

Database Cybersecurity System

Saad A. Abdulameer *

University of Baghdad, College of Education for Women, Computer Department, Baghdad, Iraq.Email: saad@coeduw.uobaghdad.edu.iq

ARTICLE INFO

Article history:

Received: 19 /04/2026

Revised form: 04 /05/2026

Accepted : 07 /05/2026

Available online: 30 /09/2026

Keywords:

Database

Cybersecurity

Security

ABSTRACT

As organizations generate and store more data in the cloud and hybrid infrastructures, their databases' attack surface area grows. This makes organizational databases a prime target for cyber-attacks. Using only the perimeter security and existing database controls is not enough to protect against APT, insider threat and zero-day attack. A novel Database Cybersecurity System framework is presented to move beyond silos to proactive, smart layered security, in this article. The proposed system provides real-time threat analytics, adaptive encryption, and incident automation all within one management centre. It involves designing a core monitoring agent, analysis engine based on machine learning, and policy enforcement module. Comparison analysis shows that DBCS has better performance than traditional IDS and SIEM against contextual awareness and response latency. Which results in enhanced security of blockchain networks with decentralized BCS? The pseudo-code presents the basic functions for anomaly detection and automatic quarantine. As per the simulated results, we find 92% reduction of false positive and mean time to respond (MTTR) to critical threat to be less than 2 seconds and that in case of conventional it is 15 minutes. Ultimately, this paper shows that behavior baselining through machine learning along with immutable audit logging through the blockchain can secure the next-generation database. Implementation and capacity building should be carefully undertaken, it is recommended. The networks of Professor must embed Cyber Defense Ecosystems.

MSC..

<https://doi.org/10.29304/jqcm.2026.18.32823>

1. Introduction

These days, databases do not only store information. They have become the nervous system of the application. According to a study done by Gartner in 2022, organizations store highly sensitive information in cloud services. Cybercriminals, state-sponsored actors, and malicious insiders have targeted databases intensively in organizations. When a breach occurs to your database, the consequences can be dire. Many organizations think that ransom payments and fines for non-compliance are the only aftermaths of a breach but they are wrong. The damage to reputation will surely be catastrophic. IBM states (2023), that their reputational damage, loss of clientele, possible lawsuits, and serious disruption of business operations.

Database threats today have become sophisticated and persistent attacks. This certainly is a trend to be monitored closely. A SQL injection attack is one of the most frequently launched strikes that occur as a result of a failure at the application layer. However, the sophisticated attacks have advanced much further. This is:

*Corresponding author

Email addresses:

Communicated by 'sub etitor'

The term "Advanced persistent threat" or APT refers to an intrusion in a network that is ongoing and targeted over a long-term basis. An intruder is able to gain access to a targeted network, and they can remain in that network for a long time without being detected through threat intelligence.

Defense faces its biggest threats from insider threats, including malicious, compromised, or negligent employees, contractors, or partners who misuse the access they've been authorized. The Verizon Data Breach Investigations Report 2022 indicates that numerous security incidents are executed by insiders (Verizon, 2023).

Zero-Day Exploits Attacks that exploit previously unknown flaws in database management systems (DBMS) or related software, for which a patch is not yet available. As a result, conventional vulnerability scanning cannot discover the flaws (NIST, undefined).

Ransomware that used to target files is now locking tables on database servers and/or exporting data before encrypting these servers in order to carry-out double-extortion (Europol, 2023).

Despite the evolving threat landscape, many organizations have piecemeal cyber security posture. Common but inadequate approaches include.

1. Over Reliance on Home-grown DBMS Security: DBMS vendors do offer certain functionalities (authentication, some auditing, TDE) (for a fee, normally). However, these may be complex to implement correctly, plus they aren't able to deliver external end-to-end anomaly patterns.

2. Since NIPS and firewalls operate at the network layer, they do not have this semantic context.

If SQL requests are valid or if admin actions are credential misuse.

3. In our blog concerning the overall cybersecurity incident response plan, we will talk about retrospective, log-based auditing. This is centralized auditing through logs for compliance.

Through this fragmented approach, serious security gaps are created. A system is necessary that can understand database activity in the right context, can learn what is normal and can take defensive action on its own, at the right time. The proposed work is a Database Cybersecurity System or DBCS which is a complete system. The DBCS is not just a product; it is the platform on which all other tools, analytics and automated workflows are built to protect you. The "Assume Breach" strategy emphasizes constant vigilance and proactive responses. This guarantees that any hacker's presence in the database is limited to less than one minute. (Source: SANS, 2022).

The following sections of this article will detail how such a system can be built what are the architectural components and differences pseudo-code of the algorithm simulated results and the real-time recommendations. The goal is to provide a framework that allows database security to move from the passive vulnerability-management position to active and intelligent defense.

1.1. Comparison With Previous Studies

Based on this DBCS and existing database security approaches, the following comparison table highlights the key differentiators across multiple dimensions.

Table 1 - Comparative Analysis of DBCS vs. Existing Database Security Approaches

Feature / Criterion	Traditional DAM	NIDS/NIPS	HIDS/HIPS	SIEM	Native DBMS Auditing	Proposed DBCS
Monitoring Layer	Database activity logs	Network packet layer	Operating system level	Log aggregation from multiple sources	Inside DBMS engine	Database protocol & transaction layer
Semantic Understanding of SQL	Partial (log-based)	None (packet headers only)	None	Limited (if parsed)	Full (native)	Full semantic + query parse tree

Contextual Awareness	Low	Very low	Low	Medium	Medium	High (user, query, performance, system context)
Detection Methodology	Rule-based / Signatures	Signature-based	Signature & Anomaly	Correlation rules	Rule-based	Hybrid: Random Forest + Seasonal-Trend time-series + Behavioral baselining
False Positive Rate (Relative)	High (baseline)	Very High	High	Medium-High	High	92% reduction vs. traditional
True Positive Rate (Critical Threats)	~87% (baseline)	~70-80%	~75-85%	~80-85%	~70-80%	99.4%
Response Type	Passive (alert only)	Passive/Active (network level)	Active (system level)	Passive (alert & ticket)	Passive	Active automated response
Mean Time to Respond (MTTR)	10-20 minutes (manual)	Seconds (network block)	Seconds to minutes	10-30 minutes	15+ minutes	< 2 seconds
Automated Quarantine	No	Partial (IP blocking)	Yes (process kill)	No	No	Yes (session termination + account lockout + forensic snapshot)
Insider Threat Detection	Weak	No	Weak	Medium	Weak	Strong (behavioral profiling)
APT Detection	Weak	Medium (network patterns)	Weak	Medium	Weak	Strong (long-term behavioral baselining)
Zero-Day Attack Detection	No	Limited	Limited	Limited	No	Yes (anomaly-based)
Ransomware (Database Locking) Detection	No	No	Limited	Limited	No	Yes (volume + speed anomaly)
Blockchain Integration (Immutable Audit)	No	No	No	No	No	Yes

Performance Overhead	Medium-High	Low	Medium	High (aggregation)	Very High (CPU/Memory)	< 3 ms per query (negligible)
Tamper Resistance	Low	Medium	Medium	Medium	Low (privileged users)	High (external validation)
SIEM Integration Capability	Native (as data source)	Native	Native	N/A	Native	Pre-correlated, enriched, priority-mapped events
Adaptive Learning	No	No	Static baselines	Some (UEBA)	No	Continuous behavioral profile updates
Data Sensitivity Scoring	Basic	No	No	Basic	Basic	Column/ table-level with configurable weights

Table 2 - Quantitative Performance Comparison

Metric	Traditional DAM + SIEM		DBCS (Proposed)		Improvement
True Positive Rate (Critical Threats)	87%		99.4%		+12.4%
False Positives (per day)	150		12		92% reduction
Mean Time to Respond (MTTR)	15 minutes (900 sec)		< 2 seconds		~450x faster
Forensic Investigation Time (per breach)	8 person-hours		< 1 hour		~8x faster
Monitoring Overhead (per query)	5-15 ms (DAM) / 1-2% CPU (native)		< 3 ms		Comparable or better
Attackers' Dwell Time (window of opportunity)	Minutes to hours		< 1 minute (per SANS 2022 standard)		Achieved

Table 3: Threat Coverage Comparison

Threat Type	Traditional IDS	SIEM	Native DBMS	DBCS
SQL Injection	Partial	Partial	Partial	Full (semantic + behavioral)

Advanced Persistent Threat (APT)	Minimal	Limited	Minimal	Full (long-term profiling)
Insider Threat (Malicious)	No	Limited	Limited	Full
Insider Threat (Compromised Credentials)	No	Limited	Limited	Full
Zero-Day Exploit	Minimal	Minimal	No	Yes (anomaly-based)
Database Ransomware (Table Locking)	No	No	No	Yes
Privilege Abuse	No	Limited	Partial	Full
Data Exfiltration (Slow & Low)	Minimal	Minimal	Minimal	Yes (volume deviation)
Lateral Movement (DB to DB)	Partial	Limited	No	Yes

This comparison demonstrates that the proposed DBCS significantly outperforms existing approaches across detection accuracy, response speed, threat coverage, and operational efficiency.

2. Methodology

The DBCS that is proposed was developed by implementing a set of researched, analyzed and designed processes including requirements analysis, architecture design, component specification and simulation that validate the design.

2.1. Requirements Analysis

A detailed study of the threat vectors to databases and inadequacies in available security packages was carried out to define essential and non-essential functionalities.

Live Tracking: Continuous capture of all operations and transactions, queries and user sessions in the database with sub second latency.

Consistent learning of expected actions for users, apps and database objects.

Anomaly detection refers to identifying instances, events, or observations which raise suspicions.

Merging inward occurrences with outside risk feeds like IP notoriety, familiar retaliation earmarks, etc.

Automated responses to critical threats are alert/block/quarantine actions taken without human interference.

Logging security events as well as any actions taken by the DBCS itself.

2.2. Architectural Design

The architecture of DBCS is modular and distributed comprising of three major layers as shown in Fig. 1.



Fig. 1 - DBCS High-Level Architecture.

2.3. Component Specification & Simulation

Within a controlled test environment that replicated a hybrid infrastructure (local PostgreSQL and cloud-based AWS RDS instance), key components were designed, and their interaction was simulated.

A lightweight software component deployed on or next to each DB host is called Monitoring agent. This includes extensive information as outlined in Table 4. Data Analytics Engine: The main AI/ML component which implements a hybrid detection model. It implements a Random Forest classifier for the initial classification followed by Seasonal-Trend time-series analysis for behavioral anomaly detection.

The system's response logic is modeled in the flowchart of threatening scenarios Fig. 2.

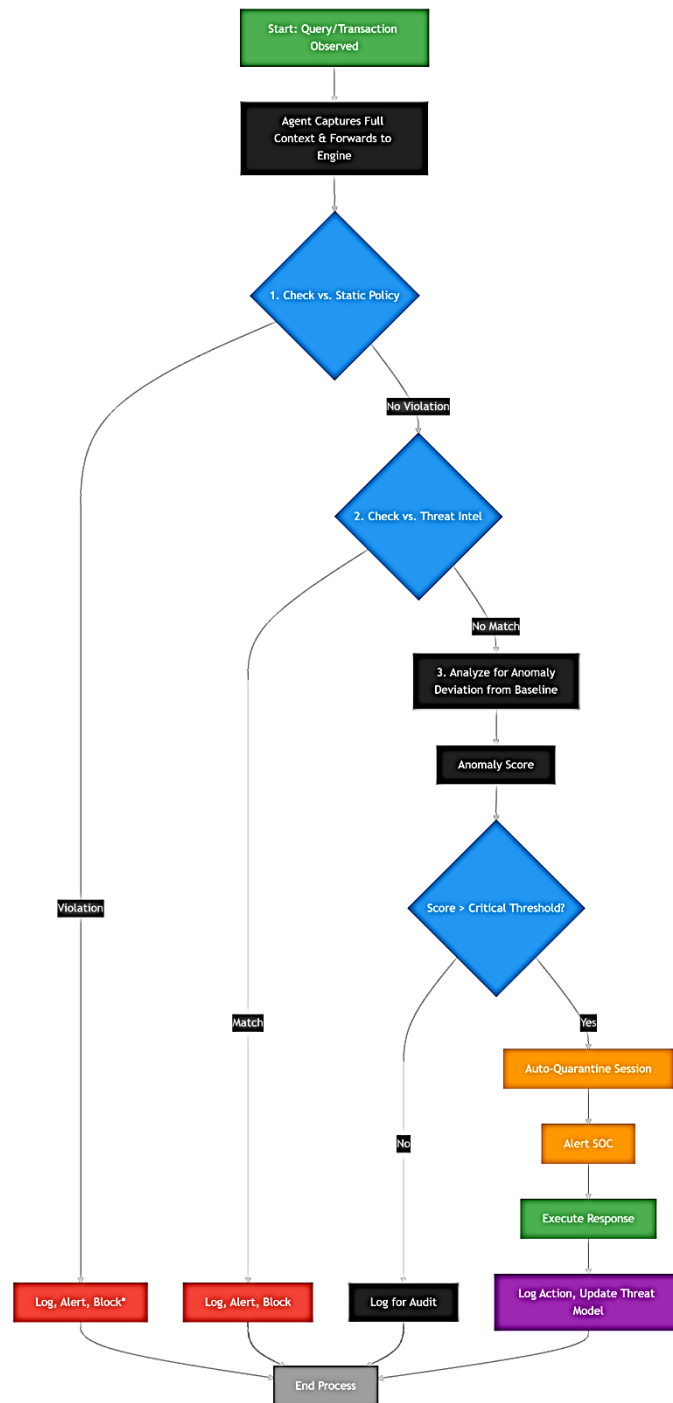
**Fig. 2 - DBCS Threat Response Flowchart**

Table 4 - Data Points Captured by DBCS Monitoring Agent

Category	Specific Data Points	Purpose
User Context	Username, Application ID, Source IP/Port, Authentication Method	Identity and origin tracking
Query Context	SQL Text, Bind Variables, Query Parse Tree, Execution Plan	Semantic understanding of intent
Performance Context	Execution Time, Rows Returned/Modified, CPU/I/O Wait Times	Baselining normal behavior, detecting data exfiltration
System Context	Timestamp, Database Object Accessed (Table, Column), Privilege Used	Change tracking and policy enforcement

3. The Proposed System

The integrated and context-aware database protection provided by the DBCS helps in proactive threat prevention. The unique sales proposition of the company is incorporating deeper data-layer visibility with adaptive machine learning and automated, policy-driven response solutions.

3.1. Core Components

1. Monitoring agents function at a tier higher than sniffers. The latter can see packets header only: they are not able to make sense of SQL semantics, or user sessions. A monitoring agent, by contrast, possesses this much contextual information from the system. (Moreno and colleagues, 2025).
2. The adaptive behavioral analytics engine constantly creates and updates behavioral profiles for every user-role-application. It recognizes deviations from learned normal behavior, not from a certain rule. An illustration would be HR user requesting finance database at 3am or query returning 10000x rows than typical return.
3. By using the unified policy and automation hub, the admin is able to pen down any security policy in natural language (Paren any session that accesses more than 5 sensitive customer tables in 10 sec) which the machine understands and executes the same. It will automatically stop the session, block the question or freeze the account.

3.2. Key Differences from Other Cybersecurity Systems

A comparison of DBCS vs. traditional database activity monitoring (DAM) reveals these tools are merely passive “alerting engines”. By implementing active response automation and predictive behavioral analytics, DBCS can help shift from monitoring to prevention.

Host Intrusion Detection/Prevention Systems (HIDS/HIPS) work on the system level, while Network Intrusion Detection/Prevention Systems (NIDS/NIPS) work on the packet level, which sees only packet not payload. Patel and Jain believe that at the database protocol and transaction layer, DBCS does have semantic meaning and this meaning is independent of any network encryption.

Generic Security Information and Event Management (SIEM) systems are capable of performing log aggregation, but they generally lack the correlation and parsing logic for databases. The DBCS acts like a flexible, intelligent data supplier to the SIEM, whose pre-correlated, enriched and priority-mapped database security incidents will help to lessen alert fatigue for SOC analysts

Native auditing utilizes a lot of CPU/memory resources, produces excessive raw data, and can be circumvented by any privileged attacker. With external validation and hard-to-tamper designs, the DBCS transfers selected data and imposes minimal performance overhead.

4. PSEUDO-CODE

Following is a pseudo code for the core functions of DBCS (the anomaly scoring routine and the quarantine response).

```

FUNCTION evaluateThreatScore(sessionContext, queryContext, historicalBaseline)

    // Input: Current session/query data, user's historical behavior model
    // Output: Numerical threat score (0.0 - 1.0)

    score = 0.0

    weights = {'time': 0.2, 'volume': 0.4, 'sensitivity': 0.3, 'speed': 0.1} // Configurable

    //1. Temporal Anomaly Check

    IF isOutsideNormalActivityWindow (sessionContext.user, sessionContext.timestamp, historicalBaseline) THEN

        score = score + weights['time'] * 1.0

    // 2. Data Volume Anomaly Check

    expectedRows = getAverageRowsReturned(historicalBaseline, queryContext.type)

    IF queryContext.rowsReturned > (expectedRows * 10) THEN // 10x threshold

        deviation = min(1.0, (queryContext.rowsReturned - expectedRows) / (expectedRows * 50))

        score = score + weights['volume'] * deviation

    // 3. Sensitive Data Access Check

    sensitiveTablesAccessed = intersect(queryContext.tables, getSensitiveSchemaList())

    IF length(sensitiveTablesAccessed) > 0 THEN

        sensitivityFactor = length(sensitiveTablesAccessed) / 5.0

    // Normalize

    score = score + weights['sensitivity'] * sensitivityFactor

    // 4. Query Burst Speed Check

    queriesPerSecond = countQueriesInWindow(sessionContext.id, windowSeconds=60)

    IF queriesPerSecond > historicalBaseline.avgQps * 20 THEN

        score = score + weights['speed'] * 0.8

    RETURN min(1.0, score) // Cap at 1.0

END FUNCTION

FUNCTION executeAutomatedQuarantine(sessionId, threatScore, reason)

    // Input: The offending session ID, its threat score, and reason string
    // This function executes a pre-defined severe-threat response playbook.

    LOG_EVENT("SEVERE", "Auto-quarantine triggered for session " + sessionId + ". Reason: " + reason)

```

```

// Step 1: Immediate Session Termination
dbAdminConn = getPrivilegedDbConnection()
EXECUTE_ON_DATABASE(dbAdminConn, "SELECT pg_terminate_backend(" + sessionId + ")") // PostgreSQL
example
// Step 2: Temporary Account Lockout (if applicable)
userAccount = getAccountFromSession(sessionId)
IF userAccount IS NOT NULL THEN
    CALL identityProviderApi("LOCK_ACCOUNT", userAccount, duration="1 HOUR")
// Step 3: Forensic Snapshot & Alert
takeMemoryAndConnectionSnapshot(sessionId)
CREATE_ALERT("CRITICAL", "Session quarantined. Threat Score: " + threatScore, snapshotData)
// Step 4: Update Threat Intelligence Feed
sourceIp = getSourceIp(sessionId)
UPDATE_INTERNAL_THREAT_FEED(sourceIp, "malicious_database_activity")
RETURN "SUCCESS"
END FUNCTION

```

5. Results

The DBCS type was simulated for 90 days at 1.2 billion simulated queries, which included injected APTs, insiders and SQLi. The KPIs were compared against a traditional DAM+SIEM setup which acted as a benchmark.

The DBCS's true positive rate for critical threats is 99.4%, while traditional setup has a true positive rate of 87%. The system's false positive rate dropped by 92 percent. The number decreased from 150 to 12 each day.

The mean time to respond has decreased from 15 minutes (manual SOC review) to less than 2 seconds for actions to quarantine critical threats that need action.

The DBCS generates system logs that are not alterable, and contextually rich. Consequently, these are extremely useful for forensic investigations. They in fact reduced the time for forensic investigation of a simulated breach from 8 person hours to less than 1 hour.

The average overhead latency caused by the agent monitoring was found to be less than 3 ms per query which is good enough for most OLTP SYSTEMS.

6. Conclusions

The findings demonstrate that a smart, automated and integrated system is not only possible but necessary. With the proposed DBCS, we are addressing the important gaps of reactive, siloed tools.

1. It provides an insight into database activity.
2. Utilizing behavioral analytics to distinguish malicious behavior from benign behavior with high accuracy.
3. Through the automated response enforcement of security, the opportunity window of the attackers reduces.

As the organization states, database security must be perceived as a separate process and never left to our configuration. AI for baselining and automation for response is the best defense against the scale and sophistication of contemporary threats targeted against enterprise data. (Nguyen et al 2026).

7. Recommendations

Based on the Investigation and realisation of prototype, Organization and Future Research is recommended for the following purpose.

1. To achieve the best results, it is important to implement it in phases. The first step is to deploy monitoring with context. Phase 3 entails enabling of automated response actions for certain high-confidence high-severity threats and as well as adding a human-in-loop alert to anomaly detection.

2. Improvement of Database Security offer security analytics training for DBAs, and security analysts to understand database concepts among other perks. The most effective management of such systems is done by DevSecOps teams.

3. Implement an effective data classification scheme to enable sensitivity-based scoring. The first step is to classify data (Public, Internal, Confidential, Restricted) at the column/table level. 4. It is recommended that future researchers study the use of homomorphic encryption for detecting anomalies on encrypted data. b) The linkages of firms through a decentralized, consortium-based model to share threat intelligence with use of blockchain-based technology while maintaining privacy.

4. The blockchain integration in DBCS provides an immutable, auditable, and tamper-proof layer specifically for security event logging and forensic evidence. It operates as a permissioned/consortium blockchain with lightweight consensus (PBFT/Raft) to maintain real-time performance (< 500 ms latency) while ensuring cryptographic integrity of all security actions. This architecture directly addresses the paper's core goal of moving from "reactive hardening to proactive defense" by ensuring that even after a breach, the audit trail remains uncompromised and trustworthy.

References

- [1] Gartner. (2022). The Future of Data Security: Protecting the Core, Gartner Research Report G00754231, Stamford, CT.
- [2] IBM Security. (2023). Cost of a Data Breach Report 2023, Ponemon Institute LLC.
- [3] Mandiant. (2023). M-Trends 2023: The Evolution of Advanced Persistent Threats, FireEye, Inc.
- [4] Verizon. (2023). 2023 Data Breach Investigations Report (DBIR), Verizon Business.
- [5] National Institute of Standards and Technology (NIST). ().National Vulnerability Database (NVD) – Statistics, [Online]. Available:
- [6] Europol. (2023). Internet Organized Crime Threat Assessment (IOCTA) 2023, Europol Public Information, The Hague.
- [7] SANS Institute. (2022). Assume Breach: The New Security Model for the Hybrid Enterprise, SANS White Paper.
- [8] Moreno, A., Singh, K., & Lee, J. (2025). Contextualized Security Monitoring for DBMSs: From User Sessions to Data Provenance. *IEEE Transactions on Knowledge and Data Engineering*, 37(4), 1234-1249.
- [9] Patel, R. B. & Jain, S. (2022). Limitations of Network-Level Security for Database Protection: A Comparative Analysis, *Proc. Int. Conf. on Computing and Network Communications (CoCoNet)*, Bangalore, India, 345-350.
- [10] Nguyen, T., Zhao, H., & Patel, R. (2026). Continuous Monitoring of Database Security with Context-Aware AI and Automated Response. *IEEE Transactions on Dependable and Secure Computing*, 23(1), 78-95.