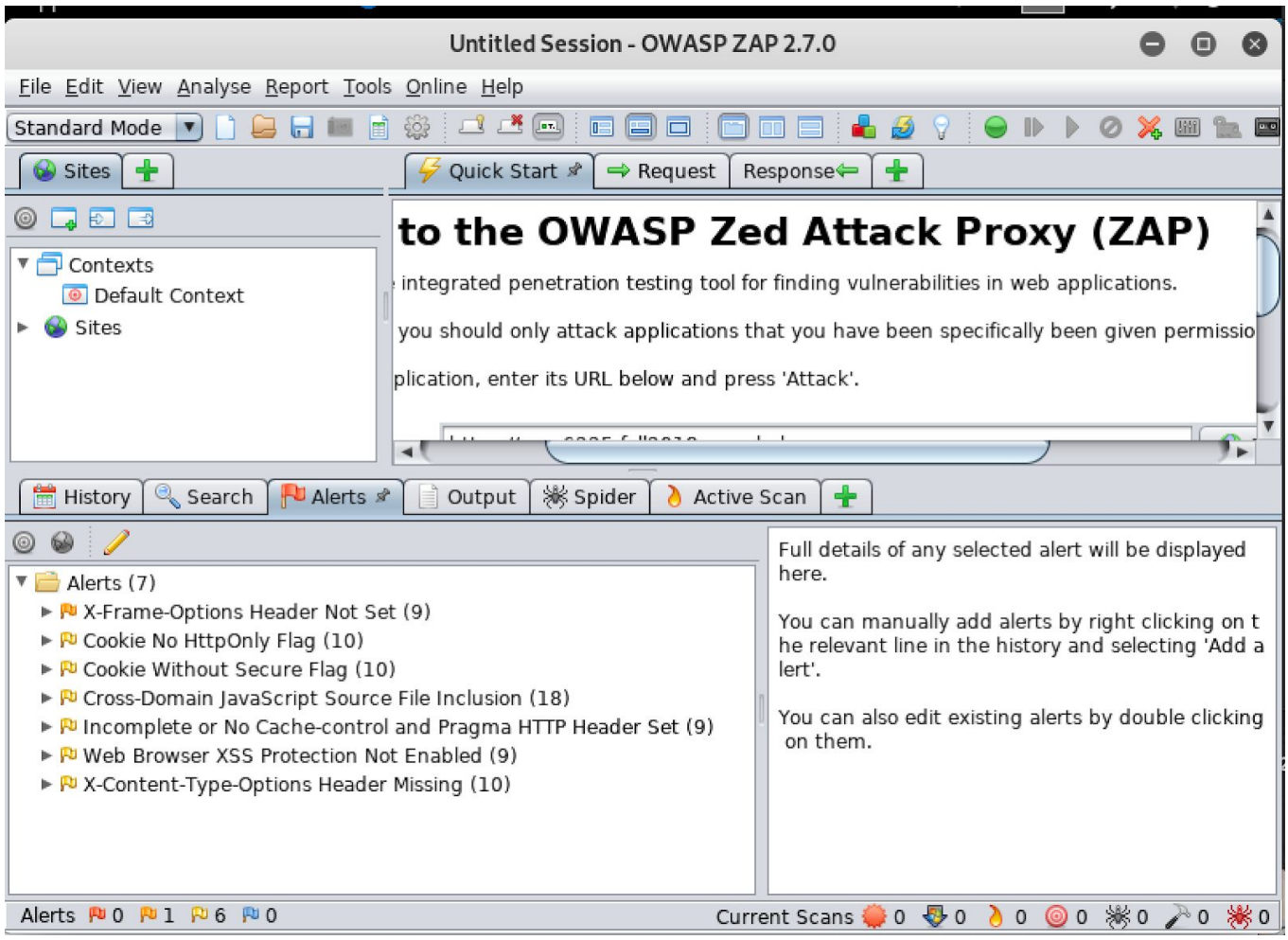
ASSIGNMENT 10

PENETRATION TESTING

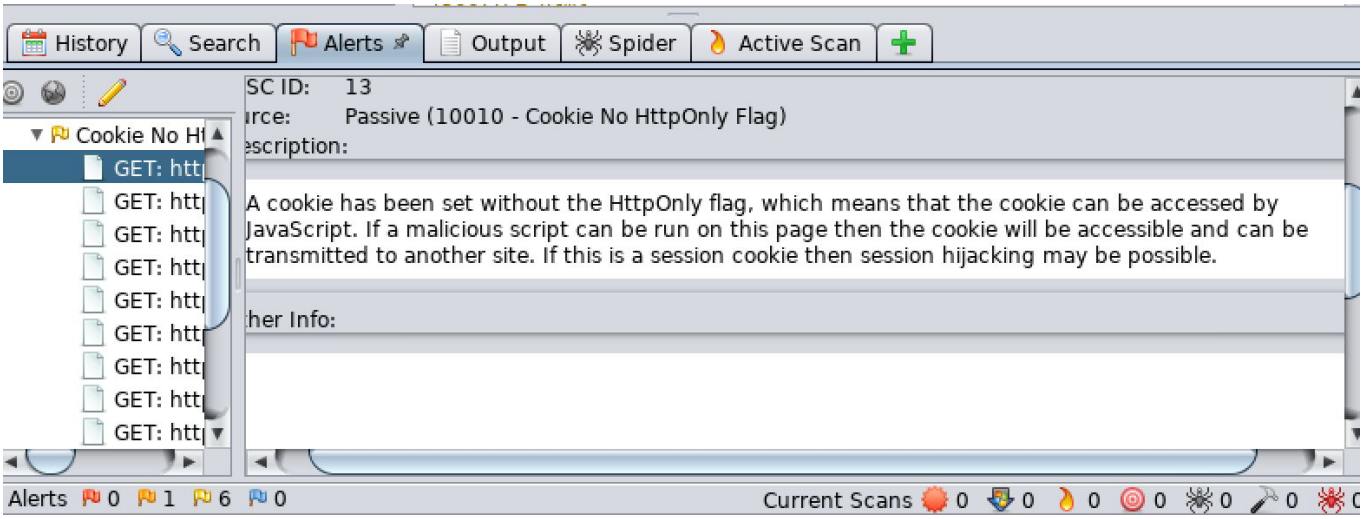
As the web application does not have a UI there aren't many vulnerabilities that can be exploited using the tools in the Kali Linux.

Some of the vulnerabilities that we detected using the tool OWASP-ZAP are listed below with the attacks that we could probably expect.

1. Session hijacking: As the cookies has been set without the HttpOnly flag. We can hijack the session with any of the tools like Ettercap, Hamster etc.



2. Sniffing and Spoofing: Another way to exploit our web app is through sniffing (interception of data by capturing the network traffic) using tools like Driftnet.



We also tried SQL injection both manually and by using SQLmap tool but since we are bcrypting the passwords by using SALT and because of some functionalities of Node.js itself there was no injectible API found across the domain.

```
[07:29:34] [INFO] target URL content is stable
[07:29:34] [INFO] testing if URI parameter '#1*' is dynamic
[07:29:34] [INFO] URI parameter '#1*' appears to be dynamic
[07:29:34] [WARNING] heuristic (basic) test shows that URI parameter '#1*' might not be injectable
[07:29:34] [INFO] testing for SQL injection on URI parameter '#1*'
[07:29:34] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[07:29:35] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[07:29:35] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause
 (FLOOR)'
[07:29:35] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[07:29:35] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)
'
[07:29:35] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[07:29:35] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace (FLOOR)'
[07:29:35] [INFO] testing 'MySQL inline queries'
[07:29:35] [INFO] testing 'PostgreSQL inline queries'
[07:29:35] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[07:29:35] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[07:29:35] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[07:29:35] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[07:29:35] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind'
[07:29:35] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[07:29:35] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[07:29:35] [INFO] testing 'Oracle AND time-based blind'
[07:29:35] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[07:29:36] [WARNING] URI parameter '#1*' does not seem to be injectable
[07:29:36] [CRITICAL] all tested parameters do not appear to be injectable. Try to increase values f
or '--level'/'--risk' options if you wish to perform more tests. If you suspect that there is some k
ind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '
--tamper=space2comment')
[07:29:36] [WARNING] HTTP error codes detected during run:
404 (Not Found) - 1 times

[*] ending @ 07:29:36 /2018-11-26/
```

```
[centos@ip-10-0-0-38 sqlmap-dev]$ python sqlmap.py -u "https://csye6225-fall2018-sureshak.me/api/time"
        ___
       __H__
 ___ ___[*]_____ ___ ___  {1.2.11.15#dev}
|_ -| . [.]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 07:29:21 /2018-11-26/

[07:29:21] [WARNING] you've provided target URL without any GET parameters (e.g. 'http://www.site.com/article.php?id=1') and without providing any POST parameters through option '--data'
do you want to try URI injections in the target URL itself? [Y/n/q] Y
[07:29:25] [INFO] testing connection to the target URL
[07:29:25] [INFO] checking if the target is protected by some kind of WAF/IPS
[07:29:25] [CRITICAL] heuristics detected that the target is protected by some kind of WAF/IPS
do you want sqlmap to try to detect backend WAF/IPS? [y/N] y
[07:29:34] [WARNING] dropping timeout to 10 seconds (i.e. '--timeout=10')
[07:29:34] [INFO] using WAF scripts to detect backend WAF/IPS protection
[07:29:34] [WARNING] WAF/IPS product hasn't been identified
[07:29:34] [INFO] testing if the target URL content is stable
[07:29:34] [INFO] target URL content is stable
[07:29:34] [INFO] testing if URI parameter '#1*' is dynamic
[07:29:34] [INFO] URI parameter '#1*' appears to be dynamic
[07:29:34] [WARNING] heuristic (basic) test shows that URI parameter '#1*' might not be injectable
[07:29:34] [INFO] testing for SQL injection on URI parameter '#1*'
[07:29:34] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[07:29:35] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[07:29:35] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause
```

ATTACK VECTORS

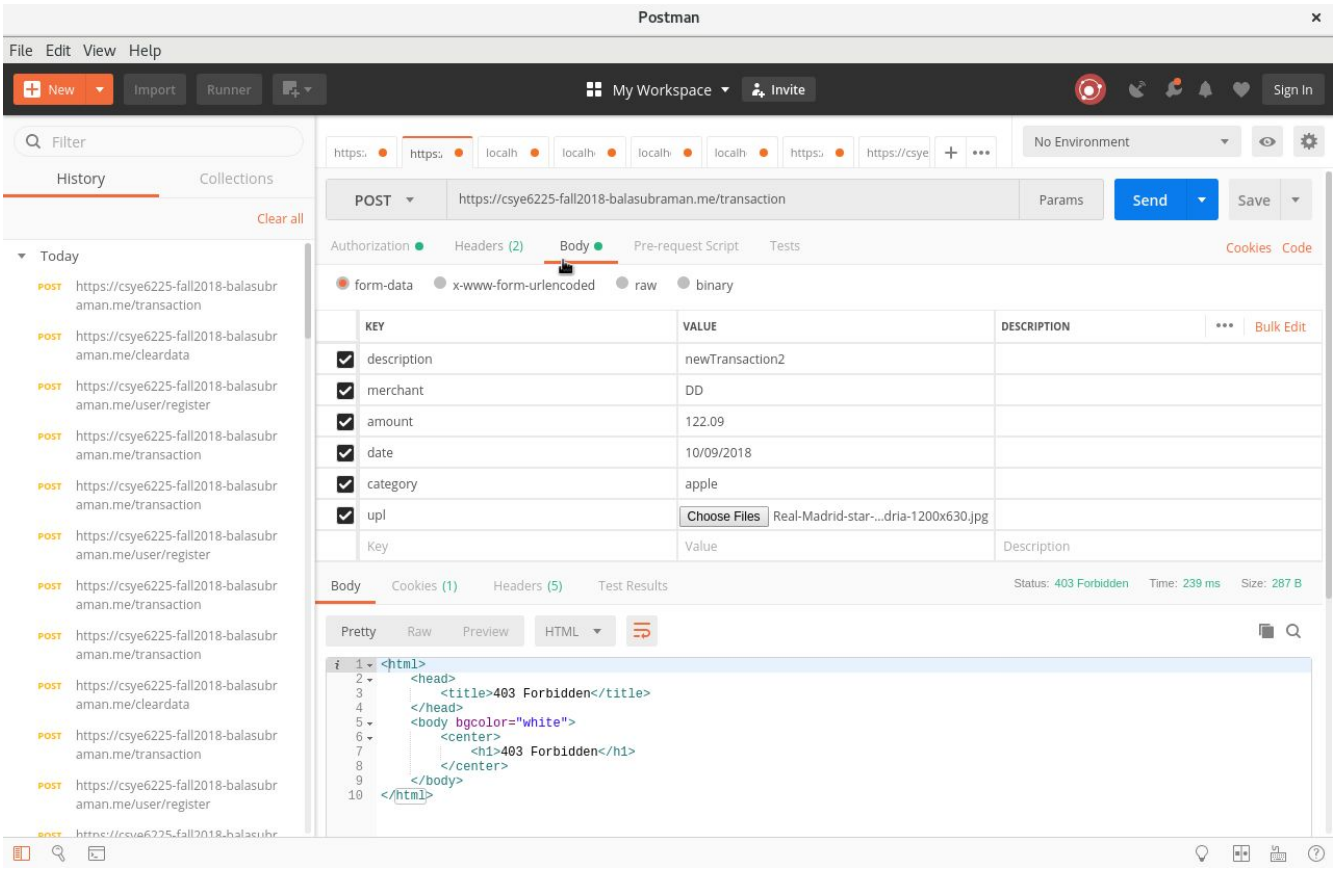
We started exploiting the web application with three of the below chosen attack vectors:

1. **Attack vector 1**: Unauthorized file upload to S3

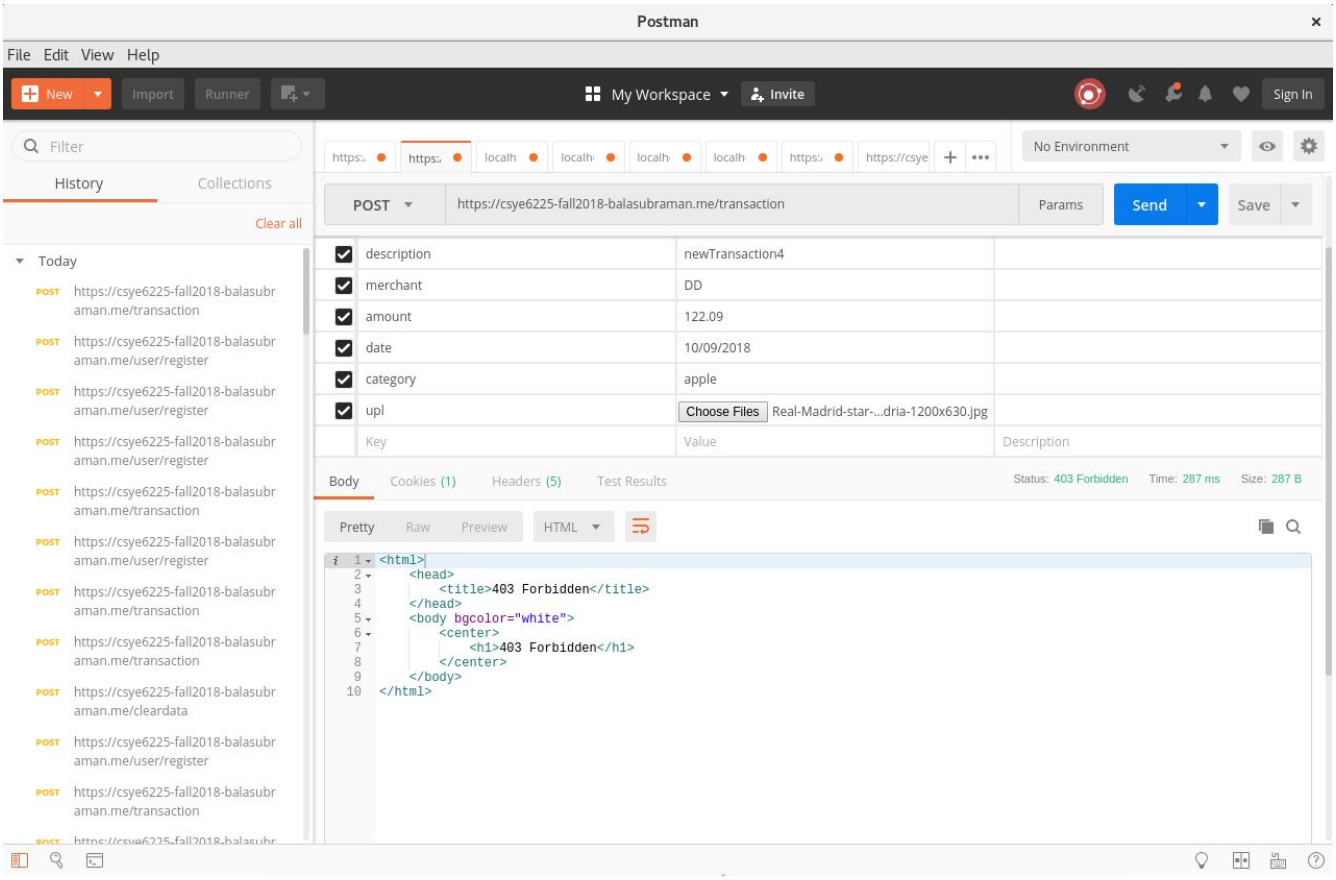
Unlimited/ Large file upload size (attachments) for each transaction is again a risk factor that might create an outage for other users of the web app. We restricted the upload of files up to the file size of 40 KB.

Result: prevents outage of the S3 storage resource for the users.

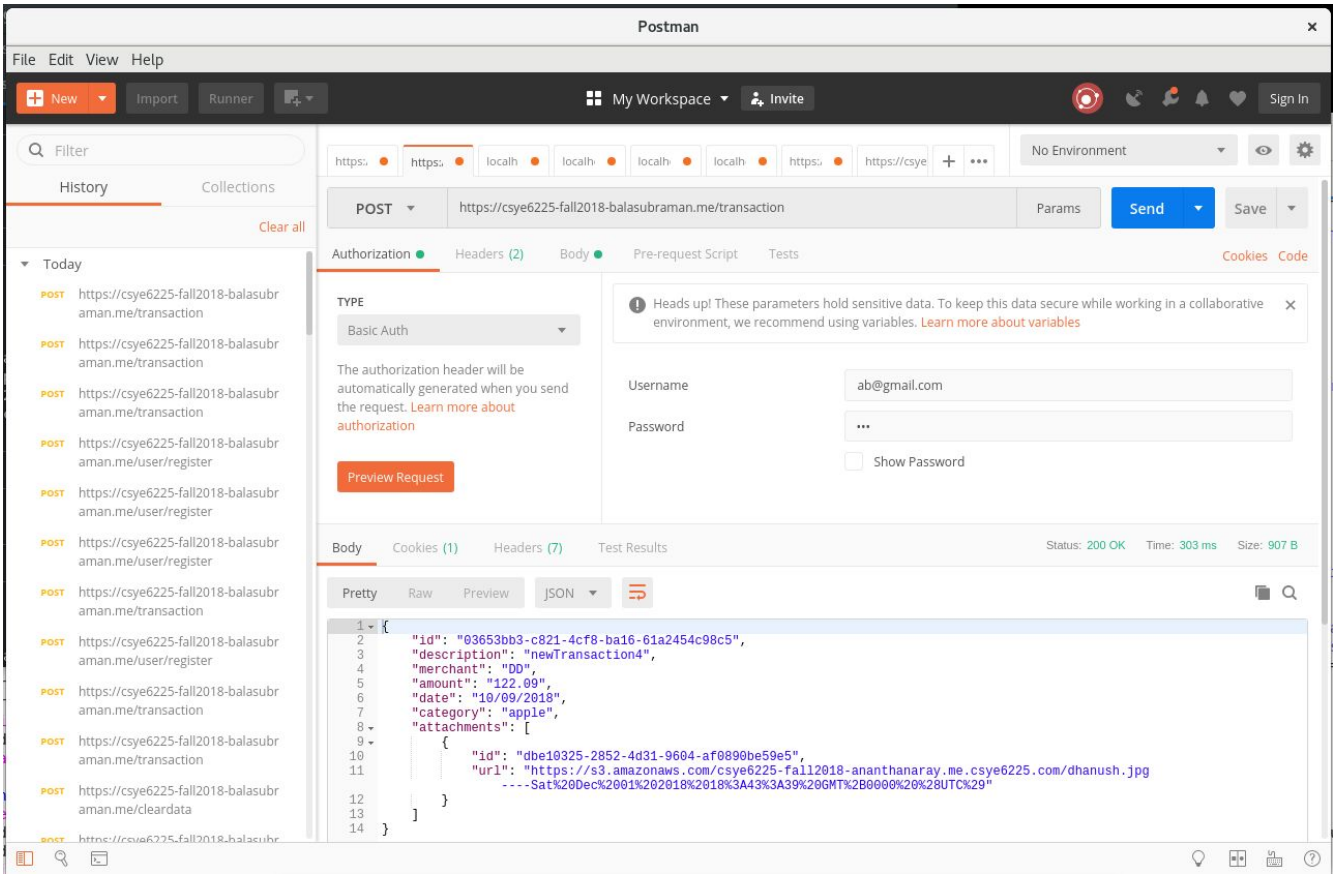
Before AWS WAF, Users can upload any size files. (Oversized file – sqlmap2.png)



After implementing the constraints on the size of file. The requests containing large file are rejected by the load balancer with 403 forbidden error. (Oversized file – sqlmap2.png)



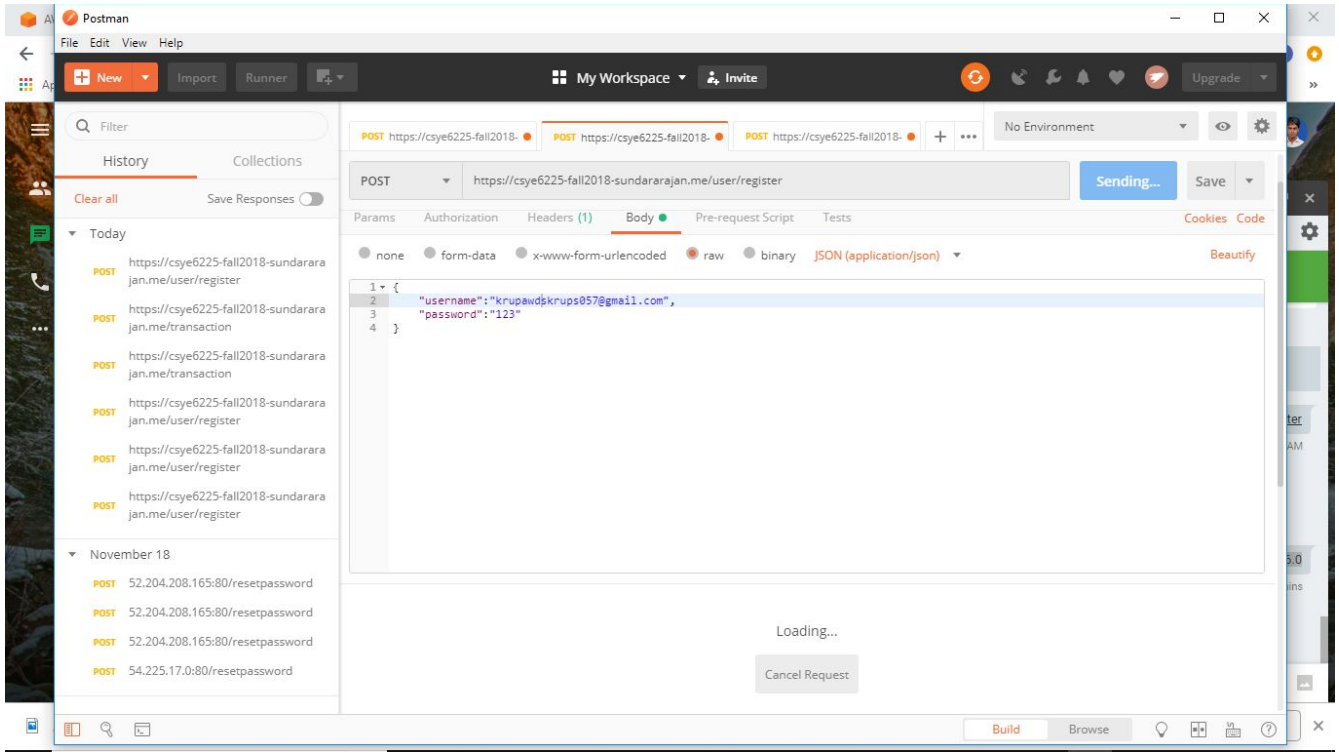
File size less that 50kB (aws.png)

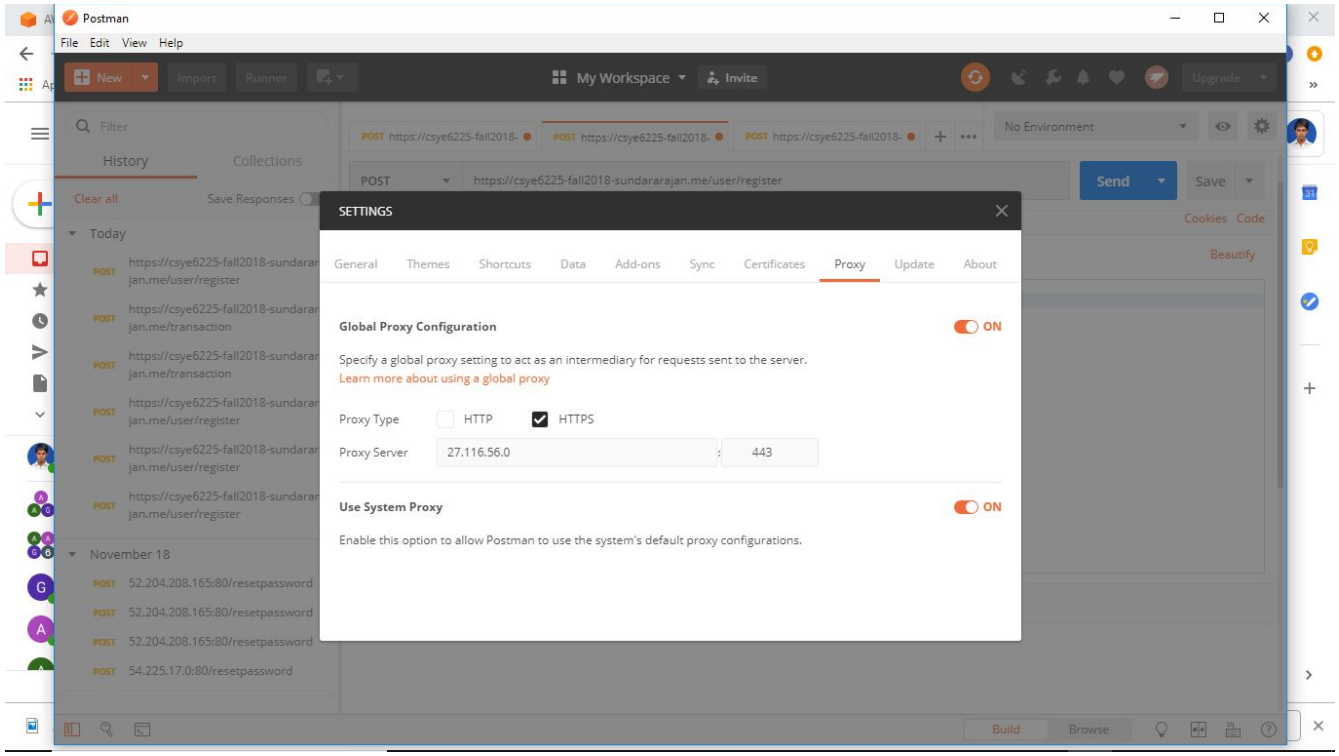


2. **Attack vector 2**: Blacklisting IP address of nefarious users.

We block the web requests that are coming from nefarious users. This will help protect the web app from the bad user activities.

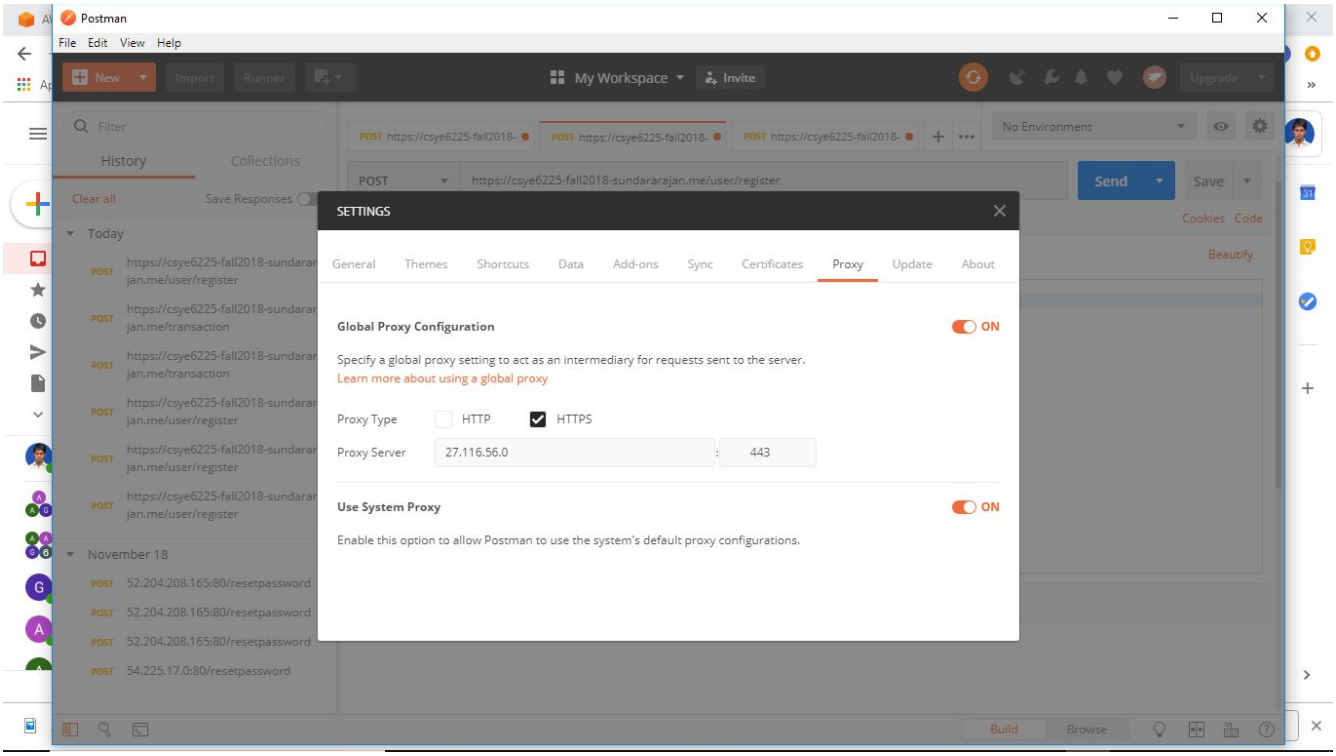
Result: Web app is safe from the access of bad users.


Postman
File Edit View Help
New Import Runner
My Workspace Invite
Upgrade
Filter
History Collections
Clear all Save Responses
Today
POST https://csye6225-fall2018-sundararajan.me/user/register
POST https://csye6225-fall2018-sundararajan.me/transaction
POST https://csye6225-fall2018-sundararajan.me/transaction
POST https://csye6225-fall2018-sundararajan.me/user/register
POST https://csye6225-fall2018-sundararajan.me/user/register
POST https://csye6225-fall2018-sundararajan.me/user/register
November 18
POST 52.204.208.165:80/resetpassword
POST 52.204.208.165:80/resetpassword
POST 52.204.208.165:80/resetpassword
POST 54.225.17.0:80/resetpassword
No Environment
POST https://csye6225-fall2018-sundararajan.me/user/register
Sending...
Save
Params Authorization Headers (1) Body Pre-request Script Tests
Cookies Code
none form-data x-www-form-urlencoded raw binary JSON (application/json)
Beautify
{
"username":"krupawd$krups057@gmail.com",
"password":"123"
}
Loading...
Cancel Request
Build Browse


Postman
File Edit View Help
New Import Runner
My Workspace Invite
Upgrade
Filter
History Collections
Clear all Save Responses
Today
November 18
POST 52.204.208.165:80/resetpassword
POST 52.204.208.165:80/resetpassword
POST 52.204.208.165:80/resetpassword
POST 54.225.17.0:80/resetpassword
No Environment
POST https://csye6225-fall2018-sundararajan.me/user/register
Send
Save
Cookies Code
Beautify
SETTINGS
General Themes Shortcuts Data Add-ons Sync Certificates Proxy Update About
Global Proxy Configuration
ON
Specify a global proxy setting to act as an intermediary for requests sent to the server.
Learn more about using a global proxy
Proxy Type HTTP HTTPS
Proxy Server 27.116.56.0 : 443
Use System Proxy
ON
Enable this option to allow Postman to use the system's default proxy configurations.
Build Browse

With the help of AWS WAF, we can block a specific IP address from accessing the web application. No response is sent from the web servers for such requests.

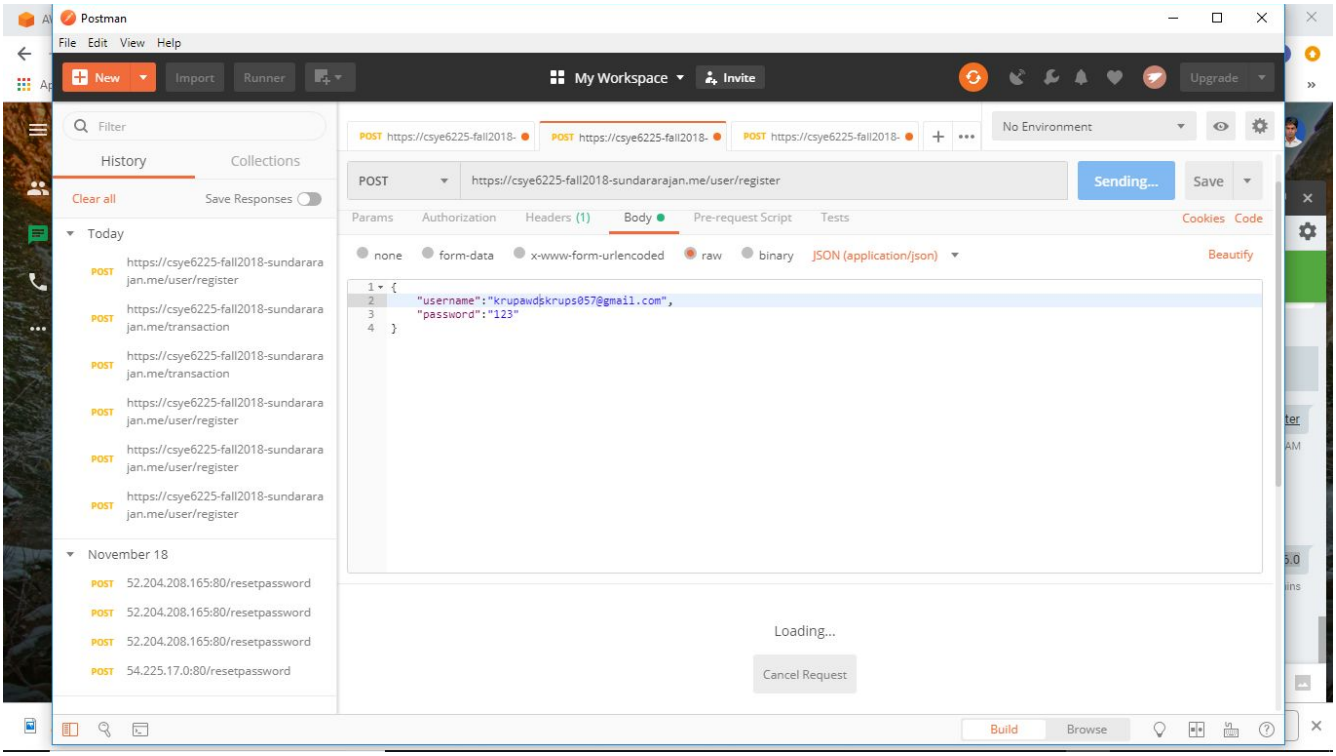
3. **Attack vector 3**: Restricting access to a whole country or a domain.



When an event occurs where the access to the website should be either accessible or in accessible from IP within a country. Then we can go for this.

Result: Prevents the access to the web app for a set/range of IP addresses

This could be possible with WAF by filtering requests that are coming from a specific country with the help of GeoLocation.



Apart from these enabling a session time is also a good practice to secure the web application.