Towards Usable Scoring of Common Weaknesses

Olutola Adebiyi[a] and Massimiliano Albanese[b]
Center for Secure Information Systems, George Mason University, Fairfax, U.S.A.
oadebiy@gmu.edu, malbanese@gmu.edu

Keywords: Vulnerability Scanning, Security Metrics, Software Weaknesses.

Abstract: As the number and severity of security incidents continue to increase, remediating vulnerabilities and weaknesses has become a daunting task due to the sheer number of known vulnerabilities. Different scoring systems have been developed to provide qualitative and quantitative assessments of the severity of common vulnerabilities and weaknesses, and guide the prioritization of vulnerability remediation. However, these scoring systems provide only generic rankings of common weaknesses, which do not consider the specific vulnerabilities that exist in each system. To address this limitation, and building on recent principled approaches to vulnerability scoring, we propose new common weakness scoring metrics that consider the findings of vulnerability scanners, including the number of instances of each vulnerability across a system, and enable system-specific rankings that can provide actionable intelligence to security administrators. We built a small testbed to evaluate the proposed metrics against an existing metric, and show that the results are consistent with our intuition.

1 INTRODUCTION

The increasing volume of common vulnerabilities and exposures entries published yearly can hinder security administrators from quickly prioritizing those that pose the highest risk to their organization. Several efforts have been made by different organizations, including NIST (Mell et al., 2006) and MITRE (Christey, 2008), to define metrics for scoring vulnerabilities and helping administrators make informed decisions about vulnerability prioritization, remediation, and mitigation. However, organizations still struggle to properly quantify and prioritize their vulnerabilities because of the many factors they need to consider. Additionally, available tools and scoring systems rely on predefined notions of risk and impact and use predefined equations to capture the primary characteristics of a vulnerability, giving administrators very little flexibility.

The first step in vulnerability management is to understand which types of vulnerabilities are the most critical for the security of an organization. MITRE's Common Weakness and Enumeration (CWE) provides a way to abstract software-specific flaws into broader classes of vulnerabilities, referred to as weaknesses, and rank these classes of vulnerabilities by their aggregate severity. Rankings of software weaknesses are published yearly by MITRE and OWASP, but these rankings do not effectively assist administrators in prioritizing remediation efforts, as many of the vulnerabilities mapped to the top-ranking weaknesses may not exist in their own system.

Many organizations also run periodic vulnerability scans to discover unpatched vulnerabilities, but without proper prioritization, the results of vulnerability scanning may not help protect critical assets from cyber attacks. According to a recent study, only about 1.4% of published vulnerabilities are known to have been exploited (Sabottke et al., 2015).

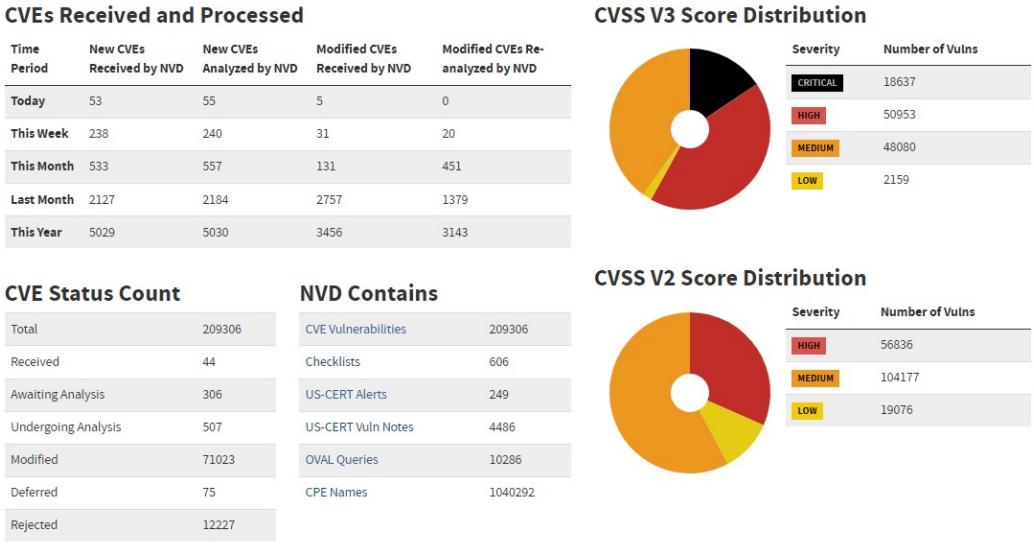
To effectively prioritize vulnerabilities and support security-related decisions that are specific to target system, it is important to identify and score the vulnerabilities that exist on that system rather than using generic rankings. Information about new vulnerabilities is constantly updated in NVD, but scoring of new vulnerabilities does not keep the pace with the rate at which new vulnerabilities are discovered. It is possible to have vulnerabilities on the system which are yet to be assigned a CVSS score. Figure 1 shows that over 800 vulnerabilities are yet to be scored as of March 8, 2023.[1]

The Mason Vulnerability Scoring Framework (MVSF) attempted to addresses some of these concerns by establishing a framework that allows users

[a] https://orcid.org/0000-0002-3207-5096
[b] https://orcid.org/0000-0002-2675-5810

[1]The live NVD Dashboard can be accessed at https://nvd.nist.gov/general/nvd-dashboard

Adebiyi, O. and Albanese, M.
Towards Usable Scoring of Common Weaknesses.
In *Proceedings of the 20th International Conference on Security and Cryptography (SECRYPT 2023)*, pages 183-191
ISBN: 978-989-758-666-8; ISSN: 2184-7711
Copyright © 2023 by SCITEPRESS – Science and Technology Publications, Lda. Under CC license (CC BY-NC-ND 4.0)



CVEs Received and Processed

Time Period	New CVEs Received by NVD	New CVEs Analyzed by NVD	Modified CVEs Received by NVD	Modified CVEs Re-analyzed by NVD
Today	53	55	5	0
This Week	238	240	31	20
This Month	533	557	131	451
Last Month	2127	2184	2757	1379
This Year	5029	5030	3456	3143

CVSS V3 Score Distribution

Severity	Number of Vulns
CRITICAL	18637
HIGH	50953
MEDIUM	48080
LOW	2159

CVE Status Count

Total	209306
Received	44
Awaiting Analysis	306
Undergoing Analysis	507
Modified	71023
Deferred	75
Rejected	12227

NVD Contains

CVE Vulnerabilities	209306
Checklists	606
US-CERT Alerts	249
US-CERT Vuln Notes	4486
OVAL Queries	10286
CPE Names	1040292

CVSS V2 Score Distribution

Severity	Number of Vulns
HIGH	56836
MEDIUM	104177
LOW	19076

Figure 1: Screenshot of the NVD Dashboard as of 6:00pm EDT on March 08, 2023.

to generate custom ranking by tuning several parameters used to calculate vulnerability and weakness scores (Iganibo et al., 2022). MVSF builds on previous work aimed at identifying variables that influence an attacker's decision to exploit a given vulnerability (Iganibo et al., 2021) but it is limited by the lack of integration with vulnerability scanning tools, which would enable customization of the rankings based on the specific vulnerabilities that exist in a given system. We address this limitation by designing a simple yet elegant and effective solution to integrate information from vulnerability scanning into the weakness scoring process. We propose two new scoring metrics that consider vulnerability scanning information at two different levels of granularity. The first metric considers average exploitation likelihoods and exposure factors across detected vulnerabilities, whereas the second metric computes weighted averages using the numbers of instances of each vulnerability as weights. The results on a small testbed indicate that these metrics can provide rankings consistent with intuition, and thus can help identify the most critical weaknesses for an organization.

The remainder of this paper is organized as follows. Section 2 discusses related work. Section 3 provides preliminary definitions and describes the base metrics which this work builds upon. Then, Section 4 describes the new metrics for scoring weaknesses and the rationale behind them. Finally, Section 5 describes the experimental setup and results, and Section 6 provides some concluding remarks and a roadmap for future work.

2 RELATED WORK

Vulnerability management aims at effectively and intelligently prioritizing remediation efforts based on actionable recommendations that consider both external variables and intrinsic features of existing vulnerabilities. Organizations have made attempts to score and rank software vulnerabilities, and find ways to assess and quantify their impact.

The U.S. government, through the National Institute of Standards and Technology maintains the National Vulnerability Database (NVD) which is a repository of vulnerability information. NVD is fully synchronized with MITRE Common Vulnerabilities and Exposures (CVE) list, and augments it with severity scores, impact ratings based on the Common Vulnerability Scoring System (CVSS). Recent work has shown the limitations of CVSS. (Ruohonen, 2019) highlighted the time delays that affect CVSS scoring work in the context of NVD. (Spring et al., 2021) indicated that the CVSS formula lacks empirical justification and does not address some risk elements.

Common Weakness Enumeration (CWE) provides a community-developed list of software and hardware weaknesses[2]. MITRE's Common Weakness Scoring System (CWSS) provides a mechanism for prioritizing software weaknesses in a consistent, flexible, open manner through a collaborative community-based effort[3].

The CWE team leverages the Common Vulnerabilities and Exposures data and the CVSS scores as-

[2]https://cwe.mitre.org/

[3]https://cwe.mitre.org/cwss/

sociated with each CVE record to publish a list of Top 25 Most Dangerous Software Weaknesses [4]. The aim is to provide security professionals with resources to help mitigate risk in their organizations. Similarly, the Open Web Application Security Project (OWASP) releases its Top 10 Web Application Security Risks list yearly[5]. The OWASP Top 10 does not not consider the number of instances of a CWE in calculation for the top 10. The ranking is subjective and difficult to replicate by users as there is no published quantitative approach to back it up.

Several metrics (Wang et al., 2009; Mukherjee and Mazumdar, 2018; Wang et al., 2019) use scores from the Common Vulnerability Scoring Systems (CVSS) or the Common Weakness Scoring Systems (CWSS) in isolation or as the dominant factor in determining the severity of a vulnerability. (Jacobs et al., 2021) developed a data-driven threat scoring system for predicting the probability that a vulnerability will be exploited within the 12 months following public disclosure. Most of these metrics cannot be easily extended to consider the effect of additional variables, and do not specifically focus on system-centric evaluations. Our metrics are designed to provide system-specific rankings of common weaknesses, building upon extensible vulnerability-level metrics.

3 PRELIMINARY DEFINITIONS

In this section, we provide an overview of generalized versions of the base vulnerability metrics that were originally introduced in (Iganibo et al., 2021), namely the *exploitation likelihood* and the *exposure factor* of a vulnerability. We also provide an overview of the metrics used by MITRE to score and rank CWEs and a generalized version of a similar CWE scoring metric introduced in (Iganibo et al., 2022). The work presented in this paper builds upon these metrics, but offers a simple yet elegant solution to make vulnerability and weakness scoring more useful in practice.

3.1 Exploitation Likelihood

The *exploitation likelihood* (or simply *likelihood*) $\rho(v)$ of a vulnerability v is defined as the probability that an attacker will attempt to exploit that vulnerability, if given the opportunity. An attacker has the *opportunity* to exploit a vulnerability if certain preconditions are met, most notably if they have access to the vulnerable host. Specific preconditions may vary depending on the characteristics of each vulnerability, as certain configuration settings may prevent access to vulnerable portions of the target software. Several variables may influence the likelihood that an attacker will exploit a given vulnerability v, including but not limited to: (i) the vulnerability's exploitability score as determined by CVSS, $\mathsf{Exploitability}(v)$; (ii) the amount of time elapsed since the vulnerability was made public, $\mathsf{Age}(v)$; and (iii) the set of known IDS rules associated with the vulnerability, $\mathsf{Known\_IDS\_Rules}(v)$. Formally, the exploitation likelihood is modeled as a function $\rho : V \rightarrow [0,1]$ defined by Equation 1.

$$\rho(v) = \frac{\prod_{X \in \mathcal{X}_l^{\uparrow}} \left(1 - e^{-\alpha_X \cdot f_X(X(v))}\right)}{\prod_{X \in \mathcal{X}_l^{\downarrow}} e^{\alpha_X \cdot f_X(X(v))}} \tag{1}$$

where $\mathcal{X}_l^{\uparrow}$ and $\mathcal{X}_l^{\downarrow}$ denote the sets of variables that respectively contribute to increasing and decreasing the likelihood as their values increase. Each variable contributes to the overall likelihood as a multiplicative factor between 0 and 1 that is formulated to account for *diminishing returns*. Factors corresponding to variables in $\mathcal{X}_l^{\uparrow}$ are of the form $1 - e^{-\alpha_X \cdot f_X(X(v))}$, where X is the variable, α_X is a tunable parameter, $X(v)$ is the value of X for v, and f_X is a monotonically increasing function used to convert values of X to scalar values, i.e., $x_1 < x_2 \implies f_X(x_1) \leq f_X(x_2)$. Similarly, factors corresponding to variables in $\mathcal{X}_l^{\downarrow}$ are of the form $\frac{1}{e^{\alpha_X \cdot f_X(X(v))}} = e^{-\alpha_X \cdot f_X(X(v))}$. It is assumed that each product evaluates to 1 when the corresponding set of variables is empty, i.e., $\prod_{X \in \mathcal{X}}(\ldots) = 1$ when $\mathcal{X} = \emptyset$.

For the analysis presented in this paper, we selected the three variables mentioned earlier, that is $\mathcal{X}_l^{\uparrow} = \{\mathsf{Exploitability}, \mathsf{Age}\}$ and $\mathcal{X}_l^{\downarrow} = \{\mathsf{Known\_IDS\_Rules}\}$, thus Equation 1 can be instantiated as follows. However, we remind the reader that the proposed approach for ranking CWEs is independent of the specific variables chosen for $\mathcal{X}_l^{\uparrow}$ and $\mathcal{X}_l^{\downarrow}$.

$$\rho(v) = \frac{\prod_{X \in \{\mathsf{Exploitability}, \mathsf{Age}\}} \left(1 - e^{-\alpha_X \cdot f_X(X(v))}\right)}{\prod_{X \in \{\mathsf{Known\_IDS\_Rules}\}} e^{\alpha_X \cdot f_X(X(v))}} \tag{2}$$

We then define $f_{\mathsf{Exploitability}}(v) = \mathsf{Exploitability}(v)$, $f_{\mathsf{Age}}(v) = \sqrt{\mathsf{Age}(v)}$, and $f_{\mathsf{Known\_IDS\_Rules}}(v) = |\mathsf{Known\_IDS\_Rules}(v)|$, but discussing how these functions are defined for each variable is beyond the scope of this work. To simplify the notation, we use α_{E}, α_{A}, and α_{K} to refer to $\alpha_{\mathsf{Exploitability}}$, α_{Age}, and $\alpha_{\mathsf{Known\_IDS\_Rules}}$ respectively. Thus, Equation 2 can be rewritten as follows.

[4] https://cwe.mitre.org/top25/

[5] https://owasp.org/Top10/

$$\rho(v) = \frac{\left(1 - e^{-\alpha_E \cdot \mathsf{Exploitability}(v)}\right) \cdot \left(1 - e^{-\alpha_A \cdot \sqrt{\mathsf{Age}(v)}}\right)}{e^{\alpha_K \cdot |\mathsf{Known\_IDS\_Rules}\}(v)|}} \quad (3)$$

3.2 Exposure Factor

The *exposure factor* $ef(v)$ of a vulnerability v is defined as the relative loss of utility of an asset due to a vulnerability exploit. The term is borrowed from classic risk analysis terminology, where the exposure factor (EF) represents the relative damage that an undesirable event – a cyber attack in our case – would cause to the affected asset. The single loss expectancy (SLE) of such an incident is then computed as the product between its exposure factor and the asset value (AV), that is $SLE = EF \times AV$. Several variables may increase or decrease the exposure factor of a vulnerability v, including but not limited to: (i) the vulnerability's impact score as captured by CVSS, $\mathsf{Impact}(v)$; and the set of deployed IDS rules associated with the vulnerability, $\mathsf{Deployed\_IDS\_Rules}(v)$. Formally, the exposure factor is defined as a function $ef : V \rightarrow [0,1]$ defined by Equation 4.

$$ef(v) = \frac{\prod_{X \in \mathcal{X}_e^{\uparrow}} \left(1 - e^{-\alpha_X \cdot f_X(X(v))}\right)}{\prod_{X \in \mathcal{X}_e^{\downarrow}} e^{\alpha_X \cdot f_X(X(v))}} \quad (4)$$

where $\mathcal{X}_e^{\uparrow}$ and $\mathcal{X}_e^{\downarrow}$ denote the sets of variables that respectively contribute to increasing and decreasing the exposure as their values increase. Similar to the likelihood, each variable contributes to the exposure factor as a multiplicative factor between 0 and 1 that accounts for diminishing returns. Factors corresponding to variables in $\mathcal{X}_e^{\uparrow}$ are of the form $1 - e^{-\alpha_X \cdot f_X(X(v))}$, and factors corresponding to variables in $\mathcal{X}_e^{\downarrow}$ are of the form $\frac{1}{e^{\alpha_X \cdot f_X(X(v))}} = e^{-\alpha_X \cdot f_X(X(v))}$.

For our analysis, we considered the CVSS impact score as the only variable affecting the exposure factor, that is $\mathcal{X}_e^{\uparrow} = \{\mathsf{Impact}\}$ and $\mathcal{X}_e^{\downarrow} = \emptyset$, thus Equation 4 can be instantiated as follows. However, as mentioned earlier, the proposed approach for ranking CWEs is independent of the specific variables chosen for $\mathcal{X}_e^{\uparrow}$ and $\mathcal{X}_e^{\downarrow}$.

$$ef(v) = \prod_{X \in \{\mathsf{Impact}\}} \left(1 - e^{-\alpha_X \cdot f_X(X(v))}\right) \quad (5)$$

We then define $f_{\mathsf{Impact}}(v) = \mathsf{Impact}(v)$ and, to simplify the notation, we use α_{I} to refer to α_{Impact}. Thus, Equation 5 can be rewritten as follows.

$$ef(v) = 1 - e^{-\alpha_{\mathsf{I}} \cdot \mathsf{Impact}(v)} \quad (6)$$

3.3 Common Weakness Score

MITRE publishes a yearly ranking of the top 25 most dangerous software weaknesses. Each weakness is scored based on the number of vulnerabilities mapped to that weakness and the average severity of such vulnerabilities. The score proposed in (Iganibo et al., 2022) is semantically equivalent to MITRE's score, but relies on the more general vulnerability metrics introduced in (Iganibo et al., 2021), which allow administrators to control the ranking by fine-tuning several parameters used in the computation of vulnerability-level metrics, whereas MITRE's score relies on fixed severity scores from CVSS. In the following, we provide the background on CWE ranking and an overview of the state of the art.

Equation 9 defines the set of CVEs mapped to each CWE W_i, and Equation 10 defines the *frequencies* of all CWEs, that is the set of cardinalities of the sets of vulnerabilities associated with each CWE. This set of frequencies is used in Equation 11 to derive a normalization factor $max(Freqs) - min(Freqs)$.

$$C(W_i) = \{CVE_j \in NVD, CVE_j \rightarrow W_i\} \quad (9)$$

$$Freqs = \{|C(W_i)|, W_i \in NVD\} \quad (10)$$

Equations 11 and 12 respectively compute a frequency and a severity score for each CWE, where the severity is based on the average CVSS score of all CVEs in that CWE category. Frequency and severity scores are both normalized between 0 and 1.

$$Fr(W_i) = \frac{|C(W_i)| - min(Freqs)}{max(Freqs) - min(Freqs)} \quad (11)$$

$$Sv_{MITRE}(W_i) = \frac{\text{avg}_{W_i}(CVSS) - min(CVSS)}{max(CVSS) - min(CVSS)} \quad (12)$$

Finally, Equation 13 defines the overall score that MITRE assigns to each CWE as the product of its frequency and severity scores, normalized between 0 and 100.

$$S_{MITRE}(W_i) = Fr(W_i) \cdot Sv_{MITRE}(W_i) \cdot 100 \quad (13)$$

In (Iganibo et al., 2022), the severity of a weakness W_i is defined as the product of the average likelihood $\rho(W_i)$ of vulnerabilities mapped to W_i and the average exposure factor $ef(W_i)$ of such vulnerabilities, as shown in Equation 14 below.

$$Sv_{MVSF}(W_i) = \rho(W_i) \cdot ef(W_i) \quad (14)$$

$$S_{MVSF}(W_i) = |C(W_i)| \cdot \underset{v \in C(W_i)}{\text{avg}} \frac{\prod_{X \in \mathcal{X}_l^{\uparrow}} \left(1 - e^{-\alpha_X \cdot f_X(X(v))}\right)}{\prod_{X \in \mathcal{X}_l^{\downarrow}} e^{\alpha_X \cdot f_X(X(v))}} \cdot \underset{v \in C(W_i)}{\text{avg}} \frac{\prod_{X \in \mathcal{X}_e^{\uparrow}} \left(1 - e^{-\alpha_X \cdot f_X(X(v))}\right)}{\prod_{X \in \mathcal{X}_e^{\downarrow}} e^{\alpha_X \cdot f_X(X(v))}} \tag{7}$$

$$S_{MVSF}(W_i) = |C(W_i)| \cdot \underset{v \in C(W_i)}{\text{avg}} \frac{\left(1 - e^{-\alpha_{\mathsf{E}} \cdot \mathsf{Exploitability}(v)}\right) \cdot \left(1 - e^{-\alpha_{\mathsf{A}} \cdot \sqrt{\mathsf{Age}(v)}}\right)}{e^{\alpha_{\mathsf{K}} \cdot |\mathsf{Known\_IDS\_Rules}\}(v)|}} \cdot \underset{v \in C(W_i)}{\text{avg}} \left(1 - e^{-\alpha_{\mathsf{I}} \cdot \mathsf{Impact}(v)}\right) \tag{8}$$

where $\rho(W_i)$ and $ef(W_i)$ are defined by Equations 15 and 16 respectively.

$$\rho(W_i) = \underset{v \in C(W_i)}{\text{avg}} \rho(v) \tag{15}$$

$$ef(W_i) = \underset{v \in C(W_i)}{\text{avg}} ef(v) \tag{16}$$

Thus, an alternative common weakness score is given by Equation 17, where the frequency $Fr(W_i)$ in Equation 13 is replaced by $|C(W_i)|$ and the average severity $Sv_{MITRE}(W_i)$ is replaced by $Sv_{MVSF}(W_i)$.

$$S_{MVSF}(W_i) = |C(W_i)| \cdot Sv_{MVSF}(W_i) \tag{17}$$

Combining Equations 1, 4, 15, 16, and 17, we can rewrite Equation 17 as Equation 7, which provides a generalization of the the expression for $S_{MVSF}(W_i)$ that was introduced in (Iganibo et al., 2022). We will use this score as the baseline for our analysis, but before introducing the proposed approach to CWE ranking and the new metrics, we can instantiate Equation 7 based on the set of variables chosen for our analysis. Thus, considering Equations 3 and 6, we can rewrite the expression for $S_{MVSF}(CWE_i)$ as Equation 8.

4 NEW METRICS

In this section, we propose two new metrics, referred to as the *system weakness score* and the *weighted weakness score*, that can turn CWE rankings into actionable intelligence for system administrators by integrating information gained through vulnerability scanning into the ranking process. The proposed approach refines the CWE scoring metric defined in Section 3.3 by only considering the vulnerabilities that were identified in the target system. To further refine the analysis, the second metric also considers the number of instances of each discovered vulnerability. In the following, we use C_s to denote the set of distinct CVEs identified during vulnerability scanning. We then use $C_s(W_i)$ to denote the set of CVEs in W_i that were identified by the scanner, that is $C_s(W_i) = C_S \cap C(W_i)$.

4.1 System Weakness Score

The score of a CWE category, as defined by Equation 7 or its instance Equation 8, is computed by considering all CVEs in that CWE category. However, as discussed earlier, any real-world system may only expose a limited number of such vulnerabilities, making the resulting score of limited utility for system administrators. Thus, we redefine the weakness-level likelihood and exposure factor used in Equation 14 using Equations 18 and 19 below, which compute averages only over the vulnerabilities identified by the scanner.

$$\rho(W_i) = \underset{v \in C_s(W_i)}{\text{avg}} \rho(v) \tag{18}$$

$$ef(W_i) = \underset{v \in C_s(W_i)}{\text{avg}} ef(v) \tag{19}$$

Accordingly, we defined a new CWE scoring metrics as follow.

$$S^{\dagger}(W_i) = |C_s(W_i)| \cdot \underset{v \in C_s(W_i)}{\text{avg}} \rho(v) \cdot \underset{v \in C_s(W_i)}{\text{avg}} ef(v) \tag{20}$$

The specific sets of variables used in the computation of the vulnerability-level likelihood and exposure factor are independent of the CWE scoring metric. For the purpose of our analysis, we will continue to use the variables identified in Section 3. Thus, combining Equations 3, 6, and 20, we obtain Equation 21.

4.2 Weighted Weakness Score

The metric introduced in the previous section and defined by Equation 20 provides more actionable intelligence to system administrators than the original metric, as it only considers vulnerabilities that were found in the system. However, in large systems, the same vulnerabilities may exist on different hosts, and some vulnerabilities may be more common than others. Therefore a seemingly less severe vulnerability may require attention if it is present on a large number of machines across the network, offering attackers multiple potential entry points. To address this concern, we further refine our metric to consider the number of instances of each vulnerability identified by the scanner.

$$S^{\dagger}(W_i) = |C_s(W_i)| \cdot \underset{v \in C_s(W_i)}{\text{avg}} \frac{\left(1 - e^{-\alpha_{\mathsf{E}} \cdot \mathsf{Exploitability}(v)}\right) \cdot \left(1 - e^{-\alpha_{\mathsf{A}} \cdot \sqrt{\mathsf{Age}(v)}}\right)}{e^{\alpha_{\mathsf{K}} \cdot |\mathsf{Known\_IDS\_Rules}\}(v)|}} \cdot \underset{v \in C_s(W_i)}{\text{avg}} \left(1 - e^{-\alpha_{\mathsf{I}} \cdot \mathsf{Impact}(v)}\right) \quad (21)$$

Let $I(v)$ denote the number of instances of vulnerability v identified by the scanner across all hosts. We can then redefine the weakness-level likelihood and exposure factor used in Equation 14 through Equations 22 and 23 below, which compute weighted averages over the vulnerabilities identified by the scanner, using the numbers of instances as weights.

$$\rho(W_i) = \frac{\sum_{v \in C_s(W_i)} I(v) \cdot \rho(v)}{\sum_{v \in C_s(W_i)} I(v)} \quad (22)$$

$$ef(W_i) = \frac{\sum_{v \in C_s(W_i)} I(v) \cdot ef(v)}{\sum_{v \in C_s(W_i)} I(v)} \quad (23)$$

Accordingly, Equation 24 defines a new CWE scoring metric that uses the weakness-level likelihood and exposure factor defined by Equations 22 and 23.

As mentioned for the previous metric, the specific sets of variables used in the computation of the vulnerability-level likelihood and exposure factor are independent of the CWE scoring metric. For the purpose of our analysis and for consistency with the $S^{\dagger}$ metric, we will continue to use the variables identified in Section 3. Thus, combining Equations 3, 6, and 24, we can obtain an equation similar to Equations 8 and 21, but we omit it due to the complexity of the expression.

5 EVALUATION

This section describes the experiments we conducted to evaluate the proposed metrics. First, we briefly describe the experimental setup, and then present the results in detail.

5.1 Experimental Setup

The test environment consisted of of a set of virtual machines with different operating systems, including Windows and Linux, and different sets of exposed vulnerabilities. Table 1 provides a summary of the machines used, including information about operating system and exposed vulnerabilities. These machines were scanned using Nessus Vulnerability Scanner to obtain an inventory of existing system vulnerabilities. Nessus uses scripts written in Nessus Attack Scripting Language (NASL) to detect vulnerabilities as they are discovered and released into the public domain. These scripts, referred to as plugins[6], include information about the vulnerability, set of remediation actions, and the algorithm to test for the presence of the security issue.

Table 1: Test machines using in the evaluation.

ID	Operating System	# CVEs
1	Windows 11	1
2	Windows 10	0
3	Windows Server 2008 R2	6
4	Windows Server 2008 R2	18
5	Linux Kernel 2.6 on Ubuntu 14.04	1
6	Linux Kernel 3.13 on Ubuntu 14.04	7
7	Linux Kernel 2.6 on Ubuntu 8.04	39

5.2 Results

A total of 72 vulnerability instances were identified on the 7 machines in our test environment, including 52 distinct CVEs. These CVEs fall under different CWE categories, as shown in Table 2. Note that NVD is only using a subset of CWE for mapping instead of the entire CWE list. Vulnerabilities that are not mapped to any CWE in this subset are included in a special category, NVD-CWE-Other. For each CWE, Table 2 shows the total number of CVEs mapped to that CWE, that is $|C(W_i)|$ and the number of CVEs found by the scanner, that is $|C_s(W_i)|$. The first is used as a basis for computing S_{MVSF} whereas the second is used in the computation of our new metrics.

For each identified CWE category, we computed the baseline score S_{MVSF} and the two new scores $S^{\dagger}$ and S^{*}. Table 2 provides a summary of the results, which reveal that different scoring metrics may lead to significantly different outcomes in terms of ranking. We can also notice that scores computed using the baseline metric may be orders of magnitude larger than our new scores. This is expected as $S_{MVSF}(W_i)$ is heavily dominated by the total number of CVEs in W_i, which is usually large.

Common weakness scoring based on the vulnerabilities that exist on a system can provide useful insights for the organization. When researchers, vendors or users report identified vulnerabilities, their scores are calculated based on metrics that approximate ease and impact of an exploit. The scoring does not consider if the vulnerability exists on a specific system or a safeguard has been implemented to protect the system. When vulnerability-level scores

[6]https://www.tenable.com/plugins/families/about

$$S^*(W_i) = |C_s(W_i)| \cdot \frac{\sum_{v \in C_s(W_i)} I(v) \cdot \rho(v)}{\sum_{v \in C_s(W_i)} I(v)} \cdot \frac{\sum_{v \in C_s(W_i)} I(v) \cdot ef(v)}{\sum_{v \in C_s(W_i)} I(v)} \tag{24}$$

Table 2: Common Weakness Scores Summary.

CWE ID	# CVEs (total)	# CVEs (scan)	S_{MVSF}	$S^\dagger$	S^*
CWE-16	74	1	54	0.68	0.68
CWE-20	4,306	14	3,145	7.64	7.73
CWE-74	568	2	421	1.76	0.88
CWE-79	12,633	1	8,341	0.68	0.68
CWE-89	4,982	1	4,093	0.88	0.88
CWE-94	1,252	1	1,037	0.88	0.88
CWE-200	2,613	14	1,709	3.99	3.65
CWE-254	9	2	6	1.65	1.65
CWE-264	703	2	501	1.70	1.13
CWE-284	187	2	137	1.76	1.76
CWE-310	196	8	143	3.59	3.77
CWE-326	223	2	155	0.68	0.68
CWE-327	262	2	180	0.70	0.70
CWE-331	27	1	19	0.70	0.70
CWE-400	1,021	1	725	0.70	0.70
CWE-617	290	2	200	1.34	1.34
CWE-732	818	2	559	1.48	1.48
CWE-787	6,478	1	5,029	0.88	0.88
Others	1,839	13	1,307	8.91	9.03

are aggregated to compute weakness level-scores, one may risk to lose sight of the problem at hand. In fact, weakness ranking approaches like MITRE's CWE Top 25 effort, while providing useful information about general trends in the security landscape, cannot provide system-specific actionable insights to administrators.

In order to help identify the most severe weaknesses that threaten a specific system and guide the prioritization of vulnerability remediation, we must only consider vulnerabilities that actually exist in the system when abstracting vulnerability-level metrics into weakness level metrics. The proposed metrics address this need, with the second metric going a step further and giving more weight to vulnerabilities that exists on multiple machines across the network.

In the following, we discuss how the different scoring metrics impact the ranking of CWEs. Tables 3, 4, and 5 show how the CWEs identified in our test environment rank based on S_{MVSF}, $S^\dagger$, and S^* respectively. The ranking based on S_{MVSF} is similar to the most recent version of MITRE's CWE Top 25. This result is expected as the authors of (Iganibo et al., 2022) claim that the two rankings are semantically equivalent and are highly correlated. In particular, the top 4 CWEs are the same, with only two position switched between the two rankings.

The ranking shown in Table 4 offers some insights about the effectiveness of our approach. For instance, CWE-79 slipped from the first position to position 17

Table 3: Ranking based on S_{MVSF}.

Rank	CWE ID	S_{MVSF}
1	CWE-79	8,391
2	CWE-787	5,029
3	CWE-89	4,093
4	CWE-20	3,145
5	CWE-200	1,709
6	Others	1,307
7	CWE-94	1,037
8	CWE-400	725
9	CWE-732	559
10	CWE-264	501
11	CWE-74	421
12	CWE-617	200
13	CWE-327	180
14	CWE-326	155
15	CWE-310	143
16	CWE-284	137
17	CWE-16	54
18	CWE-331	19
19	CWE-254	6

Table 4: Ranking based on $S^\dagger$.

Rank	CWE ID	$S^\dagger$
1	Others	8.91
2	CWE-20	7.64
3	CWE-200	3.99
4	CWE-310	3.59
5	CWE-284	1.76
6	CWE-264	1.70
7	CWE-254	1.65
8	CWE-732	1.48
9	CWE-617	1.34
10	CWE-74	1.76
11	CWE-787	0.88
11	CWE-89	0.88
11	CWE-94	0.88
14	CWE-327	0.70
14	CWE-331	0.70
14	CWE-400	0.70
17	CWE-16	0.68
17	CWE-326	0.68
17	CWE-79	0.68

based on $S^\dagger$. In fact, while over 12,000 vulnerabilities are mapped to CWE-79, only a single instance of one such CVEs was found across our test environment, thus making CWE-79 less dangerous than other CWEs. On the other side, CWE-310 jumped from position 15 to the fourth position. In fact, while CWE-310 includes only less than 200 CVEs, 8 instances of such vulnerabilities were found across our test environment.

From the ranking shown in Table 5, we can observe that CWE-310 went up one more position in

the ranking, surpassing CWE-200. This can be explained by considering that, although there are more vulnerabilities mapped to CWE-200 than to CWE-310 and the average severity is comparable across the two CWEs, the vulnerability in CWE-200 with the highest number of instances has lower-than-average severity and the vulnerability in CWE-310 with the highest number of instances has higher-than-average severity, thus shifting the weighted average in favor of CWE-310.

Finally, we can observe that CWE-20 and CWE-200 rank high in all three rankings. These can be explained by considering that these two CWEs have a significant number of mapped CVEs (4.3k and 2.6k respectively) – which increases their S_{MVSF} score – and have the largest number of vulnerability instances discovered by the scanner – which increases their $S^{\dagger}$ and S^{*} scores.

Table 5: Ranking based on S^{*}.

Rank	CWE ID	S^{*}
1	Others	9.03
2	CWE-20	7.73
3	CWE-310	3.77
4	CWE-200	3.65
5	CWE-284	1.76
6	CWE-254	1.65
7	CWE-732	1.48
8	CWE-617	1.34
9	CWE-264	1.13
10	CWE-74	0.88
10	CWE-787	0.88
10	CWE-89	0.88
10	CWE-94	0.88
14	CWE-327	0.70
14	CWE-331	0.70
14	CWE-400	0.70
17	CWE-16	0.68
17	CWE-326	0.68
17	CWE-79	0.68

In summary, the analysis of these results confirms that the proposed metrics work as expected and can effectively identify the most severe weaknesses for a given system.

6 CONCLUSIONS

Building upon the existing body of work on vulnerability metrics and ranking of common weaknesses, we have proposed a simple yet elegant approach for ranking weaknesses that integrates the results of vulnerability scanning. Accordingly, we have defined two new scoring metrics to enable the generation of system-specific rankings that can provide administrators with actionable intelligence to guide vulnerability remediation. Future work may involve establishing a collaboration with MITRE to further evaluate and possibly standardize the proposed metrics within the context of the CWE framework, and working with vendors of scanning software to explore the integration of our solution into their products.

ACKNOWLEDGEMENTS

This work was funded in part by the National Science Foundation under award CNS-1822094.

REFERENCES

Christey, S. (2008). The evolution of the CWE development and research views. Technical report, The MITRE Corporation.

Iganibo, I., Albanese, M., Mosko, M., Bier, E., and Brito, A. E. (2021). Vulnerability metrics for graph-based configuration security. In *Proceedings of the 18th International Conference on Security and Cryptography (SECRYPT 2021)*, pages 259–270. SciTePress.

Iganibo, I., Albanese, M., Turkmen, K., Campbell, T., and Mosko, M. (2022). Mason vulnerability scoring framework: A customizable framework for scoring common vulnerabilities and weaknesses. In *Proceedings of the 19th International Conference on Security and Cryptography (SECRYPT 2022)*, pages 215–225, Lisbon, Portugal. SciTePress.

Jacobs, J., Romanosky, S., Edwards, B., Adjerid, I., and Roytman, M. (2021). Exploit prediction scoring system (EPSS). *Digital Threats: Research and Practice*, 2(3).

Mell, P., Scarfone, K., and Romanosky, S. (2006). Common Vulnerability Scoring System. *IEEE Security & Privacy*, 4(6):85–89.

Mukherjee, P. and Mazumdar, C. (2018). Attack difficulty metric for assessment of network security. In *Proceedings of 13th International Conference on Availability, Reliability and Security (ARES 2018)*, Hamburg, Germany. ACM.

Ruohonen, J. (2019). A look at the time delays in CVSS vulnerability scoring. *Applied Computing and Informatics*, 15(2):129–135.

Sabottke, C., Suciu, O., and Dumitraș, T. (2015). Vulnerability disclosure in the age of social media: Exploiting twitter for predicting real-world exploits. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 1041–1056.

Spring, J., Hatleback, E., Householder, A., Manion, A., and Shick, D. (2021). Time to change the CVSS? *IEEE Security & Privacy*, 19(2):74–78.

Wang, J. A., Wang, H., Guo, M., and Xia, M. (2009). Security metrics for software systems. In *Proceedings of*

the 47th Annual Southeast Regional Conference (ACM SE 2009), Clemson, SC, USA. ACM.

Wang, L., Zhang, Z., Li, W., Liu, Z., and Liu, H. (2019). An attack surface metric suitable for heterogeneous redundant system with the voting mechanism. In *Proceedings of the International Conference on Computer Information Science and Application Technology (CISAT 2018)*, volume 1168 of *Journal of Physics: Conference Series*, Daqing, China. IOP Publishing.