



OFFENSIVE SECURITY



OFFENSIVE SECURITY

- Penetration Testing
- Vulnerability Assessment
- Red Teaming
- Phishing
- Web, Cloud, Network, Active Directory, Mobile, IOT, Hardware, VR, Web3, AI
- OSINT



WHOAMI

- OSCP, CISM, eMAPT, VHL Advanced+, Security+
- MBA
- Information Security Officer
- CEO & Co-Founder
- BJJ & Chess



HACKING MINDSET

- Think like an attacker
- What can we brake?
- What does not work?
- Fuzzing
- Reading the documentation
- Enumeration is the key
- Keep it super simple when targeting users – KISS principle

MOST IMPORTANT HACKING TOOL

A close-up, low-key photograph of a hand typing on a laptop keyboard. The scene is dimly lit, with the primary light source highlighting the keys and the hand's movement. The background is dark and out of focus, emphasizing the action of typing.

- What is the most important hacking tool?

HACKING MINDSET – EXERCISE 1



Do an open source intelligence search on yourself



What can you find online?



Did you find something that you forgot about or did not know about? If not, search again :D



Idea to start: Google Hacking "Name Surname" City



If you were to attack the user (yourself), how would you steal their social media accounts – keep in mind that this is an exercise that you should do on yourself only or where you have permission

A person wearing a dark hoodie is centered in the background, looking directly at the camera with a neutral expression. The lighting is dim, highlighting their face against a dark background.

WHAT ETHICAL HACKERS DON'T DO

- **No ethics**
- Credit cards
- WiFi Access Points
- Police & Military radio frequency
- Ordering pizza, clothing, etc.
- How to work in NSA

EXERCISE 2 - OSINT

- Where was Donald Trump on July 29th 2023 at 6PM?





Severity

Severity of vulnerability is calculated by impact on application/business



Recommendation

Recommendation how to mitigate vulnerability



Risk

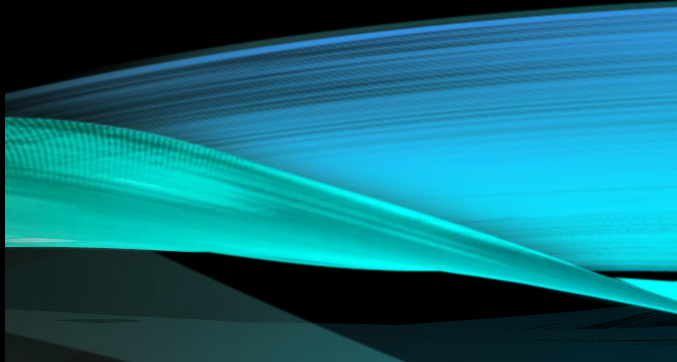
Risk count any action that might result in an impact



Description

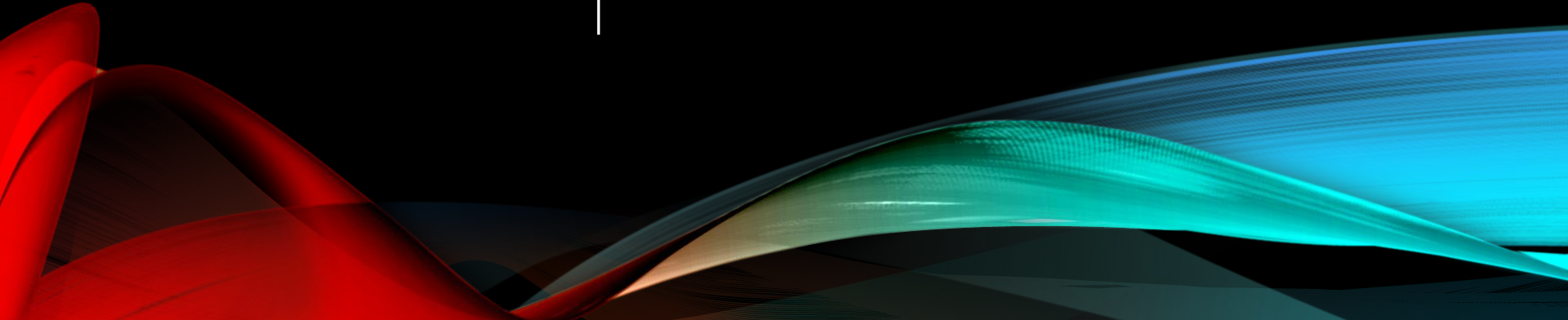
Description of particular vulnerability

VULNERABILITY BASICS

An abstract graphic in the bottom right corner featuring flowing, wavy lines in shades of blue and teal, creating a sense of motion and depth against the dark background.

What is the most
common vulnerability
found on the web today?

EXERCISE 3





COMMON BACKEND VULNERABILITIES

- Missing access control
- User enumeration
- Weak password requirements
- Outdated version of server
- Missing security headers

MISSING ACCESS CONTROLS

Description

- We have discovered that a user with a certain role can access `/api/users/*` endpoint. Accessing that particular endpoint let the potentially malicious user list all other users as well as edit their details including the user role itself

Critical

MISSING ACCESS CONTROLS



Risk



Exploiting this vulnerability allows a malicious user with Psa role to escalate privileges to Admin role by sending an HTTP request directly to the API endpoint. A malicious user with Psa role can also access other keeper user accounts and change their details



Recommendations



Implement access control on the API as well, not only on the frontend application

USER ENUMERATION

Low

Description

Attacker can do user enumeration by brute force technique to either guess or confirm valid users in a system by email address

USER ENUMERATION



Risk



User enumeration represents a risk since it opens the door for further attacks such as password brute force as well as social engineering attacks on the users



Recommendations



The error messages should be generic in order not to reveal any user information. For example the error message could be: "If the email exists in our system, we will send an email"

WEAK PASSWORD REQUIREMENTS

Description

- The product does not require that users should have strong passwords, which makes it easier for attackers to compromise user accounts

Risk

- When passwords are not strong enough, risk that attacker will compromise user accounts is much bigger

Medium

WEAK PASSWORD REQUIREMENTS

Recommendations

A product's design should require adherence to an appropriate password policy.

Specific
password
requirements
depend
strongly on
contextual
factors

Enforcement of a password complexity

Restrictions against password reuse

Restrictions against using common passwords

Restrictions against using contextual string in the password
(e.g., user id, app name)

OUTDATED VERSION OF SERVER

Description

- During the testing, we discovered that the targeted application is running on an outdated Apache server. We found out that the Apache version is 2.4.29 which was released in 2017 and has a record of a few CVE vulnerabilities

Low

OUTDATED VERSION OF SERVER



Risk



Recommendations



Found vulnerabilities related to this specific Apache version, range from local privilege escalation to denial of service vulnerabilities. We couldn't find any public proof of concepts and couldn't confirm either of the vulnerabilities



Update Apache server to the latest version

SECURITY HEADERS MISSING

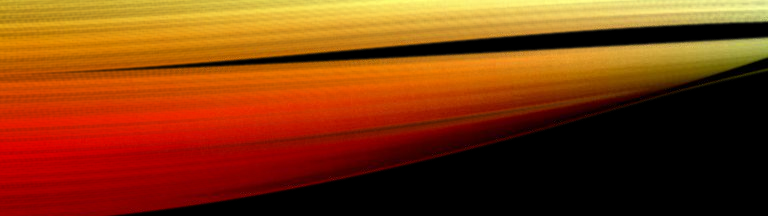
Description

It was observed that application was not sending security headers and some other headers are revealing more information about application than it is needed

Low

Risk

If you are not using security headers, your web application is under a risk of common attacks, including cross-site scripting (XSS), clickjacking, etc.



SECURITY HEADERS MISSING

- Recommendations

- It is recommended to implement the following headers:

Strict-Transport-Security

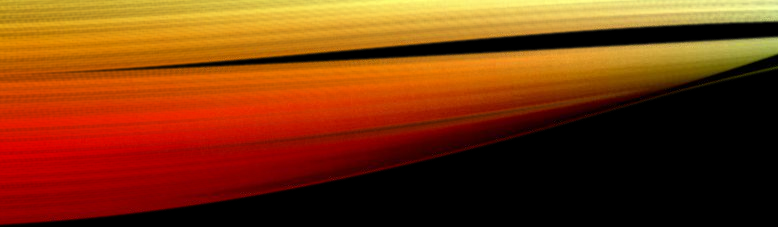
Content-Security-Policy

X-Frame-Options

X-Content-Type-Options

Referrer-Policy

Permissions-Policy



EXERCISE 4

- What is the most critical impact that you can think of or find that happened due to missing security headers?



COMMON FRONT-END VULNERABILITIES

Cross site scripting (XSS)

Source code leak

Sensitive data stored

CROSS SITE SCRIPTING (XSS)

Description

Medium

It's possible to conduct cross site scripting (XSS) attack on todo form. The vulnerability that leads to this attack is that data passed by the user is incorrectly processed and interpreted as HTML and JavaScript code. Thus, the adversary can craft a special payload by adding crafted JavaScript code in a vulnerable parameter

Payload used: `<x onclick=alert(1)>click this!`

CROSS SITE SCRIPTING (XSS)



Risks



The injection of malicious JavaScript code opens the door for further attacks like "account takeover" or "stolen credentials"



Recommendations

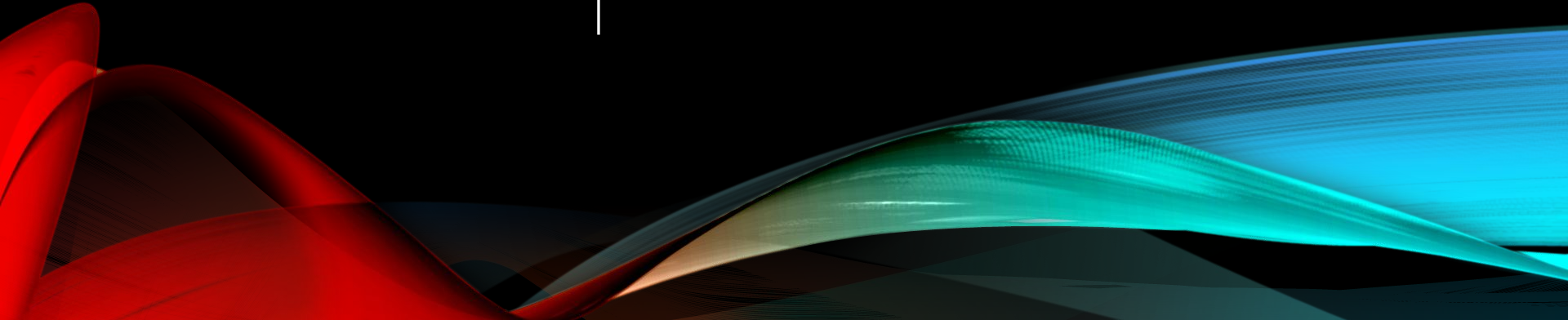


Sanitize input passed by the user and don't treat it like code on a todo form, but like text

Try it out

<https://xss-game.appspot.com>

CROSS SITE SCRIPTING (XSS) – EXERCISE 5



SOURCE CODE LEAK

Description

- We have discovered that React's map file is publicly available at the <https://website/packs/js/path.js.map>
- By having the map file we can restore the frontend's unobfuscated source code

Medium



SOURCE CODE LEAK

Risk

- By having a map file an attacker can restore JavaScript source code which can help in understanding the application logic, discover API endpoints, and even reveal comments left in the code by developers

Recommendations

- We recommend removing the map file or restricting its access to people that need it

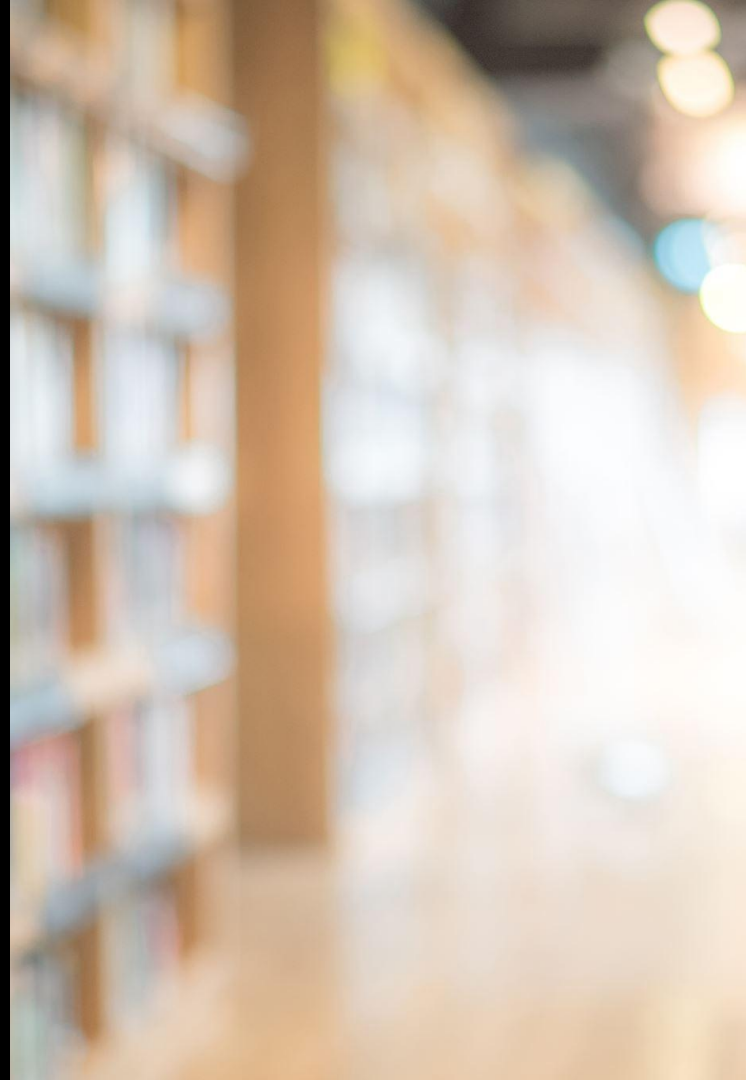
SENSITIVE DATA STORED

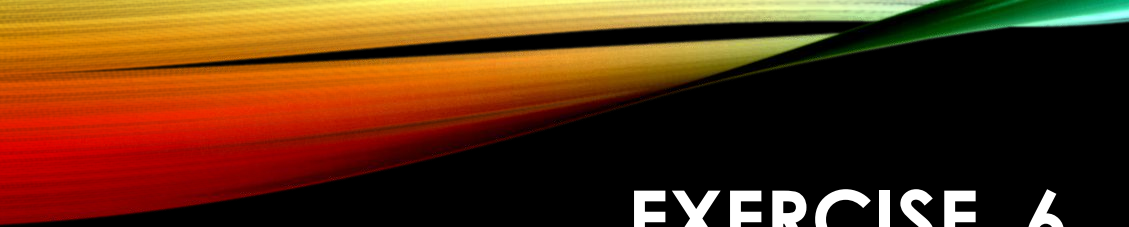
- The application stores sensitive information in the client-side storage
- This exposes the information to unauthorized read operations
- After** deobfuscation of front-end source code, hardcoded valid
- API keys have been found as comment

High

INTERESTING RESEARCH TOPICS

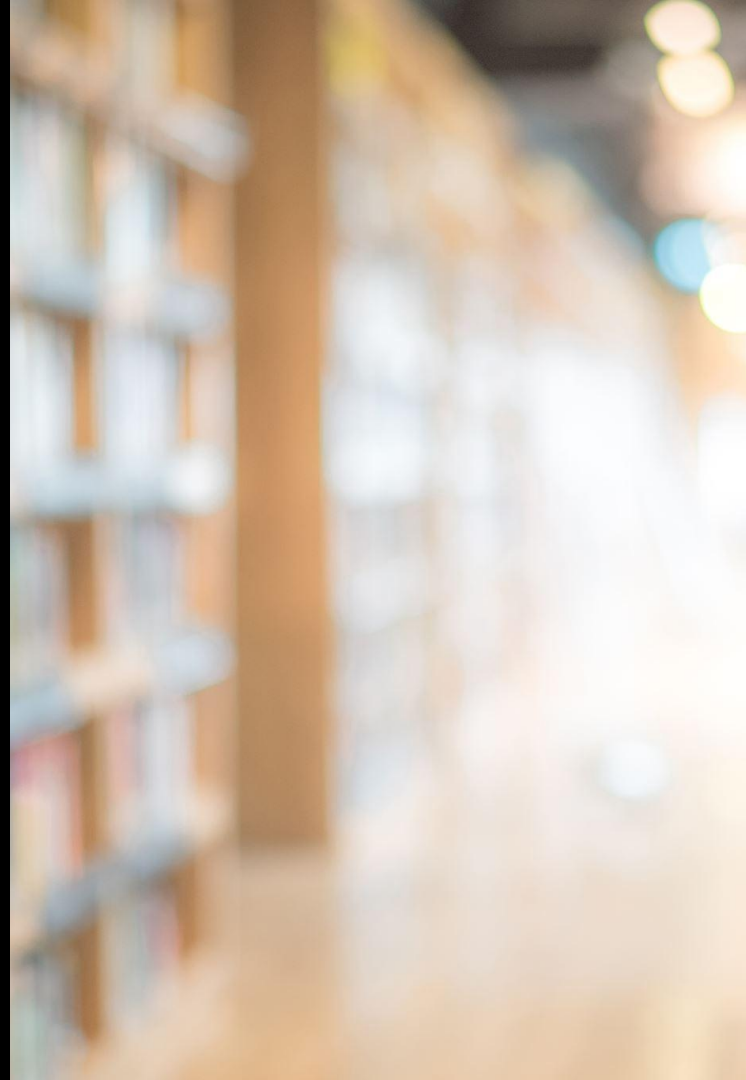
- [Shopify bounty](#)
- [Gitlab RCE](#)
- [Reddit Open Redirect](#)
- [HTTP Request Smuggling](#)
- [Exploit Development](#)





EXERCISE 6

- Take one research topic and develop a small web application vulnerable related to the research topics



QA

damjan@thecyberprotect.com