ZelDB

A Cybersecurity-Native AI Database Architecture

Evidence, Memory, Attack Paths, and Defensive Action

CORE CLAIM

ZelDB is the first integrated cybersecurity-native AI database architecture — not the first graph database, ledger, vector store, SIEM, or AI memory component in isolation.

DOCUMENT INFORMATION

Author	Haja Mo, Founder & CEO, Rocheston
Publisher	Rocheston, The Zelfire Suite
Date	May 2026
Edition	1.0
Classification	Public
Reference	ZDB-WP-2026-001

Abstract

ZelDB introduces an integrated cybersecurity-native AI database architecture in which security objects, evidence, AI memory, attack paths, blast-radius simulations, policies, encryption context, incident replay, and defensive-action provenance are native database concepts. Its core data primitive, the ***CyberCell****, represents assets, identities, incidents, policies, evidence, AI memories, and actions with built-in risk, trust, confidence, policy, evidence, encryption, graph, timeline, and action-eligibility metadata.* ***ZelQL*** *exposes security-native database operations such as TRACE, REPLAY, VERIFY, SIMULATE, WHY, and QUARANTINE. ZelDB does not claim to invent graph analytics, immutable ledgers, vector search, SIEMs, or AI memory in isolation; rather, it claims a unified database architecture for adversarial, AI-driven cybersecurity operations.*

Keywords: cybersecurity-native database, AI database, governed AI memory, CyberCell, Security Reality Graph, ZelQL, evidence-native storage, attack-path graph, blast-radius simulation, incident replay, kinetic ledger, policy-bound retrieval, datum-bound encryption, post-quantum cryptography, autonomous defense.

Publication and Intellectual-Property Notice

This paper is a public technical white paper and invention disclosure. It is precise and defensible against obvious prior-art criticism. It is *not* a legal opinion, patentability opinion, freedom-to-operate opinion, or substitute for review by qualified patent counsel.
In the United States, inventor-originated disclosures may receive a one-year grace-period exception under AIA 35 U.S.C. § 102(b)(1), subject to conditions [24]. Other jurisdictions have different and often narrower grace-period rules, and some public disclosures may affect foreign patent rights [25]. This document is published deliberately as a defensive prior-art disclosure and as category-positioning for the Zelfire Suite.

Contents

1 Executive Summary

ZelDB is a cybersecurity-native AI database architecture for the Zelfire suite. Its purpose is to make security reality — assets, identities, policies, evidence, incidents, AI memories, attack paths, encryption state, and defensive actions — directly storable, queryable, explainable, and provable at the database layer.

The central invention is not a single isolated component. Graph databases, security data lakes, query languages, vector databases, immutable ledgers, attack-path tools, and AI memory products already exist. The originality of ZelDB is the *integration* of these capabilities into a common cybersecurity-native database substrate with shared object identity, policy enforcement, evidence semantics, AI-governed retrieval, temporal replay, and action provenance.

A defensible claim must be narrow enough to survive prior art. Therefore, ZelDB does not claim to be the first graph database, first SIEM, first log store, first vector database, first attack-path tool, first blast-radius product, first immutable ledger, or first governed AI memory system. ZelDB claims a new integrated architecture: a database where the primary object model, query language, evidence model, AI memory controls, attack graph, temporal replay, policy-bound retrieval, encryption context, and kinetic action ledger are native and mutually aware.

This paper defines the invention, distinguishes public prior art, proposes defensible claim language, and provides an implementation-oriented architecture suitable for product development, investor communication, defensive publication, and patent-counsel review.

One-Sentence Category Claim

> ZelDB is a cybersecurity-native AI database architecture that represents security-relevant reality as policy-bound **CyberCells**, links those objects through a **Security Reality Graph**, governs AI memory through evidence and action eligibility, and records defensive cyber actions through a cryptographically verifiable **Kinetic Ledger**.

2 What ZelDB Claims — and What It Does Not Claim

The strongest claim is an *integrated-systems* claim. ZelDB is published as a new category architecture, not as a statement that every component is individually unprecedented.

Do Claim	Do Not Claim
First integrated cybersecurity-native AI database architecture combining CyberCells, evidence, AI memory governance, attack paths, blast-radius simulation, incident replay, policy-bound retrieval, encryption context, and defensive-action provenance.	First database, first graph database, first security data lake, first SIEM, first vector database, or first immutable ledger.
CyberCell as a cybersecurity-native database object model with risk, trust, evidence, relationships, policy, AI visibility, and action boundaries as native fields.	Exclusive ownership of the word *CyberCell*; the term has prior unrelated use in biology [23].
ZelQL as a cybersecurity-native query interface with REPLAY, SIMULATE, WHY, VERIFY, TRACE, and QUARANTINE semantics.	First security query language; XQL, KQL, ES|QL, SysQL, and others exist.
Evidence-native storage for security events, AI decisions, and defensive actions as first-class database records.	First tamper-evident logging or first cryptographic ledger; RFC 5848 and Amazon QLDB are prior art [20, 21].
Governed AI memory specifically for cybersecurity operations, including evidence weighting, prompt-injection risk, poisoning risk, tenant scope, and action eligibility.	First AI memory product; Oracle AI Agent Memory and other research/products exist [15, 16].

3 Definitions

Cybersecurity-native database. A database architecture whose primary data model and query operations are designed around cybersecurity semantics, including risk, trust, evidence, attacker reachability, policy, identity, incident time, encryption state, AI visibility, and defensive action consequences.

AI database (in this paper). A database architecture that supports AI agents not merely by storing vectors, but by *governing* what an agent may retrieve, remember, reason over, recommend, and use for action.

CyberCell. The primary data object of ZelDB. A CyberCell represents a security-relevant object such as an asset, identity, credential, session, database, container, cloud resource, secret, policy, vulnerability, incident, evidence object, AI memory, AI agent, encryption key, or defensive action.

Security Reality Graph. The graph of relationships among CyberCells, including access paths, dependencies, trust boundaries, encryption protection, evidence links, policy governance, memory usage, and action history.

Kinetic action. A real or simulated defensive cybersecurity action that changes the environment, such as isolating a host, revoking a token, rotating a credential, blocking traffic, quarantining a file, disabling a user, or triggering deception.

Action eligibility. A database-level property controlling whether a piece of data, memory, evidence, or recommendation may be used merely for reading, for reasoning, for recommendation, for simulation, or for real-world action.

4 The Problem: Cybersecurity Data Is Not Ordinary Data

Most databases store records, documents, events, vectors, or graph nodes. Cybersecurity operations need something different: security objects whose meaning changes with risk, trust, evidence quality, attacker reachability, policy, time, encryption state, and operational consequence.

An asset is not merely an asset. It may be internet-facing, unpatched, owned by a business service, connected to a crown-jewel database, protected by a key, implicated in an incident, exposed through a credential, governed by a policy, visible to an AI agent, and dangerous to isolate during business hours.

A log is not merely a log. It may be weak evidence, strong evidence, signed evidence, contradictory evidence, AI-generated evidence, human-approved evidence, stale evidence, or evidence that supports a defensive action.

An AI memory is not merely text. In cybersecurity, a memory can be poisoned, stale, tenant-scoped, policy-blocked, evidence-supported, prompt-injection-bearing, sensitive, or forbidden for action.

A defensive action is not merely an audit event. It is a consequential operation that requires provenance: who requested it, what evidence supported it, what policy allowed it, what blast radius was predicted, what actually happened, what rollback path existed, and what proof was generated.

5 Prior-Art Landscape and Boundaries

The public prior-art landscape shows a clear direction: security platforms are adding data lakes, graphs, AI reasoning, memory, evidence, and exposure analytics. This validates the market need. It also requires careful claim boundaries.

Microsoft Sentinel is positioned as a security platform unifying cloud-native SIEM, data lake, graph-enabled visibility, and intelligent reasoning tools [2, 3, 4]. This is strong adjacent prior art. ZelDB distinguishes itself by defining a database-native object model, query language, AI memory governance layer, action ledger, and evidence semantics, rather than a platform-level security data lake plus analytics layer.

Google Security Operations uses UDM as a standard data structure / schema. Google documents that SecOps stores original raw logs and structured UDM records, with UDM being a structured representation of original logs [5]. ZelDB distinguishes itself by making security objects and their policy / evidence / action semantics first-class database primitives rather than normalized event records alone.

OCSF is an open standard for cybersecurity event logging and data normalization, and it is agnostic to storage format, collection method, and implementation [6, 7]. ZelDB can ingest or emit OCSF, but ZelDB is not merely a schema; it is a database architecture built around

security-native objects and queries.

Wiz Security Graph and **Tenable Exposure Management** provide serious prior art for attack paths, risk graphs, exposure analytics, and blast-radius concepts [8, 9, 10, 11, 12]. ZelDB does not claim to invent attack graphs or blast radius. Its defensible distinction is making these concepts native to a database object model and query language integrated with evidence, AI memory, policy, time, encryption, and action provenance.

Sysdig SysQL is strong prior art for a security query language over connected entities and relationships. Sysdig describes SysQL as a lightweight query language inspired by Cypher for interacting with the Sysdig Secure datastore, and its API primarily retrieves resource metadata and executes SysQL statements [13, 14]. ZelQL is therefore claimed not as the first security query language, but as a broader query language for database-native replay, simulation, explanation, evidence verification, and AI memory quarantine.

Oracle AI Agent Memory is important prior art for governed enterprise AI memory on a database. Oracle describes it as a governed memory core on Oracle AI Database with vector search, graph traversal, relational query capabilities, governance, isolation, encryption, access control, and auditing [15, 16]. ZelMemory is therefore claimed as cybersecurity-specific governed AI memory with SOC evidence weighting, prompt-injection quarantine, poisoning risk, action eligibility, and kinetic-action safety.

Vector databases do support multi-tenancy. Pinecone documents namespace-based physical isolation for tenants in serverless architecture; Weaviate documents tenant isolation through separate shards; Milvus documents partition-level multi-tenancy with physically separated tenant data [17, 18, 19]. ZelDB does not claim vector databases lack tenant isolation. The accurate distinction is that general vector databases do not provide cybersecurity-native adversarial memory governance by default.

Cryptographic ledgers and signed logs are established. Amazon QLDB provides a transparent, immutable, cryptographically verifiable transaction log [20], and RFC 5848 describes signed syslog with origin authentication, message integrity, replay resistance, sequencing, and detection of missing messages [21]. ZelDB does not claim to invent tamper-evident logging. The defensible claim is cybersecurity evidence and defensive-action provenance as native database semantics.

Post-quantum cryptography is standardized prior art. NIST finalized FIPS 203, FIPS 204, and FIPS 205 for ML-KEM, ML-DSA, and SLH-DSA in 2024 [22]. ZelDB does not claim to invent post-quantum cryptography. Its claim is encryption-awareness and datum / context-bound access applied to CyberCells.

5.1 Prior-Art Boundary Matrix

Area	Public Prior Art	ZelDB Boundary
Security data lake	Microsoft Sentinel data lake and graph-enabled visibility [2, 4]	ZelDB is a database-native object / query / evidence / action architecture, not only a data lake.
Normalized security schema	Google UDM and OCSF [5, 6, 7]	ZelDB can use schemas but claims native semantics, not event normalization alone.
Attack paths	Wiz Security Graph; Tenable Attack Path [8, 10]	ZelDB does not claim attack paths alone; it claims native database integration with evidence, AI memory, policy, and action.
Blast radius	Tenable blast-radius query; Wiz blast-radius context [9, 11, 12]	ZelDB claims SIMULATE as a database query primitive for security and business consequence modeling.
Security query language	Sysdig SysQL [13, 14]	ZelQL claims broader replay, simulation, why, verify, and quarantine semantics.
AI memory	Oracle AI Agent Memory [15, 16]	ZelMemory claims cybersecurity-specific evidence, poisoning, prompt-injection, and action-eligibility controls.
Vector multi-tenancy	Pinecone, Weaviate, Milvus [17, 18, 19]	ZelDB avoids the false claim that vector DBs lack isolation; it claims adversarial AI memory governance.
Immutable evidence	Amazon QLDB; RFC 5848 [20, 21]	ZelDB claims evidence / action provenance as native cybersecurity database semantics.

6 The Core Invention: CyberCell, Security Reality Graph, and Governed Action

The core invention of ZelDB is the combination of three mutually reinforcing primitives: **CyberCell**, **Security Reality Graph**, and **Governed Action**.

A *CyberCell* stores a security object with built-in risk, trust, confidence, evidence, policy, AI visibility, encryption state, timeline, relationships, and permitted action boundaries.

The *Security Reality Graph* connects CyberCells through typed relationships: identity access, asset dependency, credential use, vulnerability exposure, policy governance, encryption protection, evidence support, memory use, and defensive action impact.

Governed Action ensures that data retrieval, AI reasoning, recommendation, simulation, and real-world action are separate permission states. ZelDB does not treat access to information as permission to act on it.

Together, these primitives turn a database into a security reasoning substrate. The database is

no longer a passive store; it becomes the structured memory, evidence vault, policy boundary, and action provenance layer of autonomous defense.

7 The CyberCell Data Model

The CyberCell is the primary storage unit of ZelDB. It is not simply a row, document, node, or vector. It is a security-aware database object.

A CyberCell can represent an asset, identity, credential, secret, cloud resource, container, endpoint, business service, vulnerability, exploit, policy, incident, evidence object, AI memory, AI agent, encryption key, or defensive action.

Every CyberCell may include the following native fields: object type, owner, tenant, zone, business service, sensitivity, risk score, trust score, confidence score, evidence status, policy status, encryption status, AI visibility, action eligibility, mutation history, timeline link, graph relationships, and metadata.

The novelty is not that these fields are impossible to store in an existing database. Any general-purpose database can store them as columns or JSON. The novelty is that ZelDB makes them part of the *native object semantics and query grammar*. In ZelDB, risk, trust, evidence, policy, AI visibility, and action eligibility are not annotations; they are operating concepts of the database.

8 ZelQL: Querying Security Reality

ZelQL is the cybersecurity-native query language for ZelDB. Its purpose is to express the operations security teams and AI agents actually need: find dangerous objects, trace reachability, replay incidents, verify evidence, simulate actions, explain decisions, ask AI safely, and quarantine unsafe memory.

Example ZelQL statements:

```
FIND CyberCells
WHERE risk > 80 AND zone = "production"
SHOW evidence, attack_paths, blast_radius

TRACE Identity("admin@rocheston.com")
TO    CrownJewel("customer_prod")
SHOW credentials, sessions, vulnerabilities, evidence

REPLAY Incident("INC-2026-0042")
FROM first_seen TO contained
SHOW timeline, graph, evidence, ai_decisions

SIMULATE Action("isolate_host") ON Asset("prod-web-07")
SHOW blast_radius, business_impact, rollback_plan, approval_required

WHY Memory("MEM-9201") IS quarantined
SHOW evidence, policy, poisoning_risk, prompt_injection_risk
```

Listing 1: ZelQL example statements

The defensible novelty of ZelQL is its semantics. SysQL, XQL, KQL, ES|QL, Cypher, and SQL all provide useful retrieval and analysis capabilities. ZelQL's contribution is the database-native expression of cybersecurity operations that are temporal, evidentiary, explanatory, policy-bound,

AI-governed, and action-aware.

9 Evidence-Native Storage

ZelDB treats evidence as a first-class database object. A log says something happened. *Evidence* says something happened, who observed it, how reliable the source is, what changed, which policy applied, which AI model or human decision used it, how it links to a chain of custody, and whether it can be verified.

EvidenceCell records may include source, observer, actor, target, timestamp, before state, after state, event type, source reputation, confidence, chain identifier, hash, signature, related CyberCell, related incident, related policy, related action, and compliance mapping.

The claim boundary is important: cryptographic logging, signed syslog, append-only ledgers, and immutable transaction history are prior art [20, 21]. ZelDB's contribution is the use of evidence-native records as a *common database model* for cybersecurity state, AI reasoning, policy decisions, incident replay, and defensive action provenance.

10 Attack-Path Graph Engine

ZelDB maintains attack-relevant relationships among CyberCells inside the Security Reality Graph. This allows the database to answer reachability questions: what can an attacker reach if an identity is compromised, which assets connect to crown jewels, which credentials are dangerous pivots, and which vulnerabilities matter because of where they sit in the environment.

The standalone idea of attack paths is not new. Wiz and Tenable provide commercial attack-path and exposure-graph capabilities [8, 10, 11]. ZelDB's defensible contribution is making attack paths native to the *database object model and query interface*, and making them aware of evidence, AI memory, policies, encryption, timeline, and kinetic actions.

This means an attack path in ZelDB is not only a visual graph path. It is a database fact with evidence support, policy consequences, action options, replay context, and blast-radius implications.

11 Blast-Radius Query Engine

ZelDB's Blast-Radius Query Engine answers consequence questions before compromise or containment. It is designed to answer not merely "what can an attacker reach?" but "what happens if this object is compromised, isolated, blocked, patched, revoked, or changed?"

A ZelDB `SIMULATE` query may return affected services, affected users, sensitive data exposure, business impact, attack paths removed, attack paths introduced, risk delta, trust delta, downtime estimate, required approvals, evidence requirements, and rollback plan.

Because Tenable and Wiz already use blast-radius terminology and functionality in exposure management [9, 11, 12], ZelDB does not claim the blast-radius concept itself. Its stronger claim is *simulation-oriented blast-radius querying as a native database primitive* integrated with CyberCells, policies, evidence, AI memory, and action provenance.

12 ZelMemory: Cybersecurity-Governed AI Memory

ZelMemory is the AI memory subsystem of ZelDB. It is designed for adversarial cybersecurity environments where memory can be poisoned, outdated, tenant-scoped, policy-blocked, secret-bearing, prompt-injection-contaminated, or unsafe for action.

A MemoryCell may include source, tenant, sensitivity, trust score, evidence score, freshness score, poisoning risk, prompt-injection risk, contradiction status, AI visibility, training permission, action eligibility, human verification status, revocation status, and expiration.

Oracle AI Agent Memory is strong prior art for governed enterprise AI memory on a database [15, 16]. ZelDB's distinction is the *cybersecurity-specific threat model*: the database must decide not only whether a memory is relevant, but whether it is safe, evidenced, authorized, non-poisoned, and eligible for recommendation or action in a SOC context.

ZelMemory therefore separates read, reason, recommend, simulate, execute, export, and train permissions. A memory can be visible to a human but not usable by an AI agent. It can be usable for summary but not for action. It can be preserved as evidence while quarantined from retrieval.

13 Policy-Bound Retrieval and Action-Safe AI

In ZelDB, data access is not the same as data use. This is central to the invention.

Traditional access control asks whether a principal may read or write a record. ZelDB additionally asks: may this AI model see it? May it summarize it? May it embed it? May it train on it? May it use it to recommend an action? May it use it to execute an action? Does the action require human approval? Is the target production? Is rollback available?

This is called **policy-bound retrieval**. Retrieval results are filtered and ranked not only by relevance but by tenant scope, sensitivity, evidence strength, freshness, poisoning risk, prompt-injection risk, policy constraints, and action eligibility.

The result is **action-safe AI retrieval**: context returned to an AI agent is not merely similar. It is authorized, fresh, evidence-weighted, policy-compliant, and safe for the intended use.

14 Kinetic Ledger

The Kinetic Ledger records defensive cybersecurity actions and their provenance. Each ActionCell may include requested-by, approved-by, executed-by, target, action type, evidence basis, AI recommendation, policy decision, ZelC rule, predicted blast radius, actual impact, risk before, risk after, rollback status, proof status, timestamp, and cryptographic chain reference.

General immutable ledgers are prior art, and Amazon QLDB is a clear example [20]. ZelDB's novelty is a ledger *specialized for defensive cyber action provenance*. It records not simply that data changed, but why a defensive action was allowed, what evidence supported it, what safety boundary governed it, what outcome occurred, and what proof exists.

This is especially important for autonomous defense. An AI-enabled SOC cannot safely operate if actions are merely logged after the fact. The action must be *bounded before execution*, evidenced during execution, verified after execution, and explainable during review.

15 The WHY Engine and Incident Replay

The WHY Engine answers causal questions. Why did risk increase? Why was an AI action blocked? Why was a memory quarantined? Why did a containment simulation require approval? Why is an identity high risk? Why did a policy deny export?

The WHY query is a database operation over CyberCells, EvidenceCells, MemoryCells, PolicyCells, IncidentCells, and ActionCells. The result is a structured reasoning path, not a generic AI explanation.

The Incident Replay Engine reconstructs incidents from database-native state. It can replay initial access, credential use, privilege escalation, lateral movement, detection, evidence formation, AI recommendation, human approval, containment, verification, and residual risk.

Incident timelines exist in SIEM and XDR platforms. ZelDB's distinction is REPLAY as a *database query* over connected security objects, evidence records, AI decision receipts, policies, and action ledger entries.

16 Encryption-Aware and Datum-Bound Access

ZelDB integrates encryption state into the data model. A CyberCell can know whether it is protected, by which key, under which policy, with what rotation status, and under what conditions decryption is permitted.

Because post-quantum cryptography is standardized prior art [22], ZelDB does not claim to invent post-quantum cryptography. The claim is that *encryption state and decryption context are native properties of the cybersecurity database object*.

Datum-bound access means a key alone is not enough. Access may require the correct tenant, role, policy, evidence state, datum context, AI visibility rule, and action purpose. This aligns encryption with security reality instead of treating encryption as a separate storage option.

17 Architecture

ZelDB is implemented as a layered system. The layers below are logical; they are realized through a combination of relational storage, graph indexing, vector indexing, append-only logs, policy engines, and AI services.

Layer	Responsibility	Core Objects
CyberCell Store	Stores security-native objects and their mutable state.	CyberCell, RelationshipEdge, TimelineState
Security Reality Graph	Maintains relationships, reachability, attack paths, dependencies, crown-jewel exposure, and trust boundaries.	RelationshipEdge, AttackPath, CrownJewelMap
Evidence Engine	Stores evidence as first-class, verifiable, chainable records.	EvidenceCell, ChainRecord, ProofBundle
Temporal Engine	Supports replay and before / after state reconstruction.	TimelineEvent, IncidentState, MutationRecord
Policy Engine	Controls data use, AI visibility, retrieval, export, training, simulation, and action eligibility.	PolicyCell, QueryIntent, ApprovalRule
ZelMemory	Governs AI memory retrieval under adversarial and tenant-aware conditions.	MemoryCell, MemoryTrustScore, QuarantineRecord
Kinetic Ledger	Records defensive cyber actions with evidence, approvals, blast radius, outcome, and rollback.	ActionCell, ActionReceipt, RollbackPlan
ZelQL Interface	Exposes FIND, TRACE, REPLAY, VERIFY, SIMULATE, WHY, ASK, SHOW, ANCHOR, and QUARANTINE operations.	ZelQLStatement, QueryResult, ExplainPath

18 Defensible Novelty Matrix

The following matrix gives publication-safe wording. Each claim is deliberately framed to avoid overclaiming where public prior art is strong.

ZelDB Component	Prior-Art Risk	Publication-Safe Claim
CyberCell Data Model	Moderate: schemas and entity models exist; CyberCell name has unrelated biology use.	A cybersecurity-native database object model combining risk, trust, evidence, policy, AI visibility, encryption, timeline, relationships, and action boundaries as native semantics.
ZelQL	Moderate: XQL, KQL, ES|QL, SysQL exist.	A cybersecurity-native query language for replay, simulation, explanation, evidence verification, traceability, and memory quarantine, not merely retrieval.
Evidence-Native Storage	High: signed logs and immutable ledgers exist.	Evidence as a first-class cybersecurity database primitive linked to AI decisions, policies, incidents, and actions.
Attack-Path Graph	High: Wiz, Tenable, CyGraph / Neo4j patterns exist.	Native attack-path integration inside the database object model with evidence, policy, AI memory, and action provenance.
Blast-Radius Query	High: Tenable / Wiz use blast radius.	SIMULATE as a database query primitive that models security and business consequences before action.
ZelMemory	Moderate: Oracle AI Agent Memory exists.	Cybersecurity-specific governed AI memory with evidence weighting, poisoning risk, prompt-injection quarantine, and action eligibility.
Kinetic Ledger	Low-to-moderate: general ledgers exist.	A defensive cyber action ledger recording evidence basis, approvals, AI recommendation, blast radius, outcome, rollback, and proof.
WHY Engine	Moderate: XAI exists.	Database-native causal explanation over evidence, policy, AI, action, and security state.
Incident Replay	Moderate: SIEM / XDR timelines exist.	REPLAY as a database-native traversal of connected CyberCells, evidence, policy, AI decisions, and action records.
Datum-Bound Encryption	Moderate: PQC and ABAC / ABE exist.	Encryption-aware CyberCells where decryption depends on security context, datum, tenant, evidence, and purpose.

19 Plain-English Invention Claims for Counsel Review

The following is a plain-English invention disclosure framework intended to help counsel convert the architecture into provisional or non-provisional patent claims. It is not formal patent-claim drafting.

Independent concept. A database system for cybersecurity and AI operations comprising a security-native object model, a graph of security relationships, evidence-native storage, policy-bound AI retrieval, temporal incident replay, simulation-oriented blast-radius queries, encryption-context awareness, and a defensive-action ledger, wherein the components share common object identity and enforce tenant, policy, evidence, and action boundaries at query time.

Dependent concept family 1 — CyberCell. A database object representing a security-relevant entity with native fields for risk, trust, confidence, sensitivity, tenant, owner, policy state, evidence state, encryption state, AI visibility, action eligibility, relationships, and mutation history.

Dependent concept family 2 — ZelQL. A query interface that supports retrieval, traceability, replay, evidence verification, simulation, explanation, AI-question answering, anchoring, and quarantine operations over cybersecurity-native objects.

Dependent concept family 3 — ZelMemory. A memory subsystem that stores AI-accessible memory with evidence score, trust score, freshness, poisoning risk, prompt-injection risk, tenant scope, sensitivity, training permission, recommendation eligibility, and action eligibility.

Dependent concept family 4 — Kinetic Ledger. An action-provenance ledger that records defensive cyber actions with requester, approver, executor, evidence basis, policy decision, AI recommendation, predicted blast radius, actual impact, risk delta, rollback state, and cryptographic proof.

Dependent concept family 5 — Policy-bound retrieval. A query mechanism that filters and ranks data returned to human users and AI agents based on intent, sensitivity, tenant, evidence confidence, source trust, freshness, poisoning risk, prompt-injection risk, training permission, and action eligibility.

Dependent concept family 6 — Incident replay. A temporal database operation that reconstructs an incident from connected object state, evidence chains, AI decision records, policies, and action ledger entries.

20 Critic-Response Playbook

“Microsoft Sentinel already has a data lake and graph.”

Correct. ZelDB does not claim to be the first security data lake or graph. ZelDB claims a database-native object model and query language where evidence, AI memory governance, policies, encryption context, replay, simulation, and defensive-action provenance are first-class and integrated.

“Google UDM and OCSF already normalize cybersecurity data.”

Correct. ZelDB can ingest normalized data, but it is not merely a schema. UDM and OCSF normalize events; ZelDB gives security objects native state, relationships, evidence, action boundaries, and AI-governed retrieval.

“Wiz and Tenable already do attack paths and blast radius.”

Correct. ZelDB does not claim those concepts in isolation. It claims database-native integration of attack paths and blast-radius simulation with CyberCells, policies, AI memory, evidence, incident replay, and action provenance.

“Sysdig already has SysQL.”

Correct. ZelQL is not described as the first security query language. It is a broader cybersecurity database query language with REPLAY, SIMULATE, WHY, VERIFY, and QUARANTINE semantics beyond read-only connected-entity queries.

“Oracle already has governed AI memory.”

Correct. Oracle AI Agent Memory is general enterprise agent memory. ZelMemory is cybersecurity-specific, adding evidence weighting, poisoning risk, prompt-injection quarantine, SOC policy constraints, tenant boundaries, and action eligibility for autonomous defense.

“Vector databases support multi-tenancy.”

Correct. Pinecone, Weaviate, and Milvus document multi-tenancy mechanisms. ZelDB’s claim is not tenant isolation alone; it is adversarial memory governance and action-safe retrieval.

“Immutable ledgers already exist.”

Correct. ZelDB does not invent immutable logs. It applies evidence and ledger semantics to cybersecurity state, AI decisions, and defensive-action provenance as native database concepts.

“CyberCell name is already used.”

Correct, in an unrelated biological database. ZelDB claims CyberCell as a cybersecurity-native data model, not exclusive ownership of the generic word.

21 Implementation Implications for the ZelDB Product

The first product release visibly embodies the invention rather than merely describing it. The user interface exposes the database primitives directly: CyberCell Explorer, Security Reality Graph, ZelQL Console, Evidence Vault, ZelMemory AI, Blast Radius Simulator, Incident Replay, Kinetic Ledger, Policy Center, Encryption & Keys, and Proof Bundles.

The database schema keeps object identities stable across modules. For example, an incident is not a detached record separate from evidence, memory, graph, and action history. It is a CyberCell with relationships to EvidenceCells, MemoryCells, ActionCells, PolicyCells, and TimelineEvents.

AINA Intelligence never receives raw context only because it is semantically similar. It receives context only after policy-bound retrieval verifies tenant scope, sensitivity, evidence strength, freshness, poisoning risk, prompt-injection risk, and action eligibility.

The Kinetic Ledger initially supports simulated actions even before full ZelC integration. That allows ZelDB to demonstrate before / after risk, approval flows, rollback planning, and action receipts without executing real-world containment.

The ZelQL interpreter is implemented early because ZelQL is a key category-defining feature. Even where v1 supports a subset, the product demonstrates FIND, TRACE, REPLAY, VERIFY, SIMULATE, WHY, and QUARANTINE flows.

22 Public Headline and Abstract

Public headline: ZelDB — A Cybersecurity-Native AI Database Architecture for Evidence, Memory, Attack Paths, and Defensive Action.

Public abstract: ZelDB introduces an integrated cybersecurity-native AI database architecture in which security objects, evidence, AI memory, attack paths, blast-radius simulations, policies, encryption context, incident replay, and defensive-action provenance are native database concepts. Its core data primitive, the CyberCell, represents assets, identities, incidents, policies, evidence, AI memories, and actions with built-in risk, trust, confidence, policy, evidence, encryption, graph, timeline, and action-eligibility metadata. ZelQL exposes security-native database operations such as TRACE, REPLAY, VERIFY, SIMULATE, WHY, and QUARANTINE. ZelDB does not claim to invent graph analytics, immutable ledgers, vector search, SIEMs, or AI memory in isolation; it claims a unified database architecture for adversarial, AI-driven cybersecurity operations.

23 Conclusion

ZelDB's most defensible originality is the *integrated architecture*. Public prior art exists for many component ideas: security data lakes, normalized schemas, attack graphs, blast radius, security query languages, vector database multi-tenancy, governed AI memory, signed logs, immutable ledgers, and post-quantum cryptography.

The invention is the way ZelDB unifies these into a single database-native model for cybersecurity and AI defense: CyberCells, Security Reality Graph, EvidenceCells, ZelMemory, ZelQL, policy-bound retrieval, incident replay, blast-radius simulation, encryption context, and Kinetic Ledger.

Published carefully, ZelDB defines a new category: the **cybersecurity-native AI database**. The strongest wording is not "nobody has ever done any of this." The strongest wording is: *"no public product appears to combine these capabilities as a unified database architecture with native cybersecurity semantics, governed AI memory, evidence-first storage, and defensive-action provenance."*

That is the claim that survives critics.

> **ZelDB redefines what a database is for the AI security age. Not a faster row store. Not a smarter document store. Not a new vector index. A cybersecurity-native AI database — where evidence, memory, action, and reality live together.**

References

[1] ANSI Blog. *The SQL Standard — ISO/IEC 9075:2023 (ANSI X3.135).* SQL was standardized as ANSI X3.135 in 1986 and adopted by ISO as ISO 9075:1987.
https://blog.ansi.org/ansi/sql-standard-iso-iec-9075-2023-ansi-x3-135/

[2] Microsoft Learn. *Microsoft Sentinel data lake overview.* Sentinel data lake is a purpose-built, cloud-native security data lake for ingesting, storing, and analyzing security data at scale.
https://learn.microsoft.com/en-us/azure/sentinel/datalake/sentinel-lake-overview

[3] Microsoft Learn. *What's new in Microsoft Sentinel — Build custom graphs.* Microsoft describes custom graphs across Sentinel data lake and third-party data to uncover attack paths, blast radius, and hidden relationships.
https://learn.microsoft.com/en-us/azure/sentinel/whats-new

[4] Microsoft Learn. *What is Microsoft Sentinel graph?* Sentinel graph is a unified graph analytics capability for modeling, analyzing, and visualizing complex relationships.
https://learn.microsoft.com/en-us/azure/sentinel/datalake/sentinel-graph-overview

[5] Google Cloud Documentation. *Overview of the Unified Data Model.* Google SecOps stores original raw logs and structured UDM records; UDM is a standard data structure / schema.
https://docs.cloud.google.com/chronicle/docs/event-processing/udm-overview

[6] OCSF GitHub. *OCSF Schema.* OCSF is an open standard for cybersecurity event logging and data normalization.
https://github.com/ocsf/ocsf-schema

[7] OCSF GitHub. *Open Cybersecurity Schema Framework.* OCSF is agnostic to storage format, data collection, and ETL implementation.
https://github.com/ocsf

[8] Wiz. *Wiz Security Graph.* Wiz Security Graph maps risk across code and cloud and surfaces real attack paths.
https://www.wiz.io/lp/wiz-security-graph

[9] Wiz. *Wiz Security Graph for cloud incident response.* Wiz uses blast radius in the context of cloud incident impact and mitigation.
https://www.wiz.io/blog/wiz-security-graph-enhances-cloud-incident-response

[10] Tenable Docs. *Attack Path.* Tenable Exposure Management uses graph analytics and MITRE ATT&CK to create attack paths and top attack techniques.
https://docs.tenable.com/exposure-management/Content/attack-path/attack-path.htm

[11] Tenable Docs. *Interact with Attack Path Query Data.* Tenable supports Asset Exposure and Blast Radius query types for visualizing attack paths.
https://docs.tenable.com/exposure-management/Content/attack-path/interact-with-attack-path-query-data.htm

[12] Tenable Docs. *Key Terms.* Tenable defines Node Exposure Score as a metric for understanding blast radius exposure of a node.
https://docs.tenable.com/exposure-management/Content/getting-started/key-terms.htm

[13] Sysdig Docs. *SysQL Reference Library.* SysQL is a lightweight query language developed by Sysdig, inspired by Cypher, for interacting with Sysdig Secure datastore entities and relationships.
https://docs.sysdig.com/en/sysdig-secure/sysql-reference/

[14] Sysdig Docs. *SysQL API.* SysQL API executes standard SysQL statements and retrieves resource metadata from the Sysdig Secure datastore; primarily read-only operations.
https://docs.sysdig.com/en/developer-tools/sysql-api/

[15] Oracle Blog. *Oracle AI Agent Memory: A Governed, Unified Memory Core.* Oracle AI Agent Memory provides a governed memory core on Oracle AI Database with vector search, graph traversal, relational query capabilities, governance, isolation, encryption, access control, and auditing.
https://blogs.oracle.com/developers/oracle-ai-agent-memory-a-governed-unified-memory-core-for-enterprise-ai-agents

[16] Oracle Docs. *About Agent Memory.* Oracle AI Agent Memory provides a persistent memory layer for enterprise AI agents on Oracle AI Database.
https://docs.oracle.com/en/database/oracle/agent-memory/26.4/agmea/about.html

[17] Pinecone Docs. *Implement multi-tenancy.* Pinecone documents namespace-based multi-tenancy and physical tenant isolation in serverless architecture.
https://docs.pinecone.io/guides/index-data/implement-multitenancy

[18] Weaviate Docs. *Multi-tenancy operations.* Weaviate multi-tenancy provides data isolation; each tenant is stored on a separate shard.
https://docs.weaviate.io/weaviate/manage-collections/multi-tenancy

[19] Milvus Docs. *Implement Multi-tenancy.* Milvus documents partition-level multi-tenancy with physically separated tenant data.
https://milvus.io/docs/multi_tenancy.md

[20] AWS. *Introducing Amazon Quantum Ledger Database.* Amazon QLDB provides a transparent, immutable, cryptographically verifiable transaction log.
https://aws.amazon.com/about-aws/whats-new/2018/11/introducing-amazon-qldb/

[21] RFC Editor. *RFC 5848: Signed Syslog Messages.* RFC 5848 describes origin authentication, message integrity, replay resistance, message sequencing, and detection of missing messages for syslog.
https://www.rfc-editor.org/rfc/rfc5848.html

[22] NIST. *NIST Releases First 3 Finalized Post-Quantum Encryption Standards.* NIST finalized FIPS 203 ML-KEM, FIPS 204 ML-DSA, and FIPS 205 SLH-DSA in 2024.
https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards

[23] re3data. *CyberCell Database.* The name CyberCell has prior unrelated use in a biology repository started in 2004 and closed in 2024.
https://www.re3data.org/repository/r3d100010131

[24] USPTO MPEP. *Prior-Art Exceptions Under 35 U.S.C. § 102(b)(1).* U.S. inventor-originated disclosures may have a one-year grace-period exception under AIA 35 U.S.C. § 102(b)(1), subject to conditions.
https://www.uspto.gov/web/offices/pac/mpep/s2153.html

[25] WIPO. *Grace Periods for Disclosure of Inventions.* Grace-period rules vary by jurisdiction and may be limited or unavailable outside the United States.
https://www.wipo.int/scp/en/national_laws/grace_period.pdf

ZelDB

A Cybersecurity-Native AI Database Architecture

Part of the Zelfire Suite | Rocheston | ZDB-WP-2026-001 | May 2026
Edition 1.0 | Public

Appendix A

Defensive Prior-Art Publication and Filing Declaration

Document: ZelDB - A Cybersecurity-Native AI Database Architecture | **Reference:** ZDB-WP-2026-001 | **Author:** Haja Mo, Rocheston | **Classification:** Public

A.1 Purpose of this Appendix

This appendix is added to make the publication intent explicit. The ZelDB white paper is deliberately released as a public defensive prior-art disclosure. Its purpose is to disclose the ZelDB architecture, combinations, subcombinations, data models, query semantics, evidence mechanisms, AI governance mechanisms, action-provenance mechanisms, and implementation patterns sufficiently for the public record, so that later patent applications by third parties covering the same or substantially obvious subject matter may be rejected, narrowed, challenged, or invalidated using this publication as non-patent literature prior art.

A.2 Subject Matter Disclosed as Prior Art

- The CyberCell data model: cybersecurity-native objects carrying risk, trust, confidence, evidence, policy, encryption state, AI visibility, timeline, relationships, and action eligibility.
- The Security Reality Graph: typed relationships among CyberCells for access paths, dependencies, policy governance, encryption protection, evidence support, memory use, incident state, and defensive-action impact.
- ZelQL: cybersecurity-native database operations including FIND, TRACE, REPLAY, VERIFY, SIMULATE, WHY, ASK, SHOW, ANCHOR, and QUARANTINE.
- Evidence-native storage: EvidenceCells and proof bundles linked to CyberCells, incidents, policies, AI decisions, human approvals, and defensive actions.
- ZelMemory: cybersecurity-governed AI memory using evidence score, trust score, freshness, tenant scope, sensitivity, poisoning risk, prompt-injection risk, training permission, recommendation eligibility, and action eligibility.
- Policy-bound retrieval and action-safe AI: separation of read, reason, recommend, simulate, execute, export, embed, and train permissions at query time.
- The Kinetic Ledger: a defensive-action provenance ledger recording requester, approver, executor, evidence basis, AI recommendation, policy decision, predicted blast radius, actual impact, risk delta, rollback state, and proof.
- The WHY Engine and Incident Replay Engine: database-native explanation and temporal replay across CyberCells, EvidenceCells, MemoryCells, PolicyCells, IncidentCells, ActionCells, and TimelineEvents.
- The Blast-Radius Query Engine: database-native simulation of compromise, containment, policy change, credential rotation, key revocation, AI-agent action, and business/security consequence modeling.
- Encryption-aware and datum-bound access: CyberCells with encryption context, key state, tenant context, datum requirement, evidence requirement, policy requirement, and purpose-bound decryption controls.
- The integrated Zelfire platform embodiment in which ZelDB serves as the memory, evidence, policy, graph, AI-governance, incident-replay, and defensive-action data layer.

A.3 Claim Boundary

This defensive disclosure does not claim that graph databases, SIEMs, vector databases, immutable ledgers, attack-path analytics, blast-radius terminology, security query languages, encryption systems, or AI memory products are new in isolation. The disclosed prior art is the integrated cybersecurity-native AI database architecture and the specific combinations and subcombinations described in this white paper.

A.4 Defensive Prior-Art Use

This document is intended to qualify as a publicly accessible technical publication when posted, archived, indexed, deposited, or otherwise made available to the public. It may be cited by patent offices, applicants, examiners, challengers, researchers, competitors, customers, or third parties as non-patent literature prior art against later-filed claims that attempt to cover the disclosed ZelDB subject matter or obvious variations of it. If any later patent application attempts to claim the disclosed subject matter, this publication is intended to be submitted where available through patent-office prior-art submission mechanisms, including third-party preissuance submission procedures.

A.5 Publication Record Checklist

Record Item	Recommended Entry
Canonical public URL	https://rocheston.com/zeldb/whitepaper or equivalent public page
Publication title	ZelDB: A Cybersecurity-Native AI Database Architecture
Reference number	ZDB-WP-2026-001
Author / inventor	Haja Mo, Rocheston
Publication date	Date of first public posting
Version	Edition 1.0 or current edition
Persistent archive	Zenodo DOI, Technical Disclosure Commons, Research Disclosure, IP.com, SSRN, GitHub release, Internet Archive, or equivalent
Integrity record	SHA-256 hash of the final published PDF stored on the landing page and internal records
Internal evidence	Screenshots of publication page, upload receipts, archive records, DOI record, web-crawl evidence, and source files

A.6 Rights Notice

Publication for prior-art purposes does not surrender Rocheston trademarks, trade names, copyright in the written work, or confidential implementation details not disclosed in the public document. No patent license, product license, software license, trademark license, or commercial license is granted by this notice. This appendix is not legal advice, a patentability opinion, a freedom-to-operate opinion, or a substitute for review by qualified patent counsel.

A.7 Short Form Notice for Public Web Page

ZelDB, Reference ZDB-WP-2026-001, is published by Rocheston as a public defensive prior-art disclosure. The publication discloses the CyberCell data model, Security Reality Graph, ZelQL, evidence-native storage, ZelMemory, policy-bound retrieval, Kinetic Ledger, WHY Engine, Incident Replay, blast-radius simulation, and encryption-aware / datum-bound access as an integrated cybersecurity-native AI database architecture for the Zelfire Suite. The document is intended to establish non-patent literature prior art against later-filed claims covering the disclosed subject matter or obvious variations.