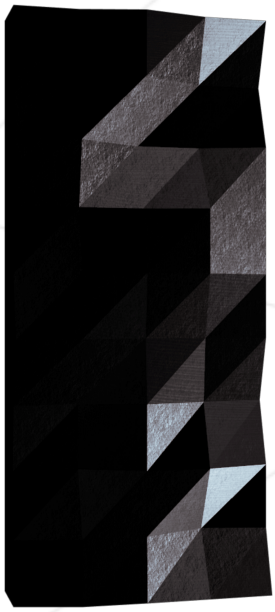

Cybersecurity Piscine

Vaccine

Summary: SQL Injection

Version: 1.00

Contents

Chapter I

Introduction

We all know how important secure programming is. In this case you will try to find `filtering errors in the data input`. `SQL Injection` is the injection of SQL commands to alter the behaviour of a program and execute commands on the database. In this project you will create a tool that is able to detect SQL injections providing a URL.

Chapter II

Mandatory part

You have to create a program called `vaccine` which allows you to perform an SQL injection by providing a url as parameter.

You need to understand how SQL injection works. There are sites where you can test this vulnerability legally.

You must not test your program on a site where you are not authorized to do so.

The tool should have a battery of tests to run against a given URL and, depending on the responses, be able to detect SQL injections. You can detect the type of database engine to make the tests more successful (2 minimum). The tests can be based on several types: union, error, boolean, time and even blind (2 minimum).

In case a website is confirmed to be vulnerable, the following can be obtained:

- The vulnerable parameters.
- The payload used.
- Database names.
- Table names.
- Column names.
- Complete database dump.

The tool must have some storage file for the data, if it does not exist it will be created on the first run.

You have to manage the following programme options:

```
./vaccine [-oX] URL
```

- Option `-o` : Archive file, if not specified it will be stored in a default one.
- Option `-X` : Type of request, if not specified GET will be used.

`You must manage at least the GET and POST methods.`

Your program:

- must be able to check at least 2 injection methods such as Union, Error, Boolean or other.
- must be able to handle two database engines such as Oracle, Mysql, SQLite, Microsoft SQL Server or other.

You can use any programming language, you should not use libraries that automate SQL injection.

If you use a language that requires compilation you must add a Makefile.

You must add a README.md file to the root indicating how the tool works and how to configure it if needed.

As always you should prepare a series of tests to show how your program works.

Chapter III

Bonus Part

You can enhance your project with some or all of the following features:

- Wider range of database engines (1 point per engine added).
- Wider range of SQL injection methods (1 point per method added).
- The tool allows you to edit various parameters of the request, e.g. the User-Agent (1 point max).

The bonus part will only be assessed if the mandatory part is PERFECT. Perfect means the mandatory part has been integrally done and works without malfunctioning. If you have not passed ALL the mandatory requirements, your bonus part will not be evaluated at all.

Chapter IV

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your folders and files to ensure they are correct.