INF226 – Software Security

Håkon Robbestad Gylterud

2019–09-04

Today

- CVE – specific vulnerabilities in specific software
- CWE - kinds of vulnerabilities
- CVSS – scoring vulnerabilties
- NVD – Connects CVE entries with other data (CWE, CVSS, ...)

CVE

CVE

Common Vulnerabilities and Exposures (**CVE**) is a database of software vulnerabilities. Maintained by **The Mitre Corporation** in USA.

The list has entries consisting of:

- **A unique number** (CVE–YYYY–XXXX) identifying the vulnerability
- A desciption
- At least one public reference

CVE examples

We go to https://cve.mitre.org/.

CVE number assignment

Assigning the CVE numbers is taken care of by the **CVE Numbering Authorities** (CNAs), which each have **different scopes**. These include:

- The Mitre Corporation (Primary CNA)
- Distributed Weakness Filing Project (For open-source projects)
- Many corporations (Google,Microsoft,Intel,Netflix,$\cdots$)

What is CVE used for?

CVE allows referencing vulnerabilities **accross systems**:

- Easier than referencing product/version/description:
  - **Easy**: CVE-2018-7492
  - Difficult: "That NULL pointer dereference in net/rds/rdma.c in Linux before 4.14.7."
- Easy to **track** vulnerability fixes:
  - From links we quickly find which Debian or Ubuntu packages contain the fixes.
- Provides a **quick way to look up vulnerabilities** for a given piece of software.

CVE numbers are often reported by vulnerability scanners which finger-print running services.

CVSS

CVSS

Common Vulnerability Scoring System (**CVSS**) is a system for assigning a **score to a vulnerability**.

Includes three kinds of metrics:

- **Base metrics**, intrinsic properties
- **Temporal metrics**, changes over the vulnerability life-time
- **Environmental metrics**, specific to the environment of the software.

CVSS results in sevaral scores on a scale from **0–10**, based on a vector of metrics.

CVSS

Two different versions of CVSS are commonly used:

- Version 2
- Version 3

Link to the specification of version 3 on syllabus page.

Base metrics in CVSS Version 2

- Access vector:†
  - Local
  - Adjacent network
  - Network
- Attack complexity (High/Medium/Low)
- Authentication (Multiple/Single/None)

†: Version 3 adds “physical”

Impact metrics in CVSS Version 2

Rated on a scale of None/Partial/Complete impact:

- Confidentiality
- Integrity
- Availability

Temporal metrics in CVSS Version 2

The following metrics change over time:

- Exploitability
- Remediation level
- Report confidence

Exploitability

Exploitability is measured on a the scale:

- Unproven
- Proof-of-concept
- Functional
- High

Remediation level

Remediation level is measured on the scale

- Official fix
- Temporary fix
- Workaround
- Unavailable

Report confidence

Report confidence is measured on the scale

- Unconfirmed
- Uncorroborated
- Confirmed

CVSS example

Impact

CVSS v3.0 Severity and Metrics:

Base Score: 5.5 MEDIUM
Vector: AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H (V3 legend)
Impact Score: 3.6
Exploitability Score: 1.8

Attack Vector (AV): Local
Attack Complexity (AC): Low
Privileges Required (PR): Low
User Interaction (UI): None
Scope (S): Unchanged
Confidentiality (C): None
Integrity (I): None
Availability (A): High

CVSS v2.0 Severity and Metrics:

Base Score: 4.9 MEDIUM
Vector: (AV:L/AC:L/Au:N/C:N/I:N/A:C) (V2 legend)
Impact Subscore: 6.9
Exploitability Subscore: 3.9

Access Vector (AV): Local
Access Complexity (AC): Low
Authentication (AU): None
Confidentiality (C): None
Integrity (I): None
Availability (A): Complete
Additional Information:
Allows disruption of service

Figure 1: CVE-2018-7492

CWE

CWE

Common Weakness Enumeration (**CWE**) is a list of common weaknesses occuring in software.

- Contains more than 600 classes of weaknesses.

Similar in intent to OWASP Top 10, but more general and fine grained.

CWE vs OWASP

1026 - Weaknesses in OWASP Top Ten (2017)

- OWASP Top Ten 2017 Category A1 - Injection - *(1027)*
  - Improper Neutralization of Special Elements used in a Command ('Command Injection') - *(77)*
  - Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection') - *(78)*
  - Argument Injection or Modification - *(88)*
  - Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') - *(89)*
  - Improper Neutralization of Special Elements used in an LDAP Query ('LDAP Injection') - *(90)*
  - XML Injection (aka Blind XPath Injection) - *(91)*
  - SQL Injection: Hibernate - *(564)*
  - Improper Neutralization of Special Elements used in an Expression Language Statement ('Expression Language Injection') - *(917)*
  - Improper Neutralization of Special Elements in Data Query Logic - *(943)*
- OWASP Top Ten 2017 Category A2 - Broken Authentication - *(1028)*
- OWASP Top Ten 2017 Category A3 - Sensitive Data Exposure - *(1029)*
- OWASP Top Ten 2017 Category A4 - XML External Entities (XXE) - *(1030)*

Figure 2: OWASP in terms of CWE

Usage of CWE

- CWE numbers are often given as output from security analysis tools (such as ZAP or SonarQube).
- CWE is a hiearchy of weakness descirptions, usefully structured:
  - By architecture concepts
  - By development concepts
  - By research concepts

NVD

NIST National Vulnerability Database

The National Vulnerability Database (**NVD**) contains analysis of known vulnerabilities:

- CVE numbers
- CWE numbers
- CVSS
- Versions affected

Vendor security advisories

Software vendors often have **separate security advisories**.

Examples:

- Microsoft Security Bulletin
- Apple Security Advisory (APPLE-SA)
- Debian Security Advisory (DSA)

It is good practise to **subscribe to advisories** of the vendors of your platform.

Security tools

Security tools

- Static analysis: inspects source code
- Dynamic analysis: inspects the running software

Examples

A **static analyser** will be able to tell you that this code is illogical:

```
boolean checked = false;

if(checked) {
    // ...
}
```

A **dynamic analyser** will be able to tell you that your program crashes when given invalid UTF-8 strings.

Kinds of dynamic analysis

- Fuzzer: feeds random data to the program to trigger anomalies.
- Crawlers: Maps out the attack surface of the program.
- Man-in-the-middle proxy: analyses data from normal usage.
- Vulnerability scans:
  - SQL injection tests
  - XSS tests
  - Anti CSRF token detecton
  - ...

Kinds of static analysis

- Program flow analysis
- Constraint analysis
- Logic tests
- Linting