CSC 681 PRINCIPLES OF COMPUTER SECURITY
Assignment-6
Moulika Bollinadi

Introduction:

OWASP Security Shepherd is a web application for security training. It helps us to enhance the manual penetration testing skills. The Security Shepherd consists of challenges which are to be accomplished successfully and obtain the result key. There are different levels for challenges such as Field Training, Private, Corporal, Sergeant, Lieutenant, Major, Admiral. These levels are in increasing order with their toughness and are based on OWASP top 10 security vulnerabilities. I used Burp Suite proxy which can intercept HTTP protocol requests and responses.

Field Training:

1. **Insecure Direct Object References:** I clicked on Refresh your profile button and captured the traffic in burp suit and I saw the "username=guest" in the intercept. I changed the username to admin and forwarded the request. I observed that I was able to obtain the result key. It did not check if the user was authorized and just obtained the result key when I manipulated the username.
2. **Poor Data Validation:** I first tested by entering "1" in the box and it said that "The number 1 is a valid number". When I changed the number to "-1", it said "Invalid Number: Number must be greater than 0". I understood that, the number needs to be changed to a negative number to gain the result key. So, I used burp suite, captured traffic by entering 1, changed 1 to -1 and forwarded the request. I was successfully able to obtain the result key.
3. **Security Misconfiguration:** I tried using few combinations of default usernames and passwords to sign-in. Few include: admin/admin, admin/ , admin/password etc. I observed that I was able to obtain the result key using "admin/password" combination.
4. **Broken Authentication and Session Management:** In-order to trick the server thinking the session was complete, I understood that we need to check what is the token value when we click "complete This Lesson". I captured the traffic in burp suite and observed that there was a cookie named "lessonComplete = lessonNotComplete". I changed this cookie value to "lessonComplete" and forwarded the request. I was successfully able to obtain the result key.
5. **Failure to Restrict URL Access:** I right clicked on "web page" and opened inspect. I observed the source code of the web page and one line said "href=adminOn;y/resultKey.jsp". I know that the result key is hidden there. In-order to make it accessible, I changed the display:none to display:text. I was able to see the "Administrator Result Page" which has the result key.
6. **Cross Site Scripting (XSS):** I tried all the given commands and was able to read XSS for "<IMG SRC="#" ONERROR="alert('XSS')"/>".
7. **Cross Site Scripting one:** I first wanted to try the commands displayed for XSS. I used "<SCRIPT>alert('XSS')</SCRIPT>" and was able to read the XSS comment box.

Private:

1. **Insecure Cryptographic Storage:** I decoded the given encoded base64 key to obtain the result key.
2. **SQL Injection:** I tried using common names to see if I could get some table information. But I couldn't. Later, I used (admin' or '1=1) but not closing the single quotes. This gave me the results of every row in the table.
3. **Insecure Cryptographic Storage Challenge 1:** It was given that, the result key has been encrypted with a famous, but easily broken, Roman cipher. I googled the roman cipher decryption and obtained the result key.
4. **Insecure Direct Object References Challenge One:** I tried to inspect the page and observed that all the user ID numbers are odd. The next odd value was 11 and I changed it to 11 and observed the result key.
5. **Poor Validation One:** I tried "1" as inputs for all and placed the order. Later, I tried "-1" as inputs for all and observed that the validation is poor. I made the total amount of the order 0$ by adding -200 not bad and 1 troll. I obtained the result key.
6. **SQL Injection One:** I used the common SQL injection command ' or '1'='1 but I couldn't get the results. I later tried the same command using double quotes i.e.; " or "1"="1 and was successfully able to obtain the result key.
7. **Session Management Challenge One:** I captured the traffic after clicking the button. I observed a Boolean statement with false and changed it to true. I was not able to get the key but instead it said, "Hacker Detected". I started to intercept again and carefully observed the traffic. As a hacker's point of view, I know that we need to look at the checksums to make sure if something is hashed. I used to decoder and tried decoding the checksum with Base64, MD5 and observed that MD5 obtains userRole=admin. I re-hashed everything to userRole=administrator and replaced in the checksum. I forwarded the request and was able to obtain the result key.
8. **Cross-Site Request Forgery:** I obtained the result key using- https://kernighan.uncg.edu/root/grantComplete/csrfLesson?userId=1635763706. The URL is obtained from the source code of <img> near GET.

Corporal:

1. **Unvalidated Redirects and Forwards:** I first tried using kernighan.uncg.edu in the below URL. But I couldn't get the result key. Later, I tried using my system IP address in the below URL and was able to obtain the result key. https://192.168.56.1/user/redirect?to=https://192.168.56.1/root/grantComplete/unvalidatedredirectlesson?userid=1702206074
2. **SQL Injection 2:** Given that, we need to enter the customer email. I know that email generally consists of "@". I tried using @' or '1'= '1, 1 @ 1. 1' or '1'='1 and couldn't obtain the result key. I later tried, '!='1@1.1 and obtained the key.
3. **NoSQL Injection Challenge One:** I intercepted the traffic in burp suite and changed theCareerName value to 'OR'1'='1. I was not able to succeed. I took the reference of OWASP testing guidelines to understand how to write the query to exploit and understood that we can add some special characters. I used x';return(true);var x='x and obtained the result key.

Sergeant:

1. **CSRF-1:** According to the given description, I used GET command and tried to obtain the result key. I was not successful. I then used the URL method instead of using GET as the command looked like an URL. I used, https://kernighan.uncg.edu/user/csrfchallengeone/plusplus?userid= 0543e8334df0810612988649900ec11670307d0b. I successfully obtained the result key.
2. **Session Management Challenge 3:** I first did a hit and trial to see if there is any username "admin". I observed that there is a admin username. I changed the password and intercepted in the burp suite. I did not observe any major difference, but I wanted to decode checksum and current values using Base64. The current value was encoded twice using Base64 and I decoded it twice. I observed the decoded answer as guest12. I understood that guest12 is the sub-user and I've to replace it with admin to get the result key. So, I encoded the admin word two times using Base64 and replaced in the current. I forwarded the request and observed that the username is admin and password is 12345678 (I set it). I logged in with the credentials and obtained the result key.
3. **Cross Site Scripting Two:** As I know, I have to display XSS dialogue box, I used different types of input types and sources for testing. I observed in the source code that script is replaced with scr.pt. I figured out that ONSELECT is not blocked and used it along with the <IMG SRC = "#" ONSELECT = "alert('XSS')"/>. I successfully obtained the result key.
4. **Insecure Direct Object Reference Challenge Two:** I decoded the user ID values using MD5 and observed that all are prime numbers such as 2,3,5,7,11. I used the next prime number 13, encoded it using MD5 and inserted in the User ID. I forwarded the request and obtained the result key.
5. **Cross Site Scripting 3:** I used the same command that I used for challenge 2. I see that the vulnerability method taken care. I tried different methods but couldn't succeed. I later found a reference in stack exchange and understood that security can be bypassed by using attributes several times. I implemented the procedure for command in challenge 2 and obtained the result.

Lieutenant:

1. **Insecure Direct Object Reference Bank Challenge:** I created a sample account with username/password: iammoulika/12345678. I logged into the account and observed that my funds are 0.0. I transferred funds using receiver A/C number/Amount as iammoulika/1000 and intercepted the packets in burp suite. I observed the senderAccountNumber=46 & receiverAccountNumber=iammoulika & transferamount=1000. I understood that, the bank doesn't validate between A/C number and Account Holder name. So, I changed senderAccountNumber to anything other than 46, receiverAccountNumber = 46 (because I want to receive the amount) & transferamount=5001000 and transferred the request. I observed that amount was transferred successfully to my account. I logged out and logged in again to obtain the result key. Initially, I did not understand where the trick was. But when I started tampering with A/C numbers, I understood that, the bank does not check for A/C names.

2. **SQL Injection 3:** I understood that I need to obtain the credit card number of customer name Mary Martin which is a result key. I first tried to obtain the user details in the table by using the general SQL command b'or'b'='b and observed the results. I used the hint because of many trials and used the command " Mary Martin' UNION SELECT creditcardnumber FROM customers WHERE customername = 'Mary Martin ". I successfully obtained the full table results and result key value.

Major:

1. **Cross Site Scripting Four:** As I need to enter the URL, by considering the previous XSS challenges, I used http" ONERROR=alert('XSS'). I could not find the result key. So, I tried using small letter http"onerror=alert('XSS'). It gave me a 404 error. I used all different combinations of capital and small letters and figured out that http"oNerror=alert('XSS') gave me the result key. I assumed to use different combinations because ON has been encoded with HTML for escape character.
2. **SQL Injection 4:** As soon as I saw two boxes for the challenge, I assumed that I've insert different SQL query combinations in password. Because it said I need to successfully login as an administrator to obtain the result key. I used username "admin" and password 'or'a'>'a';--. I later on, tried using backslash (\) for the username to escape the (') and used 'or username="admin";-- for password. The reason for specifically mentioning admin as username in password because the challenge was predicting when I used "1's or a's" and not allowing me to get the key.
3. **Poor Data Validation Two:** Because it is about data validation, I applied the same trick that I used for challenge one. I observed that, it validates for a negative number. So, I tried using different methods decimals, 0, 999 but nothing was successful. I went on incrementing the numbers randomly for the boundary conditions and obtained the result key for 99999999. If I increment another 9, it was giving me a positive value again.
4. **Cross Site Scripting 5:** I used http://a"" onselect=alert('XSS') .b.c which would typically mean "http://www.google.com". I did not complete the double quotes in-order to make sure the injection happens successfully and obtained the result key.

Admiral:

1. **Cross Site Scripting 6:** I used same command as challenge 5 as a hit and trial and was unfortunately able to obtain the result.

Conclusion: By performing this flag contest in OWASP Security Shepherd, I learned how to exploit OWASP top 10 security vulnerabilities depending upon their toughness. I understood how each vulnerability is being tested and how the hackers change the information. I successfully obtained the result key for different vulnerabilities such as Insecure Direct Object Reference, Poor Data Validation, Cross Site Scripting, Cross Site Request Forgery, SQL Injection, NoSQL Injection, Unvalidated References and Forwards, Insecure Cryptographic Storage, Failure to Restrict URL access, Security misconfigurations, Broken Authentication and Session Management. I also learned how to use Burp Suite proxy tool to intercept and change the information for few challenges.