TP Class Assignment - 01/Apr/2019

Each group must answer the questions of the following exercises in the Github area of their group until the end of 16/Apr/2019. For each day of delay, 0.15 points will be deducted from the grade of this assignment.

Exercises

1. Implementation vulnerability

Experience 1.1 - *Common Weakness Enumeration* (CWE)

Common Weakness Enumeration (CWE) classifies vulnerability classes by assigning to each vulnerability class an identifier.

Go to https://cwe.mitre.org/ and see what vulnerabilities are identified as:

- project vulnerabilities, introduced during the software design phase (design and requirements gathering) - https://cwe.mitre.org/data/definitions/701.html. Explore one of the members of this vulnerability class and see (and understand) one of the demonstrative examples.
- implementation vulnerability, introduced during software development - https://cwe.mitre.org/data/definitions/702.html. Explore one of the members of this vulnerability class and see (and understand) one of the demonstrative examples.
- operational vulnerability, caused by the environment in which the software is run or by its configuration - https://cwe.mitre.org/data/definitions/16.html. Explore one of the members of this vulnerability class and see (and understand) one of the demonstrative examples.

See also which vulnerabilities are identified as::

- *CWE/SANS Top 25 Most Dangerous Software Errors* (2011) - https://cwe.mitre.org/top25/
- *OWASP Top Ten* (2017) - https://cwe.mitre.org/data/definitions/1026.html

Experience 1.2 - *Common Vulnerabilities and Exposures* (CVE)

Common Vulnerabilities and Exposures (CVE) identifies vulnerabilities (design and implementation vulnerabilities) existing in commercial or open software, with identifier format CVE-AAAA-NNNN, being AAAA the year in which the vulnerability was cataloged and NNNN its number.

Goto https://cve.mitre.org/ and verify:

- the detail of the most recent vulnerability;
- vulnerabilities identified in Google Chrome;
- vulnerabilities identified in Facebook.

Experience 1.3 - *Common Vulnerability Scoring System* (CVSS)

The *Common Vulnerability Scoring System* (CVSS) provides a quantitative model to define the characteristics and impact of vulnerabilities, ensuring accurate and repeatable measurement to generate vulnerability impact scores.

Two common uses of CVSS are prioritizing vulnerability correction activities and calculating the severity of vulnerabilities discovered.

Explore the vulnerability calculator in https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator.

Experience 1.4 - *National Vulnerability Database* (NVD)

National Vulnerability Database (NVD) is the vulnerability repository managed by NIST. It is based on the CVE but includes the severity of the vulnerability, according to the CVSS (*Common Vulnerability Scoring System*)

Goto https://nvd.nist.gov/ and verify:

- What is the most recent vulnerability identified?
- Is this vulnerability the same most recent vulnerability found in Experience 1.2 (CVE)? What could be the reason?
- vulnerabilities identified in Google Chrome;
- vulnerabilities identified in Facebook.

Question P1.1

In https://informationisbeautiful.net/visualizations/million-lines-of-code/ you'll find (some are estimates) the number of code lines (*SLOC - Source Lines Of Code*) of some software packages / platforms.

1. Estimate the number of bugs in Facebook, car software, Linux 3.1 and all Google Internet services.
2. How many of these bugs are vulnerabilities?

Question P1.2

Consider the three types of vulnerabilities: design, implementation, and operational. Come up with two examples for each of the three types of vulnerabilities and discuss the difficulty of correction.

Question P1.3

What distinguishes a zero-day vulnerability from another non-zero-day implementation vulnerability?

Experience 1.5 - *Exploit Database*

The *Exploit Database* is an archive of public exploits, identifying the CVE of the vulnerability and/or software it exploits, for use by vulnerability researchers and penetration testers.

Go to https://www.exploit-db.com/ and verify:

- What is the most recent *exploit* identified?
- What does the *exploit* for CVE-2019-6780 allows you to do?

Experience 1.6 - *Google Hacking Database*

The *Google Hacking Database* is an archive of Google dorks (search query that returns sensitive information) that although being a form of exploits, are also used to create new exploits.

Go to https://www.exploit-db.com/google-hacking-database/ and verify:

- What is the latest *dork* identified?
- explore some *dorks*.

Experience 1.7 - C language

As seen in the previous class, the C language is a compiled language in which each C instruction is translated into machine language instructions.

Using the virtual machine, compile a program C with `gcc` with the `-S` option, in order to produce the assembly code corresponding to the binary/executable code (the assembly code is in the file with .s extension). Check the .s file and compare it with the original .c file.

As an example use the following C code:

```
#include <stdio.h>

void main()
{
  printf("Hello World\n");
}
```

Engenharia de Segurança course - Project

You can use the rest of the class for the Engenharia de Segurança project.