CISA Stakeholder-Specific Vulnerability Categorization Guide

Publication: November 2022
Cybersecurity and Infrastructure Security Agency

Table of Contents

OVERVIEW

The CISA Stakeholder-Specific Vulnerability Categorization (SSVC) is a customized decision tree model that assists in prioritizing vulnerability response for the United States government (USG), state, local, tribal, and territorial (SLTT) governments; and critical infrastructure (CI) entities. This document serves as a guide for evaluating vulnerabilities using the CISA SSVC decision tree. The goal of SSVC is to assist in prioritizing the remediation of a vulnerability based on the impact exploitation would have to the particular organization(s). The four SSVC scoring decisions, described in this guide, outline how CISA messages out patching prioritization. Any individual or organization can use SSVC to enhance their own vulnerability management practices.

THE VULNERABILITY SCORING DECISION

When CISA becomes aware of a vulnerability, there are four possible decisions, as described in table 1.

Table 1: Vulnerability Decision, Possible Outcomes

Track	The vulnerability does not require action at this time. The organization would continue to track the vulnerability and reassess it if new information becomes available. CISA recommends remediating **Track** vulnerabilities *within* standard update timelines.
Track*	The vulnerability contains specific characteristics that may require closer monitoring for changes. CISA recommends remediating **Track*** vulnerabilities *within* standard update timelines.
Attend	The vulnerability requires attention from the organization's internal, supervisory-level individuals. Necessary actions may include requesting assistance or information about the vulnerability and may involve publishing a notification, either internally and/or externally, about the vulnerability. CISA recommends remediating **Attend** vulnerabilities *sooner than* standard update timelines.
Act	The vulnerability requires attention from the organization's internal, supervisory-level and leadership-level individuals. Necessary actions include requesting assistance or information about the vulnerability, as well as publishing a notification either internally and/or externally. Typically, internal groups would meet to determine the overall response and then execute agreed upon actions. CISA recommends remediating **Act** vulnerabilities *as soon as possible*.

Scope is an important variable in the scoring decision. An organization can determine a vulnerability's scope by understanding how the boundaries of the affected system are set. Understanding whether a vulnerability—with a presence across multiple related systems—is analyzed as one or multiple vulnerabilities will also help an organization determine the vulnerability's scope.

RELEVANT DECISION POINTS

CISA uses the following decision points and associated values for making vulnerability scoring decisions (outlined in the section above). One important omission from the values for each decision point below is an “unknown” option. Instead of declaring a decision point as “unknown,” CISA identifies the value that is the most reasonable assumption

based on prior events. Such an approach requires reliable historical evidence and future events may change these assumptions over time.

(State of) Exploitation

Evidence of Active Exploitation of a Vulnerability

This measure determines the present state of exploitation of the vulnerability. It does not predict future exploitation or measure feasibility or ease of adversary development of future exploit code; rather, it acknowledges available information at time of analysis. As the current state of exploitation often changes over time, answers should be time-stamped. Sources that can provide public reporting of active exploitation include the vendor's vulnerability notification, the National Vulnerability Database (NVD) and links therein, bulletins from relevant information sharing and analysis centers (ISACs), and reliable threat reports that list either the CVE-ID or common name of the vulnerability.

Table 2: Exploitation Decision Values

Value	Definition
None	There is no evidence of active exploitation and no public proof of concept (PoC) of how to exploit the vulnerability.
Public PoC	One of the following is true: (1) Typical public PoC exists in sources such as Metasploit or websites like ExploitDB; or (2) the vulnerability has a well-known method of exploitation. Some examples of condition (2) are open-source web proxies that serve as the PoC code for how to exploit any vulnerability in the vein of improper validation of Transport Layer Security (TLS) certificates, and Wireshark serving as a PoC for packet replay attacks on ethernet or Wi-Fi networks.
Active	Shared, observable, and reliable evidence that cyber threat actors have used the exploit in the wild; the public reporting is from a credible source.

Technical Impact

Technical Impact of Exploiting the Vulnerability

Technical impact is similar to the Common Vulnerability Scoring System (CVSS) base score's concept of "severity." When evaluating technical impact, the definition of scope is particularly important. The decision point, "Total," is relative to the affected component where the vulnerability resides. If a vulnerability discloses authentication or authorization credentials to the system, this information disclosure should also be scored as "Total" if those credentials give an adversary total control of the component.

Table 3: Technical Impact Decision Values

Value	Definition
Partial	One of the following is true: The exploit gives the threat actor limited control over, or information exposure about, the behavior of the software that contains the vulnerability; or the exploit gives the threat actor a low stochastic opportunity for *total* control. In this context, "low" means that the attacker cannot reasonably make enough attempts to overcome obstacles, either physical or security-based, to achieve total control. A denial-of-service attack is a form of *limited* control over the behavior of the vulnerable component.
Total	The exploit gives the adversary total control over the behavior of the software, or it gives total disclosure of all information on the system that contains the vulnerability.

Automatable

Automatable represents the ease and speed with which a cyber threat actor can cause exploitation events. *Automatable* captures the answer to the question, "Can an attacker reliably automate, creating exploitation events for this vulnerability?" Several factors influence whether an actor can rapidly cause many exploitation events. These include attack complexity, the specific code an actor would need to write or configure themselves, and the usual network deployment of the vulnerable system (i.e., the usual exposure of the system).

Table 4: Automatable Decision Values

Value	Definition
No	Steps 1-4 of the kill chain—reconnaissance, weaponization, delivery, and exploitation—cannot be reliably automated for this vulnerability.[1] Examples for explanations of why each step may not be reliably automatable include: (1) the vulnerable component is not searchable or enumerable on the network, (2) weaponization may require human direction for each target, (3) delivery may require channels that widely deployed network security configurations block, and (4) exploitation may be frustrated by adequate exploit-prevention techniques enabled by default (address space layout randomization [ASLR] is an example of an exploit-prevention tool).
Yes	Steps 1-4 of the of the kill chain can be reliably automated. If the vulnerability allows unauthenticated remote code execution (RCE) or command injection, the response is likely yes.

Another way of thinking about *automatable* is determining what barriers are in place that prevent the vulnerability from being wormable. One effective barrier is enough to get in a **No** answer. For example, if a user needs to be authenticated and logged in, that usually prevents a vulnerability from being wormable. However, if the vulnerable system has another unpatched vulnerability that remotely and easily gives an attacker a guest account or otherwise allows code injection, then the authentication barrier is no longer effective. This is the result of "chaining" vulnerabilities to make exploitation automatable, as discussed further below.

Another example of a barrier is if the vulnerable component does not normally have open connectivity to the internet. In some cases, "normally connected" can be analyzed via services such as Shodan (www.Shodan.io) to examine whether the vulnerable component is commonly exposed to the internet by other operators. Each of these examples is enough to prevent the vulnerability from being wormable. But if there are no effective barriers—either because there are none in place or they are all ineffective due to other common unpatched vulnerabilities—then vulnerability exploitation **is** *automatable*.

When analyzing *automatable*, an analyst should explicitly step through each of the four kill chain steps and ask what the barriers are to automating each step for the vulnerability in question. Like all SSVC decision points, *automatable* should capture the analyst's best understanding of plausible scenarios at the time of the analysis. An answer of **No** does not mean that it is impossible to automate exploitation in any scenario. It means that given the information currently available, the analyst is not able to sketch a plausible path through all four kill chain steps.

Automating Reconnaissance and Vulnerability Chaining

Due to vulnerability chaining, there is some nuance as to whether reconnaissance can be automated. For example,

- Vulnerability A and Vulnerability B both impact Product X.
- Vulnerability A allows a cyber threat actor to perform remote code execution.
  - However, the actor needs prior access to the target network to exploit Vulnerability A.
- Vulnerability B allows a cyber threat actor to view sensitive information in Product X remotely without needing to be on the target network.

[1] "The Cyber Kill Chain®," Lockheed Martin, accessed August 3, 2022.

From the threat actor perspective, Vulnerability A is the more severe and ideal vulnerability to exploit but requires more work compared to Vulnerability B. But, since both vulnerabilities affect Product X, the actor can first use Vulnerability B to easily get into the target network and then use Vulnerability A to perform remote code execution. The use of multiple vulnerabilities to achieve an overall outcome is known as "vulnerability chaining." SSVC lets analysts consider vulnerability chaining when determining the correct automatable decision point.

Mission Prevalence

Impact on Mission Essential Functions of Relevant Entities

A mission essential function (MEF) is a function "directly related to accomplishing the organization's mission as set forth in its statutory or executive charter."[2] Identifying MEFs is part of business continuity planning or crisis planning. In contrast to non-essential functions, an organization "must perform a [MEF] during a disruption to normal operations."[3] The mission is the reason an organization exists, and MEFs are how that mission is realized. Non-essential functions support the smooth delivery or success of MEFs rather than directly supporting the mission. In Table 5, an "entity" is a USG department or agency, an SLTT government, or a critical infrastructure sector organization.

Table 5: Mission Prevalence Decision Values

Value	Definition
Minimal	Neither *support* nor *essential* apply. The vulnerable component may be used within the entities, but it is not used as a mission-essential component, nor does it provide impactful support to mission-essential functions.
Support	The vulnerable component only *supports* MEFs for two or more entities.
Essential	The vulnerable component directly provides capabilities that constitute at least one MEF for at least one entity; component failure may (but does not necessarily) lead to overall mission failure.

Mission prevalence is more than simply counting devices or products present. If only a few devices are impacted, but they directly provide essential functions, then this criticality is what is important.

Quantity may still be an important consideration. Sometimes being ubiquitous is enough to directly provide essential functions. Examples for the right level of detail for a "mission" are "protect critical infrastructure" or "perform health inspections." This feature measures prevalence, not impact, so it does not need to account for any compensating controls or the impact of the vulnerability on the component. (*Technical impact* and *automatable* already measure the relevant features.)

Public Well-Being Impact

Impacts of Affected System Compromise on Humans

Safety violations are those that negatively impact well-being. SVCC embraces the Centers for Disease Control (CDC) expansive definition of well-being, one that comprises physical, social, emotional, and psychological health.[4]

Each decision option lists examples of the effects that qualify for that value/answer in the various types of well-being violations. These examples are suggestive and not comprehensive or exhaustive. While technical impact captures adversary control of the computer system, public well-being impact captures wider repercussions.

[2] For information about identification of mission essential functions, see Federal Continuity Directive 2: Federal Executive Branch Mission Essential Functions and Candidate Primary Mission Essential Functions Identification and Submission Process from June 2017.
[3] Ibid.
[4] "How is well-being defined?" Health-Related Quality of Life (HRQOL), Centers for Disease Control and Prevention, August 2019, https://www.cdc.gov/hrqol/wellbeing.htm#three.

Table 6: Public Well-Being Impact Decision Values

Impact	Type of Harm	Description
Minimal	All	The effect is below the threshold for all aspects described in material.
Material **(Any one or more of these conditions hold.)**	Physical harm	Does one or more of the following: • Causes physical distress or injury to system users. • Introduces occupational safety hazards. • Reduces and/or results in failure of cyber-physical system safety margins.
	Environment	Major externalities (property damage, environmental damage, etc.) are imposed on other parties.
	Financial	Financial losses likely lead to bankruptcy of multiple persons.
	Psychological	Widespread emotional or psychological harm, sufficient to necessitate counseling or therapy, impact populations of people.
Irreversible **(Any one or more of these conditions hold.)**	Physical harm	One or both of the following are true: • Multiple fatalities are likely. • The cyber-physical system, of which the vulnerable componen is a part, is likely lost or destroyed.
	Environment	Extreme or serious externalities (immediate public health threat, environmental damage leading to small ecosystem collapse, etc.) are imposed o other parties.
	Financial	Social systems (elections, financial grid, etc.) supported by the software are destabilized and potentially collapse.
	Psychological	N/A

Mitigation Status

Status of available Mitigations, Workarounds, or Fixes for the Vulnerability

Mitigation status measures the degree of difficulty to mitigate the vulnerability in a timely manner. There are three factors to consider (defined in Table 7 below): availability, difficulty, and type.

Table 7: Mitigation Decision Values

Factor	Value	Description
Minimal	Available	The mitigation is publicly available.
	Unavailable	The mitigation is not publicly available.
System change difficulty	Low	The system has an integrated update process, and the mitigation does not require any unreasonable interruption to the normal function of the vulnerable component.
	High	Any of the following are true: • The system does not have an integrated update process. • Applying the mitigation will require exceptional downtime. • After mitigation, system functionality will be reduced below normally acceptable levels. • The regulatory environment may prevent application of mitigation.
Type	Fix	An official patch that remediates the vulnerability.
	Workaround	Some way of preventing exploitation that does not patch the underlying issue; this is often in the form of a reconfiguration of the vulnerable component or its environment.

For availability and system change difficulty, it is intuitive that **unavailable** is a worse situation than **available** and that **high** is worse than **low**. For the type of mitigation, **workaround** is worse than **fix** because workaround mitigations tend to be more complex or require dedicated system owner actions, whereas a fix (patch) often has an established and comparatively easy process for application. System change difficulty should be low unless one of the conditions listed as high are met.

Based on the CISA decision tree, the value of mitigation does not change the priority of the SSVC decision. However, mitigation information is vital for vulnerability management and at a minimum should be tracked.

DECISION TREE

SSVC combines mission prevalence and public well-being impact decision points to determine whether exploitation of the vulnerability is likely. For CISA, this metric provides likelihood of exploitation for USG, SLTT governments, and critical infrastructure entities. Table 8 shows how to determine the decision value for mission and well-being.

Table 8: Determining Mission and Well-Being Impact Value

		Public Well-Being Impact		
		Minimal	***Material***	***Irreversible***
Mission Prevalence	***Minimal***	**Low**	**Medium**	**High**
	Support	**Medium**	**Medium**	**High**
	Essential	**High**	**High**	**High**

There are two representations of the proposed decision information: Figure 1 and Table 9. The decisions represented in each are the same.

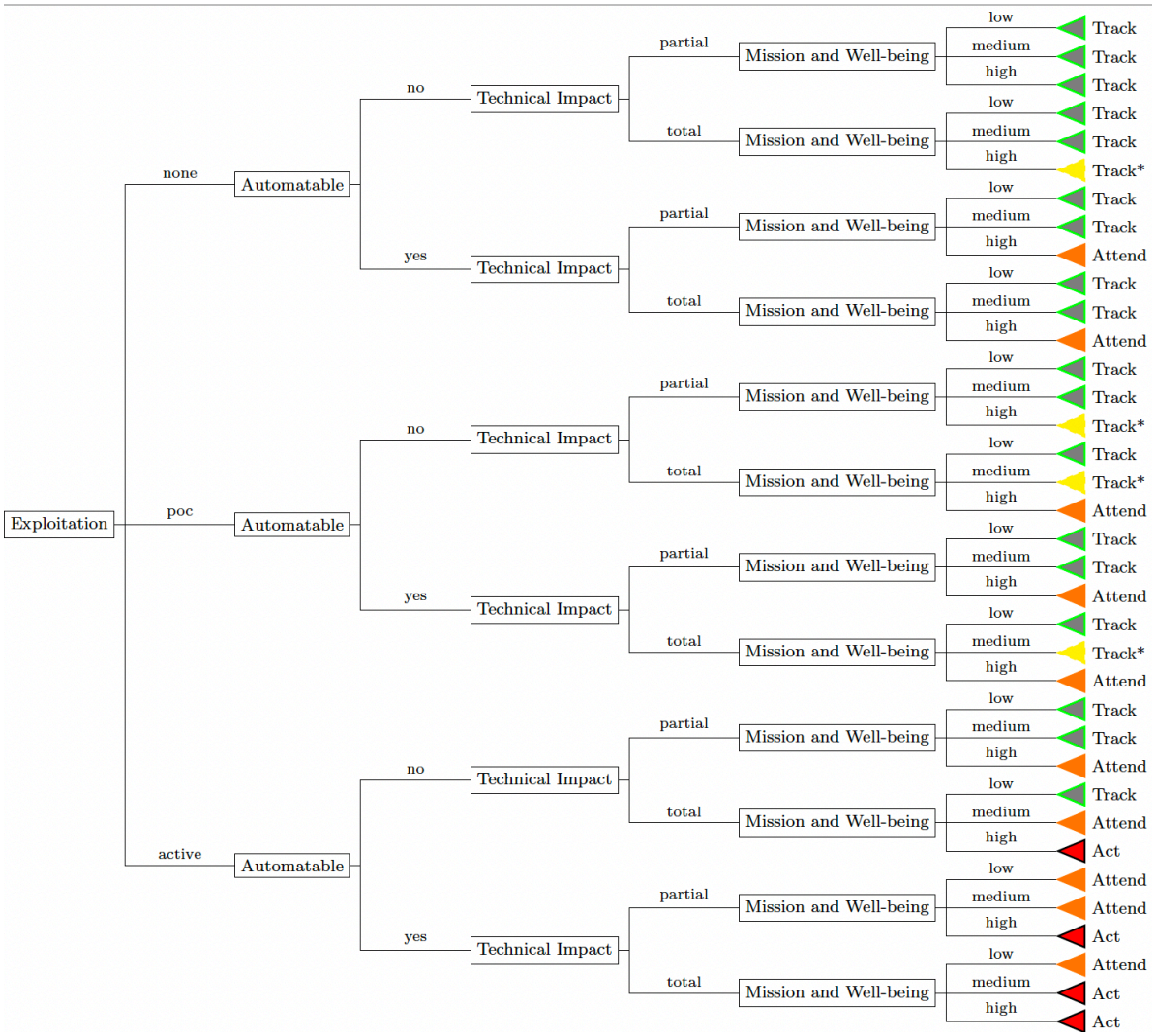

Figure 1: Decision Tree Representing Vulnerability Prioritization

Table 9: Table Representing Vulnerability Prioritization

Row Number	*Exploitation*	*Automatable*	*Technical*	*Mission and Well-Being*	*Decision*
1	none	no	partial	low	*Track*
2	none	no	partial	medium	*Track*
3	none	no	partial	high	*Track*
4	none	no	total	low	*Track*
5	none	no	total	medium	*Track*
6	none	no	total	high	*Track**
7	none	yes	partial	low	*Track*
8	none	yes	partial	medium	*Track*
9	none	yes	partial	high	*Attend*
10	none	yes	total	low	*Track*
11	none	yes	total	medium	*Track*
12	none	yes	total	high	*Attend*
13	poc	no	partial	low	*Track*
14	poc	no	partial	medium	*Track*
15	poc	no	partial	high	*Track**
16	poc	no	total	low	*Track*
17	poc	no	total	medium	*Track**
18	poc	no	total	high	*Attend*
19	poc	yes	partial	low	*Track*
20	poc	yes	partial	medium	*Track*
21	poc	yes	partial	high	*Attend*
22	poc	yes	total	low	*Track*
23	poc	yes	total	medium	*Track**
24	poc	yes	total	high	*Attend*
25	active	no	partial	low	*Track*
26	active	no	partial	medium	*Track*
27	active	no	partial	high	*Attend*
28	active	no	total	low	*Track*
29	active	no	total	medium	*Attend*
30	active	no	total	high	*Act*
31	active	yes	partial	low	*Attend*
32	active	yes	partial	medium	*Attend*
33	active	yes	partial	high	*Act*
34	active	yes	total	low	*Attend*
35	active	yes	total	medium	*Act*
36	active	yes	total	high	*Act*