Carnegie Mellon University
Software Engineering Institute

TOWARDS IMPROVING CVSS

J.M. Spring, E. Hatleback, A. Householder, A. Manion, D. Shick
Contact: cert@cert.org
December 2018

Executive summary

The Common Vulnerability Scoring System (CVSS) is widely misused[1] for vulnerability prioritization and risk assessment, despite being designed to measure technical severity. Furthermore, the CVSS scoring algorithm is not justified, either formally or empirically. Misuse of CVSS as a risk score means you are not likely learning what you thought you were learning from it, while the formula design flaw means that the output is unreliable regardless. Therefore, CVSS is inadequate. We lay out a way towards understanding what a functional vulnerability prioritization system would look like.

Misuse of CVSS appears to be widespread, but should be empirically studied. We are unaware of any systematic accounting of who uses or misuses CVSS. In some cases misuse of CVSS Base scores as direct vulnerability prioritization may be policy, for example in the U.S. government[2] and the global payment card industry.[3] We have observed many other organizations with similarly naïve vulnerability prioritization policies. Although some organizations consider CVSS carefully as one of multiple inputs to prioritization, our hypothesis based on the available evidence is that misuse is widespread.

Lack of justification for the CVSS formula

The CVSS v3.0 formula is not properly justified. This failing is related to traditional levels of scientific measurement: nominal, ordinal, interval, and ratio.[4] What mathematical operations are justified vary per measurement type. An ordinal measurement such as the common Likert scale of [completely agree, agree, disagree, completely disagree] has ordering but no distance between items. Addition, multiplication, and division are not defined between ordinal data items. For example, "fastest + fastest" does not make sense, whether we label "fastest" as "1" or not. Which statistical tests and regression are justified

[1] "Misuse" in this document means against the advice of the CVSS-SIG, which specifies, for example, "CVSS provides a way to capture the principal characteristics of a vulnerability … reflecting its severity ... to help organizations properly assess and prioritize their vulnerability management processes." See https://www.first.org/cvss/.
[2] Suggested for use by federal civilian departments and agencies via NIST guidance (e.g., SP 800-115, p. 7-4 and SP 800-40r3 pg. 4) and the DHS directive on Critical Vulnerability Mitigation (https://cyber.dhs.gov/bod/15-01/).
[3] Via PCI DSS, see: https://www.pcisecuritystandards.org/documents/ASV_Program_Guide_v3.0.pdf
[4] Stevens, S. S. (7 June 1946). "On the Theory of Scales of Measurement". Science. 103 (2684): 677–680. doi:10.1126/science.103.2684.677. Also: https://en.wikipedia.org/wiki/Level_of_measurement

with ordinal measurements is more complex. In general, non-parametric statistics should be used.[5] The CVSS documentation is not transparent about what techniques were used to derive the formula, but the equations appear to result from parametric regression. This is not necessarily unjustified. In some situations, parametric tests seem empirically robust to violations of the relevant assumptions, such as an ANOVA test of difference between two populations' responses on a Likert scale.[6]

CVSS takes ordinal data such as [unavailable, workaround, temporary fix, official fix] and constructs a novel regression formula, via unspecified methods, which assigns relative importance rankings as ratio values. The CVSS v3.0 documentation offers no evidence or argument in favor of the robustness of the given formula or construction method.[7] Since the formula, strictly speaking, commits a data type error, the burden of proof lies with the specification.[8] This complaint is not the same as asking why low attack complexity is 0.77 instead of, perhaps, 0.81; it is also not the same as wondering why the impact subscore involves a 15^{th} power instead of the 14^{th}. The complaint is that we have been given no evidence that the formula is empirically or theoretically justified.

We have a variety of other methodological questions because the CVSS v3.0 specification offers no transparency on the whole formula creation process. The initial ranking of vulnerabilities affects the result, so how this was done matters. Further, while the descriptions for the metrics are clear, how their relative importance was selected is not. Different communities would plausibly find different metrics more or less important (confidentiality versus availability, for example), as we discuss below. These various problems contribute to concern that the CVSS v3.0 formula is not robust or justified.

We suggest that the way to fix this problem is to skip converting qualitative measurements to numbers. CVSS v3.0 vectors could be mapped directly to a decision or response priority. This mapping could be represented as a decision tree[9] or a table. Different communities may want different mappings.

CVSS scores severity, not security risk

CVSS is designed to identify the technical severity of a vulnerability. What people seem to want to know, instead, is the risk a vulnerability or flaw poses to them, or how quickly they should respond to a vulnerability. If so, then either CVSS needs to change or the community needs a new system.

Much of this paper will identify salient things CVSS does not do. We have two goals in this; one is educational, to make sure people are clearly aware of the limited scope of CVSS (Base) scores. The second goal is aspirational, to identify near targets that a software-flaw risk score should, and we believe

---

5 Jamieson S. Likert scales: how to (ab) use them. Medical education. 2004 Dec; 38(12):1217-8.

6 Norman G. Likert scales, levels of measurement and the "laws" of statistics. Advances in health sciences education. 2010 Dec 1; 15(5):625-32.

7 https://www.first.org/cvss/cvss-v30-specification-v1.8.pdf.

8 In particular, page 21 of the CVSS 3.0 specification v1.8 is inadequate here.

9 See, for example, the decision trees resulting from: Burch H, Manion A, Ito Y, Vulnerability Response Decision Assistance (VRDA). Software Engineering Institute, Carnegie Mellon University. Jun 2007. https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=51036

plausibly could, address. We have no evidence that CVSS accounts for any of the following: the type of data an information system (typically) processes, the proper operation of the system, the context in which the vulnerable software is used, or the material consequences of an attack. Furthermore, vulnerabilities are not the only kind of flaw that can lead to security incidents on information systems. Organizations should make risk-based decisions about security, but CVSS does not provide that whole measure, even accounting for the Temporal and Environmental scores that aim to account for context and consequences.

We extracted the following overarching criticisms of CVSS from reports by the community since 2007. We do not detail the problems here, but rather provide a referenced discussion organized as follows:

- Failure to account for context (both technical and human-organizational)
- Failure to account for material consequences of vulnerability (whether life or property is threatened)
- Operational scoring problems (inconsistent or clumped scores, algorithm design quibbles)

Failure to account for context

There are various ways in which CVSS does not account, broadly, for context. These include vulnerabilities in shared libraries, related or chained vulnerabilities, web vulnerabilities in general, and different interpretations of a CVSS score in different communities.

A program using a shared library is to some extent affected by that library's vulnerabilities. However, as examples, whether the program sanitizes inputs and what tasks or function calls it performs can change vulnerability severity. CVSS's one-size-fits-all approach does not capture this complexity.[10]

CVSS does not handle the relationship(s) between vulnerabilities. One important reason independent scoring can be misleading is that vulnerabilities can be chained together, leveraging one to establish the preconditions for another. CVSS v3.0 adds a vector addressing the scope of impact and provides some guidance[11] about vulnerability chaining. These concepts help, but are not sufficient.[12]

Context is complicated for vulnerabilities in web browsers and plugins such as PDF viewers. The question is user identity and privileges. Figure 1 summarizes the issue. The question is whether a web-browser vulnerability is considered as a local user-space violation, or as an admin-space violation of banking and email credentials. The CVSS v3.0 Scope metric partially addresses these complexities.

---

[10] https://www.oracle.com/technetwork/topics/security/cvssscoringsystem-091884.html. The CVSS SIG has noted this as a work-item for version 3, see https://www.first.org/cvss/workitems.

[11] https://www.first.org/cvss/user-guide#3-7-Vulnerability-Chaining

[12] https://www.riskbasedsecurity.com/2017/02/cvssv3-new-system-next-problem-scope/

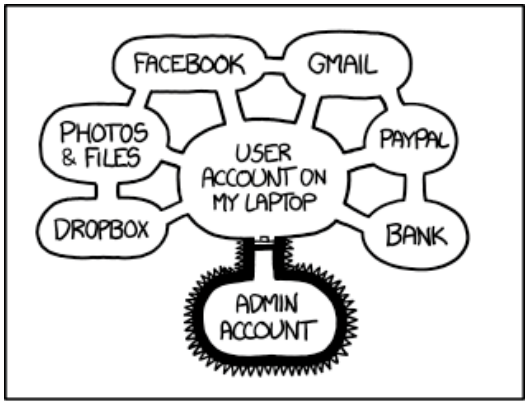

Figure 1: "Authorization" CC-BY-NC Randall Munroe, https:/xkcd.com/1200

There are distinct use cases of CVSS for varying needs. This includes vulnerability management use cases such as coordinating, constructing, or deploying updates. Additionally, different organizations in different sectors generally deploy information systems for different purposes. A scoring system should be able to accommodate each appropriately without degenerating to a single score to be applied in all contexts. CVSS makes it too easy to boil down complex data points. Furthermore, the CVSS documentation provides no guidance on how communities should interpret scores, such as how to map actions or responses to either numbers or severity categories (high, critical, etc.).

This problem is pronounced in use cases where data loss is more critical than the loss of control of the device—such as financial system compliance or privacy compliance with the General Data Protection Regulation. Furthermore, the problem also arises in cases where data integrity or availability are more critical, such as safety-critical embedded devices used in health care, transportation, and industrial control systems.[13] In some cases, this challenge has been recognized; for example, the Food and Drug Administration (FDA) is working with MITRE to adapt CVSS to a medical device context.[14] CVSS was designed for how vulnerabilities affect more traditional IT systems. Design assumptions for traditional IT vulnerabilities do not necessarily fit these other domains.[15]

Failure to account for threat and consequences of vulnerability

Criticism in this vein claims that severity scoring should consider material consequences, threat likelihood, and security issues that are not strictly defined as "vulnerabilities." Important examples include embedded devices such as cars, utility grids, or weapons systems. Intuitively, if a vulnerability will plausibly harm humans or physical property, then it is severe, but CVSS as it is does not account for this. However, the fix is not obvious. There are some recommendations from safety evaluation standards, but they are not suited to security or vulnerabilities (e.g., SIL IEC-61508).[16]

In general, severity should only be a part of vulnerability response prioritization.[17] One might also consider exploit likelihood or whether exploits are publicly available. The Exploit Code Maturity vector (in

---

[13] https://www.securityweek.com/cvss-scores-often-misleading-ics-vulnerabilities-experts
[14] https://www.fda.gov/downloads/MedicalDevices/NewsEvents/WorkshopsConferences/UCM560511.pdf
[15] https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=505311
[16] https://insights.sei.cmu.edu/cert/2015/09/cvss-and-the-internet-of-things.html
[17] For example: Farris KA, Shah A, Cybenko G, Ganesan R, Jajodia S. VULCON: A System for Vulnerability Prioritization, Mitigation, and Management. ACM Transactions on Privacy and Security (TOPS). 2018 Jun 12; 21(4):16.

Temporal Metrics) attempts to address exploit likelihood, but the default value assumes widespread exploitation, which is not realistic.[18] Separately, vendor patch development or owner patch deployment might be assisted if a high CVSS score were predictive of which vulnerabilities would be commonly exploited or have exploits publicly released. However, this correlation is not the case.[19] There is good evidence to suggest attackers are work-averse and only develop exploits when economically advantageous—which means every six to 36 months against each widely deployed software system.[20] The community of adversaries also de facto develops and democratizes various attack capabilities;[21] thus a flaw in a system with widespread adversary capability against it should be more important or pressing.

Threats do not only target vulnerabilities in information systems. There are also misconfigurations, dangerous default settings, abuse of available features, and surprising interactions between systems otherwise operating as designed. CMSS (Common Misuse Scoring System)[22] and CCSS (Common Configuration Scoring System)[23] are derived from CVSS and aim to capture misuse of available features that lead to security incidents. CMSS is far from accounting for all relevant security-related flaws in information systems. Further, it's not clear how to compare CMSS, CCSS, and CVSS scores. Both CMSS and CCSS suffer all the same mathematical errors as CVSS, so a numerical comparison is not just misleading, but undefined.

Operational problems with scoring

There are substantive problems in assigning scores, with various kinds of inconsistency (variability and inflation), vague guidelines, and technical details of scores. Based on available evidence, CVSS v3.0 concepts are understood with wide variability, even by experienced security professionals. The most accurate 50% of security professionals surveyed mis-score vulnerabilities in a range of 2–4 points.[24] Note four points is wider than the whole recommended "high" range. More than half of survey respondents are not consistently within an accuracy range of four CVSS points.

There are other sorts of alleged inconsistency with CVSS scoring. For example, seemingly similar vulnerabilities are assigned different scores by the NIST National Vulnerability Database (NVD), or security vendors attempting to follow NVD guidance.[25] In practice only six of the CVSS v2.0 scores account

---

[18] See version 4 work items. https://www.first.org/cvss/workitems

[19] Allodi, Luca and Fabio Massacci. A Preliminary Analysis of Vulnerability Scores for Attacks in Wild: The EKITS and SYM Datasets. BADGERS'12, Oct 5, 2012, Raleigh, North Carolina, USA.

[20] Allodi L, Massacci F, Williams J. The work-averse cyber attacker model: Theory and evidence from two million attack signatures. WEIS. Jun 26, 2017, San Diego, CA.

[21] Spring J, Kern S, Summers A. Global adversarial capability modeling. APWG Electronic Crime Research Symposium (eCrime). May 26, 2015. IEEE.

[22] https://www.nist.gov/publications/common-misuse-scoring-system-cmss-metrics-software-feature-misuse-vulnerabilities

[23] https://csrc.nist.gov/publications/detail/nistir/7502/final

[24] Allodi L, Cremonini M, Massacci F, Shim W. The Effect of Security Education and Expertise on Security Assessments: the Case of Software Vulnerabilities. In: WEIS 2018. See figure 1. The more accurate half of professionals estimated CVSS scores in ranges such as [+2,0] (that is, between overestimating by 2 to being correct), [+2,-2], and [0,-2].

[25] Which is different from FIRST guidance. See: https://nvlpubs.nist.gov/nistpubs/ir/2014/NIST.IR.7946.pdf

for 67% of the values assigned.[26] Furthermore, severity scores inflate over time, unrelated to community valuation of severity, both between versions[27] 1 and 2 and versions[28] 2 and 3. The poorly justified formula construction methods, discussed above, may contribute to these scoring irregularities.[29]

Vague descriptions are filled in with "assume the worst". Social engineering attacks are disproportionately influenced by this choice; if we assume the worst, we assume the user executes the code when asked, which is not quite right.[30] Furthermore, Environmental scores tend to reduce the CVSS score,[31] but they do not appear to be applied correctly (if at all) by most consumers.

Some writers have expressed concern with technical details of version 3.[32] Most of the work items for the CVSS SIG that relate to CVSS v3.0 are essentially minor technical corrections as well.[33] These considerations are generally helpful and constructive, but do not address the fundamental challenges.

A Way Forward

These challenges suggest that at least a few things need to be done. The CVSS-SIG (Special Interest Group), which standardizes CVSS as part of FIRST (Forum of Incident Response and Security Teams), has not focused on addressing these core issues. The CVSS-SIG and the community should address the flaws in CVSS, not their symptoms. The scoring formula needs to be redone, essentially from scratch, this time transparently and with adequate empirical justification. Since this level of reworking is needed, we suggest that any new algorithm aims to address the various risk elements of context and material consequences we identify above.

This task is large, but we have a few suggestions towards the goal. First, adequate user studies should be conducted to understand how organizations use CVSS in their risk assessments today. Here, *adequate* does not mean an email survey. We suggest semi-structured interviews conducted by a trained sociologist. The snowball sampling method is likely the best realistic option.

Secondly, an empirical study of the consistency of human scoring using CVSS is needed. Different people should be able to reliably create the same vector given the same information about a software flaw. While varied anecdotal evidence suggest that scoring is operationally not reliable in this way, it

---

[26] Scarfone, Karen, and Peter Mell. An Analysis of CVSS Version 2 Vulnerability Scoring. In: Empirical Software Engineering and Measurement. 2009. https://ws680.nist.gov/publication/get_pdf.cfm?pub_id=903020

[27] Jones, Jeffrey. CVSS severity analysis. Jun 2008. https://www.first.org/cvss/jones-jeff-slides.pdf

[28] http://blogs.cisco.com/security/cvssv3-study

[29] https://www.hstoday.us/subject-matter-areas/cybersecurity/hstrisk-vulnerability-mad-hatters-in-accurately-scoring-cybersecurity/

[30] https://www.riskbasedsecurity.com/reports/CVSS-ShortcomingsFaultsandFailures.pdf

[31] Frühwirth, Christian and Tomi Männistö. Improving CVSS-based vulnerability prioritization and response with context information. In: Empirical Software Engineering and Measurement. 2009.

[32] Risk Based Security published a series of blog posts describing their concerns in detail, starting with https://www.riskbasedsecurity.com/2017/01/cvssv3-newer-is-better-right/ and ending with https://www.riskbasedsecurity.com/2017/06/cvss-is-3-the-magic-number/

[33] https://www.first.org/cvss/workitems

would be preferable to understand both the extent of mismatch in practice across and within databases such as the NVD. A human lab study where participants score different vulnerabilities in a controlled environment would provide a complementary description of the problem and provide the potential for teasing out explanations. These studies should be done before devising a new scoring formula to help inform its construction, and then repeated on any new formula to help validate its robustness.

Towards this end, we suggest a robust scoring system should have the following features. Each feature of interest needs a clear rubric for valuation. Three scorers following the rubric independently should have high agreement (usually all three agree); the rubric should be empirically and transparently tested and refined until this is the case. Ideally, two-scorer agreement should be used operationally for reliability of published "scores," where scores should be the action priority category (for example, ignore, medium, high, critical) and never integers. The scores should be assigned from a vector via a transparent decision process or lookup table, not arithmetic. A different mapping should be offered for each use case identified within the community, to better account for context. The definition of the mappings should be informed by the sociological study, capturing different community's needs, context, and risks.

We expect existing risk assessment methods can be carefully applied. Threat can be roughly defined as likelihood of exploitation; however, the real item of interest is estimating expected loss. Such estimates are difficult, but can be improved by accounting for the context in which the vulnerable system is used.

Conclusion

Given problems outlined here, CVSS needs to change, or we need a new system. There are significant usability issues and formal challenges. From a usability perspective, either it must be made clear that CVSS reflects severity, not risk, or CVSS must be adjusted to make it reflect risk. From a formal perspective, once this intended usage is clarified, the challenge remains to design a scoring algorithm that is actually reliable and transparently justified. Addressing these challenges does not appear to be a priority for the CVSS-SIG, based on its current list of work items.

CERT Vulnerability Notes currently include full CVSS v2.0 scores,[34] and likely will until there is a viable alternative. The sketch of what CVSS could be, with community effort, is outlined by our way forward, above. The community needs to understand what the various interest groups want from CVSS, account for those human contexts, account for the various relevant technical contexts, and then transparently design a scoring algorithm that satisfactorily informs the decisions of those groups—dare we say, informs these decisions "scientifically".[35]

[34] https://www.kb.cert.org/vuls/help/fieldhelp/#cvss_metrics

[35] By a *science of security* we would mean "the label we should apply to the most solidly grounded bodies of knowledge about measures one can take to protect a system." See: J.M. Spring, T. Moore, and D. Pym. Practicing a Science of Security. In: Proc. 2017 New Security Paradigms Workshop, Santa Cruz, CA, USA, October 1–4, 2017.

Acknowledgements

We thank the members of the CVSS-SIG for their detailed and sincere comments on an earlier draft of this paper. The views expressed here do not necessarily represent those of the CVSS-SIG.

Contact Us

Software Engineering Institute
4500 Fifth Avenue, Pittsburgh, PA 15213-2612

Phone: 412/268.5800 | 888.201.4479
Web: www.sei.cmu.edu
Email: info@sei.cmu.edu

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.