Test Security Report

Code Revision: 1.0.0.0
Company: Acme Inc.
Report: TEST201018
Author: [Name]
Date: [Date]

VWA Security Report

VWA Security Report

TEST20101801 – SQLi - Critical

Vulnerability Exploited: SQLi

Severity: Critical

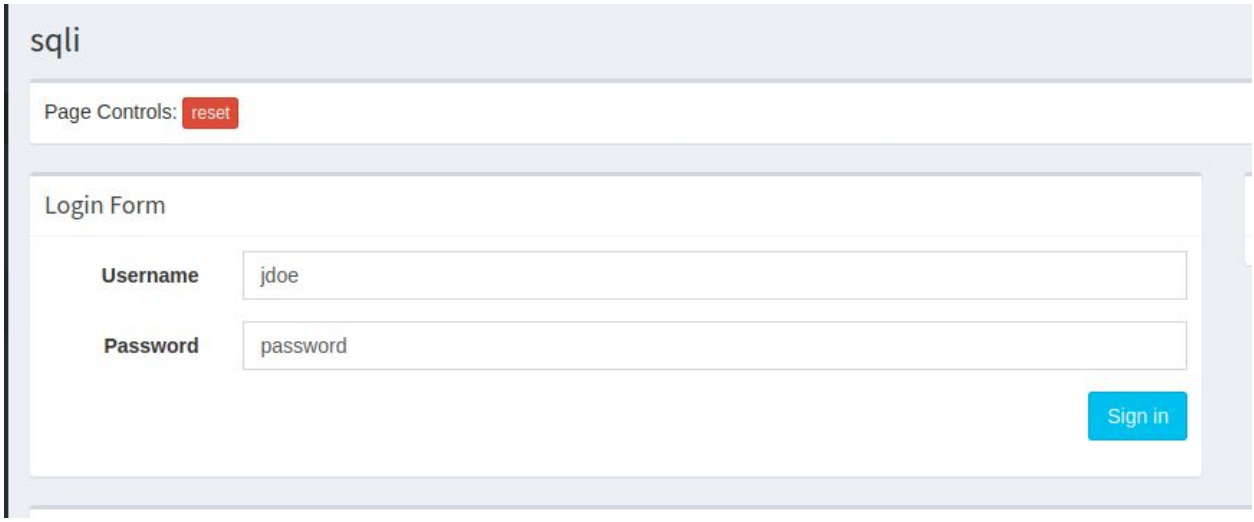
System: VWA Web Application

Vulnerability Explanation:

In the login form, we are not sanitizing the password field which is allowing SQLi to execute.

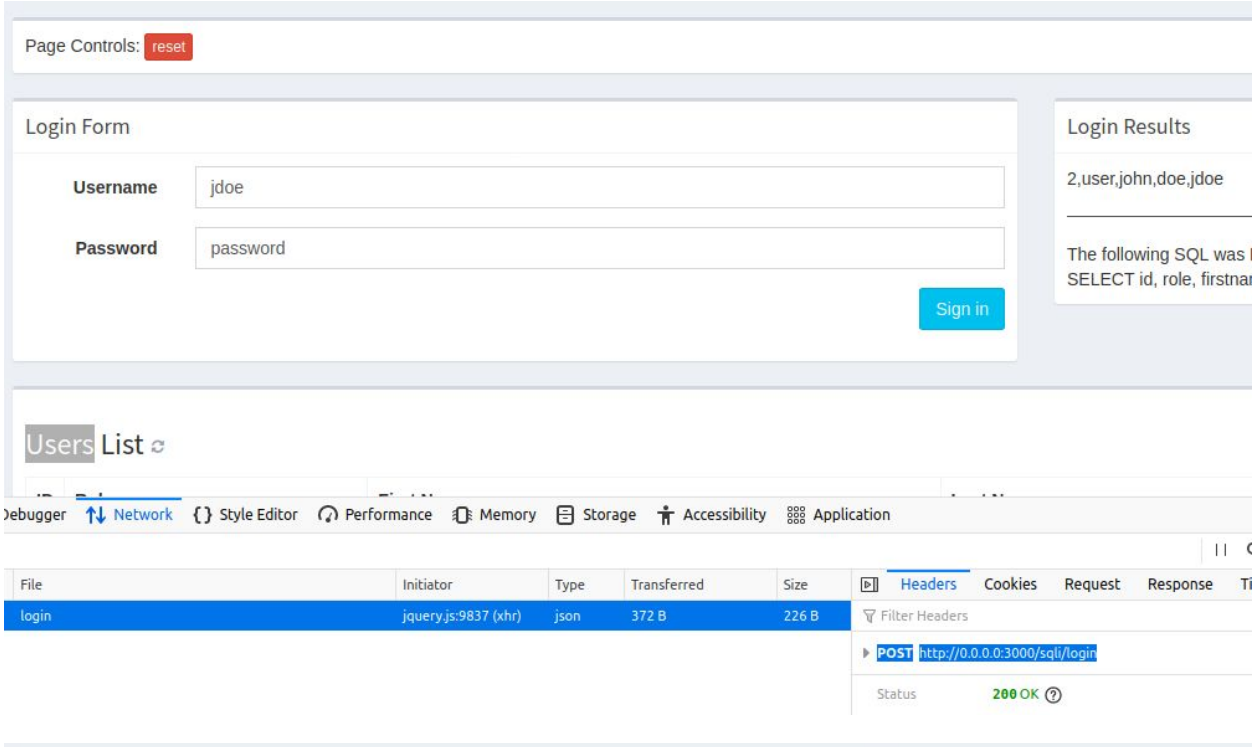
Vulnerability Walk-thru:

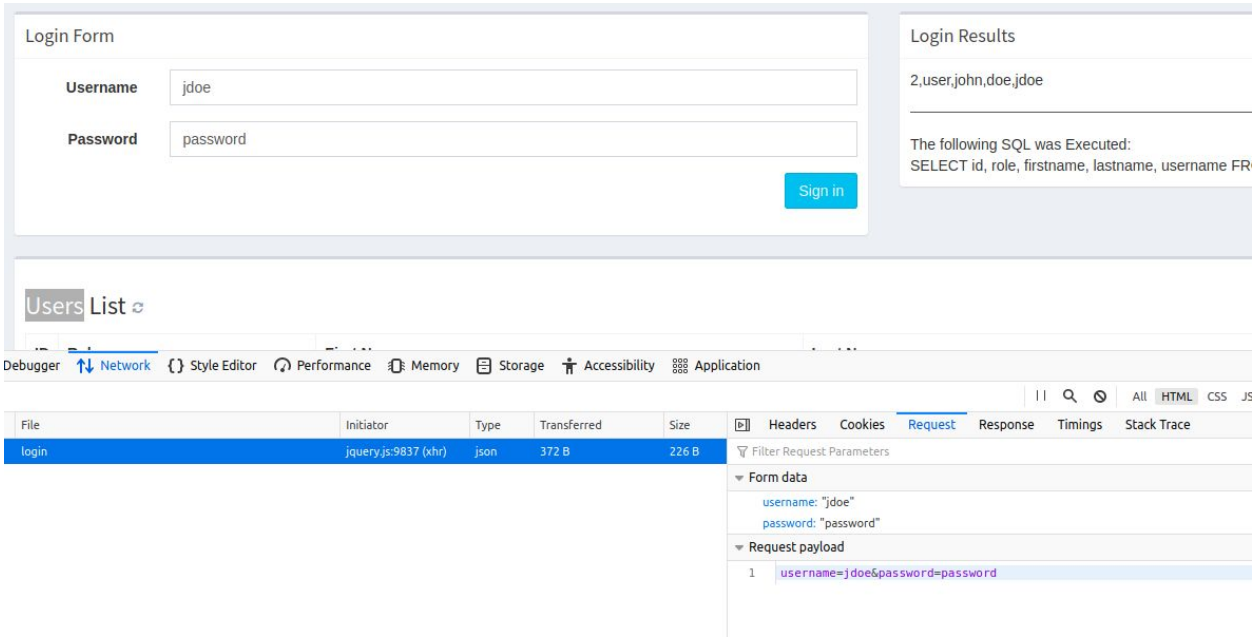
1. Got to the SQLi section of the application
2. On the page I noticed a login form



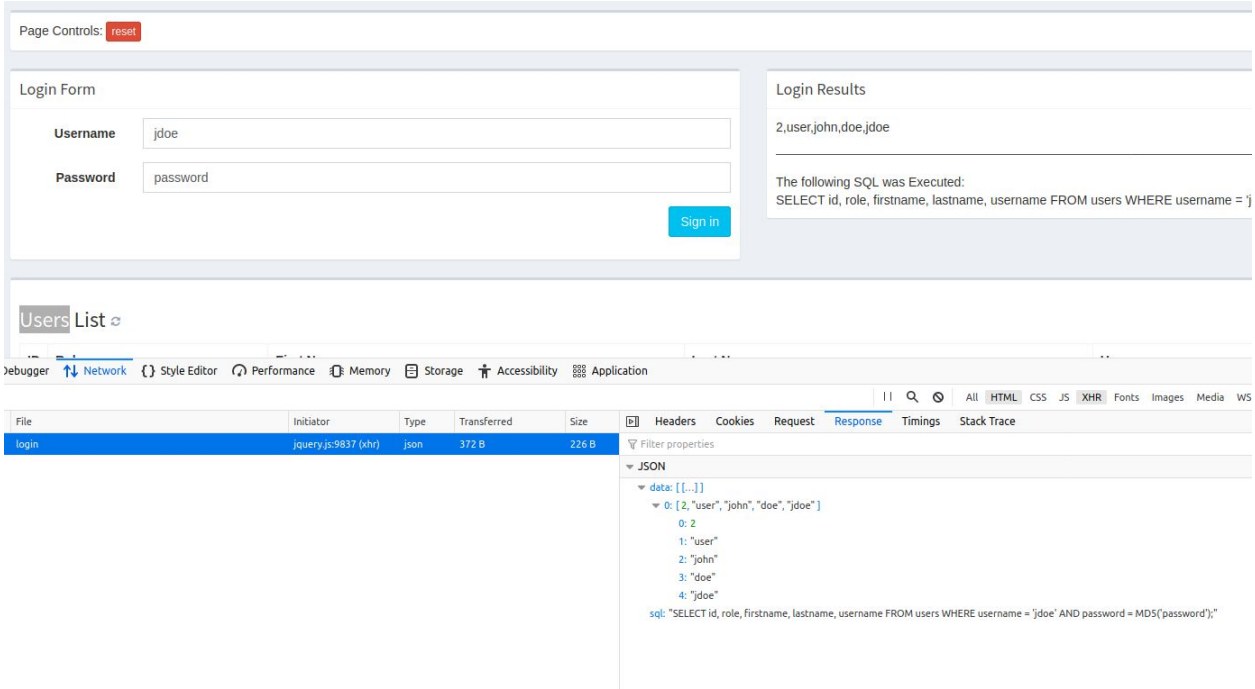
3. After looking at the network traffic flow, I can see that the login page is posting data back to the following http://0.0.0.0:3000/sqli/login.

VWA Security Report



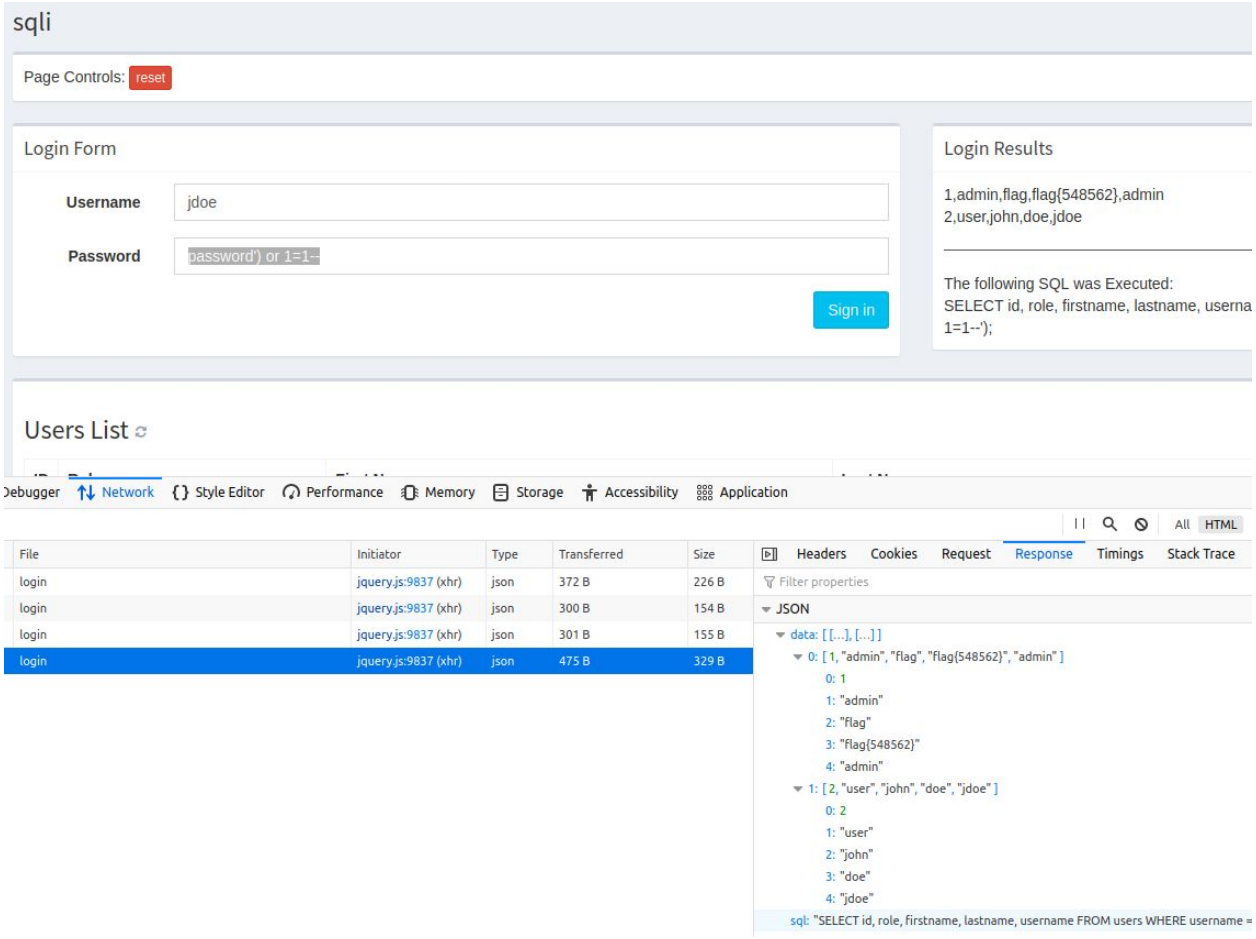


VWA Security Report



4. Next I used a simple SQLi “password' or 1=1--” to see if the login form is exploitable. This injection didn’t yield any results.
5. Next I used another SQLi “password') or 1=1--” to see if they are hashing or doing something else with that field.

VWA Security Report



6. At this point I was able to inject the SQL and return all user info.

Recommendations:

Sanitize the both the username and password fields.

https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

VWA Security Report

TEST20101802 – SQLi - Critical

Vulnerability Exploited: SQLi

Severity: Critical

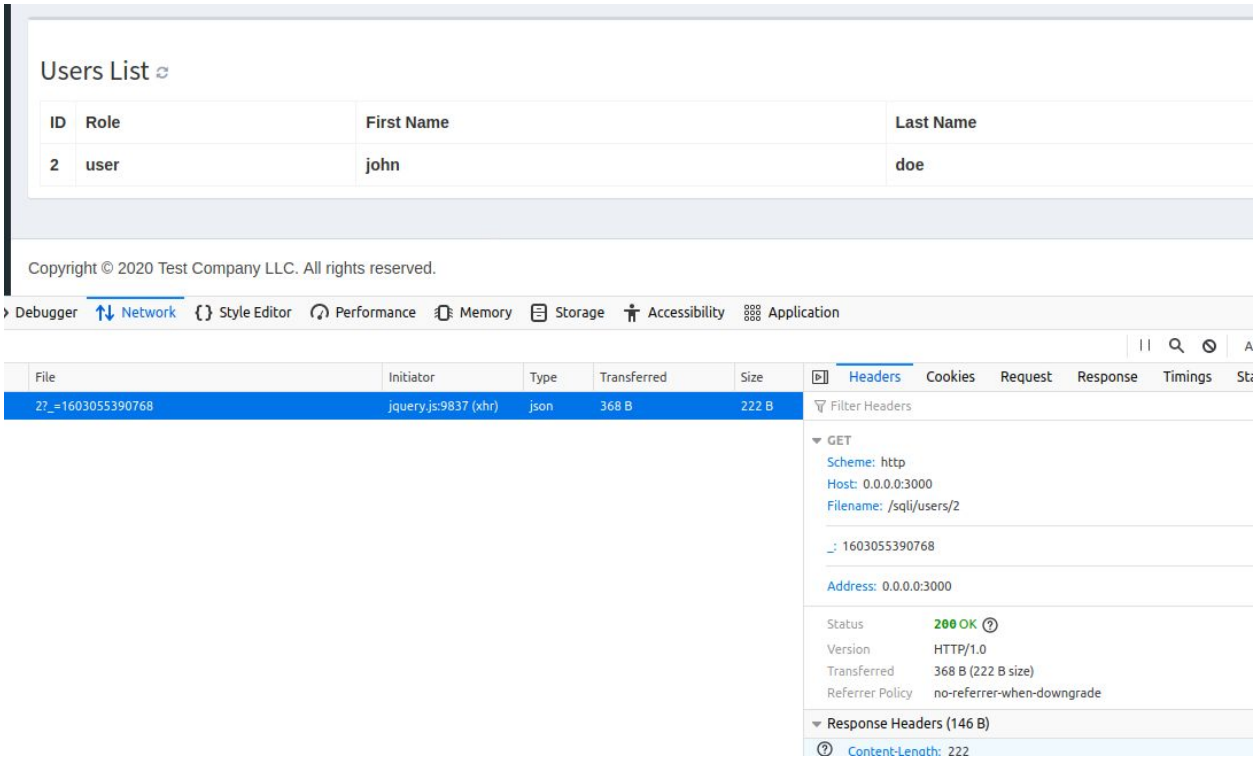
System: VWA Web Application

Vulnerability Explanation:

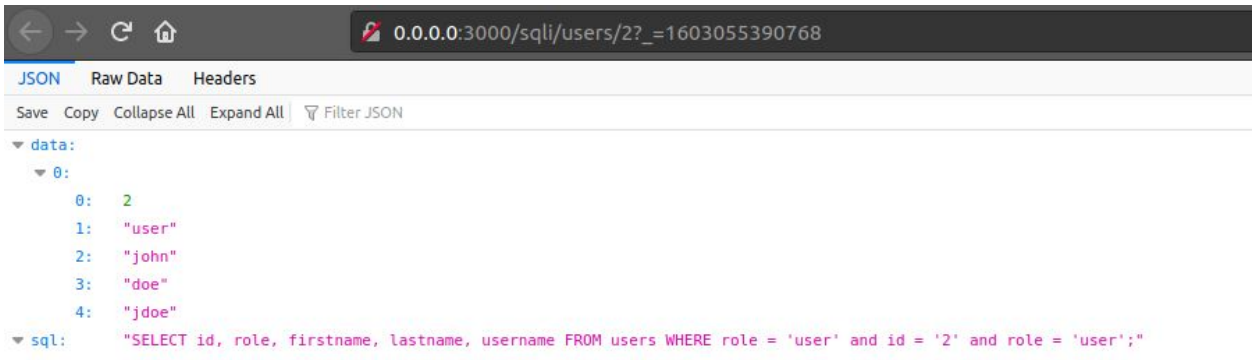
It seems that the user async endpoint to view users is vulnerable by manipulating the id field.

Vulnerability Walk-thru:

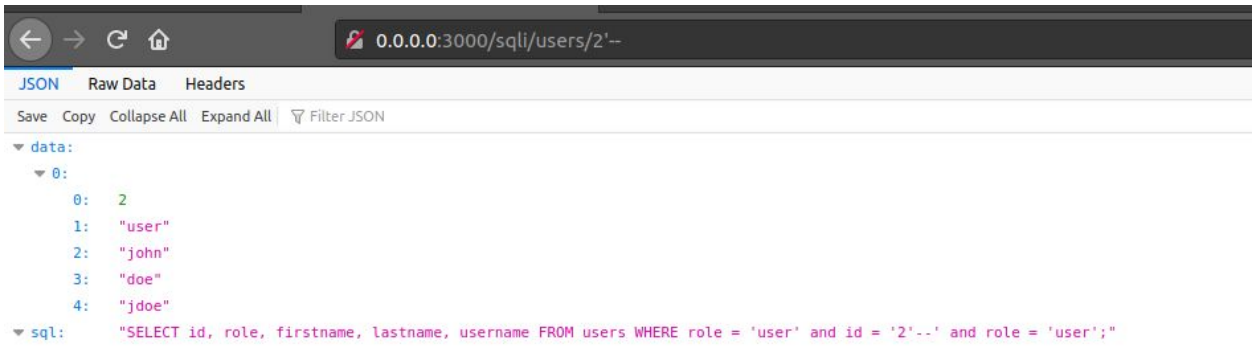
1. Got to the SQLi section of the application
2. Next I looked at the network flows and noticed that the userlist is using an async function to get the data it needed.



VWA Security Report



3. Then I did some basic tests to see if the async endpoint was vulnerable to SQLi.



4. After proving that this endpoint was injectable, I then crafted a SQLi that will expose all data in the table.

Recommendations:

Sanitize the id field that is used to only allow numbers.

VWA Security Report

https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

VWA Security Report

TEST20101803 – XSS - Critical

Vulnerability Exploited: XSS

Severity: Critical

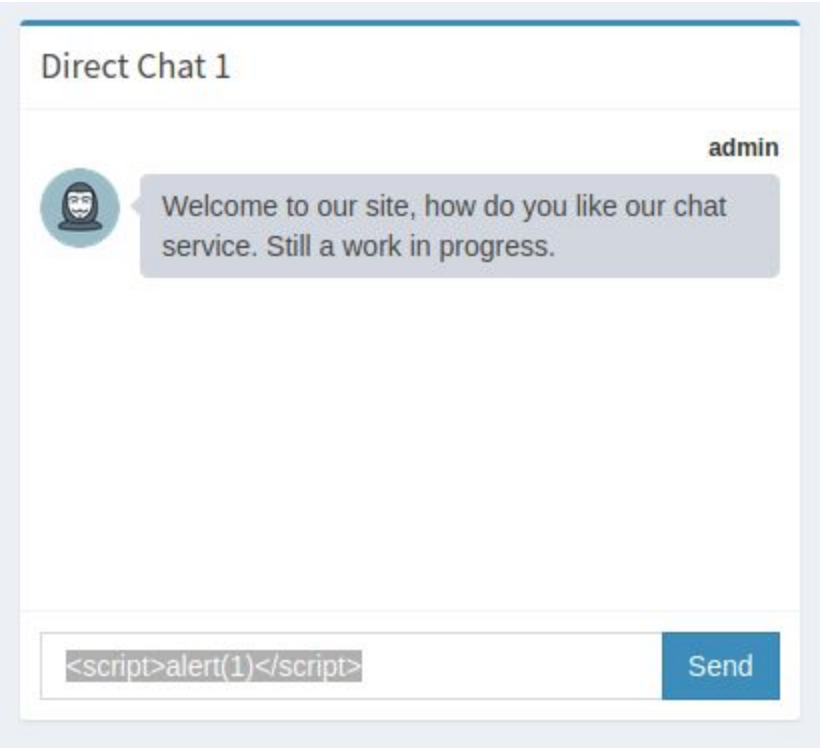
System: VWA Web Application

Vulnerability Explanation:

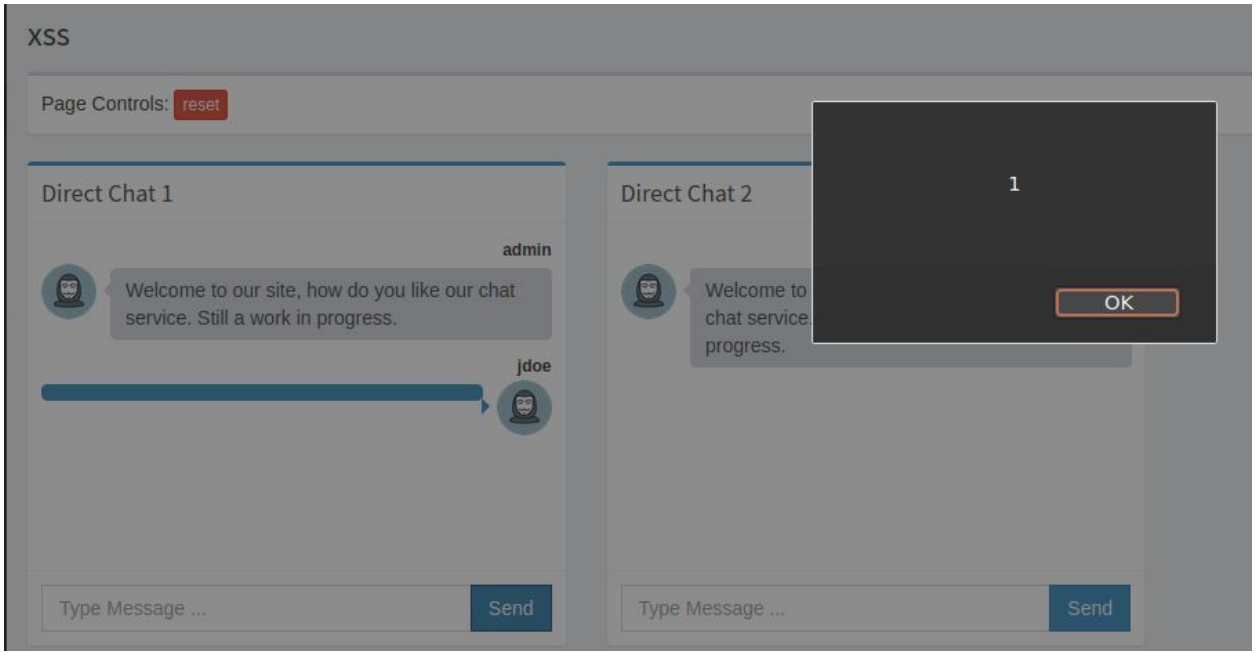
In the XSS Area we have an internal Chat service, this system is not doing any sanitizing and allowing script tags with java-script code and is executable on the client side.

Vulnerability Walk-thru:

1. Go to the XSS Section
2. Next I check to see if Direct Chat 1 was exploitable to XSS
3. Using the following XSS I was able to inject a javascript alert in Direct Chat 1 "<script>alert(1)</script>".



VWA Security Report



Recommendations:

We should use a standard library that does field sanitizing for XSS.

https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html

VWA Security Report

TEST20101804 – XSS - Critical

Vulnerability Exploited: XSS

Severity: Critical

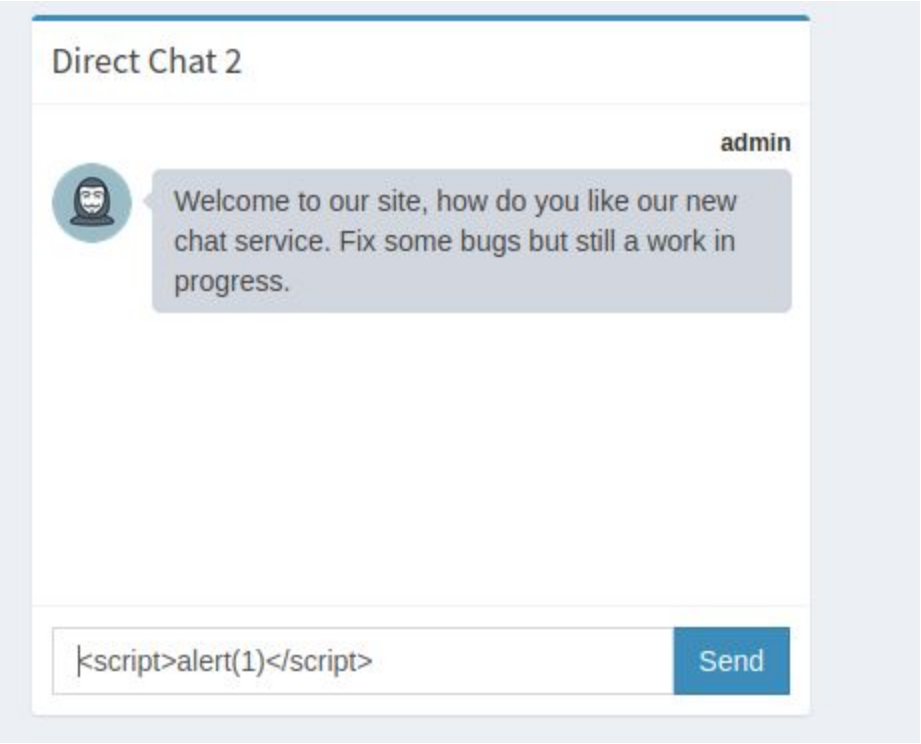
System: VWA Web Application

Vulnerability Explanation:

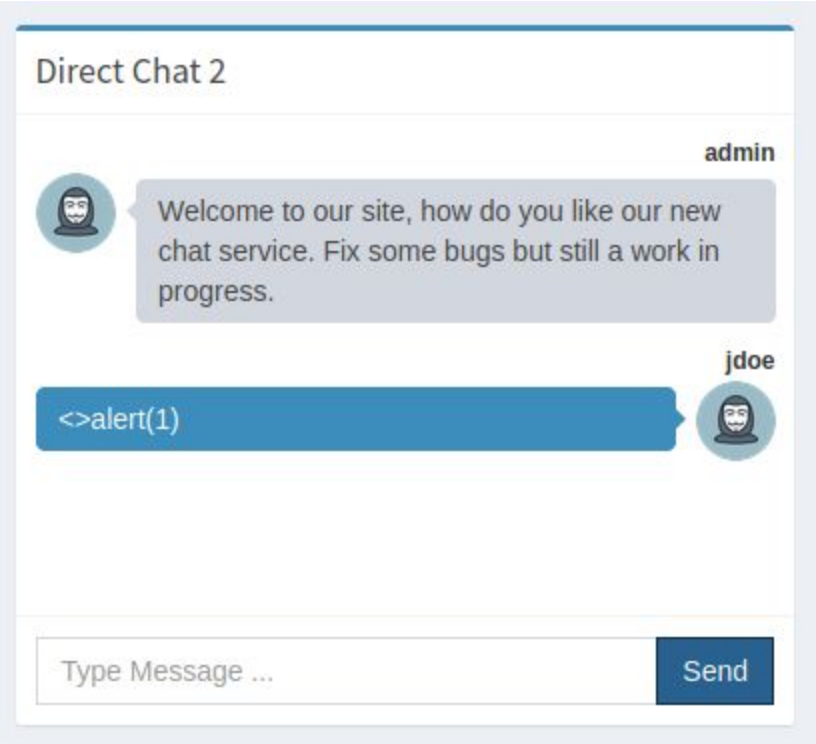
In the XSS Area we have an internal Chat service, this system seems to be sanitizing for "script" within the messages, but its not doing a full URL sanitizing and allowing img tags. That then could contain java-script code and is executable on the client side.

Vulnerability Walk-thru:

1. Go to the XSS Section
2. Next I check to see if Direct Chat 2 was exploitable to XSS
3. I first try a basic XSS "<script>alert(1)</script>" to see if the code is exploitable.

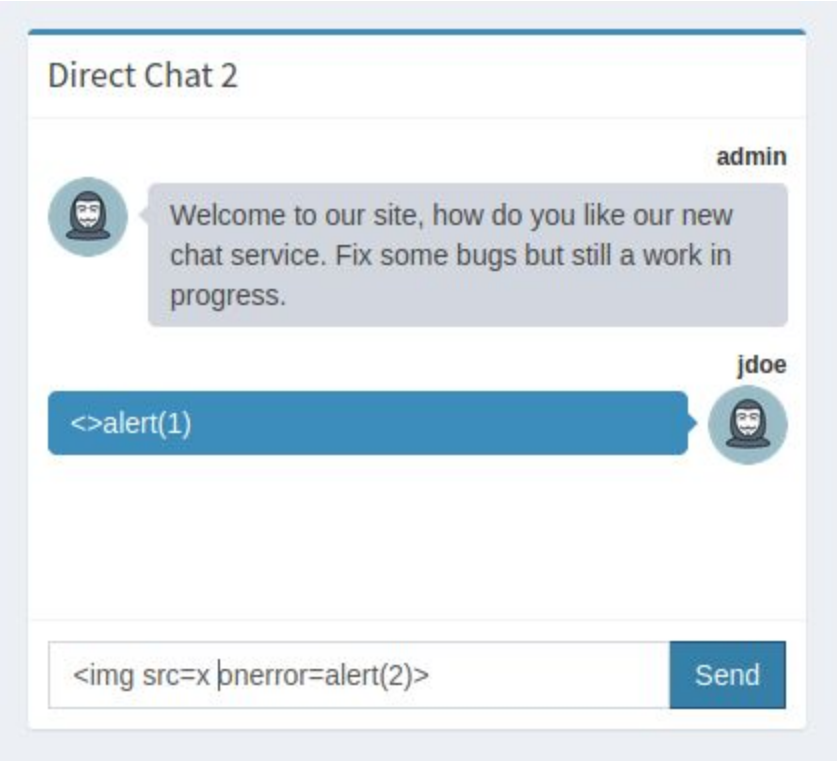


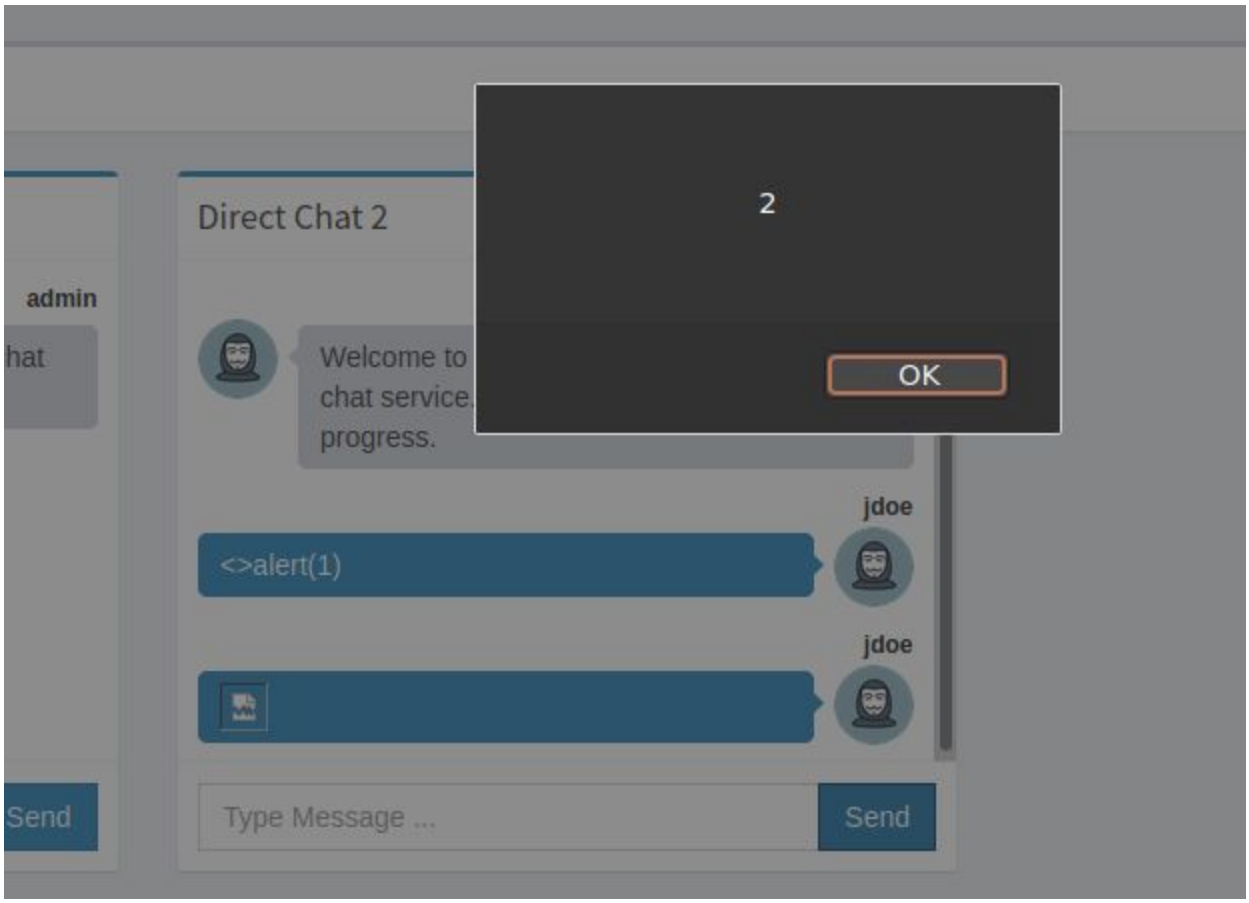
VWA Security Report



4. But the chat seems to be doing keyword replacement on the word “script”.
5. Next I try a more advanced XSS, by inserting a im tag and setting the src to a non exist file with a onerror exec.

VWA Security Report





VWA Security Report

Recommendations:

We should use a standard library that does field sanitizing for XSS.

https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html

VWA Security Report

TEST20101805 – Broken Auth - High

Vulnerability Exploited: Broken Auth

Severity: High

System: VWA Web Application

Vulnerability Explanation:

An unauthorized user is able to brute force the login page and access our system.

Vulnerability Walk-thru:

1. Go to the Broken Auth Section
2. Using a python script call "bruteforce.py"
3. I ran the following cmd.
   a. python bruteforce.py -U test-username.txt -P test-password.txt -f False http://0.0.0.0:3000/brokenauth/
4. After a min of running it was a find a working username and password combination that allowed me to login.

```
python bruteforce.py -U test-username.txt -P test-password.txt -f False http://0.0.0.0:3000/brokenauth/
[+] Login Found! {'username': 'test', 'password': 'klaster'}
This is a demo code used for this training.
```

Recommendations:

We need to track fail login attempts on both the user account and IP address, then block the IP address if they exceed 5 fail logins for 15 minutes and also alert our SOC team of this event.

https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

VWA Security Report

TEST20101806 – Broken Auth - High

Vulnerability Exploited: Broken Auth

Severity: High

System: VWA Web Application

Vulnerability Explanation:

An unauthorized user is able to brute force the login page and access our system.

Vulnerability Walk-thru:

1. Go to the Broken Auth Section
2. Using a python script call "bruteforce.py"
3. I ran the following cmd.
   a. python bruteforce.py -U test-username.txt -P test-password.txt -f False http://0.0.0.0:3000/brokenauth/
4. After a min of running it was a find a working username and password combination that allowed me to login.

```
python bruteforce.py -U test-username.txt -P test-password.txt -f False http://0.0.0.0:3000/brokenauth/login2
[+] Login Found! {'username': 'user', 'password': 'dragon'}
This is a demo code used for this training.
```

Recommendations:

We need to track fail login attempts on both the user account and IP address, then block the IP address if they exceed 5 fail logins for 15 minutes and also alert our SOC team of this event. https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html