**PENETRATION**

**TEST REPORT**

Exploiting vulnerabilities in DVWA

Presenter:

**KHIRSAGAR RISHITHA**

KHIRSAGAR RISHITHA

**CSE-Cybersecurity** undergrad student at the **Institute of Aeronautical Engineering.**

rishithakhirsagar07@gmail.com

TABLE OF CONTENTS

* EXECUTIVE SUMMARY
* SEVERITY SCALE
* METHODOLOGY
* ENUMERATION
* VULNERABILITY ASSESMENT
* EXPOITATION
* MITIIGATION
* CONCLUSION

EXECUTIVE SUMMARY

The objective of this writeup is to identify these vulnerabilities and then recommend the strategies and guidelines on how to mitigate the identified vulnerabilities.

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers in learning about web application security in a controlled classroom environment.

* **CRITICAL Severity issue** :A flaw that, if exploited, could have a severe and potentially devastating impact on the security, integrity, or availability of a system or application. These vulnerabilities are considered the most serious and demand immediate attention and remediation. Critical severity issues often pose a high risk of unauthorized access, data breaches, or the compromise of critical systems.

**SEVERITY SCALE**

* **HIGH Severity issue:** A "High Severity Issue" in the context of security vulnerabilities refers to a significant flaw that, if exploited, can have a serious impact on the security, integrity, or availability of a system or application. While not as critical as a vulnerability labeled as "Critical," a High severity issue still demands prompt attention and remediation. Organizations typically prioritize the resolution of High severity issues to minimize the risk of exploitation and potential damage.
* **MEDIUM Severity issue:** A "Medium Severity Issue" in the context of security vulnerabilities refers to a security flaw that, if exploited, could have a moderate impact on the security, integrity, or availability of a system or application. While not as critical as high or critical severity issues, medium severity issues still require attention and remediation, though they may not demand immediate action.
* **LOW Severity issue:** A "Low Severity Issue" in the context of security vulnerabilities refers to a security flaw that, if exploited, would have a minimal impact on the security, integrity, or availability of a system or application. While low severity issues are less critical than medium, high, or critical severity issues, they should still be addressed in a timely manner as part of a comprehensive security strategy.

I have employed testing methods that are widely adopted in the cyber security assessment industry. This Includes five phases:

METHODOLOGY

1. Information Gathering
2. Enumeration
3. Vulnerability Assessment
4. Exploitation
5. Reporting/Mitigation

INFORMATION GATHERING

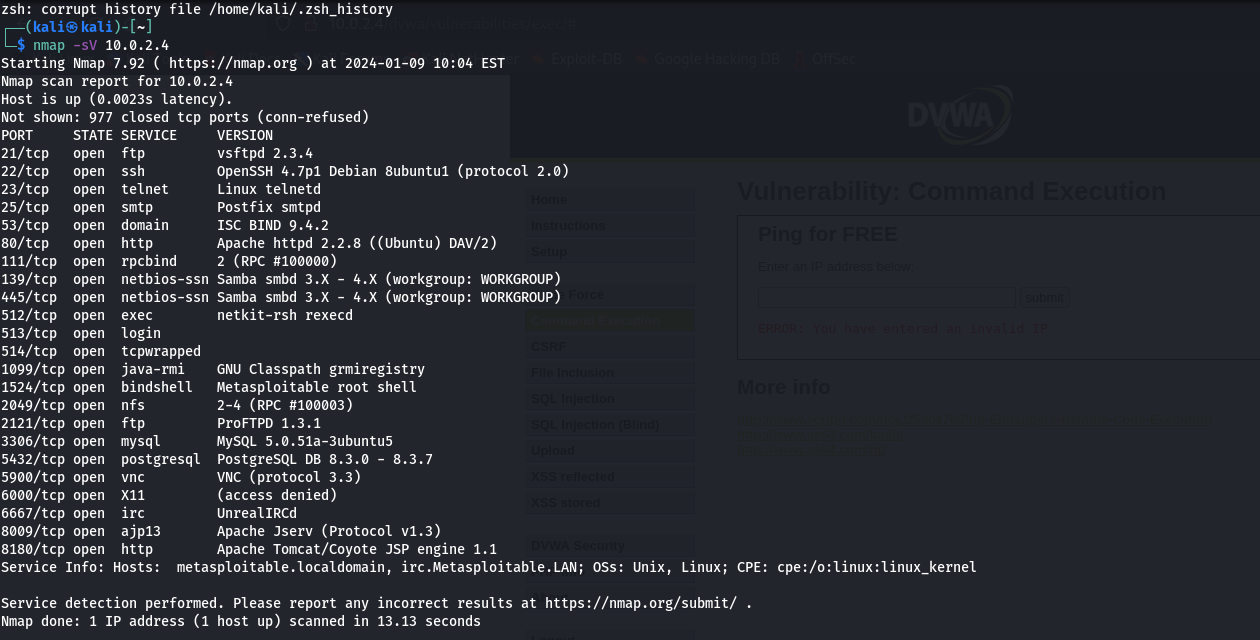
The objective of the information gathering phase is to collect details about the Damn Vulnerable Web Application (DVWA), including its version, technologies in use, and potential entry points for further analysis.

TARGET NAME: DVWA

TARGET IP ADDRESS:10.0.2.4

ENUMERATION

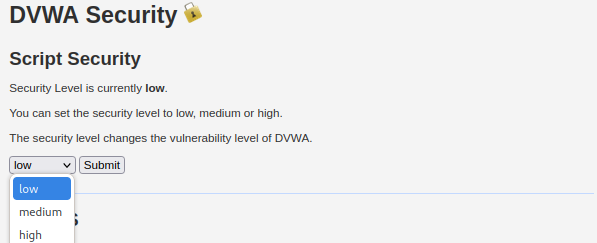
I have performed service enumeration to discover information about the services provided by DVWA server that may reveal critical details that could be leveraged to bypass security and gain an initial foothold into the system .I found that a lot ports are open.



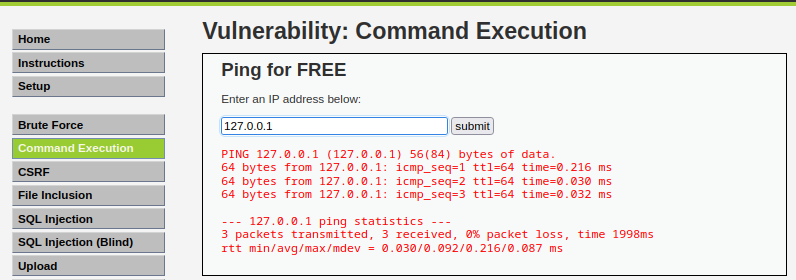
VULNERABILITY ASSESMENT

**Command Injection**

Click on the DVWA Security on the left menu and change the security level to low.



Command injection is an attack in which the goal is execution of arbitrary commands on the host operating system via a vulnerable application. Command injection attacks are possible when an application passes unsafe user supplied data (forms, cookies, HTTP headers etc.) to a system shell. In this attack, the attacker-supplied operating system commands are usually executed with the privileges of the vulnerable application. Command injection attacks are possible largely due to insufficient input validation.



## **Mitigation**

1. Avoid calling OS commands from the “client-side” or application layer

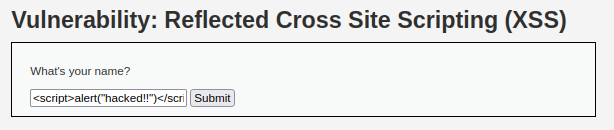
It is best to never call out to OS commands from application-layer code. Suitable alternatives include implementing built-in language libraries such as python’s “OS” library or utilizing APIs.

1. Sanitize user-supplied input Implement strong user-supplied input validation using methods such as using a whitelist of acceptable characters (input) that the application will accept or that the input contains only alphanumeric characters, no other syntax or whitespace.

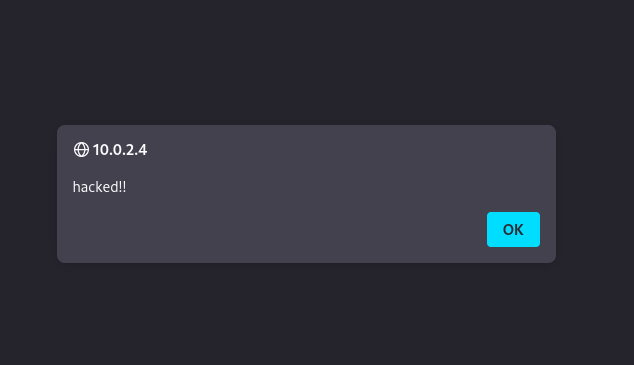
**REFLECTED XSS**

Click on the XSS reflected button on the left menu to access the challenge in the DVWA .

Enter the payload <script>alert(1)</script> in the input field and submit the request



You can see a popup box below which confirms that it is vulnerable to reflected XSS and we have successfully exploited it at low-level security.



We can replace alert(“hacked!!”) function with alert(document.cookie) in the above payload to get the cookie of the logged-in user on the victim browser.

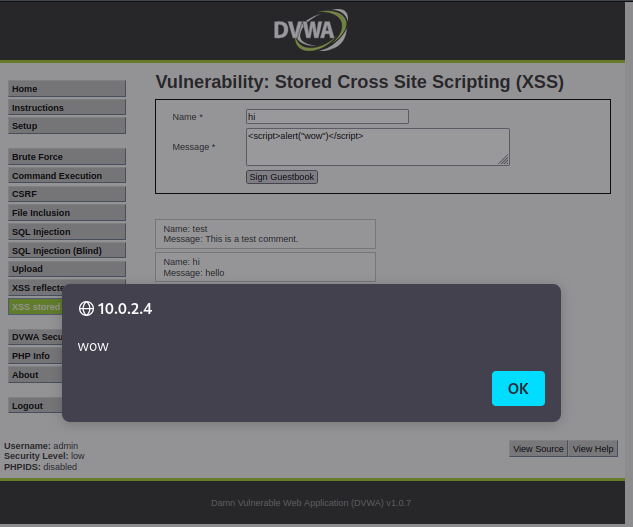
**VULNERABILITY EXPLOITED SEVERITY: LOW**

**VULNERABILITY EXPLANATION:** Reflected XSS occurs when the input supplied by the user reflects back in the browser window or inside page source of the web page. What does it mean? Let us understand it with an example, suppose I have entered some value let’s say thisisreflecting in the input field of the website, now open the source of the page by pressing CTRL+U and search for the string thisisreflecting in the page source. If this word (thisisreflecting) is reflected or present in the page source then that parameter which is accepting the input may be vulnerable to reflected XSS. Now, you can try the payload <script> alert() </script> in place of thisisreflecting in the same input field. If it is vulnerable it will give a popup.

XSS STORED

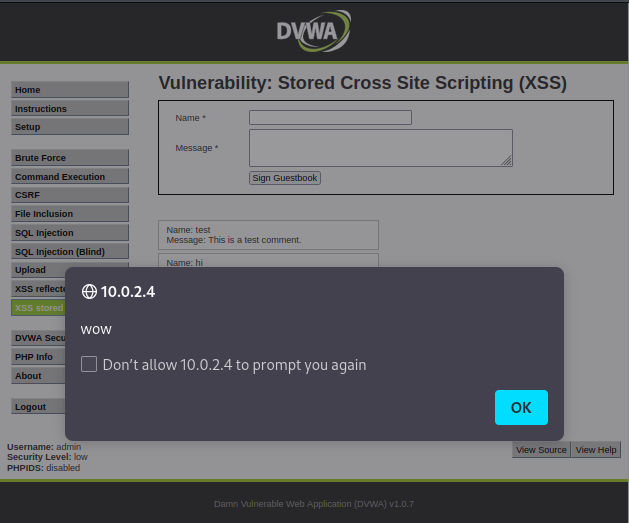
Click on the XSS stored button on the left menu to access the challenge in the DVWA .

Enter <script>alert(“wow”)</script> in Message field. Click on Sign Guestbook to submit the message. If this site is vulnerable to stored XSS vulnerability then we will get a popup when we refresh this page.



So we have successfully exploited Stored XSS at low security. Now each time we refresh the same page we will get this alert box because our XSS payload is stored in the Guestbook.

So, everytime when we open the application or when we refresh the application, the application pop ups the box as long as it is saved in the guestbook.



**VULNERABILITY EXPLOITED SEVERITY: LOW**

**VULNERABILITY EXPLANATION:** Stored XSS is the most dangerous cross-site scripting vulnerability. This type of vulnerability arises whenever a web application stores user-supplied data for later use in the backend without performing any filter or input sanitization. Since the web application does not apply any filter therefore an attacker can inject some malicious code into this input field. This malicious code can also be a valid XSS payload. So whenever any person visits the vulnerable page where malicious code is injected he will get a popup on his browser window. This will prove that the given webpage is vulnerable to Stored XSS vulnerability.

**MITIGATION:**

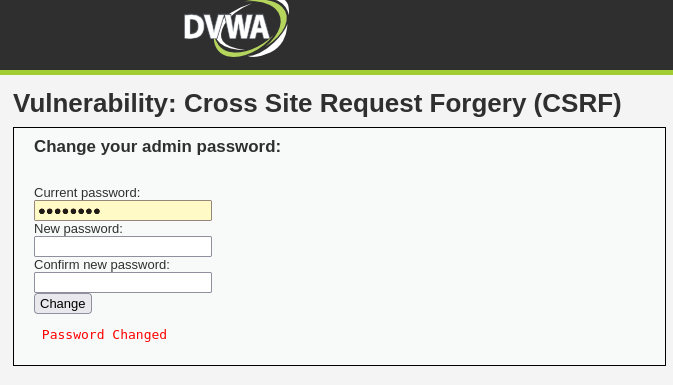
* Filter input on arrival
* Encode data on output
* Use appropriate response headers
* Content Security policy.

CSRF (Cross-site request forgery)

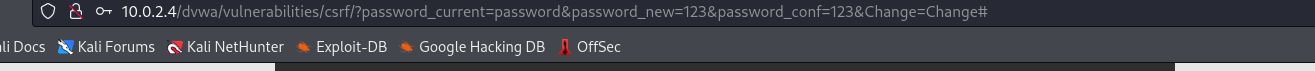
The vulnerability lies in the fact that the code doesn’t include any mechanism to verify the origin of the request. As a result, an attacker can construct a URL containing the necessary parameters (password\_new and password\_conf) and send it to a victim. If the victim clicks on the malicious link while authenticated on the vulnerable website, the code will execute the password change without any further authentication or user consent.

Now, We are going to perform the attack

First, I will Create a new password “123” and click on Change



After changing the password you can see in the url is that it lacks the necessary CSRF token. In the absence of CSRF protection, an attacker can still exploit this vulnerability by tricking the victim into clicking on the URL while logged in to the vulnerable website.



**VULNERABILITY EXPLOITED SEVERITY: LOW**

**VULNERABILITY EXPLANATION:** Cross-site request forgery is a type of malicious exploit of a website where unauthorized commands are submitted from a user that the web application trusts.In a CSRF attack, an innocent end user is tricked by an attacker into submitting a web request that they did not intend. This may cause actions to be performed on the website that can include inadvertent client or server data leakage, change of session state, or manipulation of an end user’s account.

## **Mitigation**

1. Making sure that the request you are receiving is valid

2. Making sure that the request comes from a legitimate client.

3. Implement an anti CSRF Token.

CONCLUSION

In conclusion, the penetration testing of Damn Vulnerable Web Application (DVWA) has yielded critical insights into the application's security landscape. Identified vulnerabilities, including SQL injection, cross-site scripting, and cross-site request forgery, have been meticulously documented and categorized by severity. This information provides a prioritized roadmap for remediation efforts, addressing potential risks to the confidentiality, integrity, and availability of the application.

The assessment also validated the effectiveness of existing security controls and incident response mechanisms while acknowledging noteworthy security measures. The report offers clear and actionable recommendations for remediation, emphasizing ongoing monitoring, regular updates, and continuous training for sustained security resilience. Collaboration among development, IT, and security teams is essential for the successful implementation of these measures.

Ultimately, this penetration testing report not only pinpoints vulnerabilities but serves as a strategic guide for improving DVWA's overall security posture. The findings inform decision-making, resource allocation, and the cultivation of a culture committed to continuous enhancement in web application security.