International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

AI-Driven Contextual Risk Scoring Framework for PII-Aware Penetration Testing

Umashankara Kalaiah

Staff Software Engineer
Umashankarak[at]gmail.com

Abstract: ***Modern automated penetration testing tools effectively identify application-layer vulnerabilities such as SQL Injection, Insecure Direct Object Reference (IDOR), Cross-Site Scripting (XSS), and authentication weaknesses. However, traditional vulnerability scoring systems emphasize technical severity and often overlook the business impact of exposed Personally Identifiable Information (PII), leading to suboptimal prioritization in enterprise environments. This paper presents an AI-Driven Contextual Risk Prioritization Framework for PII-aware penetration testing. The framework integrates AWS Security Agent outputs, API schema metadata, and a PII sensitivity taxonomy to construct enriched vulnerability context. An AI-based reasoning layer generates a Contextual Adjustment Factor (ACF), which is combined with PII sensitivity, exploitability, and exposure to produce a normalized risk score. Experimental results show improved prioritization of high-impact, data-sensitive vulnerabilities compared to CVSS-based approaches, while reducing the priority of lower-impact findings.***

Keywords: Penetration Testing, PII Security, Risk Scoring, Artificial Intelligence, Vulnerability Prioritization

1. Introduction

Enterprise applications increasingly process sensitive user data, including financial, identity, and personal information. While automated penetration testing tools provide vulnerability detection at scale, they lack contextual understanding of data sensitivity and real-world impact.

Traditional models such as CVSS [1] assign severity scores based on exploitability and impact but do not adequately consider:

- Sensitivity of exposed PII
- API-level data context
- Multi-stage attack chains
- Real-world exposure pathways

As a result, vulnerabilities affecting highly sensitive data may not be prioritized appropriately.

This paper introduces a hybrid framework that integrates AI-driven contextual reasoning with deterministic risk scoring to improve vulnerability prioritization in enterprise environments.

2. Related Work

2.1 CVSS-Based Vulnerability Scoring Systems

The Common Vulnerability Scoring System (CVSS) [2] is the most widely adopted framework for quantifying vulnerability severity. It evaluates technical factors such as attack vector, complexity, privileges required, and impact on confidentiality, integrity, and availability. While CVSS provides a standardized and comparable scoring mechanism, it remains context-agnostic, lacking awareness of application-level data sensitivity and business impact. Consequently, vulnerabilities with identical CVSS scores may represent significantly different real-world risks depending on exposed data assets.

2.2 Risk-Based Vulnerability Management (RBVM)

Risk-Based Vulnerability Management (RBVM) [4] enhances CVSS by incorporating environmental and organizational factors such as asset criticality, threat intelligence, exploit maturity, and exposure level. Although RBVM improves prioritization accuracy, most implementations remain infrastructure-centric, focusing on hosts and systems rather than modern API-driven applications. As a result, fine-grained prioritization based on sensitive data exposure (e.g., PII or financial records) is still limited.

2.3 AI in Cybersecurity Risk Analysis

Artificial Intelligence and Machine Learning techniques [8], [9] are widely used in cybersecurity for anomaly detection, intrusion detection, malware classification, and security automation. Recent advances in Large Language Models (LLMs) have enabled contextual understanding of security logs and vulnerability reports. However, existing approaches primarily focus on detection and summarization rather than integrating AI reasoning into quantitative risk scoring models for vulnerability prioritization.

2.4 PII-Aware Security and Data Protection Research

Data protection frameworks such as GDPR [5], HIPAA [6], and PCI DSS [7] emphasize the identification and safeguarding of sensitive personal information. Existing research focuses on data discovery, classification, and compliance enforcement. However, limited work connects PII classification directly with vulnerability assessment outputs from penetration testing tools to improve risk prioritization.

2.5 Research Gap

Despite advances in CVSS scoring, RBVM, AI-based security analytics, and PII governance, there is a lack of

integrated frameworks that combine penetration-test findings, API schema awareness, PII sensitivity modeling, and AI-driven contextual reasoning into a unified risk prioritization model. This paper addresses this gap through an AI-driven PII-aware vulnerability scoring framework.

3. Proposed Framework

3.1 System Architecture

To address the limitations of traditional vulnerability prioritization methods, this paper proposes an AI-Driven PII-Aware Pentesting Framework that combines automated penetration-test findings with contextual business-data awareness and deterministic risk scoring. Unlike conventional severity-based approaches that rely primarily on technical exploitability metrics, the proposed model integrates multiple intelligence sources, including penetration-test findings, API schema metadata, PII sensitivity taxonomy, and AI-assisted contextual reasoning. These inputs are processed through a deterministic risk engine that generates a normalized risk score between 0 and 100. Figure 1 illustrates the system architecture, while Figure 2 presents the end-to-end dataflow pipeline used for risk scoring and ranking.

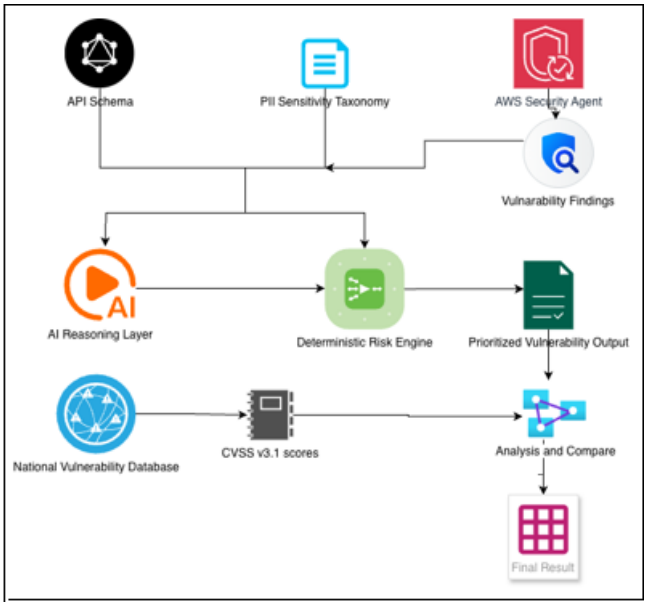

Figure 1: System Architecture

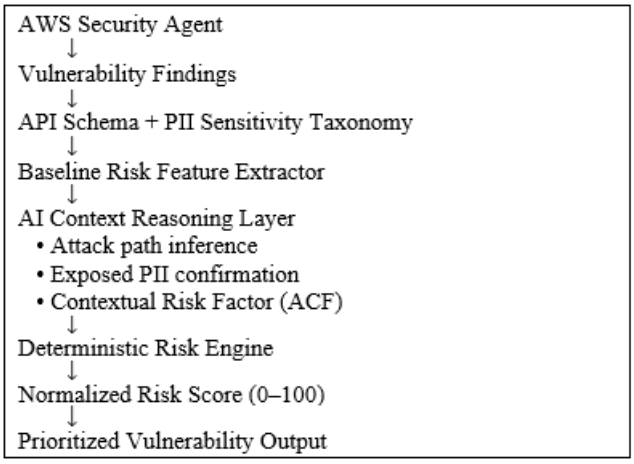

Figure 2: End to End dataflow pipeline

3.2 Input Data Sources

The proposed framework uses three structured input sources to generate context-aware vulnerability prioritization: penetration-test findings, API schema metadata, and PII sensitivity classifications. Together, these inputs enable more accurate scoring than traditional severity-only models.

3.2.1 Penetration Testing Output

The primary input source is vulnerability findings generated by **AWS Security Agent [10]**. AWS Security Agent performs autonomous penetration testing on deployed web applications and APIs, identifying issues such as SQL Injection, IDOR, XSS, authentication weaknesses, and misconfigurations.

The scan results can be exported in structured formats such as JSON and typically include the affected endpoint and vulnerability type. These findings serve as the technical input for the proposed framework.

Example Output

```
{
 "endpoint": "/user/profile",
 "vulnerability": "SQL Injection"
}
```

In practice, these scans are initiated by security engineers, DevSecOps teams, or cloud administrators during release testing or periodic security assessments.

3.2.2 API Schema Metadata

The second input source is API schema metadata, which describes endpoint fields and data objects. This allows the framework to determine what information may be exposed if a vulnerable endpoint is exploited.

Schema metadata is commonly obtained from **OpenAPI / Swagger** specifications, GraphQL schema introspection, API gateways, or internal service documentation.

Example Metadata

```
{
 "/user/profile": {
  "fields": ["user_id", "email", "ssn"]
 }
}
```

This information is typically maintained by software engineering or API platform teams.

3.2.3 PII Sensitivity Taxonomy

The third input source is a business-defined PII sensitivity taxonomy that classifies data elements according to privacy and regulatory impact if exposed. The taxonomy enables the framework to convert field names into sensitivity weights for risk scoring.

Example Taxonomy

```
{
 "ssn": "HIGH",
 "email": "MEDIUM"
}
```

Table 1: Typical Classification Level

Data Field	*Sensitivity*
SSN	HIGH
Credit Card	HIGH
Medical Record	HIGH
Email	MEDIUM
Phone Number	MEDIUM
UserName	LOW

3.3 Context Construction Layer

After collecting raw inputs from penetration-test findings, API schema metadata, and the PII taxonomy, the framework performs a Context Construction Layer. This stage combines multiple sources into a single enriched record before AI reasoning or risk scoring begins.

This layer is fully deterministic and rule-based and does not use AI. Its purpose is to convert raw vulnerability findings into business-aware security context.

The layer performs three tasks: it matches the vulnerable endpoint with API schema metadata, identifies PII fields associated with the endpoint, and generates a short contextual description of likely exposed data.

3.3.1 Derived Context Structure

```
{
 "endpoint": "/user/profile",
 "vulnerability": "SQL Injection",
 "pii_fields": ["ssn", "email"],
 "context": "Endpoint exposes database-backed user profile data"
}
```

This record is created by combining AWS Security Agent [10] findings, API schema fields, and PII taxonomy mapping.

3.4 AI Context Reasoning Layer

The AI layer performs contextual interpretation of vulnerability findings and is responsible for enriching deterministic outputs with semantic understanding. Its primary role includes attack chain inference, identification of exposed Personally Identifiable Information (PII) fields, real-world exploitation reasoning, and generation of a Contextual Risk Factor (ACF). The AI layer does not perform PII classification and does not compute final risk scores; instead, it focuses exclusively on explaining the potential impact of the vulnerability in real-world scenarios.

The layer is typically implemented using a **large language model (LLM) inference endpoint** (e.g., AWS Bedrock, OpenAI API, or self-hosted LLM) with strict prompt constraints and schema-controlled outputs.

The AI reasoning layer is deployed as a microservice. It is intentionally isolated to ensure:

- No direct access to scoring logic
- No mutation of PII taxonomy
- No state persistence between requests

3.4.1 Prompt Engineering Strategy

The AI Context Reasoning Layer uses constrained prompt engineering to restrict the model to contextual impact reasoning and generation of a bounded Contextual Adjustment Factor (ACF $\in$ [0.8, 1.8]) while preventing final risk scoring, severity assignment, or PII reclassification.

> You are a cybersecurity contextual reasoning engine.
>
> Your responsibilities:
> - Infer likely attack chains from the given vulnerability
> - Explain likely business and security impact
> - Evaluate exposure relevance of provided PII fields
> - Generate a Contextual Adjustment Factor (ACF) in the range [0.8, 1.8]
>
> ACF Guidance:
> - 0.8–0.99 = mitigating context lowers impact
> - 1.0 = neutral context
> - 1.01–1.39 = moderate contextual increase
> - 1.40–1.80 = high contextual increase
>
> You must NOT:
> - Assign CVSS severity labels
> - Compute final risk scores
> - Determine remediation priority
> - Invent new PII fields
> - Reclassify the vulnerability
>
> The ACF is only an input to a downstream deterministic scoring engine.
>
> Return valid JSON only.

3.4.2 AI Input Example

```
{
 "endpoint": "/user/profile",
 "vulnerability": "SQL Injection",
 "http_context": {
  "method": "GET",
  "parameters": ["id"]
 },
 "pii_fields": ["ssn", "email"],
 "db_context": "user_profile_table"
}
```

3.4.3 AI Output Example

```
{
 "exposed_pii": ["ssn", "email"],
 "attack_chain": [
 "SQL Injection → database extraction → user table access"
 ],
 "risk_reasoning": "SQL Injection enables direct extraction of sensitive PII from backend systems.",
 "contextual_risk_factor (ACF)": 1.5
}
```

3.4.4 Key Constraints on the AI Layer

- AI is responsible only for semantic reasoning and impact explanation
- AI does **not** perform PII detection or classification
- AI does **not** compute final risk scores
- AI output is used as supporting context for downstream deterministic scoring

3.5 PII Composite Sensitivity Model

To capture both the sensitivity and quantity of exposed Personally Identifiable Information (PII), the framework uses a composite sensitivity model that accounts for the highest-value data element as well as additional exposed fields.

$$PII_{Composite} = \min\left(7, S_{max} + \alpha(N_{pii} - 1)\right)$$

Where:

- S_{max} : maximum sensitivity score among exposed PII fields (range: 1–5)
- N_{pii}: number of unique exposed PII fields
- $\alpha = 0.25$: incremental weight assigned to each additional PII field beyond the first

This formulation ensures that exposure of highly sensitive data (e.g., SSN or passport number) receives higher priority, while broader multi-field exposures also increase risk. The final value is capped at 7 to prevent excessive inflation and maintain balance with other scoring components.

3.6 Exploitability and Exposure Model

To incorporate real-world attack feasibility into the scoring framework, the model evaluates both the inherent exploitability of the vulnerability type and the level of access required to reach the affected asset. This ensures that vulnerabilities that are easier to exploit or exposed to a broader audience receive higher prioritization.

The exploitability component reflects the technical ease and potential impact of common vulnerability classes, while the exposure component measures how accessible the vulnerable resource is from an attacker's perspective.

3.6.1 Exploitability Mapping

Table 2: Exploitability Mapping

Vulnerability Type	Score
SQL Injection	3
IDOR	3
XSS	2
Brute Force	2

Higher exploitability scores are assigned to vulnerabilities that can directly compromise data confidentiality or enable unauthorized access with minimal attacker effort. For example, SQL Injection and IDOR may directly expose backend records or sensitive user data, whereas XSS and brute-force attacks often require additional conditions or user interaction.

3.6.2 Exposure Model

Table 3: Exposure Model

Access Type	Score
Public	3
Authenticated	2
Internal	1

The exposure score reflects the reachability of the vulnerable endpoint. Publicly accessible assets receive the highest score because they are reachable by external attackers without prior access. Authenticated endpoints require valid credentials, while internal systems typically require network-level access and therefore receive lower exposure values.

Together, the exploitability and exposure factors provide an operational view of how likely a vulnerability is to be abused in practice.

3.7 Risk Scoring Function

The final risk score is calculated by combining data sensitivity, technical exploitability, exposure level, and AI-derived contextual impact into a single normalized metric. This formulation enables consistent prioritization across different vulnerability types while preserving interpretability of each contributing factor.

The raw score is defined as:

$$RS = \frac{PIIcomposite \times Ex \times Exp \times ACF}{113.4} \times 100$$

Where:
RS : Risk Score
$PIIcomposite$: composite sensitivity score of exposed PII fields (Section 3.4)
Ex : Exploitability - vulnerability exploitability score.
Exp : Exposure - access exposure score.
ACF: AI-generated Contextual Adjustment Factor, bounded within [0.8,1.8]

To normalize results to a percentage scale from 0 to 100, the framework divides by the maximum possible raw score:

MaxRaw = 7 × 3 × 3 × 1.8 = 113.4

Here, the value 7 represents the capped maximum PII composite score, 3 is the highest exploitability score, 3 is the highest exposure score, and 1.8 is the upper bound of the ACF range.

This normalization ensures that the highest theoretically possible risk condition maps to a score of 100, allowing easier comparison, thresholding, and prioritization across findings.

3.8 Proposed Framework Score range

The proposed framework produces a **normalized risk score in the range of 0 to 100**, enabling consistent comparison across vulnerabilities with varying technical and contextual characteristics.
It is important to note that a **lower score does not indicate the absence of risk**. Instead, it reflects that the vulnerability has **lower relative business impact within this model**, particularly in cases where sensitive PII exposure or high-impact contextual factors are not present. Such findings may still require remediation based on organizational policies.

The scoring model prioritizes vulnerabilities based on **data sensitivity, exploitability, exposure, and contextual reasoning**, rather than purely technical severity. As a result, higher scores correspond to vulnerabilities with greater potential for real-world data compromise.

Based on empirical calibration, the following priority bands are used:

Table 5: Proposed Framework Score Ranges

Score Range	*Risk Level*	*Operational Impact*
0 - 19	Low	**Lower prioritization:** Vulnerabilities with minimal to no PII exposure. Minimizes "Alert Fatigue"
20 - 39	Medium	**Monitored Exposure:** Risks involving low-sensitivity PII. Represents a "Defense-in-Depth" concern rather than an immediate data breach threat.
40 - 59	High	**Contextual Sensitivity:** Mid-tier PII combined with public-facing endpoints. Requires scheduled remediation to prevent data harvesting or credential stuffing.
60 - 100	Critical	**Immediate Privacy Threat:** High-value PII combined with high-exploitability vulnerabilities.

Scores above **60** typically represent vulnerabilities involving **high-sensitivity data exposure, public accessibility, or strong attack-chain potential**, and are therefore classified as *Critical Impact*.

This interpretation ensures that the scoring framework reflects **relative enterprise risk prioritization**, rather than implying absolute absence or presence of security risk.

3.8 Working Example

To illustrate the application of the proposed scoring model, consider a representative vulnerability identified by AWS Security Agent:

- **Endpoint:** /user/profile
- **Vulnerability:** SQL Injection
- **Exposed PII Fields:** SSN, Email

Step 1: Compute PII Composite Score
From the PII sensitivity taxonomy:

- SSN → 5 (High)
- Email → 3 (Medium)

Thus:
Smax = 5, Npii = 2 and alpha = 0.25

$$PIIComposite = \min(7{,}5 + 0.25 \times (2 - 1)) = 5.25$$

Step 2: Assign Exploitability Score
From Table 2:

SQL Injection = 3

Step 3: Assign Exposure Score
The endpoint is publicly accessible:

Exposure = 3

Step 4: Apply AI Contextual Adjustment Factor (ACF)
From AI reasoning:

- Direct database access with high-value PII
- Attack chain: SQL Injection → data extraction

Thus:
ACF = **1.6**

Step 5: Compute Raw Risk Score

$$RawScore = 5.25 \times 3 \times 3 \times 1.6 = 72$$

Step 6: Normalize to Final Score

$$RiskScore = (72/113.4) \times 100 = 63.5$$

4. Experimental Evaluation

To validate the proposed framework, a comparative experiment was conducted using 10 realistic web-application vulnerabilities representing common enterprise penetration-testing findings. Each vulnerability was evaluated using:

- CVSS v3.1 Base Score – standardized technical severity score
- Proposed PII-Aware Risk Score – contextual enterprise risk score [Section 3.8]

The objective was to compare traditional severity scoring with business-impact-aware prioritization.

4.1 CVSS v3.1 Scoring Background

The **Common Vulnerability Scoring System (CVSS) v3.1** [1] is an industry-standard framework maintained by FIRST for rating vulnerability severity.

The CVSS v3.1 base scores used in this study were derived using the standard CVSS calculator provided by FIRST [1]. For each vulnerability, base metrics including Attack Vector, Attack Complexity, Privileges Required, User Interaction, Scope, and impact on Confidentiality, Integrity, and Availability were assigned based on realistic exploitation scenarios consistent with OWASP guidelines [3]. The resulting base scores were used as the reference technical severity for comparison with the proposed framework.

Table 4: CVSS v3.1 Score Ranges

CVSS Score	*Severity*
0.1 - 3.9	Low
4.0 - 6.9	Medium
7.0 - 8.9	High
9.0 - 10.0	Critical

Thus, 9.8 is near-maximum severity, and 10.0 is the theoretical maximum.

4.3 Evaluation Dataset

The evaluation dataset consists of **10 representative vulnerability findings** modeled after real-world outputs from automated penetration testing tools such as AWS Security Agent [10]. The selected vulnerabilities cover a diverse set of common application-layer security issues, including injection flaws, access control weaknesses, authentication issues, client-side vulnerabilities, and data exposure scenarios.

The dataset includes the following vulnerability categories:

- SQL Injection
- Insecure Direct Object Reference (IDOR)

- Token Leakage
- Cross-Site Scripting (XSS)
- Brute-force Authentication
- Public Storage Exposure
- Weak Password Policy
- Sensitive Logging Exposure
- Internal Debug Exposure

Each test case is mapped to a realistic application endpoint and enriched with contextual information derived from API schema metadata and PII sensitivity taxonomy. This ensures that the dataset reflects both technical vulnerability characteristics and potential data exposure impact.

For each vulnerability, the following attributes are evaluated:
- **Affected Endpoint** – API or application path identified as vulnerable
- **Exposed PII Fields** – Sensitive data elements associated with the endpoint
- **CVSS v3.1 Score** – Standardized technical severity score
- **Raw Framework Score** – Unnormalized score based on PIIComposite, exploitability, exposure, and contextual factors
- **Final Normalized Score** – Scaled score (0–100) used for prioritization

The dataset is intentionally designed to include a mix of:
- High-severity vulnerabilities with sensitive data exposure
- Moderate vulnerabilities with contextual risk implications
- Lower-impact findings with minimal or no PII involvement

Source code available at:
https://github.com/umashankarak/pii-aware-pentest-demo

Table 6: Evaluation Results

Vulnerability	Endpoint	PII Fields	CVSS 3.1	Raw Score	Risk Score	Priority
SQL Injection	/user/profile	SSN, Email	9.8	72	63.5	Critical
IDOR	/payment	Credit Card	8.1	72	63.5	Critical
Token Leakage	/auth/token	Auth Token, Email	7.5	67.5	59.5	High
Brute Force Login	/login	Email	6.5	6	5.3	Low
XSS	/search	None	6.1	0	0	Low
Public Storage Exposure	/exports	SSN, Address, Email	7	113.4	100	Critical
Weak Password Policy	/admin	Admin Email	5	12	10.6	Moderate
Sensitive Logs Exposure	/logs	Email, Token	5.5	46.8	41.3	High
Admin IDOR	/admin/users	Employee SSN, Payroll	8	75.6	66.7	Critical
Internal Debug Endpoint	/debug	Internal Config	6	1	0.9	Low

5. Key Findings

The comparative evaluation highlights how incorporating PII sensitivity and contextual reasoning improves vulnerability prioritization over traditional CVSS-based approaches.

5.1 Same CVSS, Different Enterprise Risk

Vulnerabilities with similar CVSS scores can have significantly different business impact when data context is considered.
For example:
- **XSS on /search** has a CVSS score of 6.1 but exposes no sensitive data → Final Score **0.0**
- **Sensitive Logs Exposure on /logs** has a comparable CVSS score (5.5) but exposes emails and authentication tokens → Final Score **50.3**

This demonstrates that **technical severity alone is insufficient** for prioritization, as it does not capture the consequences of data exposure.

5.2 PII Exposure as a Primary Risk Driver

The proposed framework consistently assigns higher priority to vulnerabilities that expose sensitive or regulated data.
High-ranking findings include:
- /exports → bulk exposure of SSN, address, and email
- /admin/users → employee SSN and payroll data
- /user/profile → SSN and email

These scenarios represent **high-impact breach conditions**, including identity theft, financial fraud, and regulatory violations. By elevating such findings, the framework aligns prioritization with real-world enterprise risk.

5.3 Lower-Priority Handling of Non-Critical Findings

Vulnerabilities with limited business impact are appropriately deprioritized, even when their CVSS scores are moderate.

Examples include:
- Brute-force login attempts with limited data exposure
- Internal debug endpoints not externally accessible
- XSS vulnerabilities on non-sensitive pages

These findings still require remediation but can be scheduled with lower urgency compared to high-impact data exposure risks.

5.4 Reduction of Alert Fatigue

Traditional vulnerability management approaches often generate large volumes of medium-severity findings, making it difficult for security teams to focus on truly critical issues.
By incorporating PII sensitivity and contextual factors, the proposed framework:
- Reduces the relative priority of low-impact findings.
- Highlights vulnerabilities with meaningful data exposure
- Enables more focused remediation planning

This results in a reduction of **alert fatigue**, allowing security teams to allocate resources more effectively toward vulnerabilities that pose the greatest business risk.

5.5 PII-Aware Score Differentiation

A key outcome of the proposed framework is its ability to differentiate vulnerabilities based on **PII sensitivity and exposure context**, even when technical characteristics are similar.

For example:

- **SQL Injection on /user/profile** (SSN + Email) → **63.5 (Critical)**
- **XSS on /search** (no PII) → **0.0 (Low)**

Similarly:
- **IDOR on /admin/users** exposing payroll and SSN → **63.5 (Critical)**
- **Brute-force login on /login** exposing only email → **5.3 (Low)**

This demonstrates that the framework prioritizes vulnerabilities not only by exploitability, but by **what data is at risk and how it can be exploited**, resulting in more meaningful and actionable prioritization.

6. Key Contributions

The key contributions of this paper are:
- A PII-aware vulnerability prioritization model incorporating data sensitivity into risk scoring
- A hybrid architecture combining deterministic scoring with AI-driven contextual reasoning
- A normalized risk scoring function enabling consistent comparison across vulnerabilities
- A comparative evaluation demonstrating improved prioritization over CVSS-based approaches

7. Limitations

The proposed framework has several limitations that should be considered:
- **Dependence on API Schema Accuracy:** The effectiveness of the framework relies on the availability and correctness of API schema metadata. Incomplete or outdated schemas may lead to inaccurate identification of exposed data fields.
- **Requirement for Predefined PII Taxonomy:** The model depends on a predefined PII sensitivity taxonomy. Variations in organizational definitions or incomplete classification of data elements may impact scoring accuracy.
- **AI Reasoning Variability:** The quality of contextual reasoning and the resulting Contextual Adjustment Factor (ACF) are influenced by the capability and configuration of the underlying AI model.

8. Future Work

Several directions are identified for extending the proposed framework:
- **CI/CD Integration:** Incorporating the framework into continuous integration and deployment pipelines to enable automated, real-time vulnerability prioritization during development and release cycles.
- **Automated API Discovery:** Enhancing the system with automated API discovery mechanisms to dynamically identify endpoints and reduce dependence on predefined schema metadata.
- **Adaptive ACF Optimization:** Exploring reinforcement learning or feedback-driven approaches to dynamically tune the Contextual Adjustment Factor (ACF) based on historical outcomes and remediation effectiveness.

Source Code

The source code for the proposed framework is publicly available at https://github.com/umashankarak/pii-aware-pentest-demo under a general open-source license.

References

[1] FIRST.Org, *"Common Vulnerability Scoring System v3.1: Specification Document,"* 2019. Available: https://www.first.org/cvss/v3.1/specification-document
[2] FIRST.Org, *"CVSS v3.1 User Guide,"* 2019. Available: https://www.first.org/cvss/user-guide
[3] OWASP Foundation, *"OWASP Top 10: The Ten Most Critical Web Application Security Risks,"* 2021. Available: https://owasp.org/www-project-top-ten/
[4] Gartner, *"Innovation Insight for Risk-Based Vulnerability Management,"* Gartner Research, 2020.
[5] European Union, *"General Data Protection Regulation (GDPR),"* 2016.
[6] U.S. Department of Health & Human Services, *"Health Insurance Portability and Accountability Act (HIPAA),"* 1996.
[7] PCI Security Standards Council, *"Payment Card Industry Data Security Standard (PCI DSS),"* v4.0, 2022.
[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.
[9] A. Vaswani et al., *"Attention is All You Need,"* Proc. NeurIPS, 2017.
[10] Amazon Web Services, *"AWS Security Services Documentation,"* Available: https://docs.aws.amazon.com/security/
[11] Amazon Web Services, *"AWS Well-Architected Framework – Security Pillar,"* 2023.

Author Profile

Umashankara Kalaiah received Bachelor's and Master's degrees in Software Engineering. He is a Staff Software Engineer with over 22 years of experience in application architecture, cybersecurity, and application security (AppSec). He has designed and built large-scale enterprise applications across distributed and cloud-based environments, with a strong focus on secure system design and high-performance architectures. His work includes original contributions in cybersecurity, particularly in vulnerability management, secure API design, and AI-assisted security frameworks. He has been actively involved in advancing application security practices within enterprise organizations and has led multiple initiatives in integrating security into software development lifecycles (DevSecOps). He holds multiple professional certifications from the SANS Institute in cybersecurity and continues to contribute to research and innovation in AI-driven security, risk-based vulnerability prioritization, and modern cloud security architectures.