Common Vulnerability Scoring System version 4.0

Frequently Asked Questions (FAQ)

Document Version: 1.3

The Common Vulnerability Scoring System (CVSS) is an open framework for communicating the characteristics and severity of software vulnerabilities. CVSS consists of four metric groups: Base, Threat, Environmental, and Supplemental. The Base group represents the intrinsic qualities of a vulnerability that are constant over time and across user environments, the Threat group reflects the characteristics of a vulnerability that change over time, and the Environmental group represents the characteristics of a vulnerability that are unique to a user's environment. Base metric values are combined with default values that assume the highest severity for Threat and Environmental metrics to produce a score ranging from 0 to 10. To further refine a resulting severity score, Threat and Environmental metrics can then be amended based on applicable threat intelligence and environmental considerations. Supplemental metrics do not modify the final score, and are used as additional insight into the characteristics of a vulnerability. A CVSS vector string consists of a compressed textual representation of the values used to derive the score.

This document provides answers to frequently asked questions regarding CVSS version 4.0.

The most current CVSS resources can be found at https://www.first.org/cvss/

1. Introduction

During the development of version 4.0 of the Common Vulnerability Scoring System (CVSS), many excellent questions and comments were provided during the public comment period.

As a supplement to the CVSS version 4.0 Specification Document, User Guide, and Examples Document, this set of frequently asked questions (FAQs) and answers has been collected in order to aid in the adoption and understanding of the new standard.

2. Resources & Links

Below are useful references to additional CVSS v4.0 documents.

Resource	Location
Specification Document	Includes metric descriptions, formulas, and vector strings. Available at https://www.first.org/cvss/v4.0/specification-document
User Guide	Includes further discussion of CVSS v4.0, a scoring rubric, and a glossary. Available at https://www.first.org/cvss/v4.0/user-guide
Examples Document	Includes examples of CVSS v4.0 scoring in practice. Available at https://www.first.org/cvss/v4.0/examples
CVSS v4.0 Calculator	Reference implementation of the CVSS v4.0 equations, available at https://www.first.org/cvss/calculator/v4.0
JSON & XML Data Representations	Schema definition available at https://www.first.org/cvss/data-representations
CVSS v4.0 Main Page	Main page for all other CVSS resources: https://www.first.org/cvss

3. Frequently Asked Questions

3.1. Is there a prescribed way to use CVSS Base and Environmental metrics to score a vulnerability along a long supply chain?

Answer - No.

3.2. What's the difference between Attack Complexity (AC) and Attack Requirements (AT)?

The concept of Attack Complexity (AC) in CVSS v3.1 has now been expanded into two base metrics: Attack Complexity (AC) and ATtack Requirements (AT). The differences between Attack Complexity and Attack Requirements can best be understood as:

- Attack Complexity: BUILT-IN CHALLENGES: Actions must be taken by the attacker to actively evade or circumvent built-in security-enhancing conditions. Examples include: ASLR and DEP.
  Alternatively, the attacker must gather some target-specific secret before the attack can be successful. A secret is any piece of information that cannot be obtained through any amount of reconnaissance. To obtain the secret the attacker must perform additional attacks or break otherwise secure measures (e.g. knowledge of a secret key may be needed to break a crypto channel).
  Note: When performing a site-specific assessment, presence of a built-in firewall filter as a compensating control can be represented via the Modified Attack Complexity (MAC) Environmental metric.

- Attack Requirements: EXTERNAL CHALLENGES: Deployment and execution conditions or variables of the vulnerable system must be overcome. Examples include: race conditions or on-path network injection (also known as Machine in the Middle or MITM).
  Note: When performing a site-specific assessment, presence of an external firewall filter as a compensating control can be represented via the Modified Attack Requirements (MAT) Environmental metric.

3.2.1. Why did you use "AT" for Attack Requirements, instead of "AR"?

AR was already being used for Availability Requirements. The Specification will not allow two metrics to have the same abbreviation.

3.3. What are the boundaries between a Vulnerable System and Subsequent System?

For the purpose of determining Vulnerable System versus Subsequent System impacts, keep in mind these factors:

1. When thinking about the Vulnerable System, consider the scope of where the vulnerability exists and from which it will be exploited; the factors for AV, AC, AT, PR, and UI will be determined based on this. If, for example, a vulnerability exists in a specific service or module which is part of a larger unit, and is not able to stand on its own, then the scope of the Vulnerable System would almost certainly be the containing unit.
   A concrete example: an application delivered by a vendor contains multiple services, one of which contains a vulnerability; if the specific service is not independent of — or to put another way, is integral to — the larger collection of services, the vulnerability would be reported against the application, and the Vulnerable System would likewise be the application.

2. Subsequent Systems encompass anything that is not the Vulnerable System and is impacted in some way by the vulnerability in the Vulnerable System.
   Using the application/service example above, if the service can be separated from the rest of the components (e.g., the vendor may supply both, but one is not integral to the other), the scope of the vulnerability would be reported as within the service consumed by the application. In this example, the Vulnerable System would be the service rather than the application. If the application's security is impacted from the vulnerability in the service, the application would be among the Subsequent Systems affected by the vulnerability.

   Be sure to check out the Examples Document for variations of how Vulnerable System and Subsequent Systems are identified and assessed.

3.4. Supplemental Metrics do not affect the final score. What is the recommended way to consume and respond to Supplemental Metric assessment?

Consumers are encouraged to consider more than just the simple numerical score and understand the full context and nuance of a vulnerability's capabilities and impact. Also, there is important and valuable information that can be provided to the consumer that does not impact the severity of the vulnerability. For example, a denial of service (DoS) vulnerability that immediately recovers (R:A) has the same numeric score as a DoS vulnerability that leaves the system unavailable until an administrator restarts the system (R:U). However, the length of service impact may differ greatly and can be considered by the consumer when planning remediation resources.

Another example is a Border Gateway Protocol (BGP) networking vulnerability that forwards the malicious update and then crashes (AU:Y), causing a cascade outage vs. the same vulnerability that crashes without forwarding (AU:N). The technical severity is the same, but the scope of service impact is far greater for an automatable vulnerability.

Think of Supplemental Metrics as an extended vocabulary for conveying information about the extrinsic characteristics, or outward impact, of a vulnerability beyond just the technical severity. If we consider the CVSS v4.0 vector as a "vocabulary" of vulnerability assessment, the more information a supplier can provide the consumer, the more decisions can be made with complete situational awareness. Much like the expansion of the Impact Metrics, allowing suppliers to provide full disclosure on the impact of a vulnerability, Supplemental Metrics provide additional extrinsic characteristics of the vulnerability (e.g., wormable, difficulty of mitigation, objective vendor urgency, etc.) that can be used when deciding which High or Critical severity vulnerability to patch first.

It's also important to distinguish the difference between the assessment (assignment) of the Supplemental Metrics by the ***scoring provider***, versus the assessment (response plan) of the ***consumer***. Product suppliers assess the additional characteristics of the vulnerability and assign applicable Supplemental Metric values. Product consumers review the provided Supplemental Metric values and assess whether any modifications of their vulnerability response (e.g., mitigation priority) is warranted.

3.4.1. What is the guidance on what action to take for the various values of the Safety (S) Supplemental Metric: Negligible (N) and Present (P)?

When a Safety metric value of Negligible has been assessed, it means that whomever did the assessment believes that there should be no concerns about potential operational safety impact unless the vulnerable device has an obvious potential impact to human safety based on how it is being used in the consumer's environment. A Safety metric value of Present, it means that whomever did the assessment believes that there is at least some potential for an operational safety impact in most, if not all instances of its deployment. The context of the system and the consumer's concerns with respect to the use of the system may suggest very different guidance for different situations.

3.4.1.1. Additionally, what is meant by "consequences of the vulnerability meet definition of IEC 61508 consequence categories of 'marginal,' 'critical,' or 'catastrophic.'"?

The consequences of the vulnerability refer to potential Safety impacts. IEC 61508 provides some handy references for understanding potential safety categories. A snapshot of a table from Wikipedia[1] on this subject is shown below:

[1] IEC 61508 - Wikipedia

Consequence Categories

Category	Definition
Catastrophic	Multiple loss of life
Critical	Loss of a single life
Marginal	Major injuries to one or more persons
Negligible	Minor injuries at worst

In the above table, you can see the same consequence categories that have been introduced in the description of Safety metrics. This version of CVSS is focusing on considering a safety impact that either doesn't (Negligible) or does (anything more severe than Negligible) create a potential safety concern.

3.4.2. What is the guidance on what action to take for the various values of the Automatable (AU) Supplemental Metric: No (N) and Yes (Y)?

If exploitation of a vulnerability is automatable by an attacker, you should consider a higher priority for mitigating or fixing the vulnerability.

An answer of Yes to Automatable is aligned with the vulnerability being wormable,[2] that is, the vulnerability can be used by a self-replicating attack that can spread at machine speed. Because this reduces an defender's available time between receiving reports of active exploitation and attacks against their own systems, consider mitigating or remediating any Automatable vulnerabilities as soon as there is a public proof of concept exploit.

3.4.3. What is the guidance on what action to take for the various values of the Recovery (R) Supplemental Metric: Automatic (A), User (U), and Irrecoverable (I)?

The spirit and intent of adding Recovery a Supplemental metric was for consumers to consider the post attack scenario and assess the actual impact.

Recovery metric describes the method of recovery of a component or service once an attack has been carried out.

Automatic Recovery (A) : This metric describes that the system or component can recover by itself within no time. Eg: A line card in a networking device could reboot and restart on its own , without causing any service intervention. This is the same as a

[2] A "worm" is a "A computer program that can run independently, can propagate a complete working version of itself onto other hosts on a network, and may consume system resources destructively." (RFC 4949).

daemon restarting upon an attack, automatically. In these case, the downtime is almost negligible.

User Recovery (U) : This metric describes that the system or component cannot recover on its own, but requires a human intervention. Eg: A manual restart due to persistent memory leak leading to a Denial of Service on the system.

Irrecoverable (I) : This metric describes that the system or component is beyond repair and cannot recover on its own. Eg: CPU releases smoke due to colling fan malfunctioning or any hardware failure that needs a replacement.

Note that User (U) represents actions possible by an operator, consumer, or end customer. The recovery of a Component/System includes service restoration in terms of availability and performance. Auto recovery is limited in scope to the component itself.

3.4.4. What is the guidance on what action to take for the various values of the Value Density (V) Supplemental Metric: Diffuse (D) and Concentrated (C)?

Value Density is primarily a property of the system which contains the vulnerability, rather than the vulnerability itself. Therefore, Value Density is less likely to be useful for a software vendor or maintainer who is prioritizing work within one software product. Anyone that is prioritizing vulnerability actions among work that includes multiple products (such as a manager at a large supplier deciding among projects, a coordinator, or any software consumer or system owner) should consider mitigations and remediations to systems with Concentrated Value Density as a higher priority than those with Diffuse Value Density. Concentrated Value Density systems are a higher priority to mitigate or remediate because they are a higher priority for adversaries to attack.

3.4.5. What is the guidance on what action to take for the various values of the Vulnerability Response Effort (RE) Supplemental Metric: Low (L), Moderate (M), and High (H)?

The intent of the Vulnerability Response Effort (RE) Supplemental Metric was for consumers to use that information to plan out their mitigation deployments and the resources that would be needed.

- Low (L) – non intrusive change simple to deploy. Low impact to systems like updating documentation.
- Medium (M) – Software drivers and other items that can be mitigated quickly.
- High (H) – More complicated typically low level software or firmware where platform reboots would be required (Especially in a Data Center environment.)

3.4.6. What is the guidance on what action to take for the various values of the Provider Urgency (U) Supplemental Metric: Clear, Green, Amber, and Red?

It may be helpful to think of the Provider Urgency as the "low order bits" of the severity rating. For example, given two unauthenticated network-based denial of service vulnerabilities (CVSS-B 8.7), the vulnerability identified by the supplier (eg. vendor) with a higher Provider Urgency may be worth considering being remediated first.

Cases where a supplier may set a higher than neutral Provider Urgency include vulnerabilities that don't qualitatively reflect the criticality of the issue, given its CVSS-B score. Conversely, cases where a provider may set a lower Provider Urgency include vulnerabilities that result in a subjectively higher CVSS-B score than the product provider would have assessed, for example, in their public security advisory.

Note that suppliers may assess a Provider Urgency that *aligns* with the Qualitative Severity Rating Scale[3], neither raising nor lowering the base remediation priority.

While any provider along the product supply chain may provide a Supplemental Urgency rating:

Library Maintainer → OS/Distro Maintainer → Provider 1 ... **Provider n (PPP)** → Consumer

the Penultimate Product Provider (PPP) is best positioned to provide the most direct assessment of Urgency.

3.4.6.1. Additionally, why were TLP colors chosen, rather than simply Low, Moderate, and High?

The intent of the supplemental assessment Provider Urgency was to provide *additional insight* into the vulnerability's impact. Care was taken to limit this metric to specifically augmenting the assessment, rather than overriding the CVSS-B score. Had linear values such as Low, Moderate, and High been selected for Provider Urgency, consumers may have misinterpreted it as an overruled assessment of the severity, invalidating the methodically calculated CVSS-B score.Answer

3.5. How did CVSS v4.0 address the issue of scores being too clustered toward High and Critical severity ratings?

The perception that CVSS v3.1 scores were clustered toward the Critical and High ratings was not a problem the CVSS SIG was intending to solve in v4.0. It is natural for suppliers to focus their efforts to assess, announce, and provide fixes for the most severe vulnerabilities they identify in their products. As a result, more vulnerabilities are rated higher on the severity scale than others.

[3] CVSS v4.0 Specification Document, Section 6

It is the responsibility of the consumer to apply Threat and Environmental data into the assessment of the vulnerabilities (preferably using automation) to reduce the scores of the vulnerabilities that are not as important as others. For example, vulnerabilities that are not being exploited in the wild nor have proof-of-concept code publicly available will realize a significant reduction in the CVSS-BTE score. Exploited vulnerabilities will maintain their higher scores allowing the consumer to focus on what is important.

3.6. How does the value of Attack Complexity (AC) or Attack Requirements (AT) vary with the introduction of compensating controls (e.g., WAF, CSP, etc.)?

This is related to the application of Environmental Metrics into the vulnerability assessment. Please refer to "How does a consumer apply Environmental data into a CVSS Assessment?" below for more details.

3.7. Which value of User Interaction (UI), Passive (P) or Active (A), should be used for Reflected Cross-Site Scripting (XSS) or Stored Cross-Site Scripting vulnerabilities?

Reflected cross-site scripting requires that an individual click a specific link to trigger the exploit. That individual has the choice or opportunity to review the link prior to interacting with it. A conscious decision is made to interact with the payload, so this would be considered Active.

Stored cross-site scripting does not require a specially crafted link to trigger the exploit. An individual browsing a website could inadvertently trigger the exploit through standard browsing behaviors and would have no awareness or ability to avoid the payload prior to exploitation. Because there isn't a conscious decision to interact with the payload, this would be considered Passive.

3.7.1. Is clicking a link Passive (P) or Active (A) User Interaction?

Prior to clicking a link, an individual has the choice or opportunity to review the link. Because the individual is making a conscious decision to interact with the payload that is delivered via a link, this would be considered Active.

3.8. Which system is considered the Vulnerable System or the Subsequent System when assessing Reflected or Stored Cross-Site Scripting (XSS) vulnerabilities?

In a cross-site scripting vulnerability, the web application that contains the injection point is the Vulnerable System. A user's browser is the Subsequent System.

3.9. Which system is considered the Vulnerable System or the Subsequent System when assessing SQL Injection vulnerabilities?

Different scenarios may exist depending on the configuration of the applications in use. Our example CVE-2023-30545 is simple, where the web application that contains the injection point, the SQL application, and the direct impact of exploitation are all contained on a single system without subsequent system impact.

Other common scenarios include distributed system configurations in which a SQL database application that contains a vulnerability may be the Vulnerable System, and other business applications that rely on data within that database may be impacted by exploitation of the vulnerability, and would be assessed as a Subsequent System.

3.10. *"It is the responsibility of the consumer to populate the Threat Metric Exploit Maturity (E) based on information regarding the availability of exploitation code/processes and the state of exploitation techniques."*

Why is this the responsibility of the consumer and not the provider (vendor)?

This is a common question that is not unique to CVSS v4.0 but is applicable to all past versions.

If this responsibility were to be placed with the supplier/vendor, it would require ALL suppliers/vendors to have an accurate and real-time understanding of the Exploit Maturity for every vulnerability that organization has ever announced – ever. Obviously, this is not realistic and not reliable. However, this is exactly what threat intelligence sources and organizations do best. There are many free and subscription-based threat intelligence feeds available to the consumer that will allow them to apply that data to the maximum number of detected vulnerabilities in their environment.

3.11. Why does the application of Threat data only reduce the resulting CVSS Score?

This is a common question that is not unique to CVSS v4.0 but is applicable to all past versions. At first glance, this concept appears to be counter-intuitive. But there are several reasons for it that promote maturity in a Vulnerability Management (VM) program.

For a moment, consider the most dangerous vulnerabilities. If they had a base score of "10", there's no room to escalate the severity with threat intelligence or environmental criticality data. On the other hand, if they maxed out with a lower score (like "8" to

leave room for escalation based on threat), there are even more negative side-effects starting with the fact that there would be no such thing as a Critical vulnerability without the application of threat.

When vendors score their vulnerabilities for publication, they are assuming the "reasonable worst-case scenario" as per the specification. This means while performing the assessments on newly announced vulnerabilities, providers must assume that all vulnerabilities will be exploited and fully weaponized by malicious actors. It is the responsibility of the consumer to apply available threat intelligence and determine where each of their detected vulnerabilities falls in the Exploit Maturity scale. For those vulnerabilities that ARE exploited and weaponized, their scores will remain high. For those that are not exploited and don't have Proof-of-Concept (POC) code publicly available, those scores will fall and become less important.

3.12. How does a consumer apply threat intelligence data into a CVSS Assessment?

This is a common question that is not unique to CVSS v4.0 but is applicable to all past versions.

For Threat, the key is collecting and applying threat intelligence data to your vulnerability scan data using automation.

Reliable threat intelligence sources are much easier to find now than they were 10 years ago. There are dozens of inexpensive (or free) places to go to get this information in bulk or and even more if you are able to pay for a subscription service. As you would expect, not all threat intel sources are equal – and none are perfect. It is recommended to use MANY different sources of threat intelligence and combine the results before applying it to your scan data.

When you have your vulnerability threat intelligence, it should tell you where each CVE is on the threat scale. If your threat intelligence sources are not using the CVSS Exploit Code Maturity values, you may have to translate that data to be usable. When you have machine-readable threat intelligence, reconcile that data with the CVEs identified in your vulnerability scan data.

Also, it is important to remember that intelligence changes all the time. Therefore, threat intelligence data must be re-collected and re-applied to your scan data frequently (daily updates are recommended). When these jobs are automated, this maintenance becomes much easier.

3.13. How does a consumer apply Environmental data into a CVSS Assessment?

This is a common question that is not unique to CVSS v4.0 but is applicable to all past versions.

The Environmental Metric can be divided into 2 groups

1. “Security Requirements” which includes Confidentiality Requirements (CR), Integrity Requirements (IR), and Availability Requirements (AR) and represent the criticality of the vulnerable device.
2. “Modified Base Metrics” which, as the name would suggest, allows the consumer to adjust the values for any of the CVSS Base Metrics as appropriate for their environment.

It is important to understand that the Environmental Metrics are all attributes of the VULNERABLE SYSTEM. They have NO relation to the vulnerability itself.

It is highly recommended that consumers of CVSS perform an assessment of the devices on their network and document the results in a database (such as an Asset Management Database).

Security Requirements: Using the “Security Requirements” section of the User Guide as a guideline, the assessors will document CR, IR, and AR values in the database.

Modified Base Metrics: Additionally, other attributes might be gathered during this assessment that can be applied to the Modified Base Metrics. This section can be more unclear since the possibilities are nearly endless. Specific configurations and security controls in place in each consumer’s environment will be different. However, here are some examples of how this can be used.

Example 1: If some systems in a consumer’s environment are isolated from the Internet (i.e.: no inbound or outbound connections), a maximum Attack Vector (AV) value of “Adjacent” can be applied to all vulnerabilities identified on those systems.

Example 2: If a device has obvious potential to impact human safety (such as a bluetooth insulin pump or application that organizes first responders), sufficient impact to Integrity or Availability could result in the application of “Safety” as a subsequent impact.

Example 3: A web application is protected by a Web Application Firewall. Attackers attempting to exploit those vulnerabilities will find those attempts much more difficult. Therefore, an increase in the Attack Complexity (AC) would be appropriate.

Once the assessment is complete and documented, the consumer should reconcile the database with the vulnerability scan data and apply these attributes to all vulnerabilities found on each system.

3.14. Is EPSS or SSVC a replacement for CVSS?

Additional scoring systems have been recently introduced and adopted to handle complementary aspects of vulnerability assessment and patch priority. These are

welcome additions to the vulnerability scoring toolbox, providing innovative exploit prediction and decision support.

- EPSS: Exploit Prediction Scoring System
  A data-driven effort for estimating the likelihood (probability) that a software vulnerability will be exploited in the wild within 30 days.
  Reference: https://first.org/epss

- SSVC: Stakeholder-Specific Vulnerability Categorization
  A decision tree system for prioritizing actions during vulnerability management.
  References: https://cisa.gov/ssvc and https://github.com/CERTCC/SSVC

None of these scoring systems is a replacement for another, but can be used in concert to better assess, predict, and make informed decisions on vulnerability response priority.

3.15. How should the Attack Requirements metric be used with regards to system configuration?

Attack Requirements should not be used to denote uncommon or unique configuration requirements that put a system into a vulnerable state. “Execution conditions or variables of the vulnerable system” in the specification document refer only to challenges faced by an attacker in achieving successful exploitation, and not the possible configuration values that render the system vulnerable. When uncommon or unique configuration is required to put the system into a vulnerable state, producers can include those details in the vulnerability description.

The assessor should assume vulnerable configuration, as documented in CVSS v4.0 Specification Document 2.1 Exploitability Metrics.

Specific configurations should not impact any attribute contributing to the CVSS Base metric assessment , i.e., if a specific configuration is required for an attack to succeed, the vulnerable system should be assessed assuming it is in that configuration.

3.16. How is the User Interaction metric used in relation to CSRF vulnerabilities?

The CVSS Specification Document includes a mention of CSRF in the User Interaction metric Passive value. This exploit scenario would include malicious content embedded into either the vulnerable application itself, or another trusted site the targeted user regularly accesses.

However, in practice, attackers exploit many CSRF vulnerabilities through the use of malicious links to another site containing embedded malicious content designed to trigger the CSRF vulnerability.

In both cases, an attacker must trick the victim into viewing a website that contains malicious content. Depending on the most likely scenario, the analyst may choose between selecting User Interaction Passive or Active.

3.17. Why are CVSS v4.0 scores different from version v3.1 scores?

The new metric scoring system in CVSS version 4.0 is a departure from the algebra formula in CVSS version 3.0 and 3.1. Things might look a lot different when adopting CVSS v4.0. A number of questions have been asked to the CVSS SIG about these new scores, and this FAQ will help to supply some of the reasoning behind the new math.

One important note is that while the CVSS numeric score is a useful shorthand for vulnerability severity, the score itself does not describe the important context that can be conveyed as part of the entire vector string.

A) Scores for the same vulnerability are different between v3.1 and v4.0

The method for determining the numeric score is new in CVSS v4.0. The number results from expert ranking done by the CVSS SIG members. This is unique in relation to the algebraic formula of weighting metrics in CVSS v3.0 and v3.1. Naturally, some scores will be different between the two versions of the standard. We think it's an improvement.

You can read more about the ranking methodology in the CVSS User Guide.

B) Why do some unique CVSS v4.0 metrics result in the same numeric score?

The potential range of unique CVSS metrics is large. There are over fifteen million combinations of CVSS vectors that result in a numeric score. To map those many combinations of CVSS vectors to the available 101 scores between 0.0 and 10.0, those metrics were combined into similar groups called equivalency sets. The grouping means that some similar vector strings, even though distinct, had to be mapped to the same numeric score.

You can read more about equivalency sets in the CVSS User Guide.

C) Why do some metrics, such as User Interaction or Privileges Required, have less than expected impact on the resulting score?

The grouping of metrics within equivalency sets means that the User Interaction and Privilege Required metrics are grouped with combinations of Attack Vector metrics. (See Table 24 in the CVSS Specification Document). As a result, there was only so much available fidelity for numeric scores and these metrics.

This relates to the limitation of there being many more combinations of metrics than spaces in the 101 available scores in the 0 to 10 scale.

An effort was made to make scores more distinct, by adding a fuzzing feature to the CVSS v4.0 calculator. However, to avoid disrupting the ranking system too much, the scoring difference was limited to 0.1 when selecting different metrics.

D) Why do subsequent system metric scores with any combination of non- High SC, SI, or SA metrics result in the same score?

Example: The following metric strings all equate to the same numeric score, a CVSS score of 9.3.

- CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:L/SI:L/SA:L
- CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:N/SI:L/SA:L
- CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:N/SI:N/SA:L
- CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:N/SI:N/SA:N

This scoring question again relates to the grouping of metrics into equivalency sets to deal with the limited range of scores between 0 and 10. The addition of the Safety metric further reduces the potential unique scores for this set of metrics. The constraints for subsequent system metrics group each of the Low or None metrics for Subsequent System Confidentiality, Integrity, and Availability into one level. As a result, any combination of these scores (Low / Low / Low through None / None / None) become equal and the score does not change. See Table 27 in the CVSS Specification Document for the details on this grouping.

3.18. How can I implement a CVSS calculator for my own organization?

A number of community-developed CVSS calculator libraries have been developed to help with this purpose. See the end of this FAQ entry for a list of CVSS helper libraries.

This guide can help you with implementing and validating the accuracy of a calculator for your own organization. The way to derive numeric scores from vector strings is a little different with CVSS v4.0. If you want to use one of the libraries put forth by the community, or develop a CVSS calculator library of your own, you should have a way to make sure you get results consistent with the CVSS v4.0 standard.

The CVSS SIG cannot check each calculator for consistent scoring. It is up to the maintainer of each library to verify their own scoring consistency, and for each calculator implementer to check their own results.

How can you make sure your calculator is consistent with the official standard?

3.18.1. Calculator Validation Steps

If you are interested in implementing a calculator that will work for your organization, consider the following ways to validate that your calculator returns results consistent with the standard.

The Full Metric Space

There are roughly fifteen million valid CVSS v4 metric strings. Ideally, your calculator implementation should return the correct value for each, and a negative result for the trillions of other potential but invalid combinations.

Sounds like a tall task? It may be. The CVSS SIG is working on ways to check the full metric space. Until then, consider the following phased approach.

The following options for validating the output of CVSS calculator implementations are ranked from most comprehensive to least comprehensive.

If you discover an error in any of the implementations, please contact the CVSS SIG as well as the maintainer of the helper libraries so that we can assist in correcting any errors.

See the FIRST CVSS resources repo for python code to generate these vector strings as described below.

See the FIRST CVSS resources repo for documents that contain the sets of vector strings and the resulting correct scores as described below.

The Reference Set

In the github repo, this is a set of diverse metrics using each vector value across the range of metrics. This is a representative set of metrics that should test each equivalent set and is the most comprehensive test of calculator output, short of testing the full metric space.

This set of data is roughly fifty thousand records of full valid metric strings, representing each equivalent set. There are also a number of negative tests that include invalid strings as well as strings that are valid but should return zero from the calculator.

This data can be used to extensively check any calculator implementation output. Use this reference set to verify your own calculator output against the data in the set compiled from the official calculator.

Base and Threat

If you are a vendor, in all likelihood you only want to provide v4 vectors using CVSS base or possibly base and threat vectors. In the repo there is some simple code to generate the full set of roughly 100,000 base scores, or 400,000 base plus threat scores. The data set also exists as a standalone set of vectors as well as vectors and the resulting score which can be used to check the output of your calculator implementation.

Equivalent Sets

As part of the CVSS v4 math, the CVSS SIG combined metric vector strings into 270 sets. The easiest method to test your calculator is using this set, the highest order set of vectors for each set.

3.18.2. List of Calculator Libraries

Below is a list of libraries providing CVSS scoring functionality. The CVSS SIG makes no guarantees about the accuracy of the scoring output. Validate the use of these libraries with the guidance in this FAQ entry.

Origin of the official calculator
RedHatSecurity : https://github.com/RedHatProductSecurity/cvss

API implementation of the official calculator, by Akshat Vaid:
https://github.com/akshatvaid/cvss-v4-node-api

Other ports
(JavaScript/TypeScript) pandatix : https://github.com/pandatix/js-cvss
(Golang) pandatix : https://github.com/pandatix/go-cvss

(Rust) emocrab : https://github.com/emo-crab/nvd-rs/tree/main/nvd-cvss

(Python) Benjamin Edwards : https://github.com/bjedwards/cvss4py

(Typst) Drake Axelrold : https://github.com/DrakeAxelrod/cvss.typ

(Perl) Giuseppe Di Terlizzi : https://github.com/giterlizzi/perl-CVSS

3.19. How can I apply CVSS concepts to AI / LLM application issues?

There have been questions about using CVSS to evaluate issues in generative AI and other LLM applications. CVSS is designed to be used to evaluate the impact of cybersecurity vulnerabilities, impacting the confidentiality, integrity, or availability of data within an information system. See the vulnerability definition in the CVSS User Guide.

Analysts can typically apply CVSS to certain classes of cybersecurity vulnerabilities commonly seen in such applications, such as model poisoning, denial of service, or information disclosure. CVSS may not work well for issues like model inversion, inference, or prompt injection that do not have measurable impacts and relate more to bias, ethics, or legal issues.

Vulnerability analysts need to consider the security policy, threat model, and intended use of the application. Many current generative AI applications cannot presume to be totally safe from certain classes of attacks, and the usage policy reflects this. Warnings to verify output of LLMs are present in many usage policies.

While a security policy may state that the model state should be private, current applications can't always be made to withstand model inversion. While this would be a violation of the policy, and thus an impact, analysts should consider that with the current design of applications, it is difficult to prevent model inversion. See Discover ML Model Ontology in Mitre ATLAS.

Model safety restriction bypass or prompt injection are viewed typically as a configuration weakness, similar to a web server configuration flaw. See Evade ML Model and LLM Prompt Injection on Mitre ATLAS.

AI Vulnerability Scoring Rubric

1. Are there measurable confidentiality, integrity, or availability impacts? Especially to the infrastructure or underlying application?

Use CVSS according to those measurable impacts.

2. Are model outputs incorrect, biased, or potentially harmful?

If there are confidentiality impacts, compare against usage policy and the security model.

CVSS is designed to score cybersecurity vulnerabilities. The CVSS SIG acknowledges the importance of tracking ethical or legal risks which in risk management language are also caused by vulnerabilities, but are not cybersecurity vulnerabilities. CVSS should not be used to score ethical or legal vulnerabilities.

3. Are model outputs confined to a single user?

With no reflected outputs, there are no impacts to the integrity or availability of the application.

Version History

Date	Ver	Description
2023-11-01	v1.0	Initial Publication
2024-02-12	v1.1	Added FAQ on Attack Requirements and system configuration
2024-07-19	v1.2	Added FAQ on CSRF and Why of the Math
2024-08-27	v1.3	Added FAQ on Calculator Guide and AI / LLM concepts