A Context-Aware Risk Assessment Agent for Vulnerability Prioritization: Towards a Dual-Agent Architecture

Ana Jokanović[1,2], Kosta Pavlović[2], Pavle Dakić[1,2,3,4,5,*], Savo Tomović[1,2], Aleksandar Plamenac[1,2], Terézia Hubáčková[3] and Ján Lang[1]

[1]*Institute of Informatics, Information Systems and Software Engineering, Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, Ilkovičova 2, 842 16 Bratislava, Slovakia*

[2]*Faculty of Natural Sciences and Mathematics, University of Montenegro, Cetinjski Put bb, 81 000 Podgorica, Montenegro*

[3]*Department of International Relations, Slovak University of Technology in Bratislava, Vazovova 5 81 243 Bratislava, Slovakia*

[4]*Faculty of Informatics and Computing, Singidunum University, Danijelova 32, 11 000 Belgrade, Serbia*

[5]*BabyFM, Veljka Dugoševića 54, 11 050, Belgrade, Serbia*

Abstract

Automated scanners flag software vulnerabilities by the thousand, but most are not exploited in practice. Severity scores such as the Common Vulnerability Scoring System (CVSS) still judge each finding purely on its technical profile—exploitability and organizational context do not factor into the score at all. We propose a dual-agent architecture separating these tasks: a Vulnerability Analysis Agent filters non-exploitable findings through controlled verification, and a Risk Assessment Agent scores confirmed vulnerabilities via an ISO/IEC 27005 likelihood-impact matrix (1–25 scale) and identifies vulnerability chains. This paper focuses on the design and empirical evaluation of the Risk Assessment Agent; the Vulnerability Analysis Agent is presented as a conceptual design only, with all vulnerabilities assumed exploitable to approximate its output, and is not itself empirically evaluated here. Across five large language models (LLMs) and five organizational profiles, the Risk Assessment Agent showed moderate inter-agent reliability (Intraclass Correlation Coefficient, ICC(A,1)=0.688) and strong reproducibility (73.9% exact agreement over three repeated runs). Organizational context shifted impact estimates beyond CVSS base scores, and chain detection reached strong consensus in two organizations (agreement in 14/15 and 13/15 cases).

Keywords

Vulnerability management, Exploitability analysis, Intelligent agents, Large language models, Risk modeling, Attack path analysis, ISO/IEC 27005, Cybersecurity

CEUR-WS.org/Vol-4336/paper10.pdf

1. Introduction

Software vulnerabilities are flaws that, once discovered, give an attacker a way to compromise a system, extract data, or interrupt the services it supports. Scanners such as Nessus and OpenVAS often report thousands of vulnerabilities in a typical corporate network, even though only a small fraction are ever exploited in practice [1], since segmentation, hardening, and access controls block exploitation regardless of patch status, leaving security teams sorting through findings in which most of the actual risk is concentrated in a small subset. The Common Vulnerability Scoring System (CVSS) does not help with this distinction, since it assigns severity ratings based on a vulnerability's technical properties alone, without regard to organizational context or deployment characteristics [1, 2]: a critical flaw in an isolated test environment does not carry the same risk as the same flaw in a production system handling sensitive financial or medical data, yet CVSS scores both identically [2]. This is especially

SQAMIA 2026: Workshop on Software Quality, Analysis, Monitoring, Improvement, and Applications, September 16–18, 2026, Niš, Serbia

*Corresponding author.

anajokanoviccc@gmail.com (A. Jokanović); kosta@ucg.ac.me (K. Pavlović); pavle.dakic@stuba.sk (P. Dakić); savot@ucg.ac.me (S. Tomović); plamenac.a@ucg.ac.me (A. Plamenac); terezia.hubackova@stuba.sk (T. Hubáčková); jan.lang@stuba.sk (J. Lang)

0009-0009-9377-5801 (A. Jokanović); 0000-0001-6838-6585 (K. Pavlović); 0000-0003-3538-6284 (P. Dakić); 0000-0002-9388-7784 (S. Tomović); 0000-0003-1925-9533 (A. Plamenac); 0009-0005-2584-0983 (T. Hubáčková); 0000-0002-3271-7271 (J. Lang)

© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

published 2026-11-19

pronounced in healthcare IoT, where interconnected devices add domain-specific risks [3]. Our dual-agent architecture addresses this gap: the Vulnerability Analysis Agent verifies whether a detected vulnerability is practically exploitable under real operational conditions, taking into account factors such as network segmentation, patch status, and service exposure, while the Risk Assessment Agent evaluates confirmed vulnerabilities using a likelihood–impact matrix aligned with ISO/IEC 27005 [4], producing scores on a 1–25 scale, and identifies vulnerability chains representing sequential exploit paths.

The main contributions of this paper are as follows:

- A dual-agent architecture in which exploitability verification and contextual risk assessment are treated as distinct analytical stages (Section 3, Proposed Architecture)
- A context-aware risk model aligned with ISO/IEC 27005 and evaluated across five large language model (LLM) instantiations (Section 3.2, Risk Assessment Agent)
- Preliminary evidence that vulnerability chains can be identified through cross-agent consensus using MITRE ATT&CK tactics (Section 5.4, Vulnerability Chain Analysis)

The paper continues with a review of related work on vulnerability scanners, context-aware prioritization, and attack-path analysis in Section 2. Section 3 introduces the proposed dual-agent architecture, while Section 4 describes the dataset and evaluation procedure. The results are presented and discussed in Section 5, together with their practical implications and limitations. The paper closes in Section 6.

2. Related Work

Nessus and OpenVAS are widely used tools for identifying known vulnerabilities across networked systems, but discovering a vulnerability does not indicate whether it is exploitable within an operational context [5], frequently leaving security teams with a large volume of findings to manually triage. Moscovich et al. [5] demonstrate that scanner output should be complemented by exploitability verification: their AutoSploit framework combines scanner findings with controlled exploitation attempts via Metasploit, showing that exploitability depends strongly on system-specific conditions such as active network services, available components, and deployed hardening mechanisms. Complementary automated-detection approaches, such as metaheuristic-optimization-based IoT/IIoT intrusion detection [6], similarly surface exploitable weaknesses beyond static scanning.

CVSS is the dominant standard for vulnerability severity assessment; however, its limitations in organizational prioritization are well documented. Allodi and Massacci [1] find only a weak correlation between CVSS base scores and actual exploitation, and report that severity alone tends to overstate real-world risk. Eiram and Martin [7] identify fundamental CVSSv2 weaknesses, including limited granularity and a failure to support context-dependent assessment. More recent approaches focus on context-aware extensions and automation: Ahmadi Mehri et al. [8] introduce ACVRM, a framework that enables prioritization based on organizational risk appetite, while the CAVSS model extends CVSS using temporal and confidence-based metrics derived through expert-defined heuristics [9]. These approaches share the assumption that severity is only meaningful within a specific operational context.

Beyond individual vulnerability scoring, ISO/IEC 27005:2022 [4] defines a structured framework for information security risk management based on identifying assets, threats, and vulnerabilities and evaluating risk through likelihood and impact, supporting qualitative risk matrices such as a 5×5 likelihood-impact model producing scores in the range 1–25. A further line of research addresses the interdependencies between vulnerabilities: Sadlek et al. [10] formalize multi-step attack behavior using attack graphs aligned with the MITRE ATT&CK framework, while Chen et al. [11] analyze vulnerability correlation and exploit chaining in breach-and-attack simulation scenarios, showing that interdependencies between vulnerabilities are a key factor in determining overall system risk. Runtime detection of such sequences relates to reusable complex event pattern monitoring [12].

Despite extensive research in vulnerability assessment, risk scoring, and attack-path modeling, existing approaches remain fragmented: exploitability verification is typically performed in isolation,

context-aware prioritization does not incorporate vulnerability interdependencies, and attack-path models are rarely integrated with formal risk frameworks such as ISO/IEC 27005. To the best of our knowledge, no existing approach combines exploitability-verified input with ISO/IEC 27005 scoring and MITRE ATT&CK chain detection within a single organizational risk framework — the gap this paper addresses.

3. Proposed Architecture

The proposed system brings together two agents that operate in sequence: a **Vulnerability Analysis Agent** and a **Risk Assessment Agent**. The underlying design rationale separates vulnerability detection from risk evaluation, treating them as analytically distinct tasks that call for different capabilities. Only vulnerabilities confirmed as exploitable under real operational conditions enter the risk assessment stage, making the first agent's output a direct prerequisite for the second.

As illustrated in Figure 1, scanner output (A) containing CVE (Common Vulnerabilities and Exposures) identifiers and CVSS scores is passed to the Vulnerability Analysis Agent (B), which performs exploitability verification through controlled attempts. Only confirmed exploitable vulnerabilities are forwarded to the Risk Assessment Agent (D) as a **dynamic vulnerability context** — continuously updated as new vulnerabilities are discovered or existing ones are remediated. The agent additionally receives a **static organizational context** (C) covering industry sectors, compliance requirements, and technology stack, which remains fixed across assessment cycles. The agent produces individual risk scores (E) on a 1–25 scale and identifies vulnerability chains (F) representing sequential attack paths.

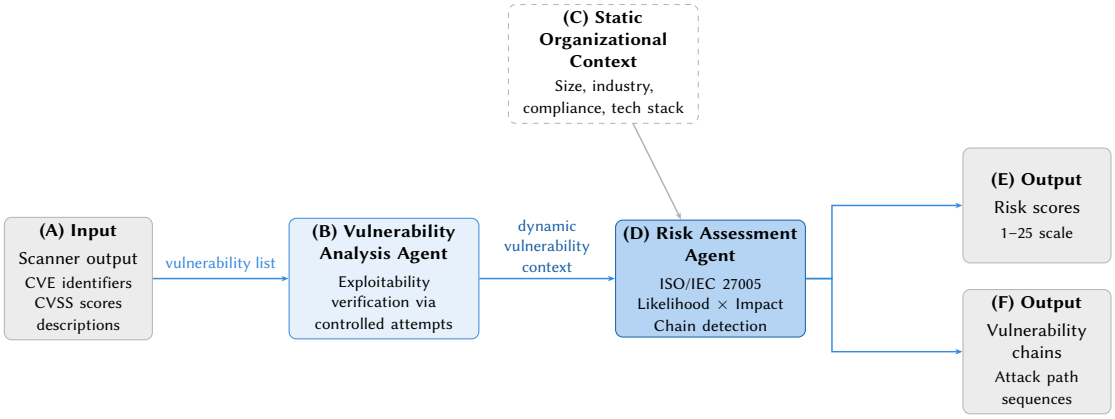


Figure 1: Proposed dual-agent system architecture. Labels: (A) scanner output, (B) Vulnerability Analysis Agent, (C) static organizational context, (D) Risk Assessment Agent, (E) individual risk scores, (F) vulnerability chains.

3.1. Vulnerability Analysis Agent

The Vulnerability Analysis Agent receives as input the raw findings produced by security scanning tools such as Nessus or OpenVAS. These findings consist of structured records containing CVE identifiers, affected service names and versions, and technical vulnerability descriptions. The scanner, however, reports all detected vulnerabilities regardless of whether they are practically exploitable in the target environment.

To address this limitation, the Vulnerability Analysis Agent is conceptually designed to verify whether a reported vulnerability is exploitable in the target environment, rather than relying solely on static severity scores or theoretical assessments. Where a directly executable exploit is available, this verification could involve controlled exploitation attempts in an isolated testbed, in the manner demonstrated by exploitation-verification frameworks such as AutoSploit, which combines scanner findings with controlled exploitation attempts via Metasploit [5]. However, such exploit-based verification cannot be universally applied to newly disclosed vulnerabilities, making full automation of this step an open

technical challenge. When a directly executable exploit is unavailable, future implementations may incorporate complementary evidence-based verification mechanisms to assess exploitability from available technical evidence (Section 6). Vulnerabilities that cannot be confirmed as exploitable would be discarded, reducing the number of findings security teams need to review before prioritization. The present study does not implement or evaluate this agent; instead, its output is approximated by assuming the reported vulnerabilities to be exploitable, allowing the experimental evaluation to focus on the Risk Assessment Agent (Section 4). In brief, given scanner output $S = \{v_1, \ldots, v_n\}$, the agent would return the confirmed set $E \subseteq S$ of exploitable vulnerabilities as input to the Risk Assessment Agent; a full pseudocode listing is omitted since the mechanism is conceptual.

3.2. Risk Assessment Agent

The Risk Assessment Agent takes two inputs. The first is a dynamic vulnerability context — the set of exploitable vulnerabilities confirmed by the Vulnerability Analysis Agent, which changes as vulnerabilities are discovered or remediated. The second is a static organizational context that describes the organization's size, workforce, infrastructure, data categories, and regulatory obligations; this remains unchanged across assessment cycles unless the organization undergoes major structural changes. Each confirmed vulnerability is then scored using a 5×5 likelihood–impact matrix in line with ISO/IEC 27005:2022 [4], with risk calculated as:

$$R = L \times I \tag{1}$$

where R denotes the resulting risk value in the range $[1, 25]$, $L \in \{1, 2, 3, 4, 5\}$ represents the likelihood of exploitation given the organizational context, and $I \in \{1, 2, 3, 4, 5\}$ represents the severity of potential impact. The resulting score is mapped to one of five risk categories, as illustrated in Table 1.

Table 1
Likelihood–impact risk matrix aligned with ISO/IEC 27005, with recommended response actions per risk band.

		Impact				
		1	**2**	**3**	**4**	**5**
Likelihood	**5**	5	10	15	20	25
	4	4	8	12	16	20
	3	3	6	9	12	15
	2	2	4	6	8	10
	1	1	2	3	4	5

Informational (1–2)	Monitor; no immediate action required
Low (3–6)	Schedule remediation in routine maintenance
Medium (7–10)	Remediate within 30 days
High (11–15)	Remediate within 7 days
Critical (16–25)	Patch immediately, within 24 hours

The 5×5 matrix was selected because it provides sufficient granularity to separate risk levels clearly while remaining easy to apply. A smaller 3×3 matrix would group too many different risks into the same category, while a larger 10×10 matrix would add unnecessary complexity. This choice is consistent with ISO/IEC 27005 recommendations [4]. Unlike CVSS-based scoring, this component does not treat vulnerabilities in isolation — the same vulnerability may receive different scores depending on organizational context.

Beyond the assessment of individual vulnerabilities, the risk assessment agent considers whether the confirmed exploitable vulnerabilities can be combined into vulnerability chains — sequences in which the successful exploitation of one vulnerability creates the conditions necessary to exploit

another. Such chains model realistic multi-step attack scenarios rather than treating each finding independently [10, 11].

The agent identifies candidate chains by analyzing logical dependencies between vulnerabilities: whether exploitation of one step materially enables the next, using ATT&CK tactic progression as a structural heuristic. This aligns with established frameworks such as MITRE ATT&CK [13] and the Cyber Kill Chain [14].

The aggregate risk of an identified chain c is computed as:

$$r_c = \min\left(25,\ \max_{v_i \in c}(r_i) + \text{round}\left(0.3 \times \sum_{v_j \in c \setminus \arg\max} r_j\right)\right) \tag{2}$$

where r_i denotes the individual risk score of vulnerability v_i. The factor 0.3 was selected empirically to ensure secondary vulnerabilities contribute without overshadowing the dominant one; scores are capped at 25. Algorithm 1 describes the agent logic.

Algorithm 1 Risk Assessment Agent

Require: Confirmed exploitable vulnerability set E, organizational context profile O
Ensure: Risk scores R, vulnerability chains CH

$R \leftarrow \emptyset, CH \leftarrow \emptyset$
▷ Step 1: Compute individual risk scores
for each vulnerability $v_i \in E$ **do**
    Assess likelihood $l_i \in \{1, \ldots, 5\}$ based on context O
    Assess impact $m_i \in \{1, \ldots, 5\}$ based on context O
    $r_i \leftarrow l_i \times m_i$
    Assign risk category based on r_i
    $R \leftarrow R \cup \{(v_i, r_i)\}$
end for
▷ Step 2: Map vulnerabilities to MITRE ATT&CK tactics
for each vulnerability $v_i \in E$ **do**
    Map v_i to primary tactic t_i from subset $T = \{\text{TA0001–TA0010}\}$ [13]
end for
▷ Step 3: Identify vulnerability chains
Valid chain: $t_a \rightarrow t_b$ where t_a precedes t_b in T
for each subsequence of E ordered by T **do**
    if subsequence forms valid tactic progression in T **then**
        $CH \leftarrow CH \cup \{\text{subsequence}\}$
    end if
end for
▷ Step 4: Compute aggregate chain risk
for each chain $c \in CH$ **do**
    $r_c \leftarrow \min\left(25,\ \max_{v_i \in c}(r_i) + \text{round}\left(0.3 \times \sum_{v_j \in c \setminus \arg\max} r_j\right)\right)$
    $R \leftarrow R \cup \{(c, r_c)\}$
end for
return R, CH

4. Methodology

This section describes the experimental setup used to evaluate the Risk Assessment Agent from the proposed dual-agent architecture. Because exploitability verification requires a controlled exploitation environment and is outside the scope of this study, the Vulnerability Analysis Agent is modeled abstractly, and all vulnerabilities are assumed exploitable to approximate its confirmed output. The evaluation focuses on assessing the reliability, context sensitivity, and chain detection capabilities of the Risk Assessment Agent across five independent LLM instantiations.

4.1. Research Questions

The evaluation reported in this paper was guided by the following research questions:

RQ1: To what extent do multiple independent LLM-based agents achieve consistent risk scores across repeated evaluations?

RQ2: How does organizational context influence the distribution of LLM-derived risk assessments for technically identical vulnerabilities?

RQ3: Can LLM agents identify plausible vulnerability chains with consistent cross-agent agreement across diverse organizational environments?

RQ4: How does risk assessment behavior vary across different LLM models?

4.2. Experimental Setup

The Risk Assessment Agent was evaluated using five widely used LLM models as independent instantiations: GPT-4o (OpenAI), Gemini 2.5 Flash (Google), Claude Sonnet 4.6 (Anthropic), Llama 3.3 70B via Groq, and Command R+ via Cohere. The evaluation followed a structured five-stage methodology executed under a hybrid model: orchestration scripts were executed locally, coordinating API calls to five cloud-hosted LLM services, with all outputs stored in a structured JSON repository, as illustrated in Figure 2.

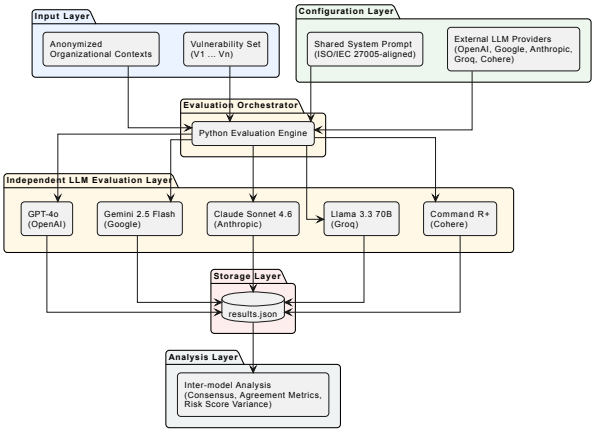


Figure 2: Evaluation architecture for the Risk Assessment Agent.

Stage 1: Dataset Construction. Five synthetic organizational profiles comprising 23 vulnerabilities in total (4–5 per profile) were constructed with CVE identifiers and CVSS base scores drawn from publicly available National Vulnerability Database (NVD) records [15], summarized in Table 2.

Table 2
Evaluation dataset summary.

ID	Sector	Vulnerabilities	CVSS Range
FSP	Financial Services	5	7.5–9.8
HE	Healthcare	5	5.3–10.0
ECR	E-Commerce	4	6.1–9.8
ESO	Energy/ICS (Industrial Control Systems)	4	6.8–10.0
ETP	Educational Technology	5	5.3–8.2

Stage 2: Agent Configuration. A shared system prompt was developed defining the ISO/IEC 27005-aligned risk assessment task, MITRE ATT&CK tactic mapping, and structured JSON output schema. Five LLM models were configured as independent instantiations of the Risk Assessment Agent, accessed via their respective cloud APIs without fine-tuning or additional training, and queried with a temperature of 0.2 to reduce output variability while preserving structured reasoning capacity; no additional hyperparameters were modified from provider defaults. Tactic mapping was performed by the LLM agent itself based on vulnerability descriptions and organizational context, guided by explicit ATT&CK

tactic definitions in the system prompt. To support reproducibility, the system prompt used to instruct the five LLMs (defining the risk assessment task, MITRE ATT&CK tactic mapping, and structured JSON output schema) is available upon request from the corresponding author. The user message for each query consisted of the corresponding organizational profile and vulnerability list, passed as structured text without additional templating.

Stage 3: Independent Evaluation. Each model independently assessed all vulnerabilities across all five organizations, producing individual risk scores (Likelihood $\times$ Impact, 1–25) and identifying vulnerability chains.

Stage 4: Reproducibility Assessment. Each evaluation was repeated three times using identical inputs. Final risk scores were computed as the median across three independent runs.

Stage 5: Inter-Agent Analysis. Results were analyzed using multiple complementary methods: inter-agent reliability was assessed via ICC(A,1), risk score distributions were examined through majority consensus analysis and severity-stratified agreement rates, agent behavioral patterns were identified through hierarchical clustering, and vulnerability chain detection was evaluated through cross-agent consensus analysis.

5. Results and Discussion

This section presents the evaluation results addressing RQ1–RQ4 in turn: inter-agent reliability and score consensus (Section 5.1, RQ1), context sensitivity (Section 5.2, RQ2), cross-model behavioral patterns (Section 5.3, RQ4), and vulnerability chain consensus (Section 5.4, RQ3), before closing with practical implications and limitations (Section 5.5).

5.1. Inter-Agent Reliability and Risk Score Analysis

To evaluate the consistency of risk assessments across the five LLM-based agents, the Intraclass Correlation Coefficient ICC (A,1) was computed across three independent evaluation runs using a two-way random-effects model with absolute agreement and single measures [16]. Across the three runs, ICC(A,1) came in at 0.700, 0.667, and 0.697, averaging 0.688 — moderate agreement by benchmarks Koo and Li proposed [16]. The agents therefore rank vulnerability risk in a broadly consistent way, although their underlying scoring calibration is not identical across models. Score-level reproducibility was further assessed by comparing each agent's output across the three runs: out of 115 agent–vulnerability pairs, 85 (73.9%) received identical scores in every run, and an additional 24 pairs stayed within a $\pm$5-point margin, raising the share of stable assessments to 94.8% (109/115). The remaining 6 pairs (5.2%) exceeded this margin, and in every case the source was Gemini, the model with the highest run-to-run variability among those tested. While 0.688 is below the conventional 0.75 threshold for strong reliability, this level of agreement can still be considered meaningful, given that context-dependent risk assessment is inherently subjective and lacks a fixed ground truth. This moderate ICC is not evenly distributed across models: all 6 outlier pairs came from Gemini, while Claude reached 91.3% exact agreement (Section 5.3, Agent Behavioral Patterns), so ICC(A,1)=0.688 is driven substantially by one unstable model rather than a uniform limitation of LLM-based scoring. Since temperature and prompt were held constant across agents, whether this instability stems from the model itself, temperature sensitivity, or prompt phrasing cannot be disentangled here and is left to the planned prompt/temperature-sensitivity study (Section 6). These results suggest that LLM-based assessment may serve as a potential complement to manual vulnerability prioritization (RQ1).

Figure 3 presents the median risk scores assigned by all five agents across the 23 evaluated vulnerabilities, color-coded according to the ISO/IEC 27005 risk categories defined in Table 1.

Majority consensus analysis — defined as at least three of five agents assigning identical median scores — revealed agreement on 12 of 23 vulnerabilities (52.2%). Four vulnerabilities reached perfect consensus across all five agents — HE-V1 (20, Critical), HE-V4 (12, High), ECR-V2 (25, Critical), and ETP-V3 (9, Medium). Grouping scores by risk category, rather than treating them as exact values, brings a clearer pattern into view. At the precise-score level, only 52.2% of vulnerabilities reached majority

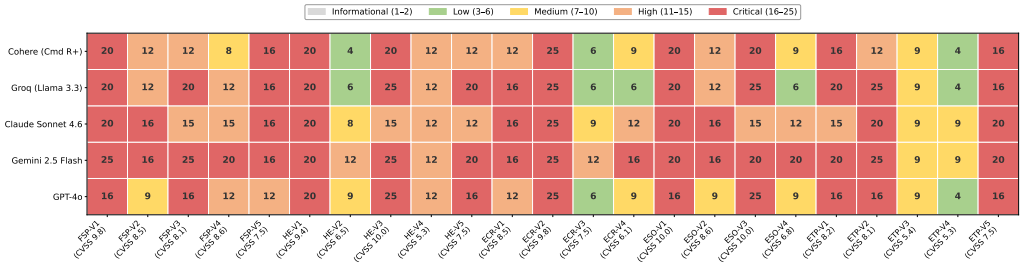


Figure 3: Risk score heatmap showing median agent scores across all vulnerabilities and organizations. Color coding follows ISO/IEC 27005 risk categories. Columns where all cells share the same color indicate full agent consensus on risk classification.

consensus; grouped into categories, that figure climbs to 78.3% (18/23). Agents, in other words, tend to disagree on the exact number but converge on roughly how severe a vulnerability is. The heatmap also points to where this breaks down — divergence is largest for vulnerabilities whose scores straddle category boundaries, as with ESO-V3, where agents' scores ranged from 15 (High) to 25 (Critical). Cases like this likely correspond to genuinely ambiguous risk scenarios that would benefit from review by a human analyst.

5.2. Context-Aware Risk Assessment

To investigate RQ2, a controlled comparison was conducted using SQL Injection (V5, CWE (Common Weakness Enumeration) identifier 89, CVSS 7.5) present across three organizationally distinct profiles: FSP (Financial Services), HE (Healthcare), and ETP (Educational Technology). Exploitability likelihood was largely stable across profiles — all five agents assigned $L = 4$ for FSP and ETP, while three of five agents assigned $L = 4$ for HE, with Claude and Command R+ consistently assigning $L = 3$ across all three runs. Variation was observed exclusively in impact estimation, as summarized in Table 3.

Table 3
Impact score distribution for SQL Injection (CWE-89, CVSS 7.5) across three organizational profiles.

Org.	Regulatory Profile	L	Impact Range	Majority Score
FSP	PCI-DSS, GDPR	4	3–4	16 (4/5)
HE	HIPAA, HITECH	3–4	3–5	no majority
ETP	FERPA, COPPA	4	4–5	16 (3/5)

FSP received a strong majority consensus of 16 (4/5 agents, $L = 4$, $I = 4$), reflecting moderate-high financial data risk under PCI-DSS. ETP showed a similar majority score but with two agents assigning $I = 5$, driven by the sensitivity of minors' academic data under FERPA and COPPA. The healthcare profile exhibited the highest divergence—with scores of 12, 16, and 20—suggesting that the complexity of Protected Health Information (PHI) exposure under the Health Insurance Portability and Accountability Act (HIPAA) introduces variability in mapping technical characteristics to impact estimates.

These findings suggest that organizational context may systematically influence impact estimation beyond CVSS base scores—likelihood remained stable across profiles, while impact diverged significantly. Highly regulated domains with overlapping compliance requirements present the greatest challenge for automated impact estimation and benefit most from human analyst review. This variation is consistent with established CVSS limitations [1] and supports context-aware frameworks aligned with ISO/IEC 27005 [4] (RQ2).

5.3. Agent Behavioral Patterns

To address RQ4, pairwise Pearson correlations were computed between agent score vectors across all 23 vulnerabilities, and hierarchical clustering was applied using average linkage with distance $d = 1 - r$,

where r denotes the Pearson correlation coefficient. The resulting analysis is presented in Figure 4.

The analysis reveals two behavioral clusters. The first comprises Llama and GPT-4o, which exhibit the strongest pairwise correlation ($r = 0.910$), indicating highly similar scoring patterns. Command R+ joins this cluster at a slightly greater distance ($r = 0.837$–0.865). The second cluster comprises Gemini and Claude, which show comparatively lower correlations with the first group ($r = 0.654$–0.757). Majority alignment analysis further supports this structure. Llama achieved the highest alignment with the ensemble majority (52.2%, 12/23 vulnerabilities), followed by Command R+ (43.5%), while GPT-4o, Gemini, and Claude each achieved 34.8%.

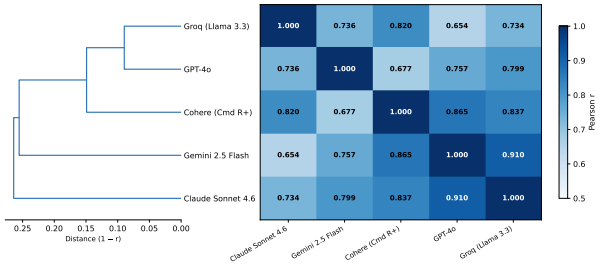


Figure 4: LLM agent behavioral similarity analysis. Left: hierarchical clustering dendrogram based on pairwise score correlation. Right: Pearson correlation matrix reordered according to clustering structure. Darker cells indicate stronger behavioral similarity.

These differences suggest that model selection has a meaningful impact on both scoring consistency and chain detection behavior, and should be considered when designing ensemble-based vulnerability assessment pipelines. In terms of chain detection, Claude demonstrated the most consistent behavior across all five organizations, while GPT-4o showed the most conservative detection pattern. The relatively uniform majority alignment rates (34.8%–52.2%) indicate that no single model dominates consensus formation, supporting the use of ensemble-based assessment over reliance on any individual model. Reproducibility also varied by model. Claude held up best across runs, with 91.3% exact agreement, while Gemini was the most erratic — its scores stayed consistent only 52.2% of the time, and every one of the six cases that exceeded the ±5-point threshold came from it (RQ4).

5.4. Vulnerability Chain Analysis

Chain detection was evaluated through cross-agent consensus analysis across three independent runs. Each chain vote corresponds to a single (agent, run) pair prior to median aggregation, yielding a maximum of 15 votes (5 agents × 3 runs). Aggregate chain risk scores were computed by applying Equation 2 to the median individual vulnerability scores across agents and runs. Table 4 summarizes the most consistently identified vulnerability chains for each organization. Additional chains were detected by individual agents; however, they are not reported here, as the analysis focuses on chains exhibiting the highest level of cross-agent consensus.

Table 4
Vulnerability chain consensus and aggregate risk scores across five organizations.

Org.	Chain	Type	Attack Path	Consensus	r_c
ETP	V1→V5	Info disclosure	File access → DB enumeration	14/15	21
HE	V1→V3	Speculative	SQLi → Log4Shell (indirect)	13/15	25
ECR	V1→V2	Direct	Auth bypass → credential extraction	7/15	25
ESO	V2→V3	Network pivot	MQTT entry → IoT RCE	7/15	24
FSP	V3→V5	Privilege escalation	JWT compromise → SQL Injection	6/15	21

Among all five organizations, ETP showed the highest agreement — 14 out of 15 agent-run combinations detected a two-step attack path between V1 (Path Traversal, CVSS 8.2) and V5 (SQL Injection,

CVSS 7.5). The only exception was Command R+ in the first evaluation run. This chain, moving from information disclosure to database enumeration, produced an aggregate risk of $r_c = 21$ (Critical), exceeding either vulnerability's individual score ($r = 16$ each) given the compounded exploitation potential in a regulated educational environment handling minors' data under FERPA and COPPA.

The Healthcare profile came in second for consensus (13/15), linking V1 (SQL Injection, CVSS 9.4) and V3 (Log4Shell, CVSS 10.0) for $r_c = 25$ (Critical), despite the two vulnerabilities operating on different attack surfaces without a direct technical dependency — a speculative chain discussed further in Section 5.5.

ECR and ESO achieved moderate consensus (7/15 each), identifying technically stronger chains: ECR V1→V2 represents a direct authorization bypass enabling credential extraction ($r_c = 25$), while ESO V2→V3 models a realistic ICS network pivot from Message Queuing Telemetry Transport (MQTT) broker compromise — where unauthenticated device registration enables access to an internal network segment — to IoT gateway Remote Code Execution (RCE) ($r_c = 24$). FSP showed the lowest consensus (6/15), with a privilege escalation chain from JWT (JSON Web Token) key compromise to SQL Injection ($r_c = 21$). The low consensus likely reflects agent preference for the alternative V1→V4 chain (RCE → container escape), which represents a more direct attack path to privilege escalation than JWT compromise enabling SQL Injection.

The results suggest that chain identification agreement varies with the clarity of technical dependencies between vulnerabilities. Lower consensus in three of five organizations indicates that chain formation is sensitive to structural exploit dependencies rather than vulnerability co-occurrence alone. Figure 5 presents per-agent detection consistency across evaluation runs.

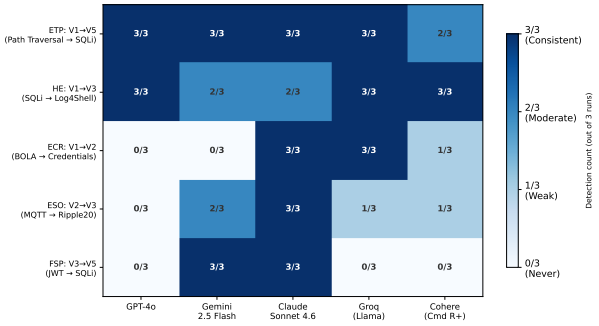


Figure 5: Agent–chain detection agreement across three independent evaluation runs. Each cell shows how many times (out of 3 runs) a given agent included the corresponding chain in its output. Darker cells indicate more consistent chain detection.

LLM agents demonstrated varying levels of chain detection agreement across the five organizations, with stronger consensus observed when clear technical dependencies existed between vulnerabilities. High consensus alone does not guarantee technical validity—as illustrated by the speculative HE case where SQL Injection and Log4Shell operate on different attack surfaces—and requires expert verification (RQ3).

5.5. Practical Implications and Limitations

The results of this study have several practical implications for security teams. The moderate inter-agent consistency, combined with high score-level reproducibility, suggests that ensemble-based LLM assessment may serve as a complementary signal for manual vulnerability prioritization, particularly in resource-constrained environments. The context sensitivity demonstrated in the SQL injection experiment highlights the importance of incorporating organizational profiles into risk scoring, rather than relying solely on CVSS severity ratings. Vulnerability chain detection can reveal multi-step attack vectors overlooked by individual scores, supporting more detailed threat modeling. The speculative Healthcare chain (V1 SQL Injection → V3 Log4Shell, Section 5.4) is an instructive cautionary result: despite operating on unrelated attack surfaces with no direct technical dependency, it was flagged with

13/15 consensus, nearly matching the technically well-founded ETP chain (14/15). High cross-agent agreement can therefore arise from the co-occurrence of two critical vulnerabilities in a sensitive environment (here, PHI-handling infrastructure under HIPAA) rather than a verified exploitation dependency, so consensus strength alone is not a reliable proxy for technical validity. LLM-flagged chains should be treated as prioritized hypotheses for expert or technical verification, not confirmed attack paths.

Several limitations of this study should be acknowledged. The evaluation dataset consists of five synthetically constructed organizational profiles informed by real-world industry characteristics, with vulnerabilities drawn from publicly available NVD records; while this reflects realistic deployment scenarios, it may not fully capture the complexity of actual enterprise environments. The vulnerability analysis agent was not implemented in this study—all vulnerabilities were assumed exploitable, which may overestimate risk in environments with strong defensive controls. The evaluation also relies exclusively on LLM-based assessment without expert ground truth validation, and was therefore designed to measure inter-agent consistency rather than correctness; future work will incorporate expert-validated ground truth to assess model accuracy. Nor does the study include a comparative evaluation against baseline approaches such as CVSS-only scoring or single-agent assessment; such a CVSS-only baseline would make the case for the proposed architecture considerably stronger and is identified as the most immediate direction for future work (Section 6) — the relative advantage of the proposed architecture over simpler alternatives therefore remains to be quantified.

6. Conclusion

This paper presented a dual-agent architecture for context-aware vulnerability risk assessment, in which the Vulnerability Analysis Agent identifies exploitable findings and the Risk Assessment Agent scores them using an ISO/IEC 27005 likelihood-impact matrix and MITRE ATT&CK-based chain detection. Across five LLM instantiations, inter-agent consistency was moderate but meaningful (ICC(A,1)=0.688), suggesting ensemble-based LLM assessment can complement manual prioritization in resource-constrained environments (RQ1). Organizational context systematically influenced impact estimates beyond CVSS base scores, with the greatest divergence in highly regulated sectors with overlapping compliance requirements (RQ2). LLM agents identified plausible vulnerability chains with strong consensus when clear technical dependencies existed, though high agreement does not guarantee technical validity (RQ3). Model selection meaningfully affected both scoring and chain detection behavior, with two distinct behavioral clusters emerging across the five models (RQ4).

Future work will proceed along four dimensions: implementing the Vulnerability Analysis Agent on a controlled testbed (e.g., Metasploitable) for end-to-end pipeline evaluation, including complementary evidence-based verification mechanisms to extend coverage beyond vulnerabilities for which a directly executable exploit is available; validating the framework on real organizational data with expert-validated ground truth and analyst review; evaluating prompt and temperature sensitivity; and conducting a comparative evaluation against a CVSS-only baseline to quantify the added value of the proposed architecture.

Acknowledgments

The authors acknowledge that this work was supported by Erasmus+ ICM 2023 No. 2023-1-SK01-KA171-HED-000148295, the Cultural and Educational Grant Agency of the Ministry of Education, Research, Development and Youth of the Slovak Republic (KEGA) - Model-based explication support for personalized education (Podpora personalizovaného vzdelávania explikovaná modelom) - KEGA (014STU-4/2024), the Slovenian Research and Innovation Agency (ARIS) (Research Core Funding No. P2-0057) and the Slovak Research and Development Agency under Contract no. APVV-23-0408.

Declaration on Generative AI

The authors did not use any generative AI tools to generate content, text, or ideas in this paper. Grammarly was used solely for grammar and spell-checking of the manuscript text.

References

[1] L. Allodi, F. Massacci, Comparing vulnerability severity and exploits using case-control studies, ACM Trans. Inf. Syst. Secur. 17 (2014) 1:1–1:20. doi:10.1145/2630069.

[2] C. Frühwirth, T. Männistö, Improving CVSS-based vulnerability prioritization and response with context information, in: Proc. 3rd International Symposium on Empirical Software Engineering and Measurement (ESEM), IEEE, 2009, pp. 535–544. doi:10.1109/ESEM.2009.5314230.

[3] T. E. Ali, F. I. Ali, P. Dakić, A. D. Zoltan, Trends, prospects, challenges, and security in the healthcare internet of things, Computing 107 (2025) 28. doi:10.1007/s00607-024-01352-4.

[4] ISO27005, ISO/IEC 27005:2022 – information security, cybersecurity and privacy protection – guidance on managing information security risks, 2022. URL: https://www.iso.org/standard/80585.html.

[5] N. Moscovich, R. Bitton, Y. Mallah, M. Inokuchi, T. Yagyu, M. Kalech, Y. Elovici, A. Shabtai, AutoSploit: A fully automated framework for evaluating the exploitability of security vulnerabilities, arXiv preprint arXiv:2007.00059 (2020). URL: https://arxiv.org/abs/2007.00059. arXiv:2007.00059.

[6] P. Dakic, M. Zivkovic, L. Jovanovic, N. Bacanin, M. Antonijevic, J. Kaljevic, V. Simic, Intrusion detection using metaheuristic optimization within iot/iiot systems and software of autonomous vehicles, Scientific Reports 14 (2024) 22884. doi:10.1038/s41598-024-73932-5.

[7] C. Eiram, B. Martin, The CVSSv2 shortcomings, faults, and failures formulation, Open letter to FIRST, Risk Based Security and Open Security Foundation, 2013. URL: https://www.semanticscholar.org/paper/The-CVSSv-2-Shortcomings-,-Faults-,-and-Failures-:-Eiram/814ccfb45f856f48bb50048d16b3dc65b7acaf03.

[8] V. Ahmadi Mehri, P. Arlos, E. Casalicchio, Automated context-aware vulnerability risk management for patch prioritization, Electronics 11 (2022) 3580. doi:10.3390/electronics11213580.

[9] B. Jung, Y. Li, T. Bechor, CAVP: A context-aware vulnerability prioritization model, Computers & Security 116 (2022) 102639. doi:10.1016/j.cose.2022.102639.

[10] L. Sadlek, P. Čeleda, D. Tovarňák, Identification of attack paths using kill chain and attack graphs, in: IEEE/IFIP Network Operations and Management Symposium (NOMS), 2022, pp. 1–6. doi:10.1109/NOMS54207.2022.9789803.

[11] J. Chen, R. Liang, M. Zhang, C. Zheng, X. Huang, H. Lu, X. Yu, Z. Tian, Vulnerability correlation, multi-step attack and exploit chain in breach and attack simulation, in: IEEE 12th International Conference on Cloud Networking (CloudNet), IEEE, 2023, pp. 398–402. doi:10.1109/CloudNet59005.2023.10490046.

[12] J. Lang, M. Jantosovic, I. Polasek, Re-usability in complex event pattern monitoring, in: 2012 IEEE 10th International Symposium on Applied Machine Intelligence and Informatics (SAMI), IEEE, 2012, pp. 265–270. doi:10.1109/SAMI.2012.6208970.

[13] MITRE Corporation, MITRE ATT&CK framework, 2024. URL: https://attack.mitre.org.

[14] Lockheed Martin, Cyber kill chain, 2011. URL: https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html.

[15] National Institute of Standards and Technology, National Vulnerability Database (NVD), 2024. URL: https://nvd.nist.gov.

[16] T. K. Koo, M. Y. Li, A guideline of selecting and reporting intraclass correlation coefficients for reliability research, Journal of Chiropractic Medicine 15 (2016) 155–163. doi:10.1016/j.jcm.2016.02.012.