# Logotipo Descripción generada automáticamente con confianza media

# 

Penetration testing

An integral part of IT security

Patricia Yue Llamazares Rol V 1.0

30 de enero de 2023

Contents

[Presentation table 2](#_Toc160128064)

[Introduction 3](#_Toc160128065)

[Penetration Testing 3](#_Toc160128066)

[OWASP Top 10 4](#_Toc160128067)

[Damn Vulnerable Web Application 4](#_Toc160128068)

[Attack types 6](#_Toc160128069)

[Used tools 6](#_Toc160128070)

[Environment setup 6](#_Toc160128071)

[Problems encountered and troubleshooting 6](#_Toc160128072)

[Attack execution 6](#_Toc160128073)

[Suggested safeguards 6](#_Toc160128074)

[Bibliography 7](#_Toc160128075)

# Presentation table

|  |  |
| --- | --- |
| **Title of the paper** | Penetration testing  An integral part of IT security |
| **Name of the author** | Patricia Yue Llamazares Rol |
| **Name of the tutor** | David Prieto Diaz |
| **Course** | Cybersecurity |
| **Keywords** | Penetration Testing, IT security, audit |
| **Summary** | |
| The following document presents the methodology necessary to perform a professional penetration testing audit. Its purpose is to find vulnerabilities in a vulnerable website, made ad hoc for cybersecurity and ethical hacking students to practice legally in a controlled environment.  The penetration testing audits are considered essential for any company, as they identify vulnerabilities often missed in routine checks, ensure compliance with the existing regulations, evaluate the defense capabilities, and reduce the risk of cyberattacks and data breaches.  Throughout this paper, a series of attacks will be performed on “DVWA” (Damn Vulnerable Web Application) with the aim of emulating a real audit with its corresponding pentesting report.  To conclude, the report is the final product offered to the client where the results of the audit are presented together with some security recommendations to solve the vulnerabilities detected, in order to always improve the security of the organization. | |

# Introduction

In today’s interconnected digital landscape, the importance of robust cybersecurity cannot be overstated. Enterprises rely heavily on technology to conduct business, manage data, and serve their customers. However, this digital dependency also exposes them to an array of threats, ranging from malicious hackers to inadvertent vulnerabilities.

Cybersecurity plays a critical role for developers and enterprises, as it has a huge impact on data protection, business continuity and competitive advantage. IT Security encompasses many different practices, technologies and policies designed to safeguard digital assets against unauthorized access, data breaches and other malicious activities.

Applications, websites, and databases developers play a crucial role in the security state of these systems, so they need to follow good practices. Among them, we have Secure Coding Practices, such as Input Validation, to prevent injection attacks, implementing robust authentication mechanisms and fine-grained authorization controls and designing APIs with security always in mind, considering encryption, rate limiting and access controls. But it is not enough to be careful when it comes to code development, a regular Security Testing is crucial to identify vulnerabilities and infrastructure weaknesses that may have been overlooked during code development. Security Testing best practices include Static Analysis, which consists in using tools to analyse code for vulnerabilities during development, Dynamic Testing, for seeking security flaws in real-world scenarios, and the central focus of this paper, Pentesting, which consists of identifying weaknesses through ethical hacking.

## Penetration Testing

Pentesting could be explained as a controlled cyber assault, where an ethical hacker takes on the role of a real attacker to uncover vulnerabilities within a computer system, application, web service, database, etc. The purpose is to identify weak spots in our defenses before malicious actors exploit them.

The essence of pentesting could be explained in three concepts:

* **Simulation of attacks**: Pentesters simulate real-world attacks, probing applications, networks, and infrastructure, with the goal of discovering exploitable weaknesses—be it flawed code, misconfigurations, or insecure protocols.
* **Ethical hacking:** Unlike malicious hackers, pentesters operate within ethical boundaries, as they use hacking techniques to expose vulnerabilities and not to cause any kind of harm.
* **Risk mitigation:** by conducting regular pentests, organizations can strengthen their security posture, safeguard sensitive data, prevent financial losses and ensure their business continuity, and thus giving confidence to customers, suppliers and partners.

## OWASP Top 10

As discussed above, knowledge about the most known attacks and how to safeguard them is extremely important if we want to achieve robust security in our organization. The OWASP Top 10 is a widely recognized standard in the field of web application security. It creates as an essential awareness document for developers, security professionals, and organizations.

The OWASP Top 10 represents a consensus on the most critical security risks faced by web applications, it identifies common vulnerabilities that can lead to data breaches, system compromise, and financial losses.

This standard helps organizations with risk prioritization with their security efforts, by addressing these top risks, they can allocate resources effectively, helps developers, who use the Top 10 as a guide to build secure code, as it promotes best practices and awareness. It´s also important because many regulations and standards reference the OWASP Top 10. Compliance with these guidelines enhances an organization’s credibility.

# Damn Vulnerable Web Application

Many cybersecurity experts believe that the best way to learn to play defense is to play offense, and the Damn Vulnerable Web Application makes it easy for application developers, novice penetration testers, and security-curious management to flex their offensive muscle in the safety of a virtual machine on their own laptop.

DVWA was created by Ryan Dewhurst, although the exact creation date is not specified, the original source for Damn Vulnerable Web Application can be found on GitHub. It is hosted in the repository maintained by Ryan Dewhurst called digininja/DVWA.

According to its creator, in the repository itself he describes DVWA as 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 to learn about web application security in a controlled classroom environment.

“The aim of DVWA is to practice some of the most common web vulnerabilities, with various levels of difficulty, with a simple straightforward interface.”

# Attack execution

## Brute force

### Definition

A brute force attack is a trial-and-error assault on passwords, encryption keys or login credentials. Attackers systematically test a vast range of possibilities until they stumble upon the correct combination.

For this attack I´ll use a list of usernames and a list of passwords, trying every password with every username till success.

### Tools

For the usernames and passwords lists, I employed the following github repositories sources:

Usernames: [giper45s usernames list](https://gist.github.com/giper45/414c7adf883f113142c2dde1106c0c4c)

Passwords: [danielmiesslers passwords list](https://github.com/danielmiessler/SecLists/blob/master/Passwords/Common-Credentials/10k-most-common.txt)

### Attack execution

For this attack, I am going to follow two different methods. First one, using the tool burpsuite and its cluster bomb option inside intruder. Then, a script I made using python for this purpose.

**Burpsuite brute force attack**

1. Interfaz de usuario gráfica, Aplicación

   Descripción generada automáticamenteOpen burpsuite and go to the proxy tab. Then, open browser.

Brute Force Image 1

1. In the browser, introduce the route <http://localhost/DVWA/login.php>.

Brute Force Image 2

1. Interfaz de usuario gráfica, Aplicación

   Descripción generada automáticamenteLog into the DVWA, and go to the Brute Force tab.

3Brute Force

1. Interfaz de usuario gráfica, Texto, Aplicación

   Descripción generada automáticamenteTurn on interception in burpsuite.

4 Brute Force

1. Tabla

   Descripción generada automáticamente con confianza mediaInterfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

   Descripción generada automáticamenteEnter a random username and password and intercept the request through burpsuite proxy. In this case, I introduced user as username and pwd as password.

5 Brute Force

6 Brute Force

1. Interfaz de usuario gráfica, Texto, Aplicación

   Descripción generada automáticamenteIn burpsuite, locate the request in http history and send it to intruder.

7 Brute Force

1. In the intruder interface, we will add the username and password parameters as payloads and select the cluster bomb attack type.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

8 Brute Force

1. We will add the usernames list in payload 1 and the password list in payload 2.

Interfaz de usuario gráfica, Aplicación, Tabla

Descripción generada automáticamente

10 Brute Force

Interfaz de usuario gráfica, Aplicación, Tabla

Descripción generada automáticamente

9 Brute Force

And then we will start the attack.

The attack window is the following one:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

11 Brute Force

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamenteIf we order by length, we can guess that the -user=admin password=password-combination is the correct one, so we check the response.

12 Brute Force

As we can see in this part of the response code, this is the correct combination.

Burpsuite community edition is a powerful and convenient tool for ethical hackers to practise, but as I had such a long list of passwords to try, with the 123 usernames list and the 10 000 passwords it would have to do 1 230 000 requests, which the program takes a pretty long time to reach. This is why a created a python code to automatically do these requests as well but works faster.

The good part about using burpsuite though, is that we can read the response code, which contains some valuable information such as the admin.jpg picture source, which we will use in the future for other attacks.

**Python code attack**

I created a simple python script in order to automatise this process into a file called bruteforce.py.

Texto

Descripción generada automáticamente

13 Brute Force

In this code, first I import the requests module, which allows us to make GET and POST requests to web servers. Then I define the file paths to the users and passwords lists, located in my virtual machine in the same directory. Next line is to open both files in a context manager to make sure that files are properly closed after reading, and then store the files content into lists, as they are easier to work with in python language using loops. Next part of the code is a loop for the brute force attempt, the loops iterate through each combination of usernames and passwords, and for each combination an HTTP GET requests is sent to the brute force DVWA URL.

In order to get the correct combination, the response content is checked for the presence of the word incorrect (as it appears every time you make a wrong attempt), so if the response does not contain “incorrect”, I assumed it implied a successful login attempt, so the corresponding username and password are printed as correct, and the loop stops.

I´m going to show now how I run this code in the terminal kali terminal.

Texto

Descripción generada automáticamente

14 Brute Force

So first we would move to the directory containing the python script and the files with the usernames and the passwords. Then, run the script through the command:

python bruteforce.py

And as a result, it will give back the correct combination of credentials.

### Safeguards

* **Adopt Strong Passwords**: Encourage users to create robust passwords. Strong passwords act as the first line of defense against unauthorized access. Consider enforcing password complexity rules and educating users about secure password practices.
* **Limit IP-based Login Attempts:** Restrict the number of login attempts from a single IP address. If an excessive number of failed login attempts occur, temporarily block further requests from that IP.
* **Implement Two-Factor Authentication (2FA)**: 2FA adds an extra layer of security by requiring users to provide a second form of authentication (such as a code sent to their phone) in addition to their password.
* **Use Specialized Login URLs**: Avoid using default login URLs (e.g., /login). Instead, create unique and less predictable login endpoints. This makes it harder for attackers to guess the login path.

## Command injection

### Definition

Command injection vulnerability consist of an insertion of malicious commands into input fields or parameters of a system, which allows executing arbitrary commands on the target server or application. This can lead to unauthorized access, data leaks, or system compromise.

### Analysis

In the down-right side of the web page, we can see a View Source button, which allows us to read the PHP code behind it.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

1 Command injection

It´s directly explained to us that the value of the $target variable (obtained from the user's input ($\_REQUEST['ip'])) is directly concatenated into the shell command. This is what will allow us to submit input that includes additional commands or special characters, which will be executed by the server.

### Attack execution

The interface looks like this:

Aplicación

Descripción generada automáticamente con confianza baja

2 Command injection

As we can see, it´s just asking us to give an IP address in order to ping a device.

Imagen que contiene Tabla

Descripción generada automáticamente

3 Command injection

If we entered the loopback IP address (used to represent the localhost or the device itself), it gives back the expected ping response. When you ping the loopback address, you are essentially pinging yourself, so the request stays within your own system and does not go out to the network. This test verifies that your network adapter and networking software are functioning properly.

But the interesting part comes when we, as mentioned, try to give some command to see if it prints out valuable data.

If we try to do it directly, like this:

Imagen que contiene Tabla

Descripción generada automáticamente

4 Command injection

Texto, Carta

Descripción generada automáticamenteIt would give back just blank, but when concatenating commands, like this:

5 Command injection

5 Command injection

The server will accept the command and follow the instructions.

### Safeguards

* **Input Validation and Sanitization**:

Validate and sanitize all user inputs before using them in system commands. Ensure that user-supplied data does not contain special characters or metacharacters that could be interpreted as commands.

Use whitelisting to allow only specific characters or patterns in input fields.

* **Avoid Concatenating User Inputs**:

Never concatenate user inputs directly into system commands. Instead, use parameterized queries or prepared statements to separate data from the command.

For example, in SQL queries, use placeholders like “?” or named parameters to bind user inputs securely.

* **Privilege Principle**:

Run your web application with the least privilege necessary. Avoid running it as a superuser (e.g., root) or with excessive permissions.

If your app requires specific privileges (e.g., accessing files), use separate user accounts with restricted permissions.

## Cross Site Request Forgery (CSRF)

### Definition

Cross-Site Request Forgery (CSRF) is a security issue where an attacker tricks a user who is already logged into a website or application causing the user to perform actions they didn’t intend to.

### Analysis

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteWe will check the PHP code the same way as before, with the view source button.

As we can read, with this code, the flaw here is that the code is not doing any mechanism to verify the origin of the request, therefore, an attacker would be able to construct an URL containing the necessary parameters, password\_new and password\_conf in this case and send it to a target.

If the victim clicks on the malicious link while authenticated on the vulnerable website, the code will execute the password change with any kind of extra authentication or user consent.

### Attack execution

In order to perform the attack, first we will make sure the Web lacks any kind of method to avoid CSRF attacks, such as CSRF tokens.

Logged into our session, we will change the password and see how the Web responds.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1 CSRF

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamenteI introduced the new password pwd, and we will see how it responds.

3CSRF

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente cx.,As you can see in the screenshot above (at the code level) and in the screenshot below (as an end user would see it), the password is changed without any secondary verification or whatever.

2CSRF

What we can understand by this behaviour, is that any user logged into the web page will be able to perform this password change with a single request, without any single session verification or any similar protocol.

The way we can take advantage of this vulnerability would be by forcing a user through social engineering techniques, or simply intercepting their request.

Texto

Descripción generada automáticamenteIn order to perform a more developed attack, I created the following html code:

4CSRF

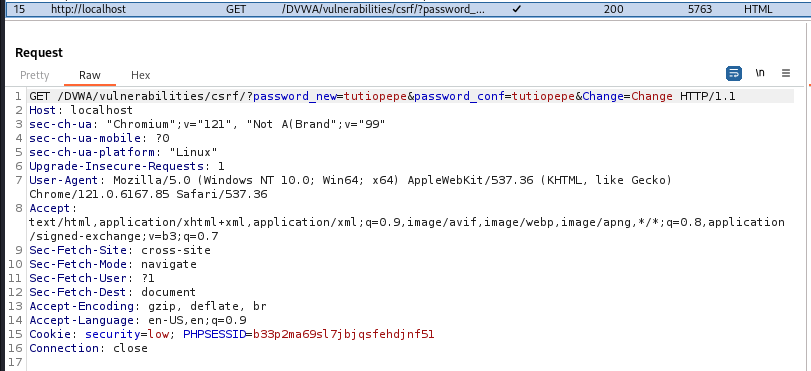
The most interesting part of the code lies in the body, in the line <a … , which aim is to hide the “malicious” URL into an apparently innocent link.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteThis code would be seen in a navigator as something like this:

5CSRF

This would be a simple example of phishing, where an attacker hides the real aim of the URL that you want the target to click. We would be tricking an user into thinking they would win a price by clicking My price link, when they actually will automatically send a request to change their password (successfully if they´re already logged into DVWA) into the one we have chosen, in this case, tutiopepe.

The request would look like the following one:

6CSRF

And, of course, when the user is already logged in, the response will be successful.

Escala de tiempo

Descripción generada automáticamente con confianza baja

7CSRF

### Safeguards

* **Use Same-Site Cookies:**

CSRF attacks are viable because cookies are sent with any requests sent to an origin related to that cookie.

Set your cookies to be same-site whenever possible. This restricts cookies to be sent only with requests originating from the same site.

* **Use CSRF Tokens:**

Implement a challenge token mechanism to prevent CSRF attacks.

Generate a unique token for each user session and include it in forms or API requests.

Validate the token on the server side before processing any sensitive actions.

* **Require Stronger Authentication for Sensitive Actions:**

For critical operations (such as changing passwords or making financial transactions), require stronger authentication.

Consider using two-factor authentication (2FA) or additional verification steps.

## File inclusion

### Definition

A file inclusion attack affects web apps that rely on a scripting runtime.

The scripting runtime refers to the framework in which scripts are executed. These scripts could be JavaScript, Python, PHP, or even SQL command queries.

In web applications, scripting runtimes are responsible for interpreting and executing dynamic code, handling user input, and interacting with the server and client-side components.

A file inclusion attack occurs when an attacker manipulates input parameters to trick a web application into including external files (either local or remote). These attacks exploit the application's use of user-supplied information (such as file paths or URLs) to dynamically load files or execute code.

### Analysis

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

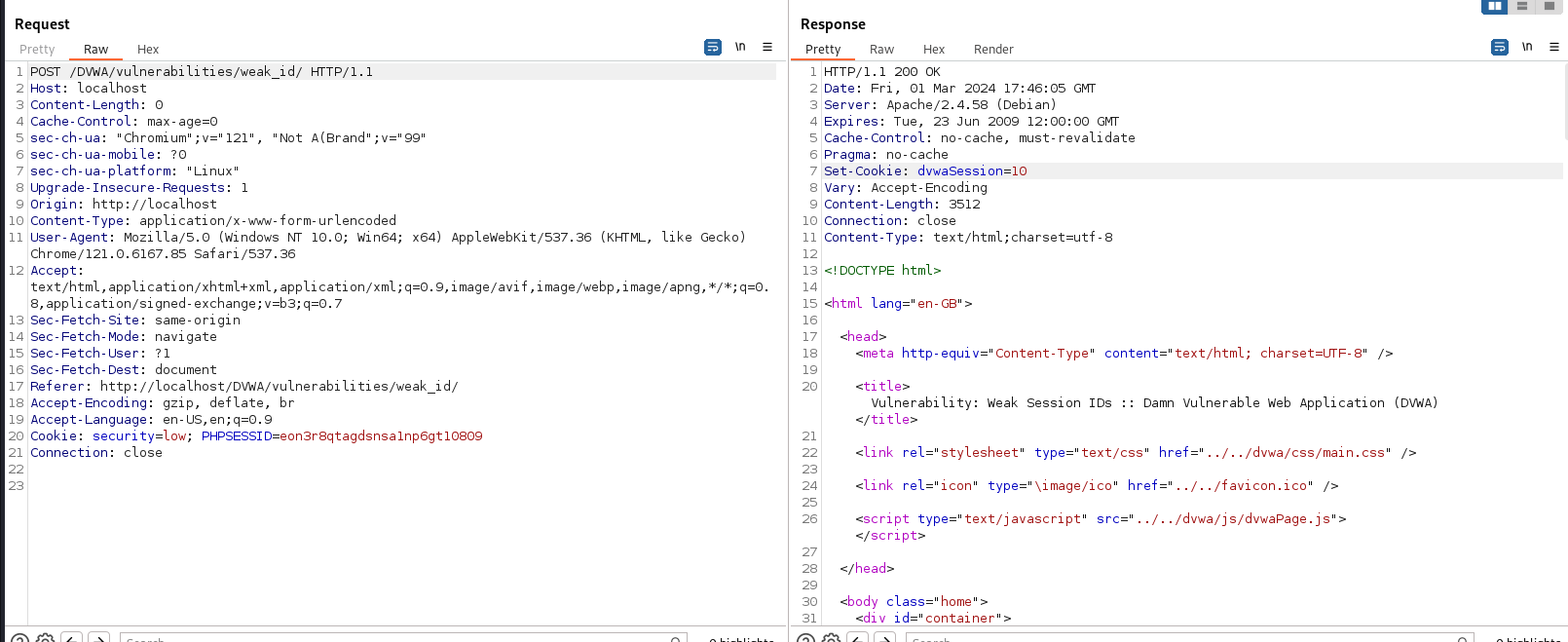
Descripción generada automáticamenteWhat we can see in this section is plain and simply three links that gives us some arbitrary information.

1 File inclusion

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamenteIn the following image we can see what happens if we click one of the links. It´s already giving way too much information. As we can see in the PHP code no kind of filter exists in the background as a process to secure the web application.

2 File inclusion



# 

# Bibliography

1. McKinsey: In their article titled [“The digital enterprises’ cybersecurity,”](https://www.mckinsey.com/capabilities/risk-and-resilience/our-insights/cybersecurity/cybersecurity-linchpin-of-the-digital-enterprise).
2. Institute of Data: Their article on [“Enhancing Cyber Security at the Enterprise Level”](https://www.institutedata.com/blog/cyber-security-at-the-enterprise-level/).
3. IJRPR: In their research paper, [“Cybersecurity Modern Techniques,”](https://ijrpr.com/uploads/V4ISSUE5/IJRPR12757.pdf)
4. Digital Security World: Their article titled [“Why Cyber Security is Crucial for Your Safety”](https://digitalsecurityworld.com/is-cyber-security-important/).
5. Springer: A systematic review of data related to cybersecurity and cyber risk is presented in their [research](https://link.springer.com/article/10.1057/s41288-022-00266-6).
6. Imperva: In their article [“Penetration Testing”](https://www.imperva.com/learn/application-security/penetration-testing/).
7. Cloudfare: in their article [“What is penetration testing? | What is pen testing?”](https://www.cloudflare.com/learning/security/glossary/what-is-penetration-testing/).
8. IBM: in their article [“What is penetration testing?”](https://www.ibm.com/topics/penetration-testing).
9. OWASP in their [Top 10 web page](https://owasp.org/Top10/).