Calculate the CVSS base score for a given vulnerability

The common vulnerability scoring system (CVSS) is a way to assign scores to vulnerabilities on the basis of their principal characteristics. This score indicates the severity of a vulnerability and on that basis, it can be categorized into low, medium, high, and critical severity which can be used by the organization to prioritize the vulnerabilities present in the system.

The CVSS score ranges from **0.0 to 10.0**, where 1.0 is considered as least severe and 10.0 is the most severe. Mapping of CVSS score with qualitative ratings:

Base Score range	Severity
0.0	**None**
0.1 – 3.9	**Low**
4.0 – 6.9	**Medium**
7.0 – 8.9	**High**
9.0 – 10.0	**Critical**

Equation to calculate the Base score

Base score = round_to_1_decimal(((0.6*Impact)+(0.4*Exploitability)-1.5)*f(Impact))

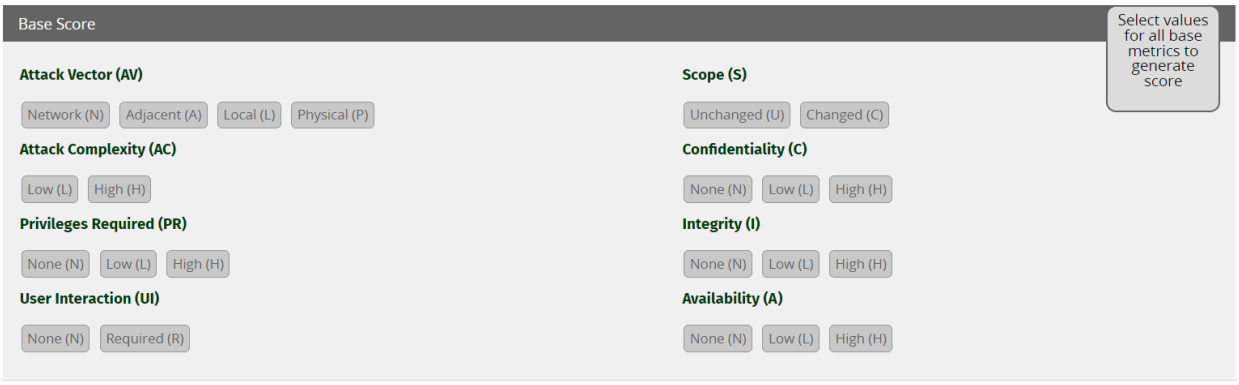
Impact = 10.41*(1-(1-ConfImpact)*(1-IntegImpact)*(1-AvailImpact))

Exploitability = 20*AccessVector*AccessComplexity*Authentication

f(Impact) = 0 if Impact = 0, 1.176 otherwise

Working

CVSS is broken down into 8 different metrics. These metrics values are used to calculate the CVSS score of a vulnerability. Let's take a look at each vector:



1. Attack Vector (AV)

This metric evaluates the potential scenario in which the vulnerability might be exploited. The more remote the attacker is, the larger will be the base score. It has 4 scenarios:

Network (N): When a vulnerability can be used remotely, i.e., through the open internet. (Metric Value 0.85)

Adjacent (A): when the victim and the malicious actor are both on the same intranet. (Metric Value 0.62)

Local (L): In order to exploit this vulnerability access to the internal network is required. (Metric Value 0.55)

Physical (P): When you have direct physical access to the victim's computer. (Metric Value 0.2)

2. Attack Complexity (AC)

This metric reflects how complex or simple it is to exploit the flaw, i.e., the least challenging attacks will receive the highest base scores.

Low (L): Executing the exploitation on target is quite simple. (Metric Value 0.77)

High (H): It is extremely difficult and has requirements that must be satisfied before the exploitation can be carried out. For instance, does your exploit need extra details about the target, such as specific configurations or settings, valid credentials (for MFA problems), or other requirements in order to function? Then in this case the Attack complexity will be high. (Metric Value 0.44)

3. Privileges Required (PR)

The metric identifies the amount of authority that an attacker must have in order to successfully exploit the vulnerability. It can have the following values:

None (N): When no privileges are required to exploit the vulnerability. (Metric Value 0.85)

Low (L): When the attacker can perform the exploit, with fewer privileges. (Metric Value 0.62)

High (H): When an attacker requires high-level privileges such as access to the admin panel to exploit the vulnerability. (Metric Value 0.27)

4. User Interaction (UI)

This metric reflects the necessity of a user—other than the attacker—taking part in the successful penetration of the vulnerable component.

None (N): The vulnerability can be exploited without any user interaction. (Metric Value 0.85)

Required (R): User interaction is required to exploit the vulnerability. (Metric Value 0.62)

5. Scope (S)

Scope indicates whether a successful attack impacts a component other than the vulnerable component.

Unchanged (U): When a vulnerability cannot affect the other resources.

Changed (C): When a vulnerability can affect the resources beyond the security controls.

6. Confidentiality (C)

This metric assesses how successfully exploiting a vulnerability affects the confidentiality of the information resources that software manages. And the Base Score is directly proportional to the loss of confidentiality.

None (N): No one can access the information other than the user.

Low (L): Your information is accessible to the general public, but not everyone can see it.

High (H): Your information is only visible to the organization's administrators.

7. Integrity (I)

Integrity metrics gauge how the exploited vulnerability affects the integrity of the data that the impacted components rely on. Information integrity refers to its dependability and authenticity.

None (N): If the exploited data is not altered.

Low (L): If the old data was obtained following some type of data alteration.

High (H): If all data is impacted, we risk losing the data's integrity.

8. Availability (A)

The impact on the target system's availability is described by the availability metric. Attacks that use up memory, CPU time, network bandwidth, or any other resource have an impact on a system's availability.

None (N): There is no issue with the availability of the resources and every user can access the application.

Low (L): Some users are not able to access the resources.

High (H): No one is able to access the resources and the server is down.

Example

Let us consider MySQL Stored SQL Injection (CVE-2013-0375) Vulnerability and now let us discuss how we can calculate the CVSS score of it.

Attack Vector (AV): Through a network connection, the attacker accesses the MySQL database that can be exploited. Therefore the value that will be selected is “Network”.

Attack Complexity (AC): The target database has to have replication turned on. That’s why the AC will be “low”.

Privileges Required (PR): An account having the capacity to alter user-provided identifiers, such as table names, is necessary for the attacker. This power is not granted to basic users by default, nor is it thought to be a trusted enough privilege to merit a High rating for this criteria. Therefore “low”.

User Interaction (UI): Replication occurs automatically; no user input is needed. Therefore the value will be “None”.

Scope (S): The MySQL server database, which the attacker logs into to carry out the attack, is the vulnerable part. A remote MySQL server database (or databases), to which this database replicates, is the component that is affected. That’s why the value will be “changed”.

Confidentiality (C): With privileges, the injected SQL can access data that shouldn't be accessible to the attacker. Therefore the value will be “low”.

Integrity (I): The injected SQL runs with privilege and can modify information the attacker should not have access to. That’s why the value selected will be “low”.

Availability (A): Despite the fact that injected code is executed with privileges, the nature of this attack prohibits arbitrary SQL statements from being executed that can impair MySQL databases' availability. Therefore the value will be “None”.

After selecting all the above values the Base Score is: