**Tipo de artículo: Artículo original**

# Desarrollo de un Pipeline en Python para el análisis de trazas

# *Development of a Python Pipeline for log analysis*

**Angel Alejandro Guerra Vilches [1*], https://orcid.org/0009-0009-4306-1261**
**Madelín Haro Pérez [2], https://orcid.org/0000-0002-5700-8244**
**Yohandra Echavarria Castillo [3], https://orcid.org/0009-0002-9454-7383**
**Mónica Peña Casanova [4], https://orcid.org/0000-0003-2500-4510**

[1] Facultad de Ciberseguridad, Universidad de las Ciencias Informáticas. La Habana. Cuba. angelagv@estudiantes.uci.cu
[2] Facultad de Ciberseguridad, Universidad de las Ciencias Informáticas. La Habana. Cuba. mharo@uci.cu
[3] Facultad de Ciberseguridad, Universidad de las Ciencias Informáticas. La Habana. Cuba. yoha@uci.cu
[4] Facultad de Ciberseguridad, Universidad de las Ciencias Informáticas. La Habana. Cuba. monica@uci.cu

*** Autor para correspondencia: angelagv@estudiantes.uci.cu**

**Resumen**

El manejo y análisis de trazas se ha convertido en un desafío y una necesidad para las entidades que operan en el ciberespacio. Las trazas, que son registros de eventos que ocurren en los sistemas, pueden contener gran cantidad de información, desde detalles sobre el comportamiento del usuario hasta indicadores de posibles ataques cibernéticos. Sin embargo, el volumen y la complejidad de estos datos pueden ser abrumadores, lo que hace esencial el análisis de trazas. En este artículo se examinará un pipeline desarrollado en Python diseñado específicamente para mejorar la eficiencia y facilitar el proceso de análisis de trazas. Este pipeline representa una serie de procesos que se pueden aplicar a distintos tipos de trazas para estandarizar sus formatos a un fichero CSV, a partir de este fichero se puede extraer información útil, inferir patrones y mejorar la capacidad para detectar y responder a posibles amenazas de seguridad o eventos anómalos en los sistemas. Las implicaciones de este pipeline podrían ser significativas, ya que podría proporcionar a las organizaciones una herramienta eficaz para mejorar sus prácticas de seguridad y fortalecer sus defensas contra las amenazas cibernéticas.

**Palabras clave:** análisis de trazas; ciberseguridad; detección de amenazas; normalización de trazas; pipeline en Python

***Abstract***

*Log management and analysis has become a challenge and a necessity for entities operating in cyberspace. Logs, which are records of events occurring in systems, can contain a wealth of information, from user behavior details to indicators of potential cyber attacks. However, the volume and complexity of these data can be overwhelming, making trace analysis essential. This article examines a Python-developed pipeline specifically designed to enhance efficiency and facilitate the log analysis process. This pipeline represents a series of processes that can be applied to different types of logs to standardize their formats into a CSV file. From this file, useful information can be extracted, patterns inferred, and the ability to detect and respond to potential security threats or anomalous events in systems improved. The implications of this pipeline could be significant, as it could provide organizations with an effective tool to improve their security practices and strengthen their defenses against cyber threats.*

***Keywords:*** *cybersecurity; log analysis; Python pipeline; log normalization; threat detection*

**Recibido: 03/09/2024**
**Aceptado: 19/11/2024**
**En línea: 01/12/2024**

## Introducción

En el campo en constante evolución de la ciberseguridad, se requieren herramientas y técnicas innovadoras para mitigar los daños ocasionados a las entidades por las amenazas emergentes. Una de estas técnicas es el análisis de trazas, un proceso que implica el estudio de los registros de eventos generados por los sistemas que posibilita conocer patrones de comportamiento y posibles indicadores de amenazas.

La gestión de trazas es el proceso que se enfoca en las actividades que se llevan a cabo para identificar los datos necesarios, procesarlos, manipularlos y mostrarlos de forma organizada, contribuyendo a la detección de incidentes y a la toma de decisiones por parte de los especialistas de seguridad (Chuvakin, 2013). De acuerdo con los autores (Cigdem, 2020) y (Peña et al., 2021) este proceso contribuye en auditorías y análisis forense, específicamente en investigaciones internas, establecimiento de líneas base y en la identificación de tendencias operacionales y problemas de comportamiento de los sistemas de información. Según el tipo de aplicaciones, se tienen trazas de software asociados directamente a la seguridad y trazas relativas a los sistemas operativos, las aplicaciones y servicios que se encuentran en ejecución.

En el ámbito de la ciberseguridad, la implementación de un pipeline se vuelve esencial para gestionar de manera efectiva la enorme cantidad de datos generados por diversas fuentes. Esta estructura permite automatizar la recopilación y el procesamiento de información, lo que no solo agiliza el análisis de amenazas y vulnerabilidades, sino que también mejora la precisión de la detección de incidentes. Al transformar datos en bruto en visualizaciones y reportes claros, un pipeline facilita la toma de decisiones informadas y rápidas, permitiendo a los equipos de ciberseguridad responder proactivamente a los riesgos y reforzar la protección de los sistemas.

En el presente artículo se propone un pipeline desarrollado en Python, diseñado con el propósito de simplificar y automatizar el análisis de trazas. Se emplean distintos algoritmos para formalizar diversos tipos de trazas, analizar los datos de las trazas, identificar patrones y destacar posibles amenazas. Adicionalmente, el diseño del pipeline es flexible y escalable, permitiendo su aplicación en diversos contextos donde las amenazas y los sistemas que se deben proteger están en constante evolución. Como resultado, se prevé apoyar el trabajo de los especialistas de seguridad informática al proporcionarles información sobre las trazas de amenazas reales para el sistema.

## Materiales y métodos

Para el desarrollo de la investigación se emplearon los siguientes métodos científicos:

- Analítico – sintético: A partir de la documentación y las trazas de prueba, se precisaron elementos claves como las formas de analizar las trazas, tipos de trazas y su estructura, que permitieron comprender y diseñar el pipeline para el análisis de trazas. De ellos se extrajeron ideas fundamentales como la importancia de la normalización de los datos y la detección de patrones en el comportamiento del usuario, y al mismo tiempo se detalló la información necesaria para la implementación del pipeline propuesto, incluyendo la estructura del código, los algoritmos utilizados y los parámetros de configuración.
- Modelación: Se utiliza en la creación, mediante abstracciones, de un objeto modelado con los rasgos esenciales y los principales componentes que conforman el análisis de trazas. Este objeto modelado incluye la recolección de trazas, la normalización de los datos, el procesamiento y análisis de las trazas y la generación de informes, y componentes principales como los scripts de recolección y normalización, los algoritmos de análisis y los formatos de salida de los informes.
- Experimentación: Se utiliza en la evaluación, en un entorno controlado, de la eficacia de cada uno de los tres scripts que componen el pipeline para el análisis de trazas.

**Tecnologías y herramientas utilizadas**

Se utilizó el lenguaje de programación Python en su versión 3.12.0 para la implementación del pipeline, debido a su versatilidad y amplio soporte para el análisis de datos. Se emplearon bibliotecas estándar de Python como:

- *csv* para la lectura y escritura de archivos CSV.
- *os* para la manipulación de archivos y directorios.
- *re* para el uso de expresiones regulares en la identificación y extracción de campos de las trazas.
- *pandas* para la manipulación y análisis de los datos.

El pipeline consta de tres scripts, cada uno con una función específica:

- Script 1 - Recopilación y normalización
- Script 2 - Limpieza y organización de los datos.
- Script 3 - Análisis de los datos.

Datos de prueba: Los datos utilizados para probar el pipeline consistieron en trazas de los sistemas de Alfresco, Apache Tomcat y Xabal eXcriba usados en la Universidad de las Ciencias Informáticas.

## Resultados y discusión

Sobre los principios y cualidades anteriormente establecidos se desarrolló el modelo del pipeline para el análisis de trazas. El modelo está formado por los componentes mencionados por el autor anteriormente e interrelacionados entre sí, como muestra la Figura 1.

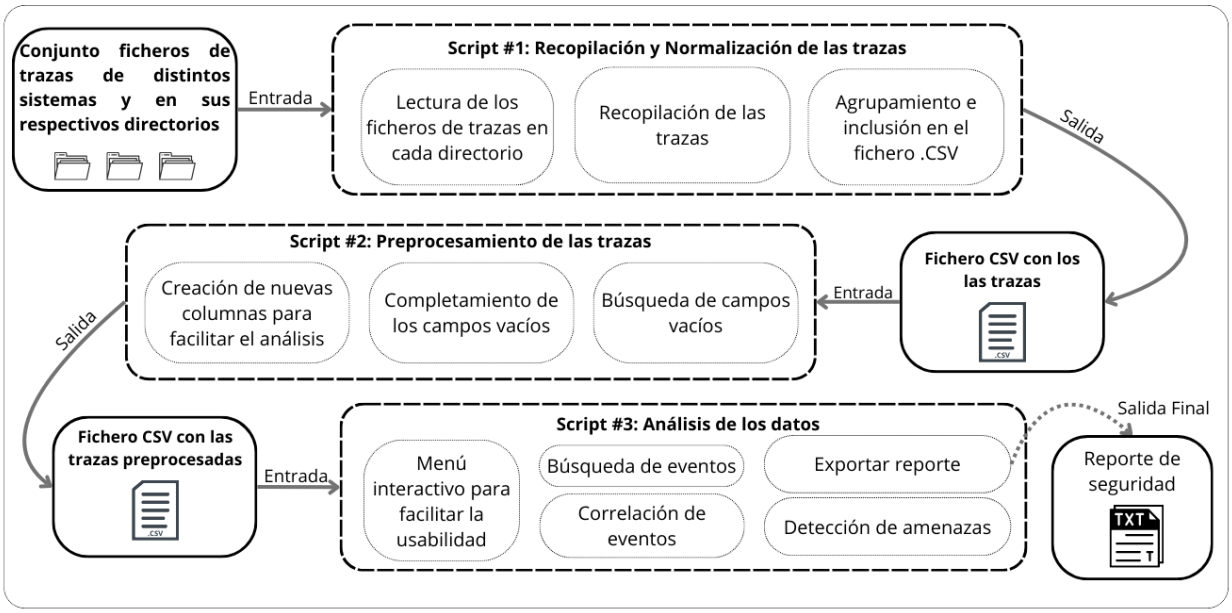


**Figura. 1** - Modelo del pipeline para el análisis de trazas.

**Información de entradas y salidas del pipeline:** Las entradas del pipeline las constituye la información referente a un conjunto de datos de trazas. Como salida se obtiene un reporte de seguridad y un conjunto de datos de trazas normalizados en un fichero con formato CSV que facilita el análisis de las trazas. La relación entre los scripts del pipeline se establece a partir de la información que se define como entrada y salida en cada uno de ellos.

Descripción de los scripts:

- Script 1. Recopilación y normalización: Este script es responsable de la recopilación de registros de eventos, también conocidos como trazas, de varios directorios. Utiliza expresiones regulares para identificar y extraer campos específicos de cada traza. Los datos extraídos se normalizan y se almacenan en un fichero CSV. Este proceso de normalización facilita el manejo de los datos y sienta las bases para el análisis posterior.
- Script 2. Limpieza y organización de los datos: El segundo script del pipeline se encarga de la limpieza y organización de los datos. Este script lee el fichero CSV generado por el Script 1, identifica los campos vacíos y los completa. Además, realiza varias transformaciones en los datos, como la separación de campos, la creación

de nuevas columnas y la eliminación de caracteres especiales, para facilitar el análisis posterior. Al final de este proceso, se obtiene un fichero CSV limpio y organizado, listo para ser procesado.

- Script 3. Análisis de los datos: El último script del pipeline se encarga del análisis de los datos. Este script toma el fichero CSV limpio y organizado generado por el Script 2 y procede a su análisis. Para facilitar la exploración de los datos, este script proporciona un menú interactivo al usuario. A través de este menú, el usuario puede explorar los datos, identificar patrones y detectar posibles amenazas. Además, como parte integral de su funcionalidad, este script genera un informe detallado con la información analizada, proporcionando así una visión integral de los eventos registrados en las trazas.

Para la validación del funcionamiento del pipeline, se utilizó el script 1 para recolectar datos de trazas de diferentes sistemas. Las rutas de los ficheros trazas se establecieron en este script. Se analizaron 784 ficheros de trazas de tres sistemas distintos, proporcionando más de 50,000 registros. Los sistemas incluyen Xabal eXcriba, Alfresco y Apache Tomcat.

Se realizó un preprocesamiento de los datos, con el script 2, para extraer 8 rasgos básicos característicos de las trazas entre los cuales están la marca de tiempo, el evento que provocó la traza y una descripción de los eventos. Posteriormente, se llevó a cabo una transformación de los datos para limpiar los conjuntos de datos existentes, aumentar su relación señal-ruido y reducir su dimensionalidad. Esto incluyó la eliminación de caracteres especiales y la transformación de la fecha a un formato numérico y único.

Con los datos preparados, se utilizó el script 3 y la biblioteca pandas de Python, para manipular y analizar las trazas. Su menú interactivo facilita el proceso de análisis, permitiendo mostrar información útil como eventos más relevantes, horarios de mayor actividad en los sistemas, errores detectados y la cantidad de usuarios conectados. Finalmente, se puede exportar un reporte de seguridad con un resumen detallado de los eventos ocurridos en el sistema.

La Figura 2 muestra un ejemplo del uso del pipeline, en la misma se evidencia el evento más registrado en el sistema y un submenú para conocer más detalles del mismo.


**Figura. 2** Resultados del pipeline.

## Conclusiones

El pipeline propuesto se presenta como una solución para los especialistas de seguridad informática para la obtención de la información de las trazas de amenazas de un sistema. Los ataques informáticos evolucionan cada día. Contar con un mecanismo para analizarlos a través de las trazas garantiza a los especialistas tener un conocimiento agregado y de forma rápida sobre lo ocurrido en sus sistemas.

Los resultados que se procesan con el pipeline tienen un beneficio adicional al poderse emplear, como datos normalizados, en el entrenamiento de algoritmos de machine learning, cumpliéndose de esta forma el objetivo de la investigación.

La valoración de los referentes teóricos corrobora la actualidad y pertinencia del problema abordado. La aplicación de Python para el desarrollo de herramientas de análisis de trazas facilita la compatibilidad entre los sistemas. Se expresan las relaciones esenciales que se establecen entre sus componentes.

## Conflictos de intereses

Los autores no poseen conflictos de intereses.

## Contribución de los autores

1. Conceptualización: Madelín Haro Pérez, Yohandra Echavarria Castillo, Mónica Peña Casanova
2. Curación de datos: Angel Alejandro Guerra Vilches, Madelín Haro Pérez, Yohandra Echavarria Castillo
3. Análisis formal: Angel Alejandro Guerra Vilches, Madelín Haro Pérez, Yohandra Echavarria Castillo
4. Investigación: Angel Alejandro Guerra Vilches, Madelín Haro Pérez, Yohandra Echavarria Castillo, Mónica Peña Casanova

5. Metodología: Madelín Haro Pérez, Yohandra Echavarria Castillo, Mónica Peña Casanova
6. Administración del proyecto:
7. Software: Angel Alejandro Guerra Vilches, Madelín Haro Pérez, Yohandra Echavarria Castillo, Mónica Peña Casanova
8. Supervisión: Mónica Peña Casanova
9. Validación: Angel Alejandro Guerra Vilches, Mónica Peña Casanova
10. Visualización: Angel Alejandro Guerra Vilches, Madelín Haro Pérez, Yohandra Echavarria Castillo, Mónica Peña Casanova
11. Redacción – borrador original: Angel Alejandro Guerra Vilches, Madelín Haro Pérez, Yohandra Echavarria Castillo, Mónica Peña Casanova
12. Redacción – revisión y edición: Angel Alejandro Guerra Vilches, Madelín Haro Pérez, Yohandra Echavarria Castillo, Mónica Peña Casanova

## Financiamiento

La investigación no requirió fuente de financiamiento externa.

## Referencias

Peña Casanova, M., Echevarría Castillo, Y., & Laborí de la Nuez, B. (2021). Arquitectura para la detección violaciones a políticas de seguridad. Revista Cubana de Ciencias Informáticas, Recuperado de https://rcci.uci.cu/?journal=rcci&page=article&op=view&path%5B%5D=2249

López Cruces, C. (2016). Diseño e implementación de una aplicación web para el análisis centralizado de trazas de seguridad. Tesis de grado.

Cigdem Bakir, V. H. (2020). Classifying database users for intrusion prediction and detection in data security. technical gazet. Tehnički vjesnik, Vol. 27 No. 6, 2020.https://doi.org/10.17559/TV-20190710100638

Toledo, A. Métodos de selección de atributos para clasificación supervisada basados en teoría de información. Trabajo de Diploma. Instituto Superior Politécnico “José Antonio Echeverría” Facultad de Ingeniería Informática, La Habana, Cuba, 2016

Cano Merchán, D. (2018). Análisis de trazas del sistema para la realización de un estudio sobre seguridad y comportamiento. Grado Universitario en Ingeniería Informática. Universidad Carlos III de Madrid.

Alonso-Alegre Díez, M. (2016). Gestión de trazas. Tesis de maestría. Máster Universitario en Seguridad Informática. Universidad Internacional de La Rioja.

González, A. C. (2015). Propuesta de Arquitectura Distribuida para la gestión de trazas. Grado Universitario en Ingeniería Informática. Universidad Carlos III de Madrid. https://e-archivo.uc3m.es/bitstream/handle/10016/22278/PFC_Abel_Cal_González.pdf

Guerra, Erick et al . Desarrollo de un sistema de gestión para la seguridad de la información basado en metodología de identificación y análisis de riesgo en bibliotecas universitarias. Inf. tecnol., La Serena , Vol. 32, No. 5, p. 145-156, oct. 2021 . Disponible en http://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0718-07642021000500145&lng=es&nrm=iso. accedido en 15 nov. 2024. http://dx.doi.org/10.4067/S0718-07642021000500145

Oliner, A., Ganapathi, A., & Xu, W. (2012). Advances and challenges in log analysis. Communications of the ACM, Vol. 55. No. 2, pp 55-61.

Shier, D. E. (2004). Well log normalization: methods and guidelines. Petrophysics-The SPWLA Journal of Formation Evaluation and Reservoir Description, Vol. 45, No. 03.

Akkurt, R., Miller, M., Hodenfield, B., Pirie, I., Farnan, D., & Koley, M. (2019, September). Machine learning for well log normalization. In SPE Annual Technical Conference and Exhibition? (p. D031S042R001). SPE. Vol.e 3,pp 157-161, ISSN 2666-5441, https://doi.org/10.1016/j.aiig.2022.11.004