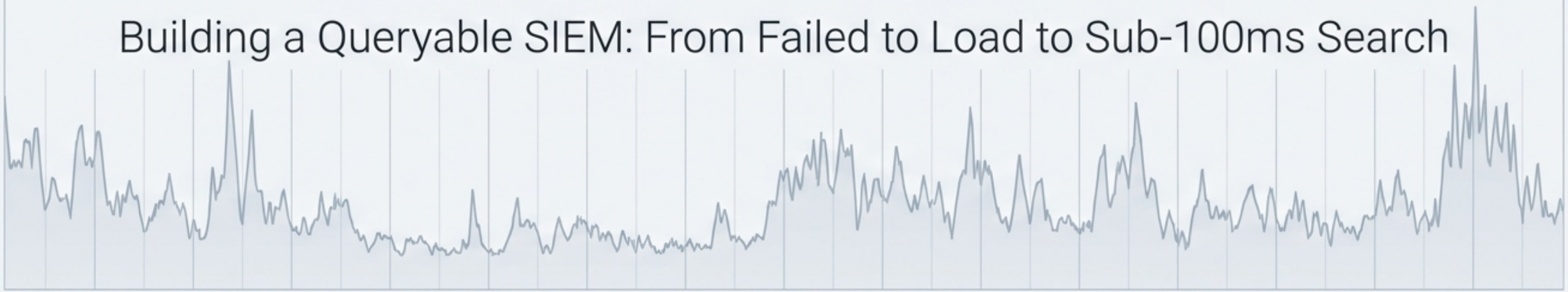
[ClickHouse] [FastAPI] [React] [Architecture-Teardown]

# Home Network Mission Control

## Building a Queryable SIEM: From Failed to Load to Sub-100ms Search



## Incident Report Card

**Symptom: 404 Error on /api/dns/query-log**

Failed to load DNS query log.

## Root Cause

Phase 1.4 SIEM cutover migrated storage to ClickHouse cleanly but dropped the SQLite-backed endpoint. The deployed dist/ was stale, calling a ghost route while CI stayed green.

## Impact

The dashboard UI became a dead end for search.

***A SIEM you can’t search is an archive.***

# The Pre-Code Workflow: 5 Architectural Decisions

| Decision | Design Choice | Expected Outcome / Rationale |
| --- | --- | --- |
| Aggregation Grain | `(eTLD+1, client)` | Cleanest signal. Subdomain depth lives one click away in the drawer. |
| Status Display | 3 numeric columns | Lossless and sortable. Matches Splunk/Elastic conventions. |
| Forensic Columns | `qtype + upstream + reply_ip` | Most-common (+N more) format via `topK(1) + uniqExact`. |
| PII Default | `pii_mode=off` | Maintains consistency with sibling routes. Toggle to reveal. |
| Time Viz | First/last + sparkline | Immediately distinguishes constant beacon from one-off pings. |

# GET /api/dns/search

Context: Queries a `client_dns_query` schema with a 180-day TTL.

| | | |
|---|---|---|
| `q` | `(string)` | Search query. Enforces a min 3-character limit on substring matches. |
| `match` | `(enum)` | `substring \| etld1` |
| `window_hours` | `(int)` | `1 \| 24 \| 168 \| 720` |
| `pii_mode` | `(enum)` | `on \| off` |
| `limit` | `(int)` | Default 200, hard cap 500. |

# Mapping Business Requirements to Database Functions

| | | |
|---|---|---|
| Requirement: Dominant Value | → | `topK(1)(col)[1]` |
| Requirement: Distinct Count | → | `uniqExact(col)` |
| Requirement: Status Splits | → | `countIf (Allowed/Blocked/Cached)` |
| Requirement: Time Boundaries | → | `min(ts) / max(ts)` |

## The Predicate Switch

| Substring Path | eTLD+1 Path |
|---|---|
| `positionCaseInsensitive(domain, q) > 0` | `etld1 = q` (Uses LowCardinality column; much faster execution) |

# Visual Equation

Row Total

=

Sum(Sparkline Buckets)

## The sparkation

The sparkline is a separate query joined client-side on `(etld1, client_mac)`. Reapplying the exact same predicate ensures bucket sums perfectly equal the aggregate row total.

## Bucket Sizing Table

| 1h Window | 24h Window | 7d Window | 30d Window |
|---|---|---|---|
| 12 x 5-minute buckets | 24 x 1-hour buckets | 28 x 6-hour buckets | 30 x 1-day buckets |

# Project Execution Phase: Parallel Gantt Timeline

**Wave 1: Backend Foundation**
(Pydantic schemas, ClickHouse query helpers, 11 tests pinning the sparkline invariant)

**Wave 2: Backend Integration**
(GET /api/dns/search router, input validation, bucket tables)

**Wave 3: Frontend**
(Types, useDnsSearch hook, inline-SVG sparkline, drawer click-through)

**Wave 4: AI Persona Reviews**
(Passing the code to bounded subagents)

**Wave 5: Deploy & Smoke Test**
(Build, restart launchd, test live queries)

# Persona Agents with Bounded Scopes Outperform Generalists.

## SIEM-Database Persona

## SIEM-Security Persona

During Wave 4, the draft code was handed to specialized subagents. Prompted to look for specific domain-level flaws (database optimization and security posture), they caught architectural logic errors, not just syntax bugs.

# Database Persona Findings

## [HIGH RISK] Full-Partition Scan

### Issue

ClickHouse `client_dns_query` sort key is `(client_mac, ts)`. A substring search across all clients bypasses the key, triggering a full-partition scan.

### Mitigation Implemented

10-second wall-clock cap (`max_execution_time`), strict 3-character minimum, and partition pruning by `ts`.

## [MEDIUM RISK] Third-Scan Waste

### Issue

Computing `total_matched_rows` ran a third full scan using `uniqExact(etld1, client_mac)`—wasted effort if results already fit under the limit.

### Fix Implemented

Short-circuit logic. If `len(rows) < limit`, `total_matched = len(rows)`. Saves one entire CH round-trip per request.

# Security Persona Findings

## [HIGH RISK] PII Gate Divergence

### Issue

The draft redacted both `MAC addresses` and pseudonyms. Sibling routes kept the pseudonym so devices remain distinguishable. Stricter is not safer; it's inconsistent.

### Fix

Aligned the PII gate to match sibling endpoint behavior exactly.

## [MEDIUM RISK] Weak Helper Signatures

Helper signatures typed `match` as a bare `str`.

Added `assert match` in `_DNS_SEARCH_VALID_MATCH` guard plus a `Literal[substring, etld1]` schema. Defense in depth.

## [LOW RISK] Raw Data Leakage

Raw error messages and MACs exposed in `data-testid` attributes.

Replaced with friendly status messages and row-index `testids`.

# The FastAPI Int-Literal Trap

**The Trap:** URL parameter arrives as a string

`?window_hours=24`

**The Failure**: Pydantic Validation

`Literal[1, 24, 168, 720]`

Pydantic checks the `Literal` match before attempting string-to-int coercion. The string 24 is not the integer 24.

**422 Unprocessable Entity**

**The Fix Path**

Type the parameter explicitly as `int` (forcing FastAPI to coerce the string first), and then apply the allowlist guard manually in the route logic.

# Wave 5: Live Smoke Test Logs

| | |
|---|---|
| ✓ | [OK] q=goog (substring, 30d): 106 rows in <100ms |
| ✓ | [OK] q=apple.com (etld1, 24h): 7 rows (Exact match path routing confirmed) |
| ✓ | [OK] q=ad (substring, 24h): 211 rows (Successfully hits the 200-default limit short-circuit) |
| ✗ | [REJECTED] q=a (substring, 24h): 400 Bad Request (3-char minimum enforced) |

***Note: The first query produced the load-bearing row...***

# The Load-Bearing Query: Proving the Invariant

| | | | | | |
|---|---|---|---|---|---|
| googleapis.com | 198 Allowed | 0 Blocked | 256 Cached | Total 454 |  |

# 454 = 454

**The Proof:** The sum of the 24 hourly buckets equals exactly 454. The invariant pinned the result on real data.

# Patterns Worth Carrying Forward

### 1 Dogfood your UI

The cheapest bug bounty is clicking through the dashboard you actually ship.

### 2 Write decisions down first

Document explicit reasoning for design choices before writing code.

### 3 Deploy bounded subagents

Persona AI with strict scopes catches architectural logic, not just typos.

### 4 Pin cross-query invariants

Integration tests must ensure related data structures (like sparklines and totals) perfectly align.

### 5 Short-circuit queries

If `len(rows) < limit`, skip the third DB scan. One line of code saves massive overhead.

### 6 Beware Literal[int] in URLs

Type as `int` to force coercion, then guard manually.

Commit 6c13af6 merged

| 13 | +1718 / −3 | 1008 |
| --- | --- | --- |
| Files Changed | Lines of Code | Backend Pytest **Passing** |

The dashboard tells the truth about the network.
The search panel makes the truth queryable.

***Build the search before someone asks why they can't find what they're looking for.***