IDATT2504 Exercise 1

Torgeir Haukaas and Ingar Asheim

September 5, 2024

Task 1: WebGoat

Task 1:.1 A1: Broken Access Control

Hijack a session

First I try to log in. The server sends a set-cookie with the "hijack_cookie". After repeating the request, it is clear that it consists of two numbers, one that increments by one or two, and the other is a timestamp. I can try to hijack a logged in users session if I can guess their cookie. When the first cookie increments by two it is likely that another user has logged in in between my attempts. Then I know the first number of their cookie, and I can iterate on the timestamp, which lies between my previous attempts.

Insecure Direct Object References

First I authenticate as Tom Cat. I then capture and inspect the response from the server which includes the fields of "role" and "userId" in addition to the others. In typical restfull fashion I can append a profile number to the end of the profile url: /WebGoat/IDOR/profile/userId. Assuming that other user ids are similar to Tom Cats, I try to iterate on the last two digits of Tom Cats id and discover the userId for Buffalo Bill, 2342388. To change the info of Buffalo Bill I change the request to PUT, content-type to application/json and append the json data I want to insert.

Task 1:.2 A3: Injection

SQL Injection(Intro)

I get the department with the query "select department from employees where userid=96134;".

I update the department with the query "update employees set department='Sales' where userId=89762;".

I add the phone column with the query "alter table employees add phone varchar(20)";.

I grant rights with the query "GRANT insert ON grant_rights TO unauthorized_user;".

I can use injection by inserting "Smith' or '1' = '1'".

I can use injection on the last field by writing "1 or '1' = '1'".

I can get the salaries by typing in "' or 1=1;–" in the Lastname field.

I can change John salary by inputting "smith'; update employees set salary = 999999 where auth_tan='3SL99A';–".

I can delete the table by using "'; drop table access_log; –".

SQL Injection(Advanced)

I can get the user data with the query "'; select * from user_system_data;–"

I can use the name field in the register form to first check if the username tom is taken first. Then I can use a "and" clause in a SQL query to see a wierd error message. Then I can use a substring function to brute force the password one letter at a time. Example query for the first letter: "tom'+and+substring(password,1,1)='t';–".

The password turned out to be: thisisasecretfortomonly.

Path traversal

put ../ before the username.

put ....// before username

put ../ before the filename

put urlencoded ../../ as id followed by file name

```
GET
/WebGoat/PathTraversal/random-
picture?id=%2e%2e%2f%2e%2e%2f%2fpath-traversal-secret HTTP/1.1
```

Task 2: Hacker101

For two flags involving XSS/javascript injection I could put javascript in a "onmouseover" event in html tags in both the title of the page to be viewed on the home page, and in the content of a page.

The /page/*id* in the url is vulnerable to sql injection. I assume the *id* is used in a select query somewhere and can be appended with e.g. "' and 1=1".

When I created a new page I noticed the index of the page skipped a few numbers. Trying the indexes revealed that index 5 was private. I could add /edit before the index to access the contents.