Information Systems Security Database Security Lab



Lab objective

- ▶ The objective of the lab is to see the different options for securing a database
- ▶ To do this, you will manipulate a MySQL database
 - User creation and rights management
 - Creating databases and tables
 - > Creating a relationship between tables and testing data access
 - Creating a view and testing data access
 - > Encryption
 - Enabling logging
 - Backup and recovery
- This lab must be done without using ChatGPT or other Al tool

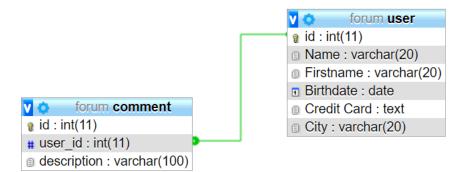
Rights management

- Install Wamp on your workstation (or XAMPP on Mac or Linux) (http://www.wampserver.com/)
- Create a MySQL database
- Create a user for this database
 - What is the SQL command executed ?
- Give the database user only the necessary rights
 - SELECT, INSERT, UPDATE, and DELETE on the previously created database
 - What is the SQL command executed ?
- In SQL tab, use an SQL command to create a second user and grant him read rights only to the database



Relationship between data

- ▶ From the admin interface, create two tables (using InnoDB storage engine) :
 - A "user" table with the following columns:
 - ✓ A column « id » (primary key)
 - ✓ A column « Name »
 - √ A column « Firstname »
 - ✓ A column « Birthdate »
 - ✓ A column « Credit card »
 - ✓ A column « City »
 - ➤ A "comment" table with the following columns :
 - ✓ A column « id » (primary key)
 - ✓ A column « description »
 - ✓ A column « user id » (foreign key with « restrict » constraint)





Relationship between data

- Create a PHP script that accesses the database
- ▶ From PHP script using PDO :
 - Create a first web page containing a form to dynamically create users paying attention to SQL injections
 - Create three users in the "user" table using the web form
 - Display user data by <u>preventing any code execution</u> (XSS) that may be stored in database values
 - Create a button to delete a user



Relationship between data

- Create a second web page containing a form to add comments and display all the comments while <u>avoiding SQL injections</u> and <u>preventing</u> <u>any code execution</u> (XSS) potentially stored in the values of the database
- Insert a comment in the "comment" table with a user ID corresponding to a user
- ➤ Insert a comment in the "comment" table with a user ID that does not exist in the "user" table. How does the database react?
- Delete the user whose ID was used in the comment table. How does the database react?
 - ✓ What are the different possible policies in the event of a constraint violation?



Access control

- ▶ To prevent the second user from accessing sensitive data (credit card number) and to group the data in a table, create a new view
 - ➤ The view will only contain the user's first and last name and the various comments (make a join) of each user
- Remove the read right on "user" table from the second user
- Give the second user access to the view
 - Create a new page displaying the view data as a second user



Encryption

Modify your application code to encrypt sensitive data (credit card number)

- ➤ The credit card number must displayed in plain text on the web page for the banking partner (in the page displaying the user's data)
- The database is considered to be hosted on a dedicated server in this scenario

Does your modified code (with encryption) protect you :

- ➤ Of the exfiltration of the unencrypted credit card number by the database administrator?
- ➤ Of the exfiltration of the unencrypted credit card number by the administrator of the server hosting the website accessing the database?
- ➤ Exfiltration of the credit card number by SQL injection on forms (if one field become vulnerable to SQL injection)?
- Exfiltration by an attacker exploiting a vulnerability on the server hosting the website accessing the database?

Audit and backup

- In phpMyAdmin, enable log query (modifying general_log variable): https://dev.mysql.com/doc/refman/8.4/en/query-log.html
 - Access previously activated logs: find the queries made on the database (SELECT, UPDATE, etc.) -> take a screenshot
 - Indicate in which file the logs are located
- In phpMyAdmin, create a database backup
 - Make any changes in the database
- In phpMyAdmin, restore the backup
 - Verify that changes are missing
- Indicate how you performed the backup/restore



Report

▶ The report of the lab must contain :

- > The entire PHP and HTML source code
- > Screenshots of your web pages, the database management interface and any other elements you think are necessary with explanations
- Particular attention must be paid to the <u>security of the source code</u> (necessary and sufficient security)

