CU CHANDIGARH UNIVERSITY
DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
Discover. Learn. Empower.

Experiment: 2.3

Student Name: Yash Kumar **UID**: 20BCS9256
Branch: BE-CSE **Section/Group:** 616 'B'
Semester: 5th **Subject Name**: WMS Lab
Subject Code: 21CSP-338

Aim:

Implementation of Session hijacking attack on http-enabled website.

Objective:

To Identify vulnerable session cookies.

Software/Hardware Requirements:

Burp Suite PC

Introduction:

Session Hijacking

The Session Hijacking attack consists of the exploitation of the web session control mechanism, which is normally managed for a session token. Because http communication uses many different TCP connections, the web server needs a method to recognize every user's connection. The most useful method depends on a token that the Web Server sends to the client browser after a successful client authentication.

The Session Hijacking attack compromises the session token by stealing or predicting a valid session token to gain unauthorized access to the Web Server.

The session token could be compromised in different ways; the most common are:

- Predictable session token;
- Session Sniffing;
- Client-side attacks (XSS, malicious JavaScript Codes, Trojans, etc.)

- Man-in-the-middle attack
- Man-in-the-browser attack

Steps/Method/Coding:

Step-1: -Ensure that Burp is correctly configured with your browser and intercept is turned off in proxy Intercept tab.

Step-2: -Visit the login page of the application you are testing in your browser and Log in to the application you are testing.

For example: -You can log in using the credentials user: user.

Step-3: -Now, In the Proxy Intercept tab, ensure "Intercept is on”. Refresh the page in your browser. Then request will be captured by Burp, it can be viewed in the Proxy "Intercept" tab.

Step-4: -We now need to investigate and edit each individual cookie.

Right click anywhere on the request and click "Send to Repeater".

Go to the Repeater tab. The cookies in the request can be edited easily in the "Params" tab.

Step-5: -By removing cookies from the request, we can ascertain the function of each cookie.

Cookies can be edited in the Request "Params" table.

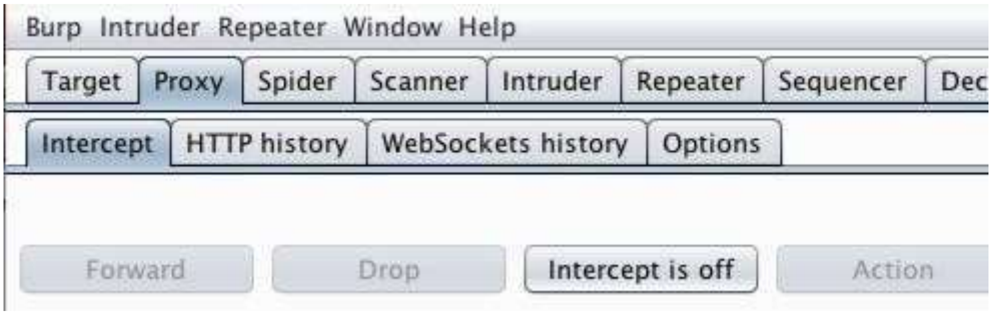
In this example we have altered the value of the "uid" cookie to 1. Alter the value

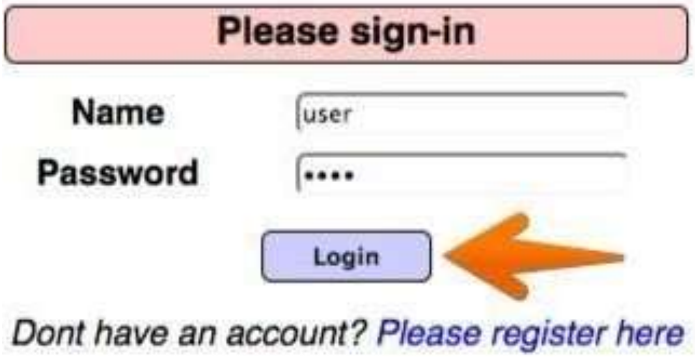
then click the "Go" button.

Step-6: -The response from the server can be viewed in the "Response" panel in Repeater. The response shows that by altering the "uid" cookie we have logged in to the application as "admin".

We have used cookies to manipulate the session and access another account with elevated privileges.

Output screenshot:



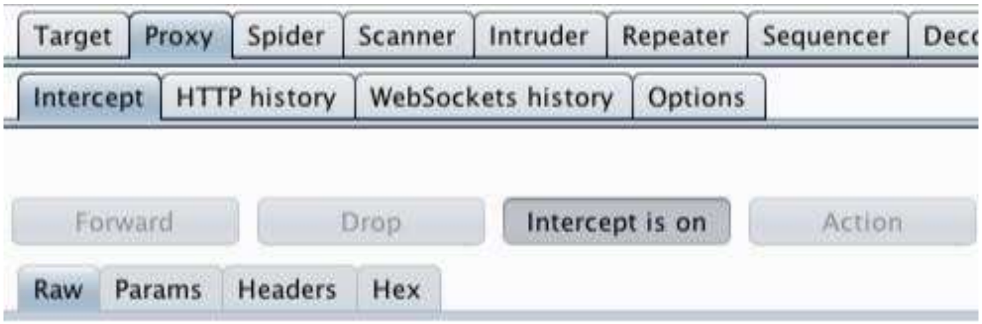


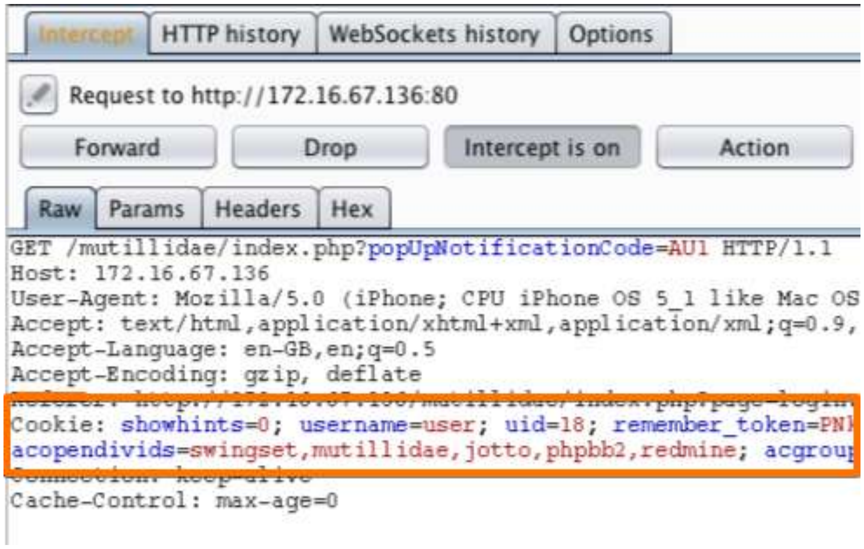
Please sign-in

Name: user

Password: ••••

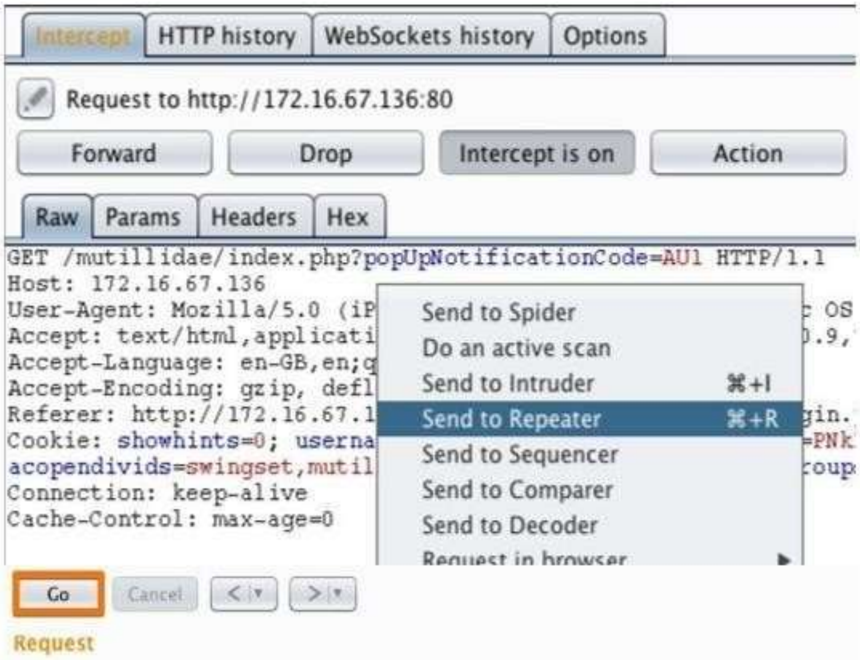
Dont have an account? Please register here





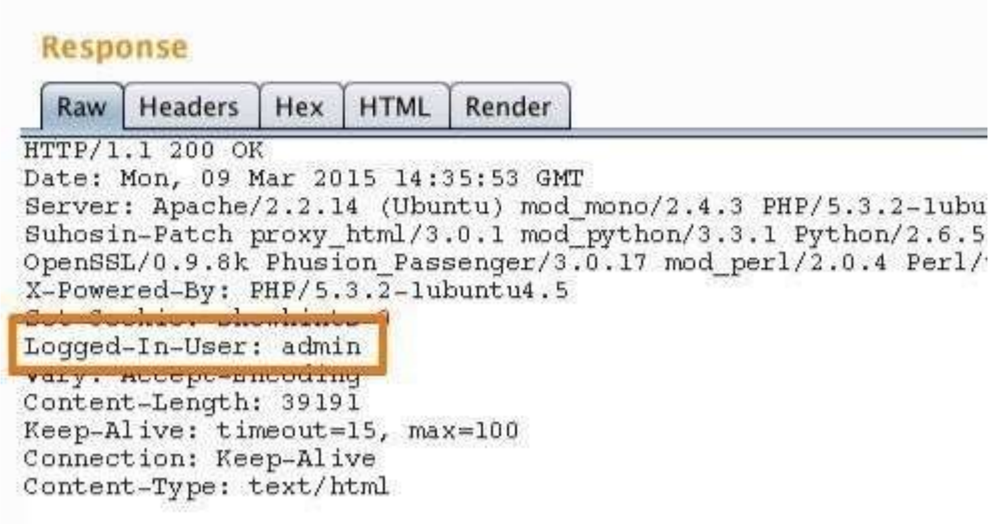
Request to http://172.16.67.136:80

```
GET /mutillidae/index.php?popUpNotificationCode=AU1 HTTP/1.1
Host: 172.16.67.136
User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 5_1 like Mac OS
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://172.16.67.136/mutillidae/index.php?page=login
Cookie: showhints=0; username=user; uid=18; remember_token=PNk
acopendivids=swingset,mutillidae,jotto,phpbb2,redmine; acgroup
Connection: keep-alive
Cache-Control: max-age=0
```



Request to http://172.16.67.136:80

```
GET /mutillidae/index.php?popUpNotificationCode=AU1 HTTP/1.1
Host: 172.16.67.136
User-Agent: Mozilla/5.0 (iP                                     c OS
Accept: text/html,applicati                                     0.9,
Accept-Language: en-GB,en;q
Accept-Encoding: gzip, defl
Referer: http://172.16.67.1                                     gin.
Cookie: showhints=0; userna                                     =PNk
acopendivids=swingset,mutil                                     coup
Connection: keep-alive
Cache-Control: max-age=0
```



```
HTTP/1.1 200 OK
Date: Mon, 09 Mar 2015 14:35:53 GMT
Server: Apache/2.2.14 (Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubu
Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 Python/2.6.5
OpenSSL/0.9.8k Phusion_Passenger/3.0.17 mod_perl/2.0.4 Perl/
X-Powered-By: PHP/5.3.2-1ubuntu4.5
Set-Cookie: showhints=0
Logged-In-User: admin
Vary: Accept-Encoding
Content-Length: 39191
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html
```

Learning Outcomes:

In the above experiment we have learnt that using session hijacking attack how the token session can be manipulated.

Note- *This work with different type of software to do hijacking on servers and websites.*