CVSS v3.0 Preview 2: Metrics / Formula / Examples
FIRST.ORG
December 2014

Introduction

In this final preview of the Common Vulnerability Scoring System 3.0 (CVSS v3.0), the CVSS Special Interest Group (CVSS-SIG) has provided the following:

- Updated Metrics Descriptions and values for the CVSS v3.0 (This Document)
- Updated vector string notation to represent vulnerability's CVSS v3.0 score in an abbreviated format. (This Document)
- CVSS v3.0 Formula (CVSS_v30_Preview2_Formula_December_2014.pdf) and online calculator <http://www.first.org/cvss/calculator/3.0>
- Examples Document (CVSS_v30_Preview2_Examples_December_2014.pdf)

Although we delayed the release by a few weeks, the CVSS-SIG hopes that this final preview with complete formula and examples will provide the necessary detail and context to ensure a productive public assessment, and ultimately a robust standard that best meets the need of the security community at large.

As with preview release 1, it is our hope that teams will fully utilize access to this preview and begin to produce CVSS v3.0 scores alongside whatever other scoring system they are using today. When the completed CVSS v3.0 standard is approved, organizations that have stored scores produced via CVSS v3.0 previews and can use that data to offer official CVSS v3.0 scoring data.

That being said, some of the changes since preview 1 have material impact to the metric decisions made by the analysts and associated scoring outcomes. Therefore any vulnerabilities previously assessed with CVSSv3.0 Preview Release 1, will need to be updated to reflect changes made in Preview Release 2. Specifically:

- When to Calculate a CVSS v3.0 Score
- Availability Impact
- Attack Complexity

WHILE THE CVSS-SIG HOPES THAT MANY WILL TAKE ADVANTAGE OF THIS PREVIEW TO HELP THEMSELVES BECOME ACQUAINTED WITH THE STANDARD, WE ASK THAT NOONE USE THIS DOCUMENT TO GIVE OFFICIAL PUBLIC CVSS V3.0 METRICS OR VECTOR STRINGS TO VULNERABILITIES.

The CVSS-SIG does not want to discourage any public commentary regarding CVSS v3.0 preview 2, but we feel the community would be disadvantaged by anyone assigning CVSS v3.0 metrics in any official, public manner (such as in a product security advisory, as the results of a vulnerability scan, in a vulnerability database, etc.) before the final specification is released.

CVSS v3.0 Preview Release 2 Comments period will be open from publication of this document through February 28th 2015. Please submit all comments to:
cvss-v3-comments@first.org

Max Heitman
Co-Chair, CVSS-SIG
max.heitman@citi.com

When to calculate the CVSS Score

CVSS v2 did not explicitly state at which point of an attack the CVSS score should be calculated. This led to variations in impact metrics between scorers.

For CVSS v3.0 the score should be calculated when an attack first causes an impact to Confidentiality, Integrity or Availability. This initial impact should be used in the score, and possible further impacts should not be considered.

For example, a vulnerability that reveals a plaintext password should be scored as having only a Confidentiality impact, even if the possibility exists for the password to be used to cause further impacts, e.g. to login to a system and modify its data.

Understanding Scope & Impact

Scope is simply a way to describe (and begin to measure) vulnerabilities that have impact to systems other than the one containing the software flaw. This is often, but not always, because of implied trust between the two systems.

The cleanest way that the SIG has found to define where one system ends and another begins is to use the concept of Authorization. Authorization Scope (or simply Scope) is defined as:

> *The resources and capabilities that are enforced by an authority. This authority can be an application, an operating system, a sandbox environment such as virtualization, or something else that provides access to resources based upon a method of identification and authorization.*

The exploitability metrics (Attack Vector, Attack Complexity, Privileges Required, and User Interaction) are always scored relative to where the attacker exploits the software flaw. This is the Exploitable Authorization Scope, or simply Exploitable Scope.

Vulnerability Impact (Confidentiality / Integrity / Availability) is always scored where the impact (C/I/A loss) is experienced. We call this the Impacted Authorization Scope or simply Impact Scope.

If the Exploitable Scope of the software flaw != Impact Scope, then we score the vulnerability as SCOPE CHANGED, and the formula will raise the numeric outcome.

If the Exploitable Scope = the Impact Scope, then we score the vulnerability as SCOPE UNCHANGED and the formula computes the Exploitability and Impact values as normal, similar to CVSS v2.

Full approved metric description of Impact and Scope are found below.

Base Metrics

Attack Vector (AV)

This metric reflects the context in which the vulnerability exploitation occurs. The values for this metric are listed in the table below. The more remote an attacker can be to the target, the greater the vulnerability score. The possible values for this metric are listed in Table 1. This rationale is that, in general, the number of potential attackers for a remotely exploitable vulnerability would be much larger than that for an attack requiring local access.

Metric Value	Description
Network (N)	A vulnerability exploitable with network access means the Exploitable Scope is bound to the network stack and the attacker's path to the vulnerable system is at the network layer. Such a vulnerability is often termed "remotely exploitable". An example of a network attack is an RPC buffer overflow.
Adjacent Network (A)	A vulnerability exploitable with adjacent network access means the Exploitable Scope is bound to the network stack and the attacker's path to the vulnerable system is at the data link layer. Examples include local IP subnet, Bluetooth, IEEE 802.11, and local Ethernet segment. For instance, a vulnerability in this category would be a bug in application software that processes Ethernet frames.
Local (L)	A vulnerability exploitable with local access means the Exploitable Scope is not bound to the network stack and the attacker's path to the Exploitable Scope is via read / write / execute capabilities. If the attacker has the necessary Privileges Required to interact with the Exploitable Scope, they may be logged in locally; otherwise, they may deliver an exploit to a user and rely on User Interaction. An example of a locally exploitable vulnerability is a flaw in a word processing application when processing a malformed document.
Physical (P)	A vulnerability exploitable with physical access requires the ability to physically touch or manipulate the Exploitable Scope. Physical interaction may be brief (evil maid

	attack) or persistent. Example of such an attack is cold boot attack [1] which allows an attacker to get access to disk encryption keys after gaining physical access to the system, or peripheral attacks such as Firewire/USB Direct Memory Access attacks.

Table 1

Attack Complexity (AC)

This metric describes the conditions beyond the attacker's control that must occur in order to place the system in a vulnerable state, this also excludes any user interaction requirements. The possible values for this metric are listed in Table 2.

Metric Value	New Description
High (H)	A successful attack depends on conditions outside 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 a specific target before successful attack can be expected. A successful attack depends on attackers overcoming one OR both of the following conditions: • The attacker must gather target-specific reconnaissance; examples of this may include: target configuration settings, sequence numbers, shared secrets, etc. • The attacker must prepare the target environment to improve exploit reliability; examples of preparation may include: repeated exploitation to win a race condition, performing a heap spray, etc.
Low (L)	Specialized access conditions or extenuating circumstances do not exist. An attacker can expect repeatable exploit success against a vulnerable target

Table 2

Privileges Required (PR)

This metric describes the privileges an attacker requires before successfully exploiting the vulnerability, and the potential impact they could inflict on a system after exploiting it. The possible values for this metric are listed in Table 3.

Metric Value	Description
High (H)	The attacker is authenticated with privileges that provide significant control over component resources. With these starting privileges an attacker can cause a Complete impact to one or more of: Confidentiality, Integrity, or Availability. Alternatively, an attacker with High privileges may have the ability to cause a Partial impact to sensitive resources.

Low (L)	The attacker is authenticated with privileges that provide basic, low-impact capabilities. With these starting privileges an attacker is able to cause a Partial impact to one or more of: Confidentiality, Integrity, or Availability. Alternatively, an attacker with Low privileges may have the ability to cause an impact only to non-sensitive resources.
None (N)	The attacker is unprivileged or unauthenticated.

Table 3

User Interaction (UI)

This metric captures the requirement for a user (other than the attacker) to participate in the successful exploit of the target information system. The possible values for this metric are listed in Table 4. This new user interaction metric will determine whether or not the vulnerability can be exploited solely at the will of the attacker, or if a user must participate by taking action.

Metric Value	Description
None (N)	The vulnerable system can be exploited without any interaction from any user.
Required (R)	Successful exploitation of this vulnerability requires a user to take one or more actions that may or may not be expected in a scenario involving no exploitation, or a scenario involving content provided by a seemingly trustworthy source.

Table 4

Scope (S)

Scope:

An important conceptual change in v3.0 (i.e., one not captured by any specific metric) is that Impact metrics are now scored relative to the impacted authorization scope, or simply Impact Scope. In CVSS v2.0, Scoring Tip #2 effectively set the Scope for all v2.0 vulnerabilities to the host system [1] (V2 Guide, 3.1.1).

Vulnerabilities scored in v2.0 presented difficulties for infrastructure application vendors when scoring vulnerabilities that would, for example, fully compromise their application, but only partially affect the host operating system. A partial impact would produce a score that was likely too low, while a complete impact was likely too high. Therefore, in v3.0, vulnerabilities are now scored relative to the exploitability scope that provides the attacker with access to exploit the vulnerability, and to the Impact Scope that experiences the vulnerability's outcome; if these scopes are not the same then the Scope metric is set to Changed. The consequence of this is that v2.0 and v3.0 scores may not always be comparable.

What is an Authorization Scope?

For CVSS v3.0, Scope is defined as the resources and capabilities that are enforced by an authority. This authority can be an application, an operating system, a sandbox environment such as virtualization, or something else that provides access to resources based upon a method of identification and authorization. In some cases, the authorization may be simple or loosely controlled, allowing any well-formed request to succeed based upon predefined rules or standards (as in the case of Ethernet traffic sent to a network switch; the switch accepts traffic that arrives on its ports and is an authority which controls the traffic flow and function to other switch ports).

For CVSS v3.0, the exploitation and impact of vulnerabilities is measured relative to one or two authorities or scopes.

First, Exploitable Scope is the authority which grants an attacker access to resources which permit the exploitation of a vulnerability (against which the Exploitability sub score metrics are measured: Attack Vector, Attack Complexity, Privileges Required, User Interaction)

Second, Impact Scope, the authority which controls access to the resources against which the Impact sub score is measured (Confidentiality, Integrity, Availability).

Metric Value	Description
Unchanged (U)	Exploitable Scope = Impact Scope. The attacker attacks and impacts the authorization scope that contains the software flaw
Changed (C)	Exploitable Scope != Impact Scope The attacker attacks the software flaw in the exploitable scope but the resulting impact is to a different authorization scope.

Table 5

Confidentiality Impact (C)

This metric measures the impact to confidentiality of 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 possible values for this metric are listed in Table 6. Increased confidentiality impact increases the vulnerability score.

Metric Value	Description
None (N)	There is no impact to confidentiality within the affected scope.
Low (L)	There is informational disclosure or a bypass of access controls. Access to some restricted information is obtained, but the attacker does

	not have control over what is obtained, or the scope of the loss is constrained. The information disclosure does not have a direct, serious impact on the affected scope.
High (H)	There is total information disclosure, resulting in all resources in the affected scope being divulged to the attacker. Alternatively, access to only some restricted information is obtained, but the disclosed information presents a direct, serious impact to the affected scope (e.g. the attacker can read the administrator's password, or private keys in memory are disclosed to the attacker).

Table 6

Integrity Impact (I)

This metric measures the impact to integrity of a successfully exploited vulnerability. Integrity refers to the trustworthiness and guaranteed veracity of information. The possible values for this metric are listed in Table 7. Increased integrity impact increases the vulnerability score.

Metric Value	Description
None (N)	There is no impact to integrity within the affected scope.
Low (L)	Modification of data is possible, but the attacker does not have control over the end result of a modification, or the scope of modification is constrained. The data modification does not have a direct, serious impact on the affected scope.
High (H)	There is a total compromise of system integrity. There is a complete loss of system protection, resulting in the entire system being compromised. The attacker is able to modify any files on the target system.

Table 7

Availability Impact (A)

This metric measures the impact to the availability of the affected Impact Scope 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 a affected Impact Scope, this metric refers to the loss of availability of the affected Impact Scope, itself, such as networked service (e.g. web, database, email, etc). 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 affected Impact Scope. The possible values for this metric are listed in Table 8. Increased availability impact increases the vulnerability score.

Metric Value	Description

None (N)	There is no impact to availability within the affected scope.
Low (L)	There is reduced performance or interruptions in resource availability. The attacker does not have the ability to completely deny service to legitimate users, even through repeated exploitation of the vulnerability. The resources in the affected scope are either partially available all of the time, or fully available only some of the time, but the overall there is no direct, serious impact to the affected scope.
High (H)	There is total loss of availability, resulting in the attacker being able to fully deny access to resources in the affected scope; 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 impact to the affected scope (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).

Table 8

Temporal Metrics

Exploitability (E)

This metric measures the current state of exploit techniques or code availability. 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. The possible values for this metric are listed in Table 9. The more easily a vulnerability can be exploited, the higher the vulnerability score.

Metric Value	Description
Unproven (U)	No exploit code is available, or an exploit is entirely theoretical
Proof-of-Concept (P)	Proof-of-concept exploit code or an attack demonstration that is not practical for most systems is available. The code or technique is not functional in all situations and may require substantial modification by a skilled attacker.
Functional (F)	Functional exploit code is available. The code works in most situations where the vulnerability exists.
High (H)	Either the vulnerability is exploitable by functional mobile autonomous code, or no exploit is required (manual trigger) and details are widely available. The code works in every situation, or is actively being delivered via a mobile autonomous agent (such as a worm or virus).
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.

Table 9

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 possible values for this metric are listed in Table 10. The less official and permanent a fix, the higher the vulnerability score is.

Metric Value	Description
Official Fix (O)	A complete vendor solution is available. Either the vendor has issued an official patch, or an upgrade is available.
Temporary Fix (T)	There is an official but temporary fix available. This includes instances where the vendor issues a temporary hotfix, tool, or workaround.
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.
Unavailable (U)	There is either no solution available or it is impossible to apply.
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.

Table 10

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 possible values for this metric are listed in Table 11. The more a vulnerability is validated by the vendor or other reputable sources, the higher the score.

Metric Value	Description
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.
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.
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.
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.

Table 11

Environmental Metrics

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 base impact metrics. That is, these metrics modify the environmental score by reweighting the (base) confidentiality, integrity, and availability impact metrics. For example, the confidentiality impact (C) metric has increased weight if the confidentiality requirement (CR) is "high." Likewise, the confidentiality impact metric has decreased weight if the confidentiality requirement is "low." The confidentiality impact metric weighting is neutral if the confidentiality requirement is "medium." This same logic is applied to the integrity and availability requirements.

Note that the confidentiality requirement will not affect the environmental score if the (base) confidentiality impact is set to "none." Also, increasing the confidentiality requirement from "medium" to "high" will not change the environmental score when the (base) impact metrics are set to "complete." This is because the impact sub score (part of the base score that calculates impact) is already at a maximum value of 10.

The possible values for the security requirements are listed in Table 12. For brevity, the same table is used for all three metrics. The greater the security requirement, the higher the score (remember that "medium" is considered the default). These metrics will modify the score as much as plus or minus 2.5.

Metric Value	Description
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).
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).
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).
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.

Table 12

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 an Exploitable Scope may be to run a listening service as "root", for which compromises might grant an attacker Confidentiality, Integrity, and Availability impacts that are all High. Yet, in the analyst's environment, that same listening 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 "Not Defined" value.

Note: 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 in fact 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.

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)	

Table 13

Vector String

The v3.0 vector string is a string representation of a set of CVSS metrics. It is commonly used to record or transfer CVSS metric information in a concise form. 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 guide in the section defining the metrics (in parentheses after each metric name and metric value).

Metrics may be specified in any order in a vector string, though the order shown in Table YY is 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 **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 YY: Base, Temporal and Environmental Vectors

Abbreviated Metric Name	Possible Abbreviated Metric Values	Mandatory or Optional
AV	[N,A,L,P]	Mandatory
AC	[L,H]	Mandatory
PR	[N,L,H]	Mandatory
UI	[N,R]	Mandatory
S	[U,C]	Mandatory
C	[N,L,H]	Mandatory
I	[N,L,H]	Mandatory
A	[N,L,H]	Mandatory
E	[X,U,P,F,H]	Optional
RL	[X,O,T,W,U]	Optional
RC	[X,U,R,C]	Optional
CR	[X,L,M,H]	Optional
IR	[X,L,M,H]	Optional
AR	[X,L,M,H]	Optional
MAV	[X,N,A,L,P]	Optional

MAC	[X,L,H]	Optional
MPR	[X,N,L,H]	Optional
MUI	[X,N,R]	Optional
MS	[X,U,C]	Optional
MC	[X,N,L,H]	Optional
MI	[X,N,L,H]	Optional
MA	[X,N,L,H]	Optional

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 have 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 have 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

Acknowledgements

This preview, more specifically the effort it represents, would not have been possible without the tireless efforts of the CVSS-SIG, our many supporters in the community, in our workplaces, and at FIRST.

Additionally the CVSS SIG would like to thank the Deloitte & Touche team for their exceptional professionalism as they assisted in the development of the CVSS v3.0 formula.

The SIG would also like to thank Jenn Dailey @srslydesign.com for providing us with a new and web ready CVSS logo.

A full and proper list of acknowledgements and thanks will be produced for the final release of CVSS v3.0.