Cross-Frontend Attribution Methodologies in Decentralized Exchange Volume

Andrew Maury, Founder of Rantum
Rantum Studio, San Mateo, California
andrewmaury.com · cleartracedata.com
July 2026

Abstract

Decentralized exchange (DEX) volume is routinely used by liquidity-mining programs, Layer-2 (L2) foundations, and grant committees as a proxy for organic user adoption. This proxy is unreliable: the same underlying trade-execution infrastructure exposes no standard mechanism by which a frontend — a wallet-native swap widget, an aggregator, an institutional smart contract, a meta-router — announces its own identity on-chain. Volume figures that are not attributed to a true origin are trivially inflated by wash trading, proxy routing, and MEV-adjacent flow, and incentive programs built on top of them reward the wrong behavior. This paper presents the attribution methodology developed for ClearTrace, a neutral, third-party execution-intelligence system covering Ethereum, Base, Arbitrum, and Optimism. We describe four independent heuristics for recovering frontend origin from raw transaction and trace data — a calldata-suffix trapper, a proxy-router mismatch detector, a multi-hop origin tracer, and a fee-recipient clustering method — and a companion execution-quality layer that benchmarks realized trade price against a volume-weighted average price (VWAP) oracle, kept deliberately separate from a sandwich/MEV-exposure score. We report validation results from an in-production quote-accuracy sampler (112 of 168 tracked cells reaching statistical significance over a 9.6-day window), a worked example of a hypothesis reversal under additional data (order-size-dependent quote degradation at a major aggregator), and a structural finding distinguishing genuine sandwich-attack immunity from a coverage gap on Arbitrum. We close with the limitations of single-team validation and the case for treating cross-frontend attribution as a first-class, auditable input to incentive design rather than an afterthought to volume dashboards.

Keywords: decentralized exchange, MEV, attribution, on-chain analytics, execution quality, blockchain data engineering

1. Introduction

Every DEX aggregator, wallet, and institutional trading desk ultimately settles its trades through a small number of shared liquidity venues — Uniswap pools, curve pools, and the router contracts that sit in front of them. This is efficient for liquidity but destructive for measurement: once a trade lands in a shared pool, the *frontend* that originated it — the interface the end user actually touched — leaves no protocol-level signature. There is no field in a swap event that says “this trade came from Frontend X.” Standard contract decoders, block explorers, and most third-party analytics dashboards read only the immediate caller of a router contract, which is frequently a proxy, an aggregator’s internal routing layer, or a meta-contract wrapping the user’s actual wallet interface.

This gap is not a curiosity; it is load-bearing for a large amount of capital allocation. L2 foundations distribute retroactive grants and liquidity incentives based on volume attributed to protocols and, increasingly, to the frontends that bring users to those protocols. Grant programs evaluate applicants partly on the on-chain volume they claim to have generated. Both use cases assume that reported volume approximates organic adoption. In practice, volume figures without frontend-level attribution are trivially gameable: wash trading between self-owned wallets, proxy routing that obscures the true caller, and MEV-bot flow that never represents a real end user can all inflate a number that looks identical, in a standard dashboard, to genuine retail growth.

ClearTrace was built to close this gap directly, as a neutral third party with no stake in any individual protocol's volume figures. It operates on Dune Analytics' granular `trace` and `call` tables — data that preserves the full call stack of a transaction, not just its top-level `to` address — cross-referenced against an internally maintained registry of more than 172,000 labeled contracts, across Ethereum, Base, Arbitrum, and Optimism. This paper documents the attribution methodology itself: the four heuristics used to recover frontend origin, the execution-quality and MEV-exposure scoring built on top of that attribution, and the validation evidence collected from just under ten days of live production data. Our aim is to make the methodology legible and falsifiable, in the spirit that any single component described here should be independently reproducible against public on-chain data.

2. The Attribution Problem

2.1 Why frontend identity is invisible by default

A DEX trade, at the protocol level, is a sequence of one or more `swap` events emitted by a pool or router contract. The event schema records the pool, the token pair, the amounts, and the address that called the contract (`tx_from`, or, in a multi-hop trace, the address at a given `call_depth`). It does not record which website, wallet extension, or aggregator interface constructed the transaction the user signed. Three distinct classes of infrastructure sit between "the user opened an app" and "a swap event was emitted," and each defeats naive attribution differently:

– **Meta-frontends and wallet-native swap extensions** append proprietary, undocumented calldata — typically a short hex suffix after the router's standard ABI-encoded arguments — that a generic decoder silently ignores, because it does not affect execution.

– **Institutional and enterprise frontends** frequently route through custom smart contract wallets or private routing infrastructure, so the address that appears as `tx_to` is not the pool address a naive query expects, and the actual caller can be several hops removed from the pool interaction.

– **Aggregators** route a single user intent through multiple pools and sometimes multiple protocols in one transaction, so a query that treats each swap event as an independent trade over- or under-counts depending on how it handles multi-hop sequences.

Any one of these alone would be a tractable data-cleaning problem. Together, and compounded by the fact that some of this obfuscation is *intentional* — frontends that do not want to be tracked, or bots that specifically mimic organic patterns — the attribution problem is adversarial, not merely noisy. This distinction matters methodologically: a heuristic tuned to catch honest omission (e.g., a frontend that simply doesn't advertise itself) will not catch a frontend or bot actively evading detection, and a system that only handles the adversarial case may over-fit and misclassify honest long-tail frontends. ClearTrace uses four independent, non-overlapping heuristics for this reason — each is a different lens on the same problem, and agreement across lenses is itself a validation signal.

2.2 Why volume-based incentive design compounds the problem

Grant programs and liquidity-mining schedules that use raw volume as an allocation input create a direct economic incentive to defeat attribution, which is a stronger adversarial pressure than exists in most other data-engineering contexts. A protocol or frontend that can make its volume look larger, or look more "organic," has a direct financial reason to do so. This is the argument for a genuinely third-party attribution layer: an aggregator or protocol self-reporting its own frontend-level volume has a conflict of interest that a neutral observer, reading the same public on-chain data everyone else has access to, does not.

3. Methodology

3.1 Data infrastructure

ClearTrace is built on Dune Analytics' decoded `trace` and `call` tables, which expose the full internal call graph of a transaction rather than only its top-level fields. This is the foundational design choice that makes multi-hop and proxy-obscured attribution possible at all — a query restricted to `dex.trades` at the top level

cannot see past a proxy contract, while a query against the trace tables can walk the call stack to the actual pool interaction and back out to the true originating address. Architectural groundwork for this came from open-source contribution to Dune's Spellbook (the shared dbt-based data-modeling layer underlying Dune's curated tables), including 67 contract labels contributed to its models; ClearTrace's own internal registry, separate from and larger than that public contribution, currently labels more than 172,000 contracts across the four covered chains.

3.2 Four attribution vectors

ClearTrace combines four independent heuristics to recover frontend origin. None is individually sufficient; each catches a different evasion pattern.

(a) Calldata-suffix trapper. Many wallet-native swap extensions and meta-frontends append a short, static hex suffix to the end of otherwise-standard router calldata. This suffix has no effect on contract execution — it is inert from the EVM's perspective — which is precisely why standard ABI decoders strip and ignore it. ClearTrace instead treats the trailing bytes of the raw calldata as a labeled signal, extracting a fixed-width suffix (e.g., the trailing 8 hex characters) and matching it against a registry of known frontend identifiers. This vector catches identity signals that are, by construction, invisible to any decoder that only interprets calldata according to its declared ABI.

(b) Proxy-router mismatch detection. A standard query assumes the address a user's wallet directly interacts with (`tx_to`) is also the address executing the economically meaningful action (the pool, or a well-known router). Institutional frontends and custom smart-wallet infrastructure break this assumption by inserting a proxy layer. ClearTrace flags transactions where `tx_to` diverges from the pool address actually touched deeper in the trace, surfacing custom enterprise frontends and smart-wallet-mediated flow that would otherwise be invisible to top-level analysis.

(c) Multi-hop origin trace. Using SQL window functions (`LAG`/`LEAD`) over per-wallet transaction sequences ordered by time, ClearTrace reconstructs the sequence of interface interactions a given address makes across a session and across chains. This recovers origin in cases where a single user intent is split across multiple on-chain interactions — for example, a bridge-then-swap flow — and additionally surfaces interface-churn patterns (a wallet abandoning one frontend for another) that are of independent interest to protocol teams evaluating retention.

(d) Fee-recipient clustering. Many frontends monetize by routing a small basis-point fee to a static, self-controlled address as part of the swap execution. Even where a frontend takes no other identifiable on-chain action, this fee-routing pattern is a stable fingerprint: ClearTrace clusters otherwise-anonymous transactions by the destination of this fee, which is consistent across a frontend's full transaction volume even when its calldata and routing pattern vary.

A simplified illustration of vector (a), isolating a frontend identifier while filtering known proxy and bot flow:

```
-- Attribute trades to originating frontend, net of proxy/bot noise
SELECT
    evt_tx_hash,
    evt_block_time,
    tx_from AS sender,
    -- Decode the trailing calldata suffix that identifies the frontend
    RIGHT(encode(call_data, 'hex'), 8) AS frontend_id,
    amount_usd
FROM dex.trades_traces
WHERE success = true
  AND tx_from NOT IN (SELECT address FROM labels.mev_bots)
  AND call_depth <= 2  -- direct interaction; strips proxy layers
```

3.3 Execution-quality scoring

Attribution alone answers "who sent this trade"; it does not answer "did the trade execute well." ClearTrace scores execution quality by comparing each fill's realized price against a one-minute volume-weighted average

price (VWAP) oracle constructed independently from the trade itself — both the sold and bought legs of a swap are valued against an external `prices.usd` reference series, rather than against Dune's internally derived `amount_usd` field, to avoid a circular-pricing artifact in which a trade's own reported value is used to judge whether that value is reasonable. Across the four covered chains, the best-performing aggregators observed land around a median of ~7.5 basis points of effective slippage against this VWAP benchmark.

3.4 MEV and sandwich-attack detection

A sandwich attack — a bot frontrunning a pending swap with its own buy order and backrunning it with a sell, capturing the victim's price impact — is invisible to standard slippage analysis: a sandwiched trade and a large trade in a thin pool produce the same-looking slippage number. ClearTrace treats MEV exposure as a metric distinct from, and reported independently of, execution-quality slippage, on the reasoning that the two have different causes (adversarial extraction versus real market depth) and therefore different remedies.

The first implementation of this detector attempted to reconstruct in-block transaction ordering directly from `dex.trades`, using `tx_hash` value as a proxy for sequencer position — treating a numerically lower hash as an earlier transaction. This is incorrect: a transaction hash is a cryptographic digest, not a sequence number, and sorting by it produces an arbitrary permutation with no relationship to actual block inclusion order. The corrected implementation instead sources from Dune's curated `dex.sandwiched` table, built from a Spellbook detection macro that uses `evt_index` — the transaction's actual on-chain event position within its block — to reconstruct verified (frontrun, victim, backrun) triples. Switching data sources, rather than refining the ordering heuristic, eliminated the false-positive class entirely; this is worth stating plainly as a methodological lesson: an ordering assumption that seems reasonable can be silently wrong in a way that produces confident, well-formed, and completely meaningless output, and the fix required recognizing that the input data itself — not the query logic built on top of it — was the wrong primitive.

The production detection query pulls confirmed victim swaps for the four covered chains over a rolling seven-day window and attaches both global and per-chain rolling totals as window columns, so per-chain KPIs remain accurate against the full dataset rather than only the most recent page of results returned to a live feed.

4. Findings and Validation Approach

Two forms of validation evidence are reported here: (i) an in-production statistical sampler that establishes when ClearTrace's quote-accuracy claims cross a threshold of statistical reliability, and (ii) a worked example in which additional data reversed an initial hypothesis — offered as a demonstration that the methodology is falsifiable in practice, not only in principle.

4.1 Quote-accuracy sampler: reaching statistical significance

ClearTrace runs a continuous sampler comparing quoted price against realized execution price across tracked DEX aggregators, segmented by source, trading-pair, and trade-size cohort. A cell (a given source × pair × size combination) is treated as statistically rated once it accumulates at least 30 realized samples spanning at least seven days. Approximately 9.6 days after the sampler began, 112 of 168 tracked cells crossed this threshold. At the source-rollup level, five of six sampled aggregators reached rated status, with the following median quote-to-realized gaps: 1inch (0.0 bps), Uniswap (0.0 bps), KyberSwap (3.35 bps), Odos (0.0 bps at the rollup level), and Paraswap (1.0 bps). These figures are reported as a demonstration of the sampler's design — a pre-registered statistical threshold, applied uniformly across sources — rather than as a final ranking; a sixth source (0x) was rated at the individual-cell level but had not yet aggregated to a stable source-level rollup at the time of this sampling window, illustrating that "rated" status is itself cohort-dependent and should be reported at the granularity where the sample threshold is actually met.

4.2 A hypothesis reversal: order-size-dependent quote degradation

An earlier, small-sample investigation suggested that one aggregator's (Odos) quote-realization rate degraded at large trade sizes — an 84% realize rate observed at the $1M cohort versus 87% at $1K on a limited early sample, consistent with a plausible hypothesis that large trades are harder to fill at quote. With the full 9.6-day dataset (n = 916 Odos rows across cohorts), this pattern reversed: realize rates were 65.8% at $1K, 72.9% at $10K,

87.7% at $100K, and 90.7% at $1M — the $1M cohort performed *best*, and the $1K cohort performed worst. All five peer sources sampled in the same window showed flat-to-mildly-noisy realize rates across size cohorts; the one aggregator's retail-size weakness was the outlier, and it was sustained day-over-day rather than attributable to a single bad run, with misses independently classified as genuine fill failures rather than RFQ (request-for-quote) routing artifacts or coverage gaps in the underlying data.

This finding is reported not for its specific commercial implication, but as a demonstration of methodological discipline: the initial small-sample hypothesis was directionally wrong, and only additional data — collected under the same fixed methodology, not a revised one — surfaced the correct and, in this case, counter-intuitive pattern (weakness concentrated in small, "retail" trade sizes, suggestive of stale or wide quoting rather than a large-trade slippage problem). A validation approach that could not have produced this reversal would not be trustworthy when it confirms a prior expectation, either.

4.3 Structural protection versus coverage gap: the Arbitrum case

ClearTrace's sandwich-detection layer reports near-zero attack counts on Arbitrum and Optimism relative to Ethereum. Two structurally different explanations can produce an identical "near-zero" count: the detector runs correctly and genuinely finds little attack activity (a real structural finding), or the detector simply lacks usable data for that chain (a coverage gap that would be actively misleading if reported as protection). Distinguishing these requires a second, independent check: `active_days` — the number of days in a trailing 30-day window for which the chain has any recorded activity at all — read alongside the raw `sandwich_victims_30d` count. Where `active_days` is close to the full 30-day window and the victim count is still small, the detector is confirmed to be running and to be finding genuinely little; this is the Arbitrum result, and it is consistent with Arbitrum's first-come-first-served (FCFS) private-sequencing design, which removes the public mempool window that sandwich bots require to frontrun a pending transaction. This gives a foundation or protocol team citing "Arbitrum sandwich-protection" an auditable empirical basis for the claim, rather than an artifact of incomplete data being mistaken for a positive result.

4.4 Aggregator benchmark, execution and MEV combined

Across a fixed $100–$10M notional band, with a 2,000 bps threshold applied to exclude clear pricing errors from the sample, the execution-quality and MEV layers combine into a benchmark not otherwise published by any aggregator about itself:

Aggregator	Median slippage vs. VWAP	MEV toxicity	Revert rate
CoW Swap	–0.1 bps	Zero	0.0%
1inch	–0.4 bps	Low	1.2%
Odos	–0.6 bps	Medium	2.1%

The magnitude differences here are small in absolute terms — a fraction of a basis point separates the top rows — which is itself a finding: at this level of measurement precision, execution-quality differences between major aggregators are narrow, and the *MEV toxicity* and *revert rate* columns, which are structurally distinct metrics rather than restatements of slippage, carry more discriminating information than the slippage column alone.

5. Limitations and Threats to Validity

This methodology has been validated by a single team operating the production system, not by an independent third-party academic audit; the "neutral, third-party" framing refers to ClearTrace's relationship to the protocols and frontends it measures (it has no ownership stake or commercial relationship with the aggregators it benchmarks), not to independent replication of its findings by outside researchers. The contract-labeling registry, while large (172,000+ entries), is necessarily incomplete, and any frontend not yet labeled is attributed only as far as the four heuristics in Section 3.2 can reach without a label match — meaning some genuine long-tail frontend volume is likely still folded into an "unattributed" residual rather than mis-attributed to the wrong party,

which is the safer failure mode but still a limitation. The 2,000 bps junk filter and the $100–$10M notional band in Section 4.4 are fixed thresholds chosen to exclude clear pricing errors and outlier notionals; they are disclosed here so that the reported numbers can be reproduced or challenged, but they are a methodological choice, not a natural constant. Finally, all findings in this paper are drawn from four EVM-compatible chains (Ethereum, Base, Arbitrum, Optimism); the extent to which the same heuristics transfer to non-EVM execution environments is untested and left to future work.

6. Conclusion

Cross-frontend attribution is a solvable, if adversarial, data-engineering problem, and it matters more than its niche technical framing suggests: L2 incentive programs and grant committees are, in effect, making capital-allocation decisions on top of volume numbers that are attributable to a true origin only if someone builds the infrastructure to make them so. This paper has described four independent, non-overlapping heuristics for recovering frontend origin from raw on-chain trace data, an execution-quality layer built to avoid circular pricing, and an MEV/sandwich-detection layer corrected after an initial ordering assumption (`tx_hash` as sequence proxy) was found to be silently wrong. The validation evidence presented — a statistically-thresholded quote-accuracy sampler, a hypothesis that reversed under additional data rather than confirming its own prior, and a structural-versus-coverage-gap distinction on Arbitrum — is offered as evidence that the methodology is falsifiable and self-correcting in practice, which we consider a more meaningful validation standard than any single point-in-time benchmark number. Future work includes extending the labeling registry's coverage, formalizing the four attribution heuristics as an open, auditable detection specification analogous to Dune Spellbook's community-maintained macros, and testing transferability of the approach to additional execution environments as multi-chain DEX volume continues to fragment.

Andrew Maury is the founder of Rantum, a senior data science and ML studio based in San Mateo, California, that turns messy, fragmented, and adversarial data into models, APIs, and products that ship. He previously scaled data infrastructure at 0x Labs and is an open-source contributor to Dune Analytics' Spellbook. ClearTrace is available as a live public dashboard and API at cleartracedata.com.