Revista Digital del Departamento de
Ingeniería e Investigaciones Tecnológicas
de la Universidad Nacional de La Matanza
ISSN: 2525-1333
Vol.:10-Nro.1 (julio-2025)

Sección de alumnos: artículo original

Ataques de Path Traversal son aterradoramente fáciles

Path Traversal attacks are scary easy

Integrantes:

Ailén Padilla
Ingeniería Informática - Universidad Nacional de La Matanza
aipadilla@alumno.unlam.edu.ar

Eduardo Couzo Wetzel
Ingeniería Informática - Universidad Nacional de La Matanza
ecouzowetzel@alumno.unlam.edu.ar

Emanuel Juárez
Ingeniería Informática - Universidad Nacional de La Matanza
emjuarez@alumno.unlam.edu.ar

Iván Roberto Sánchez Vedelago
Ingeniería Informática - Universidad Nacional de La Matanza
isanchezvedelago@alumno.unlam.edu.ar

Referentes institucionales:

Martín Zeballos
Universidad Nacional de la Matanza
mzeballos@unlam.edu.ar

Maximiliano Alejandro Downar
Universidad Nacional de la Matanza
adownar@unlam.edu.ar

Aníbal Pose
Universidad Nacional de la Matanza
apose@unlam.edu.ar

Revista Digital del Departamento de
Ingeniería e Investigaciones Tecnológicas
de la Universidad Nacional de La Matanza
ISSN: 2525-1333
Vol.:10-Nro.1 (julio-2025)

Resumen:

En este artículo se explorará path traversal, una vulnerabilidad que se encuentra mayormente en aplicaciones web que, de ser explotada, permite al atacante acceder a archivos a los cuales no debería tener acceso. Se hará foco en cómo es posible explotar esta vulnerabilidad, y en las estrategias que los cibercriminales emplean para eludir medidas de protección insuficientes. Se hablará del impacto que un ataque como este puede tener en la seguridad de un sistema y, por último, se mencionarán algunas recomendaciones para prevenir este tipo ataque.

Abstract:

This article will explore path traversal, a vulnerability commonly found in web applications that, if exploited, allows an attacker to access files they should not be able to access. The focus will be on how this vulnerability can be exploited and the strategies cybercriminals use to bypass inadequate protection measures. The impact of such an attack on a system's security will also be discussed, and finally, some recommendations to prevent this type of attack will be provided.

Palabras Clave: *salto de directorios, hacking web, vulnerabilidad*

Key Words: *path traversal, hacking web, vulnerability*

Revista Digital del Departamento de
Ingeniería e Investigaciones Tecnológicas
de la Universidad Nacional de La Matanza
ISSN: 2525-1333
Vol.:10-Nro.1 (julio-2025)

I. CONTEXTO

La vulnerabilidad path traversal sigue siendo una falla de seguridad persistente en los productos de software. La industria del software ha documentado este tipo de vulnerabilidades, junto con métodos efectivos para erradicarlas durante más de dos décadas. Sin embargo, los fabricantes de software siguen poniendo en riesgos a los clientes al desarrollar productos que permiten la explotación de dicha vulnerabilidad. Han representado aproximadamente el 5% de todos los CVE (Vulnerabilidades y exposiciones comunes) nuevos cada año desde 2014 [1], constituyen aproximadamente el 5% del catálogo de vulnerabilidades explotadas conocidas de CISA (Agencia de Seguridad Cibernética y de Infraestructura de los Estados Unidos) y los ciber criminales abusan de ellas tan ampliamente que CISA y el FBI publicaron una alerta de seguridad por diseño sobre ellas en mayo de 2024 [2].

II. INTRODUCCIÓN

Path traversal, también conocida como directory traversal, es una vulnerabilidad que generalmente se presenta en aplicaciones web, pero puede afectar a cualquier software que maneje ruta de archivos, y de ser explotada permite al atacante acceder a archivos del servidor donde está corriendo dicha aplicación. Esto puede incluir código y datos de la aplicación, credenciales para el backend y archivos confidenciales del sistema operativo. El atacante no solo puede leer estos archivos, sino que, en algunos casos, además podría llegar a modificarlos, afectando la confidencialidad e integridad de estos.

Este ciberataque es posible cuando las entradas de los usuarios se pasan a las API del sistema de archivos o cuando dicha entrada no es validada correctamente. Es un ataque que requiere muy poco conocimiento técnico. Es posible automatizarlo a través de una técnica llamada fuzzing, que implica enviar automáticamente cargas útiles de ataque típicas al objetivo [3].

III. EXPLOTANDO LA VULNERABILIDAD

Podemos imaginar una aplicación web que ofrece un archivo llamado reporte.pdf, con una URL pública como la siguiente: https://www.example.com/?file=reporte.pdf

Ésta corresponde al archivo reporte.pdf en el directorio /var/www/html, en un sistema de archivos del servidor. Para retornar el archivo, la aplicación añade el nombre del archivo solicitado a este directorio base y utiliza una

API del sistema de archivos para leer el contenido del archivo. En otras palabras, la aplicación lee desde la siguiente ruta de archivo: /var/www/reportes/reporte.pdf

Si la aplicación no implementa ninguna defensa en contra de ataques de path traversal, un atacante podría solicitar la siguiente URL para obtener el archivo /etc/passwd del sistema de archivos del servidor.
https://www.example.com/?file=../../../etc/passwd

La secuencia ../ es válida dentro de una ruta de archivo y significa subir un nivel en la estructura de directorios. Las tres secuencias ../ consecutivas suben de /var/www/images/ a la raíz del sistema de archivos, por lo que el archivo que se lee en realidad es /etc/passwd.

En los sistemas operativos basados en Unix, éste es un archivo estándar que contiene detalles de los usuarios que están registrados en el servidor, pero un atacante podría recuperar otros archivos arbitrarios utilizando la misma técnica.
En Windows, tanto ../ como ..\ son secuencias válidas para atravesar directorios. El siguiente es un ejemplo de un ataque equivalente contra un servidor basado en Windows: https://www.example.com/?file=..\..\..\windows\win.ini
Muchas aplicaciones que introducen datos de entrada del usuario en rutas de archivos implementan defensas contra path traversal. Éstas suelen poder eludirse mediante diversas técnicas.

A- Utilizar ruta absoluta: La aplicación bloquea las secuencias de path traversal, pero trata el nombre de archivo proporcionado como relativo a un directorio de trabajo predeterminado. En este caso, podríamos utilizar directamente la ruta absoluta del archivo. En este caso /etc/passwd/. Por ejemplo:
https://www.example.com/?file=/etc/passwd

B- Anidar secuencias de path traversal: En este caso, la aplicación elimina las secuencias de path traversal (de forma no recursiva). Por lo que no podemos utilizar “file=../../../etc/passwd”, pero sí podemos anidarlas para que cuando se eliminen queden las secuencias simples. Por lo tanto, podemos utilizar “file=...//....//....//etc/passwd”. El ejemplo quedaría de la siguiente forma:
https://www.example.com/?file=....//....//....//etc/pas swd

C- Codificar la URL: Aquí, la aplicación también elimina cualquier tipo de secuencia de path traversal. Pero podemos evitar esto mediante la codificación de la URL, o incluso mediante una codificación doble, de los

Revista Digital del Departamento de
Ingeniería e Investigaciones Tecnológicas
de la Universidad Nacional de La Matanza
ISSN: 2525-1333
Vol.:10-Nro.1 (julio-2025)

caracteres ../. Esto da como resultado %2e%2e%2f y %252e%252e%252f respectivamente. También, pueden funcionar varias codificaciones no estándar, como ..%c0%af o ..%ef%bc%8f. Utilizando una codificación simple, la URL quedaría de la siguiente forma:
https://www.example.com/?file=%2e%2e%2f%2e %2e%2f%2e%2e%2f%2e%2e%2fetc/passwd

D- Incluir carpeta base: La aplicación requiere que el nombre del archivo proporcionado por el usuario comience con la carpeta base esperada, como /var/www/reportes. En este caso, simplemente podemos incluir dicha carpeta seguida de secuencias de path traversal adecuadas. Por ejemplo:
https://www.example.com/?file=/var/www/reportes
/../../../etc/passwd

E- Incluir byte nulo: La aplicación requiere que el nombre de archivo proporcionado por el usuario termine con una extensión de archivo esperada, como .pdf. En este caso, podría ser posible usar un byte nulo para terminar efectivamente la ruta de archivo antes de la extensión requerida. Por ejemplo:
https://www.example.com/?file=../../../etc/passwd%
00.pdf

IV. IMPACTO Y SEVERIDAD

La única consecuencia directa de un ataque de path traversal es el acceso a información confidencial. Esta información confidencial puede utilizarse directamente o para realizar otros ataques. Si hay información confidencial almacenada en archivos del servidor, por ejemplo, fotos confidenciales de documentos o datos confidenciales en archivos de texto, el atacante puede encontrar y acceder a dichos archivos.

En otros casos, los ataques de path traversal se utilizan para acceder a archivos típicos que existen en muchos servidores web. Los atacantes pueden utilizar la información de estos archivos para encontrar otros métodos de ataque de seguridad de aplicaciones, lo que puede llevar finalmente a un ataque completo del servidor.

A continuación, se muestran algunos archivos que suelen ser el objetivo de los ataques de path traversal en servidores web basados en Linux, ya que todos estos archivos son legibles para todos los usuarios del sistema operativo:

- /proc/version: contiene la versión del kernel de Linux que se ejecuta en el sistema. Esta información permite al atacante encontrar vulnerabilidades de seguridad para ese kernel de Linux en particular.

- /proc/mounts: contiene una lista de los sistemas de archivos montados actualmente.

Esto permite al atacante intentar acceder a estos sistemas de archivos, por ejemplo, a través de ataques posteriores de path traversal.

- /proc/net/arp: contiene la tabla del protocolo de resolución de direcciones (ARP), que podría usarse para descubrir otros sistemas conectados (posibles objetivos de ataque).

- /proc/net/tcp y /proc/net/udp: contienen listas de conexiones TCP/UDP en curso, que podrían usarse para descubrir otros sistemas conectados (de nuevo, posibles objetivos de ataque).

Según el estándar CVSS v3, path traversal puede ser clasificado en 3 categorías teniendo en cuenta hasta donde se permita llegar con el ataque.

- Gravedad media: Si el ataque permitiera leer solo algunos archivos, sería calificado dentro de la categoría de vulnerabilidad media.

- Gravedad alta: En el caso de que la lectura de archivos fuera posible sobre cualquier archivo dentro del servidor, la vulnerabilidad escalaría a una gravedad alta.

- Gravedad crítica: Si el atacante tuviera la capacidad de modificar archivos, entonces la vulnerabilidad escala al nivel más alto, una gravedad crítica. (Esta categoría no fue abordada en este artículo)

V. CONCLUSIONES

La forma más eficaz de evitar vulnerabilidades de path traversal es evitar por completo pasar la entrada proporcionada por el usuario a las API del sistema de archivos. Muchas funciones de aplicaciones que hacen esto se pueden reescribir para ofrecer el mismo comportamiento de una manera más segura.

Si no se puede evitar pasar información proporcionada por el usuario a las API del sistema de archivos, se recomienda utilizar dos capas de defensa para evitar ataques:

- Validar la entrada del usuario antes de procesarla. Lo ideal es comparar la entrada del usuario con una lista blanca de valores permitidos. Si eso no es posible, verificar que la entrada contenga solo contenido permitido, como caracteres alfanuméricos únicamente.

- Después de validar la entrada proporcionada, agregar la entrada al directorio base y usar una API del sistema de archivos de la plataforma para canonicalizar la ruta. Verifique que la ruta canonicalizada comience con el directorio base esperado.

VI. REFERENCIAS Y BIBLIOGRAFÍA

A. REFERENCIAS BIBLIOGRÁFICAS

[1] CVE Details, “Vulnerabilities by type,” [Online]. Available: https://www.cvedetails.com/vulnerabilities-bytypes.php. [Accessed: 22-Jul-2025].

[2] Cybersecurity and Infrastructure Security Agency (CISA), “Secure by Design Alert: Eliminating Directory Traversal Vulnerabilities in Software,” [Online]. Available: https://www.cisa.gov/resourcestools/resources/secure-design-alert-eliminatingdirectory-traversal-vulnerabilities-software. [Accessed: 22-Jul-2025].

[3] OblivionDev, “Path Traversal Fuzzer,” GitHub Gist, [Online]. Available: https://gist.github.com/OblivionDev/4eac73e9a28fc28a9679. [Accessed: 22- Jul -2025].

Revista Digital del Departamento de
Ingeniería e Investigaciones Tecnológicas
de la Universidad Nacional de La Matanza
ISSN: 2525-1333
Vol.:10-Nro.1 (julio-2025)

B. BIBLIOGRAFIA

[4] Sarit, "What is Directory Traversal | Risks, Examples & Prevention," *Imperva Learning Center*, 20-Dec-2023. [Online]. Available: https://www.imperva.com/learn/applicationsecurity/directory-traversal/. [Accessed: 07- Jul -2025].

[5] P. Arntz, "What is a path traversal vulnerability?," *ThreatDown by Malwarebytes*, 6-Aug-2024. [Online]. Available: https://www.threatdown.com/blog/what-is-a-pathtraversal-vulnerability/. [Accessed: 06- Jul -2025].

[6] OWASP Foundation, "Path Traversal," *OWASP Community*. [Online]. Available: https://owasp.org/www-community/attacks/Path_Traversal . [Accessed: 18- Jul -2025].

[7] "Path Traversal Vulnerability | CWE-22 Weakness | Exploitation and Remediation," *ImmuniWeb*. [Online]. Available: https://www.immuniweb.com/vulnerability/pathtraversal.html#impact. [Accessed: 14- Jul -2025].

[8] C. Cilleruelo, "¿Qué es path traversal?," *KeepCoding Bootcamps*, 19-Apr-2024. [Online]. Available: https://keepcoding.io/blog/que-es-path-traversal/ . [Accessed: 18- Jul -2025].

[9] "Back to Basics: Directory Traversal," *Fastly*, 22-Aug-2023. [Online]. Available: https://www.fastly.com/blog/backto-basics-directory-traversal. [Accessed: 18- Jul -2025].

[10] Invicti, "Directory Traversal (Path traversal)," 17-May-2024. [Online]. Available: https://www.invicti.com/learn/directorytraversal-path-traversal/. [Accessed: 18- Jul -2025].

[11] "What is path traversal, and how to prevent it?," *Web Security Academy*, PortSwigger. [Online]. Available: https://portswigger.net/web-security/file-pathtraversal. [Accessed: 15- Jul -2025].

Recibido: 2025-07-2
Aprobado: 2025-07-24
Hipervínculo Permanente: https://doi.org/10.54789/reddi.10.1.5
Datos de edición: Vol. 10 - Nro. 1 – S.A. Art. 2
Fecha de edición: 2025-07-31