEncoding	var data	required	Human-readable instrument description
2 0	instrumentStatusReason	StatusReasonEncoding	var data	required	Human-readable reason for instrumentStatus, empty when no reason was supplied

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

OrderBookIncremental

SBE

schema 34 · market data

template 2

Order-by-order (L3) book updates for one instrument.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	instrumentId	InstrumentId	field	required	
2	orderBookSeq	uint64	field	required	Contiguous per-instrument update sequence, starting at one
3	transactTime	TimestampNs	field	required	Time the matching event occurred at the source
4	orders	group	group	required	
1	↳ updateAction	OrderUpdateAction	group field	required	
2	↳ side	Side	group field	required	
3	↳ orderId	OrderId	group field	required	
4	↳ price	Price	group field	required	
5	↳ quantity	Quantity	group field	required	
6	↳ orderPriority	uint64	group field	required	Opaque queue priority comparable within one price level

Rows marked ↳ are fields of the repeating group above them. Each group is preceded by a `groupSizeEncoding` header (`blockLength` , `numInGroup`).

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

TradeIncremental

SBE

schema 34 · market data

template 3

One trade on an instrument.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	<code>instrumentId</code>	<code>InstrumentId</code>	field	required	
2	<code>transactTime</code>	<code>TimestampNs</code>	field	required	Time the trade occurred at the source
4	<code>tradeSeq</code>	<code>uint64</code>	field	required	Contiguous per-instrument trade sequence, starting at one
5	<code>tradeId</code>	<code>TradeId</code>	field	required	
6	<code>orderId</code>	<code>OrderId</code>	field	required	Aggressor order ID for this trade
7	<code>price</code>	<code>Price</code>	field	required	
8	<code>quantity</code>	<code>Quantity</code>	field	required	
9	<code>aggressorSide</code>	<code>AggressorSide</code>	field	optional	
10	<code>ordType</code>	<code>OrdTypeMD</code>	field	optional	

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

OrderBookSnapshot

SBE

schema 34 · market data

template 4

One chunk of a full order-by-order (L3) book snapshot for an instrument.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
2	<code>instrumentId</code>	<code>InstrumentId</code>	field	required	
3	<code>lastOrderBookSeqProcessed</code>	<code>uint64</code>	field	required	Last <code>OrderBookIncremental</code> <code>orderBookSeq</code> included, or zero if no update has been published
4	<code>chunkNumber</code>	<code>uint32</code>	field	required	One-based chunk number within this snapshot response
5	<code>totalChunks</code>	<code>uint32</code>	field	required	
6	<code>orders</code>	group	group	required	
1	↳ <code>side</code>	<code>Side</code>	group field	required	
2	↳ <code>orderId</code>	<code>OrderId</code>	group field	required	
3	↳ <code>price</code>	<code>Price</code>	group field	required	
4	↳ <code>quantity</code>	<code>Quantity</code>	group field	required	
5	↳ <code>orderPriority</code>	<code>uint64</code>	group field	required	

Rows marked ↳ are fields of the repeating group above them. Each group is preceded by a `groupSizeEncoding` header (`blockLength` , `numInGroup`).

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

InstrumentStatusIncremental

SBE

schema 34 · market data

template 5

A live change to an instrument's trading status.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	<code>instrumentId</code>	<code>InstrumentId</code>	field	required	
2	<code>instrumentStatus</code>	<code>InstrumentTradingStatus</code>	field	required	
3	<code>instrumentStatusSeq</code>	<code>uint64</code>	field	required	Contiguous per-instrument status sequence, starting at one
4	<code>reason</code>	<code>StatusReasonEncoding</code>	var data	required	Human-readable reason for this status update, empty when no reason was supplied

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

TradingSessionStatusIncremental

SBE

schema 34 · market data

template 6

A change to the trading session status.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	<code>tradingSessionStatus</code>	<code>TradingSessionStatusCode</code>	field	required	
2	<code>tradingSessionSeq</code>	<code>uint64</code>	field	required	Contiguous trading-session status sequence, starting at one

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

InstrumentDefinitionIncremental

SBE

schema 34 · market data

template 7

Instrument additions, modifications and deletions.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
2	<code>instrumentDefinitionSeq</code>	<code>uint64</code>	field	required	Contiguous instrument-definition feed sequence, starting at one
1	<code>instruments</code>	group	group	required	
1	↳ <code>instrumentId</code>	<code>InstrumentId</code>	group field	required	
2	↳ <code>priceScale</code>	<code>Scale</code>	group field	required	
3	↳ <code>qtyScale</code>	<code>Scale</code>	group field	required	
4	↳ <code>tickSize</code>	<code>Price</code>	group field	required	Minimum valid price increment

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
5	↳ qtyIncrement	Quantity	group field	required	Smallest valid quantity step
6	↳ maxTradeVol	Quantity	group field	required	Largest quantity accepted for one order, or zero when no public limit is configured
7	↳ contractMultiplier	Decimal	group field	required	Amount of underlying represented by one contract, or one for non-contract instruments
1 2	↳ instrumentUpdateAction	InstrumentUpdateAction	group field	required	
1 3	↳ productType	ProductType	group field	required	
1 4	↳ marketSector	MarketSector	group field	required	
1 5	↳ maturityType	MaturityType	group field	optional	
8	↳ symbol	SymbolEncoding	group data	required	
9	↳ currency	CurrencyEncoding	group data	required	Currency in which prices are expressed
1 0	↳ underlyingSymbol	SymbolEncoding	group data	required	Client-facing symbol of the underlying asset, empty when not applicable
1 1	↳ description	DescriptionEncoding	group data	required	Human-readable instrument description

Rows marked ↳ are fields of the repeating group above them. Each group is preceded by a `groupSizeEncoding` header (`blockLength` , `numInGroup`).

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

TradingSessionStatusSnapshot

SBE schema 34 · market data **template 10**

The current trading session status.

Direction: Kraken → client

ID	FIELD	TYPE	ENCODING	PRESENCE	DESCRIPTION
1	lastTradingSessionSeqProcessed	uint64	field	required	Last TradingSessionStatusIncremental tradingSessionSeq reflected in tradingSessionStatus, or zero if no update has been published
2	tradingSessionStatus	TradingSessionStatusCode	field	required	Trading session status at lastTradingSessionSeqProcessed

Every message is preceded by the standard SBE `messageHeader` (`blockLength` , `templateId` , `schemaId` , `version`). See [encodings](#).

Market data enums & encodings

SBE schema 34 · market data **types**

Enumerations

OrderUpdateAction

Encoding: uint8

NAME	VALUE	DESCRIPTION
New	1	
Change	2	
Delete	3	

InstrumentUpdateAction

Encoding: uint8

NAME	VALUE	DESCRIPTION
Add	1	
Modify	2	
Delete	3	

ProductType

Encoding: uint8

NAME	VALUE	DESCRIPTION
Spot	1	
Future	2	
Option	3	

MarketSector

Encoding: uint8

NAME	VALUE	DESCRIPTION
Crypto	1	
Equity	2	
Fx	3	
TokenizedEquity	4	
Index	5	

MaturityType

Encoding: uint8

NAME	VALUE	DESCRIPTION
Monthly	1	
Quarterly	2	
Irregular	3	
Perpetual	4	
Semiannual	5	

Side

Encoding: uint8

NAME	VALUE	DESCRIPTION
Buy	1	
Sell	2	

AggressorSide

Encoding: uint8

NAME	VALUE	DESCRIPTION
Buy	1	
Sell	2	

InstrumentTradingStatus

Encoding: uint8

NAME	VALUE	DESCRIPTION
Hidden	1	
Online	2	
Maintenance	3	
CancelOnly	4	
PostOnly	5	
LimitOnly	6	
Delisted	7	
ReduceOnly	8	
SystemMaintenance	9	
Halted	10	
Closed	11	Instrument is outside its scheduled trading session

TradingSessionStatusCode

Encoding: uint8

NAME	VALUE	DESCRIPTION
Halted	1	
Open	2	
CancelOnly	101	
PostOnly	102	
LimitOnly	103	

OrdTypeMD

Encoding: char

NAME	VALUE	DESCRIPTION
Market	1	Taker order was a market order [Kraken emits: Yes]
Limit	2	Taker order was a limit order [Kraken emits: Yes]

Primitive and composite encodings

messageHeader

Composite of: `blockLength:uint16`, `templateId:uint16`, `schemaId:uint16`, `version:uint16`.

groupSizeEncoding

Composite of: `blockLength:uint16`, `numInGroup:uint16`.

InstrumentId

Primitive `uint64`, length 1, required.

Opaque, globally unique instrument identifier assigned by Instrument Registry.

OrderId

Composite of: `low:uint64`, `high:uint64`.

TradId

Composite of: `low:uint64`, `high:uint64`.

TimestampNs

Primitive `int64`, length 1, required.

Positive UTC timestamp in nanoseconds since the Unix epoch.

Scale

Primitive `uint8`, length 1, required. Maximum value 36.

Number of supported decimal places.

Decimal

Composite of: `mantissa:int64`, `exponent:int8`.

`exponent` ranges from -36 to 0.

Price

Composite of: `decimal:ref`.

Quantity

Composite of: `decimal:ref`.

SymbolEncoding

Composite of: `length:uint16`, `varData:char`.

`length` is the number of UTF-8 encoded bytes, up to 128. Used by `symbol` and `underlyingSymbol`.

CurrencyEncoding

Composite of: `length:uint8`, `varData:char`.

`length` is the number of UTF-8 encoded bytes, up to 12.

DescriptionEncoding

Composite of: `length:uint16`, `varData:char`.

`length` is the number of UTF-8 encoded bytes, up to 512.

StatusReasonEncoding

Composite of: `length:uint16`, `varData:char`.

`length` is the number of UTF-8 encoded bytes, up to 1024. An empty value means no reason was supplied.

Kraken constrains these values to the limits above before publishing and does not truncate identifiers.

A Source documents

This guide consolidates the following. Where they conflict, the latest dated document wins until reconciled.

DOCUMENT	VERSION	SECTION
Part A: migration overview	Carried over from v0.5 (12 Oct 2026), updated for MR 178 and the 19 Oct review	Sections 01–09
US Perps connectivity guide	v0.5 text; colocation per MR 178 <code>guides/us-perps/connectivity.mdx</code>	10, 11
Unified FIX v2 (order entry)	MR 178 <code>api-reference/unified-fix-v2</code>	12
Drop Copy	MR 178 <code>api-reference/drop-copy</code>	13
SBE market data	MR 178 <code>api-reference/sbe</code>	14
Equity Perps - Single-Stock Perp (Confluence)	v72 · 7 Oct 2026	04

The Kraken Colocation High Level Network Design is shared under NDA and is no longer reproduced in this guide.

B Glossary

Matching engine (ME)

The system that matches buy and sell orders.

UAT

User acceptance testing. The pre-production environment for building and certifying.

FCM

Futures commission merchant. Your clearing firm.

Drop copy

A read-only copy of your execution reports for reconciliation.

SBE

Simple Binary Encoding. A compact, fixed-layout binary format for market data.

L3 book

An order-by-order view of the book, not aggregated by price.

Sequence numbers

Contiguous counters on market data, starting at 1: `orderBookSeq`, `tradeSeq` and `instrumentStatusSeq` per instrument; `instrumentDefinitionSeq` and `tradingSessionSeq` per feed. They replace `RptSeq`.

Instrument Registry ID

The opaque uint64 `InstrumentId` used across SBE instrument definitions, subscriptions, books, trades and status. Assigned by Instrument Registry; it does not encode asset class or asset ID.

CompID

FIX identifiers for the sender and target of a session.

ClOrdID

The client-assigned order identifier. On US Perps it must be a positive integer of at most 19 digits, strictly increasing within the session.

LOA

Letter of Authorization. Lets your carrier order a cross-connect to Kraken.

Meet-me room (MMR)

The data-centre room where cross-connects between tenants terminate.

BGP

Border Gateway Protocol. Exchanges routes and enables failover across links.

MACsec

Link-layer encryption on a physical cross-connect.