Master's Thesis: Evaluation of Proposed Models by Domain Experts

Felix Berger

March 7, 2025

Introduction and Purpose

This questionnaire is part of my master's thesis, in which I have developed four models for classifying and remediating software vulnerabilities. The following pages provide a concise overview of these models and invite feedback from domain experts. Your insights will directly influence how these models are refined, ensuring they are both theoretically sound and well-suited for real-world implementation. You can fill out the questionnaire **digitally** (if supported by your PDF reader) or print it and complete it manually.

The four models described here are:

- A multi-dimensional classification approach of vulnerabilities that integrates:
  - The **Common Vulnerability Scoring System (CVSS) v. 3.1**, which rates technical severity on a 0.0–10.0 scale.
  - The **Exploit Prediction Scoring System (EPSS)**, which estimates the probability (0.0–1.0) of a vulnerability being exploited in the wild.
- An algorithm to compute a classification for a known vulnerability in the context of a given software (a single numeric score, the *Combined Severity*).
- A remediation model offering different recommendations for roles (e.g., developers, security advisors).
- A rank-ordering mechanism that highlights critical or data-incomplete vulnerabilities first.

What CVSS Considers

CVSS provides a structured way to measure the severity of software vulnerabilities. Primary metrics include:

- **Attack Vector**: Network, adjacent, local, or physical.
- **Attack Complexity**: The conditions beyond the attacker's control that must exist to exploit a vulnerability.
- **Privileges Required**: The level of privileges an attacker needs.

- **User Interaction**: Whether user action is required for successful exploitation.
- **Scope**: Whether a vulnerability in one component impacts resources in another component.
- **Confidentiality, Integrity, Availability Impacts**: The potential effects on data or system.

What EPSS Considers

EPSS is an empirical model predicting how likely a vulnerability is to be exploited in practice. It draws upon:

- **Vendor Information**: Derived from CPE via NVD.
- **Age of the Vulnerability**: Days since the CVE was published by MITRE.
- **References**: With categorical labels, e.g., MITRE CVE List, NVD.
- **Normalized Multiword Expressions**: Extracted from vulnerability descriptions (MITRE CVE List).
- **Weakness Details**: CWE identifiers from NVD.
- **CVSS Metrics**: Base vectors from CVSS 3.x (via NVD).
- **Listings on Well-Known Sites**: CISA KEV, Google Project Zero, Trend Micro's ZDI.
- **Publicly Available Exploit Code**: Exploit-DB, GitHub, Metasploit.
- **Offensive Security Tools/Scanners**: Intrigue, sn1per, jaeles, nuclei.

1) Multi-Dimensional Classification Model

Objective. This model seeks to determine the most suitable approach for assigning a single severity score to software vulnerabilities. Instead of relying exclusively on a predefined metric (e.g., CVSS v. 3.1) or a single data source (e.g., EPSS), the model aims to integrate and balance multiple factors to produce a more nuanced assessment of vulnerability severity. By systematically refining these inputs and their relative weights, the model aspires to offer a flexible and context-aware classification method that can outperform any single, static metric.

Formula Example.

$$\text{Severity Score} = \text{round}\left(\frac{w_{\text{cvss}} \times \text{CVSS} + w_{\text{epss}} \times \left(10.0 \times \text{EPSS}\right)}{2.0}\right),$$

where each w represents the relative importance assigned to its respective factor, and round is a typical rounding function. Multiplying EPSS by 10 aligns it with CVSS's 0.0–10.0 scale.

Illustrative Calculation. If CVSS = 7.5, EPSS = 0.4, and both weights = 1.0:

$$\text{Combined Severity} = \text{round}\left(\frac{7.5 + (10 \times 0.4)}{2}\right) = \text{round}(5.75) = 6.$$

Depending on internal thresholds, that might be classified as "Medium" or "High."

2) Algorithm to Compute Classification

1. **Data Retrieval**: Gather CVSS and EPSS from sources (NVD, GitHub Advisory Database, OSV, etc.).
2. **Scaling**: Multiply EPSS by 10.
3. **Combination**: Apply the chosen formula to produce a single score.
4. **Fallback**: If key data is missing, assign a placeholder score. This placeholder score is not displayed to the user. Instead, the frontend labels the severity in pink as "UNKNOWN", indicating that the severity could not be calculated. Since the maximum severity score is 10, a placeholder value of 10.1 ensures that the vulnerability is listed first on the dashboard.

3) Remediation Model (SSVC-inspired)

This model generates actions such as "apply patch immediately," "monitor regularly," or "deprioritize," guided by role-based decision flows. Below is a short example of such a flow for a developer:

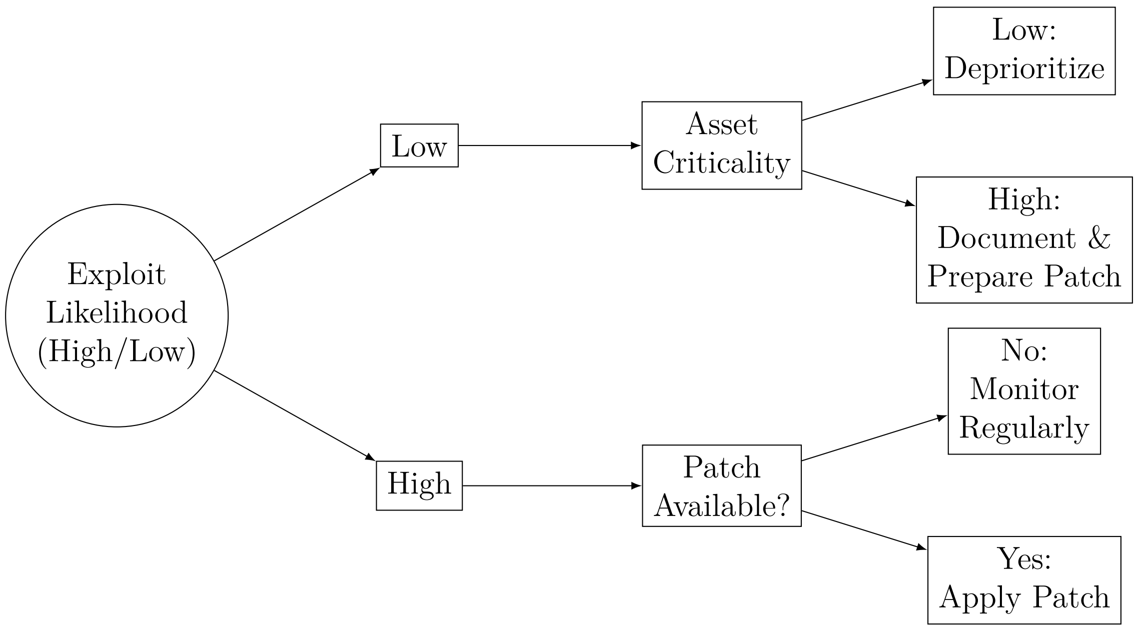

Figure 1: Abbreviated SSVC decision tree example (developer role).

Combined Outcomes (Sample Code Snippets).

- *Developer: Apply fix ASAP. Use vendor patch. High-critical system.*
- *Developer: Investigate further. No vendor patch. Coordinate with QA/team.*
- *Security-Advisor: Immediate patch. Active exploits detected. Ensure regulatory compliance.*

4) Rank-Ordering Mechanism

All vulnerabilities receive the final score (from Section 1 or 2). They are then sorted in descending order. If data is missing (*CVSS*, *EPSS*), a placeholder score (e.g., 10.1) elevates that entry to the top, prompting resolution of data gaps or running the SSVC assignment.

Questionnaire

Instructions: You can type into the text fields below (if supported by your PDF reader) or print and fill them by hand.

A) Multi-Dimensional Classification Model

1. **How clear or useful is it to merge CVSS and EPSS into a single severity score?**

2. **What approximate weightings (for general applications) would you choose for CVSS vs. EPSS, and why?**

3. **How would you rate the Multi-Dimensional Classification Model (1 = very good, 6 = insufficient)?**

B) Algorithm to Compute Classification

1. **Is the fallback (placeholder score) for missing data beneficial, or could it lead to confusion?**

2. **Please provide an instance where the computed severity might differ from your organization's internal approach or policy.**

3. **How would you rate the Algorithm to Compute Classification (1 = very good, 6 = insufficient)?**

C) Remediation Model (SSVC-inspired)

1. Do the (radically simplified) example SSVC decision flows generally align with how you handle vulnerabilities (e.g., developer vs. security advisor)?

2. **In the SSVC approach, which decision factors (e.g., Exploit Likelihood, Patch Availability, Asset Criticality) do you consider most critical for accurate**

remediation (for developers and security advisors)?

3. **How would you rate the Remediation Model (SSVC-inspired) (1 = very good, 6 = insufficient)?**

D) Rank-Ordering Mechanism

1. **Does placing highest-scored or unknown-data items on top match your typical prioritization approach?**

2. **Would grouping vulnerabilities by technology stack, business unit, or other factors be more practical?**

3. **How would you rate the Rank-Ordering Mechanism (1 = very good, 6 = insufficient)?**

E) Overall Perspective

1. **Do these integrated models (classification, remediation, rank-ordering) reflect how you would generally manage open-source vulnerabilities?**

2. **Are there any technical or organizational considerations missing that should be included?**

3. **Additional remarks or suggestions:**

Name, Role, and Signature

Organization: Capgemini Outsourcing Services GmbH

Name:

Role in Organization:

Date: **Signature:**

References (Questionnaire)

- FIRST.org (2024). “Exploit Prediction Scoring System.” `https://www.first.org/epss/`
- NIST (2024). “National Vulnerability Database.” `https://nvd.nist.gov/`
- GitHub (2024). “GitHub Advisory Database.” `https://github.com/github/advisory-database`
- OSV (2024). “Open Source Vulnerabilities Database.” `https://osv.dev/`