Common Vulnerability Scoring System v3.0: Specification Document

The Common Vulnerability Scoring System (CVSS) is an open framework for communicating the characteristics and severity of software vulnerabilities. CVSS consists of three metric groups: Base, Temporal, and Environmental. The Base group represents the intrinsic qualities of a vulnerability, the Temporal 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. The Base metrics produce a score ranging from 0 to 10, which can then be modified by scoring the Temporal and Environmental metrics. A CVSS score is also represented as a vector string, a compressed textual representation of the values used to derive the score. This document provides the official specification for CVSS v3.0.

Contents

Contents....2
Resources & Links....3
Acknowledgements....4
Introduction....5
Metrics....5
Scoring....6
Base Metrics....7
Exploitability Metrics....7
Attack Vector (AV)....7
Attack Complexity (AC)....8
Privileges Required (PR)....9
User Interaction (UI)....9
Scope (S)....9
Impact Metrics....10
Confidentiality Impact (C)....10
Integrity Impact (I)....11
Availability Impact (A)....11
Temporal Metrics....12
Exploit Code Maturity (E)....12
Remediation Level (RL)....13
Report Confidence (RC)....13
Environmental Metrics....14
Security Requirements (CR, IR, AR)....14
Modified Base Metrics....15
Qualitative Severity Rating Scale....16
Vector String....17
CVSS v3.0 XML Schema Definition....18
CVSS v3.0 Equations....18
Base....18
Temporal....19
Environmental....19
Metrics Levels....19
A Word on CVSS v3.0 Equations and Scoring....21

Resources & Links

Below are useful references to additional CVSS v3.0 documents.

Resource	Location
Specification Document	Includes metric descriptions, formulas, and vector string. Available at, http://www.first.org/cvss/specification-document
User guide	Includes further discussion of CVSS v3.0, a scoring rubric, and a glossary. Available at, http://www.first.org/cvss/user-guide
Example document	Includes examples of CVSS v3.0 scoring in practice. Available at, https://www.first.org/cvss/examples
CVSS v3.0 logo	Low and hi-res images available at, http://www.first.org/cvss/identity
CVSS v3.0 calculator	Reference implementation of the CVSS v3.0 equations, available at, http://www.first.org/cvss/calculator/3.0
JSON and XML schemas	JSON and XML schema definitions available at, https://www.first.org/cvss/data-representations

Acknowledgements

FIRST sincerely wishes to recognize the contributions of the following CVSS Special Interest Group (SIG) members, and all those who have provided valuable comments, listed in alphabetical order:

Renchie Abraham (SAP)	Luca Allodi (University of Trento)
Divya Arora (Intel)	Nazira Carlage (EMC)
Matthew Coles (EMC)	Dave Dugal (Juniper)
Mick Eckert (Bank of America)	Seth Hanford (Cisco/TIAA-CREF)
Max Heitman (Citi)	Jeffrey Heller (Sandia National Laboratories)
Art Manion (CERT/CC)	Bruce Monroe (Intel)
Scott Moore (IBM)	Dale Rich (Depository Trust & Clearing Corporation)
Sasha Romanosky (Carnegie Mellon University)	Frank Romeo (Citi)
Karen Scarfone (Scarfone Cybersecurity)	Arjuna Shunn (Microsoft)
John Stuppi (Cisco)	Masato Terada (Information-Technology Promotion Agency, Japan)
Garret Wassermann (CERT/CC)	Chuck Wergin (NIST)
Darius Wiles (Oracle)	Arnold Yoon (Dell)

FIRST would also like to thank Jennifer Daily for her creative design efforts, Deloitte & Touche LLP for their statistical assistance, Kacy Hangca (Neustar) for her tireless work facilitating our meetings, and Martin Lee (Cisco) for his analysis of nearly 30,000 CVSS v2.0 vectors assigned by 3 distinct vulnerability databases.

Finally, FIRST and the CVSS SIG would like to acknowledge the contributions and leadership of Seth Hanford and Max Heitman, chairs of the CVSS SIG.

Introduction

Software, hardware and firmware vulnerabilities pose a critical risk to any organization operating a computer network, and can be difficult to categorize and mitigate. The Common Vulnerability Scoring System (CVSS) provides a way to capture the principal characteristics of a vulnerability, and produce a numerical score reflecting its severity, as well as a textual representation of that score. The numerical score can then be translated into a qualitative representation (such as low, medium, high, and critical) to help organizations properly assess and prioritize their vulnerability management processes.

In short, CVSS affords three important benefits. First, it provides standardized vulnerability scores. When an organization uses a common algorithm for scoring vulnerabilities across all IT platforms, it can leverage a single vulnerability management policy defining the maximum allowable time to validate and remediate a given vulnerability. Next, it provides an open framework. Users may be confused when a vulnerability is assigned an arbitrary score by a third party. With CVSS, the individual characteristics used to derive a score are transparent. Finally, CVSS enables prioritized risk. When the environmental score is computed, the vulnerability becomes contextual to each organization, and helps provide a better understanding of the risk posed by this vulnerability to the organization.

This document describes the official CVSS v3.0 specification.

Metrics

CVSS is composed of three metric groups, Base, Temporal, and Environmental, each consisting of a set of metrics, as shown in Figure 1.

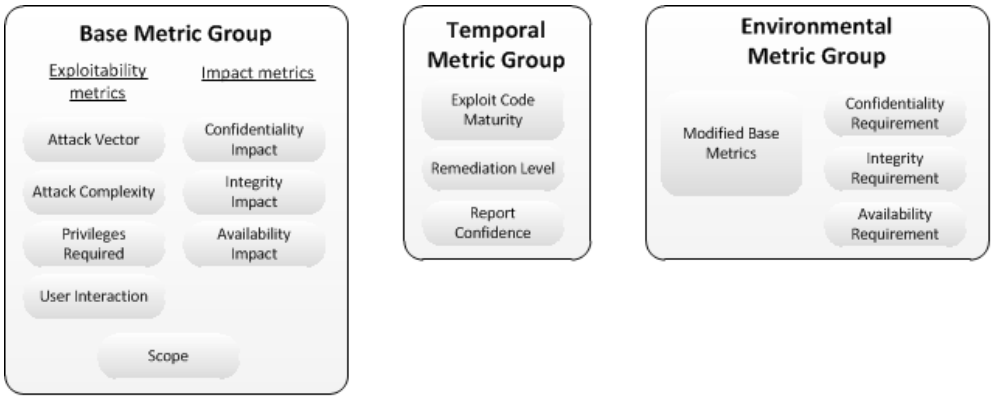

Figure 1: CVSS v3.0 Metric Groups

The Base metric group represents the intrinsic characteristics of a vulnerability that are constant over time and across user environments. It is composed of two sets of metrics: the Exploitability metrics and the Impact metrics.

The Exploitability metrics reflect the ease and technical means by which the vulnerability can be exploited. That is, they represent characteristics of the *thing that is vulnerable*, which we refer to formally as the *vulnerable component*. On the other hand, the Impact metrics reflect the direct consequence of a successful exploit, and represent the consequence to the *thing that suffers the impact*, which we refer to formally as the *impacted component*.

While the vulnerable component is typically a software application, module, driver, etc. (or possibly even a hardware device), the impacted component could be a software application, a hardware device or a network resource. This potential for measuring the impact of a vulnerability other than the vulnerable component, is a key feature of CVSS v3.0. This property is captured, and further discussed by the Scope metric below.

The Temporal metric group reflects the characteristics of a vulnerability that may change over time but not across user environments. For example, the presence of a simple-to-use exploit kit would increase the CVSS score, while the creation of an official patch would decrease it.

The Environmental metric group represents the characteristics of a vulnerability that are relevant and unique to a particular user's environment. These metrics allow the scoring analyst to incorporate security controls which may mitigate any consequences, as well as promote or demote the importance of a vulnerable system according to her business risk.

Each of these metrics are discussed in further detail below.

Scoring

When the Base metrics are assigned values by an analyst, the Base equation computes a score ranging from 0.0 to 10.0 as illustrated in Figure 2.

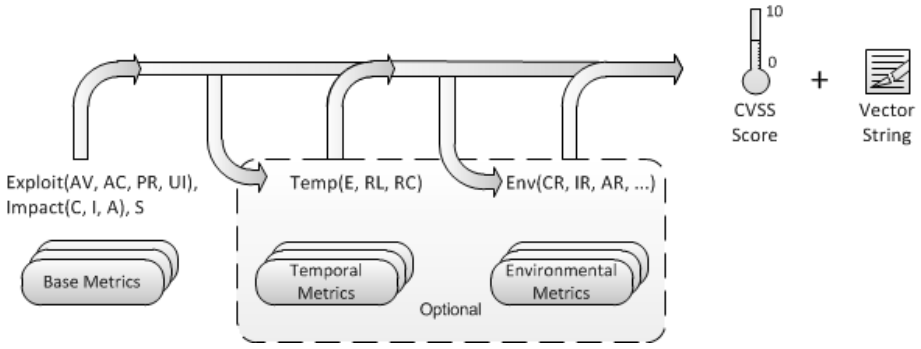

Figure 2: CVSS Metrics and Equations

Specifically, the Base equation is derived from two sub equations: the Exploitability sub score equation, and the Impact sub score equation. The Exploitability sub score equation is derived from the Base Exploitability metrics, while the Impact sub score equation is derived from the Base Impact metrics.

The Base score can then be refined by scoring the Temporal and Environmental metrics in order to more accurately reflect the risk posed by a vulnerability to a user's environment. However, scoring the Temporal and Environmental metrics is not required.

Generally, the Base and Temporal metrics are specified by vulnerability bulletin analysts, security product vendors, or application vendors because they typically possess the most accurate information about the characteristics of a vulnerability. On the other hand, the Environmental metrics are specified by end-user organizations because they are best able to assess the potential impact of a vulnerability within their own computing environment.

Scoring CVSS metrics also produces a vector string, a textual representation of the metric values used to score the vulnerability. This vector string is a specifically formatted text string that contains each value assigned to each metric, and should always be displayed with the vulnerability score.

The scoring equations and vector string are explained further below.

Note that all metrics should be scored under the assumption that the attacker has already located and identified the vulnerability. That is, the analyst need not consider the means by which the vulnerability was identified. In addition, it is likely that many different types of individuals will be scoring vulnerabilities (e.g. software vendors, vulnerability bulletin analysts, security product vendors, etc.), however, note that vulnerability scoring is intended to be agnostic to the individual and their organization.

Base Metrics

Exploitability Metrics

As mentioned, the Exploitability metrics reflect the characteristics of the *thing* that is vulnerable, which we refer to formally as the *vulnerable component*. Therefore, each of the Exploitability metrics listed below should be scored relative to the vulnerable component, and reflect the properties of the vulnerability that lead to a successful attack.

Attack Vector (AV)

This metric reflects the context by which vulnerability exploitation is possible. This metric value (and consequently the Base score) will be larger the more remote (logically, and physically) an attacker can be in order to exploit the vulnerable component. The assumption is that the number of potential attackers for a vulnerability that could be exploited from across the Internet is larger than the number of potential attackers that could exploit a vulnerability requiring physical access to a device, and therefore warrants a greater score. The list of possible values is presented in Table 1.

Table 1: Attack Vector

Metric Value	Description
Network (N)	A vulnerability exploitable with network access means the vulnerable component is bound to the network stack and the attacker's path is through OSI layer 3 (the network layer). Such a vulnerability is often termed “remotely exploitable” and can be thought of as an attack being exploitable one or more network hops away (e.g. across layer 3 boundaries from routers). An example of a network attack is an attacker causing a denial of service (DoS) by sending a specially crafted TCP packet from across the public Internet (e.g. CVE-2004-0230).
Adjacent (A)	A vulnerability exploitable with adjacent network access means the vulnerable component is bound to the network stack, however the attack is limited to the same shared physical (e.g. Bluetooth, IEEE 802.11), or logical (e.g. local IP subnet) network, and cannot be performed across an OSI layer 3 boundary (e.g. a router). An example of an Adjacent attack would be an ARP (IPv4) or neighbor discovery (IPv6) flood leading to a denial of service on the local LAN segment. See also CVE-2013-6014.
Local (L)	A vulnerability exploitable with Local access means that the vulnerable component is not bound to the network stack, and the attacker’s path is via read/ write/execute capabilities. In some cases, the attacker may be logged in locally in order to exploit the vulnerability, otherwise, she may rely on User Interaction to

	execute a malicious file.
Physical (P)	A vulnerability exploitable with Physical access requires the attacker to physically touch or manipulate the vulnerable component. Physical interaction may be brief (e.g. evil maid attack[1]) or persistent. An example of such an attack is a cold boot attack which allows an attacker to access to disk encryption keys after gaining physical access to the system, or peripheral attacks such as Firewire/USB Direct Memory Access attacks.

Attack Complexity (AC)
This metric describes the conditions beyond the attacker's control that must exist in order to exploit the vulnerability. As described below, such conditions may require the collection of more information about the target, the presence of certain system configuration settings, or computational exceptions. Importantly, the assessment of this metric excludes any requirements for user interaction in order to exploit the vulnerability (such conditions are captured in the User Interaction metric). This metric value is largest for the least complex attacks. The list of possible values is presented in Table 2.

Table 2: Attack Complexity

Metric Value	Description
Low (L)	Specialized access conditions or extenuating circumstances do not exist. An attacker can expect repeatable success against the vulnerable component.
High (H)	A successful attack depends on conditions beyond the attacker's control. That is, a successful attack cannot be accomplished at will, but requires the attacker to invest in some measurable amount of effort in preparation or execution against the vulnerable component before a successful attack can be expected.[2] For example, a successful attack may depend on an attacker overcoming any of the following conditions: • The attacker must conduct target-specific reconnaissance. For example, on target configuration settings, sequence numbers, shared secrets, etc. • The attacker must prepare the target environment to improve exploit reliability. For example, repeated exploitation to win a race condition, or overcoming advanced exploit mitigation techniques. • The attacker must inject herself into the logical network path between the target and the resource requested by the victim in order to read and/or modify network communications (e.g. man in the middle attack).

Privileges Required (PR)
This metric describes the level of privileges an attacker must possess *before* successfully exploiting the vulnerability. This metric if greatest if no privileges are required. The list of

1 See https://www.schneier.com/blog/archives/2009/10/evil_maid_attac.html for a description of the evil maid attack.

2 Note that we make no comment regarding the amount of effort required. We simply consider that some amount of additional effort must be exerted in order to exploit the vulnerability.

possible values is presented in Table 3.

Table 3: Privileges Required

Metric Value	Description
None (N)	The attacker is unauthorized prior to attack, and therefore does not require any access to settings or files to carry out an attack.
Low (L)	The attacker is authorized with (i.e. requires) privileges that provide basic user capabilities that could normally affect only settings and files owned by a user. Alternatively, an attacker with Low privileges may have the ability to cause an impact only to non-sensitive resources.
High (H)	The attacker is authorized with (i.e. requires) privileges that provide significant (e.g. administrative) control over the vulnerable component that could affect component-wide settings and files.

User Interaction (UI)
This metric captures the requirement for a user, other than the attacker, to participate in the successful compromise of the vulnerable component. This metric determines whether the vulnerability can be exploited solely at the will of the attacker, or whether a separate user (or user-initiated process) must participate in some manner. This metric value is greatest when no user interaction is required. The list of possible values is presented in Table 4.

Table 4: User Interaction

Metric Value	Description
None (N)	The vulnerable system can be exploited without interaction from any user.
Required (R)	Successful exploitation of this vulnerability requires a user to take some action before the vulnerability can be exploited. For example, a successful exploit may only be possible during the installation of an application by a system administrator.

Scope (S)
An important property captured by CVSS v3.0 is the ability for a vulnerability in one software component to impact resources beyond its means, or privileges. This consequence is represented by the metric Authorization Scope, or simply Scope.

Formally, Scope refers to the collection of privileges defined by a computing authority (e.g. an application, an operating system, or a sandbox environment) when granting access to computing resources (e.g. files, CPU, memory, etc). These privileges are assigned based on some method of identification and authorization. In some cases, the authorization may be simple or loosely controlled based upon predefined rules or standards. For example, in the case of Ethernet traffic sent to a network switch, the switch accepts traffic that arrives on its ports and is an authority that controls the traffic flow to other switch ports.
When the vulnerability of a software component governed by one authorization scope is able to affect resources governed by another authorization scope, a Scope change has occurred.

Intuitively, one may think of a scope change as breaking out of a sandbox, and an example would be a vulnerability in a virtual machine that enables an attacker to delete files on the host OS (perhaps even its own VM). In this example, there are two separate authorization authorities: one that defines and enforces privileges for the virtual machine and its users, and one that defines and enforces privileges for the host system within which the virtual machine runs.

A scope change would not occur, for example, with a vulnerability in Microsoft Word that allows an attacker to compromise all system files of the host OS, because the same authority enforces privileges of the user's instance of Word, and the host's system files.

The Base score is greater when a scope change has occurred. The list of possible values is presented in Table 5.

Table 5: Scope

Metric Value	Description
Unchanged (U)	An exploited vulnerability can only affect resources managed by the same authority. In this case the vulnerable component and the impacted component are the same.
Changed (C)	An exploited vulnerability can affect resources beyond the authorization privileges intended by the vulnerable component. In this case the vulnerable component and the impacted component are different.

Impact Metrics

The Impact metrics refer to the properties of the impacted component. Whether a successfully exploited vulnerability affects one or more components, the impact metrics are scored according to the component that suffers the worst outcome that is most directly and predictably associated with a successful attack. That is, analysts should constrain impacts to a reasonable, final outcome which they are confident an attacker is able to achieve.

If a scope change has not occurred, the Impact metrics should reflect the confidentiality, integrity, and availability (CIA) impact to the vulnerable component. However, if a scope change has occurred, then the Impact metrics should reflect the CIA impact to either the vulnerable component, or the impacted component, whichever suffers the most severe outcome.

Confidentiality Impact (C)

This metric measures the impact to the confidentiality of the information resources managed by a software component due to a successfully exploited vulnerability. Confidentiality refers to limiting information access and disclosure to only authorized users, as well as preventing access by, or disclosure to, unauthorized ones. The list of possible values is presented in Table 6. This metric value increases with the degree of loss to the impacted component.

Table 6: Confidentiality Impact

Metric Value	Description
High (H)	There is total loss of confidentiality, resulting in all resources within the impacted component being divulged to the attacker. Alternatively, access to only some restricted information is obtained, but the disclosed information

	presents a direct, serious impact. For example, an attacker steals the administrator's password, or private encryption keys of a web server.
Low (L)	There is some loss of confidentiality. Access to some restricted information is obtained, but the attacker does not have control over what information is obtained, or the amount or kind of loss is constrained. The information disclosure does not cause a direct, serious loss to the impacted component.
None (N)	There is no loss of confidentiality within the impacted component.

Integrity Impact (I)
This metric measures the impact to integrity of a successfully exploited vulnerability. Integrity refers to the trustworthiness and veracity of information. The list of possible values is presented in Table 7. This metric value increases with the consequence to the impacted component.

Table 7: Integrity Impact

Metric Value	Description
High (H)	There is a total loss of integrity, or a complete loss of protection. For example, the attacker is able to modify any/all files protected by the impacted component. Alternatively, only some files can be modified, but malicious modification would present a direct, serious consequence to the impacted component.
Low (L)	Modification of data is possible, but the attacker does not have control over the consequence of a modification, or the amount of modification is constrained. The data modification does not have a direct, serious impact on the impacted component.
None (N)	There is no loss of integrity within the impacted component.

Availability Impact (A)
This metric measures the impact to the availability of the impacted component resulting from a successfully exploited vulnerability. While the Confidentiality and Integrity impact metrics apply to the loss of confidentiality or integrity of data (e.g., information, files) used by the impacted component, this metric refers to the loss of availability of the impacted component itself, such as a networked service (e.g., web, database, email). Since availability refers to the accessibility of information resources, attacks that consume network bandwidth, processor cycles, or disk space all impact the availability of an impacted component. The list of possible values is presented in Table 8. This metric value increases with the consequence to the impacted component.

Table 8: Availability Impact

Metric Value	Description
High (H)	There is total loss of availability, resulting in the attacker being able to fully deny access to resources in the impacted component; this loss is either sustained (while the attacker continues to deliver the attack) or persistent (the condition persists even after the attack has completed). Alternatively, the attacker has the ability to deny some availability, but the loss of availability presents a direct,

	serious consequence to the impacted component (e.g., the attacker cannot disrupt existing connections, but can prevent new connections; the attacker can repeatedly exploit a vulnerability that, in each instance of a successful attack, leaks a only small amount of memory, but after repeated exploitation causes a service to become completely unavailable).
Low (L)	There is reduced performance or interruptions in resource availability. Even if repeated exploitation of the vulnerability is possible, the attacker does not have the ability to completely deny service to legitimate users. The resources in the impacted component are either partially available all of the time, or fully available only some of the time, but overall there is no direct, serious consequence to the impacted component.
None (N)	There is no impact to availability within the impacted component.

Temporal Metrics

The Temporal metrics measure the current state of exploit techniques or code availability, the existence of any patches or workarounds, or the confidence that one has in the description of a vulnerability.

Exploit Code Maturity (E)

This metric measures the likelihood of the vulnerability being attacked, and is typically based on the current state of exploit techniques, exploit code availability, or active, “in-the-wild” exploitation. Public availability of easy-to-use exploit code increases the number of potential attackers by including those who are unskilled, thereby increasing the severity of the vulnerability. Initially, real-world exploitation may only be theoretical. Publication of proof-of-concept code, functional exploit code, or sufficient technical details necessary to exploit the vulnerability may follow. Furthermore, the exploit code available may progress from a proof-of-concept demonstration to exploit code that is successful in exploiting the vulnerability consistently. In severe cases, it may be delivered as the payload of a network-based worm or virus or other automated attack tools.

The list of possible values is presented in Table 9. The more easily a vulnerability can be exploited, the higher the vulnerability score.

Table 9 : Exploit Code Maturity

Metric Value	Description
Not Defined (X)	Assigning this value to the metric will not influence the score. It is a signal to a scoring equation to skip this metric.
High (H)	Functional autonomous code exists, or no exploit is required (manual trigger) and details are widely available. Exploit code works in every situation, or is actively being delivered via an autonomous agent (such as a worm or virus). Network-connected systems are likely to encounter scanning or exploitation attempts. Exploit development has reached the level of reliable, widely-available, easy-to-use automated tools.
Functional (F)	Functional exploit code is available. The code works in most situations where the vulnerability exists.

Proof-of-Concept (P)	Proof-of-concept exploit code is available, or an attack demonstration is not practical for most systems. The code or technique is not functional in all situations and may require substantial modification by a skilled attacker.
Unproven (U)	No exploit code is available, or an exploit is theoretical.

Remediation Level (RL)

The Remediation Level of a vulnerability is an important factor for prioritization. The typical vulnerability is unpatched when initially published. Workarounds or hotfixes may offer interim remediation until an official patch or upgrade is issued. Each of these respective stages adjusts the temporal score downwards, reflecting the decreasing urgency as remediation becomes final. The list of possible values is presented in Table 10. The less official and permanent a fix, the higher the vulnerability score.

Table 10: Remediation Level

Metric Value	Description
Not Defined (X)	Assigning this value to the metric will not influence the score. It is a signal to a scoring equation to skip this metric.
Unavailable (U)	There is either no solution available or it is impossible to apply.
Workaround (W)	There is an unofficial, non-vendor solution available. In some cases, users of the affected technology will create a patch of their own or provide steps to work around or otherwise mitigate the vulnerability.
Temporary Fix (T)	There is an official but temporary fix available. This includes instances where the vendor issues a temporary hotfix, tool, or workaround.
Official Fix (O)	A complete vendor solution is available. Either the vendor has issued an official patch, or an upgrade is available.

Report Confidence (RC)

This metric measures the degree of confidence in the existence of the vulnerability and the credibility of the known technical details. Sometimes only the existence of vulnerabilities are publicized, but without specific details. For example, an impact may be recognized as undesirable, but the root cause may not be known. The vulnerability may later be corroborated by research which suggests where the vulnerability may lie, though the research may not be certain. Finally, a vulnerability may be confirmed through acknowledgement by the author or vendor of the affected technology. The urgency of a vulnerability is higher when a vulnerability is known to exist with certainty. This metric also suggests the level of technical knowledge available to would-be attackers. The list of possible values is presented in Table 11. The more a vulnerability is validated by the vendor or other reputable sources, the higher the score.

Table 11: Report Confidence

Metric Value	Description
Not Defined (X)	Assigning this value to the metric will not influence the score. It is a signal to a scoring equation to skip this metric.
Confirmed (C)	Detailed reports exist, or functional reproduction is possible (functional exploits may provide this). Source code is available to independently verify the assertions of the research, or the author or vendor of the affected code has confirmed the presence of the vulnerability.
Reasonable (R)	Significant details are published, but researchers either do not have full confidence in the root cause, or do not have access to source code to fully confirm all of the interactions that may lead to the result. Reasonable confidence exists, however, that the bug is reproducible and at least one impact is able to be verified (proof-of-concept exploits may provide this). An example is a detailed write-up of research into a vulnerability with an explanation (possibly obfuscated or "left as an exercise to the reader") that gives assurances on how to reproduce the results.
Unknown (U)	There are reports of impacts that indicate a vulnerability is present. The reports indicate that the cause of the vulnerability is unknown, or reports may differ on the cause or impacts of the vulnerability. Reporters are uncertain of the true nature of the vulnerability, and there is little confidence in the validity of the reports or whether a static Base score can be applied given the differences described. An example is a bug report which notes that an intermittent but non-reproducible crash occurs, with evidence of memory corruption suggesting that denial of service, or possible more serious impacts, may result.

Environmental Metrics

These metrics enable the analyst to customize the CVSS score depending on the importance of the affected IT asset to a user's organization, measured in terms of complementary/alternative security controls in place, Confidentiality, Integrity, and Availability. The metrics are the modified equivalent of base metrics and are assigned metrics value based on the component placement in organization infrastructure.

Security Requirements (CR, IR, AR)

These metrics enable the analyst to customize the CVSS score depending on the importance of the affected IT asset to a user's organization, measured in terms of Confidentiality, Integrity, and Availability. That is, if an IT asset supports a business function for which Availability is most important, the analyst can assign a greater value to Availability relative to Confidentiality and Integrity. Each security requirement has three possible values: Low, Medium, or High.

The full effect on the environmental score is determined by the corresponding Modified Base Impact metrics. That is, these metrics modify the environmental score by reweighting the Modified Confidentiality, Integrity, and Availability impact metrics. For example, the Modified Confidentiality impact (MC) metric has increased weight if the Confidentiality Requirement (CR) is High. Likewise, the Modified Confidentiality impact metric has decreased weight if the Confidentiality Requirement is Low. The Modified Confidentiality impact metric weighting is

neutral if the Confidentiality Requirement is Medium. This same process is applied to the Integrity and Availability requirements.

Note that the Confidentiality Requirement will not affect the Environmental score if the (Modified Base) confidentiality impact is set to None. Also, increasing the Confidentiality Requirement from Medium to High will not change the Environmental score when the (Modified Base) impact metrics are set to High. This is because the modified impact sub score (part of the Modified Base score that calculates impact) is already at a maximum value of 10.

The list of possible values is presented in Table 12. For brevity, the same table is used for all three metrics. The greater the Security Requirement, the higher the score (recall that Medium is considered the default).

Table 12: Security Requirements

Metric Value	Description
Not Defined (X)	Assigning this value to the metric will not influence the score. It is a signal to the equation to skip this metric.
High (H)	Loss of [Confidentiality | Integrity | Availability] is likely to have a catastrophic adverse effect on the organization or individuals associated with the organization (e.g., employees, customers).
Medium (M)	Loss of [Confidentiality | Integrity | Availability] is likely to have a serious adverse effect on the organization or individuals associated with the organization (e.g., employees, customers).
Low (L)	Loss of [Confidentiality | Integrity | Availability] is likely to have only a limited adverse effect on the organization or individuals associated with the organization (e.g., employees, customers).

Modified Base Metrics

These metrics enable the analyst to adjust the Base metrics according to modifications that exist within the analyst's environment. That is, if an environment has made general changes for the affected software that differs in a way which would affect its Exploitability, Scope, or Impact, then the environment can reflect this via an appropriately-modified, Environmental score.

The full effect on the Environmental score is determined by the corresponding Base metrics. That is, these metrics modify the Environmental score by reassigning the (Base) metrics values, prior to applying the (Environmental) Security Requirements. For example, the default configuration for a vulnerable component may be to run a listening service with administrator privileges, for which a compromise might grant an attacker Confidentiality, Integrity, and Availability impacts that are all High. Yet, in the analyst's environment, that same Internet service might be running with reduced privileges; in that case, the Modified Confidentiality, Modified Integrity, and Modified Availability might each be set to Low.

For brevity, only the names of the Modified Base metrics are mentioned. Each Modified Environmental metric has the same values as its corresponding Base metric, plus a value of Not Defined.

The intent of this metric is to define the mitigations in place for a given environment. It is acceptable to use the Modified metrics to describe situations that increase the Base score. For example, the default configuration of a component may be to require high privileges (PR: High) in order to access a particular function, but in the analyst's environment, there may be no privileges required (PR: None). The analyst can set MPR: None to reflect this more serious condition for their environment.

The list of possible values is presented in Table 13.

Table 13: Modified Base Metrics

Modified Base Metric	Corresponding Values
Modified Attack Vector (MAV)	The same values as the corresponding Base Metric (see Base Metrics above), as well as Not Defined (the default)
Modified Attack Complexity (MAC)	
Modified Privileges Required (MPR)	
Modified User Interaction (MUI)	
Modified Scope (MS)	
Modified Confidentiality (MC)	
Modified Integrity (MI)	
Modified Availability (MA)	

Qualitative Severity Rating Scale

For some purposes it is useful to have a textual representation of the numeric Base, Temporal and Environmental scores. All scores can be mapped to the qualitative ratings defined in Table 14.[3]

Table 14: Qualitative severity rating scale

Rating	CVSS Score
None	0.0
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 - 10.0

As an example, a CVSS Base score of 4.0 has an associated severity rating of Medium. The use of these qualitative severity ratings is optional, and there is no requirement to include them when publishing CVSS scores. They are intended to help organizations properly assess and prioritize their vulnerability management processes.

Vector String

The CVSS v3.0 vector string is a text representation of a set of CVSS metrics. It is commonly used to record or transfer CVSS metric information in a concise form.

3 Note that this mapping between quantitative and qualitative scores applies whether just the Base, or all of Base, Temporal, and Environmental metric groups, are scored.

The v3.0 vector string begins with the label “CVSS:” and a numeric representation of the current version, “3.0.” Metric information follows in the form of a set of metrics, each metric being preceded by a forward slash, “/”, acting as a delimiter. Each metric is a metric name in abbreviated form, a colon, “:”, and its associated metric value in abbreviated form. The abbreviated forms are defined earlier in this specification (in parentheses after each metric name and metric value), and are summarized in the table below.

Metrics may be specified in any order in a vector string, though Table 15. shows the preferred order. All Base metrics must be included in a vector string. Temporal and Environmental metrics are optional, and omitted metrics are considered to have the value of Not Defined (X). Metrics with a value of Not Defined can be explicitly included in a vector string if desired. Programs reading v3.0 vector strings must accept metrics in any order and treat unspecified Temporal and Environmental as Not Defined. A vector string must not include the same metric more than once.

Table 15: Base, Temporal and Environmental Vectors

Metric Group	Metric Name and Abbreviated Form	Possible Values	Mandatory?
Base	Attack Vector, AV	[N,A,L,P]	Yes
	Attack Complexity, AC	[L,H]	Yes
	Privileges Required, PR	[N,L,H]	Yes
	User Interaction, UI	[N,R]	Yes
	Scope, S	[U,C]	Yes
	Confidentiality, C	[H,L,N]	Yes
	Integrity, I	[H,L,N]	Yes
	Availability, A	[H,L,N]	Yes
Temporal	Exploit Code Maturity, E	[X,H,F,P,U]	No
	Remediation Level, RL	[X,U,W,T,O]	No
	Report Confidence, RC	[X,C,R,U]	No
Environmental	Confidentiality Req., CR	[X,H,M,L]	No
	Integrity Req., IR	[X,H,M,L]	No
	Availability Req., AR	[X,H,M,L]	No
	Modified Attack Vector, MAV	[X,N,A,L,P]	No
	Modified Attack Complexity, MAC	[X,L,H]	No
	Modified Privileges Required, MPR	[X,N,L,H]	No
	Modified User Interaction, MUI	[X,N,R]	No

	Modified Scope, MS	[X,U,C]	No
	Modified Confidentiality, MC	[X,N,L,H]	No
	Modified Integrity, MI	[X,N,L,H]	No
	Modified Availability, MA	[X,N,L,H]	No

For example, a vulnerability with Base metric values of, "Attack Vector: Network, Attack Complexity: Low, Privileges Required: High, User Interaction: None, Scope: Unchanged, Confidentiality: Low, Integrity: Low, Availability: None" and no specified Temporal or Environmental metrics would produce the following vector:

CVSS:3.0/AV:N/AC:L/PR:H/UI:N/S:U/C:L/I:L/A:N

The same example with the addition of "Exploitability: Functional, Remediation Level: Not Defined," and with the metrics in a non-preferred ordering would produce the following vector:

CVSS:3.0/S:U/AV:N/AC:L/PR:H/UI:N/C:L/I:L/A:N/E:F/RL:X

CVSS v3.0 XML Schema Definition

A CVSS XML Schema Definition (XSD) defines the structure of the XML file containing the CVSS metric values, and is useful for those wishing to store or transfer such data in XML format. The XSD is available from https://www.first.org/cvss/cvss-v3.0.xsd

CVSS v3.0 Equations

The CVSS v3.0 equations are defined below.

Base

The Base Score is a function of the Impact and Exploitability sub score equations. Where the Base score is defined as,

If (Impact sub score <= 0) *0 else,*

Scope Unchanged[4] $Roundup\left(Minimum\left[\left(Impact+Exploitability\right),10\right]\right)$

Scope Changed $Roundup\left(Minimum\left[1.08\times\left(Impact+Exploitability\right),10\right]\right)$

and the Impact sub score (ISC) is defined as,

Scope Unchanged $6.42\times ISC_{Base}$

Scope Changed $7.52\times\left[ISC_{Base}-0.029\right]-3.25\times\left[ISC_{Base}-0.02\right]^{15}$

Where,

$$ISC_{Base}=1-\left[\left(1-Impact_{Conf}\right)\times\left(1-Impact_{Integ}\right)\times\left(1-Impact_{Avail}\right)\right]$$

And the Exploitability sub score is,

4 Where "Round up" is defined as the smallest number, specified to one decimal place, that is equal to or higher than its input. For example, Round up (4.02) is 4.1; and Round up (4.00) is 4.0.

$8.22 \times AttackVector \times AttackComplexity \times PrivilegeRequired \times UserInteraction$

Temporal
The Temporal score is defined as,

$$Roundup(BaseScore \times ExploitCodeMaturity \times RemediationLevel \times ReportConfidence)$$

Environmental
The environmental score is defined as,

If (Modified Impact Sub score <= 0)	0 else,
If Modified Scope is Unchanged	*Round up(Round up (Minimum [(M.Impact + M.Exploitability) ,10]) × Exploit Code Maturity × Remediation Level × Report Confidence)*
If Modified Scope is Changed	*Round up(Round up (Minimum [1.08 × (M.Impact + M.Exploitability) ,10]) × Exploit Code Maturity × Remediation Level × Report Confidence)*

And the modified Impact sub score is defined as,

If Modified Scope is Unchanged	$6.42 \times [ISC_{Modified}]$
If Modified Scope is Changed	$7.52 \times [ISC_{Modified} - 0.029] - 3.25 \times [ISC_{Modified} - 0.02]^{15}$

Where,

$$ISC_{Modified} = Minimum\left[\left[1 - (1 - M.I_{Conf} \times CR) \times (1 - M.I_{Integ} \times IR) \times (1 - M.I_{Avail} \times AR)\right], 0.915\right]$$

The Modified Exploitability sub score is,

$$8.22 \times M.AttackVector \times M.AttackComplexity \times M.PrivilegeRequired \times M.UserInteraction$$

Metrics Levels

The metric values are defined in Table 16.

Table 16: Metric values

Metric	Metric Value	Numerical Value

Attack Vector / Modified Attack Vector	Network	0.85
	Adjacent Network	0.62
	Local	0.55
	Physical	0.2
Attack Complexity / Modified Attack Complexity	Low	0.77
	High	0.44
Privilege Required / Modified Privilege Required	None	0.85
	Low	0.62 (0.68 if Scope / Modified Scope is Changed)
	High	0.27 (0.50 if Scope / Modified Scope is Changed)
User Interaction / Modified User Interaction	None	0.85
	Required	0.62
C,I,A Impact / Modified C,I,A Impact	High	0.56
	Low	0.22
	None	0
Exploit Code Maturity	Not Defined	1
	High	1
	Functional	0.97
	Proof of Concept	0.94
	Unproven	0.91
Remediation Level	Not Defined	1
	Unavailable	1
	Workaround	0.97
	Temporary Fix	0.96
	Official Fix	0.95
Report Confidence	Not Defined	1
	Confirmed	1
	Reasonable	0.96
	Unknown	0.92
Security Requirements – C,I,A Requirements (CR)	Not Defined	1
	High	1.5
	Medium	1
	Low	0.5

A Word on CVSS v3.0 Equations and Scoring

The CVSS v3.0 formula provides a mathematical approximation of all possible metric combinations ranked in order of severity (a vulnerability lookup table). To produce the CVSS

v3.0 formula, the SIG framed the lookup table by assigning v3.0 metric values to real vulnerabilities, and a severity group (low, medium, high, critical). Having defined the acceptable numeric ranges for each severity level, the SIG then collaborated with Deloitte & Touche LLP to adjust formula parameters in order to align v3.0 metric combinations to the SIG's proposed severity ratings.

Given that there are a limited number of numeric outcomes (101 outcomes, ranging from 0.0 to 10.0), multiple scoring combinations may produce the same numeric score. In addition, some numeric scores may be omitted because the weights and calculations are derived from the severity ranking of metric combinations. Further, in some cases, metric combinations may deviate from the desired severity threshold. This is unavoidable and a simple correction is not readily available because adjustments made to one metric value or equation parameter in order to fix a deviation, cause other, potentially more severe deviations.

By consensus, and as was done with CVSS v2.0, the acceptable deviation was a value of 0.5. That is, all the metric value combinations used to derive the weights and calculation will produce a numeric score within its assigned severity level, or within 0.5 of that assigned level. For example, a combination expected to be rated as a “high” may have a numeric score between 6.6 and 9.3. Finally, CVSS v3.0 retains the range from 0.0 to 10.0 for backward compatibility.