BLIND XSS

TABLE OF CONTENTS

1. What is Blind XSS ?

A new variant of Cross Site Scripting Attack came into picture recently based on the way how it can be exploited and the state of payload triggering.

In general, Cross Site Scripting – Injecting JavaScript into reflection points on Web/Mobile Applications to steal user account information.

XSS is classified as below based on the way of Exploitation.

Reflected XSS – As name suggests one time execution whenever user clicks/open a link.

Stored XSS – Injected payload getting stored somewhere in application and will execute everytime when admin/victim visits that page.

DOM XSS/ Type Zero XSS – Different from above two scenarios where no server contact is required. Injected payload will execute immediately without contacting server due to Browser's Document Object Model Environment.

Self XSS – This comes out based on the way how it is executed. Similar to above kinds but the impact is only on the user who executes it. Hence named as Self XSS.

Universal XSS – Interesting category which can be triggered by exploiting browser based flaws to hijack user data.

Mutation XSS – Vulnerability occurs based on browser mutation when user input passed to browser DOM by using **innerHTML** and mutated into valid XSS attack.

All kinds of XSS which are classified based on way that we exploit and type of payload execution in context of application are explained above.

Blind XSS is just similar kind to Reflected/Stored XSS where payload is just stored or reflected in other internal/intranet applications instead of same application that we are testing on.

2. Detection

Below are the challenges that we face during detection and exploitation of Blind XSS.

- Where my payload is being stored
- When it will get execute
- How we can retrieve data
- What feature of application is vulnerable

To overcome these challenges we simply use some sort of framework (XSSHunter, Sleepy Puppy etc) which will send us an email or just a text message whenever our payload triggered.

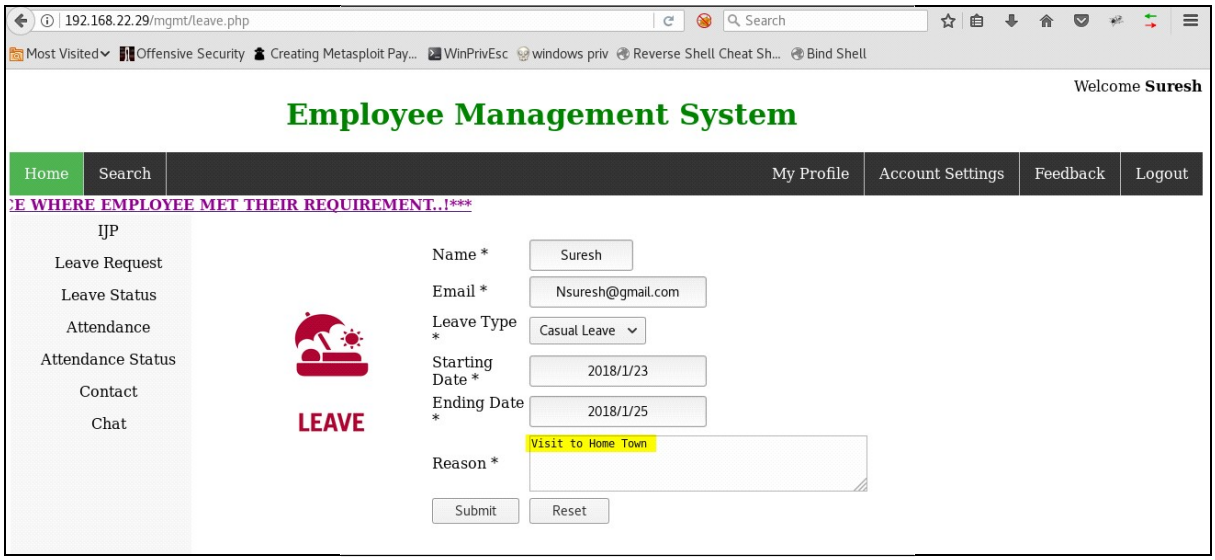
Below are the possibilities that normally having Blind XSS issues.

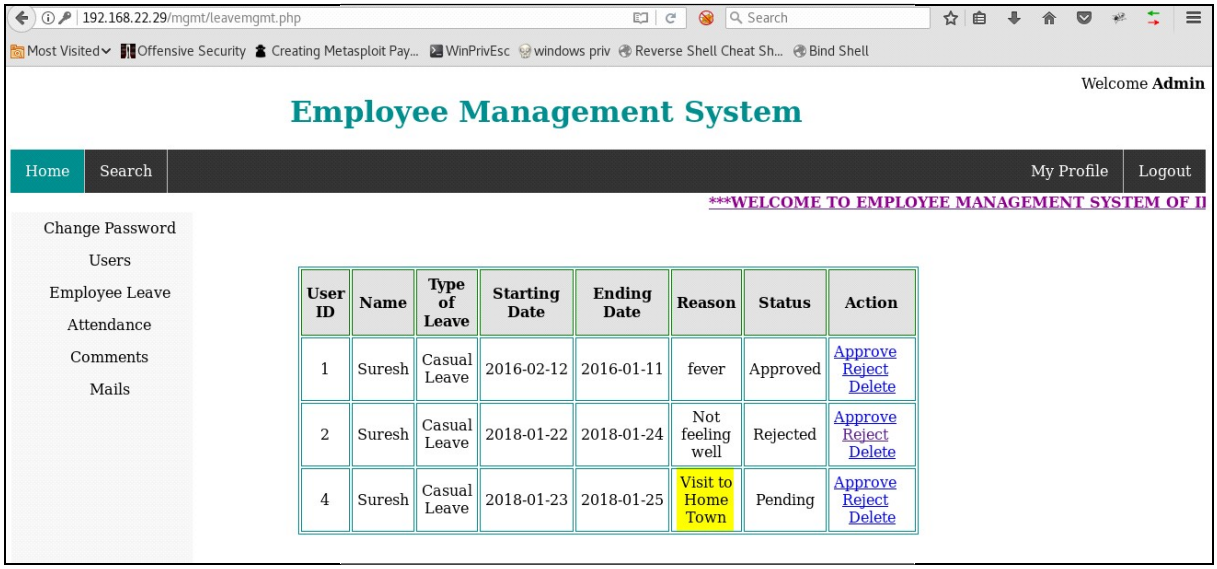
- Contact/Feedback pages
- Log viewers
- Exception handlers
- Chat applications / Forums
- Customer ticket applications
- Web Application Firewalls
- Any application that requires user moderation

3. Exploitation

We have demo application called Employee Management System where employees can update their leave status. Admin will approve the status based on employee requests.

From the above scenario we have identified the possible insertion points for Cross Site Scripting vulnerability scope where the text entered in **Reason** field is visible at admin (other intranet application) end.

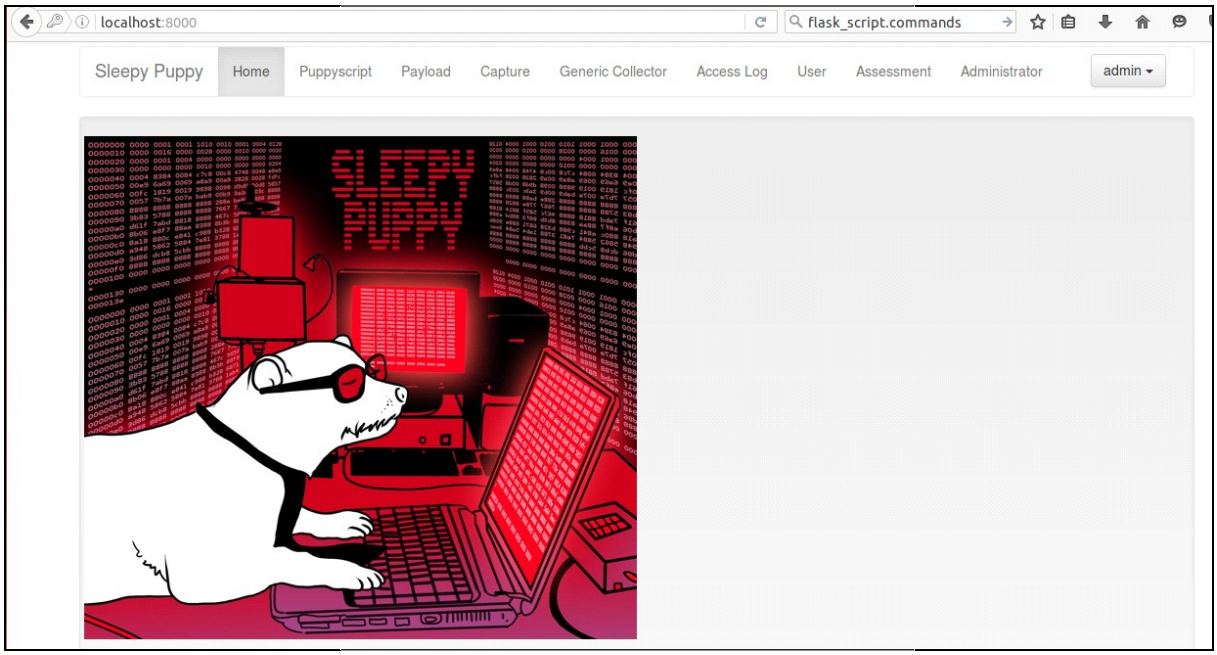


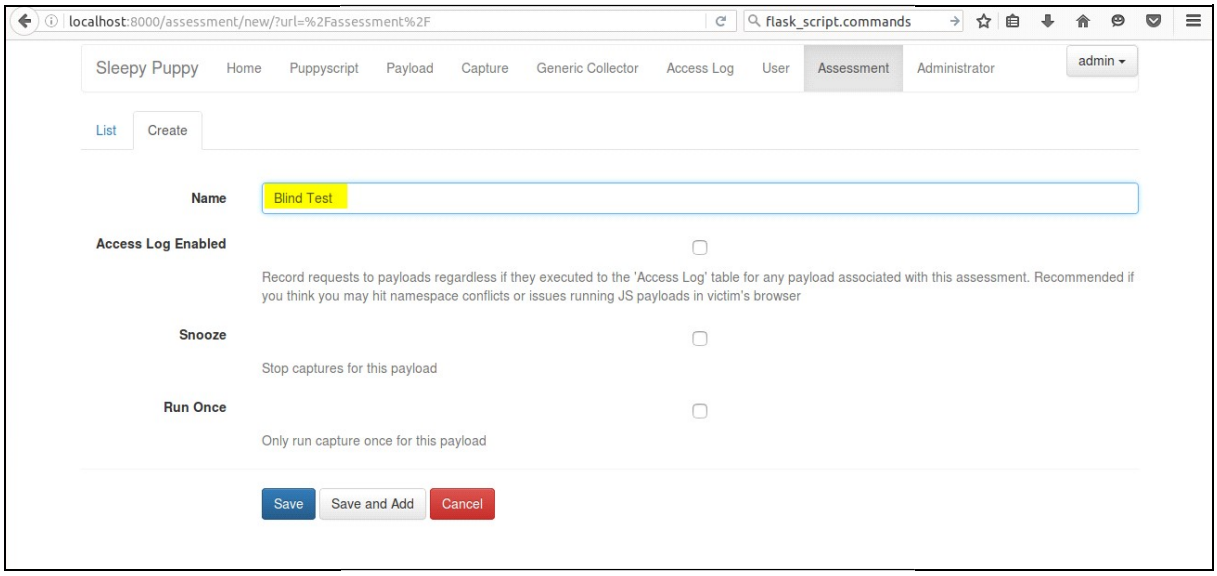


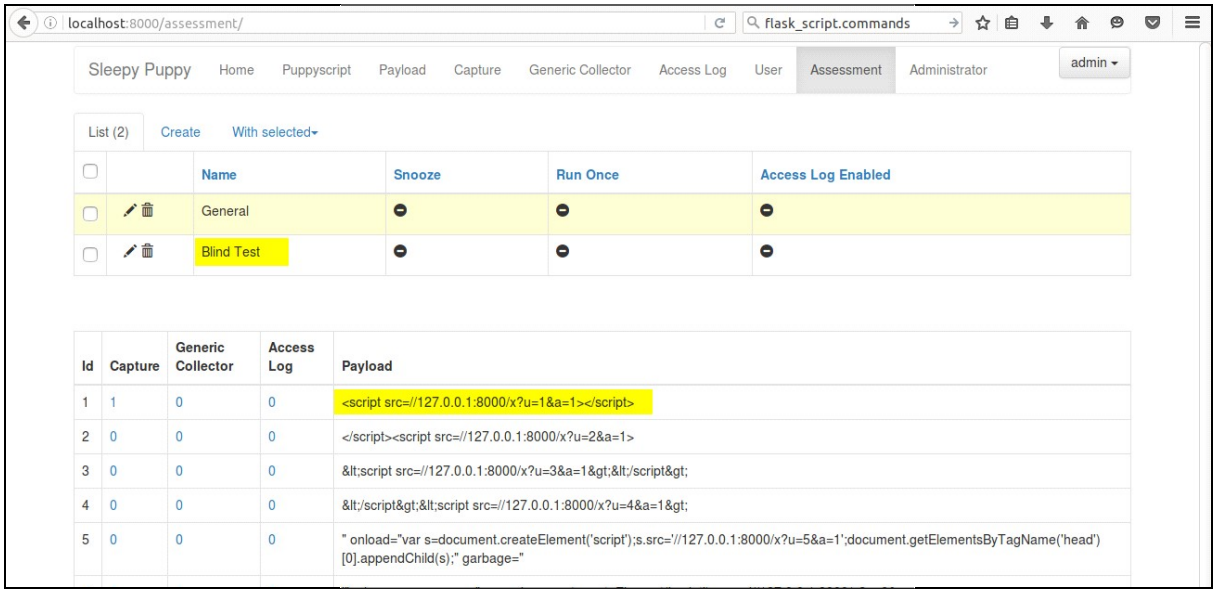
As we don't have access to admin portal and during assessment to confirm the existence of the vulnerability we will use **Sleepy Puppy** framework to automate our task to record the payload triggering and send us an email whenever payload is triggered.

Configuration of Sleepy Puppy - https://github.com/netflix/sleepy-puppy/wiki/setup

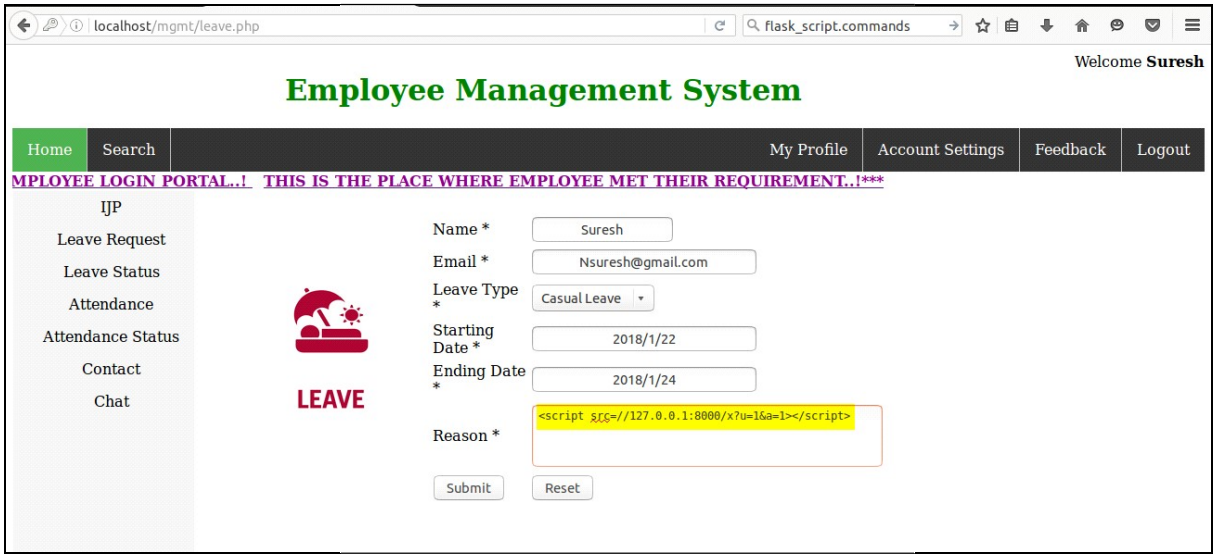
After setup we can access framework via port 8000. Create an assessment to set the alerts.

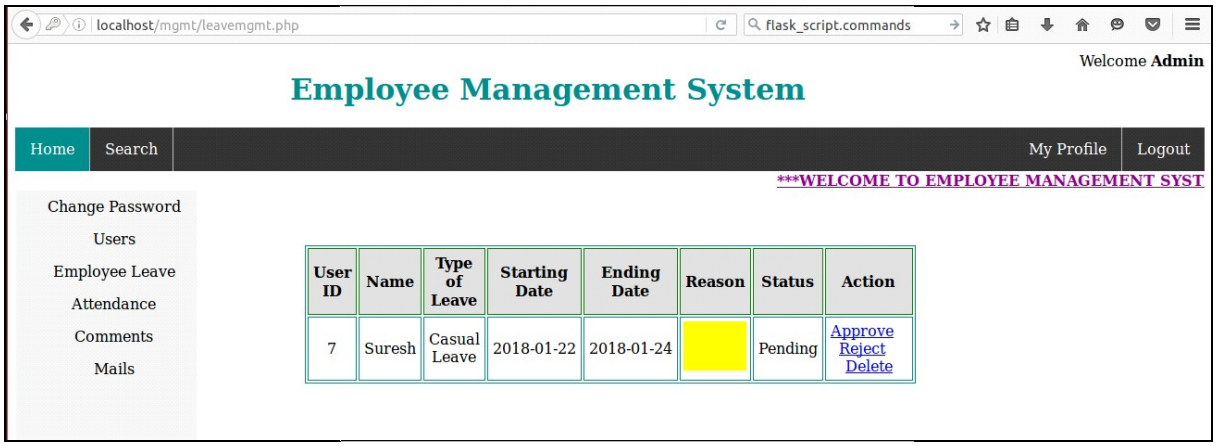




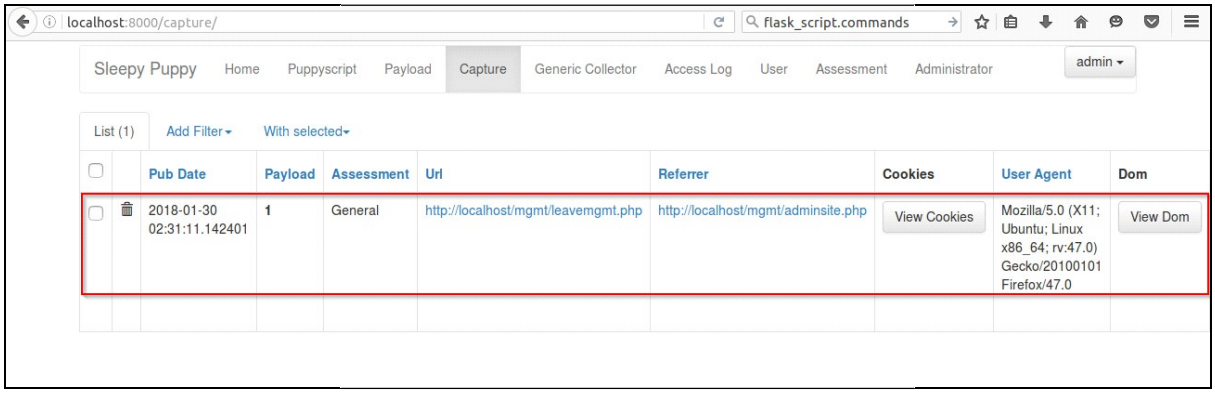


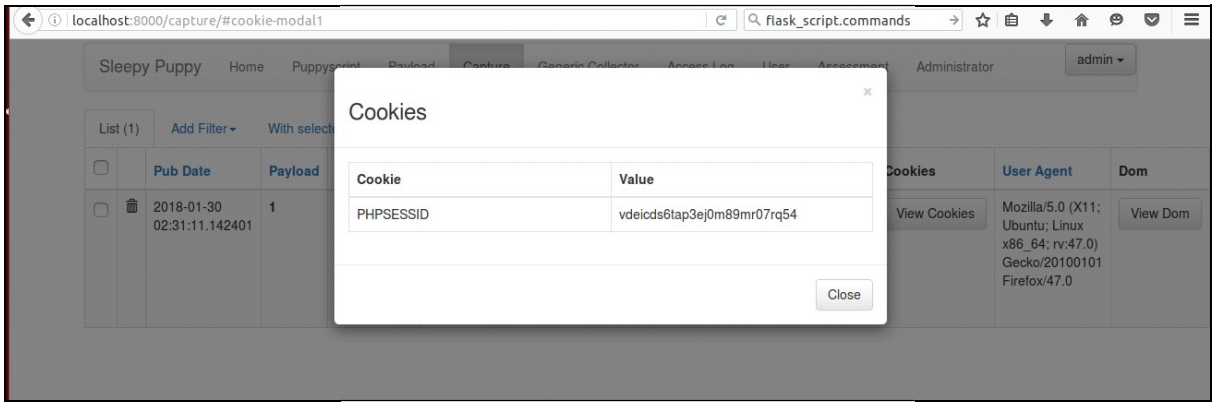
After creating assessment we can use any payload of our choice. Then navigate to any insertion point that comes under our Cross Site Scripting scope then insert the payload.

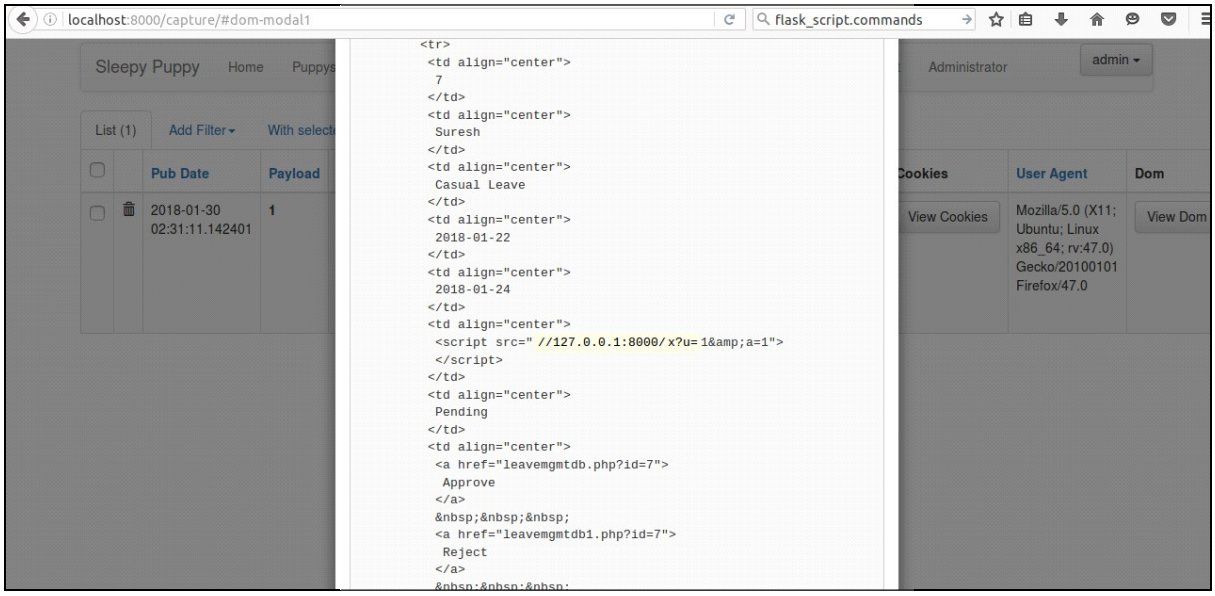




Once admin visits the leave approval page then our payload silently executes and we can see the triggered record in our Sleepy Puppy framework.



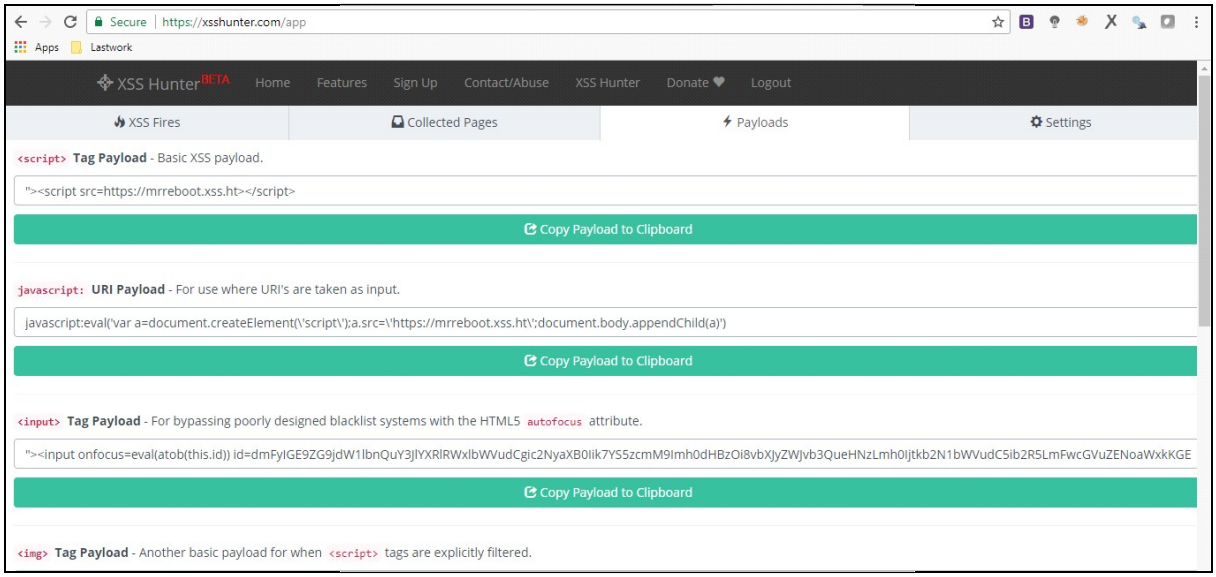




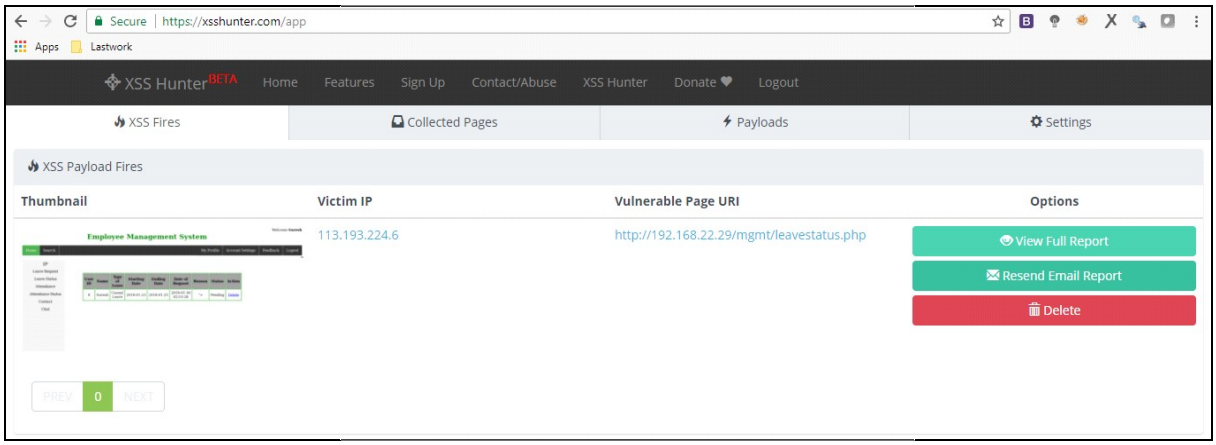
We can see the screenshot, session of admin and DOM content with the executed payload in source. But to receive the email on payload execution we need to host this in production with genuine SSL certificates.

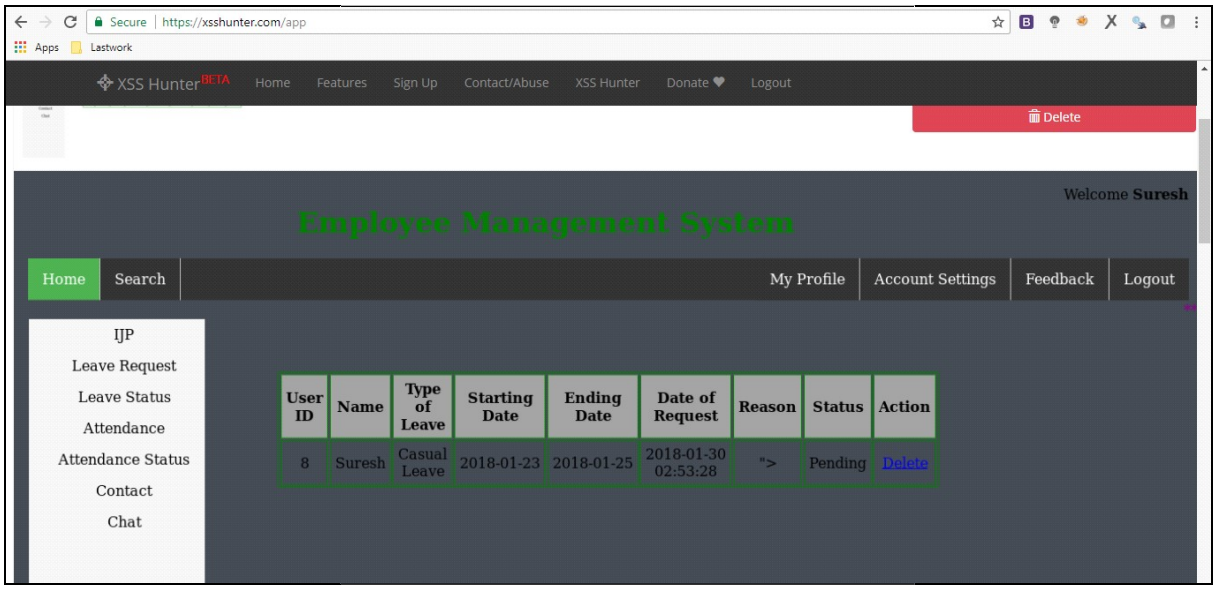
To come over these dependencies we can use XSSHunter Service (https://xsshunter.com)

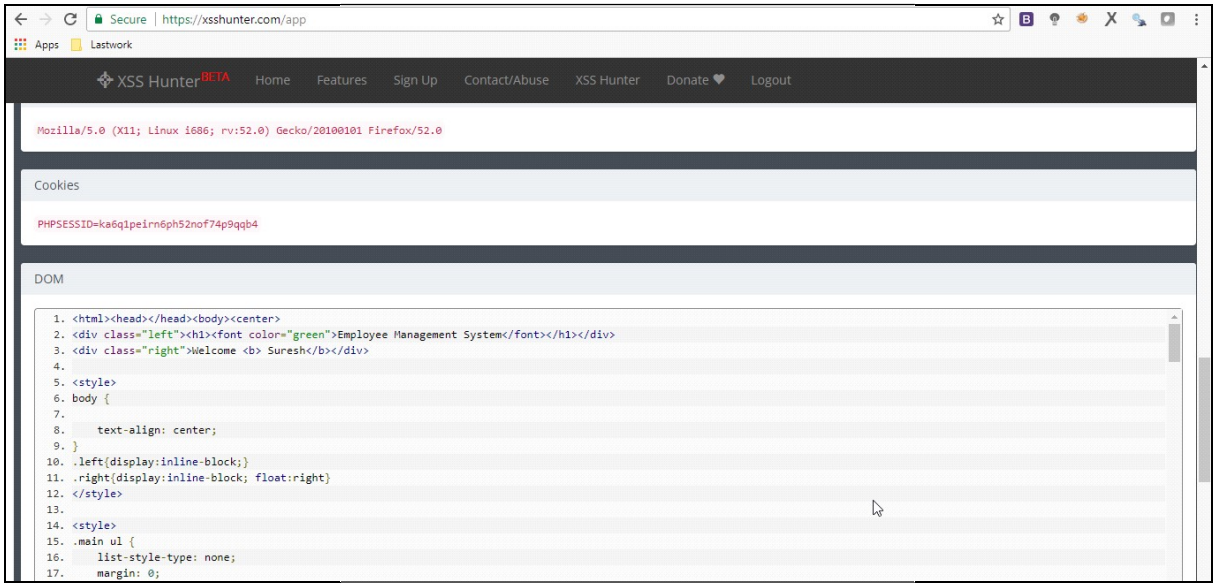
Simply register and under payloads section we have certain payloads to use.



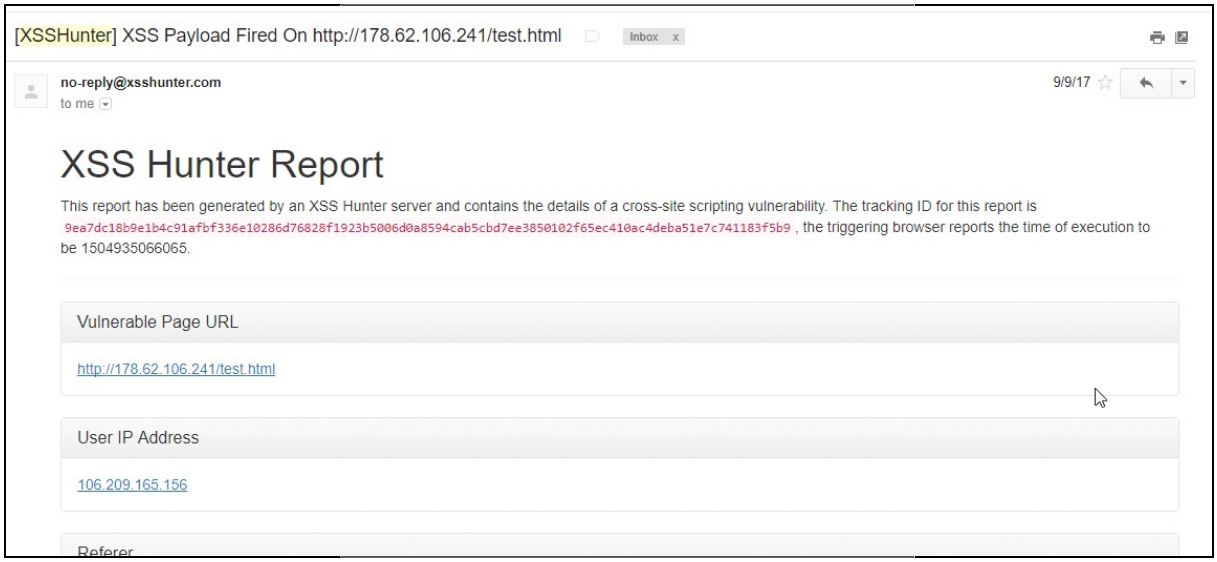
After payload triggered we can see the details as below.







We can see the mail alert from XSSHunter with full details.



4. Mitigation

Input sanitisation and HTML Entity Encoding usage will completely fix the issue.

5. References

1. https://xsshunter.com/app
2. https://github.com/netflix/sleepy-puppy/wiki/setup
3. https://brutelogic.com.br/blog/blind-xss-code/