OctoMatrix: Middleware De Seguridad Predictiva Para Aplicaciones Web Safe-by-Design

Santiago Potes Giraldo

Departamento de Ingeniería de Sistemas, Universidad Tecnológica de Pereira, Pereira, Colombia

s.potes@utp.edu.co

***Abstract*— Este artículo presenta OctoMatrix, un middleware de seguridad diseñado bajo el principio Safe by Design, orientado a la detección y mitigación temprana de vulnerabilidades web incluidas en el OWASP Top 10. OctoMatrix combina técnicas deterministas basadas en expresiones regulares con un modelo de aprendizaje automático entrenado sobre datasets sintéticos estilo Kaggle, logrando una precisión superior al 82%. Como caso de estudio se presenta su integración en Parchate Pereira, una plataforma de turismo sostenible desplegada en entornos de recursos limitados. Los resultados muestran que la arquitectura por capas permite sanitizar entradas, discriminar typos y neutralizar ataques antes de que alcancen la lógica de negocio o servicios de inteligencia artificial.**

I. Introducción

Las aplicaciones web modernas dependen cada vez más de flujos dinámicos de entrada, integraciones con servicios externos y componentes de inteligencia artificial. Este contexto amplía la superficie de ataque y hace insuficiente la validación tradicional de entradas. Vulnerabilidades como XSS, SQL Injection y XXE continúan apareciendo incluso en sistemas modernos.

En este trabajo se propone OctoMatrix, un middleware de seguridad predictiva que actúa como una capa intermedia entre el tráfico entrante y la lógica de aplicación. A diferencia de enfoques reactivos, OctoMatrix adopta una arquitectura preventiva, desacoplada y auditable.

II. Desarrollo de Contenidos

A. Arquitectura Generar del Sistema

El sistema se compone de dos elementos principales: la aplicación Parchate Pereira y el middleware OctoMatrix. La aplicación gestiona la interacción con usuarios y servicios, mientras que OctoMatrix intercepta y analiza los inputs antes de su procesamiento.

La arquitectura está organizada en capas de protección y predicción, permitiendo escalabilidad vertical y despliegue en entornos de bajo consumo.

B. Octomatrix como middleware de seguridad

OctoMatrix opera como un componente independiente que no modifica el estado de la aplicación. Su función es clasificar entradas como benignas o maliciosas y retornar un contexto de riesgo.

El middleware utiliza una estrategia de lazy loading para cargar el modelo entrenado únicamente cuando es requerido, reduciendo el tiempo de arranque y el consumo de memoria.

C. Pipeline de machine learning

El modelo de OctoMatrix es entrenado mediante un pipeline que combina:

1. Datos sintéticos de ataques OWASP Top 10
2. Patrones inspirados en CSIC 2010
3. Tráfico normal simulado

Se emplea un clasificador Random Forest junto con características TF-IDF y métricas avanzadas como entropía, densidad de símbolos y detección de patrones estructurales. El modelo entrenado es exportado como un archivo .pkl reutilizable.

D. Resultados Observados

```
Shell
Time
#
Log Message
9.2s   1       INICIANDO PIPELINE COMPLETO
DE SEGURIDAD
9.2s   2
========================================
==================
9.2s   3       Recopilando datos de
fuentes estilo Kaggle...
9.2s   4       Datos recopilados: 72
muestras
9.2s   5         - Ataques: 47
9.2s   6         - Normales: 25
9.2s   7       Procesando
características...
9.2s   8       Entrenando modelo...
9.6s   9       Modelo entrenado -
Precisión: 0.8667
9.6s   10
9.6s   11      Reporte de clasificación:
9.6s   12                    precision
```

```
recall  f1-score   support
9.6s   13
9.6s   14           Normal       1.00
0.60      0.75         5
9.6s   15           Ataque       0.83
1.00      0.91        10
9.6s   16
9.6s   17         accuracy
0.87        15
9.6s   18        macro avg       0.92
0.80      0.83        15
9.6s   19     weighted avg       0.89
0.87      0.86        15
9.6s   20
9.6s   21
9.6s   22      PRUEBA RÁPIDA DEL MODELO:
9.6s   23
----------------------------------------
---------
9.7s   24      ATAQUE | Confianza: 52.0% |
/api/users
9.7s   25      ATAQUE | Confianza: 93.0% |
/login
9.8s   26      ATAQUE | Confianza: 84.0% |
/search
9.9s   27      ATAQUE | Confianza: 87.0% |
/download
9.9s   28      Modelo exportado:
output/security_model.pkl
9.9s   29      Dataset exportado:
output/training_dataset.csv
9.9s   30      Paquete completo guardado
en: output/
9.9s   31
9.9s   32      PIPELINE COMPLETADO
EXITOSAMENTE
9.9s   33      Archivos generados en
carpeta 'output/':
9.9s   34        - security_model.pkl
(modelo entrenado)
9.9s   35        - training_dataset.csv
(datos de entrenamiento)
14.9s  36
/usr/local/lib/python3.11/dist-packages/t
raitlets/traitlets.py:2915:
FutureWarning:
--Exporter.preprocessors=["nbconvert.prep
rocessors.ExtractOutputPreprocessor"] for
containers is deprecated in traitlets
5.0. You can pass
`--Exporter.preprocessors item` ...
multiple times to add items to a list.
14.9s  37       warn(
14.9s  38     [NbConvertApp] Converting
notebook __script__.ipynb to html
16.7s  39     [NbConvertApp] Writing
323316 bytes to __results__.html
```

E. Caso De Estudio: Parchate Pereira

Parchate Pereira es una plataforma de turismo sostenible desplegada en Render. La integración de OctoMatrix permite que todos los inputs que alimentan motores de recomendación y servicios premium sean previamente sanitizados y clasificados.

Durante las pruebas de laboratorio, se observó una correcta discriminación de payloads maliciosos y typos, evitando que entradas anómalas alcancen parsers, bases de datos o servicios de IA externos

III. Futuros Pasos

Como trabajo futuro se plantea ampliar la cobertura de OctoMatrix hacia vulnerabilidades como SSRF, deserialización insegura y ataques de tipo XML Bomb. Adicionalmente, se propone incorporar análisis estructural profundo de payloads y un esquema de reentrenamiento supervisado que preserve la integridad del modelo.

A. Network identity aware correlation engine

Los trabajos futuros se orientan a expandir el modelo hacia una correlación más profunda entre identidad de red y semántica de eventos de aplicación. En particular, se propone el diseño de un motor de correlación consciente de identidad de red (network-identity–aware correlation engine), capaz de validar acciones TCP/IP observadas contra expectativas declaradas por sistemas de mensajería como Apache Kafka.

Una línea de investigación clave consiste en integrar identificadores de Capa 2 y Capa 3, tales como direcciones MAC e IP, mediante mecanismos de aprendizaje sticky, listas blancas y enlaces MAC–IP derivados de DHCP y ARP. Estos identificadores permitirían establecer una línea base de identidad de red para cada productor o consumidor autorizado.

Adicionalmente, se plantea observar metadatos de flujo TCP/IP —direcciones de origen y destino, puertos y patrones de conexión— sin inspección de carga útil. Dichos metadatos serían comparados con el modelo declarativo de intención definido por el sistema de mensajería, evaluando si el comportamiento de red concuerda con el rol esperado del componente.

Como extensión, el modelo podría integrarse con protocolos de gestión y monitoreo de red como SNMP, permitiendo incorporar métricas de interfaz, contadores y estadísticas de tráfico para enriquecer el proceso de correlación. Esta integración mantendría la independencia del protocolo de aplicación y evitaría impactos significativos en el rendimiento.

Finalmente, se considera la validación empírica del modelo en entornos multi-segmento y multi-inquilino, así como su aplicación en escenarios de detección de movimientos laterales, suplantación de identidad y ejecución no autorizada de roles. Estos pasos permitirán evaluar la viabilidad del

enfoque como complemento práctico a arquitecturas de seguridad y observabilidad existente

IV. Conclusiones

OctoMatrix demuestra que es posible integrar seguridad predictiva de forma ligera, desacoplada y eficiente en aplicaciones web modernas. La arquitectura por capas permite anticipar ataques comunes sin introducir complejidad innecesaria ni dependencias pesadas. El caso de estudio confirma que un enfoque Safe by Design mejora tanto la seguridad como la mantenibilidad del sistema.

Reconocimientos

Los autores agradecen a la comunidad académica y a los colaboradores del proyecto por el apoyo técnico y conceptual durante el desarrollo del sistema.

Referencias

[1] OWASP Foundation, "OWASP Top 10 Web Application Security Risks," 2021.

[2] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," IEEE Symposium on Security and Privacy, 2010.

[3] Scikit-learn Developers, "Random Forest Classifier Documentation," 2024.

[4] Impact Cyber Trust. Cybersecurity Datasets Repository: Dataset ID 940. Impact Cyber Trust, plataforma de datos abiertos para investigación en ciberseguridad. Disponible en: https://www.impactcybertrust.org/dataset_view?idDataset=940 (Consulta: diciembre de 2025).

[5] OWASP SpiderLabs. OWASP Core Rule Set (CRS) Regressions. Proyecto SpiderLabs, repositorio oficial de reglas de seguridad para detección de ataques web OWASP Top 10. Disponible en: https://github.com/SpiderLabs/OWASP-CRS-regressions (Consulta: diciembre de 2025).

[6] Ravi Kumar. HackOWASP 2025 – LB1 v1. Kaggle Notebook orientado al análisis y clasificación de ataques OWASP mediante técnicas de Machine Learning. Plataforma Kaggle. Disponible en: https://www.kaggle.com/code/ravi20076/hackowasp2025-lb1-v1 (Consulta: diciembre de 2025).

[7] Canadian Institute for Cybersecurity (CIC), University of New Brunswick. CIC-IDS2018 Dataset. Conjunto de datos para detección de intrusiones en redes, ampliamente utilizado en investigación académica en ciberseguridad. Disponible en: https://www.unb.ca/cic/datasets/ids-2018.html