Measuring the latency impact of an order entry and matching engine migration: evidence from Deribit's Starbase

Deribit by Coinbase
Coinbase

Abstract

We quantify the performance impact of Deribit's rollout of Starbase order entry alongside Deribit's Classic API (WebSocket, FIX, or REST). We compare a six-day pre window (5–10 August 2026) with a six-day post window (21–26 August 2026) and analyze load behavior during a high-volume two-day period (24–25 August 2026; 1.02 billion matching-engine events). In the post window, more than 90% of order entry messages are routed through Starbase. Message volume is dominated by a small minority of latency-sensitive early-adopting participants. Additionally, average daily volume is about four times higher in the post window ($1.24B versus $4.93B), so the improvement is not an artifact of a quieter post period.

The median request–response latency decreases from 4.8 ms to 76 µs, with larger improvements in the tail (p_{90}: 26.8 ms to 106 µs; p_{99}: 113 ms to 234 µs). Because Classic API timestamps are taken inside the Classic application stack whereas the Starbase measurements are wire-to-wire (Corvil), these improvements are conservative: Classic API latency is a lower bound on the corresponding wire-to-wire improvement.

Under load, the median remains largely flat while congestion appears primarily as an increase in tail latencies in the post Starbase period. Throughout August 24–25, 2026, the median matching-engine event rate is 4,404 events s^{-1} and the p_{99} rate is 23,383 events s^{-1}. A microburst case study reaches a maximum round trip latency just below 0.8 ms, and latency returns to baseline levels within 2 ms.

Differences within Starbase in the order configuration and feed type are examined. Orders using the market-maker protection (MMP) pre-margin mechanism are faster in the median by 16 µs to 24 µs, consistent with the removal of the risk processing per-request from the critical path. Public market data updates typically reach the wire slightly before private confirmations with the exception of mass quotes with many legs.

Overall, Starbase provides substantially lower and more predictable exchange-side latency while maintaining the message rates required for a fair and orderly market.

1 Introduction

1.1 Background and motivation

The Deribit Classic API is designed for a broader group of users and for Internet access and remains the recommended integration path for most participants. Those design choices carry a latency cost: participant-specific risk and session processing introduce queueing that grows with load. As activity grew from hundreds of thousands to hundreds of millions of orders per day, that cost became more visible, and incremental changes largely held latency flat rather than reducing it.

This coupling between activity and latency increases the risk of adverse-selection for market makers and can widen spreads. It also encourages latency-sensitive participants to choose configurations that minimize round-trip time, even when doing so limits feature usage. An example is a lower adoption of cross-collateral (X:SM and X:PM) due to latency concerns. Listing new products at scale would exacerbate these concerns.

1.2 Deribit Starbase

Starbase replaces Deribit's Classic matching engine with a deterministic, low-latency system. Additionally, Starbase complements the Classic API with an order entry and market data API optimized for low, predictable latency at sustained throughput. It prioritizes a wide range of features at the expense of speed and throughput. The Classic API remains fully supported and is the appropriate choice for most participants. A key design change is that risk processing is restructured to reduce per-request work on the critical path [1].

This paper studies the days surrounding the Starbase API rollout and quantifies the resulting before/after performance changes.

2 Data and Methods

2.1 Physical infrastructure and connectivity

Deribit is deployed on dedicated physical hardware in Equinix LD4 (Slough, UK). The Starbase stack runs on its own machine stack, separate from the Classic stack, so the pre/post comparison in section 3.1 holds network delays roughly fixed and differs only in the machines and software that requests traverse. Participant cross connects are equalized to a common fiber length within LD4, though not across the wider Slough campus. Hosted colocation is equalized in the same manner and shares all remaining network infrastructure with a cross connected client [10]. Starbase is additionally reachable from AWS `eu-west-2` (London) and `ap-northeast-1` (Tokyo), but both cloud regions experience higher latency than the direct

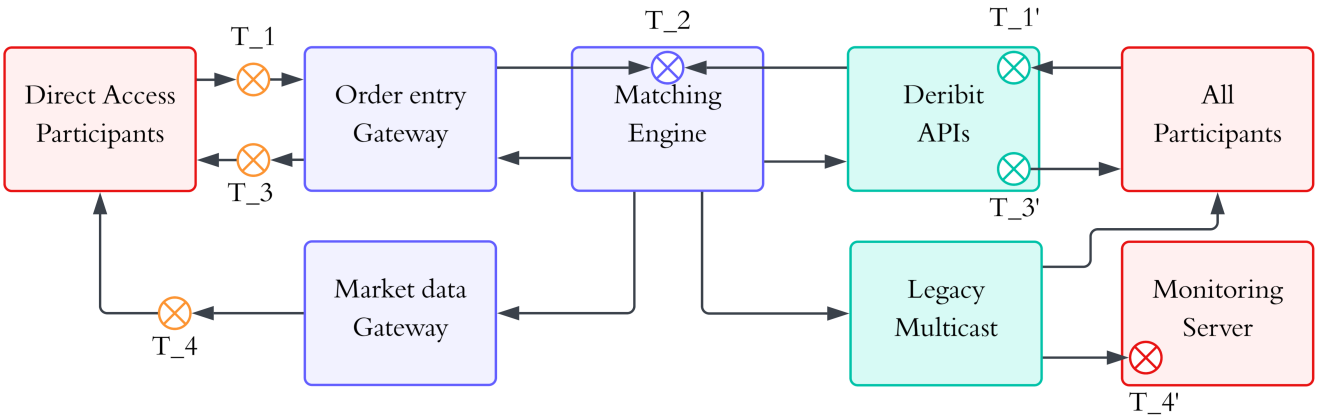

Figure 1: Timestamp measurement points used throughout this study. See table 1 for definitions of T_1–T_4 and their software-timestamp counterparts.

Table 1: Timestamp definitions and locations corresponding to figure 1.

Symbol	Resolution	Definition / location
T_1	ns	Corvil timestamp of the inbound request on the Starbase order entry path, taken right before the Starbase order entry gateway.
T_1'	μs	Classic API software timestamp of the inbound request, taken inside the Classic Deribit order entry application stack.
T_2	ns	Starbase matching engine timestamp stamped at order-book entry time.
T_3	ns	Corvil timestamp of the outbound response on the Starbase order entry path, taken on the link after the Starbase order entry gateway.
T_3'	μs	Classic API software timestamp of the outbound response, taken inside the Classic Deribit order entry application stack.
T_4	ns	Corvil timestamp of outbound multicast L3 market data on the Starbase market data path, taken after the market data gateway.
T_4'	ms	Monitoring-host software timestamp of outbound Classic multicast L2 market data taken by a colocated client machine.

Clocking. T_1, T_3, and T_4 are produced by the same Corvil aggregation tap and share a common clock. Time is synchronized across the Starbase infrastructure using PTP. Primed timestamps (T_1', T_3', T_4') originate from separate systems and are not expected to be directly comparable at nanosecond precision [14, 15]. T_4' is unused in this analysis, but it is expected that $T_4 < T_4'$ for all events.

LD4 connection methods. Because T_1 and T_3 are taken before the switch leading to the gateway (table 1), the access path to the client falls outside the measurement window: the latencies in section 3 are only exchange-side.

2.2 System architecture

After the rollout, the exchange operated both the Classic API stack and the new Starbase order entry and market data gateways in parallel.

Figure 1 indicates the measurement points. In both paths, client requests pass through exchange-side controls before reaching the matching engine. Accepted requests for both paths are serialized onto a single sequencing stream that delivers a total order of events to the Starbase matching engine.

We use the timestamp definitions in table 1 throughout. Unprimed timestamps (T_1–T_4) are Corvil/Starbase observations on the Starbase paths; primed timestamps (T_1', T_3', T_4') originate from separate Classic systems or a monitoring host. All clocks are synchronized with PTP [15].

2.3 Wire timestamps and Corvil measurement

All wire timestamps in this study are produced by Corvil, a passive network capture and analytics appliance [14]. Optical taps on the links carrying Starbase order entry and market data mirror every packet to an aggregation appliance, which timestamps each capture in hardware with nanosecond resolution. Because the taps are passive and out of band, the measurement path is fully decoupled from the production path: capture adds no latency to, and cannot exert backpressure on, the traffic it observes. This is the distinction that makes T_1, T_3 and T_4 wire-to-wire measures rather than application-level ones.

T_1, T_3 and T_4 are captured by the same aggregation appliance and therefore share a single clock. No clock offset enters the differences defined in section 2.6.4, so the only measurement error is the appliance's nanosecond-scale capture precision, which is negligible against the latencies reported in section 3.

Both taps sit inside the exchange: T_1 before the order entry gateway and T_3 on the link after it (Table 1). Nothing between the participant and those taps enters the latencies we report. The primed Classic timestamps come from separate hosts and do not share this clock, which is the basis for the comparability caveats in section 2.7.

2.4 Data sources

Classic API request/response log (CLASSIC_API_LOG) Classic order entry request/response log (request-granular) providing T_1' and T_3'.

Matching engine event log (ME_LOG) Starbase matching-engine event log providing T_2 (event-granular: mass actions expand into per-order/match events).

Corvil passive-tap captures (CORVIL_LOG) Packet captures and nanosecond timestamping from Corvil, used to reconstruct message timelines on the wire (request-granular for request/responses, event-granular for L3 market data) and providing T_1, T_3, and T_4 [14].

2.5 Study design and periods

Starbase order entry go-live occurred on 11 August 2026 at 09:00 UTC [1].

2.5.1 Pre window (Classic order entry)

5–10 August 2026 (six trading days), ending 9 hours before go-live. During this period, matching already ran on the Starbase matching engine, but order entry only traversed the Classic gateways. Average daily volume over the window is $1.24B.

2.5.2 Post window (Starbase order entry)

21–26 August 2026 (six trading days), starting roughly ten days after go-live, to allow for large scale adoption. More than 90% of the order entry messages are routed through the Starbase order entry gateways in this period. Average daily volume is $4.93B, about four times the pre-window level.

We exclude the transition interval between these windows (including go-live day) to reduce contamination from mixed routing and exchange-side and client-side tuning during rollout.

2.6 Participant populations and inclusion/exclusion

The pre window includes all participants, ordering via the Classic API. In the post window, we only consider the Starbase API (even though the Classic API is still available) and message volume is dominated by a small minority of institutional, latency-sensitive market makers and HFT firms; as a result, even with such a high share of messages on Starbase, post Starbase sample over-represents early adopters in terms of participant composition.

2.7 Definitions and measurement

2.7.1 Message types

We analyze new, cancel, amend, mass quote, and mass cancel requests and their immediate gateway responses. Session administration traffic such as logons or heartbeats are excluded.

2.7.2 Pairing

Each request is linked with its private (response to the request originator). Each market data packet is linked to the request that led to its generation: in the case of trades, mass quotes or MMP triggers, this can be a one-to-many relationship.

2.7.3 Market data latency and comparability

Market data is generated independently from private responses. We timestamp outbound market data on the wire using T_4 (table 1). For public-versus-private comparisons, we match the private messages with corresponding public messages and compare their T_4 and T_3 timestamps.

When relating market data to order entry, we compare the private response to the earliest market data message:

$$\Delta t_{\text{resp}\to\text{md}} = T_4 - T_3. \tag{1}$$

2.7.4 Latency and throughput

Definition (Latency)**.** Post-window request–response latency (Starbase gateways) is $\Delta t_{\text{post}} = T_3 - T_1$. Pre-window request–response latency (Classic gateways) is $\Delta t_{\text{pre}} = T_3' - T_1'$. We report the median, p_{90} and p_{99}, in different buckets (hours or seconds).

Definition (Throughput)**.** Throughput is the number of order entry messages per second, aggregated into one-second buckets.

2.8 Limitations and potential biases

Pre and post windows use different measurement systems ($T_3'-T_1'$ vs. $T_3 - T_1$), so absolute comparisons mix network, hardware, clock and platform effects. In particular, Classic API latencies are measured inside the Classic application stack and omit hardware and network components that are included in the Corvil wire-to-wire measure; the Classic figures are therefore a lower bound on the corresponding wire-to-wire latency. The post window over-represents early adopters; therefore, we also report like-for-like results in section 3.1. Market data timing comparisons use T_3 and T_4 and are limited to the post window.

3 Results

3.1 Latency before and after Starbase

Across the comparison window, median latency (averaged across hourly medians in the pre and post window) decreases from 4.8 ms to 76 µs (about 60×). Tail improvements are larger: The average hourly p_{90} decreases from 26.8 ms to

106 µs (approximately 250×) and p_{99} from 113 ms to 234 µs (approximately 480×). Figure 2 shows the corresponding latency percentiles over time.

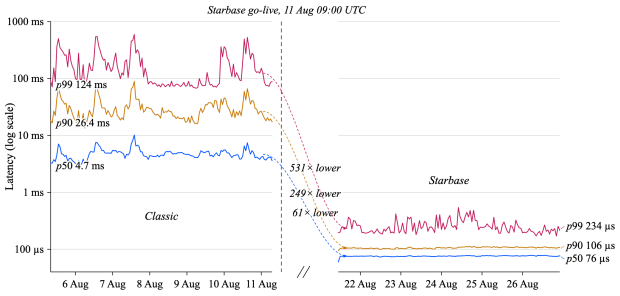

Figure 2: Hourly request–response latency percentiles in the pre and post windows.

The volatility of latency decreases substantially. Before go-live, average hourly p_{99} is about 24× the median and varies strongly intraday (roughly 80 ms to 600 ms). After go-live, it is about 3× the median and remains within a 180 µs to 420 µs band. The post window therefore has both lower latency and improved predictability.

Activity is not constant across the windows: average daily volume rises from $1.24B to $4.93B, roughly fourfold. Because Classic-stack latency degrades with load, a pre window at post-window activity levels would have been slower still, so the comparison is likely to understate the gain.

We repeat the comparison for the ten largest market makers by Classic-stack order count, restricted to participants that also trade on Starbase; together they account for 81.3% of pre-migration order flow from that group. Ranking on the highest Classic-stack activity selects the cohort on pre-window behaviour only, so the comparison does not depend on early Starbase adoption. For this cohort, the average hourly median latency decreases from 4.5 ms to 78 µs (about 60×), p_{90} from 24.5 ms to 107 µs (about 230×), and p_{99} from 92.5 ms to 226 µs (about 410×). The median improvement matches all order flow (about 60×), but the tail improvement is smaller (about 410× versus about 530× at p_{99}) likely because this cohort was already faster and more optimized than average on the Classic stack, its pre-migration p_{99} being 92.5 ms against 113 ms platform-wide. Figure 3 shows the corresponding intraday percentiles for this cohort. The advantage widens sharply under load: in the cohort's busiest Classic minute (292,801 messages), the median latency reaches 18.9 ms and p_{99} 716.7 ms, whereas the Starbase minute carrying the same message rate records 80.6 µs and 379.8 µs: reductions of about 235× and 1,900× (Table 2).

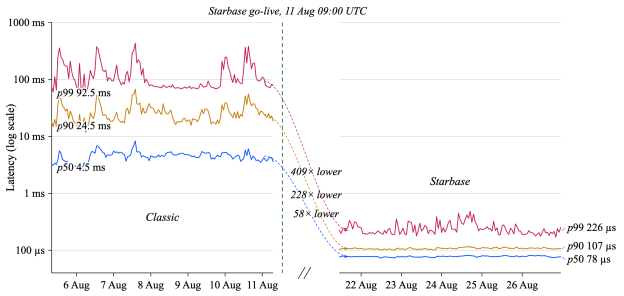

Figure 3: Request–response latency percentiles versus time of day in the pre and post windows for the ten largest market makers by Classic API order count.

Table 2: Matched-load latency for the same cohort. The Legacy minute is the *busiest* minute in the pre window; the Starbase minute is the post-window minute with the closest message count.

	Minute (UTC)	Messages	$p50$	$p90$	$p99$
Legacy	10 Aug 00:11	292,801	18.9 ms	130.5 ms	716.7 ms
Starbase	25 Aug 07:49	294,023	80.6 µs	120.6 µs	379.8 µs
Reduction			235×	1,082×	1,887×

3.2 Throughput and latency under load

To characterize gateway behavior under load, we analyze the two-day period 24–25 August 2026, during which the matching engine recorded 1.020 billion events. Aggregated into one-second buckets, the median matching-engine event log (ME_LOG) rate over the period is 4,404 events s^{-1} and the p_{99} ME_LOG rate is 23,383 events s^{-1}. Figure 4 plots the 50th, 90th, and 99th percentiles of request–response latency against ME_LOG rate. Each second contributes one observation (n = 172,800).

ME_LOG rate has a measurable but modest effect on the median, and a large one on the tail. Across bands of 0–5k, 5–10k, 10–20k and above 20k events s^{-1}, the median second's p_{50} rises from 73.6 µs to 76.5 µs, 78.5 µs and 80.9 µs—about 10% across the full range of load. Over the same bands p_{90} rises from 96.1 µs to 126.8 µs (+32%) and p_{99} from 125.5 µs to 604.6 µs, close to a fivefold increase and concentrated almost entirely in the highest band. Mean latency rises from 75.2 µs to 102.3 µs (+36%), tracking p_{90} rather than p_{50}: the distribution is right-skewed, so the mean absorbs the tail and is not a substitute for the median in describing the typical second.

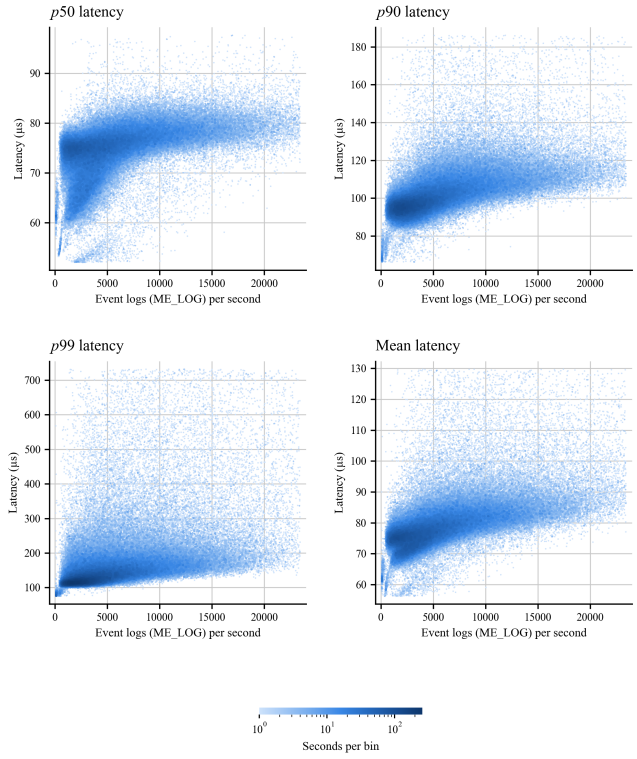

Table 3: Latency vs. matching-engine event log (ME_LOG) rate (one-second buckets).

ME_LOG s^{-1}	n	Latency (μs)			
		$p50$	$p90$	$p99$	Mean
0–5k	96,743	73.6	96.1	125.5	75.2
5–10k	48,599	76.5	104.4	154.9	80.1
10–20k	24,044	78.5	111.6	191.0	84.7
> 20k	3,414	80.9	126.8	604.6	102.3

Figure 4: Request–response latency percentiles vs. matching-engine event log (ME_LOG) rate in one-second buckets.

Notes: Each dot is one second (n = 172,800); events are rows in the matching-engine event log, with mass quotes counted per order. The graphs show the scatter plots, and the quantitative summary is in Table 3. For legibility each panel is drawn over the 0.1st–99th percentile of ME_LOG rate and up to median +5 × (p_{90} − median) of its own latency statistic, omitting 1.3%–3.2% of seconds depending on the panel. The table uses all seconds, untrimmed.

3.3 Case study: a microburst

Sections 3.1 and 3.2 summarize latency in aggregate. We next examine a single market event leading to congestion to describe microburst dynamics, including queue formation, the peak latency reached and the recovery time.

We select the busiest one-second interval in the post window outside the 08:00 UTC expiry: 25 August 2026 02:41:28 UTC, containing 23,836 order entry requests (messages). This request count is not directly comparable to the ME_LOG event rate in section 3.2. Arrivals within this second are highly non-uniform. Relative to a mean rate of 23.8 requests ms^{-1}, the densest millisecond contains 295 requests (≈ 12.4× the mean). A 4.5 ms interval beginning 353.0 ms into the second contains 691 requests.

Figure 5 shows this 4.5 ms interval. Each request is drawn as a horizontal bar ordered by arrival time: the left edge is the Corvil inbound timestamp (T_1) and the bar length is the request–response latency ($T_3 - T_1$). The left envelope traces arrival of requests and the right envelope traces dissemination of responses.

The burst produces a large, bounded, short-lived latency spike. Median latency across the congested period is 609 µs versus 79 µs for the rest of the second, and the longest single round trip is 798 µs (about 10× the baseline median). The distribution remains relatively tight: the window's 99th percentile is 775 µs, within 3% of its maximum. The latency spike is also short-lived: as figure 5 shows, the backlog drains and latency returns to its baseline within 2 ms of the burst.

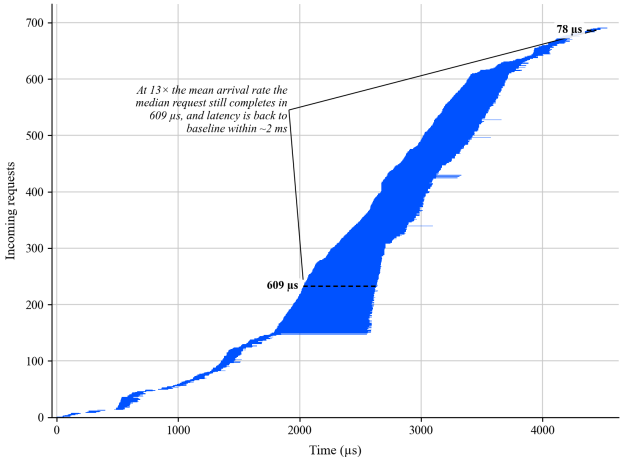

Figure 5: Microburst case study: per-request latency over the 4.5 ms interval starting 353.0 ms into 25 August 2026 02:41:28 UTC.

3.4 Public vs. private feeds

Corvil allows for a direct comparison between tapped public and private paths which are timestamped by the same clock. Figure 6 summarizes the relative latency between the two feeds.

We use this measurement to interpret public-versus-private timing via $\Delta t_{\text{resp}\to\text{md}}$ (section 2.7.3).

Figure 6 shows that $\Delta t_{\text{resp}\to\text{md}}$ is negative for 80.3% of 1,978,446,430 matched events: the public book update typically reaches the wire before the corresponding private response. The median is −2.47 µs (IQR −3.30 µs to −0.95 µs) and it is similar for new messages (−2.96 µs), cancel (−2.57 µs) and amend (−3.03 µs).

Mass quotes differ, and ordering depends on the quote size. Up to ten legs, the public messages arrive faster than private, with medians between −3.20 µs and −2.44 µs and the public path ahead in 88–94% of cases. For larger mass quotes the ordering reverses: at 11–20 legs the public update leads in only 40.2% of cases (median 1.08 µs), and by 21–40 legs it trails the private response in 89.7% of cases (median 4.79 µs).

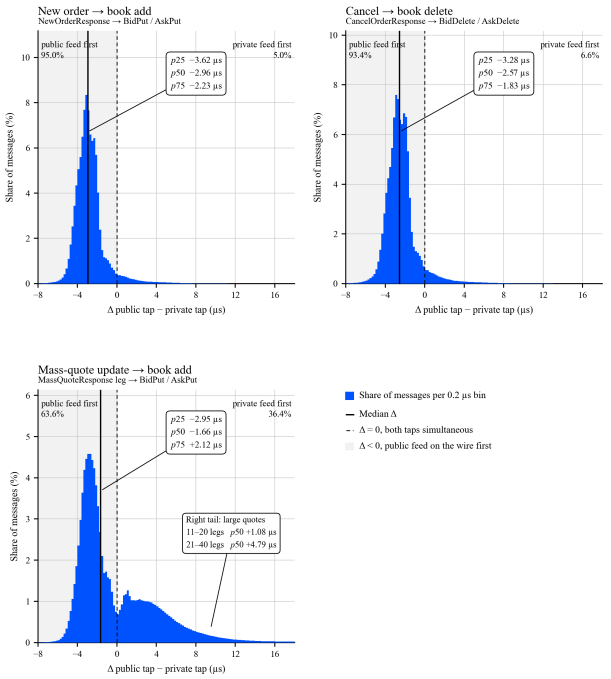

Figure 6: Public versus private market data timing at the T_4 observation point.

Notes: Table 4 reports $\Delta t_{\text{resp}\to\text{md}} = T_4 - T_3$ (µs) by matched message class for the post window (21–26 August 2026); negative values indicate the public update crossed its tap before the private response. Deribit publishes every book update twice, from redundant market data gateways A and B. Δ is taken to whichever arrived first. Mass-quote observations are per market data event: a mass quote with N updated legs contributes N rows. The mass quote graph is therefore weighted toward large quotes [11].

Table 4: $\Delta t_{\text{resp}\to\text{md}}$ by matched message class, post window.

		$\Delta t_{\text{resp}\to\text{md}}$ (µs)					
Message class	n	P1	P25	P50	P75	P99	% public first
New order	451,341,698	−5.49	−3.62	−2.96	−2.23	+4.66	95.0
Cancel	447,281,812	−5.44	−3.28	−2.57	−1.83	+5.47	93.4
Amend	175,349,307	−5.35	−3.61	−3.03	−2.37	+4.23	95.9
Mass quote	904,473,613	−5.09	−2.95	−1.66	+2.12	+22.20	63.6
All matched	1,978,446,430	−5.29	−3.30	−2.47	−0.95	+14.69	80.3

3.5 Latency effects of the MMP pre-margin system

Starbase supports MMP orders and quotes under a pre-margining model in which exposure is collateralized in advance rather than checked on the critical path. Ordinarily the check happens on arrival: before accepting an order, the engine evaluates the exposure it would add against the collateral in the portfolio, and that evaluation sits between the request and its response. Pre-margining instead bounds the worst case ahead of time. A single execution can fill at most twice a group's Max Quote Quantity, so an initial-margin hold sized to that bound is placed against the portfolio when the quotes are set, and an arriving order is already collateralized [7].

Because pre-margining removes per-request risk checks from the critical path [8], MMP orders are expected to have lower latency than non-MMP orders. Figure 7 shows this for new orders. The median is lower for MMP orders in every instrument class: 63.1 µs versus 87.0 µs for perpetual futures, 68.5 µs versus 88.9 µs for dated futures, and 72.1 µs versus 87.9 µs for options. This corresponds to a reduction of 16 µs to 24 µs, or about 18–28% of the non-MMP median (depending on instrument class).

Table 5: New-order counts and median latency with and without MMP pre-margining, by instrument class.

	New orders		Median latency (µs)		
	MMP	non-MMP	MMP	non-MMP	Δ
Perpetual futures	37,181,947	308,934,190	63.1	87.0	−23.9
Dated futures	84,700,908	72,402,250	68.5	88.9	−20.4
Options	5,373,086	82,310	72.1	87.9	−15.8

This comparison might not be entirely causal. MMP usage is concentrated among the most sophisticated participants, who may also employ other optimizations (software, networking, and operational practices) that reduce latency independent of pre-margining. In options, the interpretation is further complicated by the market structure: rate limits incentivize continuous market making through mass quotes, and mass quotes must have MMP enabled [13]. Therefore, mass quotes were excluded from this analysis.

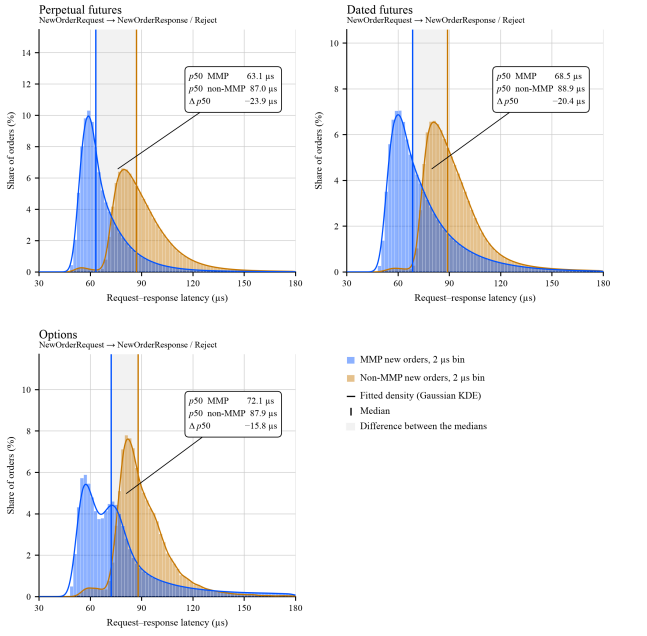

Figure 7: Request–response latency for new orders with and without MMP pre-margining.

Notes: Corvil passive-tap captures, 21–26 August 2026; new orders only, matched one-to-one to their gateway responses within each TCP session. Bars are the share of orders per 2 µs bin, the curve is a Gaussian kernel density estimate, and the shaded band spans the difference between the medians; the axis is truncated at 180 µs, beyond which fall under 1.6% of orders in every series except MMP options (6.6%).

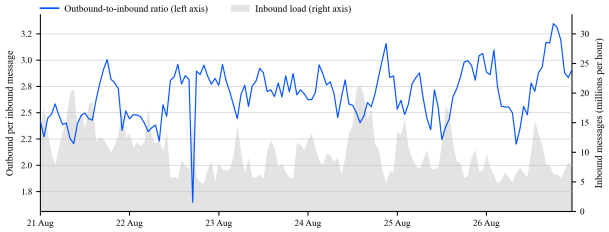

Figure 8: Outbound messages per inbound message (left axis) and hourly inbound message volume (right axis), post window.

Notes: Inbound is client requests on the Starbase order entry path (T_1); outbound is the gateway responses on that path (T_3) plus public market data messages (T_4), counted over 21–26 August 2026 in one-hour buckets. Only A-side market data is counted: every update is published twice, from redundant gateways A (195.138.37.9) and B (195.138.37.10), and the two agree to within 0.04% once duplicate captures are removed. Snapshot and recovery channels are excluded, as are `InstrumentInfo` and `InstrumentRef`—recalculated instrument state published on its own cadence rather than per book event—leaving 99.1% book updates. Session administration is excluded on both sides (section 2.7.1). The right axis is zero based.

3.6 Ratio of inbound to outbound messages

We count, per hour of the post window, inbound messages as client requests on the Starbase order entry path and outbound messages as the gateway responses on that path plus the public market data messages on one feed side. Both counts come from the Corvil captures, so they share the observation points T_1, T_3 and T_4.

Over the six days, 1.457 billion inbound requests are answered by 1.472 billion private responses and 2.344 billion public market data messages—an overall ratio of 2.62 outbound messages per inbound message. The private component is almost exactly one (1.010). The 1% excess is the unsolicited notifications the gateway emits without a request behind them (OrderFilled, OrdersCanceled, MMP triggers). The public component is 1.61 book updates per request, above one because a single mass quote updates many legs and a mass cancel removes an arbitrary number of orders.

Figure 8 shows the hourly series, and the ratio is stable: the hourly median is 2.69 (IQR 2.48 to 2.85) and it ranges from 1.65 to 3.35 across 144 hours in which inbound volume itself varies 4.6-fold. Public volume tracks that variation closely (4.2-fold), so the fan-out is a property of the protocol rather than of the traffic level.

3.7 A/B gateway (a)symmetry

Order entry is served by eight gateways arranged as four redundant pairs, one pair per product tier (BTC, ETH, Tier 2, and Tier 3) [12]. We label the odd-numbered host on each pair side A and the even-numbered host side B. The two sides are configured identically and presented to participants as interchangeable. A participant selects a side at connection time and, in practice, keeps it. If the two sides are in fact identical, they should deliver the same latency. The aggregate results of section 3.1 pool both sides and therefore do not test this.

We test for a difference during the post window (21–26 August 2026). Figure 9 summarizes the resulting latency percentiles by gateway side.

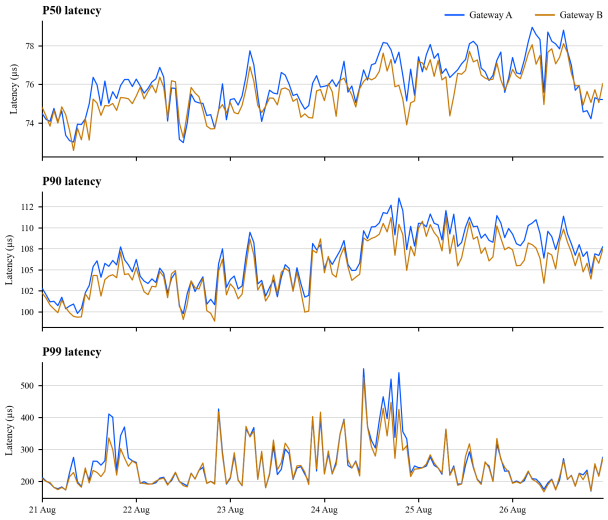

Figure 9: Request–response latency percentiles by order entry gateway side (A/B) in the post window.

Notes: Hourly buckets over 21–26 August 2026, $n = 144$ per side, pooling the four gateway pairs (BTC, ETH, Tier 2, Tier 3). The table reports the mean of the paired hourly differences $A - B$; brackets are the 95% confidence interval from Newey–West standard errors with a Bartlett kernel at 24 lags.

Table 6: Latency percentiles by gateway side (A/B), post window.

	A (µs)	B (µs)	A − B (µs)	95% CI (µs)
$p50$	76.02	75.60	+0.42	[+0.25, +0.59]
$p90$	106.34	105.19	+1.15	[+0.77, +1.53]
$p99$	254.71	248.01	+6.70	[−1.63, +15.03]

Side B is faster at all three percentiles, but the differences are small: 0.42 µs at the median and 1.15 µs at $p90$, roughly 0.6% and 1.1% of the corresponding level. Both are statistically distinguishable from zero. The $p99$ gap of 6.70 µs is larger in absolute and relative terms (2.7%), but its confidence interval is around zero, so it cannot be separated from hourly noise. The two sides are therefore close to interchangeable in practice, though not identical at the median.

4 Discussion

4.1 Interpretation and plausible mechanisms

The results in section 3 are consistent with Starbase removing large sources of queueing and scheduling variability present in the Classic stack. In the post window, we observe request–response timing on the wire ($T_3 - T_1$); in the pre window, we observe software timestamps inside the Classic stack ($T_3' - T_1'$). Because the Classic measure is taken inside the application and excludes network and edge components that are included in the wire-to-wire Corvil measure, the reported pre/post improvements are conservative (a lower bound on the corresponding wire-to-wire improvement).

After go-live, latency is both lower and more predictable

(section 3.1). Under higher load, the median changes little and congestion appears mainly as tail widening (section 3.2); the microburst case study shows bounded latency spikes with rapid recovery (section 3.3).

Participant specific latency was removed entirely via MMP pre-margining, though Starbase has low participant dependency anyway. MMP orders, which avoid per-request risk checks, are faster in the median by 16 µs to 24 µs (section 3.5) compared to non-MMP orders. Public-versus-private timing shows small but systematic path differences, with public updates typically preceding private confirmations for matched events (section 3.4).

4.2 Market-quality implications

Starbase includes a fixed-length 10 ms speed bump on aggressing orders in derivatives (perpetuals, dated futures, and options), with an exemption for perpetuals on BTC, ETH, SOL, XRP, and HYPE [9]. When request–response latency is well below this duration, market makers are likely to be protected by the speed bump in the sense that sub-millisecond differences in reaction time should not determine who is picked off. As section 3.1 shows, the vast majority of order latencies are well below the speed bump duration.

A measurement caveat is that responses for aggressing orders are sent upon entry to the speed bump. For the small fraction of orders that trade, request–response latency can therefore understate end-to-end execution latency because downstream processing occurs after the response. In the post window (21–26 August 2026), only 0.27% of CORVIL_LOG order entry requests that can mutate book state result in a trade, and the highest-trade instruments are the exempt perpetuals listed above.

Second, the relative ordering of public and private events can change participant incentives. A benefit of public events arriving before private confirmations is that it reduces the incentive to submit "canary orders" whose main purpose is to receive information early. A cost is that the party that traded may receive private confirmation later than the public update and therefore have less time to hedge before the trade is priced into the hedge market.

An open question is how often public events need to lead private confirmations in order to materially deter canary-order behavior. In our measurements, private confirmations lead public updates mainly for mass quotes (section 3.4). Canary orders, however, are more naturally expected in products where mass quotes are not typically used, such as futures and perpetuals.

Our analysis is limited in this respect because it pools all matched events and not only fills. The timing and incentives around private-versus-public ordering are most important for executed trades. A fill-only analysis is needed to quantify how often private confirmations lead public trade or book updates, and to estimate the resulting window for hedging or information acquisition.

What's clear is that participants should utilize both A and B gateways and A and B market data feeds actively as well as update their market data information with private events and order data with public events.

We do not directly measure market-quality outcomes, such as spreads, depth, fill rates, or implementation shortfall. The combination of lower latency and reduced variance in latency provides a plausible channel for improved quoting behavior, but linking performance to market outcomes requires joint analysis of latency and order-book dynamics. We leave this to future work (section 6).

5 Conclusions

Deribit's migration to Starbase is a major improvement in exchange-side performance. Median request–response latency drops from 4.8 ms to 76 µs, with larger improvements in the tail, and the post-migration distribution is substantially tighter and more stable intraday.

Under load, latency remains well-behaved: the median is largely flat with throughput and congestion appears primarily as tail widening, while a microburst case study shows a bounded increase in latency with rapid recovery (within 2 ms).

These comparisons should be read with caveats: the pre/post windows use different measurement systems and the post window over-represents early adopters, and we do not directly measure market-quality outcomes. Even so, the magnitude and consistency of the latency and predictability gains indicate a substantial upgrade for latency-sensitive trading workflows and for stable operation at high message rates.

There is more work to be done: decreasing latency at peak moments and increasing throughput will remain key as Deribit lists more products and onboards more participants. By scaling hardware and software, optimizing expensive operations and generating analyses such as this, Starbase will continue to improve as the market grows.

6 Future Work

Future analyses will connect the measured latency improvements to market outcomes and to additional sources of processing variability:

- **Liquidity and execution quality:** quantify changes in spreads, depth, queue dynamics, and implementation shortfall before and after Starbase order entry.
- **Extending microburst analysis:** identify the microburst systematically and analyze the aggregate behavior of latency through the global set of microbursts before and after Starbase.
- **Implied matching:** measure how implied order participation affects fill probability, price impact, and the timing of book updates.
- **Participant reaction speed:** estimate response times to discrete public events (book changes, trades, expiries) and assess whether faster order entry translates into faster participant behavior.
- **Transaction complexity:** categorize latency by request type and by the work each request generates (message size, mass-action fan-out, validation paths, orders matched,

number of price levels, notional executed) to identify drivers of tail latency.

Acknowledgements

Running a 24/7 exchange while replacing its core is not a controlled lab experiment; it is live surgery. We thank the Coinbase engineering, infrastructure, and exchange-operations teams for building Starbase and operating it reliably through rollout. We also thank the market makers and other early adopters who migrated early, stress-tested the system at scale, and provided feedback that helped validate production readiness. For questions, reach out to Araz Garashzade, araz.garashzade@coinbase.com.

References and further reading

[1] Deribit, *Starbase: A New Era of High-Performance Trading on Deribit*, Deribit Insights, Exchange Updates, 12 March 2026. insights.deribit.com

[2] Deribit, *Starbase API Overview*, Deribit API Documentation. docs.deribit.com/starbase/overview

[3] Deribit, *Starbase Binary API Reference*, Deribit API Documentation. docs.deribit.com/starbase/binary-api-reference

[4] Deribit, *Starbase Session Messages*, Deribit API Documentation. docs.deribit.com/starbase/session-messages

[5] Deribit, *Placing a New Order*, Deribit API Documentation. docs.deribit.com/starbase/placing-new-order

[6] Deribit, *Mass Quotes*, Deribit API Documentation. docs.deribit.com/starbase/mass-quotes

[7] Deribit, *Starbase Market Maker Protection (MMP)*, Deribit API Documentation. docs.deribit.com/starbase/mmp

[8] Deribit, *Risk Bypass*, Deribit API Documentation. docs.deribit.com/starbase/risk-bypass

[9] Deribit, *Speed Bumps*, Deribit API Documentation. docs.deribit.com/starbase/speed-bumps

[10] Deribit, *Gateway Connectivity*, Deribit API Documentation. docs.deribit.com/starbase/gateway-connectivity

[11] Deribit, *Multicast Channels*, Deribit API Documentation. docs.deribit.com/starbase/multicast-channels

[12] Deribit, *Underlying Tiers*, Deribit API Documentation. docs.deribit.com/starbase/underlying-tiers

[13] Deribit, *Starbase API Rate Limits*, Deribit API Documentation. docs.deribit.com/starbase/api-rate-limits

[14] Pico, *Corvil Analytics*. pico.net/products/corvil-analytics

[15] IEEE Std 1588-2019, *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE, 2020.

[16] Eurex, *Insights into Trading System Dynamics*. eurex.com