Bayesian Optimization with SAT Solver for Enhanced Database Security Identification

Huijuan Zhang*
School of Information Engineering, Henan Vocational College of Water Conservancy and Environment
Zhengzhou 450008, China
E-mail: juanjuan5606@163.com
*Corresponding author

Keywords: Database security, SAT solver, Bayesian optimization, gaussian process regression

Received: January 12, 2026

To solve the limitations of traditional database security detection, such as weak logical reasoning, delayed feature update, and high computational complexity under high-dimensional constraints, this study proposes a Bayesian optimization algorithm based on SAT. Security rules are formalized into CNF logical constraints, enabling automated reasoning and rule consistency verification via a satisfiability solver. A Gaussian process-driven Bayesian optimization framework with an improved Expected Improvement (EI) acquisition function dynamically updates feature weights and accelerates convergence toward high-risk regions, enhancing rare vulnerability detection and global solving efficiency. Experiments on enterprise database logs and national vulnerability datasets demonstrate an average detection rate of 97.5%, a defense success rate of 95.8%, and a response latency of 2.11 s, outperforming baseline methods. The system stability index reaches 0.93, with concurrent processing capability improved by 34.2%. These results confirm the algorithm's high-precision identification, stable defense performance, and practical deployability for database security management under complex, high-dimensional attack scenarios.

Povzetek: Članek predlaga SAT-podprto Bayesovsko optimizacijo za varnostno zaznavanje v bazah podatkov, kjer se pravila formalizirajo v CNF za avtomatsko logično sklepanje, Gaussian-process BO z izboljšanim EI pa dinamično posodablja uteži značilk in hitreje najde redke ranljivosti v visokodimenzionalnih scenarijih.

1 Introduction

With the advancement of digital transformation, databases have become a critical infrastructure in government, financial, and healthcare systems, making their security essential for data protection and business continuity. However, modern database threats are increasingly intelligent, covert, and diverse. Attacks such as SQL injection, unauthorized access, malicious updates, and sensitive data leakage pose significant challenges to traditional security mechanisms [1–2]. Existing database security detection methods mainly rely on static rule matching and heuristic feature-based approaches. These methods often suffer from limited logical reasoning capability, delayed feature updates, and high computational complexity under high-dimensional constraints, making real-time and adaptive defense difficult to achieve [3–4]. SAT solvers have been applied in security verification due to their strong logical reasoning and constraint-solving abilities [5–6]. Nevertheless, Database Security Identification (DSI) requires not only logical consistency but also dynamic optimization and rapid adaptation under imbalanced and evolving attack scenarios [7–8].

In response to the above problems, industry scholars have conducted in-depth research on the DSI method. To address the issue of privacy protection for encrypted database queries, Xie et al. proposed an access pattern hidden search method based on distributed point functions. This method could improve query efficiency while protecting database privacy and preventing external attacks and data leaks, and was suitable for database environments with high security requirements [9]. To effectively detect malicious transactions in the database, Singh and Jindal proposed a user behavior analysis method based on trust factors. Through sequence pattern mining technology, this method could identify potential intrusion behaviors, reduce database security risks, and improve the robustness [10]. To solve the security problems in campus networks, Wang and Long proposed a protection system framework based on cloud data and intrusion detection algorithms. This framework combined cloud computing and intrusion detection technology to provide real-time threat monitoring and defense in terms of database security [11]. To improve the security management level of shared databases, Deng et al. proposed a blockchain-based security access control system. It used the transparency and non-tamperability of the blockchain to enhance the access control and data security of the database, and was suitable for database scenarios that require high security protection [12].

Table 1: Comparison of this article with related literature

Work	Main Focus	Core Method	Limitation (Gap)
[9]	Encrypted query privacy	Access-pattern hiding search	No logical rule consistency verification
[10]	Malicious transaction detection	Trust-based sequential mining	Limited Boolean constraint modeling
[11]	Network security framework	Cloud-based IDS architecture	Lacks formal rule reduction and solver reasoning
[12]	Secure access control	Blockchain-based mechanism	No high-dimensional optimization or adaptive weighting
This work	Database security identification	SAT reduction + Bayesian optimization	/

To address these limitations, this study proposes a Bayesian-Driven SAT-based Security Identification (BD-SAT-Sec) algorithm. The method formalizes security rules into Conjunctive Normal Form (CNF) and employs a SAT solver for automated logical reasoning. By integrating Bayesian optimization, the framework enables dynamic feature weighting and accelerated search in high-risk regions, achieving accurate security event detection and real-time defense optimization in complex database environments. The novelty of this work lies in establishing a unified logical–probabilistic framework rather than simply combining two techniques. While SAT solvers ensure formal logical consistency of security rules, they lack adaptive learning capability. Conversely, Bayesian optimization provides uncertainty-guided adaptation but does not incorporate formal rule verification. By embedding Bayesian-driven weight updating into SAT branching and variable prioritization, the proposed BD-SAT-Sec enables both CNF-level logical validation and adaptive search toward high-risk, long-tail attack regions. This bidirectional integration advances database security research from heuristic detection toward a verifiable and dynamically optimized security identification paradigm.

This study proposes a unified DSI framework that combines SAT-based formal verification with Bayesian adaptive optimization. It investigates whether CNF modeling ensures rule consistency while Bayesian weight updating improves rare-vulnerability detection and Scene Coverage (SC) without sacrificing efficiency. The study aims to enhance conflict detection, robustness under long-tail attacks, and runtime stability in enterprise deployment, ultimately delivering an interpretable and deployable logical–probabilistic security mechanism. Table 1 shows a comparison between this article and related literature.

2 Methods

2.1 DSI model construction and SAT question specification

To achieve intelligent identification of database security events and optimization of vulnerability defense, this study systematically models the security identification process of the database and formalizes the security detection task into a SAT problem-solving process [13-15]. In the process of rule formalization, hierarchical modeling of the granularity of security rules is studied. For coarse-grained rules (such as access permission level judgment), direct variable mapping is used. For fine-grained rules (such as cross-table dependencies or transaction-level operations), clause clusters are generated through Abstract Syntax Tree (AST) structure splitting and mapped to Boolean variable sets, respectively. Rule granularity directly affects the number of CNF clauses and variable coupling degree, thereby increasing the complexity of SAT search and affecting the convergence trajectory of subsequent Bayesian optimization. The finer the granularity, the higher the degree of intersection between Boolean encoding scales and constraints [16]. The core of DSI is to validate the input transaction flow through a set of rules to distinguish safe traffic from potential attack traffic. The overall process of the system is shown in Figure 1.

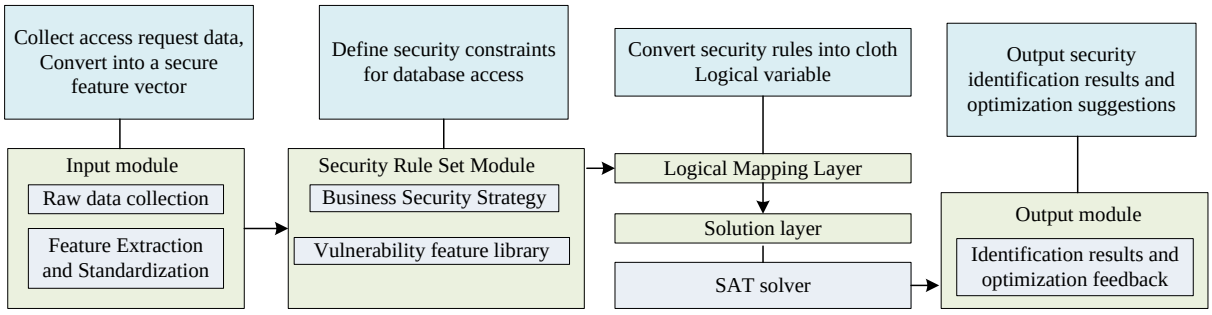


Figure 1: Framework of DSI based on SAT mapping

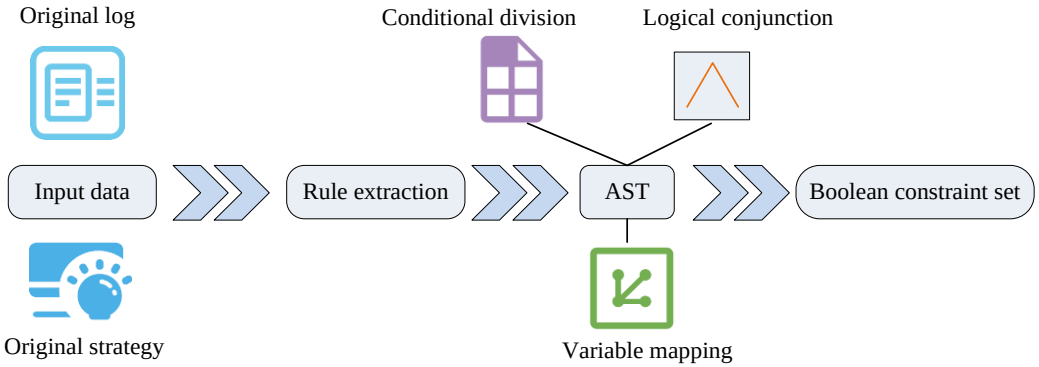


Figure 2: SAT-based formalization process of security recognition rules

In Figure 1, the input module is responsible for collecting and extracting transaction or access request data and converting it into structured security feature vectors. The security rule set module defines security constraints based on business logic and access policies. The logic mapping layer converts rule conditions into Boolean variable expressions, and implements logic verification and safety determination through the SAT solver in the solution layer. Finally, the output module generates recognition results and feeds back optimization signals for updating safety feature weights. In this system, safety rule R can be abstracted into a conditional logic expression, as shown in equation (1) [17].

$$R_i : P_i \to Q_i \tag{1}$$

In equation (1), P_i is the precondition (trigger condition, such as access user level, data sensitivity level, etc.). Q_i is the security verification condition (such as access permission, request scope or encryption status). Rule base $R = \{R_1, R_2, \cdots, R_m\}$ constitutes a complete safety constraint system. To achieve an automated solution, the security identification rule set is encoded as a SAT model. First, the set of Boolean variables is defined as $V = \{x_1, x_2, \ldots, x_n\}$. Among them, each variable x_i corresponds to a security attribute or condition (such as user type, request method, data field integrity, etc.). Each rule R_i is converted into a Boolean clause $C_i = (l_1 \vee l_2 \vee \cdots \vee l_k)$. Among them, l is text. The conjunction of multiple clauses forms a complete CNF form, as shown in equation (2). In the CNF generation stage, clause pre-screening and variable resolution strategies are used to prune redundant clauses. The preprocessing stage reduces invalid clauses by about 18% on average, significantly shrinking the search space. This stage is independent of the Bayesian optimization module and makes a fundamental contribution to the overall convergence acceleration.

$$F(V,C) = \bigwedge_{i=1}^{m} C_i \tag{2}$$

In equation (2), V represents the set of Boolean variables, and C represents the set of clauses. F is the entire SAT security identification model, and m is the logical constraints contained in it. The solution goal is to find a set of truth value assignments $S = \{x_i \mid x_i = \text{True or False}\}$ such that $F(V,C) = \text{True}$. In DSI, the solution that satisfies $F(V,C) = \text{True}$ corresponds to a safe behavior. If the solution fails, it means there is a logical conflict or potential security vulnerability. The system modeling process is shown in Figure 2.

In Figure 2, the rule extraction module first converts the original SQL access logs and security policies into an AST. Subsequently, through conditional partitioning, variable mapping, and logical conjunction operations, a standardized set of Boolean constraints is generated. Reducing invalid variables through preprocessing steps can significantly reduce the solution space and improve the solution efficiency [18]. Since database security events are highly imbalanced, the SC metric is introduced to evaluate model generalization across heterogeneous attack scenarios. Since SC is influenced by feature dimensionality and CNF clause density, a scale normalization mechanism is incorporated. Specifically, the raw coverage ratio is adjusted using a feature-based normalization term and a clause-density correction factor, mitigating bias from varying feature scales and constraint complexities. The normalized SC formulation is presented in equation (3).

$$f(S) = SC_S = \frac{Count(T \mid T.p \subseteq S.p)}{Count(T)} \tag{3}$$

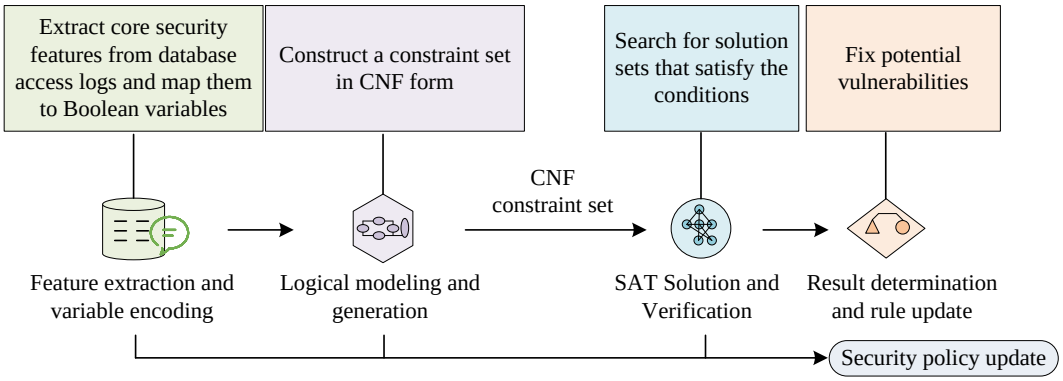


Figure 3: Process of SAT-based database security problem reduction and solving

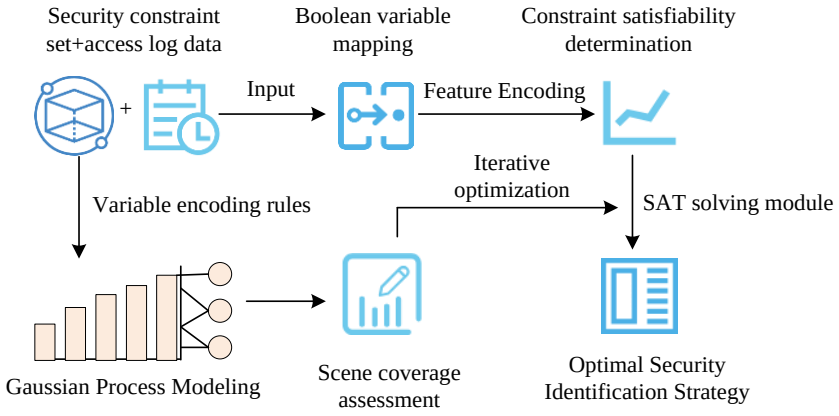


Figure 4: Running flow diagram of the BD-SAT-Sec algorithm

In equation (3), $f(S)$ is the SC function. SC_S is the SC index. T is a collection of security event samples. $T.p$ is the set of characteristic attributes of the sample. $S.p$ is the set of characteristic attributes of the solution results. $Count(T)$ is the number of samples of all security events. $\text{Count}(T \mid T.p \subseteq S.p)$ is the number of events covered by solution S. In the model optimization stage, a weighted objective function is defined by introducing the safety constraint vector $\mathbf{w} = (w_1, w_2, \cdots, w_n)$, as shown in equation (4).

$$\max f(S, \mathbf{w}) = \sum_{i=1}^{n} w_i \cdot x_i \tag{4}$$

In equation (4), $f(S, \mathbf{w})$ is the security recognition goodness function, and n is the total number of features. Through SAT solving, a unified framework from rule conflict detection to feature interactive verification can be realized. The solution process is shown in Figure 3.

Figure 3 presents four stages: feature extraction and encoding, CNF constraint construction, SAT solving, and result update. Access-log features are mapped to Boolean variables, and security policies are formalized as CNF constraints. The SAT solver searches for feasible assignments, while unsatisfied clauses trigger policy refinement. To handle scale and complexity, small CNF instances are solved using DPLL for exact verification. Large instances use local-search heuristics combined with BD-SAT-Sec for global optimization. This process converts complex security rules into computable SAT problems, enabling efficient and structured security identification.

2.2 BD-SAT-Sec algorithm design

After completing the construction of the DSI model and SAT problem specification, to further improve the solution efficiency and security feature identification capabilities, this study designs the BD-SAT-Sec. This algorithm deeply integrates the SAT logical solution framework and the BO mechanism, and achieves adaptive optimization of security identification rules by dynamically adjusting feature weights and sampling strategies during the search process. The overall operation process of the algorithm is shown in Figure 4.

In Figure 4, the input module receives the security constraints and access log data of the database, and after being mapped into a Boolean variable form by the feature encoding module, it enters the SAT solver. The BO engine dynamically updates the feature weight vector $\mathbf{w}$ based on the historical search results to guide the solution direction to converge toward a solution area with higher safety coverage. The optimization module evaluates the candidate solutions output from the SAT solution, calculates its SC according to the objective function $f(S, \mathbf{w})$, and updates the prior distribution. Finally, the algorithm outputs the optimal security identification strategy to achieve interpretable and reversible database security optimization. The core of the BD-SAT-Sec algorithm lies in the construction of a

three-layer collaborative mechanism. The Gaussian process based on the RBF kernel is used to estimate the objective function distribution and uncertainty of the candidate solution. The Expected Improvement (EI) acquisition function selects the sampling direction based on the mean-variance trade-off strategy. The Boolean variable weight update mechanism maps the posterior gradient to the feature importance space. The three form a structure of "agent prediction - sampling decision - weight feedback". Among them, EI mainly affects the stability of SC fluctuations, GPR affects the convergence speed and global exploration scope, and the variable weight mechanism mainly determines the detection rate improvement and rare vulnerability identification capabilities. Assuming that the objective function is $f(S,\mathbf{w})$, its prior distribution can be modeled as a Gaussian process, as shown in equation (5).

$$f(S,\mathbf{w}) \sim \mathrm{GP}(m(S),k(S,S')) \tag{5}$$

In equation (5), $m(S)$ is the mean function, reflecting the expected value of the objective function. $k(S,S')$ is the covariance function, which is used to measure the correlation between different solution points. The definition of radial basis kernel function is shown in equation (6).

$$k(S,S') = \sigma_f^2 \exp\left(-\frac{\| S-S' \|^2}{2l^2}\right) \tag{6}$$

In equation (6), σ_f^2 is the signal variance. l' is the kernel width parameter, which controls the smoothness between sampling points. The surrogate model achieves dynamic approximation of the SAT search area by continuously updating the posterior distribution. At the same time, to avoid excessive biasing of the search trajectory by the RBF kernel parameters, adaptive scaling of the kernel width parameters under different instance sizes is studied, and the sampling variance is normalized through the posterior uncertainty calibration mechanism. When the kernel width is small, the model tends to converge locally; When the kernel width is large, the global exploration capability is enhanced. Therefore, the consistency of the sampling paths under different constraint scales is maintained through the scale normalization strategy. The collaboration diagram of BO and SAT solvers is shown in Figure 5.

In Figure 5, the initial sampling point set is generated by random or heuristic strategies and sent to the SAT solver to obtain the corresponding security identification performance value. The BO engine updates the proxy model parameters based on these observed data. The acquisition function calculates the exploration value of potential points and selects the next sampling point. The point is fed into the SAT solver for verification.

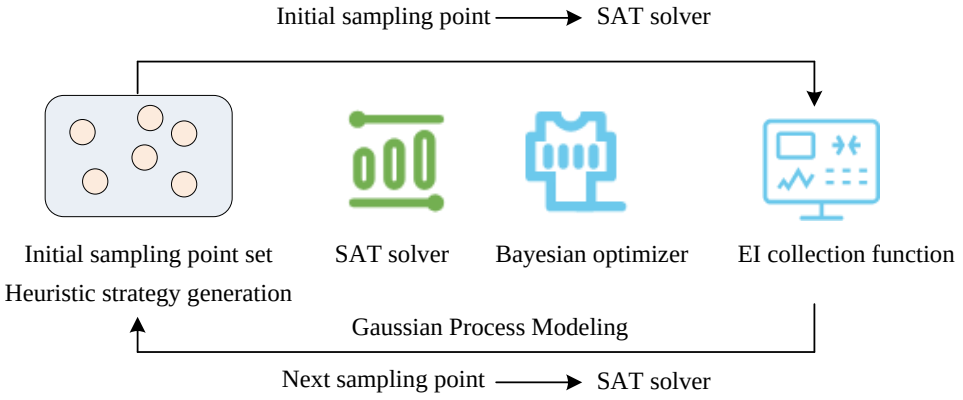


Figure 5: Collaborative diagram of BO and SAT solver

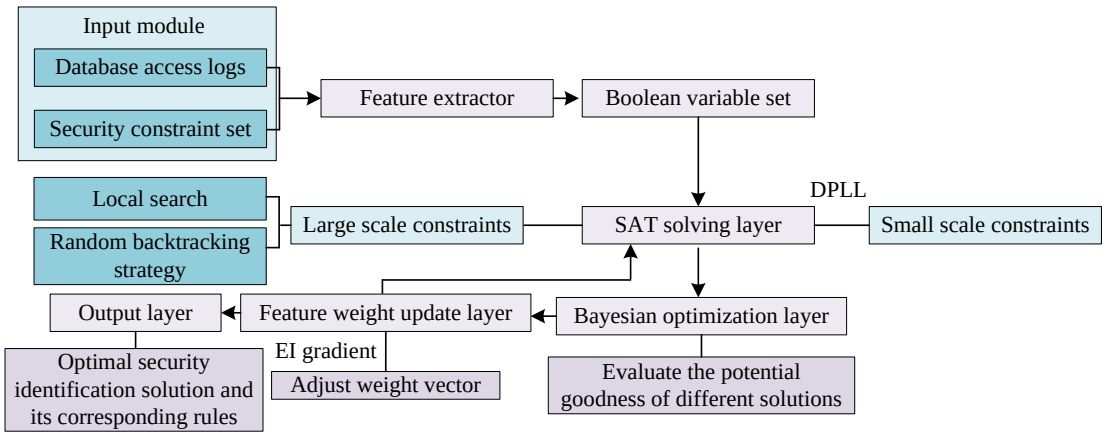


Figure 6: Structural diagram of the BD-SAT-Sec algorithm

To balance global search and local refinement in the SAT solution process, BD-SAT-Sec uses an improved EI acquisition function. In the t -th round of iteration, assuming that the current optimal target value is f^*, the expression of EI is as shown in equation (7).

$$\mathrm{EI}(S) = \mathrm{E}[\max(0, f(S) - f^*)] = (m(S) - f^*)\Phi(Z) + \sigma(S)\phi(Z) \quad (7)$$

In equation (7), Φ and ϕ are the cumulative distribution function and probability density function of the standard normal distribution, and Z is the prediction variance. Combined with the BO results, BD-SAT-Sec dynamically adjusts the weight coefficients of SAT variables to strengthen important features. The weight update rule is defined as shown in equation (8).

$$w_i^{(t+1)} = w_i^{(t)} + \eta \cdot \frac{\partial \mathrm{EI}(S)}{\partial x_i} \quad (8)$$

In equation (8), η is the learning rate parameter, which is used to control the weight update amplitude. To enhance the transparency of feature evolution, the feature weight change sequence is recorded in each iteration, and the weight dominance index is calculated, which is used to characterize the dominance degree of different security features in the search cycle. In experiments, it is found that the authority verification feature dominates the search direction in the early stage. The weight of cross-table dependent features increases significantly in the middle and later stages, which is positively correlated with the improvement in rare attack identification. The overall structure is shown in Figure 6.

Figure 6 illustrates the BD-SAT-Sec architecture. The input module converts access logs and security constraints into Boolean variables via feature extraction. The SAT layer applies a dual strategy: DPLL for small-scale CNF instances and local search with random backtracking for large-scale constraints. The BO layer evaluates candidate solutions using a surrogate model and selects the next sampling point via the acquisition function. Feature weights are updated based on the EI gradient and fed back to guide SAT solving. The output layer produces the optimal security identification solution and corresponding rules. By integrating Bayesian probabilistic modeling with SAT-based logical reasoning, BD-SAT-Sec achieves efficient global optimization for DSI. The BD-SAT-Sec pseudo-code is shown in Table 2.

To ensure reproducibility, the model uses a fixed feature schema covering identity, network, behavior, query, data, logical, and optimization attributes. Boolean features are directly mapped to SAT variables. Categorical features use one-hot encoding. Numerical features are discretized into three equal-frequency bins before Boolean mapping. Assuming the encoded feature set is $F = \{f_1, f_2, \cdots f_d\}$ with total dimension $d = 4200$, where each feature corresponds to a SAT variable. The system includes identity (e.g., user level), network (IP credibility), behavior (abnormal frequency), query (SQL injection flags, AST anomalies), data (sensitive access, modification), logical (clause satisfaction, rule conflict), and optimization features (SC, weight coefficients). All features are normalized before Bayesian optimization.

Examples of rule-to-CNF conversion are as follows: Example 1 (Privilege Exceeding Boundary Rule): The original rule is that requests by low-privileged users to access sensitive fields must be blocked. Assuming A = Low-privilege user, B = Sensitive field access, and C = Request allowed. Logical expression: $(A \wedge B) \rightarrow \neg C$, CNF form: $(\neg A \vee \neg B \vee \neg C)$. Example 2 (SQL Injection Rule): The original rule is that detection of SQL injection must trigger defense. Assuming: D = SQL injection detection; E = Defense triggered. CNF: $D \rightarrow E$ $(\neg D \vee E)$. Example 3 (Conflict Rule): If $(E \wedge \neg E)$ exist simultaneously, the model is UNSAT, and the conflict rule number is recorded.

The SAT solver configuration is as follows: Backend: PySAT 0.1.8 + Glucose3; Preprocessing: Single clause propagation, plain text elimination (enabled); Branching strategy: VSIDS; Restart strategy: Luby sequence, conflict interval 100; Clause learning: Enabled (based on activity deletion); Large-scale CNF (>8000 clauses): WalkSAT, maximum flip 10^5 times, noise 0.3; Single instance timeout: 60 s; Random seed: 42.

The interaction mechanism between weights and SAT decisions: The weight vector obtained from Bayesian optimization is used to adjust the SAT branch order. Variable polarity is selected based on historical satisfaction and determined by weighting. Weights are updated after each SAT solution and remain unchanged during a single solution process.

Table 2: BD-SAT-Sec pseudo code

Input:
L, R // logs, rules
B, m0 // BO budget, initial samples
α // weight learning rate
SAT_cfg // DPLL (small CNF) / local-search (large CNF)
θ0, ε EI // GP hyperparams, EI stop threshold (optional)
Output: x*, y* // best SAT solution, best fitness
Procedure BD-SAT-Sec(L, R, B, m0, α, SAT_cfg, θ0):
1: F ← ExtractFeatures(L)

```
CNF ← PreprocessCNF( EncodeRulesToCNF(R, F) )
w ← InitWeights(|F|)
D ← ∅ ; (x*, y*) ← (null, -∞)
for k = 1..(m0 + B) do
    if k ≤ m0 then
        x ← InitSample(F, w)
    else
        GP ← FitGP(D, θ0)
        a ← ImprovedEI(GP, y*)
        x ← ArgMax(a, SearchSpace(F, w))
    end if

    s ← SATSolveGuided(CNF, w, SAT_cfg)
    y ← EvaluateObjective(s, L, R, w)      // e.g., λ1·SC(s)+λ2·G(s,w)
    D ← D ∪ {(x, y)}
    if y > y* then (x*, y*) ← (s, y)
    if k > m0 then                         // weight update after BO starts
        g ← BayesianGradient(GP, x)
        w ← NormalizeOrClip(w + α·g)
        if EI(a, x) < ε  EI then break
    end if
end for
return x*, y*
Function EvaluateObjective(s, L, R, w):
return λ1·SC(s) + λ2·G(s, w)
```

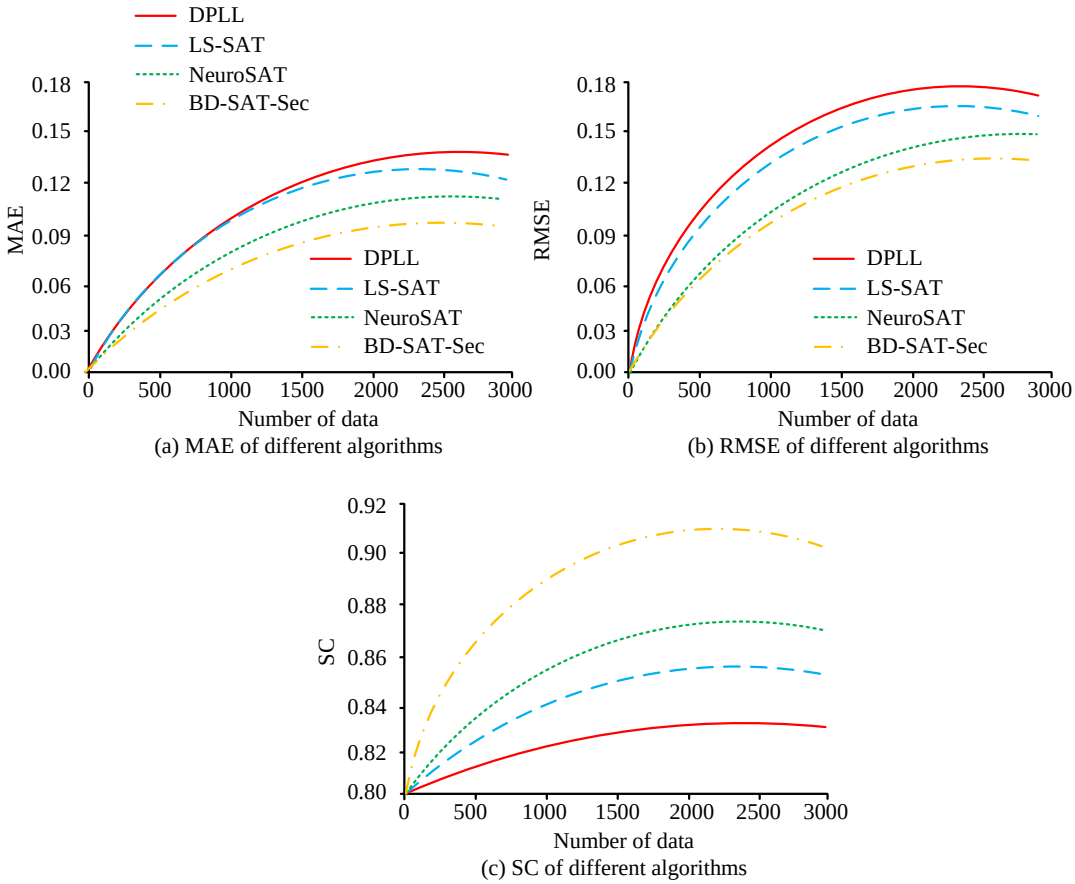


Figure 7: Overall performance comparison of different algorithms in DSI

3 Results

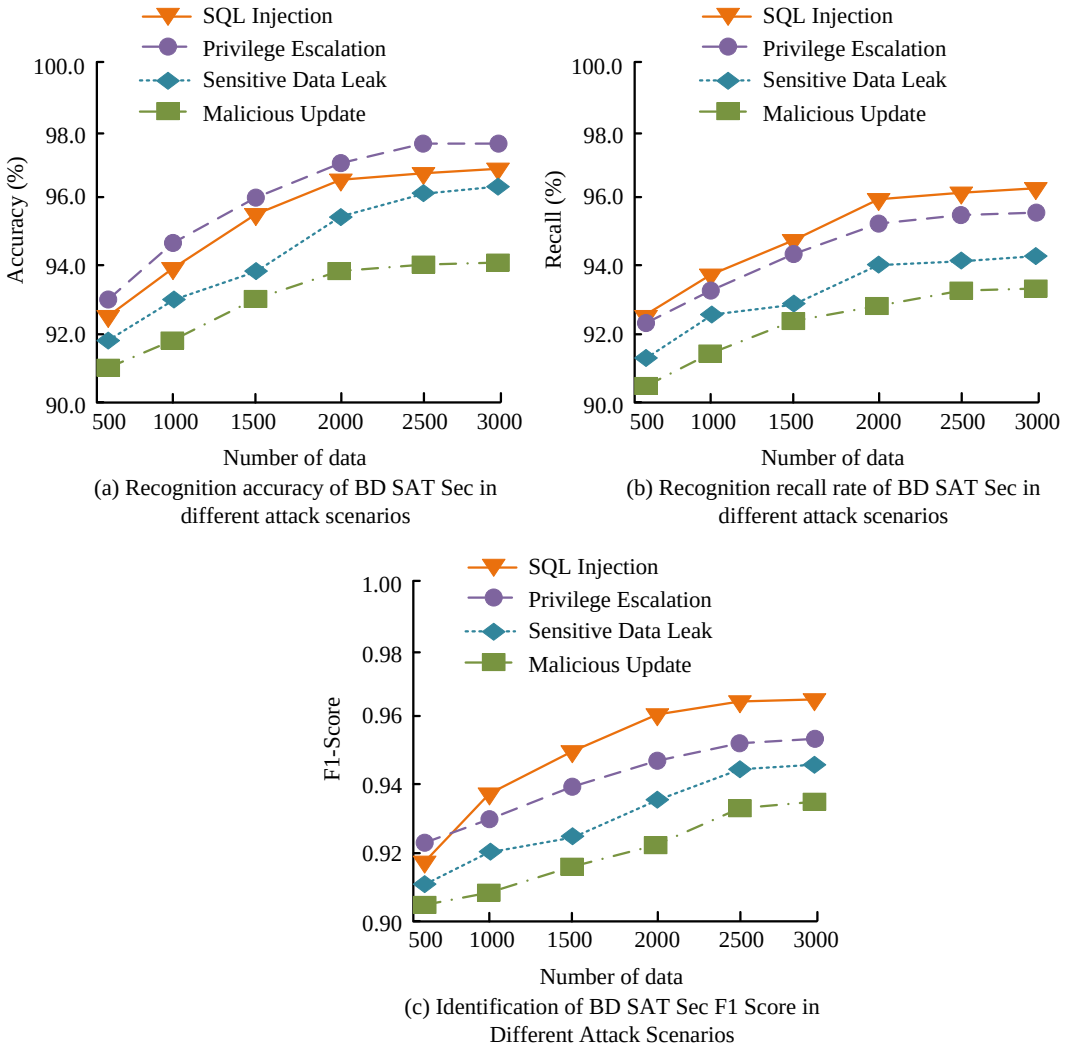
3.1 Model performance testing and comparative experiments

To facilitate reproduction, the experiment is conducted in a Windows 11 Pro environment with an Intel Core i9-13900K processor, 64 GB of DDR5 memory, and an NVIDIA RTX 4090 GPU. The implementation uses Python 3.10, with SAT solving based on PySAT 0.1.8 (Glucose3 is used as the solver backend; Preprocessing, such as single clause propagation and clause simplification are enabled; A single instance timeout of 60 seconds is used; A fixed random seed is used for each run). Bayesian optimization is performed using Scikit-Optimize 0.9.0, with a Gaussian process as the surrogate model and the RBF kernel function. Key parameters are

set as follows: learning rate of 0.01, maximum number of Bayesian optimization iterations of 200, EI threshold of 10^{-4}, and initial number of sampling points of 20. The dataset includes 1,287,436 enterprise SQL transactions collected over six months, cleaned to 1,042,315 valid records. Additionally, 1,482 SQL-related vulnerabilities are mapped to 312 structured rule templates, and 18,000 synthetic attack traces are injected, covering SQL injection (34%), privilege escalation (27%), sensitive field leakage (22%), and malicious update/delete (17%). Ground truth labels are generated using rule-based detection, CVE signature matching, and expert verification on 8,000 stratified samples. The final dataset contains 890,215 normal and 152,100 malicious samples (imbalance ratio 1:5.85). Data are split chronologically into training (70%), validation (20%), and test (10%) sets to avoid temporal leakage. All experiments are repeated five times with fixed random seeds, and results are reported as averages across runs. Rare vulnerability samples are balanced through the Synthetic Minority Oversampling Technique (SMOTE). The comparison algorithm selects representative methods in the current field: Neural-guided SAT solver (NeuroSAT), Local Search SAT Solver (LS-SAT), and baseline algorithm DPLL. These three methods represent the learning-guided paradigm, the heuristic local optimization paradigm, and the formal logical reasoning paradigm, respectively. This baseline selection covers three technical approaches: neural methods, heuristic methods, and precise reasoning methods, providing a structured comparative perspective. Comprehensive evaluation indicators include Mean Absolute Error (MAE), Root Mean Square Error (RMSE), SC, and algorithm running time. The experiment verifies the comprehensive advantages of the BD-SAT-Sec model in identification accuracy and solution efficiency. Although the task is framed as a security identification problem, MAE and RMSE are reported because BD-SAT-Sec internally optimizes a continuous security risk score derived from the weighted objective function (Equation (4)), which combines SC and logical satisfiability confidence. Specifically, each transaction is assigned a normalized risk value in [0,1] predicted by the Bayesian surrogate model, and MAE and RMSE measure the deviation between predicted risk scores and ground-truth binary labels (0 for normal, 1 for malicious) before thresholding. Final detection outcomes are obtained by applying an adaptive threshold (optimized on the validation set) to convert risk scores into binary decisions, from which detection rate, precision, recall, F1-score, and defense success rate are computed. To ensure statistical robustness, all experiments are repeated five times with fixed but different random seeds and reported with mean ± standard deviation. In addition, 95% Confidence Intervals (CI) are computed using bootstrapping (1,000 resamples) on the test set. For example, the detection rate of BD-SAT-Sec (97.5%) corresponds to a 95% CI of [96.8%, 98.1%], confirming the statistical reliability of the reported improvements over baseline methods. The comparison of its security identification performance with the comparison algorithm is shown in Figure 7.

In Figure 7(a), the MAE of the BD-SAT-Sec algorithm slowly increases from 0.03 to approximately 0.08, and then stabilizes after 2,000 data points. DPLL achieves a final MAE of 0.17, while LS-SAT and NeuroSAT perform in between. In Figure 7(b), the RMSE of BD-SAT-Sec stabilizes at around 0.12, the lowest overall value. In Figure 7(c), BD-SAT-Sec achieves a high coverage level of 0.90 when the number of samples exceeds 1,000. The results indicate that BD-SAT-Sec maintains high recognition rate and a high scene adaptability across a larger sample range, and its overall performance surpasses that of the comparative algorithms. The recognition stability analysis of BD-SAT-Sec under different attack scenarios is shown in Figure 8.

In Figure 8(a), accuracy reaches about 98.6% for SQL injection and 97.9% for permission violations. Sensitive field leakage and malicious updates are slightly lower, at 95.8% and 94.6%, due to their cross-table dependencies and more complex logical features. In Figure 8(b), recall follows a similar trend: 98.3% (SQL injection), 97.5% (permission violation), 95.1% (sensitive leakage), and 93.9% (malicious update). In Figure 8(c), the F1-scores are 0.98, 0.97, 0.95, and 0.94, respectively. These results show that BD-SAT-Sec maintains balanced high precision and recall across diverse attack types, highlighting the effectiveness of its BO-based feature adaptation mechanism. The impact of different Bayesian acquisition functions on convergence is presented in Figure 9.

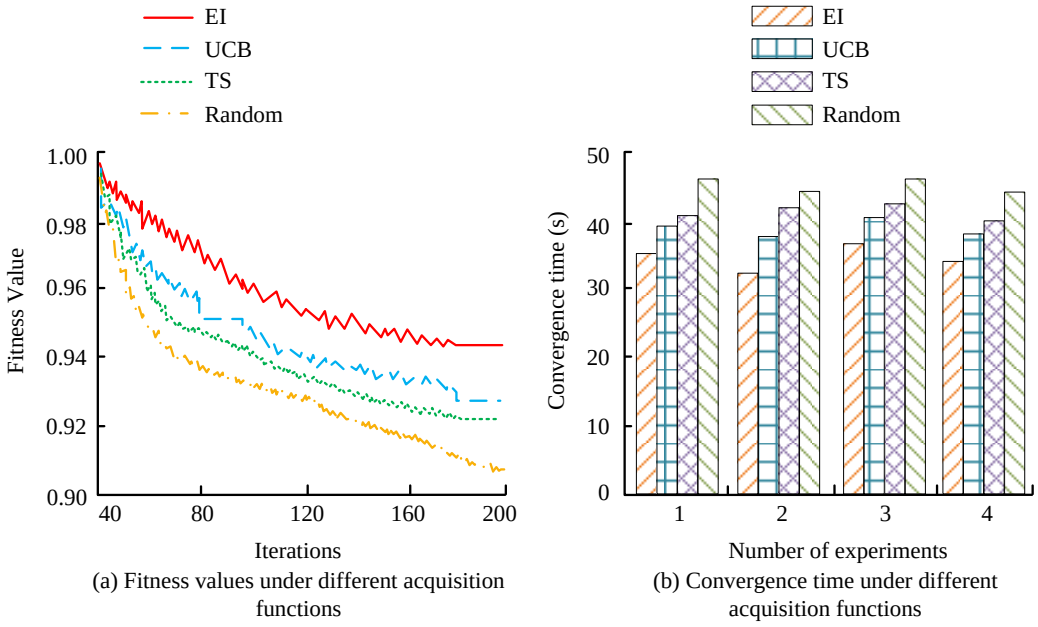


(a) Recognition accuracy of BD SAT Sec in different attack scenarios

(b) Recognition recall rate of BD SAT Sec in different attack scenarios

(c) Identification of BD SAT Sec F1 Score in Different Attack Scenarios

Figure 8: Analysis of recognition stability of BD-SAT-Sec in different attack scenarios



(a) Fitness values under different acquisition functions

(b) Convergence time under different acquisition functions

Figure 9: Influence of different Bayesian acquisition functions on algorithm convergence performance

In Figure 9(a), the EI curve always maintains the highest level and stabilizes after about 160 iterations, with a final fitness value of about 0.948. The Upper Confidence Bound (UCB) and Thompson Sampling (TS) curves are stable around 0.935 and 0.928. Random sampling drops the fastest and finally remains at around 0.910. In Figure 9(b), the EI strategy has the shortest average convergence time (about 35 s), UCB and TS are 39 s and 40 s, while the Random strategy is the slowest, with an average convergence time close to 46 s. EI shows consistent stability in four experiments, indicating its good repeatability and time efficiency in high-dimensional constrained optimization tasks. The EI acquisition function is superior to other strategies in terms of fitness value and time performance, which proves its core role in BD-SAT-Sec and can significantly improve the global convergence speed and security feature optimization accuracy of the SAT solution.

3.2 Security identification effect analysis

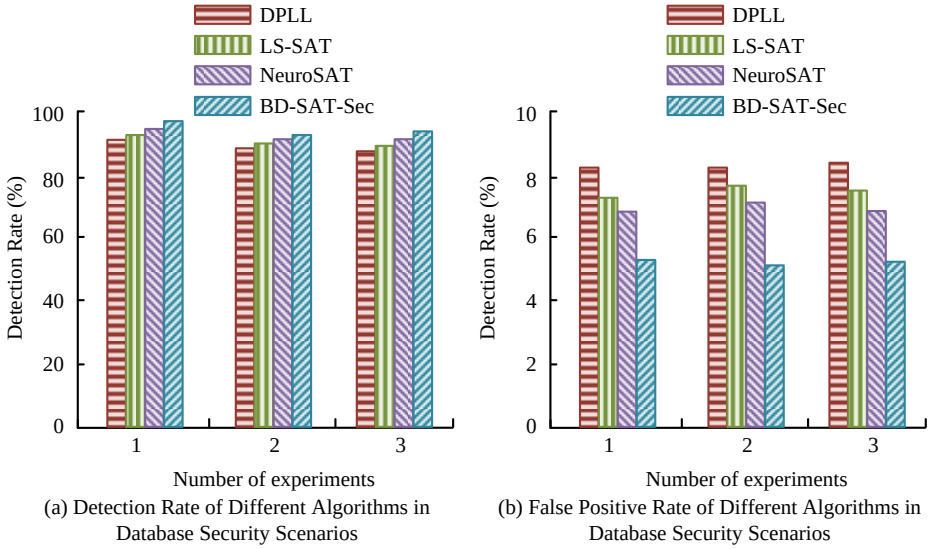
To isolate the causal contribution of each module, the study constructs a variety of controlled model configurations and gradually enables CNF pruning, DPLL precise solution, Bayesian surrogate modeling, and feature weight update mechanisms under different load conditions for comparative analysis.

In Table 3, under low load, expanding from SAT Baseline (91.2%, SC 0.83, 2.74 s) to Full BD-SAT-Sec increases detection to 97.5% (+6.3%), SC stability to 0.92 (+0.09), reduces delay to 2.11 s (−23.0%), and improves stability index from 0.86 to 0.93. Under high load, the Baseline drops to 88.7% detection and 0.79 SC, while BD-SAT-Sec maintains 96.8% and 0.90, with only −0.7% and −0.02 decreases, compared to −2.5% and −0.04 for the Baseline. CNF pruning and DPLL provide feasibility support, GPR+EI improves SC stability by 0.06 (0.79→0.85) under high load, and adaptive weighting contributes the largest detection gains (low load +2.9%, high load +4.5%) while reducing delay and enhancing stability, confirming both independent and collaborative module contributions. The comparison of vulnerability detection and defense success rates of different algorithms in database security scenarios is shown in Figure 10.

In Figure 10(a), BD-SAT-Sec achieves the highest detection rate, averaging 97.5%, outperforming NeuroSAT (94.3%), LS-SAT (92.7%), and DPLL (90.8%). In Figure 10(b), it maintains the lowest false alarm rate at around 5%, 1.6%, and 2.4% lower than NeuroSAT and LS-SAT. In Figure 10(c), its defense success rate reaches about 95.8%, 4.6% higher than NeuroSAT and well above DPLL (87.3%). These results confirm superior detection accuracy, false alarm control, and defense effectiveness, reflecting strong robustness in dynamic security environments. The gains are attributed to BO-guided feature reweighting and adaptive sampling. Table 4 further evaluates stability and scalability in deployment scenarios.

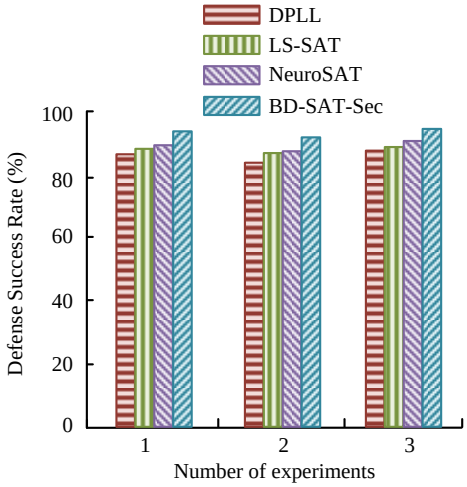
Table 3: Architectural contribution and sensitivity decomposition under varying load conditions

Model Variant	Architectural Components Activated	Load Condition	Detection Rate (%)	SC Stability	Response Latency	Stability Index
SAT Baseline	CNF Pruning + DPLL Exact Solving	Low	91.2	0.83	2.74	0.86
SAT + BO	CNF + DPLL + RBF-GPR Surrogate + EI Acquisition	Low	94.6	0.88	2.42	0.89
Full BD-SAT-Sec	CNF + DPLL + RBF-GPR + EI + Adaptive Weighting	Low	97.5	0.92	2.11	0.93
SAT Baseline	CNF Pruning + DPLL Exact Solving	High	88.7	0.79	3.08	0.82
SAT + BO	CNF + DPLL + RBF-GPR Surrogate + EI Acquisition	High	92.3	0.85	2.65	0.87
Full BD-SAT-Sec	CNF + DPLL + RBF-GPR + EI + Adaptive Weighting	High	96.8	0.90	2.24	0.91



(a) Detection Rate of Different Algorithms in Database Security Scenarios

(b) False Positive Rate of Different Algorithms in Database Security Scenarios



(b) Defense Success Rate of Different Algorithms in Database Security Scenarios

Figure 10: Comparison of vulnerability detection and defense success rates of different algorithms in database security scenarios

Table 4: Stability and scalability evaluation of different algorithms in database deployment environment

Model	Average CPU utilization rate (%)	GPU stable fluctuation amplitude (%)	Concurrent connection processing capability (Connections/s)	System stability index	Memory usage (GB)
DPLL	72.4	8.9	184	0.83	4.8
LS-SAT	69.3	7.2	203	0.86	4.2
NeuroSAT	66.1	6.3	218	0.89	3.9
BD-SAT-Sec	63.8	5.1	247	0.93	3.4
Average	67.9	6.9	213.0	0.88	4.1
Improvement over DPLL	12.0	42.7	34.2	12.0	29.2

Table 5: Comparison of vulnerability detection and defense success rates of different algorithms in database security scenarios

Model	Detection Rate (%)	False Positive Rate (%)	Defense Success Rate (%)	Response Latency (s)	Precision (%)
DPLL	90.8	8.4	87.3	3.26	88.2
LS-SAT	92.7	7.6	89.1	2.84	90.5
NeuroSAT	94.3	6.8	91.2	2.57	82.8
BD-SAT-Sec	97.5	5.2	95.8	2.11	96.4
Average	93.8	7.0	90.9	2.70	92.0
Improvement over DPLL	7.4	38.1	9.7	35.3	9.3

In Table 4, BD-SAT-Sec performs best across all indicators. Its average CPU usage is 63.8%, about 8.6% lower than DPLL (72.4%). The GPU stable fluctuation range is 5.1%, lower than NeuroSAT's 6.3% and LS-SAT's 7.2%, indicating improved energy stability under high concurrent workloads. BD-SAT-Sec achieves 247 connections/s, which is 34.2% higher than DPLL, demonstrating stronger throughput and real-time response under high-frequency access requests. Its system stability index reaches 0.93, showing the smallest performance variation and highest reliability during long-term and high-concurrency operation. These results confirm that the BO-driven dynamic parameter adjustment and adaptive SAT allocation strategy enable superior stability, resource efficiency, and scalability, supporting enterprise-level real-time security deployment. Table 5 further compares time efficiency and resource usage across algorithms.

In Table 5, BD-SAT-Sec achieves a detection rate of 97.5%, outperforming NeuroSAT by 3.2% and DPLL (90.8%), indicating improved threat coverage and reduced missed detections. Its false positive rate is 5.2%, representing reductions of 38.1% and 31.6% compared with DPLL and LS-SAT, respectively, demonstrating higher decision precision. The defense success rate reaches 95.8%, 4.6% higher than NeuroSAT, confirming stronger adaptive repair capability. With a response latency of 2.11 s, the lowest among all methods and 35.3% faster than DPLL, the BO-driven dynamic weighting mechanism significantly enhances solving efficiency and real-time responsiveness. Overall, BD-SAT-Sec outperforms baseline algorithms in accuracy, false alarm control, defense effectiveness, and response speed, validating its convergence stability and practical suitability for high-intensity, real-time database security protection. Beyond numerical gains, the results advance the state-of-the-art in three aspects. First, the higher detection rate (97.5%) with lower false positives shows that combining SAT-based verification with Bayesian optimization improves both logical consistency and predictive stability. Second, improved SC under long-tail attacks confirms that uncertainty-driven weight adaptation strengthens rare-vulnerability detection. Third, enhanced concurrency and reduced latency demonstrate that formal constraint reasoning remains feasible for real-time deployment when paired with adaptive search. Overall, BD-SAT-Sec moves database security from heuristic detection toward a unified, logically verifiable, and dynamically optimized framework.

4 Discussion

In recent years, research on SAT satisfiability and constraint modeling has provided a theoretical basis for complex logical reasoning. Vyalyi conducted research from the perspective of the satisfiability of algebraic formulas in finite fields, focusing on theoretical complexity and solvability analysis, but did not involve optimization search or security identification problems in adversarial environments in practical application scenarios [19]. In contrast, BD-SAT-Sec applies SAT modeling to the formal expression of database security rules and combines adaptive optimization mechanisms to improve identification performance in dynamic environments. Ulrich-Oltean et al. studied the SAT encoding adaptive selection problem of pseudo-Boolean and integer constraints, and improved the solution efficiency by optimizing the encoding strategy. However, its optimization goal mainly focused on performance improvement at the solver level, rather than objective function optimization for security identification tasks, and did not introduce a probabilistic uncertainty modeling mechanism [20]. BD-SAT-Sec introduces Bayesian optimization as a proxy model driving mechanism. Under the premise of ensuring logical satisfiability, adaptively guides the search to focus on high-risk areas, realizing task-oriented security identification optimization. Existing SAT research mainly focuses on theoretical solvability or coding efficiency, while BD-SAT-Sec realizes a unified framework for formal logic verification and uncertainty-driven search. Under high-dimensional constraints, it takes into account rule consistency verification, rare vulnerability identification capability, and convergence efficiency, providing a new solution for DSI that is geared towards practical tasks.

5 Conclusion

The accuracy and real-time performance of DSI were directly related to the defense capability and operational stability of the information system. To address limited logical expression ability and insufficient solution efficiency of traditional security detection methods, this study proposed the BD-SAT-Sec algorithm. It integrated logical constraint solving and probabilistic optimization to realize formal modeling and dynamic feature updating of database security rules. In experiments, the algorithm performed excellently on multi-dimensional performance indicators. Under the unified evaluation protocol, BD-SAT-Sec had a detection rate of 97.5%, a false alarm rate of 5.2%, a defense success rate of 95.8%, and a response delay of 2.11 s. On the deployment side, the CPU usage was 63.8%, the GPU stable fluctuation was 5.1%, the concurrent processing capacity was 247 connections/s, and the system stability index was 0.93. Compared with DPLL, the concurrency capability was increased by approximately 34.2%, and the response delay was reduced by approximately 35.3%. Compared with NeuroSAT, the detection rate was increased by 3.2%, and the defense success rate was increased by 4.6%. BD-SAT-Sec realized the deep integration of SAT logical reasoning and dynamic BO, enhancing the intelligence, adaptability, and real-time nature of DSI. Although BD-SAT-Sec demonstrates strong performance, several practical challenges remain for real-world deployment. Large-scale distributed databases may significantly increase CNF complexity, requiring incremental or distributed SAT solving to maintain efficiency. In

addition, Bayesian weight adaptation may be affected by concept drift or adversarial feature manipulation, necessitating drift-aware updating mechanisms. Finally, latency-sensitive environments such as financial systems may demand further lightweight optimization or hardware acceleration. Future research will therefore focus on scalable SAT architectures, online drift-aware optimization, privacy-preserving collaborative security learning, and robustness evaluation against rule-evasion attacks to enhance practical adaptability.

6 Funding

The research is supported by the Key Research Projects of Higher Education Institutions in Henan Province for 2026 "Method for Detecting Missing Variants and Genotypes Based on High-Precision Data and Deep Learning" (No:26A520019).

References

[1] Indu Singh, and Rajni Jindal. Outlier based intrusion detection in databases for user behaviour analysis using weighted sequential pattern mining. International Journal of Machine Learning and Cybernetics, 15(7):2573-2593, 2024. https://doi.org/10.1007/s13042-023-02049-4

[2] Weimin Li, Weihong Tian, Zhengmao Yan, Zitong Li, Jie Gao, and Fan Wu. Coraldb: A collaborative database for data sharing based on permissioned blockchain. IEEE Transactions on Mobile Computing, 23(9):8886-8901, 2024. https://doi.org/10.1109/TMC.2024.3357499

[3] Asif Iqbal, Siffat Ullah Khan, Mahmood Niazi, Mamoona Humayun, Najm Us Sama, Arif Ali Khan, and Aakash Ahmad. Advancing database security: A comprehensive systematic mapping study of potential challenges. Wireless Networks, 30(7):6399-6426, 2024. https://doi.org/10.1007/s11276-023-03436-z

[4] Tian Xia, Qingchun Hou, Ning Zhang, Qihuan Dong, Weiran Li, and Chongqing Kang. Database generation for data-driven power system security assessment under uncertainty. IEEE Transactions on Power Systems, 39(5):6168-6182, 2024. https://doi.org/10.1109/TPWRS.2024.3352825

[5] Jie Zhong, Qinyao Pan, Wenying Xu, and Yang Liu. Asymptotical stabilization for probabilistic Boolean control networks under operators and inputs constraints. IEEE Transactions on Automatic Control, 68(7):4313-4320, 2022. https://doi.org/10.1109/TAC.2022.3203019

[6] Stephan Eggersglüß, Sylwester Milewski, Janusz Rajski, and Jerzy Tyszer. A new static compaction of deterministic test sets. IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, 31(4):411-420, 2023. https://doi.org/10.1109/TVLSI.2023.3240246

[7] Küng. A systematic literature review of authorization and access control requirements and current state of the art for different database models. International Journal of Web Information Systems, 20(1):1-23, 2024. https://doi.org/10.1108/IJWIS-08-2023-0131

[8] E. Gupta, S. Sural, J. Vaidya, and V. Atluri. Enabling attribute-based access control in NoSQL databases. IEEE Transactions on Emerging Topics in Computing, 11(1):208-223, 2022. https://doi.org/10.1109/TETC.2022.3162002

[9] Hongcheng Xie, Yu Guo, Yinbin Miao, and Xiaohua Jia. Access-pattern hiding search over encrypted databases by using distributed point functions. IEEE Transactions on Computers, 74(3):1066-1078, 2024. https://doi.org/10.1109/TC.2024.3504288

[10] Indu Singh, and Rajni Jindal. Trust factor-based analysis of user behavior using sequential pattern mining for detecting intrusive transactions in databases. Journal of Supercomputing, 79(10):11101-11133, 2023. https://doi.org/10.1007/s11227-023-05090-w

[11] Shaorong Wang, and Guiling Long. Research on campus network security protection system framework based on cloud data and intrusion detection algorithm. Soft Computing, 27(10):6835-6844, 2023. https://doi.org/10.1007/s00500-023-08115-x

[12] Jianxin Deng, Gang Liu, and Xiangming Zeng. Blockchain-based security access control system for sharing squeeze casting process database. Integrating Materials and Manufacturing Innovation, 13(1):92-104, 2024. https://doi.org/10.1007/s40192-023-00337-z

[13] Mojtaba Rafiee. Multi-adjustable join schemes with adaptive indistinguishably security. IEEE Transactions on Dependable and Secure Computing, 21(4):4024-4034, 2023. https://doi.org/10.1109/TDSC.2023.3343872

[14] Nicolai Fiege, Martin Kumm, and Peter Zipf. Bit-level optimized constant multiplication using boolean satisfiability. IEEE Transactions on Circuits and Systems I: Regular Papers, 71(1):249-261, 2023. https://doi.org/10.1109/TCSI.2023.3327814

[15] Tom Servranckx, José Coelho, and Mario Vanhoucke. A genetic algorithm for the Resource-constrained project scheduling problem with alternative subgraphs using a boolean satisfiability solver. European Journal of Operational Research, 316(3):815-827, 2024. https://doi.org/10.1016/j.ejor.2024.02.041

[16] Boqin Qin, Yilun Chen, Haopeng Liu, Hua Zhang, Qiaoyan Wen, and Linhai Song. Understanding and detecting real-world safety issues in rust. IEEE Transactions on Software Engineering, 50(6):1306-1324, 2024. https://doi.org/10.1109/TSE.2024.3380393

[17] Xiaoli Li, Ling Zhao, Haobin Shen, Hanlin Du, and Zhida Guo. Automatic detection method of website vulnerabilities based on an associated data drive.

Journal of Web Engineering, 24(2):217-242, 2025. https://doi.org/10.13052/jwe1540-9589.2423

[18] Li Shang, Zi Zhang, Fujian Tang, Q. Cao, Hong Pan, and Zhibin Lin. Signal process of ultrasonic guided wave for damage detection of localized defects in plates: From shallow learning to deep learning. Journal of Data Science and Intelligent Systems, 3(2):149-164, 2025. https://doi.org/10.47852/bonviewJDSIS32021771

[19] M. N. Vyalyi. Testing the satisfiability of algebraic formulas over the field of two elements. Problems of Information Transmission, 59(1):57-62, 2023. https://doi.org/10.1134/S0032946023010052

[20] Felix Ulrich-Oltean, Peter Nightingale, and James Alfred Walker. Learning to select SAT encodings for pseudo-Boolean and linear integer constraints. Constraints, 28(3):397-426, 2023. https://doi.org/10.1007/s10601-023-09364-1