International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.1521
(03/2016)

SERIES X: DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

Cybersecurity information exchange – Vulnerability/state exchange

Common vulnerability scoring system 3.0

Recommendation ITU-T X.1521

ITU-T X-SERIES RECOMMENDATIONS

DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

PUBLIC DATA NETWORKS	X.1–X.199
OPEN SYSTEMS INTERCONNECTION	X.200–X.299
INTERWORKING BETWEEN NETWORKS	X.300–X.399
MESSAGE HANDLING SYSTEMS	X.400–X.499
DIRECTORY	X.500–X.599
OSI NETWORKING AND SYSTEM ASPECTS	X.600–X.699
OSI MANAGEMENT	X.700–X.799
SECURITY	X.800–X.849
OSI APPLICATIONS	X.850–X.899
OPEN DISTRIBUTED PROCESSING	X.900–X.999
INFORMATION AND NETWORK SECURITY	
General security aspects	X.1000–X.1029
Network security	X.1030–X.1049
Security management	X.1050–X.1069
Telebiometrics	X.1080–X.1099
SECURE APPLICATIONS AND SERVICES	
Multicast security	X.1100–X.1109
Home network security	X.1110–X.1119
Mobile security	X.1120–X.1139
Web security	X.1140–X.1149
Security protocols	X.1150–X.1159
Peer-to-peer security	X.1160–X.1169
Networked ID security	X.1170–X.1179
IPTV security	X.1180–X.1199
CYBERSPACE SECURITY	
Cybersecurity	X.1200–X.1229
Countering spam	X.1230–X.1249
Identity management	X.1250–X.1279
SECURE APPLICATIONS AND SERVICES	
Emergency communications	X.1300–X.1309
Ubiquitous sensor network security	X.1310–X.1339
PKI related Recommendations	X.1340–X.1349
CYBERSECURITY INFORMATION EXCHANGE	
Overview of cybersecurity	X.1500–X.1519
Vulnerability/state exchange	**X.1520–X.1539**
Event/incident/heuristics exchange	X.1540–X.1549
Exchange of policies	X.1550–X.1559
Heuristics and information request	X.1560–X.1569
Identification and discovery	X.1570–X.1579
Assured exchange	X.1580–X.1589
CLOUD COMPUTING SECURITY	
Overview of cloud computing security	X.1600–X.1601
Cloud computing security design	X.1602–X.1639
Cloud computing security best practices and guidelines	X.1640–X.1659
Cloud computing security implementation	X.1660–X.1679
Other cloud computing security	X.1680–X.1699

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T X.1521

Common vulnerability scoring system 3.0

Summary

Recommendation ITU-T X.1521 on the common vulnerability scoring system (CVSS) provides an open framework for communicating the characteristics and impacts of information and communication technologies (ICT) vulnerabilities in the commercial or open source software used in communications networks, end user devices, or any of the other types of ICT capable of running software. The goal of the Recommendation is to enable ICT managers, vulnerability bulletin providers, security vendors, application vendors and researchers to speak from a common language of scoring ICT vulnerabilities.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T X.1521	2011-04-20	17	11.1002/1000/11062
2.0	ITU-T X.1521	2016-03-23	17	11.1002/1000/12614

Keywords

Common vulnerability scoring system, CVSS, CYBEX, , metrics.

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

© ITU 2016

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

Table of Contents

Introduction

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 Recommendation provides the official specification for CVSS v3.0.

CVSS v3.0 represents radical improvements over CVSS v2.0 and is not backward compatible with it. During the use of CVSS v2.0, several limitations of that specification came to light. Some of them are: scoring vulnerabilities in virtual environment, representing "indirect" vulnerabilities like cross site scripting, lack of ability to capture interdependencies between applications within the same system and capturing actions of a user other than attacker. More information of v3.0 improvements is given in Appendix I, clause 2.

While working to address these limitations, CVSS working group realised that it is not possible to maintain backward compatibility with the CVSS v2.0. While acknowledging that lack of compatibility will cause some issues with existing systems that use and process CVSS v2.0, it is our belief that version 3.0 brings sufficient value that would compensate for the inconvenience. We are strongly advising users and vendors that are currently producing and processing CVSS v2.0 to migrate to CVSS v3.0.

While the CVSS v2.0 specification will continue to be available for historic purposes it will cease to be in force. Implementers of tools and processes are strongly encouraged to adopt the CVSS v3.0 specification while keeping support for v2.0 in order to process existing vulnerabilities already scored with the v2.0 specification.

Recommendation ITU-T X.1521

Common vulnerability scoring system 3.0

1 Scope

This Recommendation provides a standardized approach for communicating the characteristics and impacts of ICT vulnerabilities using temporal and environmental metrics that apply contextual information to more accurately reflect the risk to each user's unique environment.

This Recommendation is technically equivalent and compatible with the "Common Vulnerability Scoring System (CVSS) version 3", 10 June 2015 which can be found at the website http://www.first.org/cvss

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

None.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following term defined elsewhere:

3.1.1 vulnerability [b-ITU-T X.1500]: Any weakness that could be exploited to violate a system or the information it contains.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 access: A subject's ability to view, modify, or communicate with an object. Access enables the flow of information between the subject and the object.

3.2.2 availability: The reliable and timely access to data and resources by authorized individuals.

3.2.3 confidentiality: A security principle that works to ensure that information is not disclosed to unauthorized subjects.

3.2.4 integrity: A security principle that makes sure that information and systems are not modified maliciously or accidentally.

3.2.5 risk: The relative impact that an exploited vulnerability would have to a user's environment.

3.2.6 threat: The likelihood or frequency of a harmful event occurring.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

A	Availability Impact
AC	Attack Complexity
AR	Availability Requirement
ARP	Address Resolution Protocol
AV	Attack Vector
C	Confidentiality Impact
CIA	Confidentiality, Integrity and Availability
CPU	Central Processing Unit
CR	Confidentiality Requirement
CVE	Common Vulnerability Exposure
CVSS	Common Vulnerability Scoring System
CWE	Common Weakness Enumeration
DMA	Direct Memory Access
DNS	Domain Name System
DOM	Document Object Model
DoS	Denial-of-Service
E	Exploit code maturity
I	Integrity impact
ICT	Information and Communication Technologies
ID	Identifier
IP	Internet Protocol
IR	Integrity Requirement
ISC	Impact Sub Score
IT	Information Technology
LAN	Local Area Network
MA	Modified Availability
MAC	Modified Attack Complexity
MAV	Modified Attack Vector
MC	Modified Confidentiality impact
MI	Modified Integrity
MPR	Modified Privileges Required
MS	Modified Scope
MUI	Modified User Interaction
NIST	National Institute of Standards
OS	Operating System

OSI	Open Systems Interconnection
PCI DSS	Payment Card Industry Data Security Standard
PR	Privileges Required
RC	Report Confidence
RL	Remediation Level
RPC	Remote Procedure Call
S	Scope
SCAP	Security Content Automation Protocol
SQL	Structured Query Language
TCP	Transmission Control Protocol
UI	User Interaction
USB	Universal Serial Bus
VM	Virtual Machine
XSS	Cross Site Scripting

5 Conventions

None.

6 About common vulnerability scoring system

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 Recommendation provides the official specification for CVSS v3.0.

6.1 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 Recommendation describes the official CVSS v3.0 specification.

6.1.1 Metrics

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

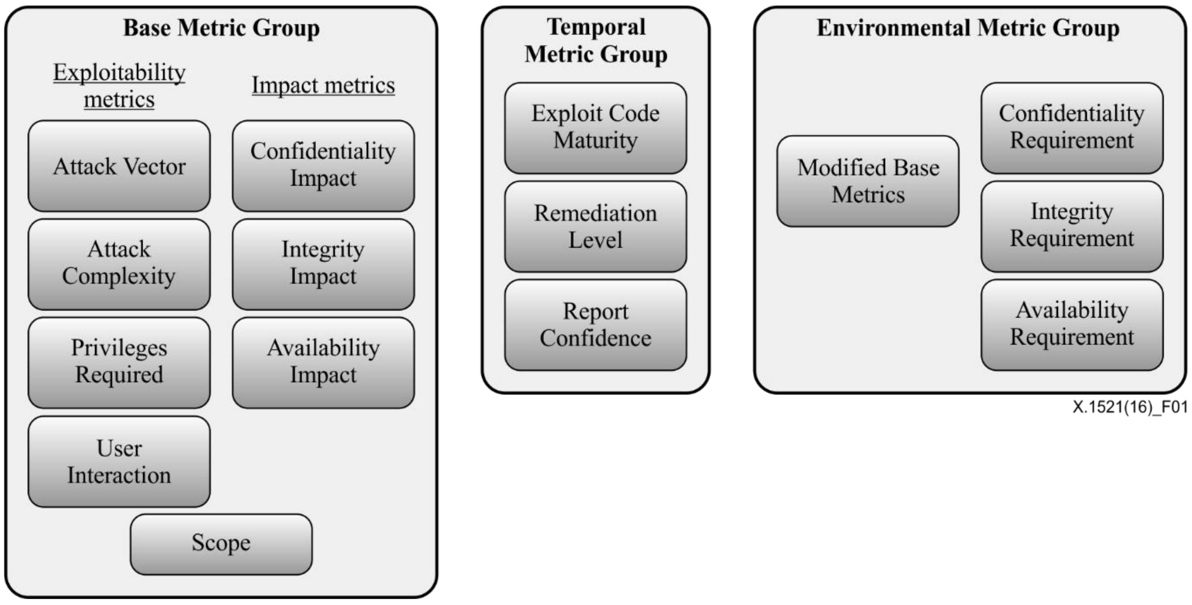


Figure 1 – CVSS v3.0 metric groups

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.

6.1.2 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.

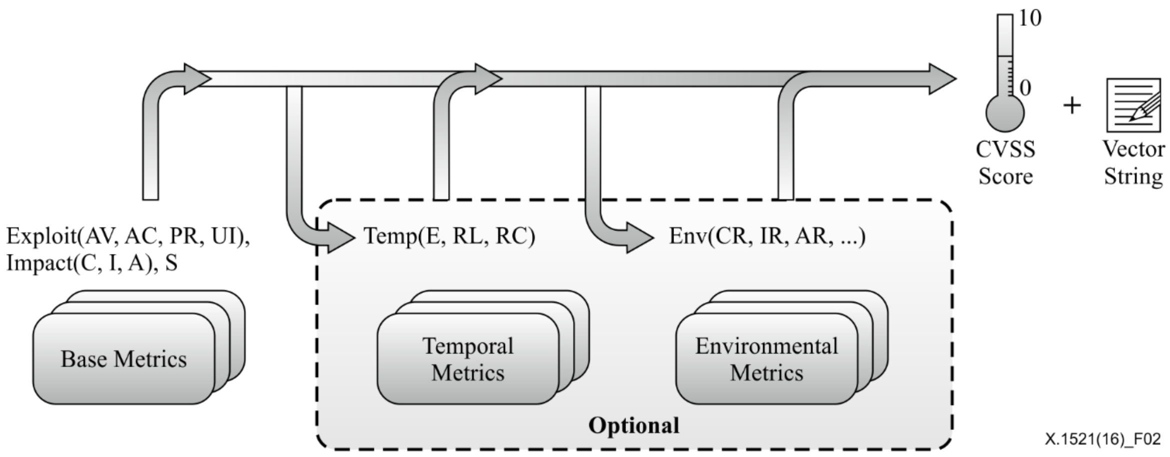


Figure 2 – CVSS metrics and equations

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.

6.2 Base metrics

6.2.1 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.

6.2.1.1 Attack vector (AV)

This metric reflects the context by which vulnerability exploitation is possible. This metric value (and consequently the Base score) will be greater the more remote an attacker is (in terms of logical and physical distance) 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 open systems interconnection (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 transmission control protocol (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 Internet protocol (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 address resolution protocol (ARP) (IPv4) or neighbour discovery (IPv6) flood leading to a denial of service on the local area network (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, he or she may rely on user interaction (UI) 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 access to disk encryption keys after gaining physical access to the system, or peripheral attacks such as Firewire/universal serial bus (USB) direct memory access attacks.

6.2.1.2 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.

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

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 himself or 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).

6.2.1.3 Privileges required (PR)

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

6.2.1.4 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.

[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.

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.

6.2.2 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, central processing unit (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 operating system (OS) (perhaps even its own virtual machine (VM)). In this example, there are two separate authorization authorities: one that defines and enforces privileges for the virtual machine and its users, and the other 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.

6.2.3 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.

6.2.3.1 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.

6.2.3.2 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.

6.2.3.3 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 only a 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.

6.3 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.

6.3.1 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.

Table 9 – Exploit code maturity

Metric value	Description
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.

6.3.2 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.

6.3.3 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.

6.4 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.

6.4.1 Security requirements (CR, IR, AR)

These metrics enable the analyst to customize the CVSS score depending on the importance of the affected information technology (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).

6.4.2 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)	

6.5 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.

6.6 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.

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

[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.

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

6.7 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

6.8 CVSS v3.0 equations

The CVSS v3.0 equations are defined below.

6.8.1 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] $Roundup(Minimum[(Impact + Exploitability), 10])$

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

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

Scope Unchanged $6.42 \times ISC_{Base}$

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

Where,

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

And the Exploitability sub score is,

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

6.8.2 Temporal

The Temporal score is defined as,

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

6.8.3 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)*

[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.

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,

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

The Modified Exploitability sub score is,

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

6.8.4 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

Table 16 – Metric values

Metric	Metric value	Numerical value
	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

6.8.5 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.

Appendix I

CVSS v3.0 user guide

(This appendix does not form an integral part of this Recommendation.)

I.1 Introduction

This guide supplements the formal CVSS v3.0 specification document by providing additional information, highlighting relevant changes from v2.0, as well as providing scoring guidance and a scoring rubric.

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.

CVSS affords three important benefits:

− 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.

− 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.

− CVSS helps prioritize 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 a vulnerability to the organization.

Since its initial release in 2004, CVSS has enjoyed widespread adoption. In September 2007, CVSS v2.0 was adopted as part of the Payment Card Industry Data Security Standard (PCI DSS). In order to comply with PCI DSS, merchants processing credit cards must demonstrate that none of their computing systems has a vulnerability with a CVSS score greater than or equal to 4.0. In 2007 National Institute of Standards (NIST) included CVSS v2.0 as part of their security content automation protocol (SCAP).[5] In April 2011, CVSS v2.0 was formally adopted as an international standard for scoring vulnerabilities (ITU-T X.1521).[6]

I.2 Changes in CVSS v3.0

Given the widespread adoption of CVSS v2.0, a number of opportunities for improvement had been identified, prompting the development of v3.0. These are described in detail below.

I.2.1 Scope, Vulnerable component and Impacted component

CVSS v2.0 presented difficulties for vendors when scoring vulnerabilities that would fully compromise their software, but only partially affect the host operating system. In v2.0 vulnerabilities are scored relative to the host operating system, which led one application vendor to adopt a "Partial+" impact metric convention.[7] CVSS v3.0 addresses this issue with updates to where the impact metrics

---

5 See http://scap.nist.gov/.

6 See https://www.itu.int/rec/T-REC-X.1521-201104-I/en.

7 For example, see http://www.oracle.com/technetwork/topics/security/cvssscoringsystem-091884.html.

are scored and a new metric called Scope (discussed further below). Therefore, an important conceptual change in CVSS v3.0 is the ability to score vulnerabilities that exist in one software component (that we refer to formally as the *vulnerable component*) but which impact a separate software, hardware, or networking component (that we refer to formally as the *impacted component*), as illustrated in Figure I.1.[8]

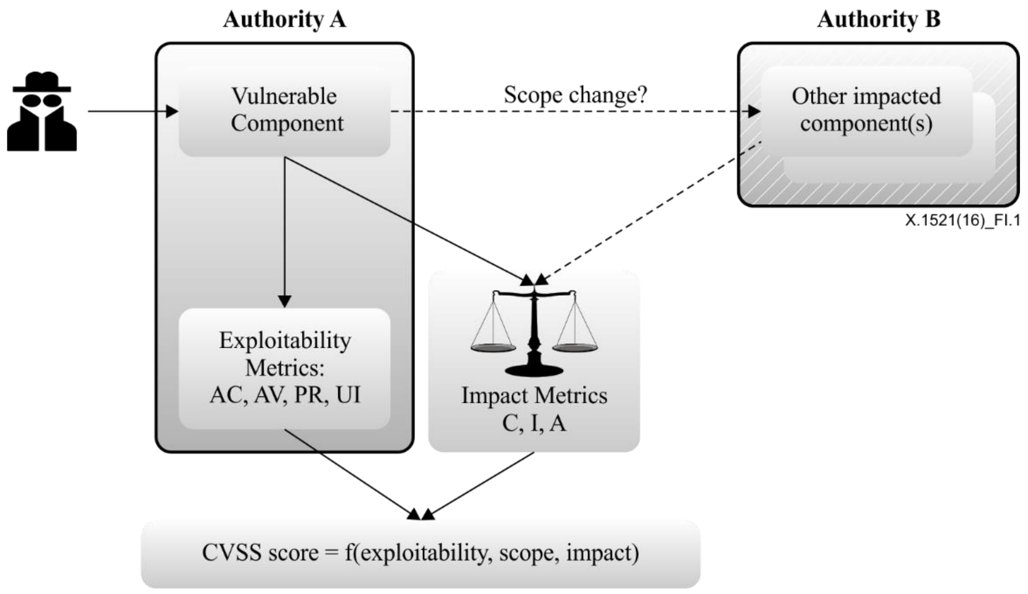

Authority A
Authority B
Vulnerable Component
Scope change?
Other impacted component(s)
X.1521(16)_FI.1
Exploitability Metrics: AC, AV, PR, UI
Impact Metrics C, I, A
CVSS score = f(exploitability, scope, impact)

Figure I.1 – Scope change

For example, consider a vulnerability in a virtual machine that compromises the host operating system. The vulnerable component is the virtual machine, while the impacted component is the host operating system. Because these two components independently manage privileges to computing resources, they therefore represent separate (authorization) authorities. In Figure I.1, the virtual machine is managed by "Authority A," while the host OS is managed by "Authority B." When two authorities are involved in a vulnerability exploit, CVSS considers that a *scope change* has occurred. This condition is captured by the new metric, Scope.

As depicted in Figure I.1, when scoring vulnerabilities in CVSS v3.0, the Exploitability metrics are scored relative to the vulnerable component. That is, they are scored by considering the component that suffers the coding flaw. On the other hand, the Impact metrics are scored relative to the impacted component. In some cases, the vulnerable component may be the same as the impacted component, in which case, no scope change has occurred. However, in other cases, there may be an impact to the vulnerable component, as well as to the impacted component. In these cases, a scope change has occurred, and the Confidentiality, Integrity and Availability Impact metrics should reflect the impact to either the vulnerable component, or the impacted component, whichever is most severe.

In the case of a vulnerability that allows the theft of a password file, while there may be subsequent steps the attacker takes to commit unauthorized account access, the most direct outcome is a loss of confidentiality of the local system file. As such, there would be no scope change. However, in the case of a vulnerability that allows a router's ARP table to be overwritten by an attacker, there are two impacts. First, to the router's system file (Integrity impact to the vulnerable component), and second, to those Internet services served by the router (Availability impact to affected systems). Because the score should reflect the most severe outcome, the impact metric score may reflect either the Integrity

[8] Note that while the vulnerable component will be a software program (host operating system, Internet application, device driver, etc.,) the impacted component may be either another software program, a hardware device, or a network resource.

loss to the vulnerable component, or the Availability loss to the Internet services, whichever is more severe.[9]

I.2.2 Access vector

The Access Vector (from v2.0) has been renamed to Attack Vector, but still generally reflects the "remoteness" of the attacker relative to the vulnerable component. That is, the more remote an attacker is to the vulnerable component (in terms of logical and physical network distance), the greater the Base score will be. Further, this metric now distinguishes between local attacks which require local system access (such as with an attack against a desktop application) and physical attacks which require physical access to the platform in order to exploit a vulnerability (such as with a firewire, USB, or jailbreaking attack).

I.2.3 Attack complexity

Access complexity (from v2.0) conflated two issues: any software, hardware, or networking condition beyond the attacker's control that must exist or occur in order for the vulnerability to be successfully exploited (for example a software race condition, or application configuration), and the requirement for human interaction (for example, requiring a user execute a malicious executable). Therefore, Access complexity has been separated into two metrics, Attack complexity (which addresses the former condition) and **User interaction** (which addresses the latter condition).

I.2.4 Privileges required

The new metric, **Privileges required,** replaces the Authentication metric of v2.0. Instead of measuring the *number of times* an attacker must separately authenticate to a system, Privileges Required captures the *level of access* required for a successful attack. Specifically, the metric values High, Low, and None reflect the privileges required by an attacker in order to exploit the vulnerability.

I.2.5 Impact metrics

The **Confidentiality, Integrity and Availability impact** metric values from v2.0 of None, Partial, and Complete have been replaced with None, Low and High. Rather than representing the overall *percentage* (proportion) of the systems impacted by an attack, the new metric values reflect the overall *degree* of impact caused by an attack. For example, while the Heartbleed[10] vulnerability only caused a loss to a small amount of information, the impact was quite severe. In CVSS v2.0, this would have been scored as Partial, while in CVSS v3.0, this is appropriately scored as High.

Additionally, in the example above, the impact metrics now reflect the consequence to the impacted component. And the impacted component may or may not be the same as the component that possesses the vulnerability being exploited.

I.2.6 Temporal metrics

The influence of Temporal metrics has been reduced in v3.0, relative to v2.0. Exploitability has been renamed Exploit code maturity to better represent what the metric is measuring.

I.2.7 Environmental metrics

The environmental metrics target distribution and collateral damage potential have been replaced by modified factors which accommodates mitigating controls or control weaknesses that may exist within the user's environment that could reduce or raise the impact of a successfully exploited vulnerability.

---

9 See the Examples document which accompanies this guide for more information.

10 See http://heartbleed.com/.

I.2.8 Qualitative rating scale

Some organizations created systems to map CVSS v2.0 Base scores to qualitative ratings. CVSS v3.0 now provides a standard mapping from numeric scores to the severity rating terms None, Low, Medium, High and Critical, as explained in the CVSS v3.0 specification document. The use of these qualitative severity ratings is optional, and there is no requirement to include them when publishing CVSS scores.

Organizations using CVSS v3.0 scores that wish to use an *alternate* severity rating system are asked to use different rating terms or to clearly state that their ratings do not comply with the CVSS v3.0 specification, to avoid confusion.

I.2.9 Summary of changes

An important consequence of these changes is that v2.0 and v3.0 scores may not always be comparable. For example, a vulnerable application that could result in its complete compromise would have been scored in v2.0 with Confidentiality, Integrity and Availability impact metric values of Partial. Whereas in v3.0, this same vulnerability would now be scored with the equivalent Confidentiality, Integrity and Availability impact metric values of High.

A summary of changes from v2.0 are presented in Table I.1.

Table I.1 – CVSS v2.0 to v3.0 changes

Version 2.0	Version 3.0
Vulnerabilities are scored relative to the overall impact to the host platform.	Vulnerabilities now scored relative to the impact to the impacted component.
No awareness of situations in which a vulnerability in one application impacted other applications on the same system.	A new metric, Scope, now accommodates vulnerabilities where the *thing suffering the impact* (the impacted component) is different from *the thing that is vulnerable* (the vulnerable component).
Access vector may conflate attacks that require local system access and physical hardware attacks.	Local and physical values are now separated in the Attack vector metric.
In some cases, Access complexity conflated system configuration and user interaction.	This metric has been separated into Attack complexity (accounting for system complexity) and User interaction (accounting for user involvement in a successful attack).
In practice, the Authentication metric scores were biased toward two of three possible outcomes, and not effectively capturing the intended aspect of a vulnerability.	A new metric, Privileges required, replaces Authentication, and now reflects the greatest privileges required by an attacker, rather than the number of times the attacker must authenticate.
Impact metrics reflected percentage of impact caused to a vulnerable application.	Impact metric values now reflect the degree of impact, and are renamed to None, Low and High.
The Environmental metrics of Target distribution and Collateral Damage potential were not found to be useful.	Target distribution and Collateral damage potential have been replaced with Mitigating factors.
CVSS v2.0 could not accommodate scoring multiple vulnerabilities used in the same attack.	While not a formal metric, guidance on scoring multiple vulnerabilities is provided with Vulnerability chaining.
No formal qualitative scoring guidelines were provided.	Numerical ranges have been mapped to a 5-point qualitative rating scale.

I.3 Scoring guide

Below are a number of suggestions for analysts when scoring vulnerabilities with CVSS v3.0.

I.3.1 CVSS scoring in the exploit lifecycle

When understanding when to score the impact of vulnerabilities, analysts should constrain impacts to a reasonable final impact which they are confident an attacker is able to achieve. Ability to cause this impact should be supported by the Exploitability sub score as a minimum, but may also include details from the vulnerability's description. For example, consider the following two vulnerabilities:

In vulnerability 1, a remote, unauthenticated attacker can send a trivial, crafted request to a web server which causes the web server to disclose the plaintext password of the root (administrator) account. The analyst only knows from the Exploitability sub score metrics and the vulnerability description that the attacker has access to send a crafted request to the web server in order to exploit the vulnerability. Impact should stop there; while an attacker *may* be able to use these credentials to later execute code as the administrator, it is not known that the attacker has access to a login prompt or method to execute commands with those credentials. Gaining access to this password represents a direct, serious loss of Confidentiality only:

– Base score: 7.5 [CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N].

In vulnerability 2, a local, low-privileged user can send a trivial, crafted request to the operating system which causes it to disclose the plaintext password of the root (administrator) account. The analyst knows from the Exploitability sub score metrics and the vulnerability description that the attacker has access to the operating system, and can log in as a local, low privileged attacker. Gaining access to this password represents a direct, serious loss of Confidentiality, Integrity and Availability because the analyst can reasonably issue commands as the root / administrator account (assume that the attacker could log out from her own account and log back in as root):

– Base score: 7.8 [CVSS:3.0/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H].

I.3.2 Confidentiality and integrity, versus Availability impacts

The Confidentiality and Integrity metrics refer to impacts that affect the *data* used by the service. For example, web content that has been maliciously altered, or system files that have been stolen. The Availability impact metric refers to the *operation* of the service. That is, the Availability metric speaks to the performance and operation of the service itself – not the availability of the data. Consider a vulnerability in an Internet service such as web, email, or domain name system (DNS) that allows an attacker to modify or delete all web files in a directory would incur an impact to Integrity only, rather than Availability. The reason is that the web service is still performing properly – it just happens to be serving back altered content.

I.3.3 Local vulnerabilities exploited by remote attackers

In CVSS v2.0, Scoring Tip 5 stated: "When a vulnerability can be exploited both locally and from the network, the Network value should be chosen. When a vulnerability can be exploited both locally and from adjacent networks, but not from remote networks, the Adjacent Network value should be chosen. When a vulnerability can be exploited from the adjacent network and remote networks, the Network value should be chosen." This guidance sometimes led to confusion in cases where an attacker would trick a user into downloading a malformed document from a remote web server, exploiting a file parsing vulnerability. In such case, analysts using CVSS v2.0 would treat these vulnerabilities as "network," producing scores with metric strings of:

– AV:N/AC:M/Au:N/C:P/I:P/A:P, or AV:N/AC:M/Au:N/C:C/I:C/A:C.

This guidance has been improved in CVSS v3.0 by clarifying the definitions of the Network and Adjacent values of the Attack vector metric. Specifically, analysts should only score for Network or Adjacent when a vulnerability is bound to the network stack. Vulnerabilities which require user interaction to download or receive malicious content (which could also be delivered locally, e.g., via USB drives) should be scored as Local.

For example, a document parsing vulnerability, which does not rely on the network in order to be exploited, should typically be scored with the Local value, regardless of the method used to distribute such a malicious document (e.g., it could be a link to a web site, or via a USB stick).

I.3.4 Cross site scripting vulnerabilities

In CVSS v2.0, specific guidance was necessary to produce non-zero scores for cross-site scripting (XSS) vulnerabilities, because vulnerabilities were scored relative to the host operating system that contained the vulnerability. A typical XSS vulnerability produced a score which described a partial integrity impact due to modification of the web server's response to the client: AV:N/AC:M/Au:N/C:N/I:P/A:N. This persisted even for document object model (DOM)-based cross site scripting (XSS) vulnerabilities which, while they may be triggered by interaction with the server, are exploited entirely at the client-side (e.g., when server-delivered JavaScript parses the request string sent to the server).

This is one of the key scenarios for which Scope was designed – where impacts are suffered not by the vulnerable component (e.g., the web server, or the JavaScript delivered by the web server), but by a component whose privileges are managed by a separate authority (e.g., the client's browser environment). Therefore, under CVSS v3.0, cross-site scripting vulnerabilities do not have to be constrained to the limited or non-existent impacts to the server, and can now be scored for impacts that are realized at the client. A reflected XSS vulnerability that allowed an attacker to deliver a malicious link to a victim and execute JavaScript in their browser might be scored:

– CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N

I.3.5 Man in the middle

CVSS v3.0 now explicitly accommodates scoring man-in-the-middle attacks. While not specifically addressed in v2.0, in v3.0, this type of attack is addressed with the Attack complexity metric.

I.3.6 Hardware vulnerabilities

In addition, while CVSS is primarily designed for scoring vulnerabilities and impacts to software, v3.0 is now better suited for also scoring impacts that include hardware components and networking effects.

I.3.7 Vulnerability chaining

CVSS is designed to classify and rate individual vulnerabilities. However, it is important to support the needs of the vulnerability analysis community by accommodating situations where multiple vulnerabilities are exploited in the course of a single attack to compromise a host or application. The scoring of multiple vulnerabilities in this manner is termed Vulnerability chaining. Note that this is not a formal metric, but is included as guidance for analysts when scoring these kinds of attacks.

When scoring a chain of vulnerabilities, it is the responsibility of the analyst to identify which vulnerabilities are combined to form the chained score. The analyst should list the distinct vulnerabilities and their scores, along with the chained score. For example, this may be communicated within a vulnerability disclosure notice posted on a webpage.

In addition, the analyst may include other types of related vulnerabilities that could be chained with the vulnerabilities being scored. Specifically, the analyst may list generic types (or classes) of related vulnerabilities that are often chained together, or provide further descriptions of required preconditions that must exist. For example, one might describe how certain kinds of structured query language (SQL) Injection vulnerabilities are precursors to a cross-site scripting (XSS) attack, or how a particular kind of buffer overflow would grant local privileges. Listing the generic types or classes of vulnerabilities provides the minimum information necessary to warn other users, without potentially informing attackers about new exploit opportunities.

Alternatively, the analyst may identify (in the form of a machine readable and parseable list of vulnerabilities as common vulnerability exposure (CVE) identifiers (IDs) or common weakness enumeration (CWEs)) a complete list of specific related vulnerabilities that are known to be (or are very likely to be) chained to one or more of the chained vulnerabilities being scored in order to exploit an IT system. In the event that a vulnerability can be exploited only after other preconditions are met (such as first exploiting another vulnerability), it is acceptable to combine two or more CVSS scores to describe the chain of vulnerabilities by scoring for the least-restrictive Exploitability sub score metrics and scoring for the most-impactful Impact sub score metrics. The following example uses the Exploitability, Scope and Impact sub scores to describe the chain:

Vulnerability A is: AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H, and as can be seen from the vector, requires a local, low-privileged user in order to exploit. Whereas Vulnerability B is, AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:L which provides an unprivileged, remote attacker the ability to execute code on a system with Low impacts if a local user interacts to complete the attack. Therefore, given both A & B, Chain C could be described as the chain of B -> A: AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H which combines the Exploitability of B, the scope is unchanged in both cases, and the Impact of A, because if one can exploit B and gain the code execution as a local user from it, then one has satisfied the prerequisite to subsequently launch A causing an impact from vulnerability A.

I.4 Glossary of terms

Authority: A computing container that grants and manages privileges to resources. Examples of authorities include, a database application, an operating system and a sandbox environment.

Chained score: The Base score produced by scoring two or more chained vulnerabilities.

Chained vulnerabilities: See Vulnerability Chaining.

Component: Refers to either a software or hardware component.

Software component: A software program or module that contains computer instructions to be executed. E.g., an operating system, Internet application, device driver.

Hardware component: A physical computing device.

Impacted component: The component (or components) that suffer(s) the consequence of the exploited vulnerability. This (they) can either be the same component as the vulnerable component, or, if a scope changed has occurred, a different one.

Privileges: A collection of rights (typically read, write and execute) granted to a user or user process which defines access to computing resources.

Resources: A software or network object that is accessed, modified, or consumed by a computing device. E.g., computer files, memory, CPU cycles, or network bandwidth.

Scope: The collection of privileges defined and managed by an authorization authority when granting access to computing resources.

Vulnerability: A weakness or flaw in a software (or hardware) component.

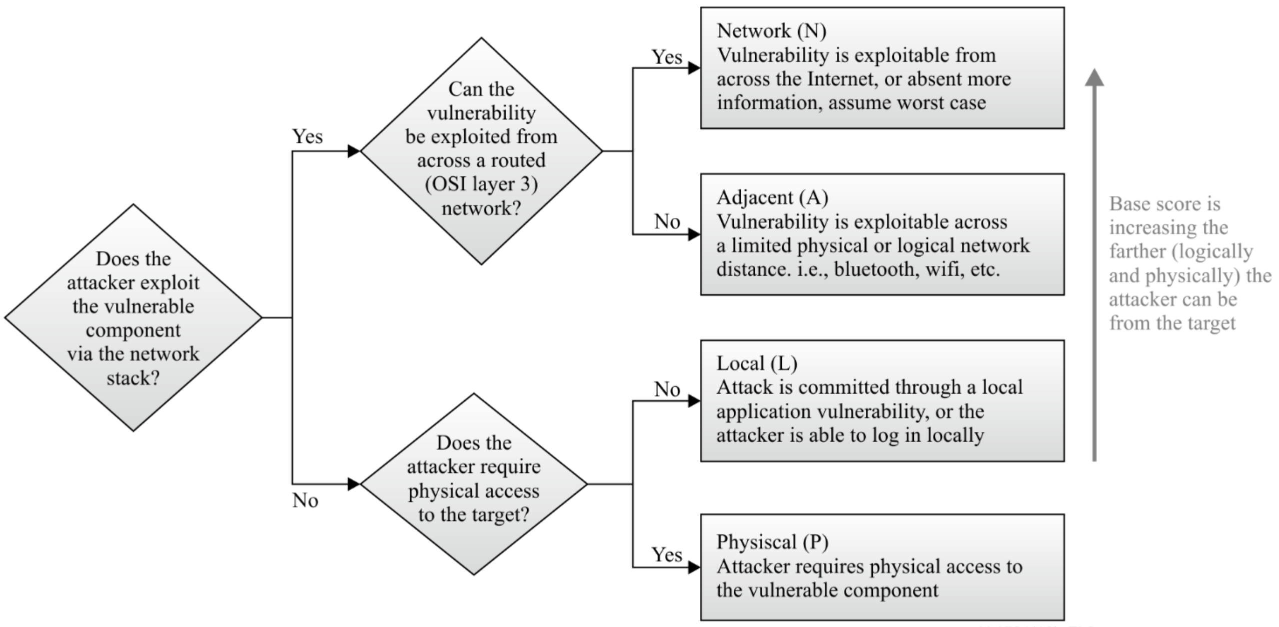
Vulnerability chaining: The sequential exploit of multiple vulnerabilities in order to attack an IT system, where one or more exploits at the end of the chain require the successful completion of prior exploits in order to be exploited. See also the definition available at http://cwe.mitre.org/documents/glossary/#Chain.

Vulnerable component: The software (or hardware) component that bears the vulnerability, and that which would be patched.

I.5 Scoring rubric

The scoring rubric provides a quick reference to scoring vulnerabilities in v3.0. It is meant to supplement existing scoring discussion found in the Specification document.

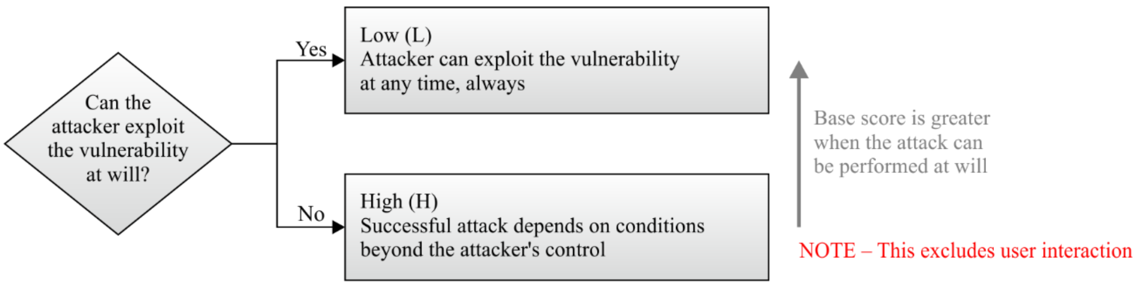
I.5.1 Attack vector



Does the attacker exploit the vulnerable component via the network stack?
Yes
No
Can the vulnerability be exploited from across a routed (OSI layer 3) network?
Yes
No
Does the attacker require physical access to the target?
No
Yes
Network (N) Vulnerability is exploitable from across the Internet, or absent more information, assume worst case
Adjacent (A) Vulnerability is exploitable across a limited physical or logical network distance. i.e., bluetooth, wifi, etc.
Local (L) Attack is committed through a local application vulnerability, or the attacker is able to log in locally
Physical (P) Attacker requires physical access to the vulnerable component
Base score is increasing the farther (logically and physically) the attacker can be from the target

X.1521(16)_FI.2

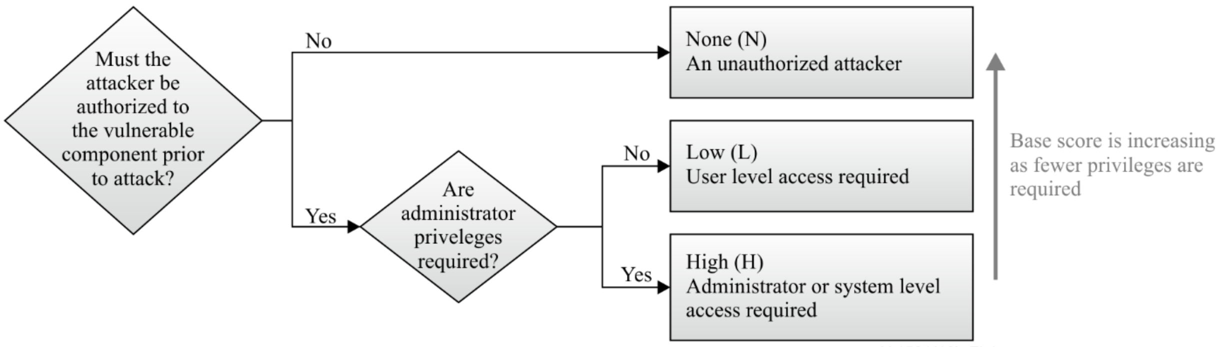
I.5.2 Attack complexity



Can the attacker exploit the vulnerability at will?
Yes
No
Low (L) Attacker can exploit the vulnerability at any time, always
High (H) Successful attack depends on conditions beyond the attacker's control
Base score is greater when the attack can be performed at will
NOTE – This excludes user interaction

X.1521(16)_FI.3

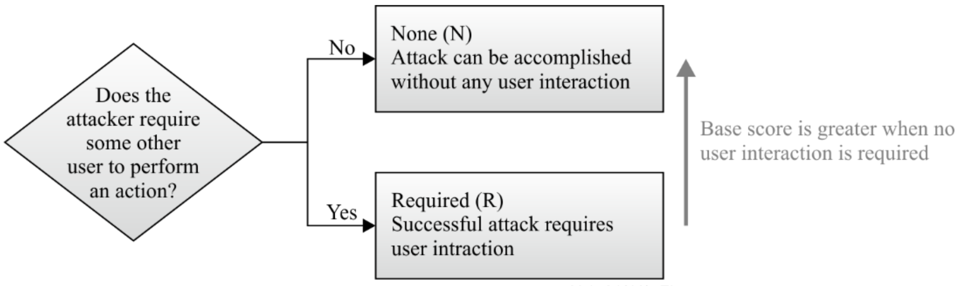
I.5.3 Privileges required



Must the attacker be authorized to the vulnerable component prior to attack?
No
Yes
Are administrator priveleges required?
No
Yes
None (N) An unauthorized attacker
Low (L) User level access required
High (H) Administrator or system level access required
Base score is increasing as fewer privileges are required

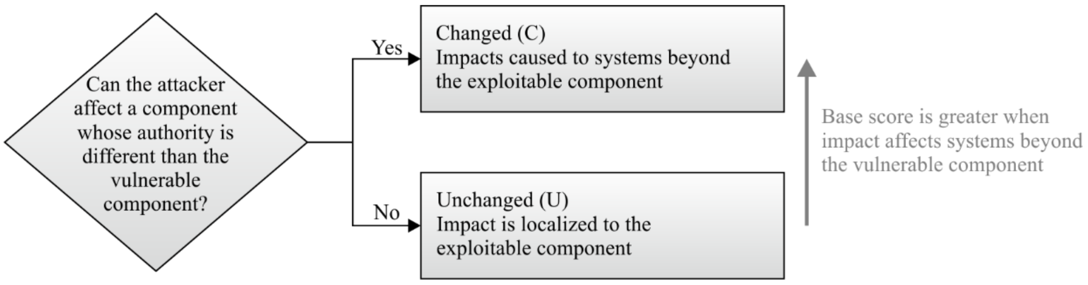
X.1521(16)_FI.4

I.5.4 User interaction


Does the attacker require some other user to perform an action?
No
None (N) Attack can be accomplished without any user interaction
Yes
Required (R) Successful attack requires user intraction
Base score is greater when no user interaction is required

X.1521(16)_FI.5

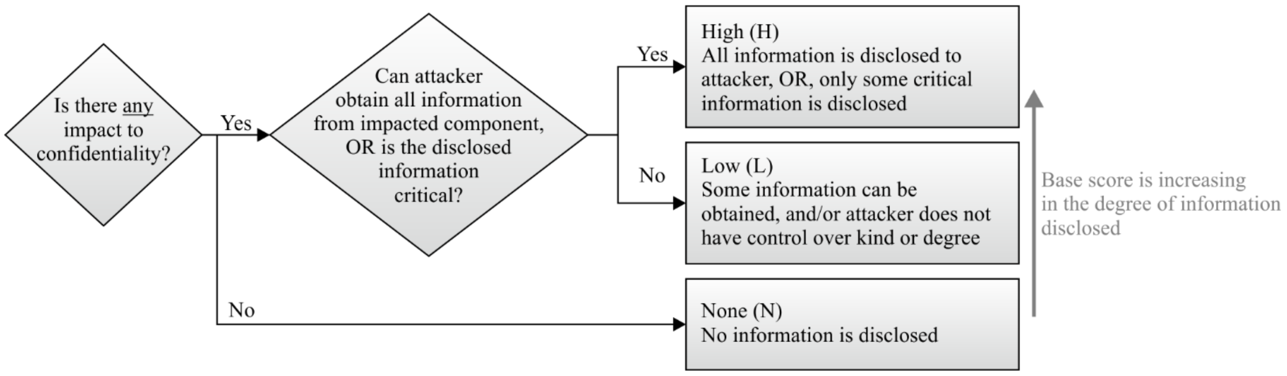
I.5.5 Scope


Can the attacker affect a component whose authority is different than the vulnerable component?
Yes
Changed (C) Impacts caused to systems beyond the exploitable component
No
Unchanged (U) Impact is localized to the exploitable component
Base score is greater when impact affects systems beyond the vulnerable component

X.1521(16)_FI.6

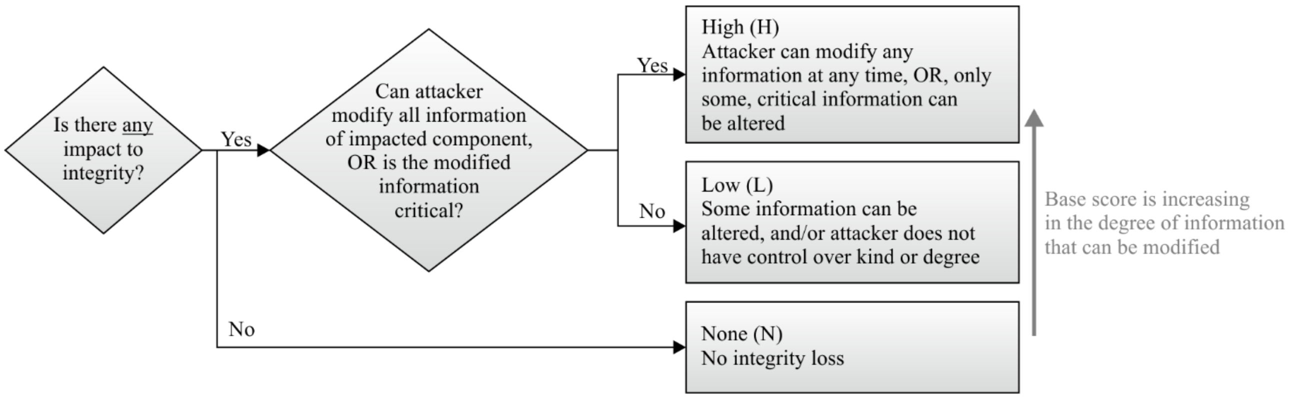
NOTE – If Scope change has not occurred, Confidentiality, Integrity and Availability impacts reflect consequence to the vulnerable component, otherwise they reflect consequence to the component that suffers the greater impact.

I.5.6 Confidentiality impact


Is there any impact to confidentiality?
Yes
Can attacker obtain all information from impacted component, OR is the disclosed information critical?
Yes
High (H) All information is disclosed to attacker, OR, only some critical information is disclosed
No
Low (L) Some information can be obtained, and/or attacker does not have control over kind or degree
No
None (N) No information is disclosed
Base score is increasing in the degree of information disclosed

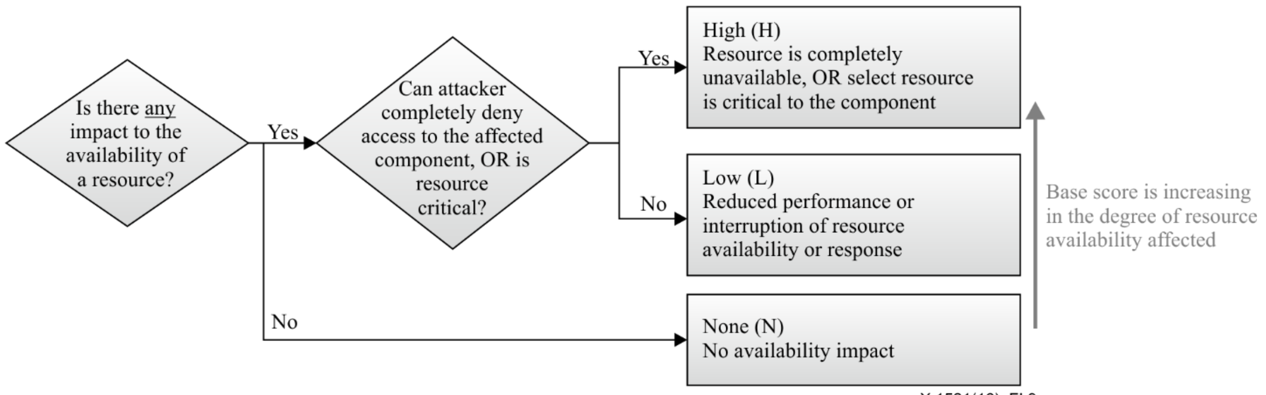
X.1521(16)_FI.7

I.5.7 Integrity impact


Is there any impact to integrity?
Yes
Can attacker modify all information of impacted component, OR is the modified information critical?
Yes
High (H) Attacker can modify any information at any time, OR, only some, critical information can be altered
No
Low (L) Some information can be altered, and/or attacker does not have control over kind or degree
No
None (N) No integrity loss
Base score is increasing in the degree of information that can be modified

X.1521(16)_FI.8

I.5.8 Availability impact


Is there any impact to the availability of a resource?
Yes
Can attacker completely deny access to the affected component, OR is resource critical?
Yes
High (H)
Resource is completely unavailable, OR select resource is critical to the component
No
Low (L)
Reduced performance or interruption of resource availability or response
No
None (N)
No availability impact
Base score is increasing in the degree of resource availability affected

X.1521(16)_FI.9

Appendix II

Resources and links

(This appendix does not form an integral part of this Recommendation.)

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
XML schema	Schema definition available at: https://www.first.org/cvss/cvss-v3.0.xsd

Bibliography

[b-ITU-T X.1500]	Recommendation ITU-T X.1500 (2011), *Overview of cybersecurity information exchange*
[b-ITU-T X.1524]	Recommendation ITU-T X.1524 (2011), *Common weakness enumeration.*

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Terminals and subjective and objective assessment methods
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	**Data networks, open system communications and security**
Series Y	Global information infrastructure, Internet protocol aspects and next-generation networks, Internet of Things and smart cities
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2016