CDM Vulnerability Management (VUL) Capability

Department of Homeland Security
Office of Cybersecurity and Communications
Federal Network Resilience

Table of Contents

1 Purpose and Scope

This toolkit outlines and documents issues of relevance to implementing the Vulnerability Management (VUL) Capability as part of Continuous Diagnostics and Mitigation (CDM). This toolkit provides general information on VUL and implications thereof. Further, this toolkit highlights potential considerations that technical implementers as well as managers may have when understanding how their organization can effectively implement VUL to better manage cybersecurity risk.

Additional considerations, inquiries, and suggestions for revision or addition can be submitted to: cdm.fnr@hq.dhs.gov. This toolkit will be updated as required.

2 FOREWORD

It is important to note the distinction between the terms *vulnerability* and *defect*. This distinction helps navigate some of the nomenclature issues that arise when discussing the subtle difference between these related concepts. A *vulnerability* is a flaw or weakness in system security procedures, design, implementation, or internal controls that could be exercised (accidentally triggered or intentionally exploited) and result in a security breach or a violation of the system's security policy. In the context of CDM, the term *defect* refers to the existence of a piece of software that has known vulnerabilities. *Defect* is the term of art used in the CDM context, while *vulnerability* refers more broadly to the definition as captured in NIST SP 800-30.

3 THREAT / ATTACKS

1. QUESTION: WHAT TYPES OF ATTACKS ARE WE TRYING TO ADDRESS WITH VUL?

Answer: VUL addresses known software vulnerabilities and attacks against those specific vulnerabilities.

Attack Description: Attackers continuously scan for systems that have software that may be unpatched, and for which there exists a publically known exploit. These systems are at extremely high risk of being exploited by malicious actors.

Background: The Common Vulnerabilities and Exposures (CVE) list (http://cve.mitre.org/index.html) tracks common flaws—or vulnerabilities—in computer software. A list of common vulnerabilities (including SQL injections, authentication issues, and buffer errors) can be found in the CVE section of the National Vulnerability Database (NVD). Attackers can exploit software by exploiting these known vulnerabilities that are published openly. Once malicious actors gain access, they can cause harm to resources and data residing on systems and networks.

Related Resources
NIST Guidance:
• NIST SP 800-40v2 -Creating a Patch and Vulnerability Management Program: http://csrc.nist.gov/publications/nistpubs/800-40-Ver2/SP800-40v2.pdf
• National Vulnerability Database: http://nvd.nist.gov/

According to a recent Center for Strategic and International Studies (CSIS) report, 75% of attacks target known vulnerabilities that could be patched; more than 90% of successful attacks require only the most basic techniques; and 96% of successful breaches can be avoided if the victim puts in place simple or intermediate controls.

2. QUESTION: WHAT IS A VULNERABILITY?

Answer: According to NIST's National Vulnerability Database, and for the purpose of Vulnerability Management, a vulnerability is a flaw or weakness in system security procedures, design, implementation, or internal controls that could be exercised (accidentally triggered or intentionally exploited) and result in a security breach or a violation of the system's security policy. Exposures through vulnerabilities can allow attackers to gather information, hide activities, compromise systems or data, and/or gain access to critical systems or data. A listing of vulnerabilities can be found in the Common Vulnerabilities and Exposures (CVE) Database (http://cve.mitre.org/). Furthermore, a listing of common weaknesses can be found on the MITRE common weakness enumeration (CWE) website (http://cwe.mitre.org/).

Common classes of vulnerabilities (derived from CWE) that exist in software include:

- Buffer overflows
- Structure and validity problems
- Channel and path errors
- Authentication errors
- Code evaluation and injection
- Randomness and predictability

A recent report from CSIS[1] found that CDM can stop 85% of cyberattacks by searching for, finding, fixing, and report the worst cyber problems first in near-real time.

3. QUESTION: HOW IS VUL RELATED TO PATCH MANAGEMENT?

Answer: Patch management supports Vulnerability Management as a means to automate patching of software in response to vendor-discovered vulnerabilities. NIST 800-137 (http://csrc.nist.gov/publications/nistpubs/800-137/SP800-137-Final.pdf#page=63) lists automated patch management tools in a section that combines the Vulnerability Management and Patch Management Security Automation Domains.

Effective patch management is a key (but not the only) requirement for effective vulnerability management. Vulnerability Management uses automated tools to find CVEs that are included in a report to be fixed, but does not itself focus on their remediation. Patch management tools often report what patches are present and assist with the automated patching of systems, but these tools do not necessarily correlate what they detect on systems to a set of known vulnerabilities.

[1] James A. Lewis. *Raising the Bar for Cybersecurity*. Washington, DC: CSIS, 2013.

If the results of two toolsets conflict (e.g., one says a patch is missing, and the other says it is present), the conflict registers as a defect of a false positive or false negative. Further examination is required to determine which is the case.

4. QUESTION: HOW CAN VUL ADDRESS KNOWN VULNERABILITIES IN MY ORGANIZATION?

Answer: In order to manage missing patches, it is essential to know which patches or other vendor-supplied mitigations are missing and can potentially be exploited by the adversary. Once these patches and mitigations have been identified, the organization can perform routine analysis to identify existing unpatched software on assets that could potentially allow unauthorized access.

A recommended process to remediate these issues includes a rigorous software upgrade and patch management regimen to address existing non-compliant assets, which, along with regular recurring searches for future non-compliant assets, significantly reduces the chances of a successful attack.

To manage this risk, there are several options:

Primary Methods

- Quickly patch the unpatched vulnerability to an acceptable patch level.
- Quickly assign the machine to a quarantined group where it can be examined and the lack of patching can be investigated.

Compensating Methods

- The SWAM Capability could provide compensating controls by disallowing software that has demonstrated too much vulnerability risk over time compared to its alternatives or too much vulnerability risk to the value to the business.

VUL focuses on preventing known vulnerabilities by finding and removing them BEFORE they can be exploited.

5. QUESTION: DOES VUL ADDRESS 0-DAY ATTACKS?

Answer: No, Vulnerability Management addresses only known attacks. It can help minimize the impact of 0-day attacks through virtualization and sandboxing but does not prevent malicious actors from taking advantage of 0-day exploits. SWAM may help prevent these 0-day attacks through proper use of whitelisting software. See question 8, referring to compensating controls.

4 INTEGRATION

6. QUESTION: WHAT CAPABILITIES SUPPORT VUL?

Answer: The HWAM, SWAM, and CSM capabilities support Vulnerability Management.

Before implementation VUL, it is essential to:
1. Know and catalog hardware through HWAM
2. Document allowed and disallowed software through SWAM
3. Manage the configurations of settings, and ensure that software is configured as intended through CSM

Only after these three phases are complete can enough information be passed on to the Vulnerability Management capability to check for known vulnerabilities.

7. QUESTION: WHAT CAPABILITIES DOES VUL SUPPORT?

Answer: Vulnerability Management does not support any other capabilities.

8. QUESTION: WHAT OTHER CAPABILITIES PROVIDE "COMPENSATING CONTROLS" TO VUL?

Answer: In Phase 1 of CDM, CSM and SWAM can provide compensating controls for Vulnerability Management.[2]

By simply enabling or changing certain configuration settings within software (e.g., the Data Execution Prevention setting in Windows) as part of CSM, vulnerabilities can be mitigated and controlled throughout an operating system.

SWAM can provide compensating controls by disallowing software that has demonstrated too much vulnerability risk over time compared to its alternatives or too much vulnerability risk compared to its business value.

If the SWAM inventory tool identifies software products at the release and patch level, this data may be used to identify the need for patches that the vulnerability scanner misses.

[2] More compensating controls will be added in CDM Phases 2 and 3.

5 DESIRED STATE

9. QUESTION: WHAT IS THE VUL "DESIRED STATE"?

Answer: Vulnerability Management's desired state *is a set of known software and versions and their association with known CVEs, such that the absence of CVEs can be verified for all software and versions on a given network.*

Background: Typically, a security bulletin is published for each known vulnerability; these bulletins provide options for mitigation and in some cases, recommendations for remediation. Knowing the desired state helps identify all known vulnerabilities from the NVD that may exist by comparing released vulnerabilities with the current inventory. If the hardware or software susceptible to the vulnerability exists, the problem can then be remediated.

Should patch managers be held responsible for CVEs for which the vendor has not provided patches? Patch managers should not be held responsible for such CVEs unless removing the software is an option. The risk should be acknowledged, but since the solution resides with the software, not the patching function, the risk belongs to the software provider or the business function that requires the high-risk software, not with the patch managers.

Most vulnerability scanners don't scan for every CVE in the NVD. An alternate definition of the desired state is to know all the CVEs that existing scanners can identify. In this case, the organization should use SWAM as a tool to identify CVEs that are present but that the scanners miss.

Ideally, all CVEs should be patched, but given the rate at which new CVEs are found and patches are released, reaching a steady state of zero CVEs is infeasible.

10. QUESTION: WHAT DOES THE ORGANIZATION NEED TO SPECIFY ABOUT ITS DESIRED STATE?

Answer: The organization must specify how it identifies needed patches as well as what levels of risk and vulnerabilities it considers acceptable.

The NVD should be used to identify needed patches. This tool is particularly useful for Vulnerability Management because the NVD provides a published list of disclosed vulnerabilities; other capabilities such as SWAM, HWAM, and CSM are unique to each organization and cannot rely on a published list.

Since it is infeasible to have zero CVEs, the organization must express its risk tolerance and the level of vulnerabilities it considers unacceptable.

11. QUESTION: WHAT DATA SHOULD BE RECORDED IN THE VUL DESIRED STATE SPECIFICATION FOR IN-SCOPE HARDWARE AND SOFTWARE?

Answer: The desired state specification should include a listing and details of all software that exist on an organization's network and the acceptable version level for each piece of software so the organization can know which corresponding CVEs might be present. The following data should be recorded:

Data Item	Justification
Expected CPE (vendor, product, version, release level) or equivalent	For identifying applicable CVEs
Prioritization of vulnerabilities for remediation	For lowering the risk to an organization (Some vulnerabilities are more critical than others and must be patched sooner.)
Listing of all applicable, approved patches for all software in the enterprise	For identifying all patches that must (and are authorized to) be applied from those identified in the National Vulnerability Database. In other words: data necessary to compare desired state and actual state or to determine existence of specific defects [conditions]) (The patches must be approved prior to implementation, as outlined in the SWAM and CSM capabilities.)

By design, this is a minimal list of data items that can function as a starting point. There are many operational and security reasons that more data may be required.

Most vulnerability tools today look for the absence of a patch rather than the vulnerability itself, so recording CPE is required for effective vulnerability management

12. QUESTION: IS RELYING ON THE NATIONAL VULNERABILITY DATABASE (NVD) A GOOD PRACTICE FOR VUL?

Answer: Yes, relying on the NVD is a good practice for Vulnerability Management.

The NVD is the U.S. Government's repository of standards-based vulnerability management data. The NVD does have shortfalls however, in that it does not index or track custom code. Keep this in mind if custom code is present in your organization's environment. The CDM program office will provide additional guidance as new vulnerabilities are published that are not accounted for in the NVD.

NVD contains tools, pointers, and recommendations for vulnerability management, security measurement, and compliance using the Security Content Automation Protocol (SCAP).

6 ACTUAL STATE

13. QUESTION: WHAT IS "ACTUAL STATE"?

Answer: The actual state is the list of CVEs present by device that exist within your organization. Your organization can determine actual state by using a tool or capability that is able to collect this information, or you can use the actual state as identified by the SWAM capability (for example, by correlating the version/patch level of a software asset with the respective entry in the NVD).

Background: If your organization's tools and capabilities are not able to check for all desired state vulnerabilities, then those vulnerabilities are assumed to be present if the asset inventory contains the corresponding software. It is important to compare the list of known vulnerabilities against the list of actual vulnerabilities, as timing is critical to resolving vulnerabilities before they are exploited. Any data that is recorded by automated means should be recorded in a way that is easily comparable to the desired state of vulnerabilities found on the network. Because of the shortfalls identified with vulnerability scanners, it is important to compare patch management, software management, and vulnerability identification software results to determine if a CVE could be present.

14. QUESTION: WHAT DATA SHOULD BE RECORDED IN "ACTUAL STATE"?

Answer: The actual state should record the date and time that the following data was collected:

Data Item	Justification
Actual CPE (vendor, product, version, release level) or equivalent	For reporting software versions
Listing of all software (version, release and patch level) found in the enterprise	For use in determining the difference between desired and actual states
CVEs that are present, and the level of each (high, medium, low) as indicated by the NVD	To determine what patches or other mitigation actions are needed to reduce risk associated with vulnerabilities
Date and time of data collection	For documenting when the checks were accomplished
Device the data was collected from	For identifying the device checked

By design, this is a minimal list of data items that can function as a starting point. There are many operational and security reasons that more data may be required.

15. QUESTION: HOW DOES MY ORGANIZATION DETERMINE ITS ACTUAL STATE?

Answer: Your organization can determine its actual state by discovering, identifying, and locating the current version numbers of software that exists anywhere within its area of responsibility. This is commonly done by using tools or capabilities that can query or interrogate machines that respond with version levels of software resident on their systems. Current version numbers can also be found by using data collected in the SWAM and/or CSM capability.

Background: Organizations should not rely on a single method to determine whether a vulnerability exists on its network. Cross-checking your HWAM and SWAM inventories against the NVD is only one method. Other methods should be used to identify and track the version numbers of your software to ensure it is not susceptible to any identified vulnerabilities. Once your organization determines whether its software is susceptible to identified vulnerabilities, you can manage each known vulnerability through the Common Vulnerability Scoring System (CVSS). The CVSS measures the severity of a vulnerability compared to other vulnerabilities so remediation efforts can be prioritized. It is critical that false positives are resolved in the fixing phase. For example, a vulnerability reported on a piece of software not identified on your SWAM inventory must be resolved to ensure the vulnerability does not truly exist.

7 FINDING DEFECTS

16. QUESTION: DOES THE NVD CONTAIN ALL THE VULNERABILITIES THAT ARE IMPORTANT?

Answer: The NVD is a necessary place to start, but it may not cover every vulnerability that could be present in your environment. If there is custom software on a network, it must be able to be checked for vulnerabilities and patch levels in order for it to be well-managed under Vulnerability Management.

17. QUESTION: HOW DOES AN ORGANIZATION FIND AND MANAGE KNOWN VULNERABILITIES?

Answer: The CMaaS provider finds and manages known vulnerabilities using the NVD as a basis.

Background: Persistent processes for identifying vulnerabilities on all software and hardware assets is the key step. After understanding the known vulnerabilities that can affect an organization, the CMaaS provider can research the NVD for Common Vulnerabilities and Exposures (CVE) and prioritize remediation efforts based on the severity of the vulnerabilities recorded. Correlating the desired state (a prioritized listing of all possible vulnerabilities on your assets) with the actual state (a listing of all vulnerabilities that exist across the organization inventory) can help identify the order in which vulnerabilities are remediated, documented as accepted as risks, or removed.

8 FIXING DEFECTS

18. QUESTION: WHAT OPTIONS ARE AVAILABLE FOR ADDRESSING THE DIFFERENCE BETWEEN ACTUAL AND DESIRED STATE?

Answer: Once an organization has identified the differences between actual and desired states (also known as *defects*), it is able to understand the differences and determine the appropriate corrective actions. Defects found can fall into 3 main categories, with the response options listed below:

Defect Type	Detection Rule	Response Options
Unpatched software/wrong version	A patch is needed but is not applied.	Apply the patch or accept the risk score.
Other CVE issue	Some other vendor fix is needed.*	Fix it or accept the risk score.
Non-reporting devices	The device is in the HWAM desired or actual state, but not in the VUL actual state with timely-enough data.	Restore reporting or declare the device missing/uninstalled/retired in HWAM.

Organizations can reference CVSS in order to work on the action(s) that have the highest priority or present the greatest amount of risk. CVSS can measure how serious a given vulnerability is compared to other vulnerabilities so remediation efforts can be prioritized.

*An organization must build a test for custom fix, as the vulnerability is considered mitigated only when you can prevent it from being exploited. An organization must verify that mitigation is successful before claiming the vulnerability is not a finding. Blocking such vulnerabilities at the firewall can reduce your risk score (the severity decreases), and in essence, an organization can consider the vulnerability not present. If mitigation limits access, the vulnerability is still present and still considered a finding, but your risk score is lower locally.

19. QUESTION: WHY IS THE GAP BETWEEN DESIRED STATE AND ACTUAL STATE IMPORTANT TO THE ORGANIZATION?

Answer: The gap between the desired state and actual state is important to the organization because the gap represents known, exploitable vulnerabilities that currently exist on a network. These vulnerabilities are what adversaries consider "low-hanging fruit" and what malicious actors will first focus on when trying to exploit a network. Organizations must work to address this difference on a continual basis. Your organization can reconcile differences between desired and actual states using methods discussed in Question 18.

Many vulnerabilities are known and shared and purchased in security and in adversary circles prior to any announcement that a patch is available.

20. QUESTION: HOW CAN WE MINIMIZE OUR EXPOSURE TO KNOWN VULNERABILITIES?

Background: There will never be a 100% effective way to prevent vulnerabilities. Your organization can reduce risk by using processes and procedures to routinely check and remediate vulnerabilities throughout your systems and networks. These activities can limit your exposure to vulnerabilities and minimize their impact.

Answer: The following actions can be taken to *reduce* your exposure to known vulnerabilities on systems throughout your enterprise:

1. **Network design and separation of exposed domains in design and placement of network systems.** In cases where systems cannot be patched, your organization can reduce its exposure to vulnerabilities by segmenting its services.

2. **Policy that dictates where systems may be added and how they are accredited and tested.** By having a policy that disallows systems that have not been tested for vulnerabilities, your organization can minimize its exposure to vulnerabilities.

3. **Software, hardware, and configuration best practices to ensure that security is built into the system's development life cycle.**

4. **Ensuring that known software is patched to the most recent patch level.**