Journal Book

Written by: H.R.T Peiris
IT21163340

Sri Lanka Institute of Information Technology

JOURNAL BOOK

IE2062 - Web Security

Peiris H.R.T
IT21163340
Y2 S2 CS Weekend Group

Table of Contents

01. Introduction 5

02. 2023/04/17 - 2023/04/23 Week Progress 6

- i.What is a bug bounty program? 7
- ii.OWASP Top 10 Security Risks and Vulnerabilities 8
- iii.Bug bounty websites 15
- iv.Bug Bounty methodologies 16
- v.Bug Bounty tools 18

03. 2023/04/24 - 2023/04/30 Week Progress 21

- CSP (Content Security Policy) 21
- XSS (Cross-Site Scripting) 22
- CSRF (Cross-Site Request Forgery) 23
- SQLi (SQL injection) 24

04. 2023/05/01 - 2023/05/07 Week Progress 25

- Learning Knockpy Tool 25
- Learning Uniscan Tool 25
- Learning Pwnxss Tool 26
- Learning Netsparker Tool 26
- Learning Nikto Tool 27

05. 2023/05/08 - 2023/05/14 Week Progress 28

- Selected Domains 28
- Challenges 29
- Found Vulnerabilities 30
  - i.Weak Ciphers Enabled (Medium) 30
  - ii.Out-of-date Version (Lodash) - Critical Vulnerability 30
  - iii.[Possible] Cross-site Request Forgery 31
  - iv.Missing X-XSS-Protection Header 32
  - v.[Possible] Phishing by Navigating Browser Tabs (Medium) 32
  - vi.Robots.txt Detected 33
  - vii.Internal Server Error 34
  - viii.[Possible] Internal IP Address Disclosure (Medium) 34
  - ix.Missing X-Frame-Options Header 35
  - x.Misconfigured Access-Control-Allow-Origin Header 36

06. 2023/05/15 - 2023/05/21 Week Progress....37
Smart Contract....37
Blockchain....37
Try to do Smart Contract CTF....38
Challenges....39
SolidityScan....39
07. 2023/05/22 - 2023/05/27 Week Progress....40
08. Conclusion....41
References....42

01.Introduction

I am investigating the areas of bug bounty schemes and smart contracts as part of my research. Safety of internet infrastructure, including smart contracts, is increasingly dependent on bug bounties. Bug bounty programmes are extremely useful for ethical hackers because they give businesses a one-of-a-kind opportunity to discover and fix vulnerabilities in their systems and apps. My next bug bounty adventure journal entry will cover a wide range of subjects. First, I'll define bug bounties and describe the rigorous approach I used to discovering, exploiting, and reporting security vulnerabilities in exchange for monetary rewards. Incorporating this guiding principle into my work allowed me to take a logical and successful approach to designing and implementing my bug bounty initiatives.After plunging deeply into the murky realm of bug bounties, I feel confident in my understanding of the methods employed by bug bounty hunters.

I used several technological approaches for bug bounties. I was able to do a better job and uncover security problems more quickly with the aid of attack frameworks, reconnaissance and information gathering tools, vulnerability scanners, and other such technology. Methods used and their impact on my bug-hunting adventure are discussed. Nikto stands out as one of the most helpful of the well-known tools I've used to scan web servers for common vulnerabilities and misconfigurations. But an Open Web Application Security Project ZAP (Zed Attack Proxy) called Knockpy has been successful in finding vulnerabilities in web apps. The automated vulnerability scanner Netsparker utilises a hybrid approach, combining dynamic application security testing (DAST) and interactive application security testing (IAST), to identify security flaws in online applications and APIs.

I've spent a lot of time thinking about potential security holes in bug bounties and smart contracts as part of my research. I elaborate on the importance, exploitability, and implications of these flaws. I detail the impacted software and infrastructure and the methods I used to verify and reproduce the issues. The challenges I faced and the methods I used to overcome them during my bug-hunting quest are then detailed. When faced with an issue, I like to think of it as an opportunity to demonstrate my problem-solving skills by suggesting potential solutions, other methods, and insights. Finally, I summarise the most helpful things I learned from the bug bounty initiative. Here I reflect on my experiences in bug bounty programmes and share what I learned, the skills I picked up, and the doors it opened for me. Bug bounties may be rapidly found and reported if the discoverer is conversant with the vulnerabilities that smart contracts may have and uses the necessary tools.

02.2023/04/17 - 2023/04/23 Week Progress

As part of our study, we are now doing more research on the bug prize programme. We are looking at a wide range of different things. I just started with the bugbounty programme, but I didn't know how to get started or what to do at first. The bug prize programme has just begun. The bug prize programme is just getting off the ground. At this point, all that's left to do is get the bugbounty programme up and going. At this point, setting up the bugbounty programme is all that needs to be done to get it up and running. At this point, all that is needed to get the bugbounty programme up and going is to set it up. As a direct result of the above fact, I have been doing study on the strategy I need to use at the start of the project and the goals I need to reach in order to successfully finish such an amount of work. Because of this, I decided to do some study on YouTube to learn more about what a bug prize is and how to get one. I did this so I could help other people who want to get bug rewards better. By watching movies on YouTube feeds like PHDSECURITY and UNIXGuy, for example, I was able to learn a lot that turned out to be very useful. John Hammond was also a very reliable source of information. First, choose your business from the list of choices the bug reward programme has given you. Install Kali Linux to see if you can win the bug prize. Install the software that will gather information about internet apps.

i. What is a bug bounty program?

A bug bounty program is program provided by businesses or organizations to encourage security researchers and ethical hackers to identify and report security flaws in their systems, software, or apps. The initiative provides incentives or financial compensation to those who effectively locate and disclose these vulnerabilities. The incentives may be monetary or non-financial, such as praise or gifts.

The bug bounty program's primary goal is to enable ethical hackers to examine a web application that belongs to a third party and report bugs, which include vulnerabilities, flaws, and potential exploitations in the web application's hardware, software, and firmware. According to the ethical hackers' service, they are ready to offer prizes for such a service.

The standard structure of bug bounty programs includes specific instructions and regulations for participation, such as what kinds of vulnerabilities qualify for compensation, how to report them, and what the incentives will be. Before providing the prize, companies may also establish a mechanism for evaluating and confirming reported vulnerabilities to make sure they are real and significant.

These businesses and organizations permit external resources to scan their systems for vulnerabilities and report such vulnerabilities to the relevant business. They must patch their web application's vulnerabilities, start a bug bounty program with the assistance of ethical hackers throughout the world, and stop black hat and grey hat hackers from breaking into their systems and compromising them.
Nowadays, we can observe that a lot of bug bounty programs have expanded quickly from small businesses to huge businesses and governmental institutions.

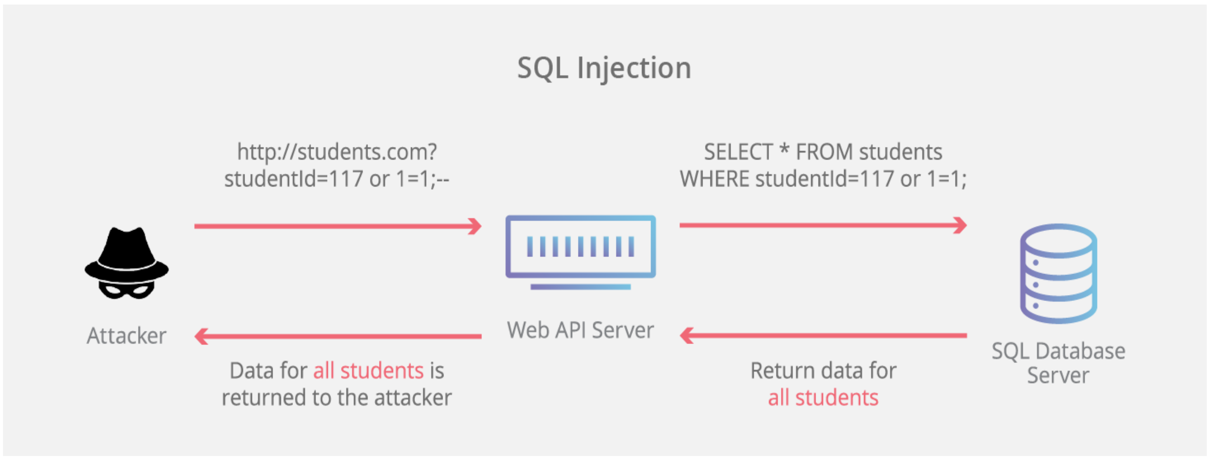
ii. <u>OWASP Top 10 Security Risks and Vulnerabilities</u>

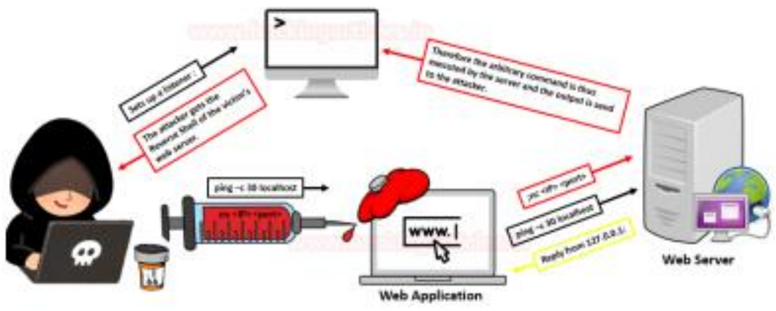
The OWASP Top 10 vulnerabilities is a list of the most common and potentially harmful security concerns that online applications must contend with. The Open Web Application Security Project (OWASP), a non-profit organization that offers knowledge and tools to assist enterprises in improving the web application security that they provide for their customers, is responsible for compiling and routinely updating this list. We are able to identify the top 10 most vulnerable websites in every region in the globe. Every three to four years, OWASP will revise and publish a new version of its list of the top 10 web application vulnerabilities. The full name of this organization is the Open Web Application Security Project (OWASP). The top 10 risks identified by OWASP are shown below, along with the potential consequences of each vulnerability and advice on how to circumvent them. The comprehensive list was compiled with the assistance of a wide variety of expert sources, such as security consultants, security suppliers, and security teams from enterprises of all types. It is commonly considered to be an essential resource for determining the most effective procedures for securing online applications.

OWASP Top 10 vulnerabilities are:

1. Injection:

   A vulnerability known as injection gives an attacker the ability to submit malicious input to a program, which may subsequently carry out instructions that were not intended or access sensitive data. An attacker has successfully injected their own code into a program when they have successfully exploited a vulnerability in the program's code. As a result of the fact that the software is unable to differentiate between its own code and code that has been introduced in this manner, attackers are able to utilize injection attacks to access protected regions and private information as if they are trusted users. SQL injections, command injections, CRLF injections, and LDAP injections are a few examples of different types of injections.


SQL Injection
http://students.com?studentId=117 or 1=1;--
SELECT * FROM students WHERE studentId=117 or 1=1;
Attacker
Web API Server
SQL Database Server
Data for all students is returned to the attacker
Return data for all students

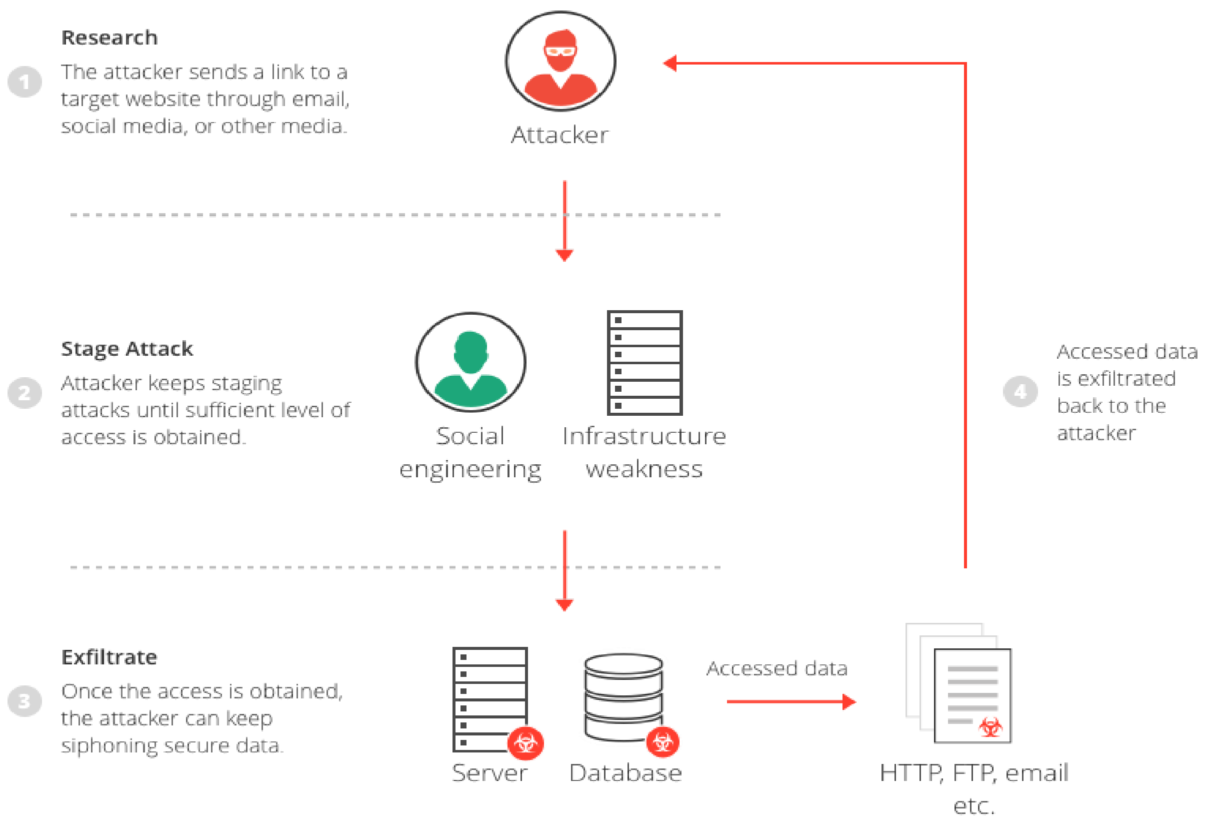


Sets up a listener
The attacker gets the Reverse Shell of the victim's web server.
Therefore the arbitrary command is thus executed by the server and the output is send to the attacker.
ping -c 10 localhost
nc <ip> <port>
ping -c 10 localhost
Reply from 127.0.0.1
www.
Web Application
Web Server

Figure 2:OS command injection

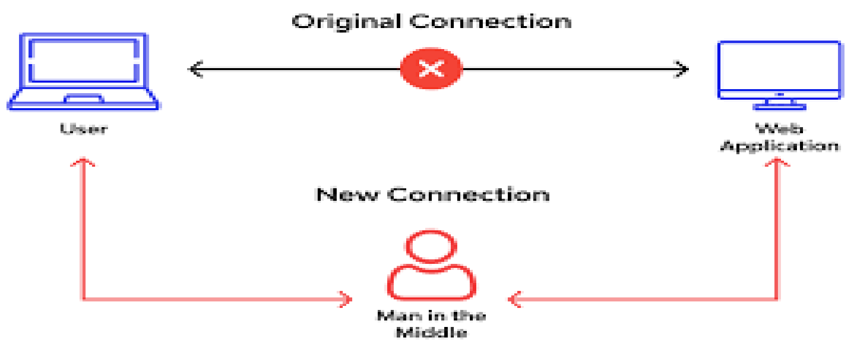
2. Broken Authentication:

   This refers to weaknesses in the user authentication and session management processes of online applications, which may result in illegal access or session hijacking. Authentication and session management calls that have been improperly implemented pose a significant threat to data security. In the event that malicious actors become aware of these vulnerabilities, they may be able to simply assume the identities of genuine users.


Research
1
The attacker sends a link to a target website through email, social media, or other media.
Attacker
Stage Attack
2
Attacker keeps staging attacks until sufficient level of access is obtained.
Social engineering
Infrastructure weakness
4
Accessed data is exfiltrated back to the attacker
Exfiltrate
3
Once the access is obtained, the attacker can keep siphoning secure data.
Server
Database
Accessed data
HTTP, FTP, email etc.

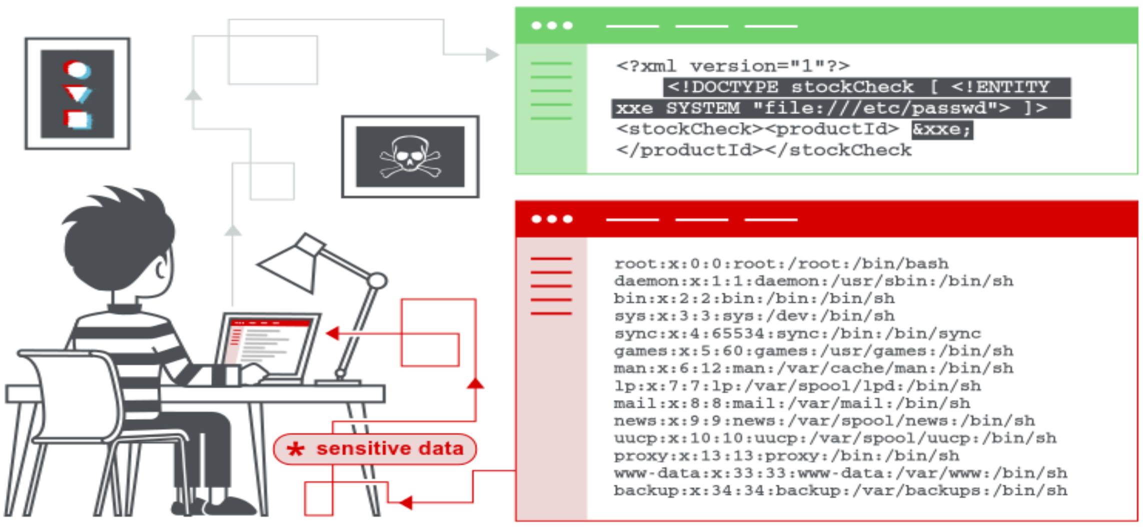
3. Sensitive Data Exposure:

When private information is shared with those who are not allowed to see it, a security risk known as sensitive data exposure has been created. This may occur if an application fails to safeguard or encrypt sensitive data as required, or if sensitive data is delivered across channels that are not encrypted or secure. Both of these scenarios are possible. APIs, or application programming interfaces, are a fantastic time saver since they enable developers to link their application to third-party services such as Google Maps. Unfortunately, certain APIs depend on insecure data transfer techniques, which cybercriminals may exploit to get access to usernames, passwords, and other sensitive information.



4. XML External Entities (XXE):

   Applications that use XML input might be vulnerable to a security flaw known as XML External Entities (XXE), which stands for XML External Entity. Due to this security flaw, an attacker may be able to access arbitrary files, carry out server-side request forgery (SSRF) attacks, or run remote code on the server. Because of insecure code, integrations, or dependencies, this risk develops when attackers have the ability to upload or add hostile XML material. An SCA scan may identify potential dangers in third-party components that have already been discovered to have vulnerabilities, and it can notify you about those dangers. A reduction in the chance of an XML entity attack is another benefit of turning off XML external entity processing.

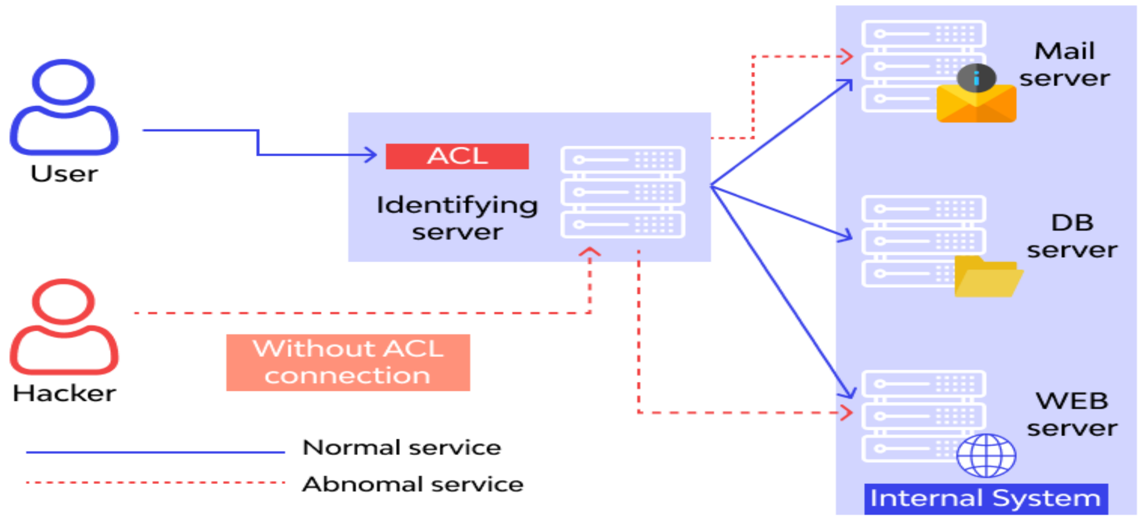


```
<?xml version="1"?>
    <!DOCTYPE stockCheck [ <!ENTITY
xxe SYSTEM "file:///etc/passwd"> ]>
<stockCheck><productId> &xxe;
</productId></stockCheck
```

```
root:x:0:0:root:/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
```

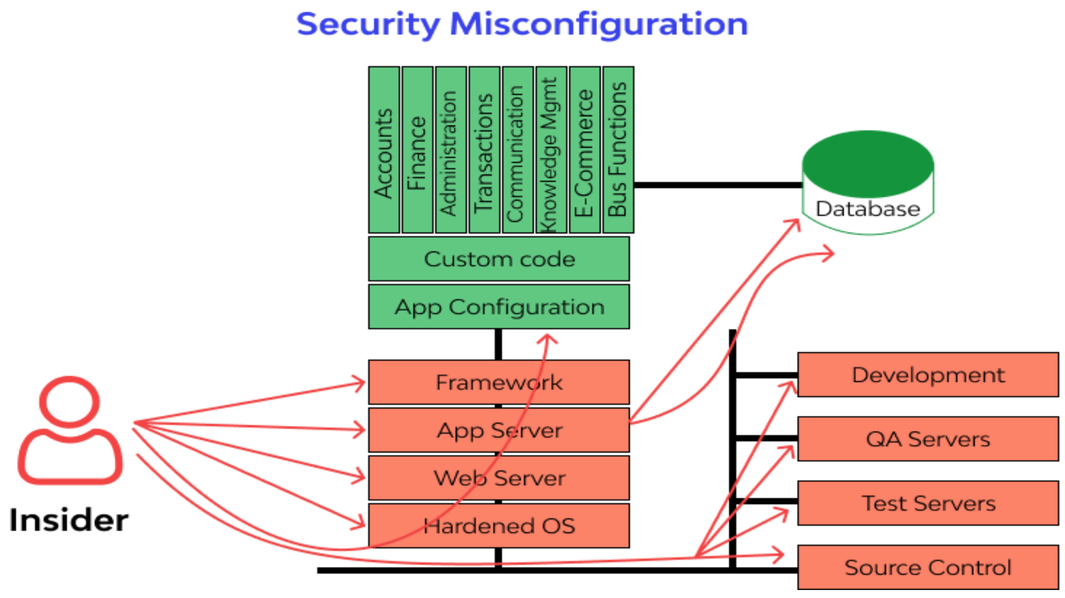
5. Broken Access Control:

   This refers to defects in the manner in which online applications enforce access rules, which make it possible for an attacker to access unapproved sites or functionality. Attackers are able to grab everything they want with relative ease if authentication and access control measures are not implemented correctly. It is possible for unauthenticated or unauthorized users to get access to sensitive data and systems, as well as user privilege settings, if there are weaknesses in the access control mechanism.



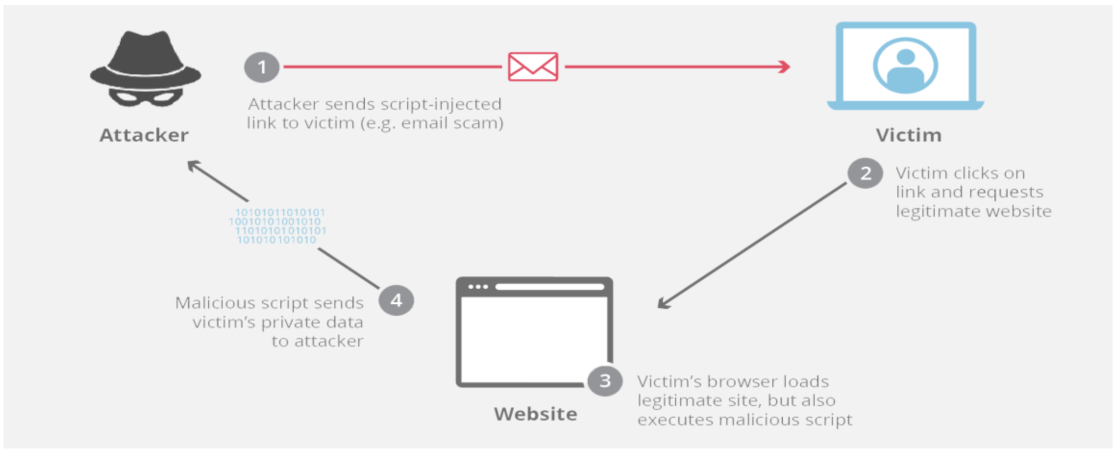
6. Security Misconfiguration:

   Misconfigurations in security relate to vulnerabilities that are brought about by components of a web application that have been poorly configured or protected. These components include servers, frameworks, and databases.



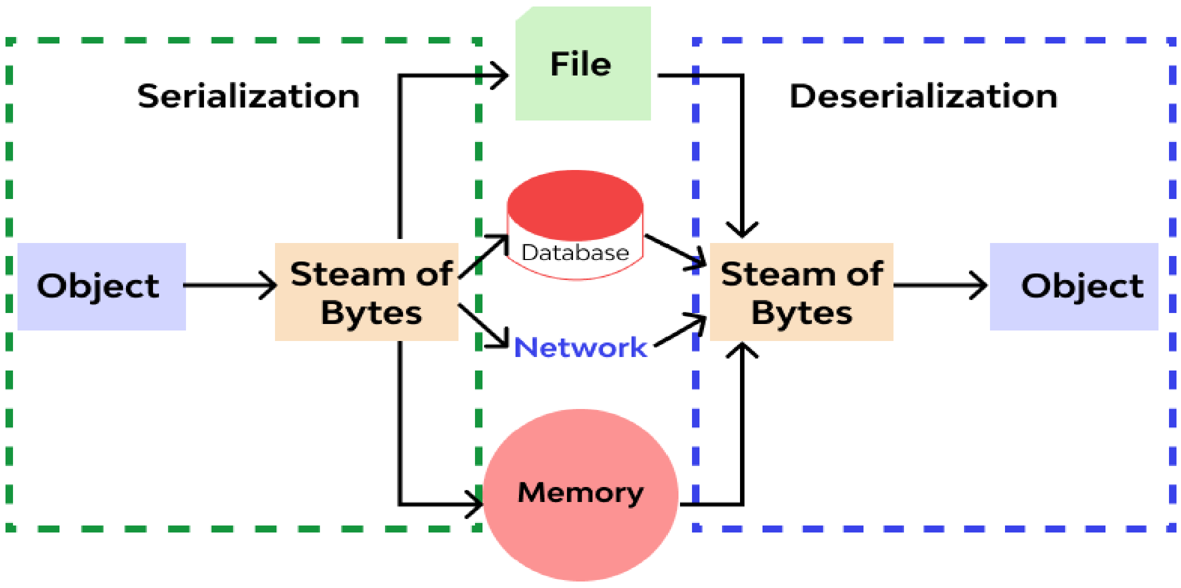
7. Cross-Site Scripting (XSS):

   Attackers may collect data from or send instructions to your application using a technique known as cross-site scripting. This technique takes use of APIs and DOM manipulation. Cross-site scripting expands the attack surface for threat actors, giving them the ability to remotely manipulate browsers, hijack user accounts, examine browser history, and transmit Trojans and viruses, and more.


Attacker
1
Attacker sends script-injected link to victim (e.g. email scam)
Victim
2
Victim clicks on link and requests legitimate website
3
Victim's browser loads legitimate site, but also executes malicious script
Website
4
Malicious script sends victim's private data to attacker

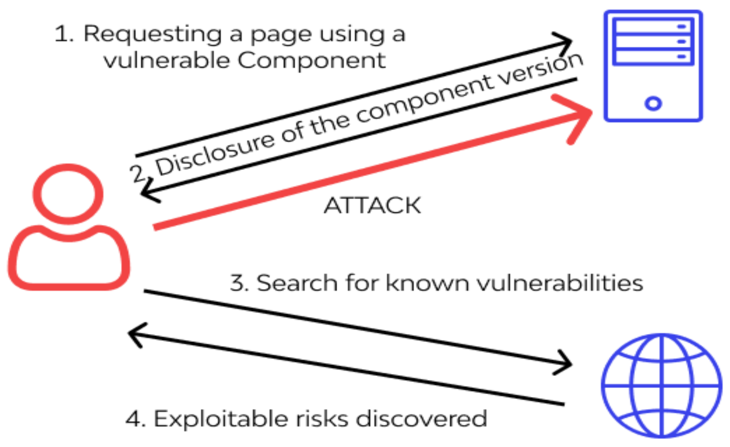
8. Insecure Deserialization:

   Deserialization is the process of turning serialized data (such as JSON or XML) back into its original form in memory. This process is known as "deserialization." Deserialization, which is the process of recovering data and objects that have been written to discs or otherwise preserved, may be used by attackers to remotely execute code in your application or to open a door to other assaults. Common serialization systems, such as JSON and XML, transform an object into a text format that is either structured or binary. This format is known as the serialization format. This vulnerability manifests itself when an adversary makes use of data that is not to be trusted in order to manipulate an application, launch a denial of service (DoS) attack, or run unanticipated code in order to alter the behavior of the programme.



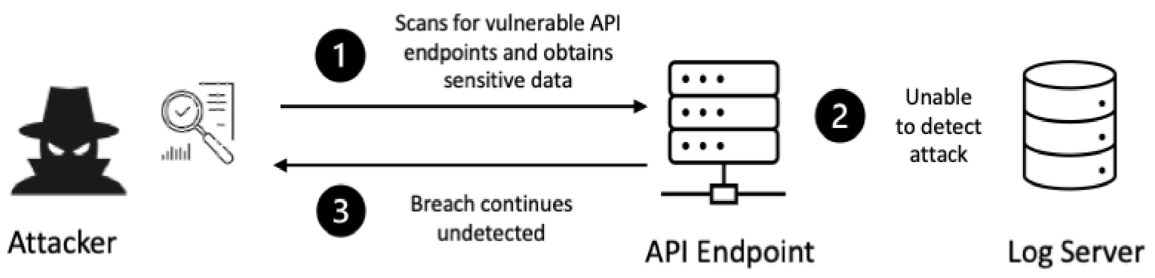
9. Using Components with Known Vulnerabilities:

   This is a reference to the vulnerabilities that may occur as a result of online applications making use of third-party components, libraries, or frameworks that are known to have security flaws. Attackers are able to take advantage of APIs, dependencies, and other third-party components if they are not themselves secure. This is true regardless of how safe your own code may be.


1. Requesting a page using a vulnerable Component
2. Disclosure of the component version
ATTACK
3. Search for known vulnerabilities
4. Exploitable risks discovered

10. Insufficient Logging and Monitoring:

    An application has a security vulnerability known as insufficient logging and monitoring if it does not record and monitor significant security events in an acceptable manner. This vulnerability may develop when an application fails to sufficiently log and monitor essential security events. Because of this, it may be difficult or even impossible for security teams to notice security breaches or other harmful activity in a timely way and to react appropriately to them.

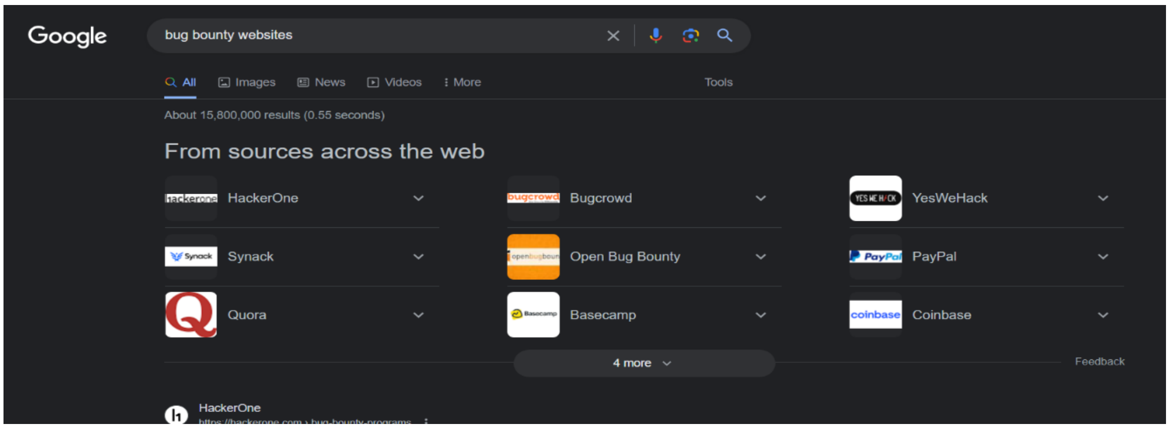

Scans for vulnerable API endpoints and obtains sensitive data
1
2
Unable to detect attack
3
Breach continues undetected
Attacker
API Endpoint
Log Server

iii. Bug bounty websites

An online platform known as a bug bounty website links businesses or organizations with a community of ethical hackers and security researchers who may find and disclose security flaws in their systems, software, or applications. These platforms provide businesses with a single area to host their bug bounty programs, define the parameters of the engagement, and compensate security researchers and ethical hackers who successfully find and disclose vulnerabilities. Before providing the prize, the platform also offers methods and tools for triaging and confirming the reported vulnerabilities. Companies may access a global network of security specialists by using bug bounty websites to strengthen their security posture and defend against online attacks.

These are,

- HackerOne
- Bugcrowd
- Synack
- Intigriti
- Coblat
- Penester Lab
- GitHub WebGoat
- OWASP Juice Shop
- PortSwigger
- Hacker101
- Bugcrowd University



iv. <u>Bug Bounty methodologies</u>

Bug Bounty Methodologies for finding vulnerabilities and reporting them to bug bounty programs are referred to as bug bounty methodology. These methodologies include the many approaches and techniques utilized by security researchers and testers. Testers may find vulnerabilities in software applications, computer systems, and network infrastructure more quickly and accurately with the use of these approaches.

Some common bug bounty methodologies:

1. Reconnaissance: To do this, information must be gathered about the system or application that is the target, such as its architecture, technological stack, and possible attack surfaces. This information may be put to use to determine possible points of vulnerability and entry points for attacks.

2. Scanning: This entails employing automated tools to perform a vulnerability scan on the system or application that is the focus of the scan. Scanners for online applications, networks, and vulnerabilities are examples of the types of tools that fall under this category.

3. Fuzzing: This method includes doing tests on the system or application by providing massive volumes of data that is either unexpected or faulty in order to locate vulnerabilities. Tools such as AFL, Peach Fuzzer, and Spike Proxy are examples of this category.

4. Exploitation: Taking advantage of a vulnerability in order to acquire unauthorized access or control of the system or application that is the target of the attack is required. Tools such as Metasploit, ExploitDB, and SQLMap are examples of this category.

5. Social engineering: Using human contact in order to obtain access to sensitive information or to exploit vulnerabilities is required for this method. Phishing, pretexting, and baiting are all examples of methods that fall under this category.

v. **Bug Bounty tools**

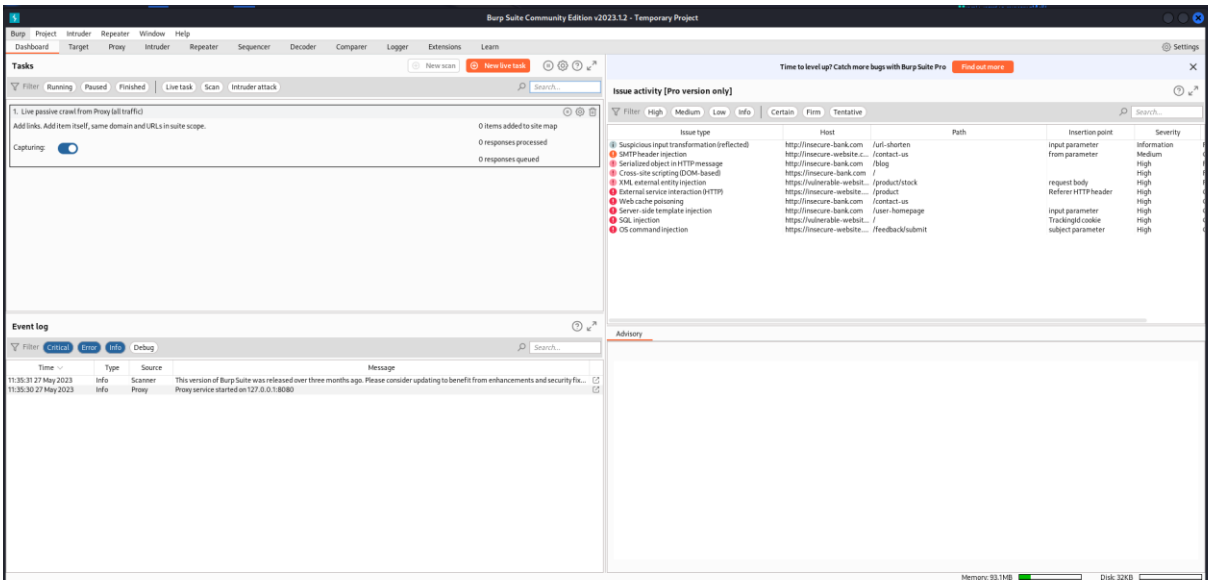
A Bug Bounty tools are software resource and tools used by security researchers and ethical hackers to find and exploit flaws in systems, programs, or software. A number of open-source and paid software tools are often included in these toolkits, and they may be used for activities including network scanning, web application testing, vulnerability analysis, and exploit building.

In order to meet the unique requirements and preferences of an ethical hacker or security researcher, Bug Hunter Toolkits may be adapted and customized. They are a crucial tool for individuals working in the cyber security industry since they are continually changing and upgrading as new vulnerabilities are found and new technologies are developed.

A Bug Hunter Toolkit could include, as some examples of software tools:

- Nmap
  Nmap, which stands for "Network Mapper," is a strong tool for scanning networks. It can be used to find hosts, scan ports, and list services. It helps security experts find open ports, services running on those ports, and computers that might be vulnerable.

- Burp Suite
  Burp Suite is a complete tool for testing web applications. Bug prize programmes use it a lot. It can do things like monitor HTTP requests and change them, look for security holes, and map programme structures. Burp Suite has a lot of different tools, like Proxy, Scanner, Repeater, Intruder, and more.


Burp Suite Community Edition v2023.1.2 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extensions Learn
Settings
Tasks
New scan
New live task
Filter Running Paused Finished Live task Scan Intruder attack
Time to level up? Catch more bugs with Burp Suite Pro
Find out more
Filter High Medium Low Info Certain Firm Tentative
Event log
Filter Critical Error Info Debug
Advisory
Memory: 93.1MB
Disk: 32KB

1. Live passive crawl from Proxy (all traffic)

Add links. Add item itself, same domain and URLs in suite scope.

Capturing:

0 items added to site map
0 responses processed
0 responses queued

Issue activity [Pro version only]

Issue type	Host	Path	Insertion point	Severity
Suspicious input transformation (reflected)	http://insecure-bank.com	/url-shorten	input parameter	Information
SMTP header injection	http://insecure-website.c...	/contact-us	from parameter	Medium
Serialized object in HTTP message	http://insecure-bank.com	/blog		High
Cross-site scripting (DOM-based)	https://insecure-bank.com	/		High
XML external entity injection	https://vulnerable-websit...	/product/stock	request body	High
External service interaction (HTTP)	https://insecure-website....	/product	Referer HTTP header	High
Web cache poisoning	http://insecure-bank.com	/contact-us		High
Server-side template injection	http://insecure-bank.com	/user-homepage	input parameter	High
SQL injection	https://vulnerable-websit...	/	TrackingId cookie	High
OS command injection	https://insecure-website....	/feedback/submit	subject parameter	High

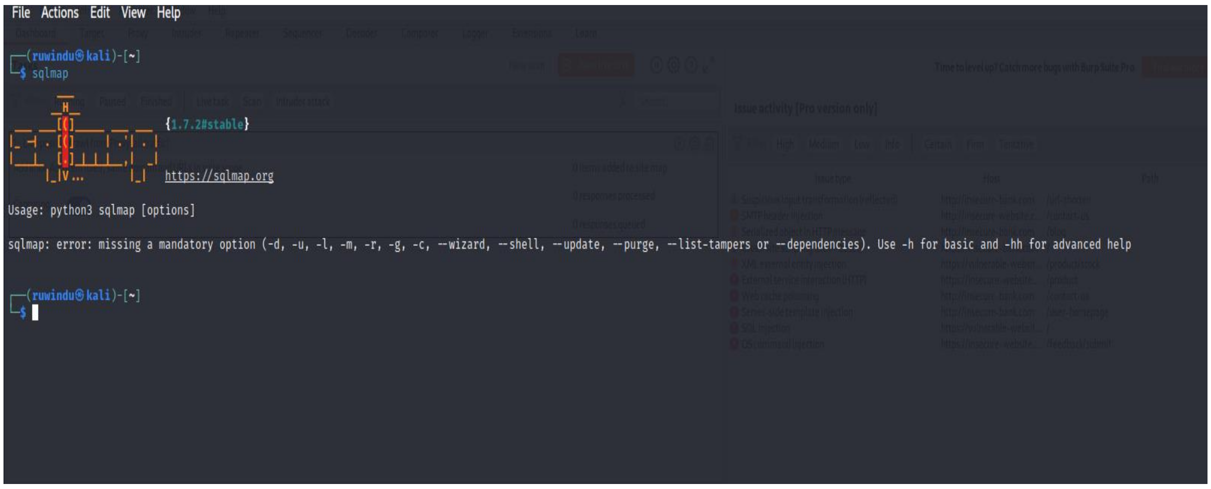
Time	Type	Source	Message
11:35:31 27 May 2023	Info	Scanner	This version of Burp Suite was released over three months ago. Please consider updating to benefit from enhancements and security fix...
11:35:30 27 May 2023	Info	Proxy	Proxy service started on 127.0.0.1:8080

- ➢ Metasploit

Metasploit is a tool for penetration testing that lets experts find, verify, and take advantage of weaknesses in different systems and apps. It comes with a wide range of modules, attacks, payloads, and post-exploitation features that help testers find security flaws and show how they affect the system.Wireshark - A network protocol analyzer.

- ➢ Sqlmap

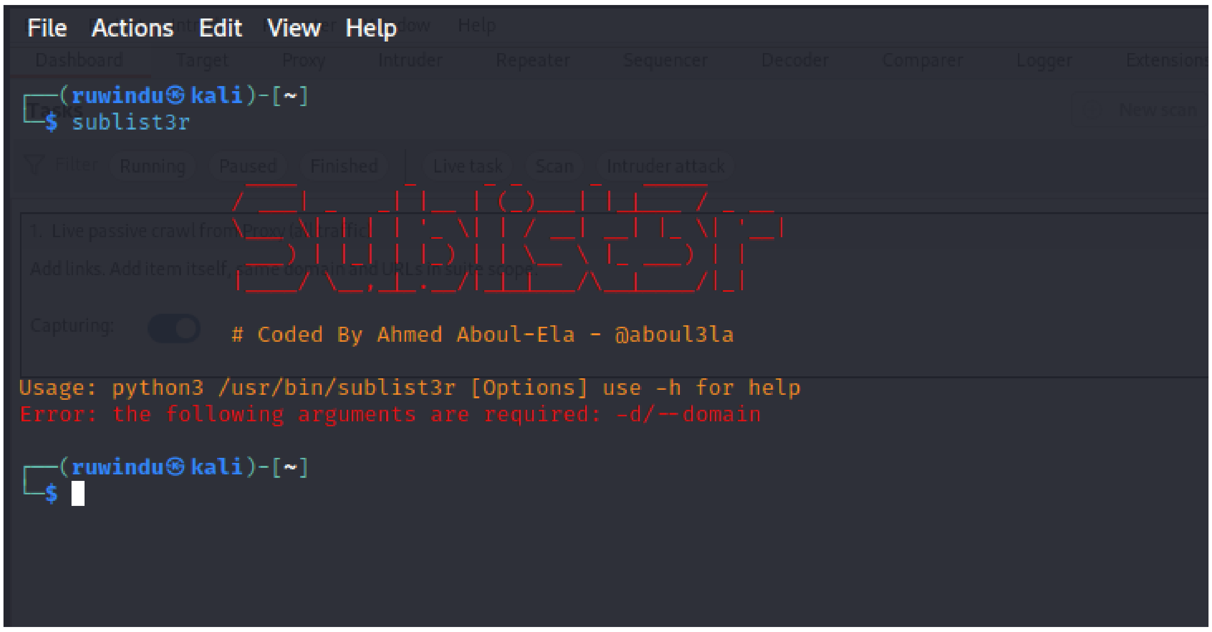
sqlmap is a popular open-source tool that helps find SQL attack flaws and use them automatically. It lets users find database backends, get data, and even execute commands on the target server.Netsparker - A vulnerability scanner, and it is an automated and highly configurable tool.



```
┌──(ruwindu㉿kali)-[~]
└─$ sqlmap
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.7.2#stable}
|_ -| . [(]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

Usage: python3 sqlmap [options]

sqlmap: error: missing a mandatory option (-d, -u, -l, -m, -r, -g, -c, --wizard, --shell, --update, --purge, --list-tampers or --dependencies). Use -h for basic and -hh for advanced help

┌──(ruwindu㉿kali)-[~]
└─$ ▌
```

- Sublist3r

Sublist3r finds website subdomains. It's Python. "Brute force" and public data do this. Google, Yahoo, Bing, Baidu, Ask, Netcraft, Virustotal, ThreatCrowd, DNSdumpster, and ReverseDNS can find subdomains. Subbrute, a built-in tool, can test all domain subdomains. Subbrute uses a huge list of terms to find DNS records and subdomains. Open resolvers help Subbrute finish the list and test all items.



```
┌──(ruwindu㉿kali)-[~]
└─$ sublist3r

                 ____        _     _ _     _   _____
                / ___| _   _| |__ | (_)___| |_|___ / _ __
                \___ \| | | | '_ \| | / __| __| |_ \| '__|
                 ___) | |_| | |_) | | \__ \ |_ ___) | |
                |____/ \__,_|_.__/|_|_|___/\__|____/|_|

                # Coded By Ahmed Aboul-Ela - @aboul3la

Usage: python3 /usr/bin/sublist3r [Options] use -h for help
Error: the following arguments are required: -d/--domain

┌──(ruwindu㉿kali)-[~]
└─$ 
```

03.2023/04/24 - 2023/04/30 Week Progress

CSP (Content Security Policy)

This day was devoted to learning about content security policy (CSP) and how to keep my web apps safe. By establishing and enforcing content restrictions on web pages, CSP protects them from XSS and data injection attacks. I learned that CSP communicates with the browser using an HTTP response header to control what resources may be loaded and executed. Scripts, stylesheets, images, fonts, and more may all have their origins specified in the CSP header.

XSS attacks may be avoided with the help of Content Security Policy. By restricting script origins to the same origin or whitelisted domains, CSP helps prevent the injection of malicious scripts by attackers. This protects user information against XSS attacks, both reflected and stored. Data injection and clickjacking are both avoided by CSP. By implementing content source restrictions, including those for frames and iframes, CSP may protect web applications against exposure to harmful or untrusted content from external sources. CSP additionally provides customizable security policy settings and strictly regulates content sources. The flexibility of these directives allows programmers to strike a good balance between security and usability in their applications. "script-src', "style-src," "img-src," "font-src," "connect-src," and "default-src" are all examples of directives. In addition, I learned about the "report-uri' command in CSP. With this directive, programmers may provide a URL to which browsers should notify violations of a certain policy. Information on attempted breaches and other suspicious behaviour is useful for finding and fixing security flaws. It is important to thoroughly evaluate and test how CSP may influence the functioning of online applications. If set up and tested correctly, CSP has the potential to significantly enhance the security of web-based applications by reducing instances of content injection and malicious code execution.

In conclusion, improving the safety of online applications requires familiarity with and adoption of Content Security Policy. Developers may reduce exposure to XSS attacks, data injection, and other web-based vulnerabilities by adopting a well-thought-out strategy. I'm eager to learn more about real-world applications of CSP and the best practises for incorporating it into my future projects to make the web a safer place for everyone.

XSS (Cross-Site Scripting)

In this lesson, I gained a deeper understanding of Cross-Site Scripting (XSS), a critical vulnerability in many web applications with potentially devastating consequences for both users and administrators. As a web developer, you need to be aware of XSS and secure your code accordingly.

Cross-site scripting is a security risk when an attacker is able to inject malicious scripts into a website or web application. An attacker may, for instance, inject scripts into user input that would be displayed and executed by other users' browsers if the input isn't fully checked or sanitised.

In my research, I came across three main types of XSS attacks: stored XSS, reflected XSS, and DOM-based XSS. The attacker injects malicious scripts that are stored on the server and pose a danger to everyone who accesses the compromised content in the future. When a user enters text and immediately sees that text repeated in the response, a kind of XSS known as "reflected XSS" has occurred, which might trick unwary users into running the malicious scripts that were injected. DOM-based XSS occurs when malicious JavaScript on the client side alters the DOM in a way that allows for script execution.

After conducting some study, I realised that XSS attacks provide several security risks, including the disclosure of private user information, the theft of a user's session, the defacement of web pages, and the complete control of a website. To lessen the possibility of XSS vulnerabilities and the potential damage they may do, developers should take protective measures.

In conclusion, XSS attacks pose a significant risk to online services and their users. As a web developer, you need to know about the many kinds of XSS attacks and how to defend against them. By using input validation, output encoding, and security features like Content Security Policy, the risk of XSS vulnerabilities may be mitigated and the security and integrity of web-based applications protected. My intention is to prevent XSS and create more secure websites by including these practises into my code.

CSRF (Cross-Site Request Forgery)

I only learned about the threat of Cross-Site Request Forgery (CSRF) today. In order to steal sensitive information or perform harmful actions, hackers might utilise CSRF to pose as genuine users. Web developers need to understand CSRF to keep their code secure and reliable.

An attacker uses Cross-Site Request Forgery when they take control of a victim's browser and use it to launch attacks on a specific website. This kind of attack takes advantage of a user's session's inherent trust by targeting vulnerabilities in the security measures put in place to verify the authenticity of requests.

My findings are consistent with those of other CSRF attacks. In many cases, cybercriminals will trick their victims into visiting a rogue website or opening an email containing a request for a service that needs authentication. To the targeted website, the victim's browser will forwards the authenticated user's request if they view the malicious website or click the malicious link in the infected email.

If a CSRF attack succeeds, it might have catastrophic results. If a hacker gains access to a user's account, they might potentially change the password, conduct unauthorised transactions, delete data, or meddle with other settings. The security of the entire web app and the affected users is compromised as a result.

Finally, Cross-Site Request Forgery is a severe security risk that might lead to malicious acts on the part of website users. As a web developer, you should be aware of and prepared for common online security concerns including cross-site request forgery (CSRF) tokens, the same-origin policy, and incorrect access limitations. Following best practises and doing regular security testing helps protect both the application and its users from CSRF attacks. My goal is to dramatically reduce CSRF vulnerabilities by incorporating these security practises into my process.

SQLi (SQL injection)

Today, I expanded my knowledge about SQL Injection (SQLi), a frequent and severe vulnerability that can compromise the security of database-connected web applications. Developers of websites need a solid understanding of SQLi to secure their projects against hackers, data breaches, and other sorts of abuse.

An attacker can take advantage of a web application's SQL Injection vulnerability by inserting bogus SQL statements into the application's input fields or query parameters. An attacker can execute arbitrary SQL queries on the underlying database by evading validation and sanitization with these altered statements.

As I researched, I learned about SQL Injection attacks and their possible consequences. Unauthorised users may get access to sensitive information, alter existing data, or even issue commands to the server. The fallout might be trivial, such as a data leak, or catastrophic, such as the entire compromise of the programme and its infrastructure.

In conclusion, SQL Injection is a serious security risk that may open the door to hacking, information disclosure, and other forms of criminal activity. Any competent web developer should be familiar with and employ preventative measures including input validation, parameterized queries, and the principle of least privilege. Adopting these practises and doing regular security audits would considerably reduce the likelihood of SQL Injection vulnerabilities and ensure the safety of our web-based applications. I intend to include these safety measures into my development process to successfully avoid SQLi.

04. 2023/05/01 - 2023/05/07 Week Progress

Learning Knockpy Tool

Today I had the opportunity to experiment with Knockpy, a Python-based tool for subdomain reconnaissance. When performing security audits or penetration testing, Knockpy's various features come in handy since they allow you to enumerate and locate subdomains.

I discovered that Knockpy use many strategies for locating subdomains that are similar to a particular domain. It gathers information using domain name system (DNS) lookups, brute-force assaults, and web search engine scraping. The purpose of Knockpy is to bring these methods together to provide a comprehensive set of related subdomains.

Finally, Knockpy is a useful tool for subdomain reconnaissance since it aids in the discovery and enumeration of subdomains associated with a target domain. As it uses a wide variety of techniques to find subdomains, including direct brute-force guessing, DNS queries, and search engine scraping, it is a versatile tool. It is critical, however, that this tool be utilised morally and legally. I'm looking up to exploring Knockpy's documentation and putting its subdomain discovery features to use in my security assessments.

Learning Uniscan Tool

Recently, I got the chance to investigate Uniscan, a well-liked open-source tool for scanning websites for security flaws. Uniscan provides a wide variety of tools that help improve the safety of websites and online apps.

My investigation led me to learn that Uniscan is a tool for automating the process of scanning online applications for common vulnerabilities like SQL Injection, Cross-Site Scripting (XSS), File Inclusion, and Remote File Execution. It runs a battery of tests on the target website, checking for things like correct input, proper handling of parameters, and proper validation of responses.

Finally, Uniscan has shown to be an effective automated vulnerability scanning solution for websites. Its characteristics and functions allow for a more thorough evaluation of security by programmers and experts in the field. However, manual testing and code reviews are essential to complement its use for complete security. I'm excited to learn more about Uniscan and start using it as part of my security testing routine so that I can make my online apps more secure.

Learning Pwnxss Tool

Today, I was able to make use of PwnXSS, a sophisticated framework for testing and exploiting Cross-Site Scripting (XSS) vulnerabilities. PwnXSS's robust set of features and capabilities makes it useful for security experts who want to test the effectiveness of XSS defences and assess the robustness of web-based programmes.

PwnXSS's many helpful features have greatly simplified my XSS testing and exploitation efforts. Multiple forms of XSS payloads are possible, including persistent XSS, reflected XSS, and DOM-based XSS. These payloads may be customised and altered to target a broad range of vulnerabilities and attack methods. PwnXSS is amazing since it can perform both manual and automatic XSS testing.

In conclusion, PwnXSS is an effective framework for investigating and taking advantage of XSS vulnerabilities on websites. In-depth XSS audits may be performed with its help because of features like manual and automated testing, payload building, and countermeasure bypass techniques. Nonetheless, it is of the utmost importance that this framework be utilised correctly and in line with all relevant laws and ethical norms. I'm looking up to learning more about PwnXSS and incorporating its capabilities into my XSS testing tactics to strengthen my penetration tests.

Learning Netsparker Tool

Today I had the opportunity to look at the web application security scanner known as Netsparker, and in this blog I'll talk about my findings. In-depth security assessments and fortifying online infrastructures are both possible with the help of Netsparker's many features.

My research indicates that Netsparker's primary use is to speed up the time it takes to scan and test web-based applications for vulnerabilities. Cross-site scripting (XSS), SQL injection, and remote file inclusion are just some of the security flaws that can be uncovered by the tool's dynamic and static analyses.

Finally, Netsparker is a powerful web application security scanner that may potentially identify security holes in a website. Some of the ways it helps with security audits include through sophisticated crawling, attack simulation, and detailed reporting. However, in order to complete a thorough evaluation, automated scanning technologies like Netsparker must be supplemented with manual analysis and other testing techniques. As such, I'm interested in learning more about Netsparker so that I may start using it in my own security testing.

Learning Nikto Tool

Nikto is a popular web vulnerability scanner that helps detect security flaws in web servers and apps; I had the chance to learn more about it today. Nikto's many useful features and tools make it easy to conduct thorough security audits and make sure websites are safe for users.

Through my investigation, I learned that Nikto is adept at identifying the most frequent types of vulnerabilities and misconfigurations that leave a web server or application vulnerable to attack. It runs a battery of tests on the target, checking for things like known vulnerabilities, obsolete software and hardware, and server setup.

Finally, Nikto has shown to be an invaluable tool for web vulnerability scanning, revealing hidden vulnerabilities in web servers and apps. The huge vulnerability database, many scanning modes, and in-depth reporting all work together to make this tool a valuable part of any thorough security audit. For a complete assessment, however, it is essential that this instrument be used properly and in tandem with manual analysis. I'm excited to learn more about Nikto and start using it in my security testing practises to make the web a safer place for everyone.

05.2023/05/08 - 2023/05/14 Week Progress

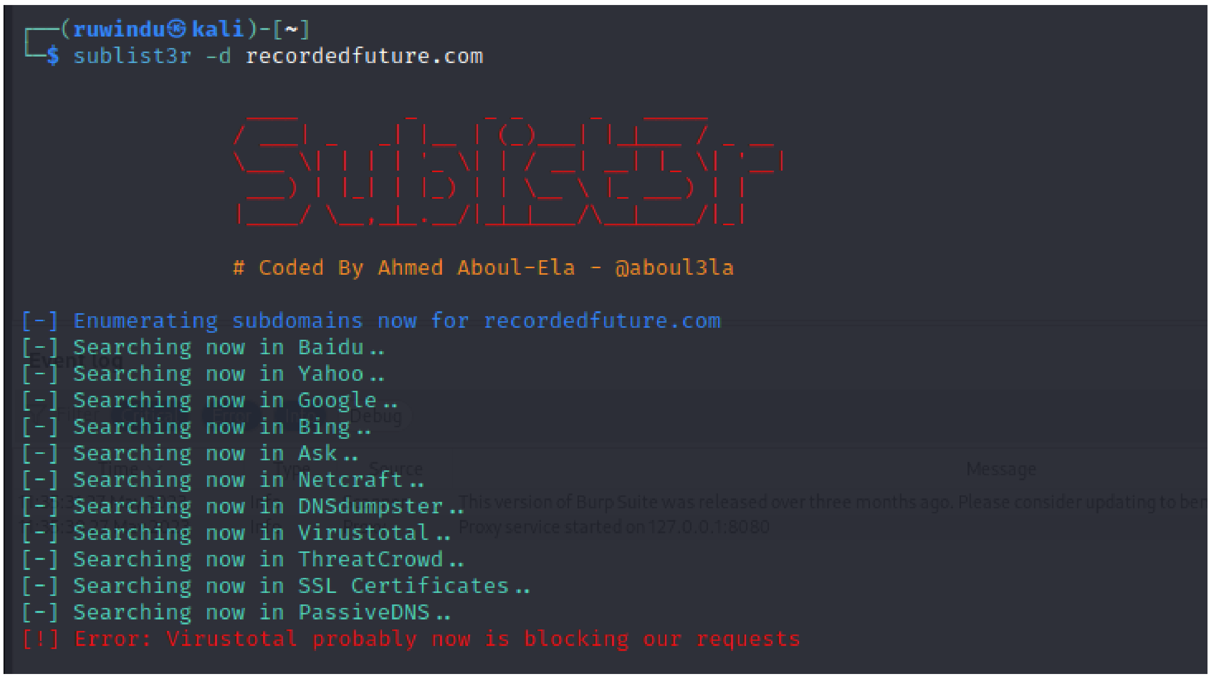
Gathering intelligence should make use of the instruments that have been supplied for use in reconnaissance. These tools should be used. After you have finished the process of classifying the significant information that is included within this material, the next step is to get rid of any information that isn't necessary any more. The subsequent step was to investigate the particulars of each website after hackerone had assisted in the selection of 10 unique domains as the starting point for the investigation. This step was carried out after the one that came before it had been finished successfully. I utilized the Uniscan and nikto applications that were available for usage on Kali Linux during the second week of the challenge in order to collect information on a single domain. This was done in order to complete the challenge. We employed a variety of programs, such as owasp-zap, knockpy, uniscan, and sublist3, amongst others, in order to analyze the probability of security issues in the system that was intended to protect consumers. This system was created to protect users from potential threats. Using the program Netsparker, I was able to identify a significant number of vulnerabilities and openings in the security of the system. I was able to do this since the system had been compromised. It's possible that malevolent actors took advantage of these weaknesses and gaps in the system.

Selected Domains

http://recordedfuture.com
api.recordedfuture.com
therecord.media
app.recordedfuture.com
https://www.getharvest.com
http://harvestapp.com
https://id.getharvest.com
https://www.semrush.com/
https://www.wickr.com/
https://www.yuga.com/

Challenges

Because of a problem with the sublist3r tool, we were forced to make use of a separate program known as knockpy in order to find the subdomains that are related to the domains. This was done in order to determine which subdomains are connected to which domains.



```
┌──(ruwindu㉿kali)-[~]
└─$ sublist3r -d recordedfuture.com

                 ____        _     _ _     _   _____
                / ___| _   _| |__ | (_)___| |_|___ / _ __
                \___ \| | | | '_ \| | / __| __| |_ \| '__|
                 ___) | |_| | |_) | | \__ \ |_ ___) | |
                |____/ \__,_|_.__/|_|_|___/\__|____/|_|

                # Coded By Ahmed Aboul-Ela - @aboul3la

[-] Enumerating subdomains now for recordedfuture.com
[-] Searching now in Baidu..
[-] Searching now in Yahoo..
[-] Searching now in Google..
[-] Searching now in Bing..
[-] Searching now in Ask..
[-] Searching now in Netcraft..
[-] Searching now in DNSdumpster..
[-] Searching now in Virustotal..
[-] Searching now in ThreatCrowd..
[-] Searching now in SSL Certificates..
[-] Searching now in PassiveDNS..
[!] Error: Virustotal probably now is blocking our requests
```

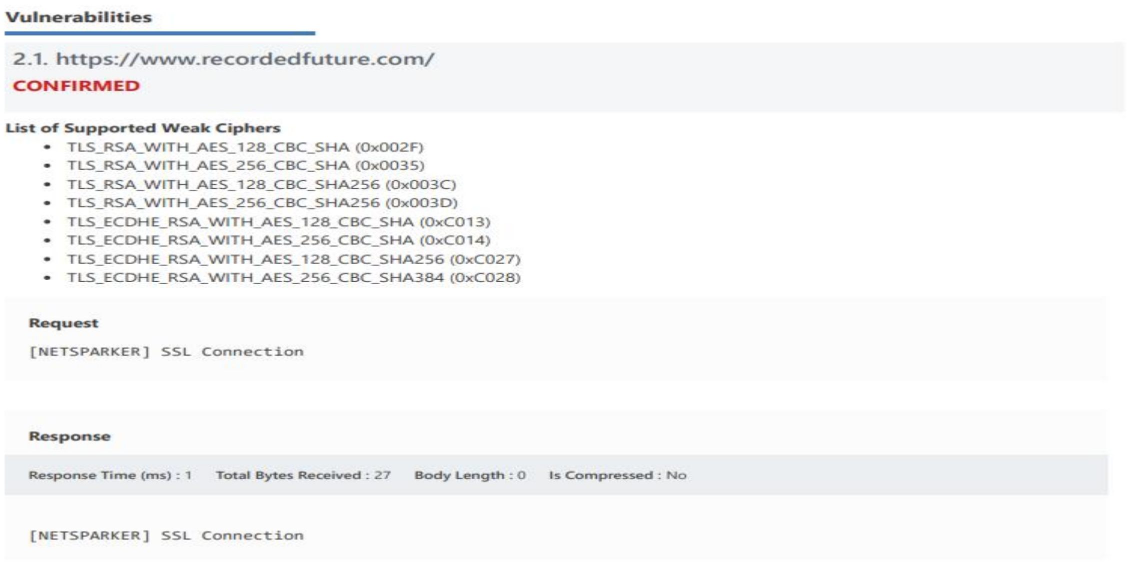
After performing a thorough vulnerability scan on each of the remaining domains, I felt confident in saying that they were all secure. I spent a lot of time and effort on this process. After analysing these websites extensively with the sophisticated Netsparker programme, I discovered a broad range of security holes that the site's administrators should fix immediately. These vulnerabilities might be used by adversaries. Inadequate encryption, cross-site scripting (XSS), and many other types of security weaknesses fall under this umbrella term.I meticulously created 10 in-depth reports based on the 10 domains and the specific vulnerabilities that each domain offered to guarantee that there was appropriate documentation and that fixes were carried out efficiently. These reports were put up using the information revealed by the many exposed vulnerabilities in each domain. Each report contains an in-depth examination of the vulnerabilities discovered, including a discussion of the vulnerabilities' potential consequences, the impacted parts of the system, and suggested solutions. The weak points of the system that were identified in each report are also detailed. I also paid close attention to the severity and urgency associated with each vulnerability.

Found Vulnerabilities

i. Weak Ciphers Enabled (Medium)

During the normal security audit I ran upon an alarming discovery today: a web application or server was using a particularly vulnerable cypher. For the purposes of this article, "weak cyphers" will refer to any and all encryption algorithms and/or protocols that are known to contain flaws or are otherwise not up to modern cryptographic standards.

Turning on insecure cyphers puts at risk the privacy and authenticity of all data sent over the network. Data breaches and unauthorised access may result if attackers are able to decode passwords, financial data, or personal details using these insecure cyphers.



Vulnerabilities

2.1. https://www.recordedfuture.com/
CONFIRMED

List of Supported Weak Ciphers
- TLS_RSA_WITH_AES_128_CBC_SHA (0x002F)
- TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
- TLS_RSA_WITH_AES_128_CBC_SHA256 (0x003C)
- TLS_RSA_WITH_AES_256_CBC_SHA256 (0x003D)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xC013)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xC014)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xC027)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xC028)

Request
```
[NETSPARKER] SSL Connection
```

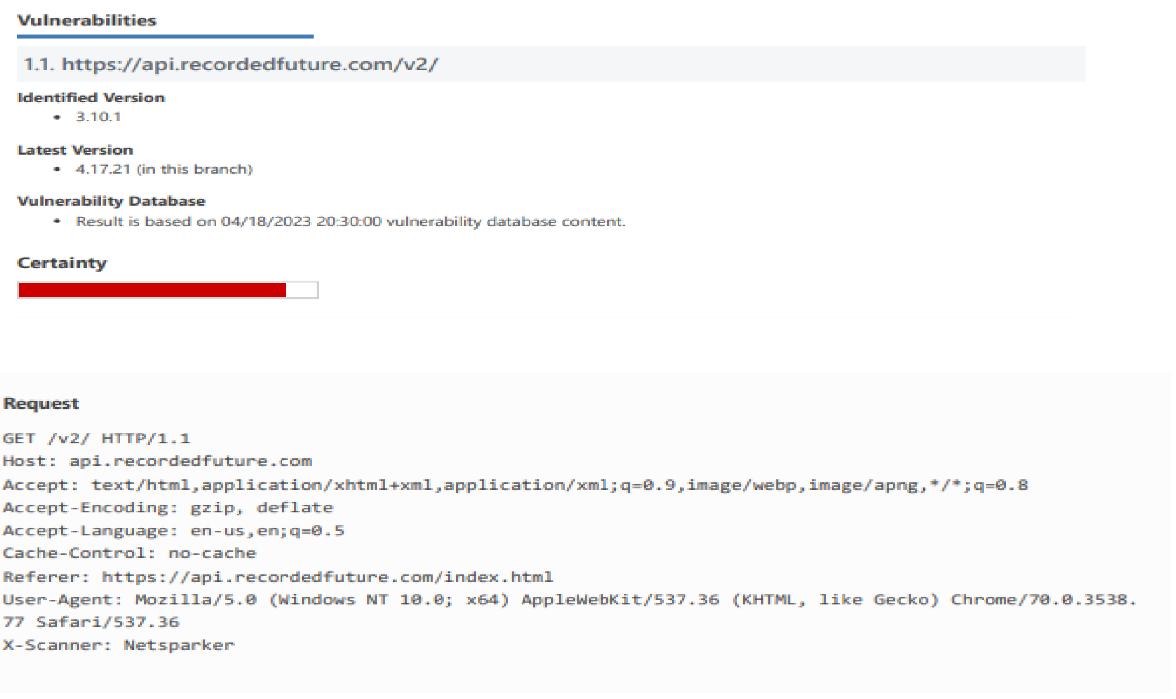
Response

Response Time (ms) : 1 Total Bytes Received : 27 Body Length : 0 Is Compressed : No
```
[NETSPARKER] SSL Connection
```

ii. Out-of-date Version (Lodash) - Critical Vulnerability

An older version of the Lodash library was being utilised in a web project, which I discovered during a recent security audit. Lodash is widely used because of the useful functionalities it gives to developers in JavaScript.

The security implications of using an out-of-date version of Lodash are cause for worry. Libraries are much like any other piece of software in that they might contain previously discovered security flaws that are fixed in subsequent releases. Using an out-of-date version increases the likelihood that security flaws in the web app may be exploited by malicious users.



Vulnerabilities

1.1. https://api.recordedfuture.com/v2/

Identified Version
- 3.10.1

Latest Version
- 4.17.21 (in this branch)

Vulnerability Database
- Result is based on 04/18/2023 20:30:00 vulnerability database content.

Certainty

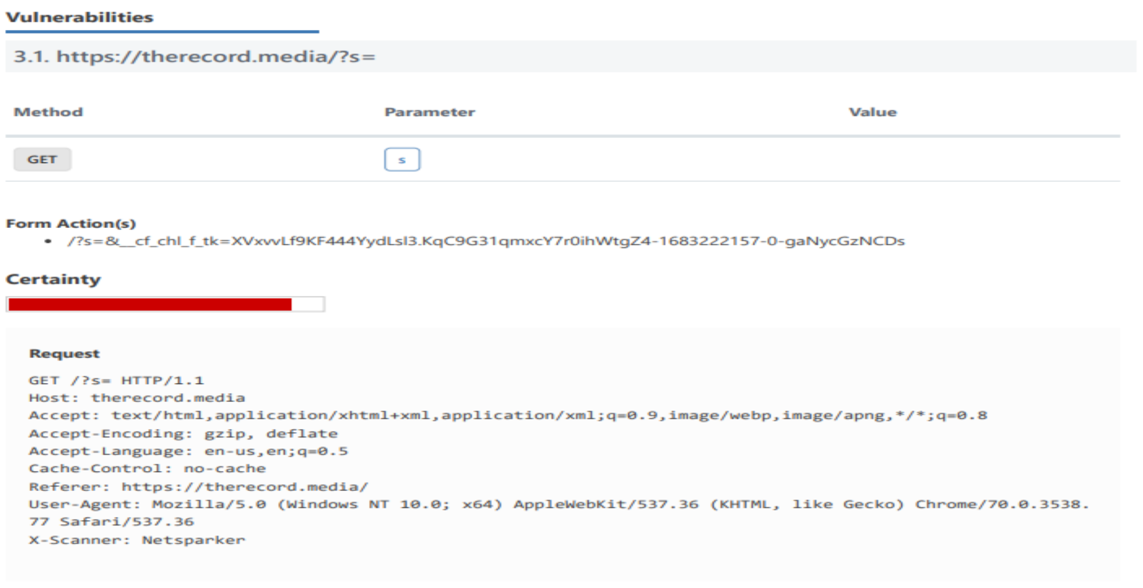
Request

```
GET /v2/ HTTP/1.1
Host: api.recordedfuture.com
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
Cache-Control: no-cache
Referer: https://api.recordedfuture.com/index.html
User-Agent: Mozilla/5.0 (Windows NT 10.0; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.
77 Safari/537.36
X-Scanner: Netsparker
```

iii. [Possible] Cross-site Request Forgery

Recently, I performed a security audit and discovered a possible CSRF vulnerability in a web application. Without the victim's knowledge or agreement, malicious actors can use a technique known as cross-site request forgery (CSRF) to impersonate them on a website and take their desired action.

To execute a CSRF attack, an adversary must convince a user's browser to send an unauthorised request to a target website in which the user has already been authenticated. Several methods exist for this to occur, including the insertion of malicious code into a website or email, the use of carefully constructed links to trick users into clicking on them, and the exploitation of security flaws in third-party integrations.



Vulnerabilities

3.1. https://therecord.media/?s=

Method	Parameter	Value
GET	s	

Form Action(s)
- /?s=&__cf_chl_f_tk=XVxvvLf9KF444YydLsl3.KqC9G31qmxcY7r0ihWtgZ4-1683222157-0-gaNycGzNCDs

Certainty

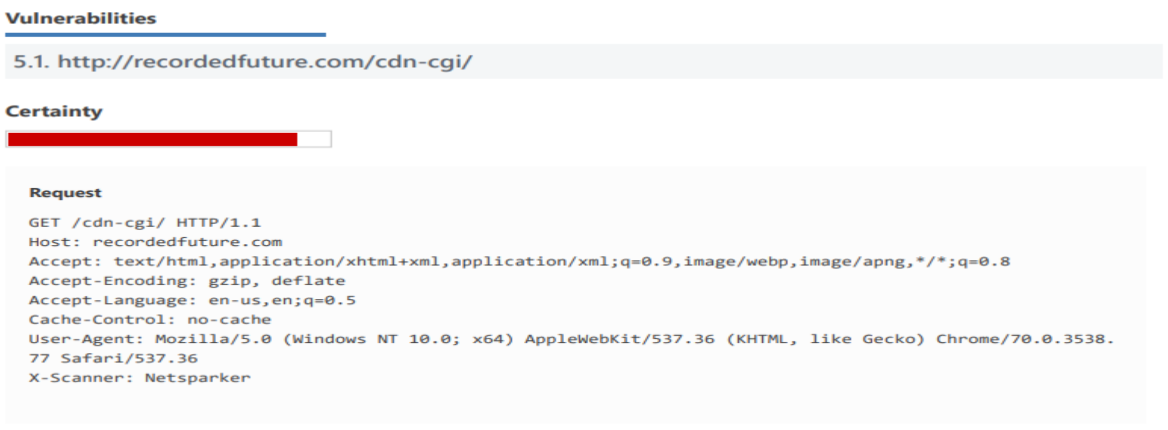
Request

```
GET /?s= HTTP/1.1
Host: therecord.media
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
Cache-Control: no-cache
Referer: https://therecord.media/
User-Agent: Mozilla/5.0 (Windows NT 10.0; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.
77 Safari/537.36
X-Scanner: Netsparker
```

iv. Missing X-XSS-Protection Header

I found a potential security hole in a web app during a recent audit: it did not include the X-XSS-Protection header. By activating the browser's native XSS protection filters, the X-XSS-Protection header is a security feature that helps against Cross-Site Scripting (XSS) attacks.

Setting the X-XSS-Protection header to "1" or "1; mode=block" activates XSS protection in current browsers. By preventing malicious scripts or code from being injected into web pages, XSS attacks are mitigated in browsers that have this feature enabled.

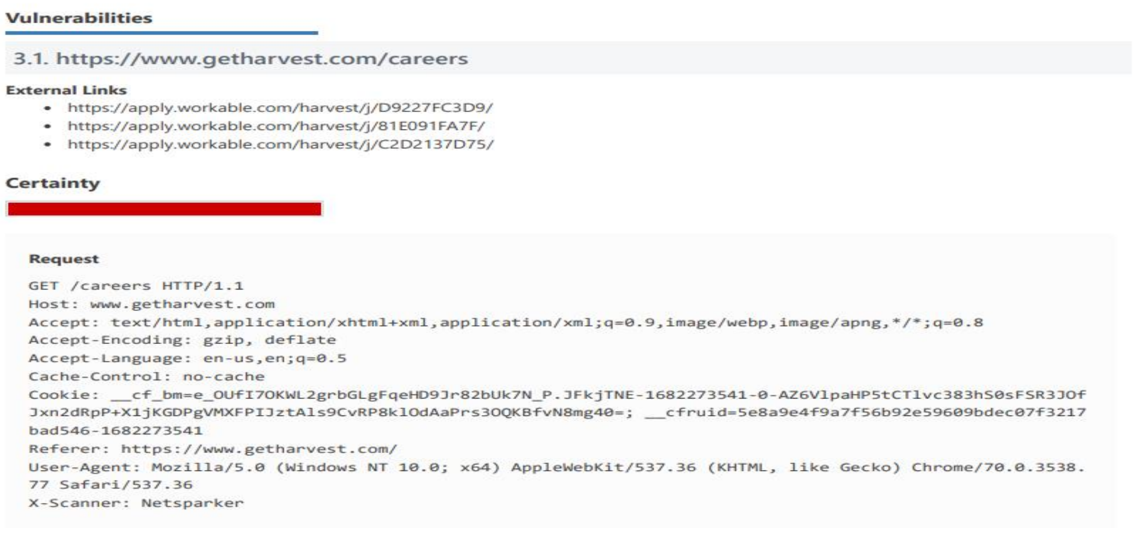


```
GET /cdn-cgi/ HTTP/1.1
Host: recordedfuture.com
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
Cache-Control: no-cache
User-Agent: Mozilla/5.0 (Windows NT 10.0; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.
77 Safari/537.36
X-Scanner: Netsparker
```

v. [Possible] Phishing by Navigating Browser Tabs (Medium)

Recently, I conducted a security audit and discovered a worrying attack method: phishing via browser tab navigation. The purpose of this method is to fool users into divulging private information or taking unintended activities by altering the visual look of browser tabs.

The use of JavaScript or other scripting techniques to alter the content or title of a browser tab is a common tactic in phishing campaigns that employ browser tab navigation. Users may be tricked into visiting malicious websites that seem like banking portals or login pages by exploiting vulnerabilities or falling for social engineering tricks.


Vulnerabilities

3.1. https://www.getharvest.com/careers

External Links

- https://apply.workable.com/harvest/j/D9227FC3D9/
- https://apply.workable.com/harvest/j/81E091FA7F/
- https://apply.workable.com/harvest/j/C2D2137D75/

Certainty

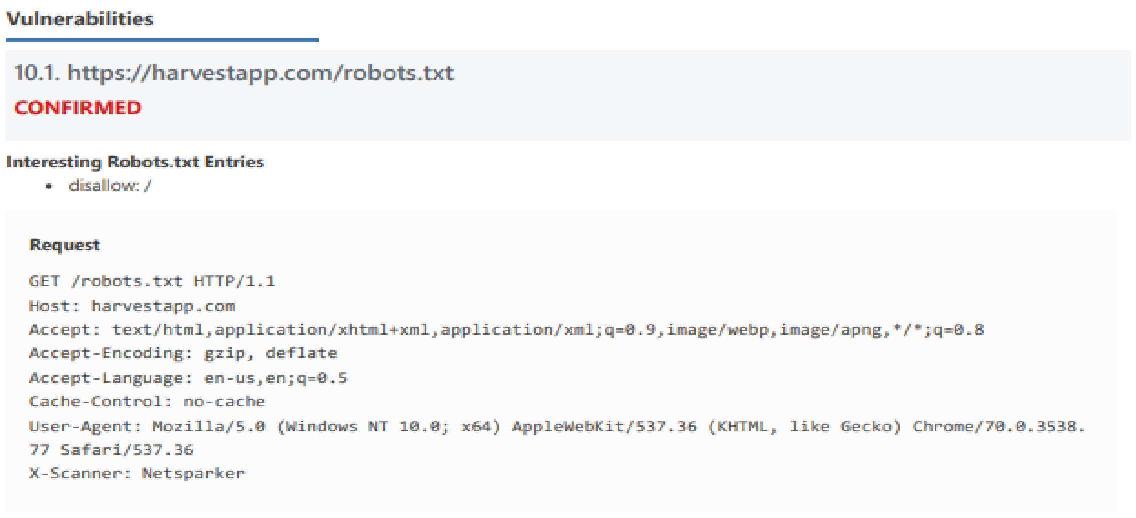
Request

```
GET /careers HTTP/1.1
Host: www.getharvest.com
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
Cache-Control: no-cache
Cookie: __cf_bm=e_OUfI7OKWL2grbGLgFqeHD9Jr82bUk7N_P.JFkjTNE-1682273541-0-AZ6VlpaHP5tCTlvc383hS0sFSR3JOf
Jxn2dRpP+X1jKGDPgVMXFPIJztAls9CvRP8klOdAaPrs3OQKBfvN8mg40=; __cfruid=5e8a9e4f9a7f56b92e59609bdec07f3217
bad546-1682273541
Referer: https://www.getharvest.com/
User-Agent: Mozilla/5.0 (Windows NT 10.0; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.
77 Safari/537.36
X-Scanner: Netsparker
```

vi. Robots.txt Detected

I discovered a robots.txt file on a web app during a recent security audit. Webmasters can instruct web crawlers and search engine robots not to index certain pages of their site by including them in the robots.txt file.

Site administrators can restrict web crawlers' access to specific pages using the robots.txt file, which is read by search engines and other automated bots. It includes directives, such as permitting or disallowing particular user agents or directories, and usually lives in the root directory of a website.


Vulnerabilities

10.1. https://harvestapp.com/robots.txt

CONFIRMED

Interesting Robots.txt Entries

- disallow: /

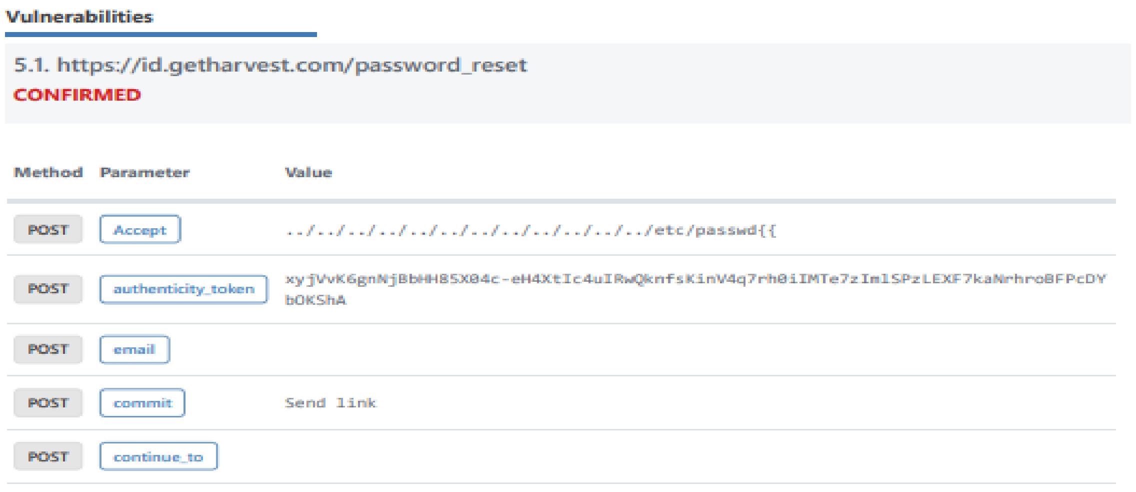
Request

```
GET /robots.txt HTTP/1.1
Host: harvestapp.com
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
Cache-Control: no-cache
User-Agent: Mozilla/5.0 (Windows NT 10.0; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.
77 Safari/537.36
X-Scanner: Netsparker
```

vii. Internal Server Error

When I tried to use a web-based app during my most recent query, I kept getting an Internal Server Error. The HTTP status code 500 corresponds to the standard error message "Internal Server Error" and indicates that the server ran into an issue while trying to fulfil the request.

To sum up, an Internal Server Error happens when the server encounters an unexpected problem while attempting to fulfil a request. You'll need to perform some investigating, debugging, and repairing to get the web app back to regular functioning. As a security professional, it is my mission to identify the root causes of these issues and implement permanent solutions to ensure that all users may browse the web with complete peace of mind.



Vulnerabilities

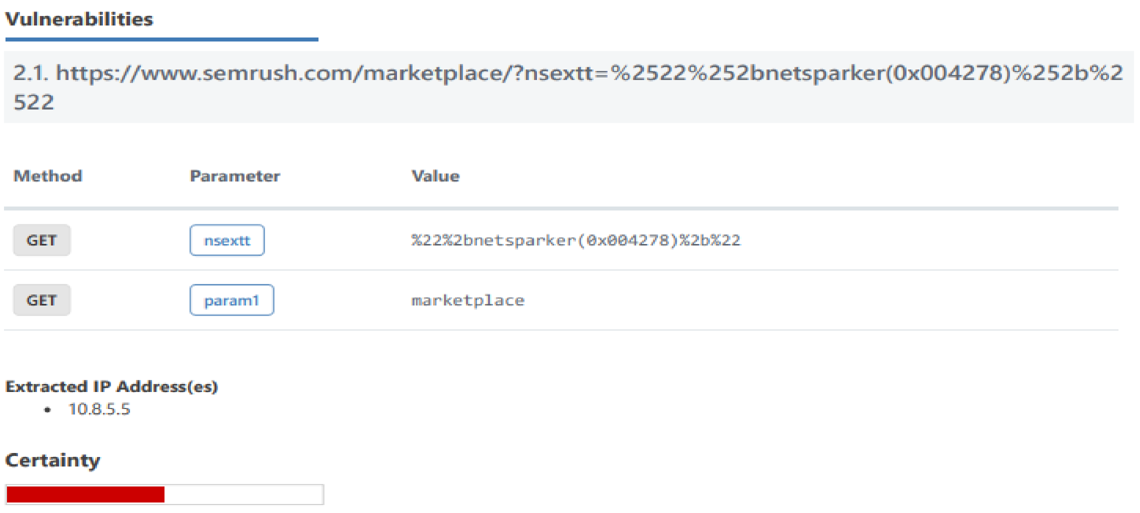
5.1. https://id.getharvest.com/password_reset
CONFIRMED

Method	Parameter	Value
POST	Accept	`../../../../../../../../../../../../etc/passwd{{`
POST	authenticity_token	`xyjVvK6gnNjBbHH85X04c-eH4XtIc4uIRwQknfsKinV4q7rh0iIMTe7zIml5PzLEXF7kaNrhroBFPcDYbOKShA`
POST	email	
POST	commit	`Send link`
POST	continue_to	

viii. [Possible] Internal IP Address Disclosure (Medium)

While doing a security assessment on a website, I recently discovered a potential vulnerability involving the leaking of internal IP addresses. Internal IP address exposure occurs when a website inadvertently discloses the IP addresses of its servers or other infrastructure components.

The exposure of internal IP addresses may pose a significant security risk because attackers get knowledge of the network's layout and may be able to launch targeted attacks or take advantage of vulnerabilities in certain systems.



Vulnerabilities

2.1. https://www.semrush.com/marketplace/?nsextt=%2522%252bnetsparker(0x004278)%252b%2522

Method	Parameter	Value
GET	nsextt	`%22%2bnetsparker(0x004278)%2b%22`
GET	param1	`marketplace`

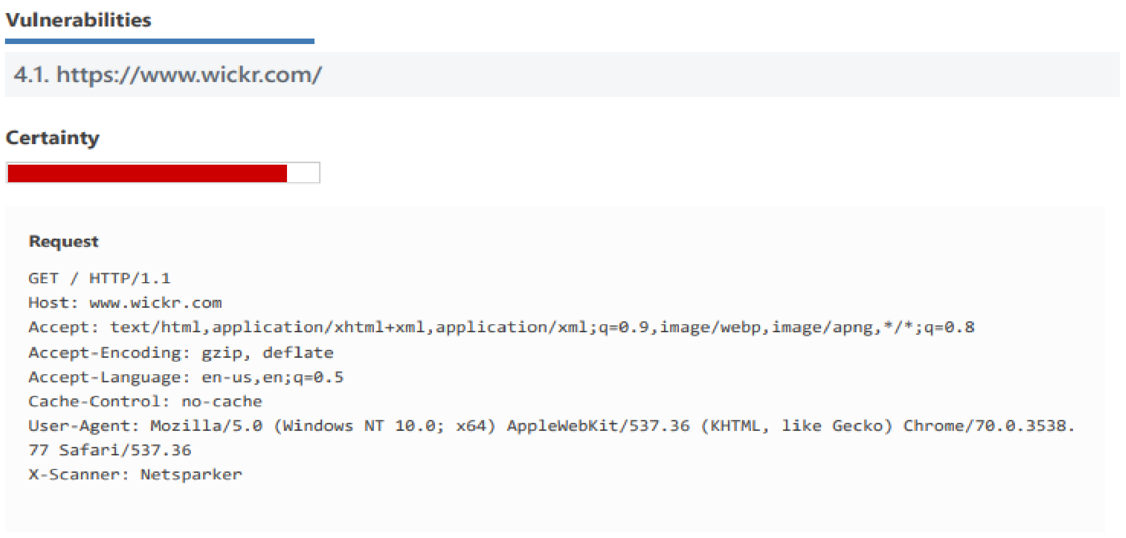
Extracted IP Address(es)
- 10.8.5.5

Certainty

ix. Missing X-Frame-Options Header

Recently, I was conducting a security audit and discovered a potential flaw in a web application due to the lack of the X-Frame-Options header. By restricting how a page may be placed into a frame or iframe on another website, the X-Frame-Options header serves as a security barrier against clickjacking assaults.

The web programme is susceptible to clickjacking attacks since the X-Frame-Options header is missing. If an attacker can get a user to click on a hidden or decoy element on a website, they can perform an unwanted activity or take over the user's session.



Vulnerabilities

4.1. https://www.wickr.com/

Certainty

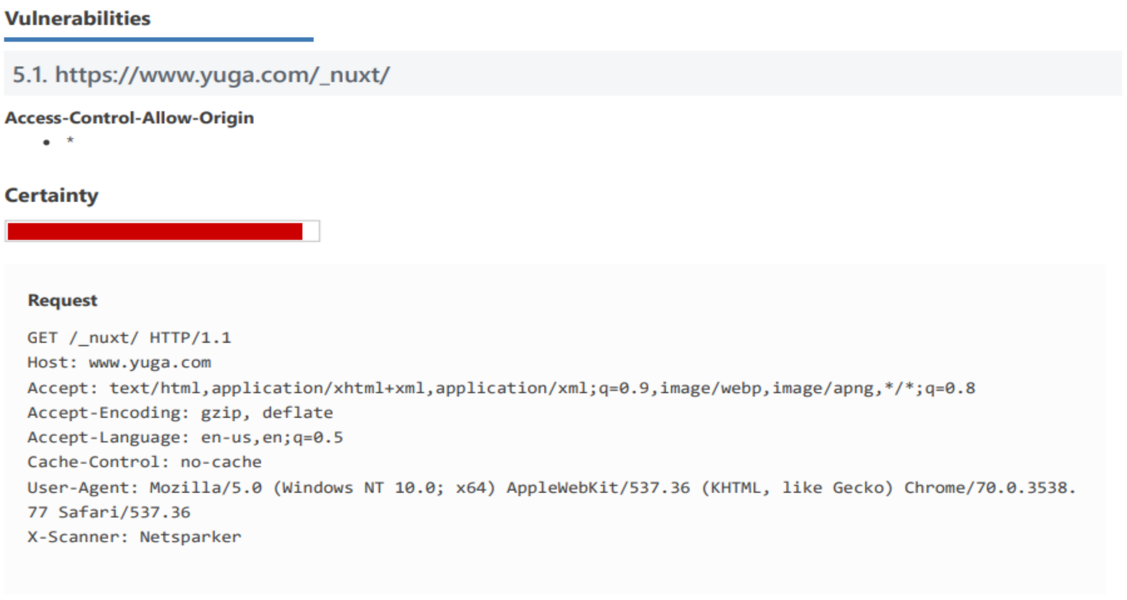
Request

```
GET / HTTP/1.1
Host: www.wickr.com
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
Cache-Control: no-cache
User-Agent: Mozilla/5.0 (Windows NT 10.0; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.
77 Safari/537.36
X-Scanner: Netsparker
```

x. Misconfigured Access-Control-Allow-Origin Header

During a recent application security assessment, I found that the web server's Access-Control-Allow-Origin header was set incorrectly. Using the Access-Control-Allow-Origin header, you may limit requests to your server from specific domains.

A cross-origin request is made when a webpage from one domain tries to submit a request to another site. Web browsers include a built-in security measure called the Same-Origin Policy that blocks potentially dangerous cross-origin requests by default. However, web apps can get around this limitation by using the Access-Control-Allow-Origin header or another CORS technique.



Vulnerabilities

5.1. https://www.yuga.com/_nuxt/

Access-Control-Allow-Origin

- *

Certainty

Request

```
GET /_nuxt/ HTTP/1.1
Host: www.yuga.com
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
Cache-Control: no-cache
User-Agent: Mozilla/5.0 (Windows NT 10.0; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.
77 Safari/537.36
X-Scanner: Netsparker
```

06.2023/05/15 - 2023/05/21 Week Progress

Smart Contract

Today, I got to learn more about the interesting field of smart contracts. As a tech nerd, I've always been curious about blockchain and its applications, and smart contracts in particular have piqued my interest.

Let's begin with a basic definition of a smart contract. The contract's provisions are hardcoded into the programme itself, making it a self-executing agreement. The contract is kept on a distributed ledger, or blockchain, which provides immutability, transparency, and decentralisation. Smart contracts are distinguished from traditional contracts by their capacity to enforce contractual obligations with little to no involvement from third parties and no reliance on mutual faith.

Ethereum is one of the most well-known blockchain systems because of its support for smart contracts, so that's where I started looking. The Solidity programming language built on Ethereum provides a robust infrastructure for creating and deploying smart contracts.

In spite of these difficulties, I have faith in smart contracts. Their ability to revolutionise established structures and give people more agency through decentralised, automated, and reliable transactions is mind-blowing. I can't wait to learn more about this emerging field and watch it mature into a crucial aspect of our information infrastructure.

Blockchain

Blockchain technology is a novel idea that is causing a change in many different areas, and I had the opportunity to learn more about it today. Blockchain has intrigued me because of its purported ability to improve the security and decentralisation of data storage.

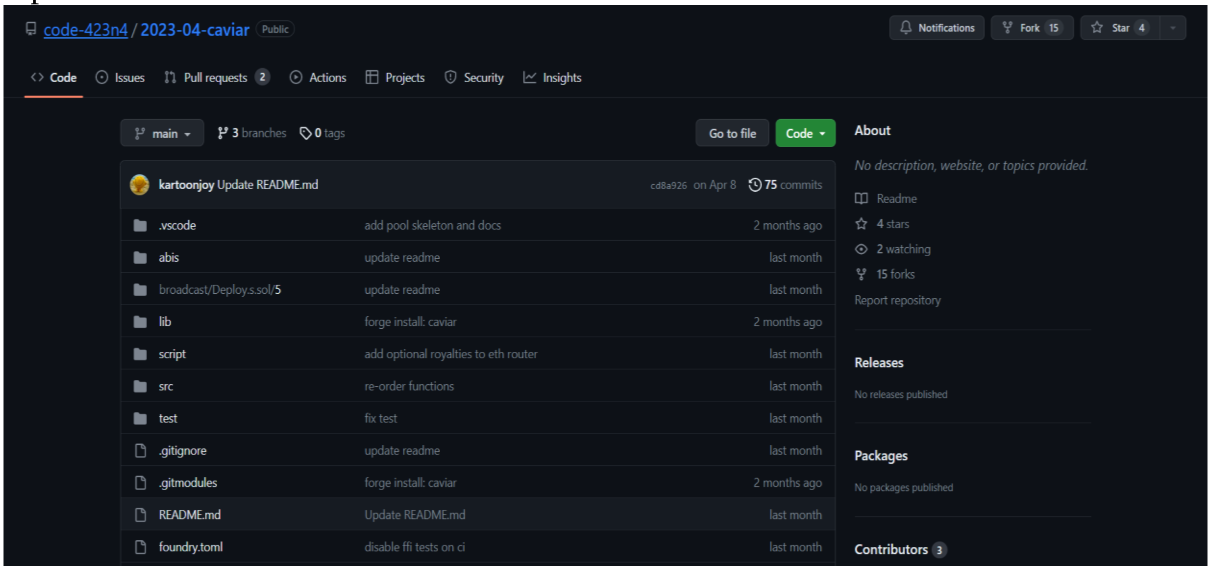
Blockchain, at its core, is a distributed ledger that records and verifies information or transactions over a network of computers known as nodes. The distinctive features of blockchain technology are its decentralisation, immutability, transparency, and security. The way we do business with one another might potentially alter as a result of these skills.

My initial research was to understand the fundamentals of blockchain technology. What I picked up is that blocks of records and transactions are linked together in a cryptographically secure chain. Due to the blockchain's chaining mechanism, altering or tampering with older blocks is computationally impossible, ensuring the data stored there is safe and immutable.

As my reading on blockchain comes to a close, I find myself further impressed by the revolutionary possibilities of this emerging technology. Blockchain's distributed ledger structure has the potential to empower users, boost public faith in institutions, and usher in sweeping changes across several industries. However, a critical perspective is required while engaging with this technology. As we move towards a decentralised and safe future, I'm interested to see how blockchain gets adopted by different sectors.

Try to do Smart Contract CTF

During this period, I reached out to a wide range of different external resources in order to obtain in-depth insights into the use of smart contracts and the method in which they are used. The primary goal was to prepare ourselves for a smart contract Capture The Flag (CTF) challenge, which required an in-depth familiarity with smart contract vulnerabilities. Canvior, which was found on GitHub, was one of the many tools and resources that were investigated in order to obtain a better grasp of smart contract analysis. This understanding was achieved via the use of a range of internet-based tools and resources.
Despite the fact that I was unable to fully download the Slither tool due to unanticipated issues, I proceeded to use the internet tools that were made available to me in order to assist me in completing the CTF challenge. I used these resources in order to help me complete the CTF task. This was done so that it would be easier for me to do the assignment. I was able to produce a report by making the most of the resources and information that were available to me at the time and by accumulating as much material as I possibly could get my hands on. I used this strategy to write the report.


code-423n4 / 2023-04-caviar Public

kartoonjoy Update README.md — cd8a926 on Apr 8 — 75 commits

.vscode	add pool skeleton and docs	2 months ago
abis	update readme	last month
broadcast/Deploy.s.sol/5	update readme	last month
lib	forge install: caviar	2 months ago
script	add optional royalties to eth router	last month
src	re-order functions	last month
test	fix test	last month
.gitignore	update readme	last month
.gitmodules	forge install: caviar	2 months ago
README.md	Update README.md	last month
foundry.toml	disable ffi tests on ci	last month

About
No description, website, or topics provided.
Readme
4 stars
2 watching
15 forks
Report repository

Releases
No releases published

Packages
No packages published

Contributors 3

Challenges

The lack of instructive videos and materials that went into detail regarding smart contracts made it extremely difficult to educate oneself on the topic because they were only available in limited supply on the internet. To get over this challenge will need a significant amount of work and concentration. It took more than a week to do the necessary study on smart contracts and locate the necessary web tools to utilize.

SolidityScan

Today I looked at the safety of smart contracts by using Solidity Scan. Solidity Scan is a static analysis tool for Solidity smart contracts on the Ethereum blockchain and other blockchains. My most important findings:
The Basics of Solid-State Tomography Security flaws in Solidity smart contracts may be discovered with the use of Solidity Scan, a static analysis tool. It performs a comprehensive security audit of the whole contract codebase.The use of Solidity Scan is crucial because it allows programmers to spot problems with smart contracts at an early stage and correct them. This method strengthens the safety of smart contracts, lessens the possibility of exploits, and safeguards user assets.

Features:

- Identifying security flaws: Solidity Scan examines contract code for common security flaws including reentrancy attacks, integer overflow/underflow, unchecked external calls, and more. Issues that might result in financial loss or harm to the contract are uncovered.

- The programme also analyses code quality, best practises, and compliance with the Solidity coding guidelines. It proposes ways to improve the code's readability, maintainability, and performance.

- Solidity Scan's gas estimation feature helps the Ethereum network scale efficiently and reduces transaction fees. It pinpoints the contract areas that can benefit from gas optimisation in terms of cost savings and productivity gains.

- Solidity Scan may be integrated with a variety of development processes, including continuous integration and code review. This allows for instant problem detection and resolution thanks to automated scanning and immediate feedback.

07.<u>2023/05/22 - 2023/05/27 Week Progress</u>

Finally, I have finished and successfully turned in my reports for the bug reward, smart contract, and diary. In order to successfully complete the assignment, you have to submit these three reports. Each report was painstakingly crafted in order to fulfil the objectives and deliver an in-depth analysis. The findings of the study as well as its interpretations were presented in the journal article, which served to shed light on the most important discoveries. This study on smart contracts has paid particular attention to the ways in which they may go wrong, how serious the hazards are, and what can be done about them. Specifically, this research has focused on the ways in which they might go wrong. An analysis of the overall results of the programme was given in the concluding section of the bug bounty report. In this part, the implications of the vulnerabilities that were discovered and the countermeasures that were implemented were discussed. When it comes to locating and eliminating potential dangers, having these reports finished represents a significant advancement in safety since it makes it possible to adopt a more preventative approach to management of the situation.

08.Conclusion

In conclusion, participating in the bug bounty programme has been an excellent way for me to expand my knowledge of cybersecurity while also providing me with a great deal of personal satisfaction. There were a lot of bumps on the road during my studies, but they all helped me learn something new and improve my problem-solving skills. During my bug bounties, I researched the difficulties of discovering and reporting vulnerabilities. Using a methodical approach, I was able to locate vulnerabilities in a wide variety of online services and infrastructures. The importance of bug-bounty programmes in safeguarding digital ecosystems, including smart contracts, was highlighted by this experience.

My research into smart contracts also provided me with a thorough understanding of the flaws that exist in these agreements. I looked into what these vulnerabilities meant, if they could be exploited, and how severe they actually were, all while actively trying to reproduce and verify them. Because to my work fixing these issues, smart contract systems are now more secure and reliable than ever before. Finding vulnerabilities in smart contracts and online services on my bug bounty tour presented me with a number of interesting challenges that helped me grow as a professional and as a person. This study has provided me with invaluable skills, deepened my knowledge of cybersecurity, and reinforced the necessity of continued efforts to strengthen the safety of digital ecosystems.

References

[1] "youtube," [Online]. Available: https://www.youtube.com/watch?v=psu3GTKS_us&t=12s.

[2] "csp," [Online]. Available: https://scotthelme.co.uk/csp-cheat-sheet/.

[3] Blockchain. [Online]. Available: https://www.youtube.com/watch?v=SSo_EIwHSd4.

[4] XSS. [Online]. Available: https://portswigger.net/web-security/cross-site-scripting.

[5] XXE. [Online]. Available: https://portswigger.net/web-security/xxe.

[6] "SQL," [Online]. Available: https://portswigger.net/web-security/sql-injection.

[7] [Online]. Available: https://www.veracode.com/security/owasp-top-10.

[8] "Bug," [Online]. Available: https://www.techtarget.com/whatis/definition/bug-bounty-program.

[9] "BUG HUNTING METHODOLOGY FOR BEGINNERS," [Online]. Available: https://infosecwriteups.com/bug-hunting-methodology-for-beginners-20b56f5e7d19.

[10] "Bug Bounty Methodology — Bug Hunting Checklist (PART-1)," [Online]. Available: https://apexvicky.medium.com/bug-bounty-methodology-bug-hunting-checklist-part-1-3274ad868209.

[11] "Top 10 security tools for bug bounty hunters," [Online]. Available: https://resources.infosecinstitute.com/topic/top-10-security-tools-for-bug-bounty-hunters/.