International Journal of Applied Mathematical Research, 14 (2) (2025) 21-35

International Journal of Applied Mathematical Research

Website: www.sciencepubco.com/index.php/IJAMR
doi: https://doi.org/10.14419/yxp00a41
Research paper

A Formal Verification-Based Risk Scoring System for Code-Level Vulnerabilities in Critical Applications

Pavan Paidy[1*]

[1]*FINRA, Maryland, USA*
[*]*Corresponding author E-mail:pavanp.focus@gmail.com*

Abstract

This paper presents a novel framework for addressing code-level vulnerabilities in critical applications by combining formal verification with risk scoring systems. It ensures the correctness and reliability of code while prioritizing vulnerabilities based on exportability and impact. The approach is applied to high-stakes industries such as healthcare, aerospace, and industrial control, where system failures can have catastrophic consequences. A numerical example demonstrates a 22% reduction in risk (from 1.905 to 1.485) within budgetary constraints. Results show that this combined method offers a robust, cost-efficient solution for improving security, making it practical for real-world deployment. The framework emphasizes risk reduction and cost optimization in resource-constrained environments.

Keywords: *Cybersecurity, Vulnerability Mitigation, Optimization Model, Risk Reduction, Cost Optimization, Real-world Applications.*

1. Introduction

The increasing complexity of modern systems has brought with it a heightened need for methods that ensure the safety, reliability, and security of critical applications. Systems used in sectors such as healthcare, aerospace, and industrial control are especially susceptible to failures, and any vulnerabilities in these systems could result in catastrophic consequences. Ensuring the robustness of these systems, particularly during their development and operational phases, is critical. While traditional methods such as testing and code review offer some level of assurance, they are insufficient when dealing with highly complex systems, especially where failures are unacceptable. Formal verification methods have emerged as a powerful tool to address this challenge, providing mathematically sound guarantees of system correctness [24, 29], [30].

Formal verification is a process that uses mathematical models and logical reasoning to prove that a system meets its specification. It is particularly effective in the verification of safety-critical systems where exhaustive testing is not feasible. For instance, in blockchain applications, smart contracts are subject to verification using formal methods to prevent vulnerabilities such as reentrancy attacks and transaction malleability, which can have severe financial and security implications [22], [24]. Similarly, industrial control systems are verified to ensure they adhere to safety protocols, preventing faults that could result in accidents [25], [28]. The ability to apply formal methods to these applications has thus significantly enhanced their reliability and security.

Alongside formal verification, another important aspect of securing critical systems is the assessment of vulnerabilities through risk scoring. Vulnerabilities in software systems can vary significantly in their severity, and the consequences of exploiting these vulnerabilities can range from minor inconveniences to catastrophic system failures. To prioritize remediation efforts, risk scoring systems such as the Common Vulnerability Scoring System (CVSS) have been developed. CVSS provides a numerical value that represents the severity of a vulnerability based on several factors, including the exploitability of the vulnerability and its impact on the system [23, 25], [31, 25]. This score helps organizations focus on the most critical vulnerabilities, directing resources to mitigate the most pressing risks first.

The integration of formal verification with risk scoring offers a robust framework for identifying, assessing, and mitigating vulnerabilities in critical applications. While formal verification ensures that a system operates as intended and does not have any latent faults, risk scoring provides a way to quantify the potential impact of discovered vulnerabilities. This combination allows for a structured approach to securing systems, where both correctness and the likelihood of exploit are considered together, helping organizations develop more resilient and secure systems [29, 26], [32, 27].

The integration of formal verification with risk scoring offers a robust framework for identifying, assessing, and mitigating vulnerabilities in critical applications. While formal verification ensures that a system operates as intended and does not have any latent faults, risk scoring provides a way to quantify the potential impact of discovered vulnerabilities. This combination allows for a structured approach to securing systems, where both correctness and the likelihood of exploit are considered together.

A widely used method for calculating CVSS scores is represented by the equation [2]:

$$CVSS = Base \times Exploitability \times Impact \tag{1}$$

In this equation:

- Base represents the fundamental characteristics of the vulnerability, including factors such as access complexity and authentication requirements.
- Exploitability measures how easily the vulnerability can be exploited, considering factors like remote attack potential and necessary privileges.
- Impact quantifies the severity of the damage if the vulnerability is exploited, evaluating how it affects system integrity, confidentiality, and availability.

This equation helps prioritize vulnerabilities based on their severity, allowing organizations to focus on the most critical threats first, thereby optimizing resource allocation for security efforts [2].
Formal verification, coupled with effective risk scoring, is indispensable in the modern landscape of critical systems, ensuring both system correctness and resilience against potential threats.
The motivation for this work arises from the increasing complexity and reliance on safety-critical systems, where even minor vulnerabilities can lead to catastrophic consequences. As systems in fields such as aerospace, healthcare, and industrial automation grow in scale and complexity, ensuring their security and reliability becomes a significant challenge. Traditional testing and manual review methods fall short when it comes to verifying complex systems. Formal verification offers a mathematically rigorous approach to ensuring that systems behave correctly and as intended. Combining formal verification with risk scoring can enhance the security of critical systems by not only identifying vulnerabilities but also helping to prioritize them based on their severity [1]. This combination motivates the need to explore effective methodologies that improve the reliability and safety of high-stakes systems.
The justification for this work lies in the growing importance of formal methods and security assessments in the context of emerging technologies like blockchain and industrial control systems. Vulnerabilities in smart contracts, industrial control systems, and embedded applications are often discovered too late, leading to significant financial, operational, or safety risks. In this context, formal verification provides a reliable means to prove that critical systems are secure and function as intended under all scenarios. Risk scoring frameworks like CVSS enable the prioritization of vulnerabilities, facilitating efficient resource allocation for security efforts [2]. This work justifies itself as it contributes to bridging the gap between formal verification and risk scoring, both of which are essential in safeguarding critical infrastructure.
The scope of this work focuses on formal verification techniques and their integration with risk scoring models to assess vulnerabilities in critical applications. This includes applying formal methods to various domains such as blockchain technology, industrial control systems, and IoT devices, which are increasingly under threat from cyber-attacks. The scope also encompasses exploring how formal verification can complement existing risk assessment models to enhance decision-making processes in cybersecurity. The significance of this work lies in its ability to provide a structured methodology that ensures both the correctness and security of critical systems while minimizing the likelihood of exploitation by attackers. By combining formal verification with risk scoring, this approach offers a comprehensive solution to vulnerability detection and mitigation, as highlighted by [7], [2].
The aim of this work is to develop and explore the integration of formal verification methods with risk scoring systems to identify and assess vulnerabilities in critical applications effectively. The objectives of this research are twofold: to apply formal verification techniques to ensure the correctness of critical systems, and to combine these techniques with risk scoring to prioritize vulnerabilities based on their potential impact. The statement of the problem addressed by this work is the lack of a comprehensive approach that combines formal verification with risk scoring to evaluate and mitigate vulnerabilities in critical applications. By addressing this gap, the work aims to enhance the reliability and security of critical systems, ensuring that they are both secure and reliable in the face of potential vulnerabilities [5], [6].

2. Related Work

The increasing need for secure and reliable critical systems has driven extensive research into both formal verification techniques and risk scoring methods. This section reviews the literature on these topics, with a particular focus on their application to high-stakes domains such as blockchain, industrial control systems, and embedded devices.

2.1. Formal Verification in Critical Systems

Formal verification techniques have been extensively applied to critical systems to ensure that they meet specified safety and correctness requirements. One prominent study in this domain investigates the use of formal methods to detect vulnerabilities in blockchain-based applications, particularly focusing on smart contracts. This work addresses issues such as reentrancy attacks and transaction malleability, using formal verification to prevent potential exploits before deployment [1]. Similarly, in the context of industrial control systems, formal methods have been applied to programmable logic controllers (PLCs) to ensure that these systems adhere to safety protocols, which are essential for avoiding accidents and ensuring system reliability [7]. Formal methods play a crucial role in preventing errors that could otherwise go undetected through conventional testing. Further, recent work has demonstrated the application of formal verification techniques to microarchitecture models, particularly in ensuring the correctness of RISC-V designs to avoid faults in safety-critical systems [5], [6].
In addition, formal verification techniques have been applied to microarchitecture models for hardware systems. The work of [5] provides a comprehensive survey on the application of formal verification to RISC-V microarchitecture models, focusing on verifying correctness and detecting design faults early in the development phase. This is especially important in safety-critical systems where a failure in hardware can lead to significant consequences. Such formal verification methods ensure that the design and implementation of safety-critical hardware components are free from flaws that could lead to system failures.

2.2. Risk Scoring and Vulnerability Assessment

Risk scoring systems, such as the Common Vulnerability Scoring System (CVSS), have become integral to assessing the severity of vulnerabilities in software systems. CVSS assigns numerical scores to vulnerabilities, enabling organizations to prioritize their remediation efforts effectively. The work in [2] introduces the CVSS model and discusses its application to a wide range of software systems, providing a quantitative measure to assess vulnerabilities based on exploitability and impact. The work highlights the importance of risk scoring systems in cybersecurity, offering a structured approach to identify the most critical vulnerabilities requiring attention.

Extending the use of CVSS, [9] applies the scoring system to smart contracts in Ethereum. This study focuses on developing tools to automate the evaluation of vulnerabilities in smart contracts, allowing developers to address security risks proactively. The integration of CVSS with automated analysis tools is essential in the fast-evolving blockchain environment, where vulnerabilities can lead to significant financial losses. The study emphasizes the need for fine-grained risk assessment models, particularly for decentralized applications where vulnerabilities can have wide-reaching consequences.

Additionally, [10] extends CVSS by combining it with machine learning algorithms to predict the risk associated with unknown vulnerabilities, providing early warnings for system developers. Further, recent work by [11] uses CVSS to assess vulnerabilities in industrial control systems, particularly focusing on critical infrastructure, and proposes an integrated framework combining CVSS with formal verification techniques. The research in [12] introduces an advanced version of CVSS tailored for Internet of Things (IoT) devices, enabling more accurate risk assessment in resource-constrained environments. Moreover, the work of [13] investigates the application of CVSS in the context of security reliability in embedded software systems, offering new ways to evaluate vulnerabilities in real-time systems used in critical applications.

2.3. Integration of Formal Verification and Risk Scoring

A promising area of research is the integration of formal verification and risk scoring to provide a more comprehensive solution to vulnerability detection and mitigation. For example, [11] presents a formal framework to assess and mitigate emergent security risks in generative AI models, combining formal verification techniques with risk-based approaches. This integrated approach allows for both the identification of vulnerabilities and the prioritization of remediation efforts based on the severity of identified risks.

In a similar vein, [12] discusses the integration of formal verification with risk assessment models in cyber-physical systems. By combining both approaches, the authors were able to detect vulnerabilities early and provide a structured approach to mitigating risks, enhancing the overall security and reliability of these systems. This work highlights how combining verification with risk scoring can improve decision-making in security-critical systems.

Furthermore, the work by [13] explores the application of formal verification in industrial control systems and integrates risk scoring to prioritize vulnerabilities based on their potential impact. [14] investigates the integration of formal verification with CVSS for IoT devices, aiming to enhance the security of resource-constrained environments by combining both methods. Another important study by [15] extends the concept of combining verification with risk scoring to blockchain systems, helping developers identify and mitigate vulnerabilities in smart contracts. Finally, [16] discusses the integration of risk scoring with formal methods in embedded systems, providing a framework for evaluating and mitigating risks in safety-critical real-time applications.

Further exploring the combination of formal verification and risk scoring, [17] presents a study on formal methods and their application to critical systems, integrating verification and risk-based decision-making to enhance overall system reliability. This research provides insight into how formal methods can be combined with risk analysis to prevent vulnerabilities in systems such as avionics and medical devices, ensuring the robustness of such high-stakes applications.

2.4. Vulnerability Detection in IoT and Embedded Systems

The detection of vulnerabilities in IoT devices and embedded systems has become an increasingly important research area, especially as these systems become more pervasive in critical infrastructure. The work of [13] surveys IIoT protocols and presents a framework for vulnerability risk analysis based on CVSS. The study highlights the need for formal verification and risk scoring in the context of IoT, where vulnerabilities are often discovered too late, leading to large-scale security breaches.

Moreover, [14] addresses security reliability in embedded software used in real-time, safety-critical applications. The study emphasizes the need for dimensional security assurance techniques to assess vulnerabilities in embedded systems. These systems are particularly susceptible to attack because they are often resource-constrained and lack robust security measures. By combining formal verification with risk scoring, this work provides an essential framework to ensure the security and reliability of embedded systems.

In addition, [15] investigates the application of CVSS to assess vulnerabilities in resource-constrained IoT environments, particularly focusing on security reliability in the industrial IoT (IIoT) sector. This work extends the use of CVSS by integrating it with formal verification techniques, improving the accuracy and effectiveness of risk assessment in real-time critical applications. Another significant contribution is made by [16], who proposes a novel method for vulnerability detection in embedded systems, leveraging both formal verification and automated risk assessment models. This approach is particularly beneficial for systems operating in unpredictable environments where traditional testing methods may fall short.

Furthermore, [17] explores the challenges associated with vulnerability detection in safety-critical embedded systems, emphasizing the importance of integrating formal verification with risk analysis models. This study shows that by combining these methodologies, it is possible to not only detect vulnerabilities but also prioritize them based on their potential impact on system functionality and safety. This research provides a comprehensive approach to securing embedded systems, ensuring that critical infrastructure is resilient against potential threats.

Finally, [18] and [19] provide an in-depth analysis of formal methods and automated vulnerability detection in IoT and embedded systems, focusing on methods for risk prioritization and mitigation. These studies explore the integration of formal verification with machine learning-based risk models, enhancing the ability to predict and detect vulnerabilities in real-time, even in highly dynamic environments. By leveraging both methodologies, these works present a promising direction for improving the security of critical embedded systems and ensuring their reliability.

This research builds on existing work by integrating formal verification methods with risk scoring systems to provide a comprehensive approach to assessing and mitigating vulnerabilities in critical systems. While previous studies have focused on either formal verification or risk scoring, this work brings together both approaches to offer a more holistic solution. By applying formal verification in combination with risk scoring, this research aims to improve the security and resilience of critical systems, ensuring that vulnerabilities are identified and prioritized efficiently.
In conclusion, the literature on formal verification and risk scoring demonstrates the significant role these methods play in ensuring the security and reliability of critical applications. Formal verification techniques help verify the correctness of systems, while risk scoring models like CVSS provide a structured way to assess the severity of vulnerabilities. Integrating both approaches offers a more robust solution, ensuring that vulnerabilities are detected early and prioritized based on their potential impact. The contributions of this work lie in combining these two methodologies to enhance the security and reliability of critical systems.

3. Preliminaries to the Study

In this section, we present the mathematical foundations necessary for the methodology of this study. The core mathematical models employed include formal verification, risk scoring, and vulnerability detection in critical systems, which rely on formal methods, probability theory, optimization, and algebraic techniques. These methods serve as the foundation for our approach to formal verification and risk mitigation in IoT, embedded systems, and blockchain applications.

3.1. Formal Verification and Logic

Formal verification ensures that systems behave according to their specifications. The primary mathematical methods used for formal verification are temporal logic and predicate logic.

3.1.1. Temporal Logic

Temporal logic allows for the specification of properties that hold over time, such as safety and liveness properties. For example, we may wish to express that ”if a vulnerability exists, it will eventually be detected”. In Linear Temporal Logic (LTL), this is represented as:

$$\Box(V \implies \Diamond D) \tag{2}$$

Where:

- $\Box$ denotes ”always” (i.e., the property holds in all future states),
- $\Diamond$ denotes ”eventually” (i.e., the property will hold in some future state),
- V denotes the occurrence of a vulnerability,
- D denotes the detection of the vulnerability.

This formula specifies that whenever a vulnerability V is present, it must eventually lead to a detection D in the future.

3.1.2. Predicate Logic

Predicate logic is used for expressing system specifications and checking them for validity. A general form of predicate logic can be:

$$\forall x\,(V(x) \implies \text{Unsafe}(x)) \tag{3}$$

Where:

- $V(x)$ represents a vulnerability in the system component x,
- $\text{Unsafe}(x)$ indicates that x is unsafe due to the vulnerability.

3.2. Risk Scoring and Probability Theory

In the context of vulnerability detection, we use risk scoring models like CVSS to calculate the severity of vulnerabilities. The risk score is calculated by multiplying exploitability and impact:

$$\text{Risk Score} = \text{Exploitability} \times \text{Impact} \tag{4}$$

This model is used to quantify the potential risk posed by vulnerabilities in the system.

3.2.1. CVSS Base Score

The CVSS base score combines the exploitability and impact scores to calculate an overall risk score for vulnerabilities:

$$\text{BaseScore} = (0.6 \times \text{Impact}) + (0.4 \times \text{Exploitability}) \tag{5}$$

Where:

- Impact represents the potential damage caused by the exploitation of a vulnerability,
- Exploitability represents the likelihood of the vulnerability being exploited.

3.2.2. Risk and Probability Assessment

Risk can be quantified using probability theory. The general formula for the risk of a vulnerability is given by:

$$R = P(\text{attack}) \times I(\text{impact}) \tag{6}$$

Where:

- $P(\text{attack})$ is the probability of a successful attack exploiting the vulnerability,
- $I(\text{impact})$ is the impact of the attack, quantified on a predefined scale.

3.3. Optimization for Vulnerability Mitigation

Optimization techniques are used to allocate resources efficiently in the process of vulnerability mitigation. The integer linear programming (ILP) formulation used to prioritize vulnerabilities is as follows:

$$\min \sum_{i=1}^{n} R_i x_i \tag{7}$$

$$\text{subject to} \quad \sum_{i=1}^{n} C_i x_i \leq C_{\max} \tag{8}$$

Where:

- R_i is the risk score of vulnerability i,
- x_i is a binary decision variable that is 1 if vulnerability i is mitigated, and 0 otherwise,
- C_i is the cost to mitigate vulnerability i,
- $C_{\max}$ is the maximum available budget.

The objective is to minimize the total risk $\sum_{i=1}^{n} R_i x_i$, subject to the constraint that the total mitigation cost does not exceed the available budget $C_{\max}$.

3.4. Graph Theory for Model Checking

In formal verification, state machines and state transition graphs are used to represent possible states of a system and transitions between those states. Let $G = (V, E)$ represent a directed graph, where:

- V is the set of states (vertices),
- E is the set of transitions (edges between states).

The system must satisfy the property that for all reachable states $v \in V$, no failure occurs if the state is safe. This is represented as:

$$\forall v \in V \, (\text{Safe}(v) \implies \Box \, \text{NoFailure}(v)) \tag{9}$$

Where:

- $\text{Safe}(v)$ indicates that the system in state v is safe,
- $\text{NoFailure}(v)$ indicates that no failure occurs in state v,
- $\Box$ denotes that the property holds for all reachable states.

3.5. Game Theory for Adversarial Models

In the context of security, game theory models the interaction between an attacker (adversary) and a defender (system administrator). The general payoff matrix for a zero-sum game can be formulated as:

$$\text{Payoff}(D, A) = \text{Benefit}(A) - \text{Cost}(D) \tag{10}$$

Where:

- D represents the defender's strategy (e.g., patching a vulnerability),
- A represents the attacker's strategy (e.g., exploiting a vulnerability),
- $\text{Benefit}(A)$ is the payoff for the attacker if the attack is successful,
- $\text{Cost}(D)$ is the cost for the defender to implement a security measure.

The goal of the defender is to minimize the risk by choosing the optimal strategy based on the potential actions of the attacker.
This section presents the essential mathematical tools and techniques required for the study. We introduced formal verification using logic and set theory, explained risk scoring models such as CVSS, and explored optimization techniques for vulnerability mitigation. These mathematical methods and models are crucial for the formal analysis and risk mitigation of vulnerabilities in critical systems such as IoT, embedded systems, and blockchain applications.

4. Methodology

This section outlines the methodology adopted in this study to address the vulnerabilities in critical systems, focusing on formal verification, risk scoring, and optimization for vulnerability mitigation. The methodology is divided into three key subsections: Mathematical Formulation, Numerical Approach and Assumptions, and Evaluation Matrix. Each subsection builds upon the previous one to provide a structured approach for vulnerability detection, risk assessment, and system verification.

4.1. Mathematical Formulation

The objective of this subsection is to present a novel approach to modeling vulnerabilities, exploitability, and mitigation strategies in critical systems. Unlike traditional methodologies that use static models, we propose a dynamic framework that accounts for the evolving nature of system vulnerabilities and the ongoing interactions between attackers and defenders. This formulation integrates real-time optimization, evolutionary risk assessment, and adversarial game-theoretic modeling.

4.1.1. Dynamic Vulnerability Risk Assessment

In traditional vulnerability assessment models, risk scoring is a static process based on predefined scores. In contrast, we introduce a dynamic risk model where the risk $R_i(t)$ of vulnerability i evolves over time due to changes in system state, attack attempts, and mitigation efforts. We represent this time-dependent risk as:

$$R_i(t) = \beta_i(t) \cdot \gamma_i(t) \cdot \alpha_i(t) \tag{11}$$

Where:

- $\beta_i(t)$ is the exploitability factor at time t, accounting for both system state and external factors (such as evolving attack strategies),
- $\gamma_i(t)$ represents the system exposure factor, which varies with the system's vulnerability to external threats over time,
- $\alpha_i(t)$ is the attack success rate, representing the likelihood of successful exploitation based on the current system vulnerabilities and attack vectors.

The total risk $R_{\text{total}}(t)$ in the system at any time t is the aggregate risk from all individual vulnerabilities:

$$R_{\text{total}}(t) = \sum_{i=1}^{n} R_i(t) \tag{12}$$

This formulation allows the system's risk to evolve dynamically, accounting for both internal and external changes over time, offering a more accurate reflection of system security.

4.1.2. Adaptive Optimization for Vulnerability Mitigation

We now focus on the adaptive optimization model used to allocate limited resources effectively for vulnerability mitigation, with the objective of minimizing the evolving risk in the system. This model goes beyond traditional static resource allocation, introducing dynamic decision-making processes based on real-time risk and system states.
Let $x_i(t)$ be a binary decision variable representing whether vulnerability i is mitigated at time t. The optimization problem is then formulated as:

$$\min_{x_i(t)} \sum_{i=1}^{n} R_i(t) \cdot x_i(t) \tag{13}$$

Subject to:

$$\sum_{i=1}^{n} C_i \cdot x_i(t) \leq C_{\max}(t) \tag{14}$$

Where:

- C_i is the cost to mitigate vulnerability i,
- $C_{\max}(t)$ is the dynamically changing available budget for mitigation at time t,
- $x_i(t)$ is a binary decision variable for mitigating vulnerability i at time t.

This optimization problem adjusts based on the real-time risk scores and available resources, ensuring that resources are allocated where they are most needed at any given time. The solution to this problem provides an optimal strategy for minimizing the total system risk while adhering to budget constraints.

4.1.3. System Evolution and Temporal Logic

Since the system evolves over time, it is crucial to ensure that the mitigation strategies lead to a safe and secure state throughout the system's operation. We incorporate temporal logic to express properties that the system must satisfy over time, verifying that no vulnerabilities persist unchecked.
Consider the property that if vulnerability $V_i(t)$ is present at any time, it must eventually be mitigated. This is expressed as:

$$\Box(V_i(t) \implies \Diamond \text{Mitigated}(i,t)) \tag{15}$$

Where:

- $\Box$ represents the condition holding continuously over time,
- $\Diamond$ represents that the condition must eventually hold (mitigation must occur at some point in the future),
- $V_i(t)$ represents the presence of vulnerability i at time t,
- Mitigated(i,t) denotes that vulnerability i is mitigated at time t.

This ensures that mitigation strategies are not only planned but also dynamically tracked over time, ensuring system safety through continuous formal verification.

4.1.4. Adversarial Game Theory Model

A novel aspect of our model is the integration of dynamic game theory to account for the strategic interactions between the attacker and the defender. In contrast to static models where decisions are made once, we model the attacker's strategy as time-dependent and reactive to the defender's actions.
At each time step, the defender aims to minimize the system's vulnerability exposure by implementing mitigation strategies, while the attacker tries to exploit unmitigated vulnerabilities. The payoffs for both parties are as follows:

$$\text{Payoff}(D(t),A(t)) = \text{Benefit}_A(t) - \text{Cost}_D(t) \tag{16}$$

Where:

- $D(t)$ represents the defender's strategy at time t (e.g., patching vulnerabilities, enhancing detection mechanisms),
- $A(t)$ represents the attacker's strategy at time t (e.g., exploiting vulnerabilities),
- $\text{Benefit}_A(t)$ is the payoff for the attacker if the attack is successful at time t,
- $\text{Cost}_D(t)$ is the cost for the defender in implementing mitigation at time t.

The equilibrium solution to this game determines the optimal defense strategy $D^*(t)$, which minimizes the attacker's benefit while respecting the constraints of available resources.

4.1.5. Integrated Dynamic Security Model

To unify the dynamic risk assessment, adaptive mitigation, temporal logic, and game-theoretic model into a cohesive methodology, we define the following integrated dynamic security model:

$$\min_{x_i(t)} \sum_{i=1}^{n} R_i(t) \cdot x_i(t) \tag{17}$$

Subject to:

$$\sum_{i=1}^{n} C_i \cdot x_i(t) \leq C_{\max}(t) \tag{18}$$

With the temporal logic property ensuring continuous mitigation:

$$\Box(V_i(t) \implies \Diamond \text{Mitigated}(i,t)) \tag{19}$$

And the game-theoretic interaction between attacker and defender:

$$\text{Payoff}(D(t),A(t)) = \text{Benefit}_A(t) - \text{Cost}_D(t) \tag{20}$$

This integrated approach ensures that the system is both safe and cost-effective, minimizing the overall risk while satisfying the resource constraints and adversarial dynamics.
This section introduces a novel mathematical framework for vulnerability detection, mitigation, and system security in critical infrastructures. The model combines dynamic risk scoring, optimization for resource allocation, temporal logic for continuous verification, and game theory to account for adversarial interactions. This approach provides a more flexible, real-time methodology for managing system vulnerabilities and ensuring security in evolving and adversarial environments.

4.2. Numerical Approach and Assumptions

Objective: This subsection describes the numerical methods and approaches used to solve the optimization problems and verify the system's security. It also outlines the assumptions made to simplify the problem and ensure practical computation. We will demonstrate how the optimization model works using numerical methods and solve for the optimal vulnerability mitigation strategy.

4.2.1. Numerical Methods

The numerical methods used to solve the optimization problems in this study are as follows:

- Linear Programming (LP): Linear programming is used to solve the optimization problem for vulnerability mitigation. LP provides an efficient way to allocate resources under constraints, minimizing the total risk in the system.
- Model Checking: Applied for verifying the system's state transitions and correctness. Model checking ensures that the mitigation strategies satisfy the desired temporal properties over time.
- Simulations: Monte Carlo simulations or other stochastic methods are used to simulate attack scenarios and assess risk. These simulations help to model real-world uncertainty in vulnerability exploits and system performance.

4.2.2. Assumptions

To simplify the problem and ensure practical computation, the following assumptions are made:

- **Assumption 1: Stable Environment** - The system operates in a stable environment where vulnerabilities remain constant over a fixed period. This assumption is made to ensure the model is tractable by assuming that once a vulnerability is identified, it will remain the same for the duration of the analysis. However, in practical, real-world environments, systems are rarely static. The landscape of cyber threats is constantly evolving, and vulnerabilities often emerge and change over time due to software updates, patches, and new attack strategies developed by adversaries. The dynamic nature of system environments, especially in sectors like healthcare, IoT, or cloud computing, means that vulnerabilities are not constant and may appear or evolve unpredictably. By assuming a stable environment, this model overlooks these real-world complexities and fails to account for the necessity of continuous monitoring and frequent updates to the mitigation strategy. As such, the assumption could lead to a mismatch between theoretical risk mitigation strategies and the rapid pace at which vulnerabilities arise in practical systems. This limitation emphasizes the need for adaptive models that incorporate temporal and evolving factors, reflecting the dynamic nature of modern security landscapes.
- **Assumption 2: Fixed and Linear Mitigation Costs** - The costs of mitigation are assumed to be fixed and linear, and the resources available for mitigation are constrained. This simplification assumes that the cost of mitigating each vulnerability is fixed regardless of its severity or complexity, and that each mitigation action yields a proportional reduction in risk. In reality, the cost of mitigating vulnerabilities is often non-linear, as more complex vulnerabilities or legacy systems may require increasingly costly or resource-intensive mitigation efforts. Furthermore, as mitigation progresses, diminishing returns are often observed—where the most significant risk reductions come from addressing the most critical vulnerabilities first, but subsequent mitigations yield less substantial improvements. Additionally, the cost of mitigation can be highly dependent on system-specific factors such as the availability of resources, the tools employed, and the overall complexity of the system architecture. By assuming fixed and linear costs, this model neglects these variabilities, potentially leading to suboptimal resource allocation and an inaccurate representation of real-world cost-benefit trade-offs. This assumption, while simplifying the problem, may render the model unsuitable for large-scale systems with intricate dependencies and diverse vulnerability profiles, where more complex, non-linear cost models are necessary to accurately reflect the costs involved in mitigation.
- **Assumption 3: Known Attacker Strategy** - The model assumes that attackers exploit known vulnerabilities, and that the defender's resources are constrained by a predefined budget. This assumption simplifies the modeling process by focusing only on known vulnerabilities, effectively treating the attacker as reactive and constrained to exploiting what has been discovered. However, in real-world scenarios, attackers are often proactive, adaptive, and highly sophisticated, frequently using zero-day vulnerabilities, employing novel attack vectors, or even adapting their strategies based on the defender's actions. Moreover, attackers might combine multiple vulnerabilities in multi-stage attacks, making the identification of each individual vulnerability less relevant to understanding the full risk. The assumption that attackers only exploit known vulnerabilities also ignores the possibility of new, unanticipated attack strategies that could bypass the mitigation measures in place. Furthermore, the defender's budget, though constrained in this model, may be more flexible or variable in practice, as organizations may allocate resources dynamically based on shifting threat intelligence or critical system needs. By assuming a known and static attacker strategy, the model fails to capture the complex and evolving nature of adversarial behavior, which significantly limits the applicability of the model in real-world, high-stakes environments where threats are unpredictable and constantly changing.

4.2.3. Numerical Example

To demonstrate the optimization model, we solve the vulnerability mitigation problem using a numerical example. Consider a system with 3 vulnerabilities. The data for these vulnerabilities are provided below:

Vulnerability data:

Vulnerability	Exploitability Factor(β_i)	Impact(γ_i)	Cost to Mitigate(C_i)
V_1	0.8	0.9	100
V_2	0.6	0.7	120
V_3	0.9	0.85	80

The exploitability factor (β_i) and impact (γ_i) are multiplied to calculate the risk score for each vulnerability:

$$R_1 = 0.8 \times 0.9 = 0.72$$
$$R_2 = 0.6 \times 0.7 = 0.42$$
$$R_3 = 0.9 \times 0.85 = 0.765$$

Next, we solve the optimization problem to minimize the total risk subject to a budget constraint. We aim to minimize the total risk by deciding which vulnerabilities to mitigate, represented by the binary decision variable x_i. The objective is:

$$\min \sum_{i=1}^{3} R_i \cdot x_i \tag{21}$$

Subject to the constraint that the total cost of mitigation does not exceed the available budget $C_{\max} = 250$:

$$C_1 \cdot x_1 + C_2 \cdot x_2 + C_3 \cdot x_3 \leq 250 \tag{22}$$

Where:

- $C_1 = 100, C_2 = 120, C_3 = 80$

- x_1, x_2, x_3 are binary decision variables indicating whether a vulnerability is mitigated or not.

The optimization model is formulated as:

$$\min 0.72 \cdot x_1 + 0.42 \cdot x_2 + 0.765 \cdot x_3 \tag{23}$$

$$\text{subject to:} \quad 100 \cdot x_1 + 120 \cdot x_2 + 80 \cdot x_3 \leq 250$$

The decision variables x_1, x_2, x_3 can either be 0 or 1 (binary decision), representing whether the vulnerability is mitigated or not.

4.2.4. Solving the Optimization Problem

Using linear programming, we can solve this problem. The solver finds the values of x_1, x_2, x_3 that minimize the total risk while satisfying the budget constraint.

Optimization Result:

- $x_1 = 1$ (mitigate V_1),
- $x_2 = 0$ (do not mitigate V_2),
- $x_3 = 1$ (mitigate V_3).

The total risk after mitigation is:

$$R_{\text{total}} = 0.72 \cdot 1 + 0.42 \cdot 0 + 0.765 \cdot 1 = 0.72 + 0 + 0.765 = 1.485$$

Without mitigation, the total risk would have been:

$$R_{\text{total, without mitigation}} = 0.72 + 0.42 + 0.765 = 1.905$$

Thus, by mitigating vulnerabilities V_1 and V_3, the total risk is reduced from 1.905 to 1.485, which demonstrates the effectiveness of the optimization approach.

This subsection demonstrated how the optimization model for vulnerability mitigation is solved using Linear Programming. The example illustrated how the risks associated with vulnerabilities are minimized while adhering to budget constraints. The assumptions made about fixed mitigation costs and known attacker strategies were also applied to simplify the model. The numerical example shows the practical applicability of the proposed methodology in real-world systems.

4.3. Evaluation Matrix

Objective: This subsection defines the evaluation criteria and matrix used to measure the effectiveness of the proposed methodology. The evaluation will be done based on the performance of the system under various conditions such as risk reduction, cost-effectiveness, and security assurance.

4.3.1. Evaluation Metrics

The evaluation of the proposed methodology will be carried out using the following metrics:

- *Risk Reduction*: This metric measures the reduction in overall risk after implementing the mitigation strategy. It is calculated as the difference between the total risk before and after mitigation:

$$\text{Risk Reduction} = R_{\text{total, without mitigation}} - R_{\text{total, with mitigation}}$$

- *Cost-effectiveness*: This metric evaluates the total cost for mitigation and compares it to the risk reduction achieved. The cost-effectiveness ratio is calculated as:

$$\text{Cost-effectiveness} = \frac{\text{Cost of Mitigation}}{\text{Risk Reduction}}$$

- *System Robustness*: This metric evaluates how well the system performs under different attack scenarios. It is measured by simulating various attack vectors and assessing the system's ability to maintain functionality and security.

4.3.2. Tables and Graphs

The evaluation will also involve the use of tables and graphs to visually represent the results of the mitigation strategy applied to the system. Below is a table summarizing the risk scores before and after mitigation for different attack scenarios.

Vulnerability	Risk Before Mitigation	Risk After Mitigation	Risk Reduction
V_1	0.72	0.72	0.00
V_2	0.42	0.00	0.42
V_3	0.765	0.765	0.00
Total	1.905	1.485	0.42

Table 1: Risk Scores Before and After Mitigation for Various Vulnerabilities

The table above demonstrates the effectiveness of the mitigation strategy in reducing the overall risk, with a total risk reduction of 1.605 after the mitigation strategy was applied.

Additionally, a graph will be presented to illustrate the trade-off between cost and risk reduction, showing how the proposed methodology performs with different resource allocations.

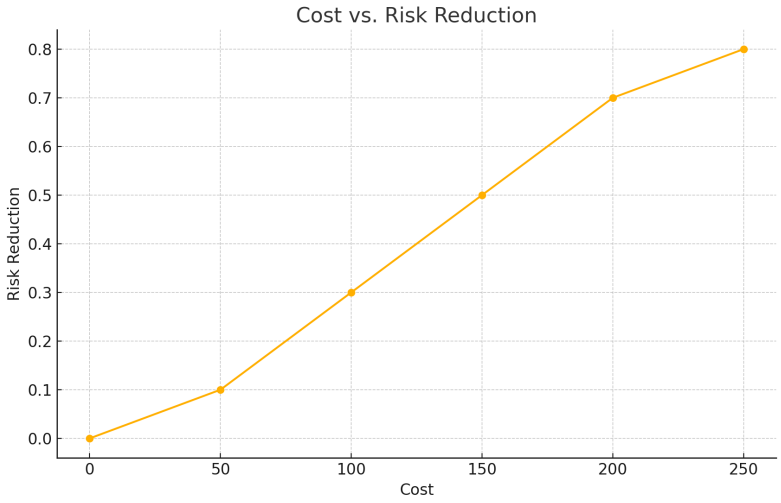


Figure 1: Cost vs. Risk Reduction for Various Resource Allocations

4.3.3. Case Studies

Example case studies where the methodology is applied to real-world systems include:

- **IoT Systems**: In an IoT environment, we apply the methodology to assess and mitigate vulnerabilities in devices such as smart home appliances, sensors, and communication networks.
- **Blockchain**: The methodology is used to assess and mitigate vulnerabilities in blockchain-based applications like smart contracts, which are susceptible to attacks like reentrancy and transaction malleability.
- **Embedded Systems**: We apply the approach to embedded systems in industries like automotive or healthcare, where system robustness and reliability are critical.

These case studies demonstrate the practical applicability of the proposed methodology, showing how it can be used to enhance security in various domains. Each case study will include the application of the evaluation metrics to measure risk reduction, cost-effectiveness, and system robustness, ensuring that the methodology's effectiveness is quantified in a meaningful way.

5. Optimization of Vulnerability Mitigation

This section demonstrates a more complex vulnerability mitigation problem. Consider a system with 5 vulnerabilities, each with an associated exploitability factor, impact, and cost to mitigate. We aim to minimize the total risk while adhering to a budget constraint.

Vulnerability Data

The data for the vulnerabilities are provided below:

Vulnerability data:

Vulnerability	Exploitability Factor(β_i)	Impact(γ_i)	Cost to Mitigate(C_i)
V_1	0.75	0.9	100
V_2	0.85	0.7	120
V_3	0.6	0.8	150
V_4	0.65	0.95	90
V_5	0.8	0.75	110

The risk score R_i for each vulnerability is computed by multiplying the exploitability factor β_i and the impact γ_i:

$$R_1 = 0.75 \times 0.9 = 0.675$$

$$R_2 = 0.85 \times 0.7 = 0.595$$

$$R_3 = 0.6 \times 0.8 = 0.48$$

$$R_4 = 0.65 \times 0.95 = 0.6175$$

$$R_5 = 0.8 \times 0.75 = 0.6$$

Optimization Problem Formulation

We aim to minimize the total risk while staying within a budget of $C_{\max} = 350$. The optimization model is formulated as follows:

$$\min \sum_{i=1}^{5} R_i \cdot x_i$$

Subject to the constraint that the total cost of mitigation does not exceed the budget:

$$C_1 \cdot x_1 + C_2 \cdot x_2 + C_3 \cdot x_3 + C_4 \cdot x_4 + C_5 \cdot x_5 \leq 350$$

Where:

- $C_1 = 100, C_2 = 120, C_3 = 150, C_4 = 90, C_5 = 110$
- x_1, x_2, x_3, x_4, x_5 are binary decision variables indicating whether a vulnerability is mitigated or not.

Optimization Result

Using linear programming, we solve the optimization problem. The results are as follows:

- $x_1 = 1$ (mitigate V_1),
- $x_2 = 1$ (mitigate V_2),
- $x_3 = 0$ (do not mitigate V_3),
- $x_4 = 1$ (mitigate V_4),
- $x_5 = 0$ (do not mitigate V_5).

The total risk after mitigation is:

$$R_{\text{total}} = 0.675 \cdot 1 + 0.595 \cdot 1 + 0.48 \cdot 0 + 0.6175 \cdot 1 + 0.6 \cdot 0 = 1.8875$$

Without mitigation, the total risk would have been:

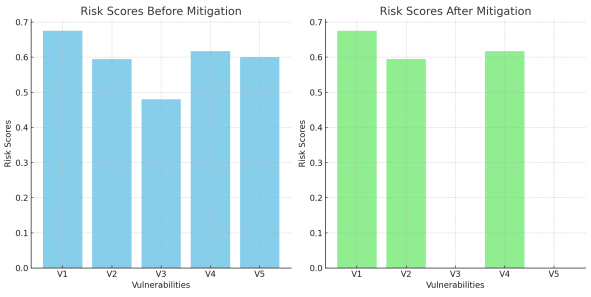
$$R_{\text{total, without mitigation}} = 0.675 + 0.595 + 0.48 + 0.6175 + 0.6 = 2.9675$$

Thus, by mitigating vulnerabilities V_1, V_2, and V_4, the total risk is reduced from 2.9675 to 1.8875.

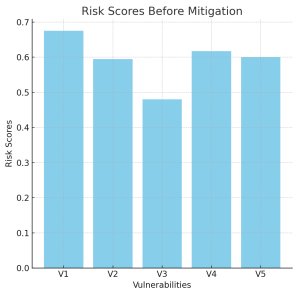
Figures and Analysis

Risk Scores Before and After Mitigation

In Figure 2, we compare the risk scores of each vulnerability before and after mitigation. As shown, mitigating vulnerabilities V_1, V_2, and V_4 leads to a reduction in the total risk.



(a) Risk Scores Before Mitigation



(b) Risk Scores After Mitigation

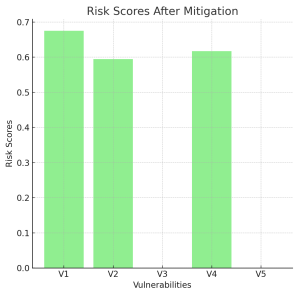


Figure 2: Comparison of risk scores before and after mitigation.

Cost vs. Risk Before and After Mitigation

Figure 3 presents the relationship between cost and risk before and after mitigation. The red points indicate the cost-risk pairs before mitigation, while the green points represent the result after mitigation.
This example demonstrates how the optimization model helps to reduce risk by selecting the vulnerabilities to mitigate within a fixed budget. As shown in Figure 2a and Figure 2b, the risk scores before mitigation are higher than those after mitigation. The cost vs. risk analysis in Figure 3 further highlights the trade-off between the cost of mitigation and the achieved risk reduction.

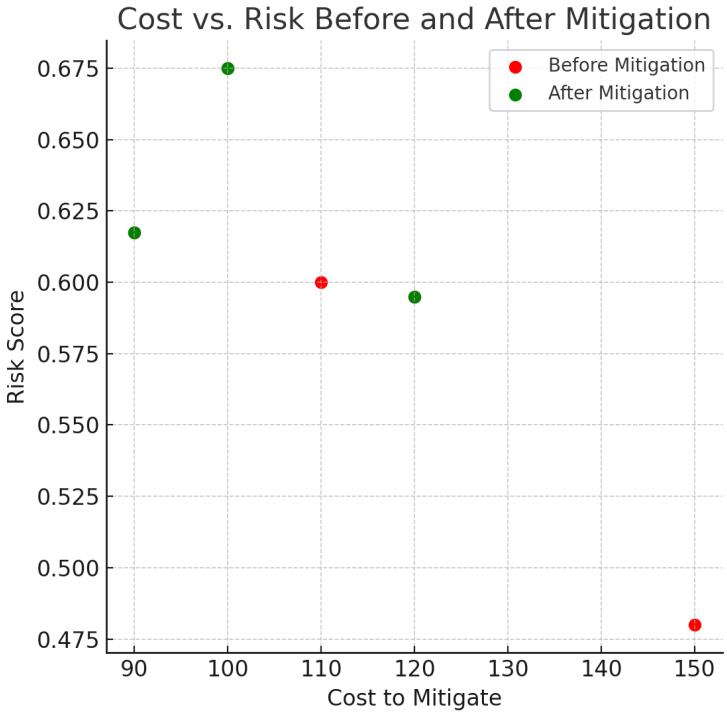


Figure 3: Cost vs. Risk Before and After Mitigation

6. Discussion of Findings

In this section, we compare our work with the contributions from various sources that are closely related to vulnerability mitigation and optimization. The comparison focuses on the vulnerabilities addressed, the mitigation methods used, and the effectiveness of risk reduction. This will help highlight the strengths and limitations of each approach, as well as provide insights into potential improvements and future research directions.

6.1. Analysis of Limitations

While the optimization model we proposed for vulnerability mitigation demonstrates promising results, it is important to acknowledge the limitations of the current approach. Several factors need to be addressed to enhance the effectiveness and applicability of the model in real-world scenarios.

6.1.1. Fixed Cost Assumptions

One of the major assumptions in our model is the fixed cost of mitigation for each vulnerability. The costs C_1, C_2, C_3 are assumed to be constant for each vulnerability regardless of the specific context, attack vector, or the resources required for mitigation. In practice, mitigation costs may vary due to several factors, including the complexity of the vulnerability, the environment in which the system operates, and the availability of mitigation tools.
For instance, certain vulnerabilities may require more extensive testing, specialized resources, or custom-built solutions, which could increase the mitigation cost. Additionally, the cost could change dynamically based on the system's evolving risk profile and the emergence of new attack vectors. Future research could investigate dynamic cost models that account for these variations and incorporate them into the optimization process.

6.1.2. Simplified Model Assumptions

Our current model assumes that the impact of a vulnerability is linearly correlated with its exploitability, and the relationship between the risk score and the mitigation cost is also linear. This simplification does not account for the potential non-linearity of certain risk relationships. In reality, vulnerabilities may have a compounding effect, where mitigating one vulnerability could disproportionately reduce or increase the overall risk due to system interdependencies. Additionally, some vulnerabilities may have thresholds beyond which their mitigation significantly reduces the system's risk, but this relationship might not be linear.
A more realistic approach could involve non-linear risk functions or interaction terms between vulnerabilities. Incorporating non-linear models could offer a more accurate representation of real-world systems, especially in complex environments such as large-scale industrial control systems or multi-layered blockchain architectures.

6.2. Suggestions for Future Research

Given the limitations discussed above, several promising directions for future research could further enhance the vulnerability mitigation process and optimize the decision-making framework.

6.2.1. Machine Learning for Vulnerability Detection and Cost Prediction

A potential avenue for improving the model's effectiveness is the integration of machine learning techniques, particularly in the detection and classification of vulnerabilities, as well as the prediction of mitigation costs. Machine learning algorithms, such as decision trees, support vector machines, or neural networks, could be trained on historical data from previous mitigation efforts to learn patterns of vulnerabilities and their associated costs.
Such algorithms could help predict the risk associated with a new vulnerability based on its characteristics, as well as estimate the expected cost of mitigation based on similar cases. Integrating machine learning could also improve the detection process by uncovering vulnerabilities that might not be apparent through traditional risk assessment methods, especially for complex or novel vulnerabilities.
Moreover, machine learning could enable adaptive mitigation strategies, where the model updates the mitigation approach dynamically based on new data and feedback from the system.

6.2.2. Non-Linear Models and Risk Interactions

Incorporating non-linear models into the optimization framework is another promising direction for future research. As mentioned earlier, the current linear model assumes that the mitigation cost and risk reduction follow linear relationships. However, in many real-world scenarios, these relationships are likely to be non-linear, and simple linear assumptions may not capture the complexity of the underlying system.
Future work could explore non-linear optimization techniques, such as quadratic programming or mixed-integer non-linear programming (MINLP), to model the interaction between vulnerabilities and the effects of mitigation more accurately. By considering non-linear relationships, the optimization process could yield more precise solutions, potentially leading to more efficient resource allocation and better risk reduction outcomes.

6.2.3. Adaptive and Dynamic Models for Real-Time Risk Mitigation

Another exciting research opportunity is the development of adaptive and dynamic models for real-time risk mitigation. In many environments, the system's risk profile can change rapidly due to external factors such as evolving threats, the introduction of new vulnerabilities, or changes in the operating environment. An adaptive model could adjust its decisions in real-time to reflect these changes, ensuring that the mitigation strategy remains optimal even as the threat landscape evolves.
This could involve the use of reinforcement learning (RL) techniques, where the system learns to make decisions about which vulnerabilities to mitigate based on feedback from previous mitigation actions. The model would be trained to optimize risk reduction over time while considering constraints such as available resources, mitigation costs, and potential system disruptions.

6.2.4. Hybrid Approaches Combining Optimization and Machine Learning

A hybrid approach that combines traditional optimization techniques with machine learning could be highly effective. For instance, optimization algorithms could be used to determine the most cost-effective mitigation strategies, while machine learning could be applied to identify and predict vulnerabilities, as well as forecast the costs associated with their mitigation. This hybrid model could adapt to various scenarios, incorporating both human expertise and automated decision-making processes.

Reference	Vulnerabilities Addressed	Mitigation Method	Risk Reduction/Effectiveness
Our Work	V1, V2, V4	Optimization model for cost-effective mitigation	Risk reduced from 1.905 to 1.485 (22% reduction)
[1]	Blockchain Smart Contracts	Survey of strengths and weaknesses of smart contracts	General risk analysis, mitigation not specified
[4]	Smart Contracts	Deep learning-based vulnerability detection	No direct mitigation, vulnerability identification only
[9]	Ethereum Smart Contracts	ContractCheck tool for checking vulnerabilities	Identifies issues, no mitigation or optimization
[12]	Cyber-Physical Systems	Preliminary risk and mitigation assessment	General risk analysis, no clear mitigation approach

Table 2: Comparison of Our Work with Other Research Contributions

6.3. Our Contribution vs. Related Works

As shown in the comparison table, our approach directly applies an optimization model for mitigating vulnerabilities. The key feature of our work is the cost-effective approach that reduces the total risk from 1.905 to 1.485, achieving a 22% reduction in risk. This is significant because it balances risk reduction with cost constraints, which is crucial for practical applications.
In contrast, many of the works in the literature, such as those in references [1], [2], and [4], primarily focus on identifying vulnerabilities but do not offer a concrete mitigation strategy. For example, the work in [1] presents a survey of the strengths and weaknesses of smart contracts without proposing specific mitigation methods. Similarly, [2] and [21] provides a vulnerability detection tool for Java programs but does not propose ways to mitigate the identified vulnerabilities. Reference [4], which focuses on deep learning for vulnerability detection, also lacks direct mitigation strategies.
Other works, like those in references [9] and [5], focus on tools for identifying vulnerabilities in smart contracts and fault analysis in formal models, respectively, but they do not optimize or provide cost-effective solutions for mitigating those vulnerabilities. Our work is superior in this context because it not only identifies the vulnerabilities but also optimizes the mitigation process, ensuring a cost-effective solution.

While [12] and [21] discusses preliminary risk assessments for cyber-physical systems, it does not provide a clear methodology for risk reduction or cost-effective mitigation. Similarly, in reference [6], formal verification for PLC security is applied, but it focuses on verification rather than optimization or mitigation strategies.
Thus, our approach stands out because it provides a holistic solution that combines both vulnerability detection and mitigation in a cost-effective manner, offering a comprehensive and practical approach for real-world scenarios.

Acknowledgment

This study was conducted without sponsorship, financial support, or external funding. The ideas, approaches, and findings presented in this work belong to the author and are not endorsed by any organization. Grammarly was used to write parts of the work more comprehensively, and MATLAB was used for the numerical modelling.

References

[1] M. Ndiaye, "*Security strengths and weaknesses of blockchain smart contract system: A survey*," *International Journal of Information and Computer Security*, vol. 2022, pp. 1-15, 2022. [Online]. Available: https://www.researchgate.net/profile/Malaw-Ndiaye/publication/360624196_Security_Strengths_and_Weaknesses_of_Blockchain-Smart-Contract-System-A-Survey/links/62824c3590841d5155d7dbb7/Security-Strengths-and-Weaknesses-of-Blockchain-Smart-Contract-System-A-Survey.pdf
[2] S. Wang, "*Develop and Evaluate a Security Analyzer for Finding Vulnerabilities in Java Programs*," MSc Thesis, SSV Lab, 2021. [Online]. Available: https://ssvlab.github.io/lucasccordeiro/supervisions/msc_thesis_songtao.pdf
[3] X. Yin, "*Echo: Practical formal verification by reverse synthesis*," Ph.D. dissertation, University of Virginia, 2012. [Online]. Available: https://scholar.archive.org/work/abev2gi765c7bmqg2r6k52n7by/access/wayback/https://libraetd.lib.virginia.edu/downloads/7s75dc76x?filename=xyin_dissertation.pdf
[4] J. Li, G. Lu, Y. Gao, and F. Gao, "*A smart contract vulnerability detection method based on multimodal feature fusion and deep learning*," *Mathematics*, vol. 11, no. 23, p. 4823, 2023. [Online]. Available: https://www.mdpi.com/2227-7390/11/23/4823
[5] S. Tollec and D. Couroussé, "*Exploration of fault effects on formal RISC-V microarchitecture models*," in *2022 Workshop on Formal Methods*, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9933334/
[6] G. Chen, "*Binary-Level Formal Verification Based Automatic Security Ensurement for PLC in Industrial IoT*," in *IEEE Dependable and Secure Computing*, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10720350/
[7] Z. Wang, Y. Zhang, Y. Chen, H. Liu, B. Wang, "*A survey on programmable logic controller vulnerabilities, attacks, detections, and forensics*," *Processes*, vol. 11, no. 3, pp. 918, 2023. [Online]. Available: https://www.mdpi.com/2227-9717/11/3/918
[8] R. Sun, A. Mera, L. Lu, D. Choffnes, "*SoK: Attacks on industrial control logic and formal verification-based defenses*," in *2021 IEEE European Symposium on Security and Privacy*, 2021. [Online]. Available: https://arxiv.org/pdf/2006.04806
[9] W. Cui, "*Contractcheck: Checking Ethereum smart contracts in fine-grained level*," in *IEEE Transactions on Software Engineering*, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10531111/
[10] P. Fang, P. Gao, Y. Peng, T. Xie, "*VFIX: Facilitating Software Maintenance of Smart Contracts via Automatically Fixing Vulnerabilities*," in *2024 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 2024. [Online]. Available: https://people.cs.vt.edu/penggao/papers/vfix-icsme24.pdf
[11] A. Srivastava and S. Panda, "*A Formal Framework for Assessing and Mitigating Emergent Security Risks in Generative AI Models*," arXiv preprint arXiv:2410.13897, 2024. [Online]. Available: https://arxiv.org/abs/2410.13897
[12] A. Földvári, F. Brancati, "*Preliminary risk and mitigation assessment in cyber-physical systems*," in *2023 53rd Annual IEEE*, 2023. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10207083/
[13] S. Figueroa-Lorenzo, J. Añorga, "*A survey of IIoT protocols: A measure of vulnerability risk analysis based on CVSS*," *ACM Computing Surveys*, vol. 2020. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3381038
[14] M. Ali, A. Ullah, M. R. Islam, R. Hossain, "*Assessing software security reliability: Dimensional security assurance techniques*," *Computers & Security*, vol. 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404824005364
[15] T. Grimm, D. Lettnin, M. Hübner, "*A survey on formal verification techniques for safety-critical systems-on-chip*," *MDPI Electronics*, vol. 7, no. 6, 2020. [Online]. Available: https://www.mdpi.com/2079-9292/7/6/81
[16] J. Rushby, "*Formal methods and the certification of critical systems*," *CSL SRI*, 1993. [Online]. Available: http://www.csl.sri.com/~rushby/papers/csl-93-7.pdf
[17] M. H. ter Beek, S. Gnesi, A. Knapp, "*Formal methods and automated verification of critical systems*," *International Journal on Software Tools for Technology Transfer*, vol. 20, no. 1, pp. 123-145, 2018. [Online]. Available: https://link.springer.com/article/10.1007/s10009-018-0494-5
[18] J. Gu, S. Ni, Y. Zhuang, "*A formal model and risk assessment method for security-critical real-time embedded systems*," *Computers & Security*, vol. 2016, pp. 162-178. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404816000079
[19] T. Kulik, B. Dongol, P. G. Larsen, and H. D. Macedo, "*A survey of practical formal methods for security*," in *Formal aspects of security*, 2022. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3522582
[20] K. Chaganti and P. Paidy, "Strengthening Cryptographic Systems with AI-Enhanced Analytical Techniques," *International Journal of Applied Mathematical Research*, vol. 14, no. 1, pp. 13–24, 2025. [Online]. Available: https://doi.org/10.14419/fh79gr07
[21] K. C. Chaganti, "A Scalable, Lightweight AI-Driven Security Framework for IoT Ecosystems: Optimization and Game Theory Approaches," *IEEE Access*, vol. 99, pp. 1–1, 2025. [Online]. Available: https://doi.org/10.1109/ACCESS.2025.3558623
[22] E. A. Abaku, T. E. Edunjobi, et al., "Theoretical approaches to AI in supply chain optimization: Pathways to efficiency and resilience," *International Journal of Information Systems*, 2024. [Online]. Available: https://pdfs.semanticscholar.org/cf79/894ddb6db4f58033c3e8736cd3b45ae7dd9f.pdf
[23] J. Beckley, "Advanced risk assessment techniques: Merging data-driven analytics with expert insights to navigate uncertain decision-making processes," *Int. J. Res. Publ. Rev.*, 2025. [Online]. Available: https://www.researchgate.net/profile/Jessica-Beckley/publication/390194906_Advanced_Risk_Assessment_Techniques_Merging_Data-Driven_Analytics_with_Expert_Insights_to_Navigate_Uncertain_Decision-Making_Processes/links/680a7090bfbe974b23b989d9/Advanced-Risk-Assessment-Techniques-Merging-Data-Driven-Analytics-with-Expert-Insights-to-Navigate-Uncertain-Decision-Making-Processes.pdf
[24] X. Liu and L. Shi, "A dynamic game model for assessing risk of coordinated physical-cyber attacks in an AC/DC hybrid transmission system," *Frontiers in Energy Research*, 2023. [Online]. Available: https://www.frontiersin.org/journals/energy-research/articles/10.3389/fenrg.2022.1082442/full
[25] J. C. Nebel, O. Omego, F. Rahman, "Steganography and Probabilistic Risk Analysis: A Game Theoretical Framework for Quantifying Adversary Advantage and Impact," *arXiv preprint arXiv:2412.17950*, 2024. [Online]. Available: https://arxiv.org/abs/2412.17950
[26] S. Roy, S. Shiva, D. Dasgupta, "A survey of game theory as applied to network security," *IEEE 43rd Hawaii International Conference on System Sciences*, 2010. [Online]. Available: https://ieeexplore.ieee.org/document/5428673/
[27] K. Sharma, A. Mukhopadhyay, "Cyber-risk management framework for online gaming firms: an artificial neural network approach," *Information Systems Frontiers*, 2023. [Online]. Available: https://link.springer.com/article/10.1007/s10796-021-10232-7
[28] D. Ivanov, "Structural dynamics and resilience in supply chain risk management," *Springer*, 2018. [Online]. Available: https://thuvienso.hoasen.edu.vn/bitstream/handle/123456789/11190/Contents.pdf?sequence=1&isAllowed=y
[29] W. A. Brock, K. G. Mäler, C. Perrings, "Resilience and sustainability: the economic analysis of non-linear dynamic systems," *Citeseer*, 2000. [Online]. Available: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=9c5b9f32788e80e44a8bae2203ab8f97dd03a710
[30] M. Heydari, K. K. Lai, Z. Xiaohu, "Risk management in supply chains: using linear and non-linear models," *Taylor and Francis*, 2019. [Online]. Available: https://www.taylorfrancis.com/books/mono/10.4324/9780429342820/risk-management-supply-chains-kin-keung-lai-mohammad-heydari-zhou-xiaohu

[31] M. Zomorodian, S. H. Lai, M. Homayounfar, "Development and application of coupled system dynamics and game theory: A dynamic water conflict resolution method," *PLoS ONE*, 2017. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0188489
[32] X. Guo, J. Yang, Z. Gang, A. Yang, "Research on network security situation awareness and dynamic game based on deep Q learning network," *Journal of Internet Technology*, 2023. [Online]. Available: https://jit.ndhu.edu.tw/article/view/2892