Proposal to recast CVSS base score equation to improve exploit prediction

Ben Church – 10006197
Mohammed Alghamdi - 10163250

Introduction

Software vulnerabilities are discovered and reported at rates which exceed security and reliability teams' capacities to address them. To make better use of limited time and resources, vulnerabilities must be prioritized based on the risk they pose to the affected companies. The Common Vulnerability Scoring System (CVSS) was developed to meet this need [1]. The CVSS provides several scores in the range 0-10 based on metrics pertaining to the vulnerability in question. These are the base score, temporal score, and environmental score. The base score conveys the risk inherent in the vulnerability by using metrics which describe both the likelihood of exploitation, and the severity of the consequences of an exploitation. The temporal and environmental scores are intended as reassessments of the base score, taking into account factors which change the associated risk such as remediation deployment or particularly sensitive systems. The base score is the most commonly used in literature as it is widely available from databases and remains constant, since the other scores provide reassessment. For these reasons, we focus on the CVSS base score. The equation used to compute the base score is shown in the appendix for convenience.

The CVSS base score considers both the likelihood and severity of exploitation occurring using two sub-scores, the Exploitability and Impact sub-scores, respectively. Each of these sub-scores is computed from three corresponding metrics. For the Exploitability sub-score, these metrics are the Access Vector, Access Complexity, and Authentication. The Access Vector metric describes how an attacker can access the vulnerability, whether it can be done remotely through a network, through an adjacent network, or whether local access is required. The Access Complexity is the most subjective metric, indicating whether the complexity of exploiting the vulnerability is low, medium, or high. The last Exploitability metric is Authentication, indicating whether multiple, a single, or zero instances of user authentication are required to exploit the vulnerability. The three Impact metrics examine each of three attributes of a computer system, Confidentiality, Integrity, and Availability. The score for each metric depends on whether the corresponding system attribute is either completely, partially but not completely, or not at all compromised.

The final, one-dimensional score in the range 0-10 is intended to make prioritization of vulnerabilities simple, but it does not convey all the information available from the individual metrics values. For this reason, whenever a database records CVSS base scores for its

vulnerability entries, the values for these metrics are presented with the score. This may be important information for an analysis of the CVSS since its metrics will not always correspond to their intended purposes. For example, whatever the Exploitability sub-score, if the vulnerability offers no or limited capacity to compromise aspects of the system, there will likely be little incentive for an attacker to exploit it. The CVSS base score metrics are apparently chosen so as to eliminate subjective variability in vulnerability scoring, with the notable exception of the Access Complexity metric. For example, zero, one, or more than one instance of user authentication is objective and quantitative, as is zero, partial but bounded or incomplete, or completely compromised system confidentiality. Despite the information considered and available from the base score, and the efforts made to design it for standardization through the use of mostly quantitative and objective metrics, there are issues which have impeded the widespread acceptance of the CVSS.

Problem Description

Vulnerabilities' CVSS base scores are often poorly correlated with instances of exploitation of these vulnerabilities. Allodi and Massacci [2], investigated the effectiveness of the CVSS for predicting real exploitation in a retroactive study. They examined entries in the National Vulnerability Database (NVD - https://nvd.nist.gov/) which reports a large number of vulnerabilities with their CVSS base score. The frequencies of these vulnerabilities' appearances in several databases reporting exploits in the wild by vulnerability allowed them to correlate CVSS scores with exploitation. They found that prioritizing vulnerabilities based on their CVSS base scores was equivalent to fixing vulnerabilities in a random order in terms of the amount of exploits which would have been prevented.

Therefore the CVSS base score does not fulfill its intended function of conveying the risk inherent in the vulnerabilities. Some companies respond to the vulnerability prioritization problem with scoring systems of their own. For example, Younis *et al.* [3] investigated Mozilla, and Google vulnerability reward programs (VRPs), reasoning that principles of economic selection at least warrant investigation into comparing CVSS base scores with the VRP ratings. Moreover, they remarked that Chrome vulnerability reports provided with proof of exploitation are more likely to result in a financial reward. Different organizations resorting to their own vulnerability ranking systems rather than using the CVSS, a scoring system purpose-designed for standardization of vulnerability comparison, reflects the power of economics over academic inclination. However, the issue of standardization remains if one hopes create opportunities for cooperation and specialization among corporations facing an increasing number of vulnerabilities to address, or to develop tools as a third party for others to use.

The form of the CVSS base score equation is meant to reflect both the risk of a vulnerability being exploited, and the severity of the consequences if exploitation occurs. Arguably, with the exception of the Access Complexity metric, the metrics used to compute the base score are

quantitative and unambiguous. For example, there is should be no ambiguity in whether a vulnerability requires zero, a single, or multiple instances of user authentication to exploit. The impact metrics, pertaining to system confidentiality, integrity, and availability, reflect whether none, some but not all, or all of each of these can be compromised on a system with the vulnerability. This reduces variability due to human factors, towards consistent standardization. The base score equation however, contains numerical values the metric variables can take on depending on their cases, and a number of scalar factors. For example, the Impact and Exploitability sub-scores are multiplied by 0.6 and 0.4, respectively, in computation of the final base-score. The variable and scalar values are apparently arbitrarily assigned to reflect the relative importances of the various metrics and sub-scores in the determination of the overall base score. Human subjectivity may enter the scoring system through the particular choices for these values.

Proposed Work

A better correlation between CVSS base score values and exploitation probability in the wild might be achieved by varying a subset of m of the n scalar parameters in the equation as an m-dimensional optimization problem. The variable to be optimized is some correlation coefficient between the CVSS base score and vulnerability exploitation probability. Possible choices for these parameters are highlighted with yellow in the base score equation, in the appendix. There are n = 26 in v2.10 of the CVSS base score equation. These may be varied largely independently, subject to natural constraints. For example, when an Impact metric indicates complete failure, the variable should take a larger value than when the failure is partial or non-existent. The base scores for each vulnerability under consideration can be computed with each iteration of the base score equation recasting.

Before the recast base scores can be correlated with exploit probability, this probability must be quantified. Bozorgi *et al.* [4] classified vulnerabilities from the Open-Source Vulnerability Database (OSVDB - https://blog.osvdb.org/) based on whether the vulnerability report was classified as having available or rumored/private exploits indicating positive exploit occurrences, or classified as unavailable or unknown exploits indicating negative exploit occurrences. Vulnerabilities with unclassified exploit statuses were not considered. Whereas Bozorgi *et al.* used a support vector machine (SVM) to predict exploitation by learning from existing vulnerability descriptions with their exploit classifications, we may focus on the correlation between base scores and exploitation. The intraclass correlation coefficient is suited to correlating the base score variable with the binary, positive versus negative, vulnerability exploitation classification.

In an earlier work, Younis and Malaiya [5] compared the efficacy of CVSS to Microsoft Security Response Center (https://technet.microsoft.com/en-us/security/ff943560.aspx) scores in predicting real exploitations. They examined retrieved entries from the NVD and Microsoft

Security Bulletin for vulnerabilities in Microsoft's Windows 7 and Internet Explorer. Vulnerabilities were divided into those that were predicted as exploitable by these scoring systems, and those that were not. These predictions were compared to the presence or absence of exploit reports in the Exploit Database (EDB - https://www.exploit-db.com/) to generate both scoring systems' confusion matrices. Sensitivity and precision metrics can be calculated from confusion matrices, and provide indications of how effectively a system is predicting exploitation. We may likewise calculate sensitivity or precision as variables to optimize in our base score equation recasting.

The works done by these authors [1, 3, 4] have demonstrated the limits of applicability of the CVSS base score as a predictor of vulnerability exploitation in the wild. Authors such as Bozorgi *et al.* [4] and Khazaei *et al.* [6] have proposed remedies in the form of methods for computationally predicting exploitation and CVSS base scores, respectively, from attributes extracted from vulnerability descriptions. However, until an exploitation prediction tool, such as the SVMs for exploitation prediction used by Bozorgi *et al.* are commonplace in industry, the utility of a standardized vulnerability scoring system remains. To this end, we propose an investigation of the optimality of the CVSS base score equation parameters. We will investigate optimizing interclass correlation of base scores between vulnerabilities either with, or without, reported exploits. We will also investigate sensitivity and precision of CVSS base score exploitability prediction as variables for optimization.

References

[1] Mell P, Scarfone K, and Romanosky S. "A complete guide to the common vulnerability scoring system version 2.0" Published by FIRST-Forum of Incident Response and Security Teams. 2007.

[2] Allodi L, and Massacci F. "Comparing Vulnerability Severity and Exploits Using Case-Control Studies" ACM Transactions on Information and System Security. 2014; 17(1):1-20.

[3] Younis AA, Malaiya YK, and Ray I. "Evaluating CVSS Base Score Using Vulnerability Rewards Programs" International Federation for Information Processing. 2016, 471:62-75.

[4] Bozorgi M. Lawrence KS, Savage S, and Voelker GM. 2010 "Beyond Heuristics: Learning to Classify Vulnerabilities and Predict Exploits" Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. 2010; 105-114.

[5] Younis AA, and Malaiya YK. "Comparing and Evaluating CVSS Base Metrics and Microsoft Rating System" IEEE International Conference on Software Quality, Reliability and Security. 2015; 252-261.

[6] Khazeai A, Ghasemzadeh M, and Derhami V. "An automatic method for CVSS score predicition using vulnerabilities description" Journal of Intelligent & Fuzzy Systems. 2016; 30:89-96.

Appendix – CVSS base score equation

The base equation is the foundation of CVSS scoring. The base equation (v2.10) is:
BaseScore = round_to_1_decimal(((0.6*Impact)+(0.4*Exploitability)–1.5)*f(Impact))
Impact = 10.41*(1-(1-ConfImpact)*(1-IntegImpact)*(1-AvailImpact))
Exploitability = 20* AccessVector*AccessComplexity*Authentication
f(impact)= 0 if Impact=0, 1.176 otherwise
AccessVector = case AccessVector of
requires local access: 0.395
adjacent network accessible: 0.646
network accessible: 1.0
AccessComplexity = case AccessComplexity of
high: 0.35
medium: 0.61
low: 0.71
Authentication = case Authentication of
requires multiple instances of authentication: 0.45
requires single instance of authentication: 0.56
requires no authentication: 0.704
ConfImpact = case ConfidentialityImpact of
none: 0.0
partial: 0.275
complete: 0.660
IntegImpact = case IntegrityImpact of
none: 0.0
partial: 0.275
complete: 0.660
AvailImpact = case AvailabilityImpact of
none: 0.0
partial: 0.275
complete: 0.660