Vulnerability Management

Introduction to Processes and Standards

Klaus Möller

WP8-T1

Webinar, 27th of May 2021

Public

www.geant.org
The Road Ahead: Vulnerability Management

• Processes & Standards - 27\textsuperscript{th} of May
  - Processes: ISO 29147 & 30111
  - Standards: CVE, CVSS & CPE

• Vulnerability Information Dissemination - 8\textsuperscript{th} of June
  - How to get and distribute vulnerability information in your organization

• Patch Management - 11\textsuperscript{th} of June
  - How to keep track and fix vulnerabilities
The Road Ahead: Finding Vulnerabilities I

• Local Vulnerability Scanning - 28th of June
  - Finding vulnerabilities from inside

• Network Vulnerability Scanning - 30th of June
  - How to plan and conduct network scans
  - Tools: Nmap, OpenVAS

• Penetration Tests - 5th of July
  - Why, when and how
  - Examples of pen-test tools: ZAP, Metasploit
The Road Ahead: Finding Vulnerabilities II

• Code Audits - 14\textsuperscript{th} of July
  - How to increase the quality of your code

• Vulnerability Disclosure - 16\textsuperscript{th} of July
  - How to properly deal with found vulnerabilities

• Breach and Attack Simulation - 19\textsuperscript{th} of July
  - What would happen if vulnerabilities in your organization are exploited
What we will cover today

• What are vulnerabilities?
• Vulnerability management processes
  - ISO 29147:2018 - Vulnerability disclosure
  - ISO 30111:2019 - Vulnerability handling process
• Standards to assess vulnerabilities and their impacts
  - CVE
  - CVSS
  - CPE
  - Etc.

Source: MITRE
What is a Vulnerability?

- ENISA: “The existence of a weakness, design, or implementation error that can lead to an unexpected, undesirable event compromising the security of the computer system, network, application, or protocol involved.”

- ISO/IEC 27005: “A weakness of an asset or group of assets that can be exploited by one or more threats, where an asset is anything that has value to the organization, its business operations and their continuity, including information resources that support the organization’s mission.”

- IETF RFC 4949: “A flaw or weakness in a system's design, implementation, or operation and management that could be exploited to violate the system's security policy.”
Vulnerabilities & Risk

Threats \(\xrightarrow{\text{exploit}}\) Vulnerabilities

Vulnerabilities \(\xrightarrow{\text{affect}}\) Risks

Risks \(\xrightarrow{\text{affect}}\) Assets

Security Measures \(\xrightarrow{\text{protect against}}\) Vulnerabilities

Risks \(\xrightarrow{\text{lead to}}\) Security Requirements

Assets \(\xrightarrow{\text{make attackable}}\) Values

Security Requirements \(\xrightarrow{\text{are implemented by}}\) Security Measures

Values \(\xrightarrow{\text{have}}\) Assets
Design Error

• Fundamental flaws in protocols or software design
• Typical cases
  – Clear text authentication in protocols (telnet, ftp, ...)
  – Weak or outdated encryption or hash algorithms: MD3/4/5, SHA-1, DES, ...  
  – Flawed authentication protocols: WEP, WPA-2/3
  – Reliance on IP addresses for authentication
• Most difficult to fix - requires re-design of protocols or algorithms
• Fix usually breaks compatibility
• Systems left vulnerable in transition period (downgrade attacks)
Implementation Error

- Developer has made an error in designing or programming the software
  - No input validation: Buffer overflows, Format string bugs, XSS, SQL-Injection etc.
  - Broken access control: Session fixation, running processes with wrong privileges, etc.
  - Improper error handling
  - Race conditions
  - And many more ...

- Fixing requires analysis of the vulnerable code
- Requires testing of the corrected code
  - Open Source: Anybody can contribute (but who does?)
- Needs to be deployed in form of software upgrades (patches)
- To be conducted by developers (and system administrators)
Configuration Error (aka Weakness)

• A mistake in software configuration
  – By system administrator or user

• Like
  – Open accounts with no, known, or weak passwords
  – Active content enabled in web-browser or e-mail client
  – Unneeded network services: RPC interfaces, database mgmt., etc.
  – Disabled security functions: firewall, anti-virus scanner, auto-update, etc.

• Usually easy to fix by correcting the flawed configuration
  – Detection process somewhat different from other vulnerability types

• Can be done in the field, no outside dependency
Hardware Error

- Special case - fix typically requires replacing the hardware
- Replacement problems
  - Devices at hard to reach places (field sensors, inside machines, etc.)
  - Costs to replace expensive hardware or large number of cheap devices
  - Time needed to replace large number of deployed devices
- Often errors in hardware design
  - Re-design needed before new hardware can be built
  - Hardware upgrade cycles are much longer than software (re-tooling)
- Software patches to hardware vulnerabilities are workarounds
  - Often with serious performance impact
  - Often no complete mitigation
Operational Errors

- Flaws in the way operations are organized or carried out
- Typical flaws:
  - Blindly trusting phone calls
  - Blindly trusting web-links in e-mails or messages
  - Unsupervised (external) personnel in security areas
  - Unauthorized personnel in security areas
- Typically exploited by social engineering
  - Sometimes without any attack on hard- or software
- Fixing can be difficult - changing human behavior is tricky
Vulnerability handling vs. disclosure

ISO/IEC 29147:2018
Vulnerability disclosure

ISO/IEC 30111:2019
Vulnerability handling process
What will be covered in the course

Webinar: Vulnerability disclosure (Finding Vulnerabilities II)
- Develop vulnerability disclosure policy
- Develop capability to receive and publish vulnerability information
- Receive vulnerability report from external source
- Acknowledge receipt
- Inform reporter
- Publish advisory

Webinar: Vulnerability Information Dissemination
- Develop vulnerability handling policy and operational framework
- Identify vulnerability from internal source
- Verify report
- Vulnerability verified?
  - Yes: Develop and deploy remediation
  - No: Inform reporter
- Engage in post-remediation activities

Module: Finding Vulnerabilities I
- Receive vulnerability report from external source
- Acknowledge receipt
- Develop vulnerability disclosure policy

Module: Finding Vulnerabilities II
- Develop capability to receive and publish vulnerability information
- Receive vulnerability report from external source
- Acknowledge receipt
- Inform reporter
- Publish advisory

Webinar: Patch Management
Vulnerability handling policy (ISO 30111)

- Define and clarify organizations intentions when investigating and remediating vulnerabilities
- Internal part
  - Who is responsible, safeguards against premature disclosure
- Public part
  - How the organization will interact with external vulnerability finders
- How to process and resolve potential vulnerabilities
  - Investigation - is the vulnerability real, what are the consequences, etc.
  - Triage - prioritize handling of vulnerabilities
  - Remediation - how to deal with the found & confirmed vulnerabilities
Operational framework (ISO 30111)

- Covers all operational aspects (besides engineering)
- Defines a role to decide on vulnerabilities internally
  - And who assumes that role
- Defines a point of contact to the outside
  - E-mail: security@... (typically)
- Remediation - how to address a vulnerability
  - Patch, fix, upgrade, configuration or documentation change
  - Compare TARA principle
Related Discussion: Risk Management Strategies - TARA

- Transfer
  - Pass the risk on to somebody else
    - Business partner, insurer, etc.

- Accept
  - Do nothing
    - Often the only way

- Risk Treatment

- Reduce
  - Mitigate or
  - Share w/ partner

- Avoid
  - Try not to engage in the risky activity

- Measures can be combined
Typical Vulnerability Handling Timescale

- **Verify Report**: Days, given good quality initial report
  - Reproduce/understand bug, identify affected products
- **Fix**: Days, unless there’s a fundamental problem
  - Uses information from comprehensive evaluation
  - Look for workarounds as well as bugfixes
  - Check related code and design process for the same bug (ideally)
- **Test**: Weeks
  - Does it fix all problems, on all products?
    - Many different versions, platforms, languages to check
  - Does it break anything else? Start again if so
- **Release**, dependent on schedule/urgency
The Problem with Vulnerability Naming

- Vulnerabilities are referenced in many different contexts/products
- Are they talking about the same vulnerability?
  - Is “a vulnerability in the Linux x.y.z kernel network stack” the same as in “Linux kernel a.b network code problem”?
  - Compare names given to malware by AV vendors
- Or are they different?
Standards: Common Vulnerabilities and Exposures

- Idea: Give each vulnerability a unique identifier
- I.e. the CVE-Identifier: CVE-YYYY-NNNNN
  - Also called CVE-Name, CVE-Number, or CVE-ID
- Attached to the CVE-Identifier is additional information
  - (Technical) Details, References
  - Severity
  - Affected platforms
- All in one central repository
- CVE-IDs are assigned by CVE Numbering Authorities (CNAs)
  - If you find a vulnerability, ask your CSIRT or the vendor’s PSIRT
  - More about this in another webinar
Example: CVE-2016-1234

Current Description
Stack-based buffer overflow in the glob implementation in GNU C Library (aka glibc) before 2.24, when GLOB_ALTDIRFUNC is used, allows context dependent attackers to cause a denial of service (crash) via a long name.

References to Advisories, Solutions, and Tools
By selecting these links, you will be leaving NIST workspace. We have provided these links to other web sites because they may have information that would be of interest to you. No inferences should be drawn on account of other sites being referenced, or not, from this page. There may be other web sites that are more appropriate for the purpose. NIST does not necessarily endorse the views expressed, or concur with the facts presented on these sites. Further, NIST does not endorse any commercial products that may be mentioned on these sites. Please address comments about this page to nvd@nist.gov.

Weakness Enumeration

<table>
<thead>
<tr>
<th>CWE-ID</th>
<th>CWE Name</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>CWE-119</td>
<td>Improper Restriction of Operations within the Bounds of a Memory Buffer</td>
<td>NIST</td>
</tr>
</tbody>
</table>

Known Affected Software

Configurations

Hyperlink

Resource
- Mailing List
- Issue Tracking
- Patch
- Third Party Advisory

Severity
- CVSS Version 3.x
- CVSS 3.x Severity and Metrics:
  - NIST: NVD
  - Base Score: 7.5 HIGH

Up to (excluding) 2.24
- cpe:2.3:a:glibc:.........
- cpe:2.3:a:glibc:-w64*
- cpe:2.3:a:glibc:0.1:.......
- cpe:2.3:a:glibc:0.4:.......
- cpe:2.3:a:glibc:0.4.1:.......
- cpe:2.3:a:glibc:0.5:.......
- cpe:2.3:a:glibc:0.6:.......
- cpe:2.3:a:glibc:0.6.1:....
- cpe:2.3:a:glibc:0.10:....
- cpe:2.3:a:glibc:0.10.1:.......
- cpe:2.3:a:glibc:0.10.2:....
Standards: Common Vulnerability Scoring System

• Measure for the severity of a vulnerability (Score)
  - 0 (None) - least severe
  - 0.1-3.9 (Low)
  - 4.0-6.9 (Medium)
  - 7.0-8.9 (High)
  - 9.0-10.0 (Critical) - most severe

• More precisely: Three scores
  - Base
  - Temporal - changes over time
  - Environmental - depends on the organizations setup

• Plus context information about exploitability (Vector)
Standards: Common Vulnerability Scoring System

Base Metric Group:
- Exploitability metrics
  - Attack Vector
  - Attack Complexity
  - Privileges Required
  - User Interaction
- Impact metrics
  - Confidentiality Impact
  - Integrity Impact
  - Availability Impact
  - Scope

Temporal Metric Group:
- Temp (E, RL, RC)
  - Exploit Code Maturity
  - Remediation Level
  - Report Confidence

Environmental Metric Group:
- Env (CR, IR, AR, ...)
  - Modified Base Metrics
  - Confidentiality Requirement
  - Integrity Requirement
  - Availability Requirement

Exploit (AV, AC, PR, UI)
Impact (C,I,A),S

CVSS Score + Vector String

Source: https://www.first.org/cvss
CVSS Demo
Standards: Common Platform Enumeration

- Question: What is affected by a vulnerability?
- To answer we need “... a standardized method of describing and identifying classes of applications, operating systems, and hardware devices ...”
- A series of XML schemata that define
  - The structure of names for individual platforms (Naming)
    - “… the logical structure of Well-formed Names (WFNs)”
  - A standard to combine multiple WFNs with logical expressions (i.e., AND, OR, NOT) so that multiple products and platforms can be matches (Applicability Language)
  - Rules to parse and match (compare) WFNs (Name Matching)
  - A repository of registered names (Dictionary) each entry identifying a single class of IT product
- What if a vulnerability is found a product that doesn’t have a CPE (yet)?
CPE in Practice: GNU C Library

Multiple WFNs are grouped with logical operators (i.e. “or”, “not”):

cpe:2.3:a:gnu:glibc:2.2.3:*:*:*:*:*:*:* OR
cpe:2.3:a:fedoraproject:fedora:23:*:*:*:*:*:*:*
# CPE Example: CVE-2016-1234

## Known Affected Software Configurations

### Configuration 1

Switch to CPE 2.2

| CPE:2.3:a:gnu:libc:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |

Hide Matching CPE(s)

- cpe:2.3:a:gnu:libc:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |
- cpe:2.3:a:gnu:libc:0.1:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |
- cpe:2.3:a:gnu:libc:0.4:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |
- cpe:2.3:a:gnu:libc:0.4.1:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |
- cpe:2.3:a:gnu:libc:0.5:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |
- cpe:2.3:a:gnu:libc:0.6:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |
- cpe:2.3:a:gnu:libc:1.00:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |
- cpe:2.3:a:gnu:libc:1.01:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |
- cpe:2.3:a:gnu:libc:1.02:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |

Showing 10 of 117 matching CPE(s) for the range. View All CPEs here

### Configuration 2

| CPE:2.3:o:opensuse:leap:42.1:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |

Hide Matching CPE(s)

- cpe:2.3:o:opensuse:leap:42.1:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |

### Configuration 3

| CPE:2.3:o:fedoraproject:fedora:23:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |

Hide Matching CPE(s)

- cpe:2.3:o:fedoraproject:fedora:23:*::*:***:*:*:*:*:*:*:*:*:*:*:*:*:x64:* |

Denotes Vulnerable Software

Are we missing a CPE here? Please let us know.
CPE Example: nmap & Linux

# nmap -O localhost
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-17 17:27 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000030s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 996 closed ports

<table>
<thead>
<tr>
<th>PORT</th>
<th>STATE</th>
<th>SERVICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>22/tcp</td>
<td>open</td>
<td>ssh</td>
</tr>
<tr>
<td>25/tcp</td>
<td>open</td>
<td>smtp</td>
</tr>
<tr>
<td>631/tcp</td>
<td>open</td>
<td>ipp</td>
</tr>
<tr>
<td>6667/tcp</td>
<td>open</td>
<td>irc</td>
</tr>
</tbody>
</table>

Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32
OS details: Linux 2.6.32
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.56 seconds

> cat /etc/os-release
NAME="openSUSE Leap"
VERSION="15.2"
ID="opensuse-leap"
ID_LIKE="suse opensuse"
VERSION_ID="15.2"
PRETTY_NAME="openSUSE Leap 15.2"
ANSI_COLOR="0;32"
CPE_NAME="cpe:/o:opensuse:leap:15.2"
BUG_REPORT_URL="https://bugs.opensuse.org"
HOME_URL="https://www.opensuse.org/"
More Standards

- Building open standards to automatically process incoming vulnerability information
  - According to your strategy/policy
- Goal: Systems are automatically (securely) configured and/or patched
  - Common Configuration Enumeration (CCE)
  - Common Weaknesses Enumeration (CWE)
  - *Security Content Automation Protocol (SCAP)*
  - Asset Identification, Asset Reporting Format (ARF)
  - *Open Vulnerability Assessment Language (OVAL)*
  - Open Checklist Interactive Language (OCIL)
  - Trust Model for Security Automation Data (TMSAD)
  - Extensible Configuration Checklist Description Format (XCCDF)
  - Software Identification (SWID)
  - Asset Summary Reporting (ASR)
What have you learned?

• Two main standards: ISO 29147 and ISO 30111
  – Vulnerability disclosure
  – Vulnerability handling

• Open standards making vulnerability information machine readable
  – CVE - Vulnerability identifier
  – CVSS - severity score
  – CPE - affected platform

• Next webinar: human readable vulnerability information - security advisories
Thank you

Any questions?

Next webinar: *Vulnerability Information Dissemination*,
8th of June 2021

www.geant.org
References:

- CVE Specification: https://cve.mitre.org/
- National vulnerability database: https://nvd.nist.gov/vuln/
- CPE Specification: https://cpe.mitre.org/specification/
- FIRST CVSS page: https://www.first.org/cvss/
- FIRST CVSS course: https://www.first.org/education/trainings