ATTACK DEFENSE

by PentesterAcademy

Name	Transaction Replay II
URL	https://attackdefense.com/challengedetails?cid=1405
Type	REST: JWT Advanced

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Step 1: Check the IP address of the machine.

Command: ifconfig

```
root@attackdefense:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.1.1.6  netmask 255.255.255.0  broadcast 10.1.1.255
        ether 02:42:0a:01:01:06  txqueuelen 0  (Ethernet)
        RX packets 576  bytes 102448 (100.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 627  bytes 2506537 (2.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.215.101.2  netmask 255.255.255.0  broadcast 192.215.101.255
        ether 02:42:c0:d7:65:02  txqueuelen 0  (Ethernet)
        RX packets 20  bytes 1584 (1.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 889  bytes 1863682 (1.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 889  bytes 1863682 (1.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@attackdefense:~#
```

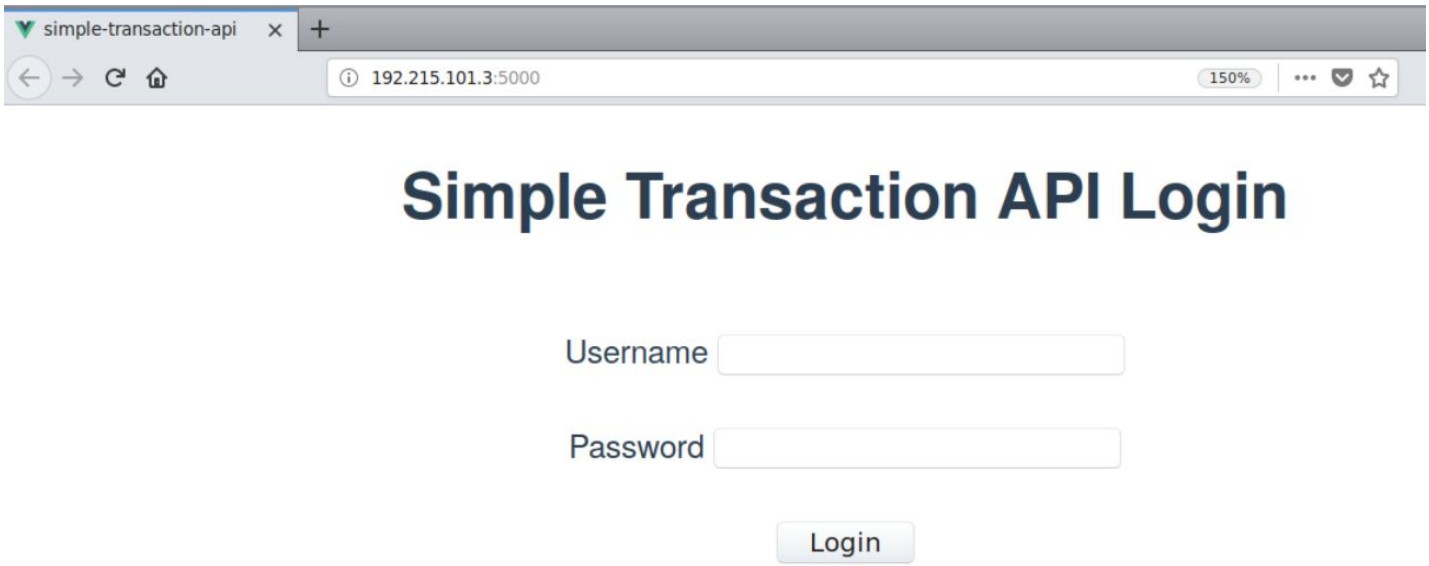
The IP address of the machine is 192.215.101.2.

Therefore, the bank transaction API is running on 192.215.101.3, at port 5000.

Step 2: Viewing the Transaction API.

Open the following URL in firefox.

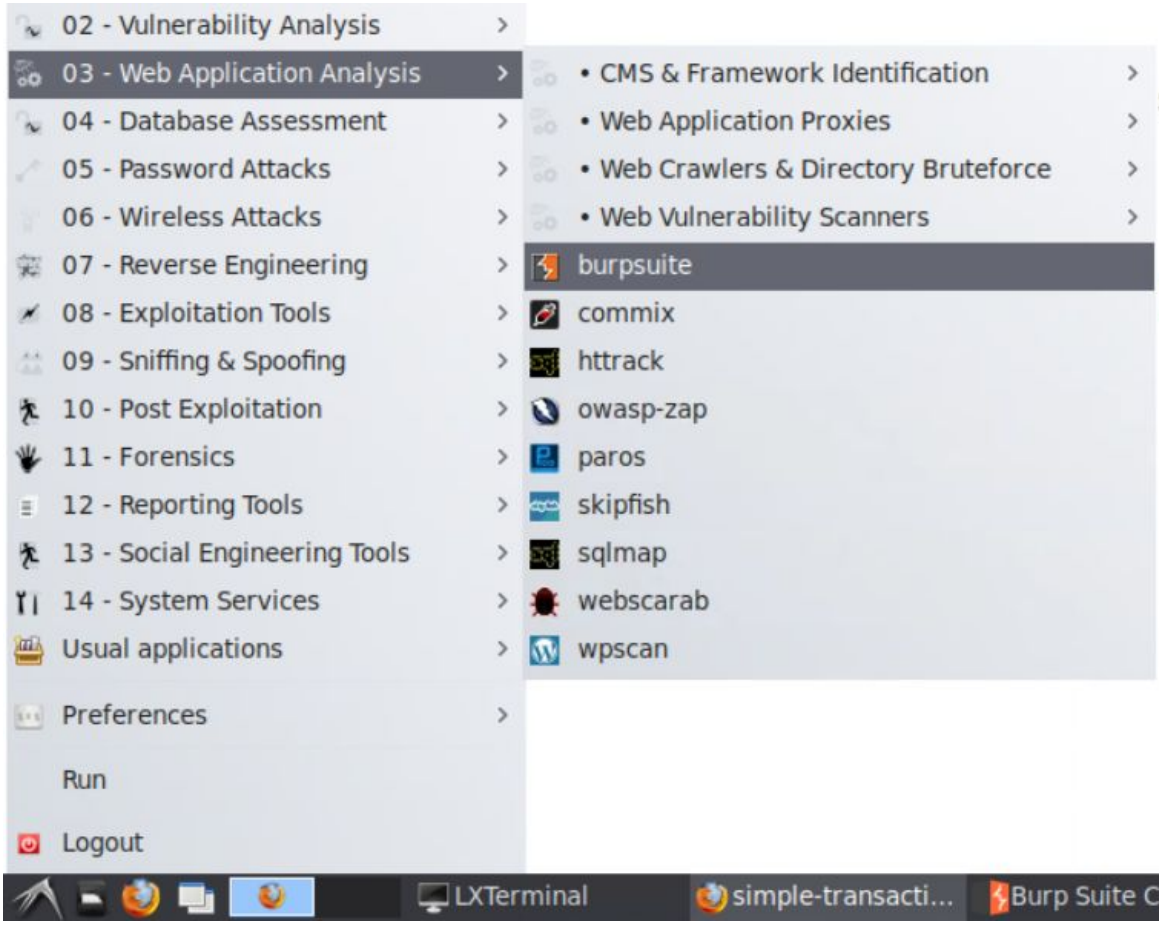
URL: http://192.215.101.3:5000



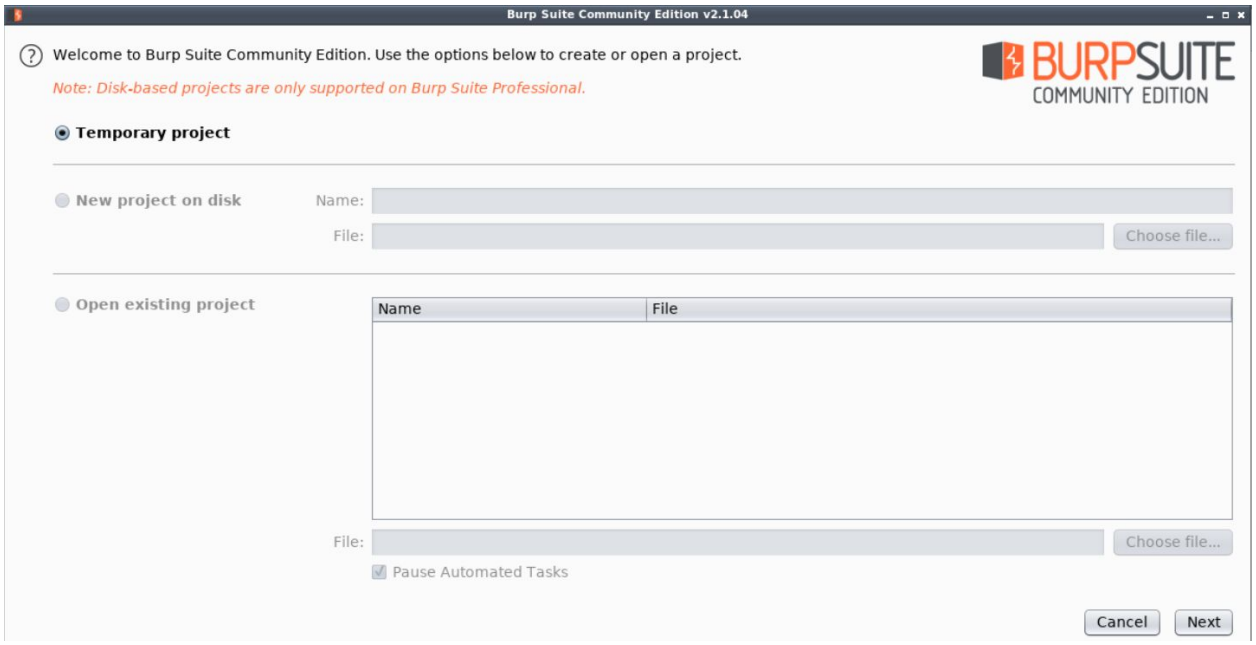
Step 3: Configuring the browser to use BurpSuite proxy and making BurpSuite intercept all the requests made to the API.

Launch BurpSuite.

Select Web Application Analysis > burpsuite

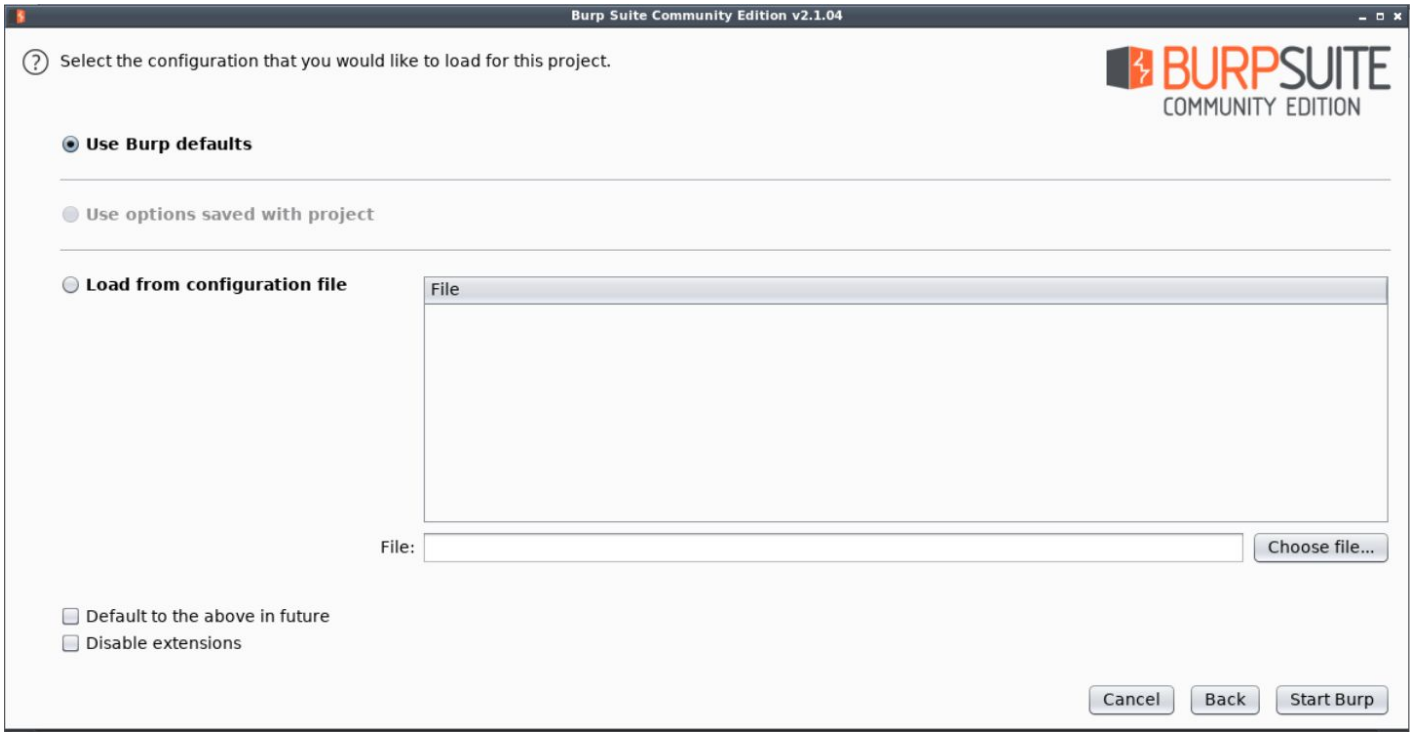


The following window will appear:

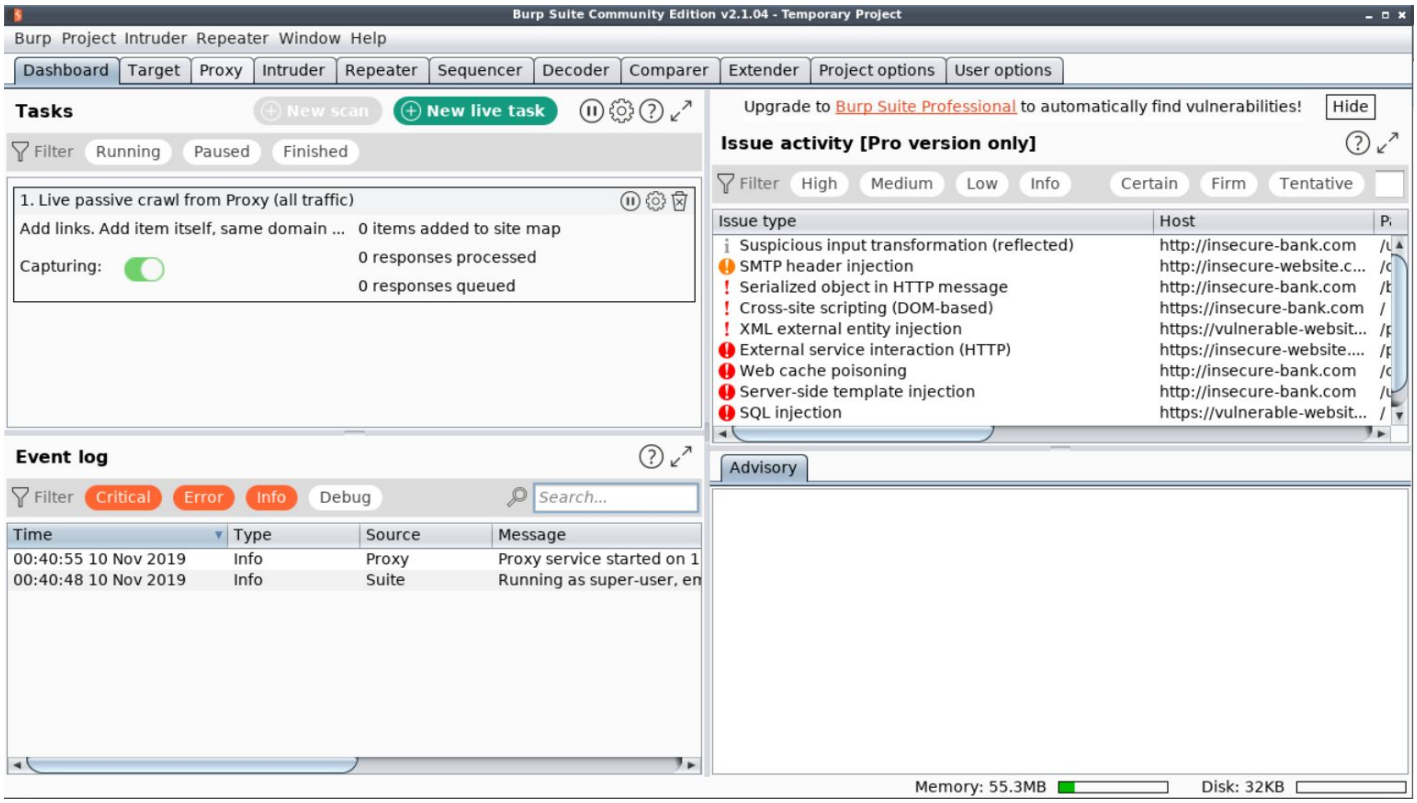


Click Next.

Finally, click "Start Burp" in the following window:

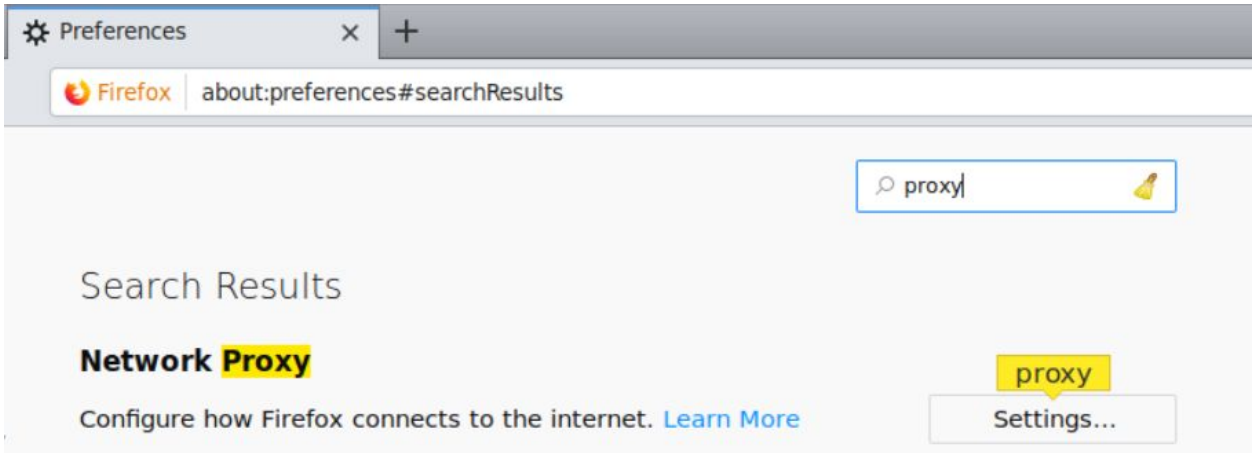


The following window will appear after BurpSuite has started:

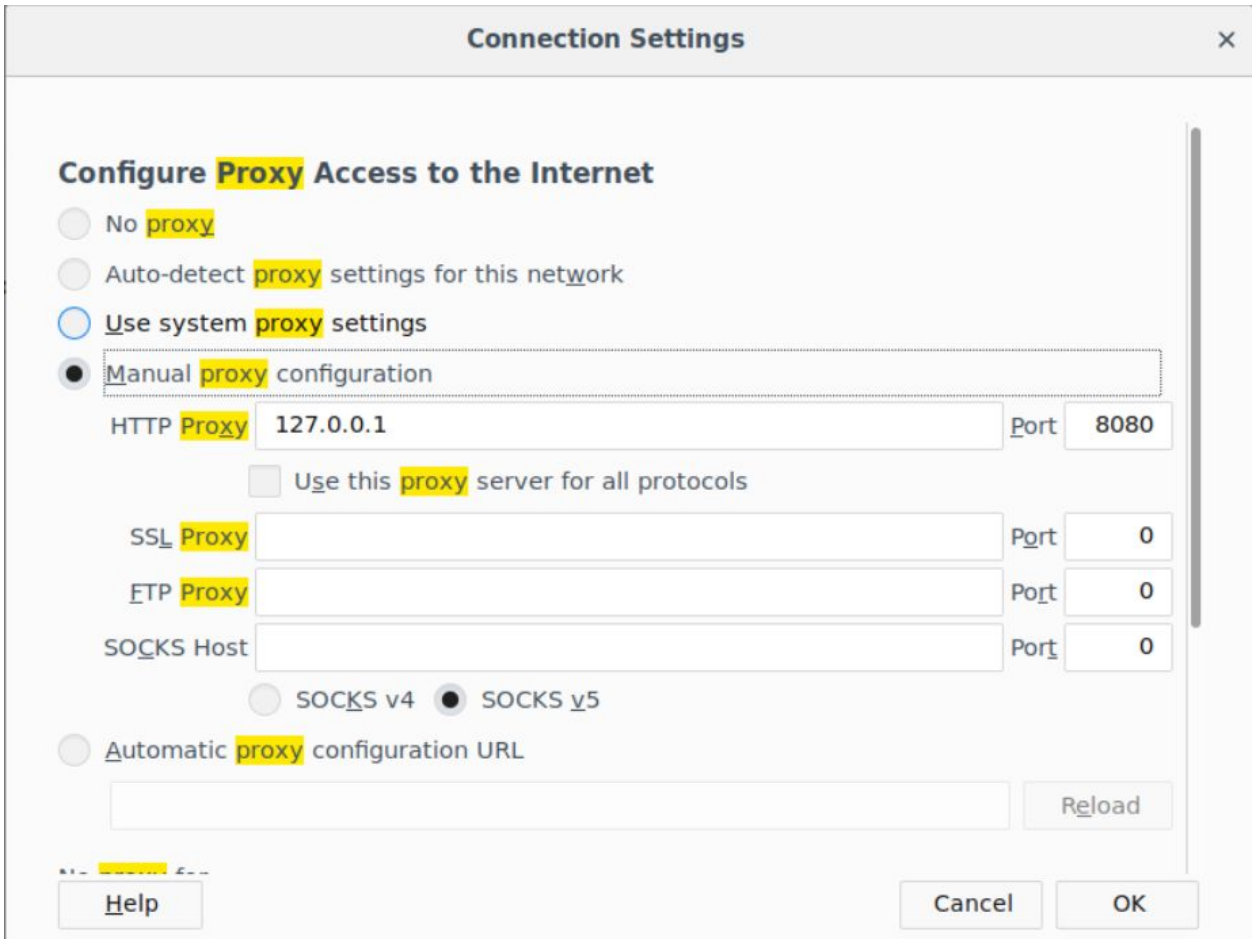


Configure the browser to use the Burp proxy listener as its HTTP Proxy server.

Open the browser preference settings and search for network proxy settings.



Select Manual Proxy Configuration and set the HTTP Proxy address to localhost and the port to 8080.



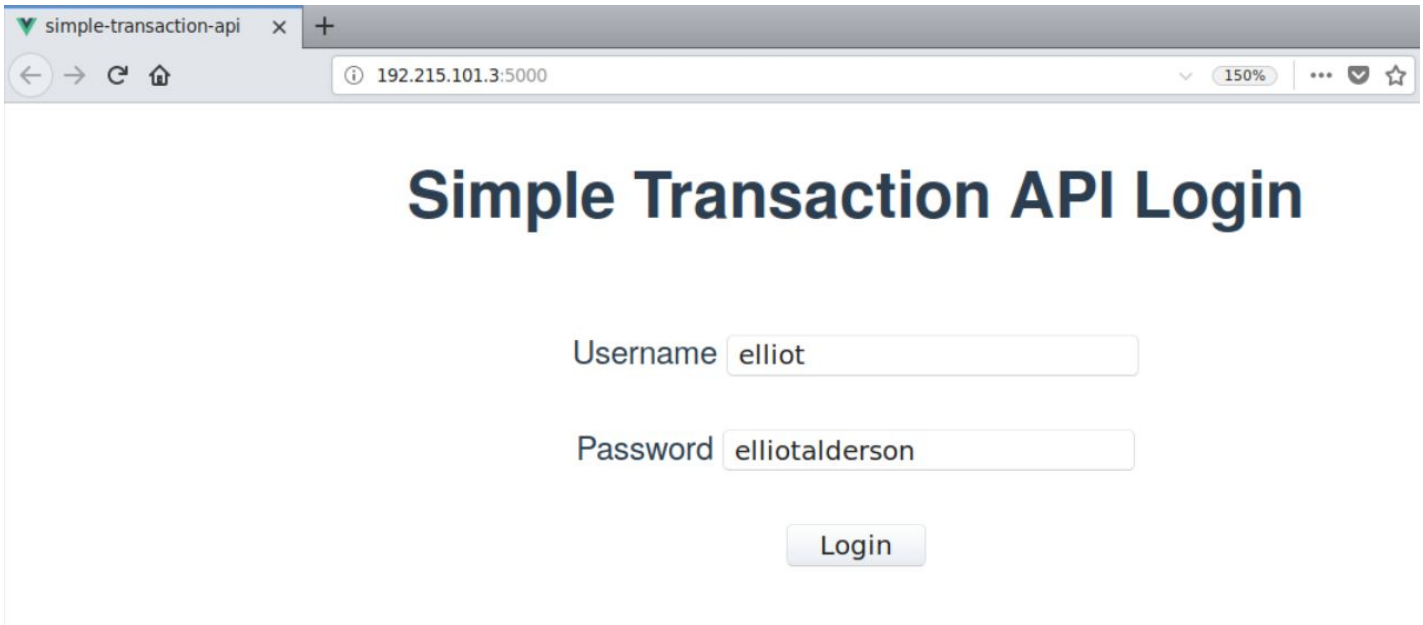
Click OK.

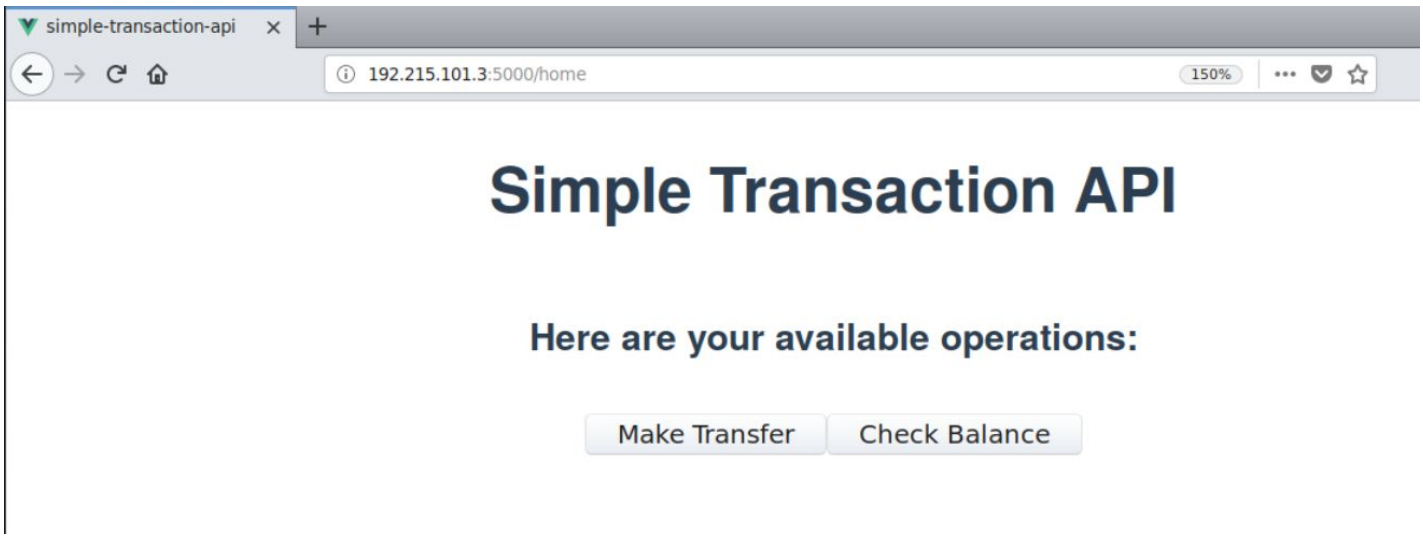
Everything required to intercept the requests has been setup.

Step 4: Interacting with the Transaction API and understanding its flow.

Login using the provided credentials:

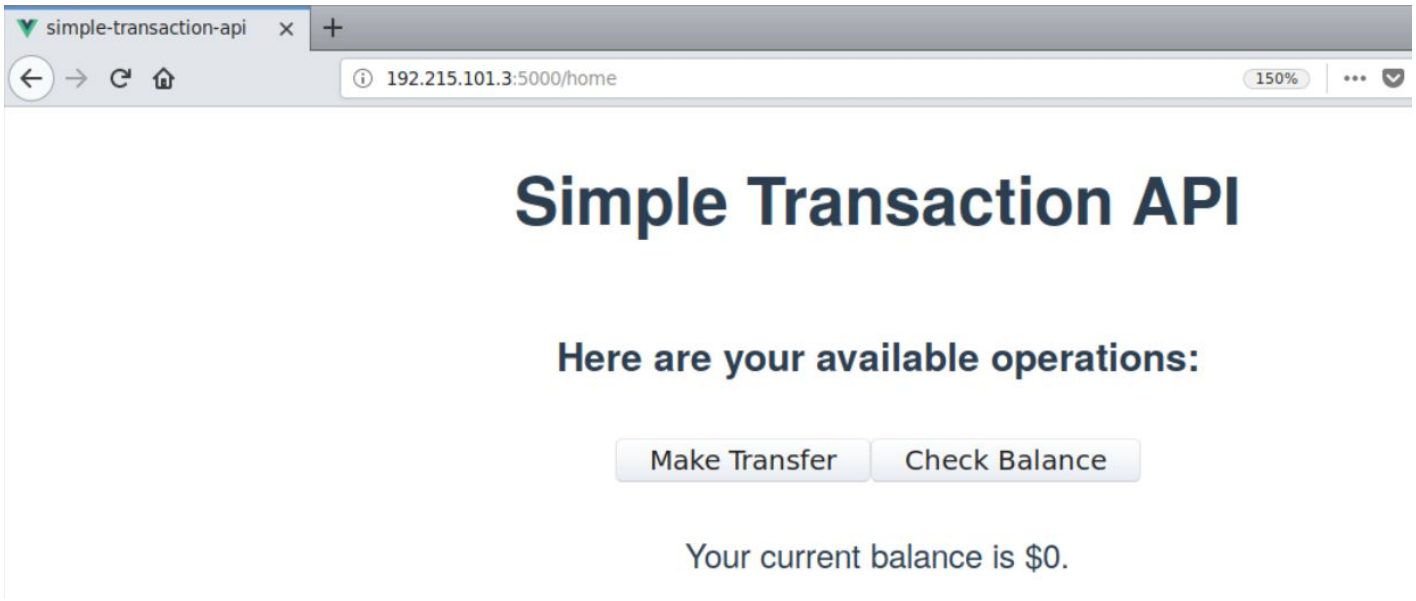
Username: elliot
Password: elliotalderson



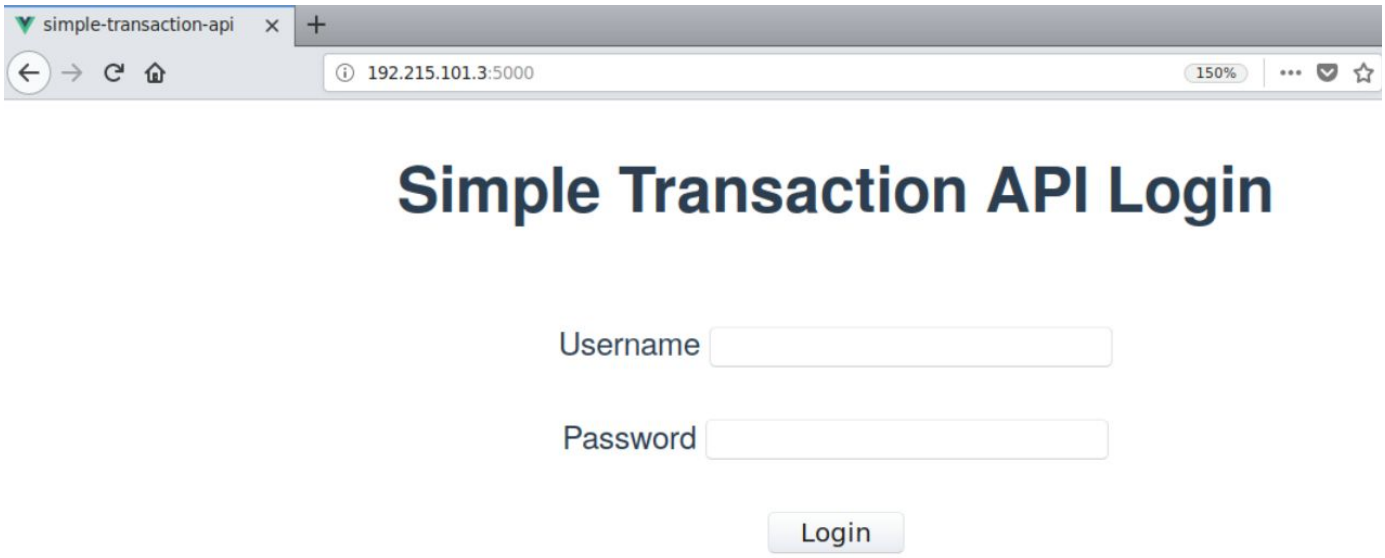


Note: Make sure that intercept mode in BurpSuite is off.

Click on “Check Balance” to check the current account balance.

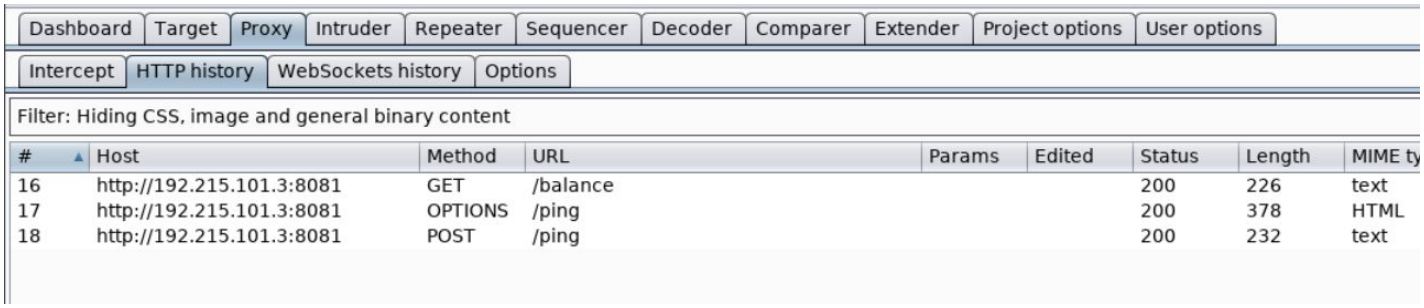


Click on "Make Transfer".



On clicking "Make Transfer", the user logs out.

Check the HTTP history in BurpSuite:



The first request is to get the balance:

Intercept | HTTP history | WebSockets history | Options

Filter: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status	Length	MIME type
16	http://192.215.101.3:8081	GET	/balance			200	226	text
17	http://192.215.101.3:8081	OPTIONS	/ping			200	378	HTML
18	http://192.215.101.3:8081	POST	/ping			200	232	text

Request | Response

Raw | Headers | Hex | Render

```
HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 1
Access-Control-Allow-Origin: http://192.215.101.3:5000
Vary: Origin
Server: Werkzeug/0.16.0 Python/2.7.15+
Date: Thu, 14 Nov 2019 08:48:06 GMT

0
```

The response, 0 in this case is the current user balance.

Analyze the request and response for "/ping":

This was the final request to the API and this would have logged the user out.

Request:

Intercept | HTTP history | WebSockets history | Options

Filter: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status	Length	MIME type
16	http://192.215.101.3:8081	GET	/balance			200	226	text
17	http://192.215.101.3:8081	OPTIONS	/ping			200	378	HTML
18	http://192.215.101.3:8081	POST	/ping			200	232	text

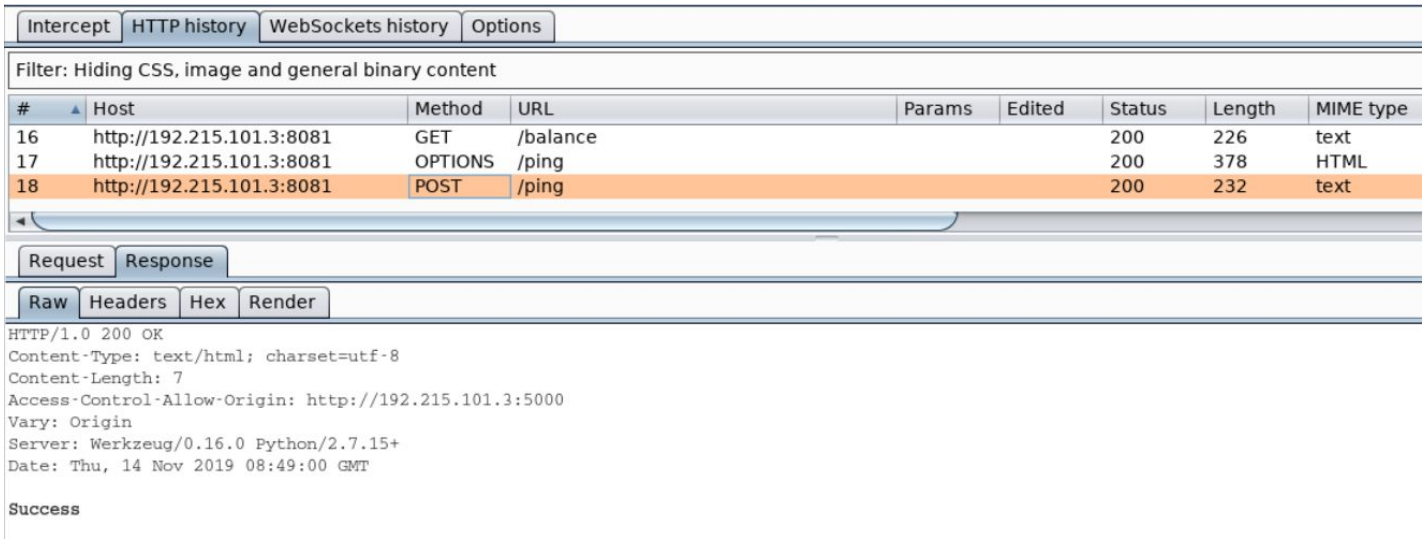
Request | Response

Raw | Headers | Hex

```
POST /ping HTTP/1.1
Host: 192.215.101.3:8081
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.215.101.3:5000/home
Content-Type: application/json;charset=utf-8
Content-Length: 2
Origin: http://192.215.101.3:5000
Connection: close

{}
```

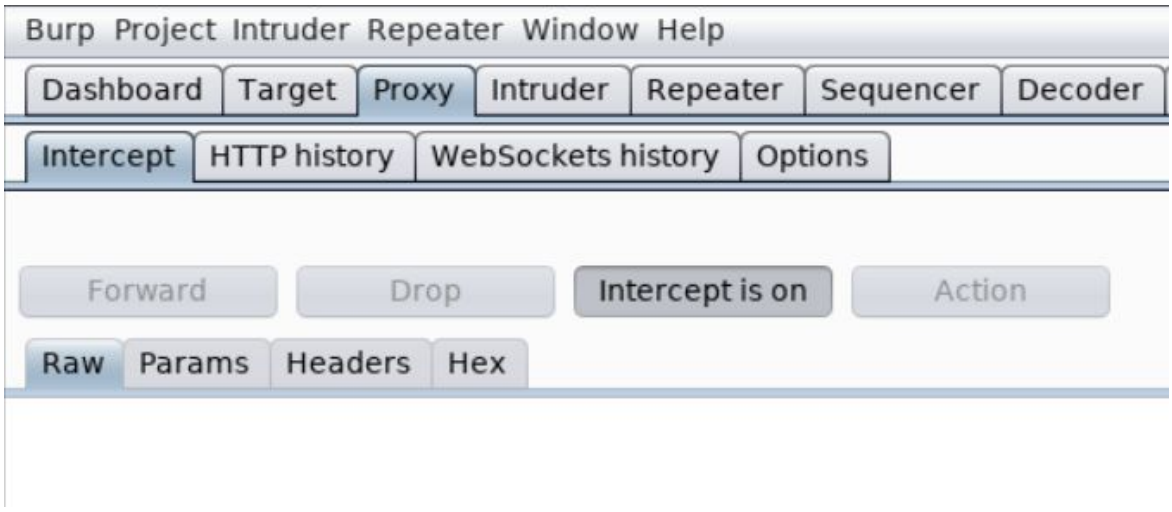
Response:



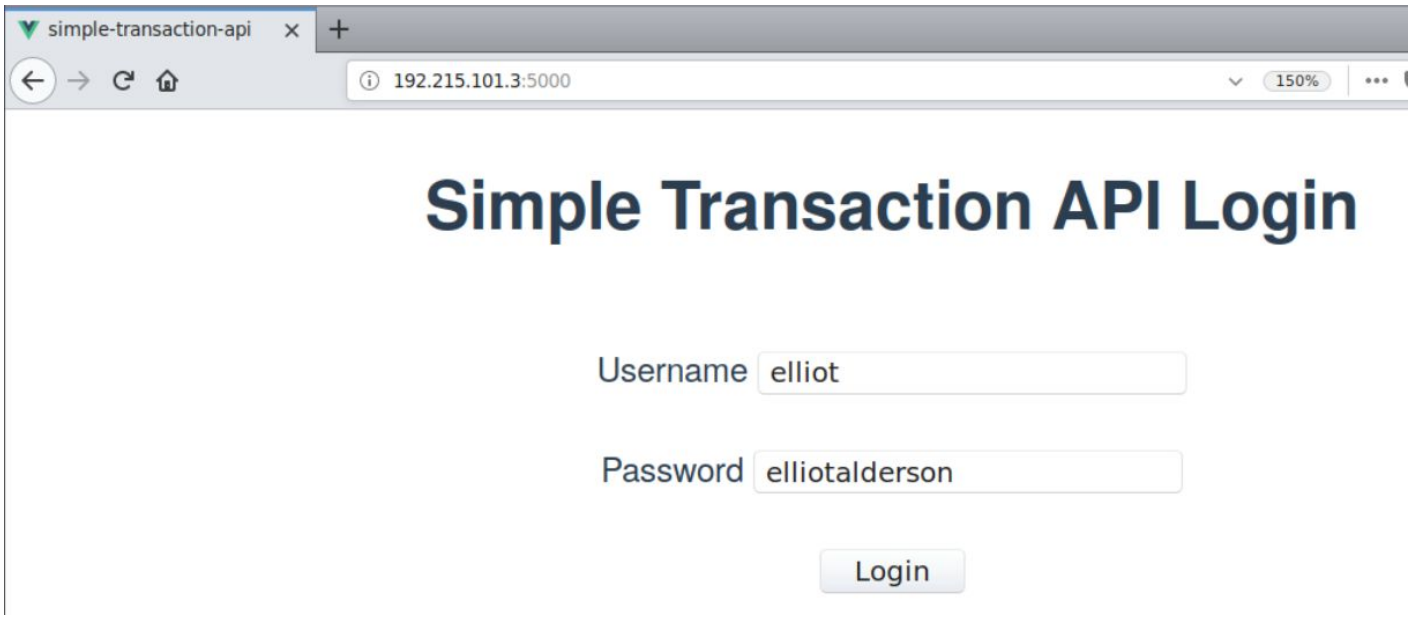
The response returns a "Success" message.

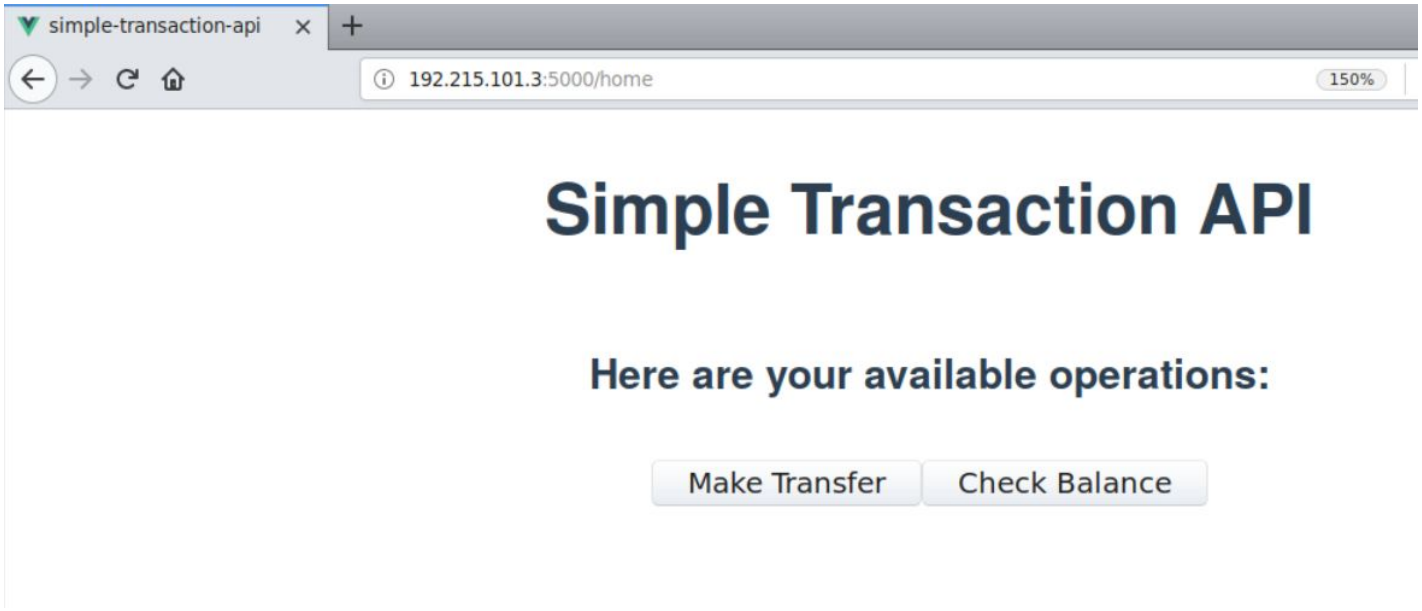
This most probably indicates that the user was successfully logged out.

Turn on the intercept mode in BurpSuite and block any request to /ping.



Login to the webapp again.





Click on "Make Transfer" again.

Dashboard | Target | Proxy | Intruder | Repeater | Sequencer | Decoder | Comparer

Intercept | HTTP history | WebSockets history | Options

Request to http://192.215.101.3:8081

Forward | Drop | Intercept is on | Action

Raw | Headers | Hex

```
OPTIONS /ping HTTP/1.1
Host: 192.215.101.3:8081
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Access-Control-Request-Method: POST
Access-Control-Request-Headers: content-type
Origin: http://192.215.101.3:5000
Connection: close
```

Drop the above request to "/ping" otherwise it would issue a logout request.

Dashboard | Target | Proxy | Intruder | Repeater | Sequencer | Decoder | Comparer

Intercept | HTTP history | WebSockets history | Options

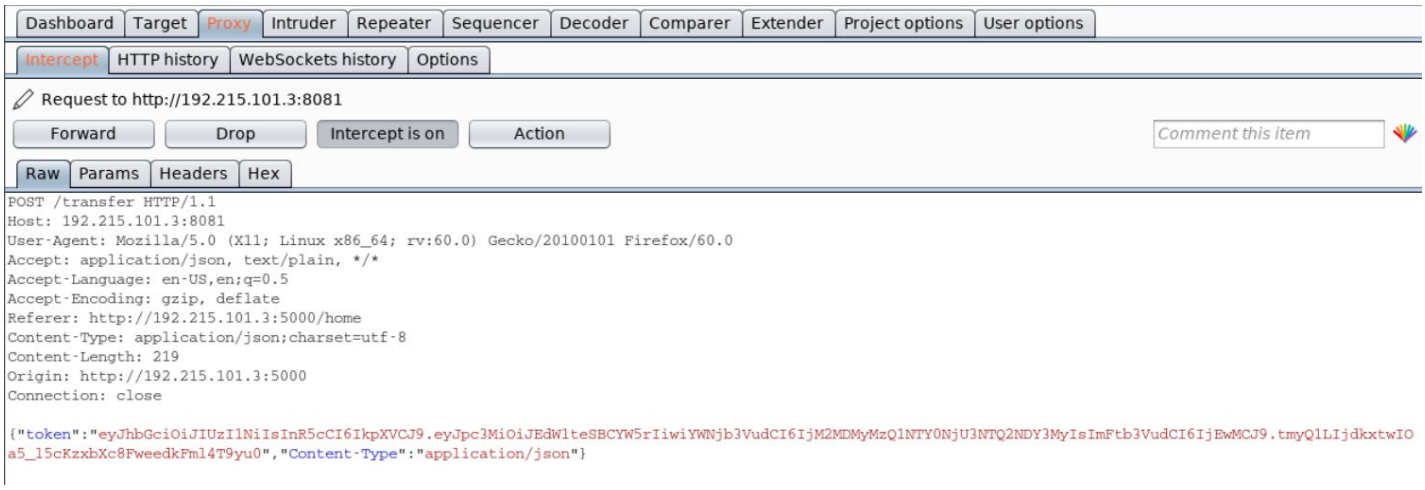
Request to http://192.215.101.3:8081

Forward | Drop | Intercept is on | Action

Raw | Headers | Hex

```
OPTIONS /transfer HTTP/1.1
Host: 192.215.101.3:8081
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Access-Control-Request-Method: POST
Access-Control-Request-Headers: content-type
Origin: http://192.215.101.3:5000
Connection: close
```

Notice the next request is to /transfer. Forward this request.



The next request is a POST request to /transfer.

This request contains a JWT Token.

JWT Token:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJEdW1teSBCYW5rIiwiYWNjb3VudCI6IjM2MDMyMzQ1NTY0NjU3NTQ2NDY3MyIsImFtb3VudCI6IjEwMCJ9.tmyQ1LIjdkxtwIOa5_l5cKzxbXc8FweedkFml4T9yu0

Decoding the payload part of the token using base64 utility:

Command: echo eyJpc3MiOiJEdW1teSBCYW5rIiwiYWNjb3VudCI6IjM2MDMyMzQ1NTY0NjU3NTQ2NDY3MyIsImFtb3VudCI6IjEwMCJ9 | base64 -d

```
root@attackdefense:~# echo eyJpc3MiOiJEdW1teSBCYW5rIiwiYWNjb3VudCI6IjM2MDMyMzQ1NTY0NjU3NTQ2ND
Y3MyIsImFtb3VudCI6IjEwMCJ9 | base64 -d
{"iss":"Dummy Bank","account":"360323455646575464673","amount":"100"}root@attackdefense:~#
root@attackdefense:~#
```

Note: Sometimes decoding the header or payload using base64 utility might result in an error. It happens because JWT token uses base64UrlEncode algorithm. It strips off all the "=" signs which serve as the padding character in base64 encoded data.

The token contains the following claims:

1. iss (Issuer) Claim - The name of the entity that issued the token.

2. account Claim - Identifies the account number of the bank user.
3. amount Claim - Identifies the amount that is to be transferred from the bank to the account holder.

Note: The account and the amount are non-standard claims. They are not the part of JWT specs.

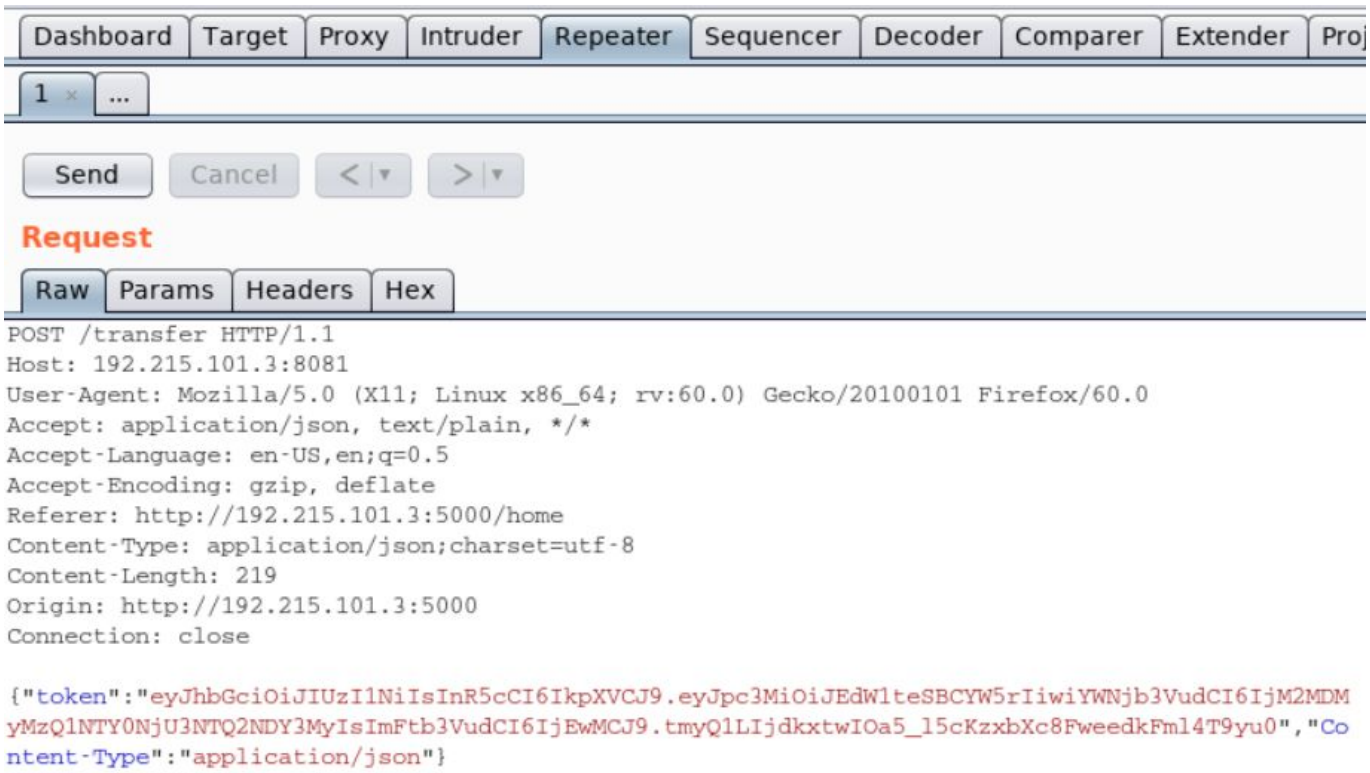
Using this token, the bank transfers $100 to the sender.

Information: The JTI (JWT ID) claim provides a unique identifier for a JWT Token. It can be used to prevent the token from being replayed.

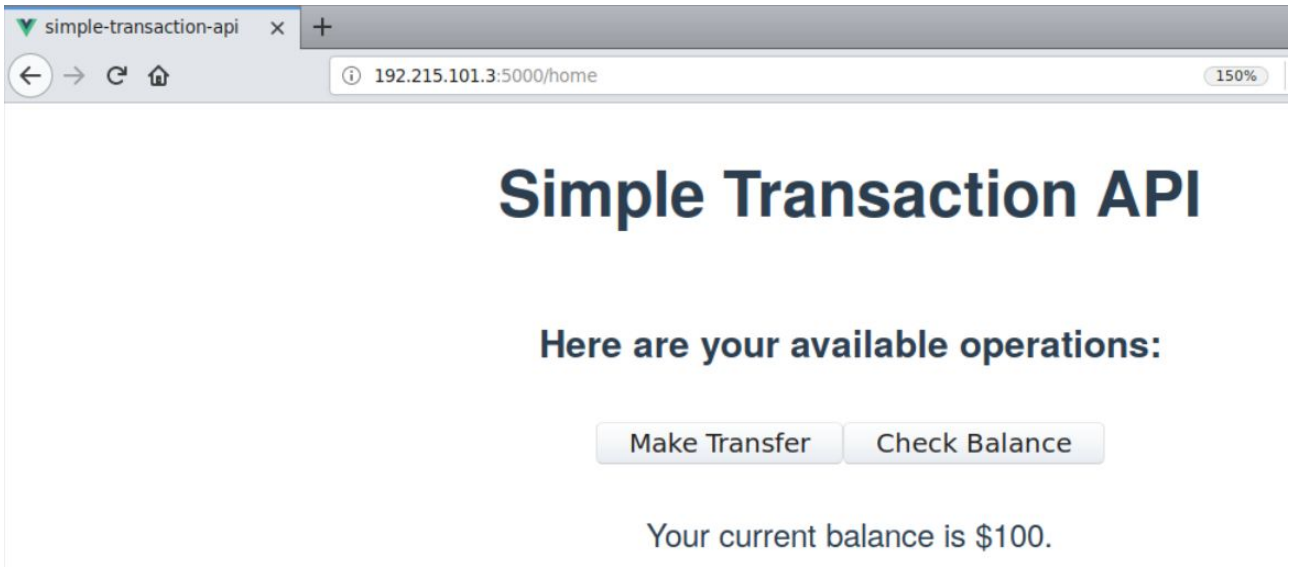
Since there is no JTI claim associated with the token, this token could be used in the subsequent requests to increase the account balance further (a replay attack against JWT Tokens).

Step 5: Retrieving the Golden Ticket from the bank server.

Send this transaction request to repeater and turn off the intercept mode.



Check the balance in the browser.



The current balance has become $100.

As previously mentioned, since there is no JTI field in the token, the request could be replayed to increase the account balance.

Send the /transfer request (sent to the repeater) multiple times and notice the response. The response reflects the current balance of the sender.

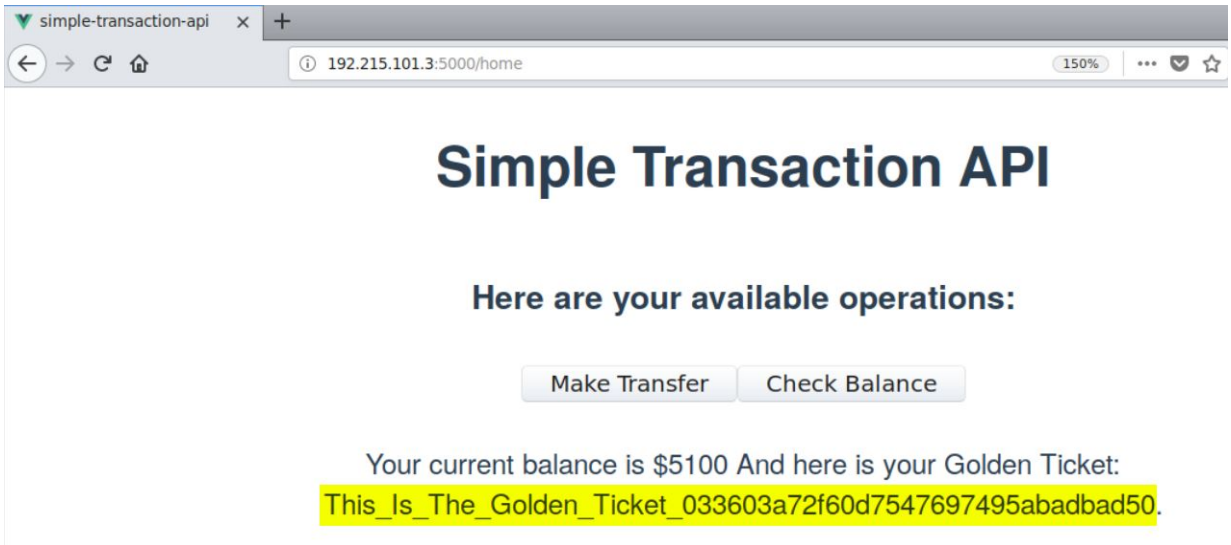
The current balance is $800 in this case, as shown in the Response window.

Issue the same request repeatedly until the balance exceeds $5000.

The balance has become $5100, as reflected in the Response window.

Since the current balance has exceeded $5000, the golden ticket could be retrieved from the server.

Navigate to the browser window and check the balance (click on "Check Balance" button).



Golden Ticket: This_Is_The_Golden_Ticket_033603a72f60d7547697495abadbad50

References:

1. JWT RFC (https://tools.ietf.org/html/rfc7519)