Common Vulnerabilities and Exposures (CVE) and Open Vulnerability Assessment Language (OVAL)

Open Vulnerability Assessment Language (OVAL)

Open Vulnerability Assessment Language (OVAL) is a publicly available information security international standard used to evaluate and detail the system's current state and issues. OVAL is also co-supported by the office of Cybersecurity and Communications from the U.S. Department of Homeland Security. OVAL provides a language to understand encoding system attributes and various content repositories shared within the security community. The OVAL repository has over 7000+ definitions for public use. Additionally, OVAL is also used by the U.S. National Institute of Standards and Technology's (NIST) Security Content Automation Protocol (SCAP) which brings together community ideas for automating vulnerability management, measurement, and ensuring systems meet policy compliance.

OVAL Process

The goal of the OVAL language is to have a three-step structure during the assessment process that consists of:

1. Identifying a system's configurations for testing
2. Evaluating the current system's state
3. Disclosing the information in a report

The information can be described in various types of states, including: Vulnerable, Non-compliant, Installed Asset, and Patched.

OVAL Definitions

The OVAL definitions are recorded in an XML format to discover any software vulnerabilities, misconfigurations, programs, and additional system information taking out the need to exploit a system. By having the ability to identify issues without directly exploiting the issue, an organization can correlate which systems need to be patched in a network.

The four main classes of OVAL definitions consist of:

1. OVAL Vulnerability Definitions: Identifies system vulnerabilities
2. OVAL Compliance Definitions: Identifies if current system configurations meet system policy requirements
3. OVAL Inventory Definitions: Evaluates a system to see if a specific software is present
4. OVAL Patch Definitions: Identifies if a system has the appropriate patch

Additionally, the OVAL ID Format consist of a unique format that consists of "oval:Organization Domain Name:ID Type:ID Value". The ID Type can fall into various categories including: definition (def), object (obj), state (ste), and variable (var). An example of a unique identifier would be oval:org.mitre.oval:obj:1116.

Scanners such as Nessus have the ability to use OVAL to configure security compliance scanning templates.

Common Vulnerabilities and Exposures (CVE)

Common Vulnerabilities and Exposures (CVE) is a publicly available catalog of security issues sponsored by the United States Department of Homeland Security (DHS). Each security issue has a unique CVE ID number assigned by the CVE Numbering Authority (CNA). The purpose of creating a unique CVE ID number is to create a standardization for a vulnerability or exposure as a researcher identifies it. A CVE consists of critical information regarding a vulnerability or exposure, including a description and references about the issue. The information in a CVE allows an organization's IT team to understand how detrimental a problem could be to their environment.

Stages of Obtaining a CVE

Stage 1: Identify if CVE is Required and Relevant
Identify if the issue found is a vulnerability. According to the CVE Team, "A vulnerability in the context of the CVE Program is indicated by code that can be exploited, resulting in a negative impact to confidentiality, integrity, OR availability, and that requires a coding change, specification change, or specification deprecation to mitigate or address." Additionally, research should verify there is not a CVE ID already in the CVE database.

Stage 2: Reach Out to Affected Product Vendor
A researcher should ensure they have made a good faith effort to contact a vendor directly. Researchers can reference CVE's Documents on Disclosure Practices for additional information.

Stage 3: Identify if Request Should Be For Vendor CNA or Third Party CNA
If a company is a part of participating CNA's, they can assign a CVE ID for one of their products. If the issue is for a participating CNA, researchers can contact the appropriate CNA organization here. If the vendor is not a participating CNA, a researcher should attempt to reach out to the vendor's third-party coordinator.

Stage 4: Requesting CVE ID Through CVE Web Form
The CVE Team has a form that can be filled out online here if the methods above do not work for CVE requests.

Stage 5: Confirmation of CVE Form
Upon submitting the CVE Web Form mentioned in Stage 4, an individual will receive a confirmation email. The CVE team will contact the requestor if any additional information is required.

Stage 6: Receival of CVE ID
Upon approval, the CVE Team will notify the requestor of a CVE ID if the affected product's vulnerability is confirmed. Please note that the CVE ID is not public yet at this stage.

Stage 7: Public Disclosure of CVE ID
CVE IDs can be announced to the public as soon as appropriate vendors and parties are aware of the issue to prevent duplication of CVE IDs. This stage ensures that all associated parties are aware of the problem before being publicly disclosed.

Stage 8: Announcing the CVE
The CVE Team asks researchers who are sharing multiple CVEs to ensure each CVE indicates the different vulnerabilities. Additional information can be found here.

Stage 9: Providing Information to The CVE Team
At this stage, the CVE Team asks that the researcher help provide additional information to be used in the official CVE listing on the website. The U.S. National Vulnerability Database (NVD) maintains this information online in their database as well.

Responsible Disclosure

Security researchers and consultants constantly reference the CVE database since it consists of thousands of vulnerabilities that could be leveraged for exploitation. In addition, there are also times when individuals may come across an issue they have never seen in the wild or it has never disclosed while digging into a specific software or program.

Responsible disclosure is essential in the security community because it allows an organization or researcher to work directly with a vendor providing them with the issue details first to ensure a patch is available before the vulnerability announcement to the world. If an issue is not responsibly disclosed to a vendor, real threat actors may be able to leverage the issues for criminal use, also referred to as a zero day or an 0-day.

Examples

CVE-2020-5902

CVE-2020-5902 is an unauthenticated, remote code execution vulnerability in the BIG-IP Traffic Management User Interface (TMUI). The issue is exploitable when TMUI is available through the BIG-IP management port and leads to a complete system takeover since an attacker could execute code, edit files, and enable or disable services on the remote host.

CVE-2021-34527

CVE-2021-34527, also known as PrintNightmare, is a remote code execution vulnerability within the Windows Print Spooler service. The Windows Print Spooler service can be abused due to the service improperly handling privileges file operations. The issue requires a user to be authenticated but allows complete takeover of a system from remote or local code execution. The issue is extremely dangerous since it allows an attacker to fully control a domain since it exploits servers (including domain controllers) and workstations.