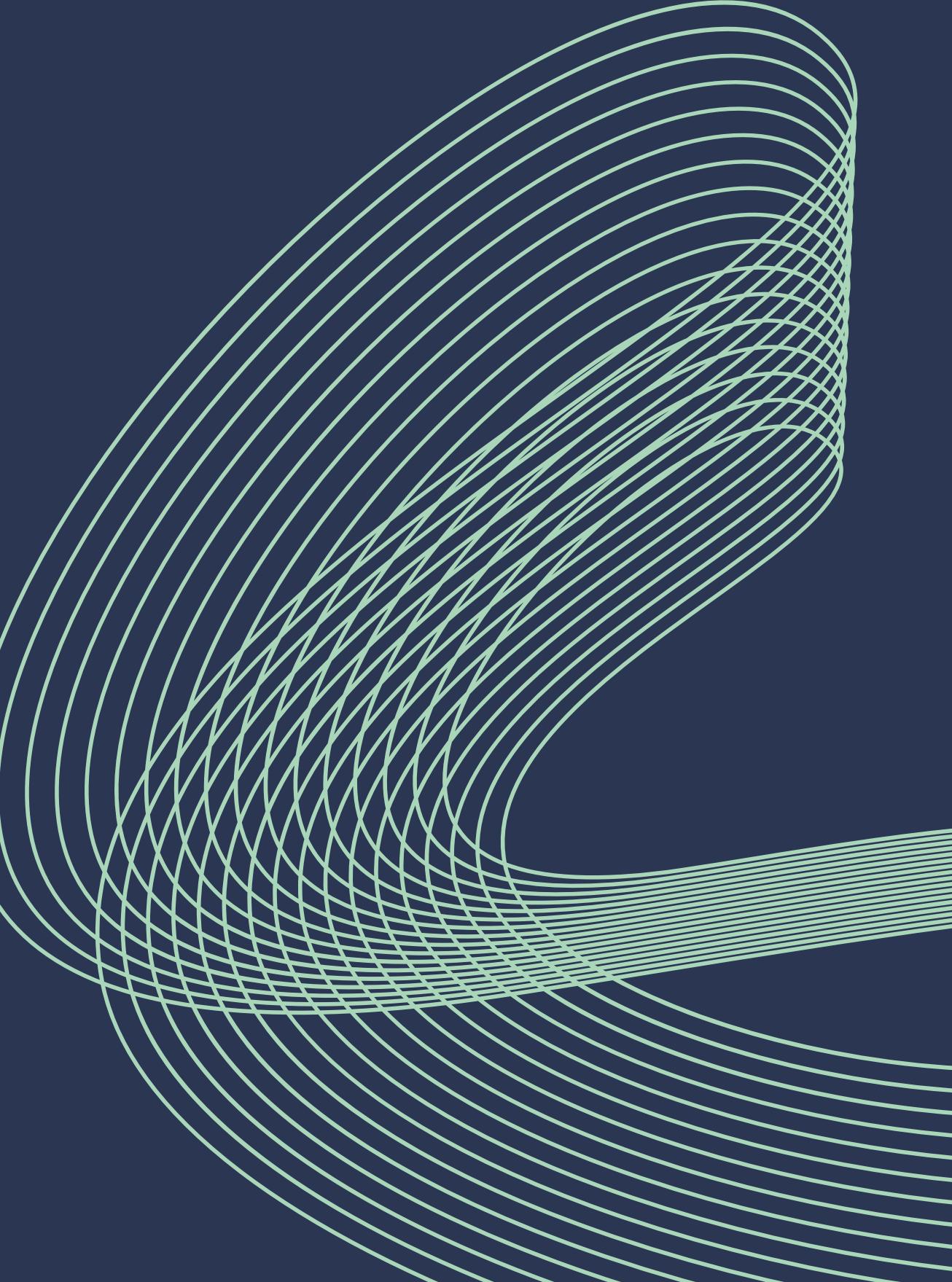


# VAPT REPORT

bWapp

PRESENTED TO  
Md Mehedi Hasan Babu

PRESENTED BY  
Md Rezvee Parvez



# AGENDA

Executive Summary	3
Client Information	4
Scope of Assessment	5
Methodology	6
Nikto Scan Report	7
PHP Code injection	8
Broken Auth	10
IDOR	12
Directory Transversal	14
RFI & LFI	16
Contact Information	18



# EXECUTIVE SUMMARY

This Vulnerability Assessment and Penetration Testing (VAPT) engagement was conducted for Bwapp with the objective of evaluating the security posture of its critical systems, applications, and supporting infrastructure. The assessment was designed to identify security weaknesses, determine the feasibility of potential attack vectors, and provide actionable recommendations to strengthen the organization's overall cybersecurity resilience.



[BACK TO AGENDA](#)

# CLIENT INFO

Client name : bWapp

Date : 09 December, 2025

Assesment Type : Web Application VAPT



Client : bWapp

[BACK TO AGENDA](#)



---

# SCOPE OF ASSESSMENT

---

The assessment was limited to the following vulnerabilities:

- Php Code Injection
- Captcha Bypass
- IDOR-Secret Change
- Directory Traversal
- Remote and Local file inclusion

[BACK TO AGENDA](#)



# METHODOLOGY

- Nikto Vulnerability Scan for the web Server
- Burpsuite for request Interception
- Manual exploitation
- Black-Box approach
- All findings are validated and documented



Client : bWapp

[BACK TO AGENDA](#)

# NIKTO SCAN REPORT

## Key Findings:

- Directory Indexing Enabled (High)

open directories: /passwords/logs/documents

Risk: Sensative files can be viewed

Fix: Disable directory listing

- Sensative files accessable (High)

config.inc - contains user and password

phpinfo.php- Gets the php information

Risk:Information disclosure

Fix: Remove or restrict these files

- Outdated Software

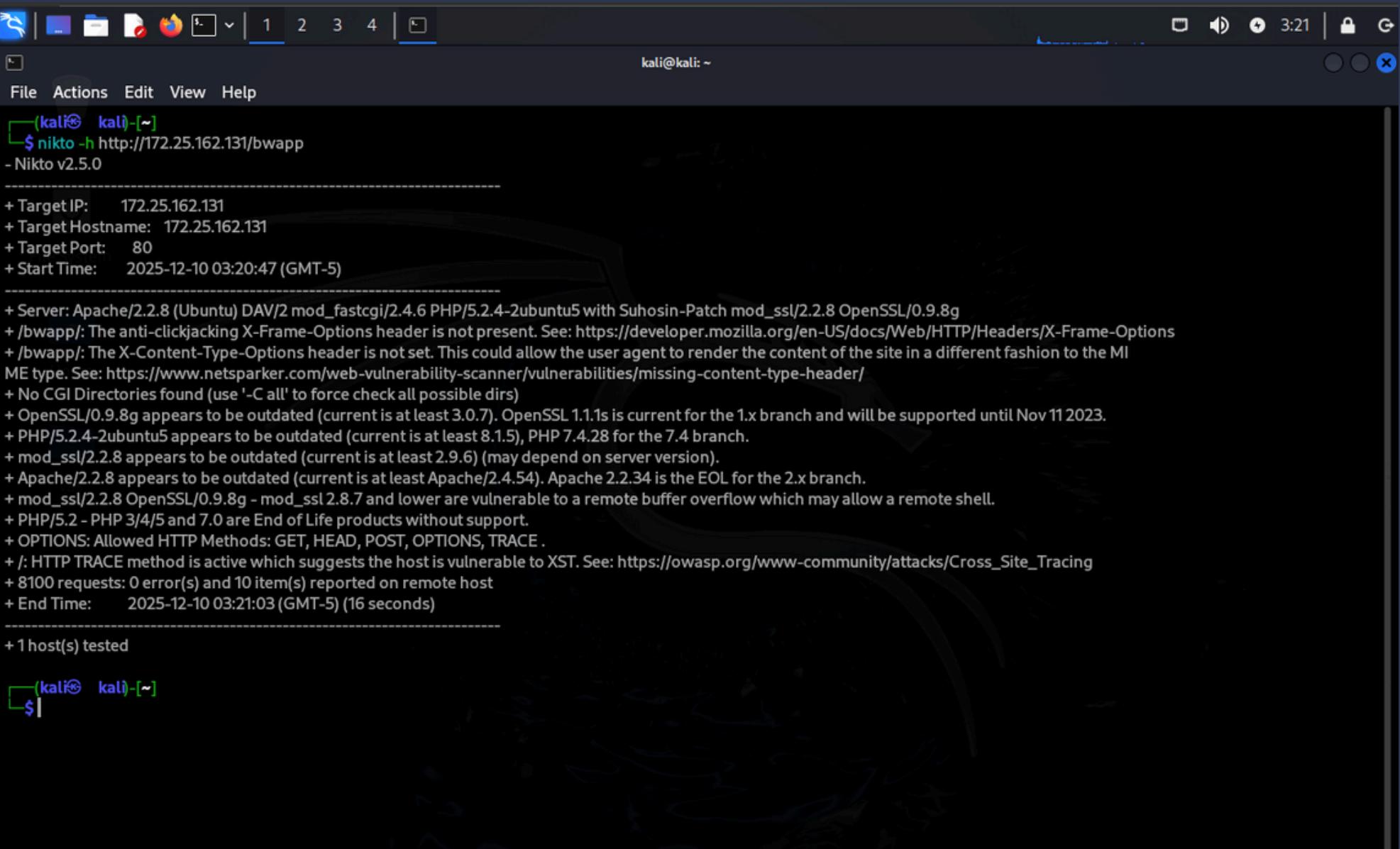
php5.2.4

Apache 2.2.8

OpenSSL 0.9.8

Risk: Contains known vulnerabilities

Fix: Upgrade to latest version



The screenshot shows a terminal window on a Kali Linux system (kali:kali) running the Nikto web scanner against the target IP 172.25.162.131. The application being scanned is bWapp. The output of the scan is as follows:

```
(kali㉿ kali)-[~]
$ nikto -h http://172.25.162.131/bwapp
- Nikto v2.5.0

+ Target IP: 172.25.162.131
+ Target Hostname: 172.25.162.131
+ Target Port: 80
+ Start Time: 2025-12-10 03:20:47 (GMT-5)

+ Server: Apache/2.2.8 (Ubuntu) DAV/2 mod_fastcgi/2.4.6 PHP/5.2.4-2ubuntu5 with Suhosin-Patch mod_ssl/2.2.8 OpenSSL/0.9.8g
+ /bwapp/: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /bwapp/: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OpenSSL/0.9.8g appears to be outdated (current is at least 3.0.7). OpenSSL 1.1.1s is current for the 1.x branch and will be supported until Nov 11 2023.
+ PHP/5.2.4-2ubuntu5 appears to be outdated (current is at least 8.1.5), PHP 7.4.28 for the 7.4 branch.
+ mod_ssl/2.2.8 appears to be outdated (current is at least 2.9.6) (may depend on server version).
+ Apache/2.2.8 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ mod_ssl/2.2.8 OpenSSL/0.9.8g - mod_ssl 2.8.7 and lower are vulnerable to a remote buffer overflow which may allow a remote shell.
+ PHP/5.2 - PHP 3/4/5 and 7.0 are End of Life products without support.
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE.
+ /: HTTP TRACE method is active which suggests the host is vulnerable to XST. See: https://owasp.org/www-community/attacks/Cross_Site_Tracing
+ 8100 requests: 0 error(s) and 10 item(s) reported on remote host
+ End Time: 2025-12-10 03:21:03 (GMT-5) (16 seconds)

+ 1 host(s) tested

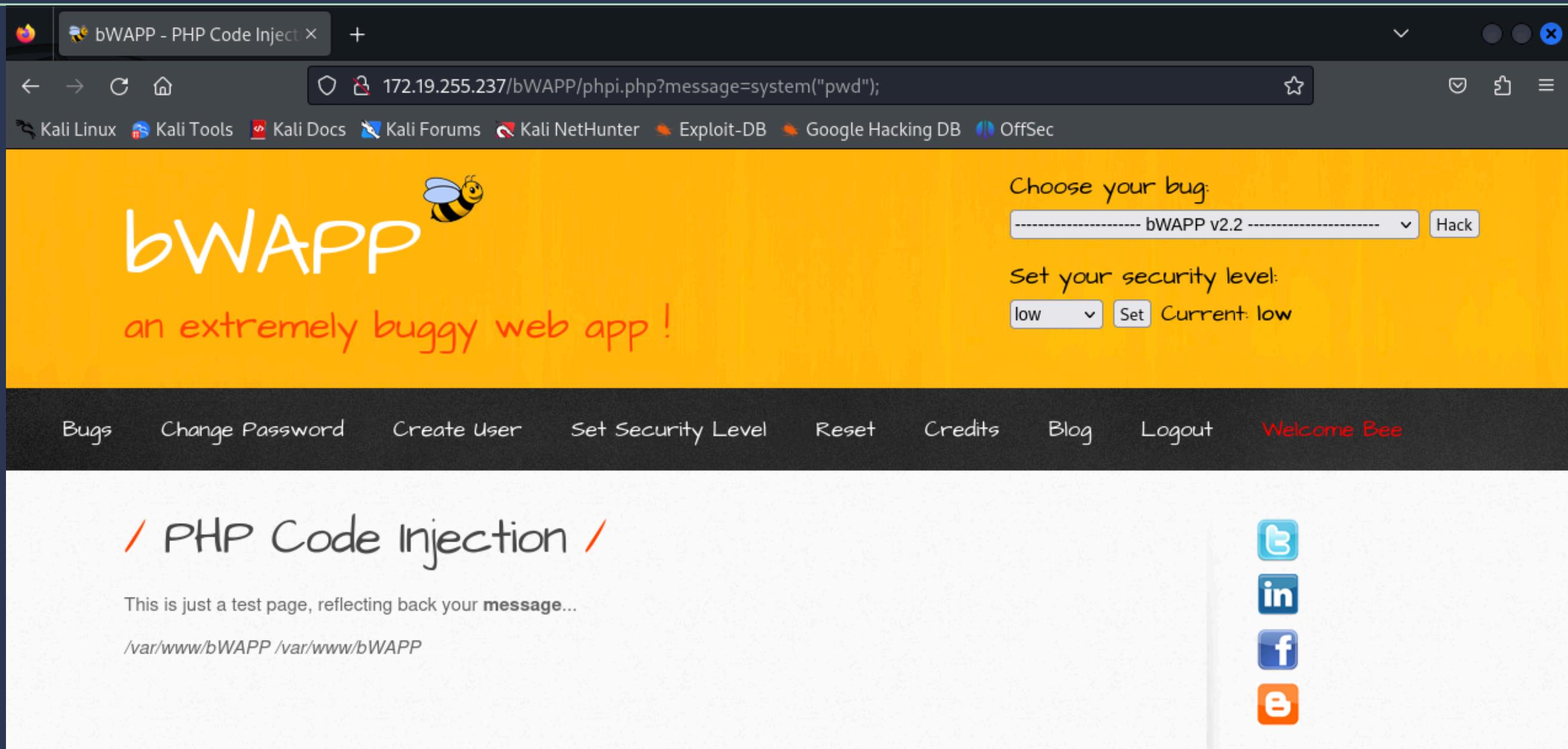
(kali㉿ kali)-[~]
$
```

## Target Information

- Target IP: 172.25.162.131
- Application : bWapp

# PHP CODE INJECTION

PHP code injection is a security vulnerability where an attacker is able to insert and execute malicious PHP code on a server. This happens when a web application takes user-supplied input and processes it as PHP without proper validation or sanitization.



## Vulnerability : Php Code Injection

**Description:** The web application does not properly check or filter user input. Because of this, an attacker can inject OS commands into a PHP function (like `system()`), allowing them to execute commands on the server. This leads to Remote Code Execution (RCE).

# PHP CODE INJECTION

## Affected Assets:

- URL: 172.19.255.237/bWapp/phpi.php
- Vulnerable Parameter : System

Severity: Critical (CVSS:9.8)

An attacker can gain full access to server

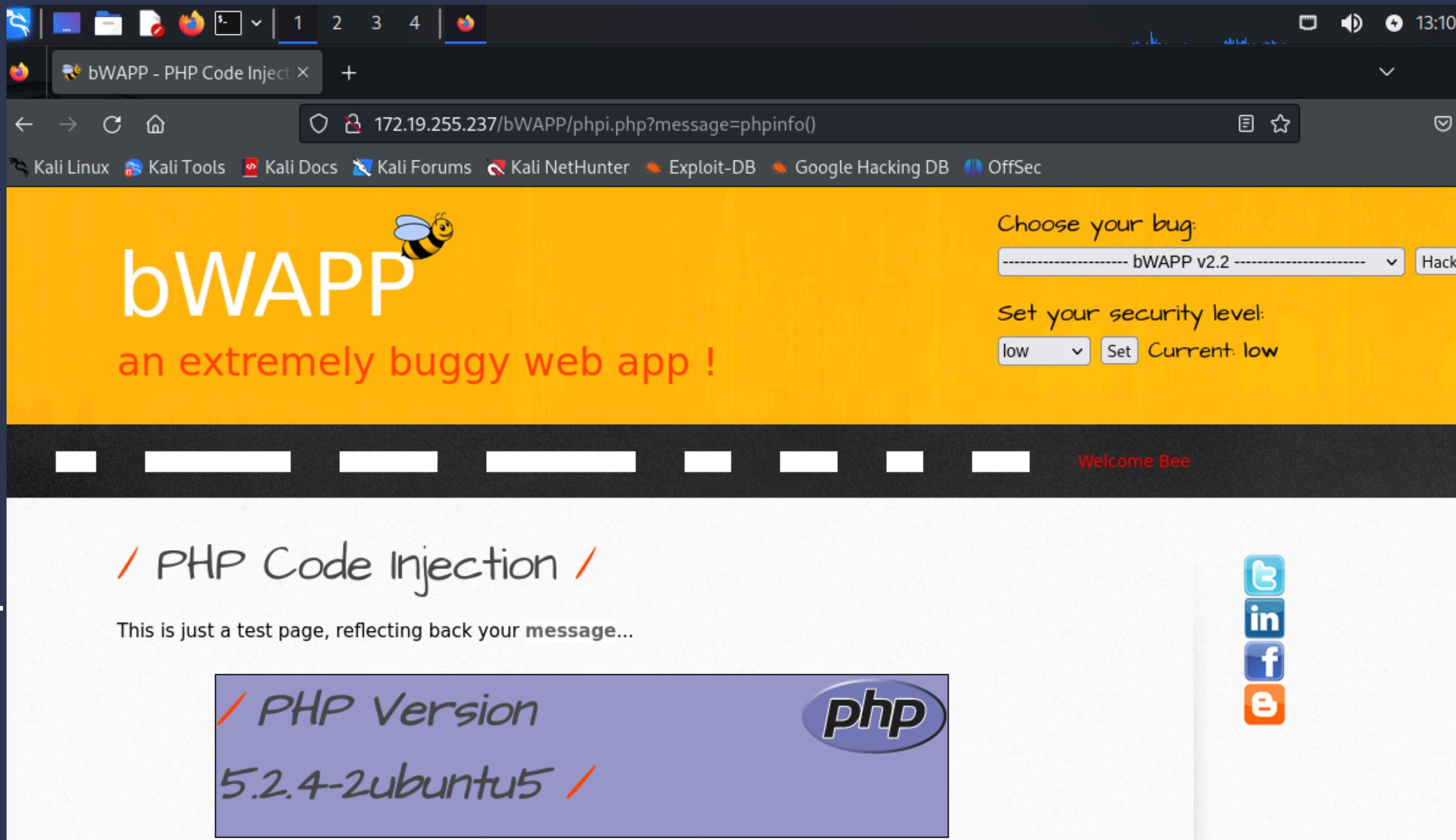
## Steps to Reproduce (PoC)

- Open the vulnerable URL.
2. Inject a command into the system parameter.
3. Payload used: phpinfo().

The server's php information pops up

## Remediation:

- Validate and sanitize all user inputs
- Avoid using dangerous PHP functions (system(), exec(), etc.)
- Disable risky functions in php.ini under disable\_functions
- Use whitelisting when only specific commands or inputs are allowed



# BROKEN AUTHENTICATION - CAPTCHA BYPASS

CAPTCHA bypass is when an attacker finds a way to defeat or avoid a CAPTCHA challenge that is meant to distinguish humans from bots.

## / Broken Auth. - CAPTCHA Bypassing /

Enter your credentials (bee/bug).

Login:

Password:

Re-enter CAPTCHA:

Successful login!

### Vulnerability : Broken Authentication

**Description:** The loginpage uses a CAPTCHA for verification, but the server does not properly validate it. The CAPTCHA answer is not required during login, and the server accepts the request even if the captcha\_user parameter is missing. This allows an attacker to log in without solving the CAPTCHA

# BROKEN AUTHENTICATION - CAPTCHA BYPASS

CAPTCHA bypass is when an attacker finds a way to defeat or avoid a CAPTCHA challenge that is meant to distinguish humans from bots.

- 
- Impact

Severity : High

CVSS Score : 9.4

- Proof of Concept

A login request with valid credentials and a CAPTCHA value was captured.

Using Burp Suite, the CAPTCHA parameter was removed from the POST request:

log in= bee&password= bug&form=submit

- Remediation

Enforce server-side CAPTCHA validation. Reject login attempts if the CAPTCHA is missing or incorrect.

Store the CAPTCHA value in the session and validate it properly.

Validate all user input to prevent similar authentication bypass issues.

---

# INSECURE DIRECT OBJECT REFERENCE (IDOR)

The application trusts user-controlled input for sensitive objects, so an attacker can change a parameter (like secret=123) and access or modify another user's data.

The screenshot shows the bWAPP web application interface. At the top, there is a yellow header bar with the bWAPP logo (a bee icon next to the text "bWAPP") and the tagline "an extremely buggy web app!". To the right of the logo are dropdown menus for "Choose your bug:" (set to "bWAPP v2.2") and "Hack", and a "Set security level" button (set to "low"). Below the header is a black navigation bar with links: Bugs, Change Password, Create User, Set Security Level, Reset, Credits, Blog, Logout, and Welcome Bee. The main content area has a white background. It features a title "Insecure DOR (Change Secret)" in a stylized font. Below the title is a form field labeled "Change your secret." with a placeholder "New secret:". A "Change" button is located below the input field. A green success message "The secret has been changed!" is displayed at the bottom of the form. To the right of the form, there is a vertical column of social media sharing icons for Twitter, LinkedIn, Facebook, and Email.

## Vulnerability : Insecure Direct Object Reference

**Description:** The application contains an IDOR vulnerability in the user secret-change functionality. When updating the secret phrase, the application relies directly on a user-controlled parameter login=bee in the POST request. The server does not verify whether the authenticated user is authorized to modify the secret of the user specified in this parameter.

# INSECURE DIRECT OBJECT REFERENCE (IDOR)

The application trusts user-controlled input for sensitive objects, so an attacker can change a parameter (like secret=123) and access or modify another user's data.

---

## **Impact**

**Severity : High**

**CVSS Score : 8.2**

This vulnerability allows unauthorized modification of sensitive user information. An attacker can change another user's secret phrase, which may be used for account recovery or secondary authentication. This compromises both the integrity and confidentiality of user accounts.

## **Proof of Concept:**

RequestCapture

Exploitation (IDOR)

Result

## **Remediation:**

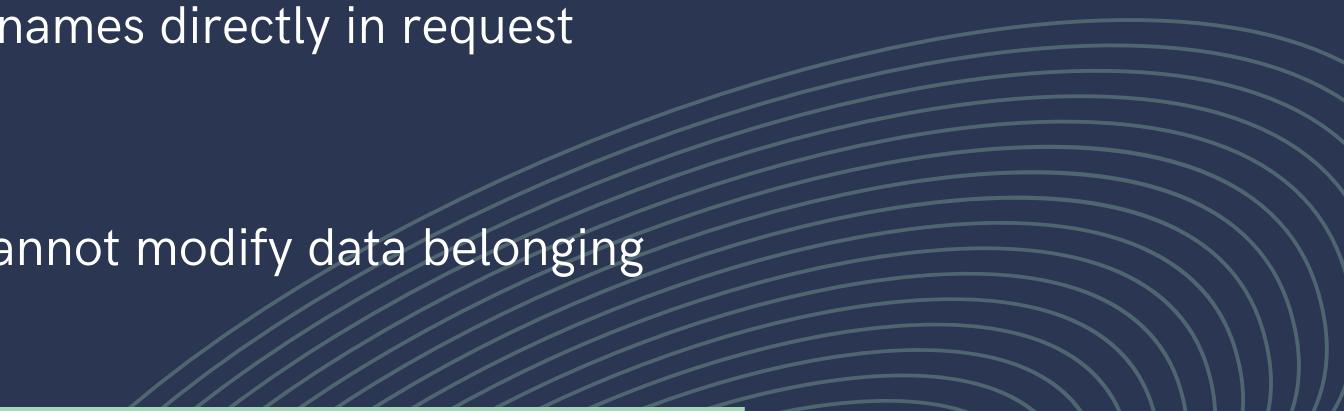
Enforce Server-Side Authorization Checks: Validate that the user ID in the request matches the authenticated user's ID from the session. Reject any mismatched or unauthorized requests.

### Avoid Direct Object References:

Use indirect identifiers (e.g., mapped tokens) instead of exposing user IDs or login names directly in request parameters.

### Apply Least Privilege:

Ensure user sessions only have permissions necessary for their own account and cannot modify data belonging to other users.



# DIRECTORY TRAVERSAL

Directory traversal is a vulnerability where an attacker manipulates file-path input to access files and folders outside the intended directory.

The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal title is 'Directory Traversal - Files'. It displays a list of system users and their corresponding UIDs, GIDs, and shell paths. The list includes root, daemon, bin, sys, sync, games, man, lp, mail, news, uucp, proxy, www-data, backup, list, irc, gnats, nobody, libuuid, and dhcp. The terminal has a dark theme with white text. At the top of the screen, there is a horizontal menu bar with various links: Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, Google Hacking DB, and UTSec. On the right side of the terminal window, there are four social media sharing icons: Twitter, LinkedIn, Facebook, and Email.

```
root:x:0:0:root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
```

## Vulnerability : Directory Traversal

**Description:** The application is vulnerable to a Directory Traversal attack. The page parameter is directly used to build the file path without proper validation. Because of this, an attacker can use .. sequences to access files outside the web root directory.

# DIRECTORY TRAVERSAL

---

**Impact**

**Severity : High**

**CVSS Score : 7.5**

**A successful attack allows an attacker to read arbitrary system files, which can leads to Information Disclosure, System Enumeration, Exposure of sensitive configuration files such as database credentials**

**Proof of Concept:**

- Exploitation

The attacker used atraversal path to access a sensitive system file

- Result

The server returned the contents

**Remediation:**

Validate Input

Canonicalize Paths

Use Whitelists

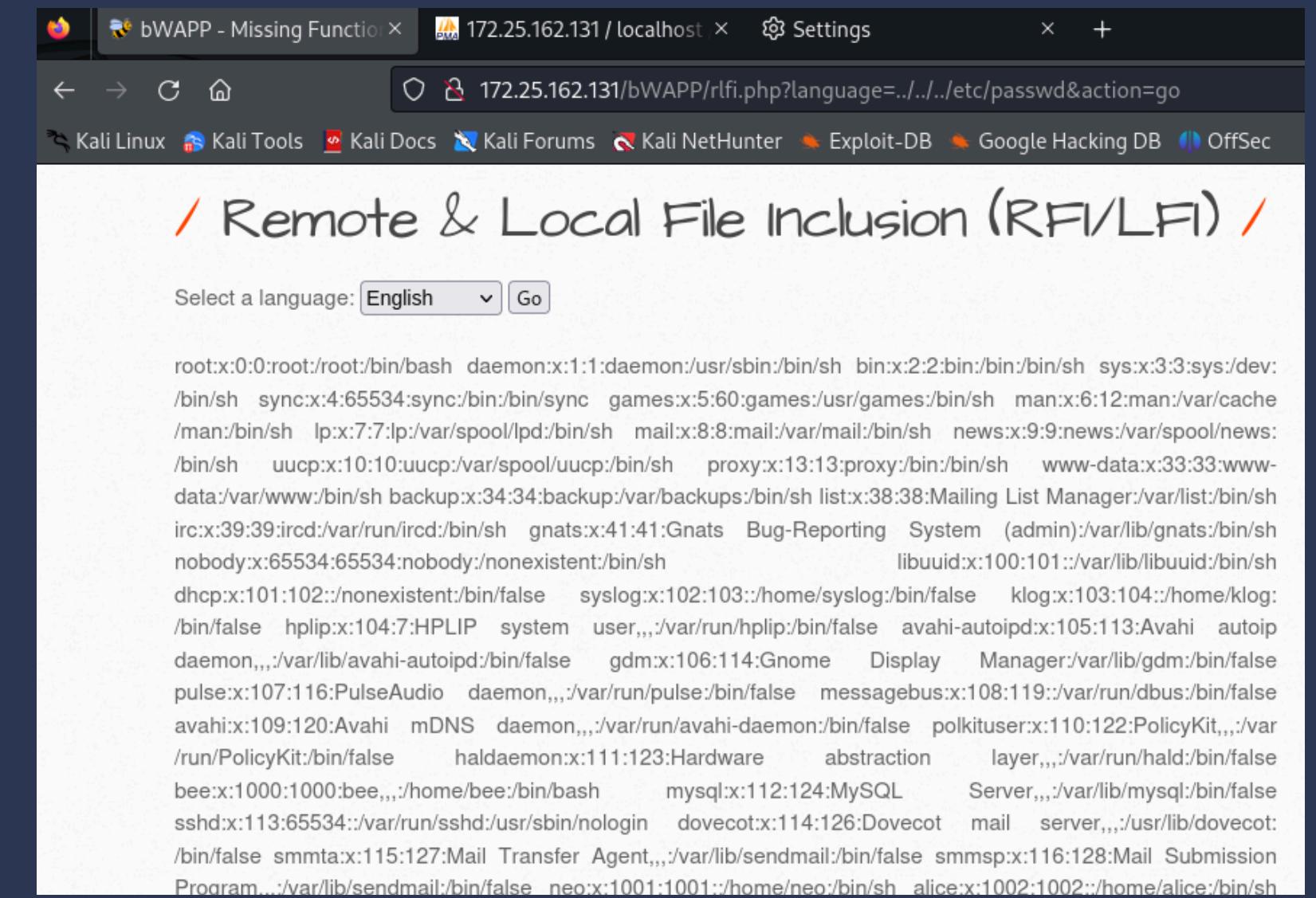
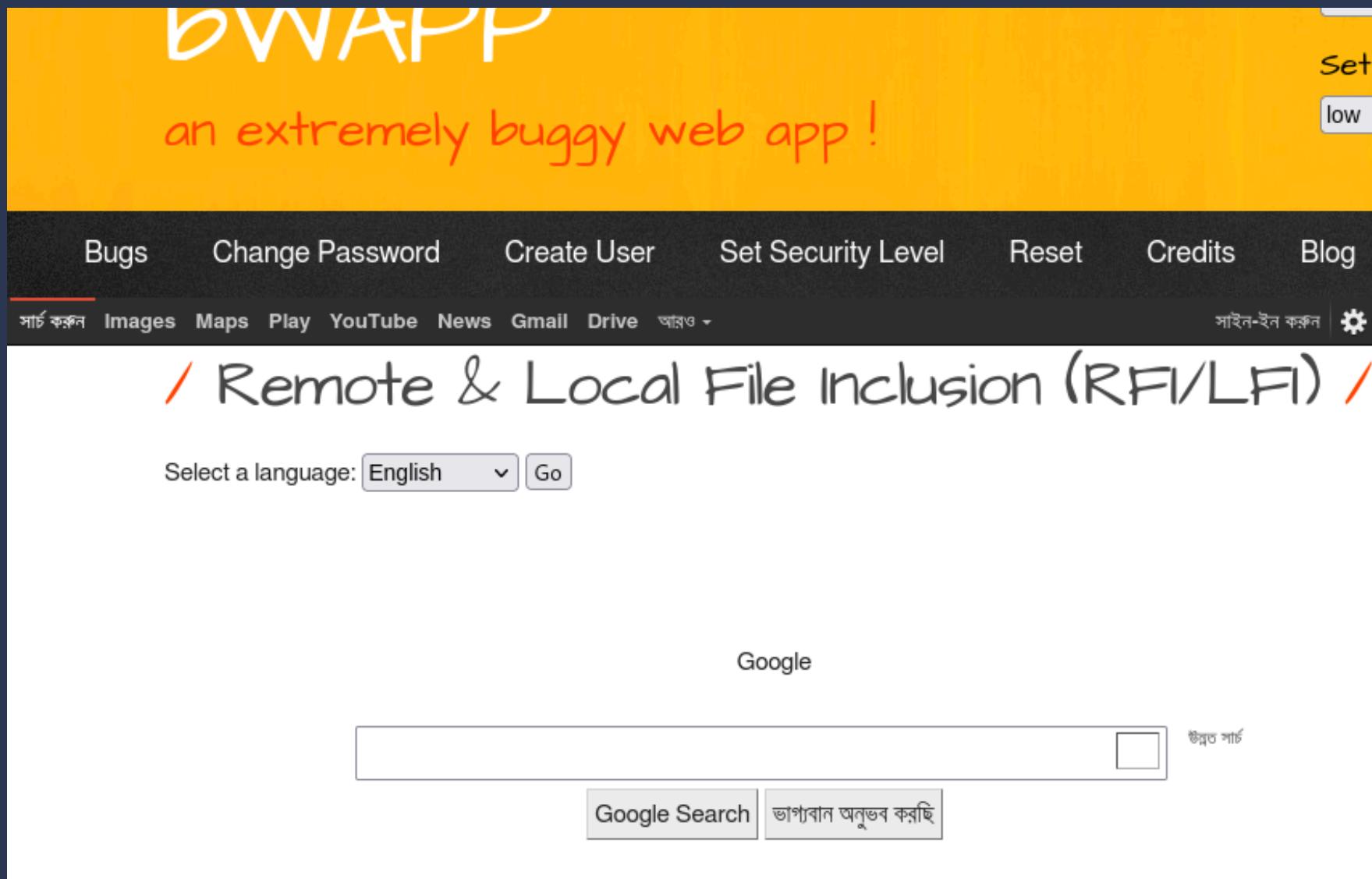
Least Priviledge

---



# REMOTE AND LOCAL FILE INCLUSION

The attacker tricks the application into loading or executing files already on the server, such as logs or configuration files.



## Vulnerability : Directory Traversal

**Description:** The application is vulnerable to a Directory Traversal attack. The page parameter is directly used to build the file path without proper validation. Because of this, an attacker can use .. sequences to access files outside the web root directory.

# REMOTE AND LOCAL FILE INCLUSION

---

## Impact

**Severity : Critical**

**CVSS Score : 9.0**

**A successful attack allows an attacker to read arbitrary system files, which can lead to Information Disclosure, System Enumeration, Exposure of sensitive configuration files such as database credentials**

## Proof of Concept:

- RFI Example

The server returned Google.com

- LFI Example

The server returned the contents of the server

## Remediation:

**Sanitize and validate all input (remove ../, http://, etc.).**

Implement strict allowlists for files and paths.

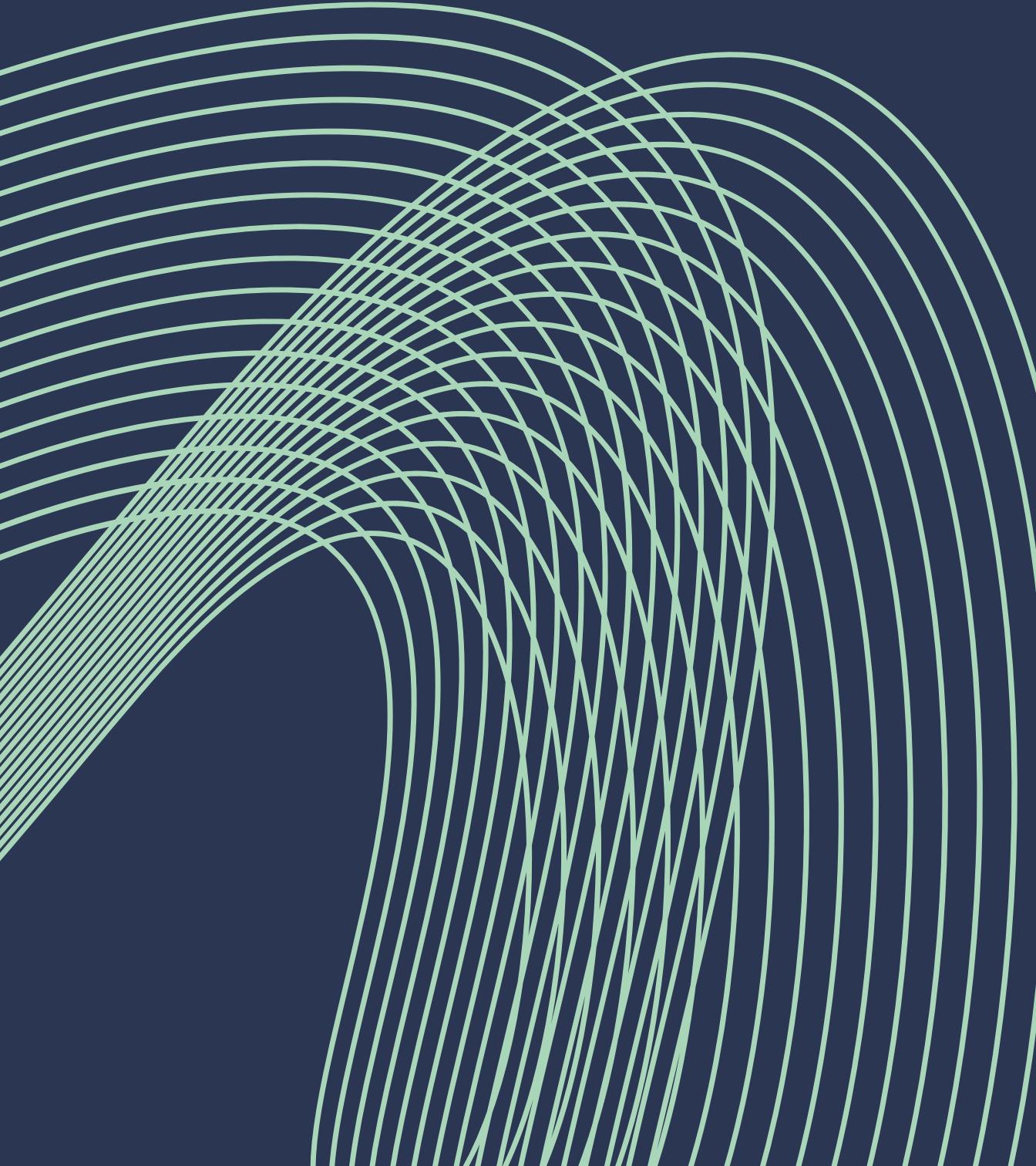
Disable allow\_url\_fopen and allow\_url\_include in PHP.

Use secure frameworks or templating systems instead of direct includes.

Set proper file permissions to limit accessible directories.

Keep server and PHP versions updated with security patches.

---



 THANK YOU !

---

FOR INQUIRIES,  
CONTACT US.

---

Email

[rezvx@proton.me](mailto:rezvx@proton.me)

Social Media

[www.linkedin.com/in/rezvx](https://www.linkedin.com/in/rezvx)

[www.rezv.me](http://www.rezv.me)