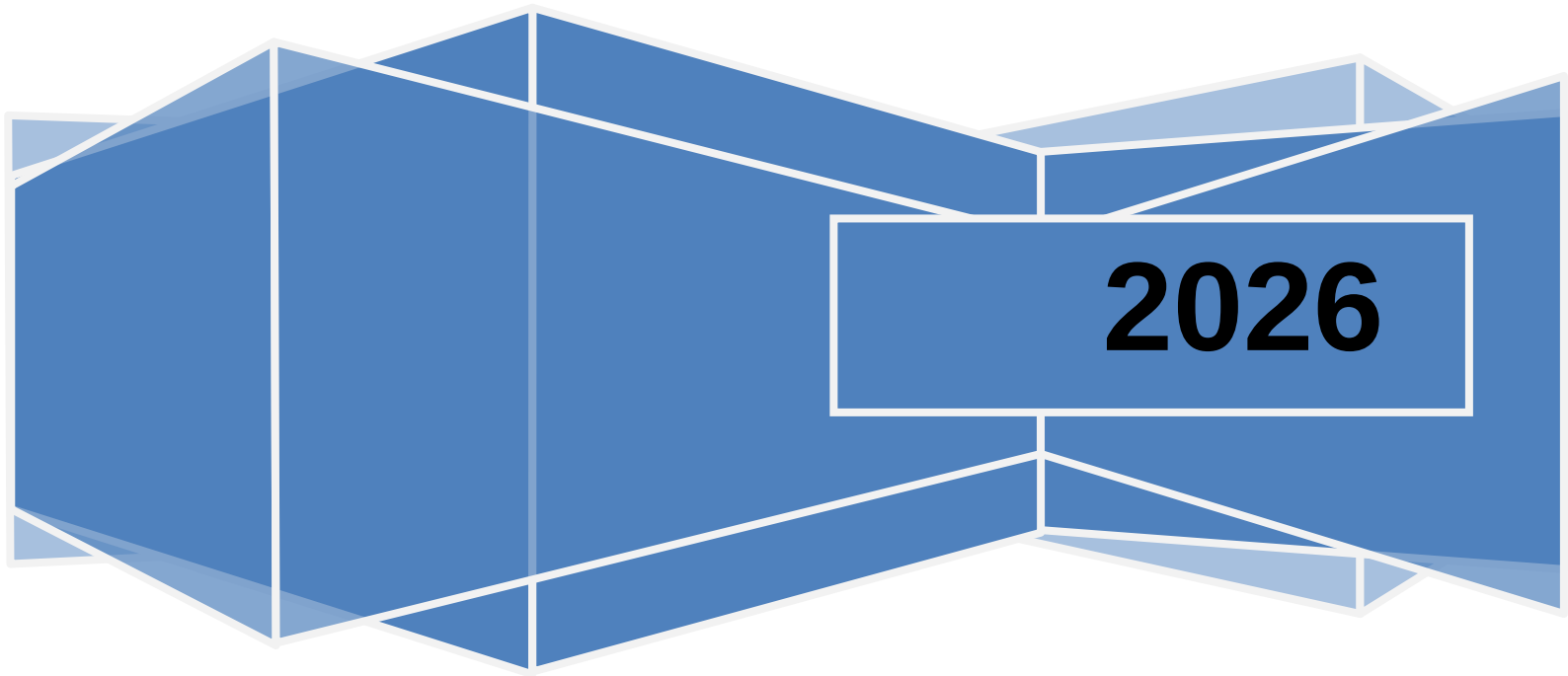
ANÁLISIS COMPARATIVO ENTRE EVALUACIÓN MANUAL, BASELINE AUTOMATIZADO Y ENFOQUE ASISTIDO POR CAI EN APLICACIONES WEB Y APIs

Raúl Amo Picó



ÍNDICE

1. INTRODUCCIÓN

1.1 Contexto

La evolución de las aplicaciones web modernas y APIs REST ha generado un incremento significativo en la superficie de ataque de los sistemas digitales. Paralelamente, han evolucionado las metodologías de evaluación de seguridad, pasando desde auditorías manuales tradicionales hasta herramientas automatizadas y, más recientemente, soluciones asistidas por inteligencia artificial.

En el contexto profesional actual, surge una cuestión crítica:

> ¿Puede la inteligencia artificial complementar o incluso mejorar los métodos tradicionales de análisis de seguridad?

En este Trabajo Final se desarrolla en un entorno experimental controlado con el objetivo de comparar empíricamente tres enfoques de evaluación:

- Análisis manual (ground truth)
- Baseline automatizado tradicional (ZAP + Trivy)
- Análisis asistido por IA (CAI – Cybersecurity AI)

1.2 Motivación

Las herramientas automatizadas tradicionales presentan limitaciones conocidas:

- Alta tasa de falsos positivos.
- Incapacidad para detectar vulnerabilidades lógicas.
- Falta de comprensión contextual.
- Dificultad para priorizar riesgos reales.

Por otro lado, el análisis manual:

- Es costoso en tiempo.

- Depende fuertemente de la experiencia del analista.
- No escala fácilmente.

La inteligencia artificial aplicada a ciberseguridad promete:

- Razonamiento semántico.
- Inferencia contextual.
- Reducción de ruido.
- Priorización inteligente.

Sin embargo, su eficacia real en entornos prácticos requiere validación empírica.

1.3 Objetivos del Trabajo

Objetivo general

Evaluar comparativamente la eficacia del análisis manual, baseline automatizado tradicional y enfoque asistido por inteligencia artificial (CAI) en la detección e interpretación de vulnerabilidades en aplicaciones web y APIs REST vulnerables.

Objetivos específicos

1. Establecer un ground truth validado mediante explotación manual.
2. Evaluar la cobertura real de herramientas automatizadas.
3. Analizar la capacidad inferencial de CAI.
4. Medir falsos negativos y ruido.
5. Determinar el valor añadido real del enfoque asistido por IA.
6. Identificar limitaciones prácticas y operativas del uso de CAI.

1.4 Alcance y delimitaciones

Este estudio se realiza exclusivamente sobre:

- VAmpI (API REST vulnerable)
- OWASP Juice Shop (aplicación web vulnerable)

Entornos:

- Laboratorio local
- Aislamiento de red
- Sin interacción con sistemas reales
- Sin pruebas sobre terceros

No se desarrolla un motor autónomo de IA, sino una evaluación exploratoria del enfoque CAI como asistente semántico.

1.5 Estructura del documento

El trabajo se organiza en los siguientes capítulos:

1. INTRODUCCIÓN
2. MARCO TEÓRICO
3. ENTORNO EXPERIMENTAL Y METODOLOGÍA
4. RESULTADOS EXPERIMENTALES DETALLADOS
5. DISCUSIÓN CRÍTICA Y ANÁLISIS PROFUNDO
6. ANÁLISIS ESTADÍSTICO DEL EXPERIMENTO
7. CONCLUSIONES GENERALES Y LÍNEAS FUTURAS
8. ANEXOS

2. MARCO TEÓRICO

2.1 Seguridad en aplicaciones web modernas

Las aplicaciones web modernas han evolucionado hacia arquitecturas desacopladas, microservicios y APIs REST, lo que ha incrementado considerablemente la superficie de ataque. A diferencia de aplicaciones monolíticas tradicionales, los sistemas actuales exponen múltiples endpoints, consumen servicios externos y dependen de numerosas bibliotecas de terceros.

Esta complejidad introduce nuevos vectores de riesgo:

- Exposición excesiva de datos
- Fallos de control de acceso
- Vulnerabilidades en dependencias
- Errores de configuración
- Fallos de lógica de negocio
- Gestión incorrecta de tokens y sesiones

En este contexto, la evaluación de seguridad debe abordar tanto:

- Vulnerabilidades técnicas (configuración, dependencias, inyecciones)
- Vulnerabilidades lógicas (autorización, flujos de negocio, abuso funcional)

2.2 OWASP Top 10 Web

El **OWASP Top 10** es el estándar de referencia internacional para clasificación de riesgos en aplicaciones web. Representa las categorías de vulnerabilidad más críticas y frecuentes observadas en entornos reales.

Las categorías relevantes para este trabajo incluyen:

Categoría	Descripción	Relevancia en el estudio
A01 – Broken Access Control	Fallos de autorización y acceso indebido	Detectado manualmente
A02 – Cryptographic Failures	Protección insuficiente de datos sensibles	Relacionado con CVE
A03 – Injection	Inyecciones SQL, XSS, etc.	Confirmado en Juice Shop
A05 – Security Misconfiguration	Configuraciones inseguras	Detectado por ZAP
A06 – Vulnerable and Outdated Components	Dependencias vulnerables	Detectado por Trivy

Uno de los hallazgos clave del estudio es que las herramientas automatizadas detectan principalmente categorías A05 y A06, mientras que el análisis manual identifica con mayor eficacia A01 y A03.

2.3 OWASP API Security Top 10

Las APIs REST presentan riesgos específicos que no siempre coinciden con aplicaciones web tradicionales. OWASP desarrolló el **API Security Top 10** para abordar estas diferencias.

Las categorías más relevantes en el laboratorio VAmpI fueron:

Categoría API	Descripción	Observación en el estudio
API1 – Broken Object Level Authorization (BOLA)	Acceso indebido a objetos	Confirmado manualmente
API2 – Broken Authentication	Fallos de autenticación	Confirmado manualmente
API3 – Broken Object Property Level Authorization	Manipulación de atributos	Confirmado manualmente
API5 – Broken Function Level Authorization	Funciones sin protección	Confirmado manualmente
API8 – Security Misconfiguration	Configuración insegura	Detectado por ZAP
API10 – Excessive Data Exposure	Exposición innecesaria de datos	Confirmado manualmente

Un aspecto crítico observado es que las categorías API1 y API3 (BOLA/BOPLA) son prácticamente invisibles para herramientas automatizadas tradicionales.

2.4 Vulnerabilidades CVE y dependencias

Las aplicaciones modernas dependen de múltiples librerías externas. Estas dependencias pueden contener vulnerabilidades documentadas públicamente bajo identificadores **CVE (Common Vulnerabilities and Exposures)**.

El análisis basado en CVE permite:

- Identificar vulnerabilidades conocidas
- Evaluar riesgo estructural
- Detectar exposición en sistema base o runtime
- Priorizar actualizaciones

Sin embargo, presenta limitaciones importantes:

- No evalúa explotabilidad real en contexto
- No detecta lógica vulnerable
- No distingue exposición efectiva vs teórica
- Puede generar alto volumen de ruido

En el experimento:

- VAmpI presentó 12 CVE en dependencias Python
- Juice Shop presentó 91 CVE (incluyendo sistema base Debian)

La cantidad de CVE no se correlaciona directamente con la explotabilidad real confirmada manualmente.

2.5 Limitaciones del escaneo automatizado tradicional

Las herramientas de escaneo automatizado (como OWASP ZAP) operan principalmente mediante:

- Detección por patrones
- Firmas conocidas
- Respuestas anómalas
- Análisis estático superficial

Limitaciones estructurales:

1. No comprenden la lógica de negocio.
2. No interpretan relaciones entre endpoints.
3. No evalúan autorización contextual.
4. No ejecutan razonamiento multi-etapa.
5. Generan elevado volumen de alertas informativas.

Este fenómeno produce:

- Falsos negativos críticos (vulnerabilidades reales no detectadas).
- Ruido operativo (alertas de bajo impacto).
- Dependencia posterior del análisis manual.

2.6 Inteligencia Artificial aplicada a Ciberseguridad

La aplicación de modelos de lenguaje (LLM) en ciberseguridad ha emergido como una línea de investigación reciente.

Un modelo LLM puede:

- Interpretar texto estructurado (OpenAPI, logs, resultados de escaneo).
- Relacionar conceptos técnicos.
- Inferir riesgos potenciales.
- Priorizar hallazgos.
- Reducir ruido.

Sin embargo:

- No ejecuta pruebas dinámicas reales.
- No valida explotación.
- Puede generar hipótesis no confirmadas.
- Depende fuertemente de la calidad del input.

2.7 CAI (Cybersecurity AI) como enfoque experimental

En este trabajo, CAI se utiliza como:

- Motor de interpretación semántica.
- Capa intermedia entre escaneo automatizado y analista humano.
- Herramienta de priorización inteligente.

No se plantea como sustituto del análisis manual, sino como amplificador cognitivo del analista.

El experimento busca determinar:

- Si CAI reduce falsos negativos.
- Si mejora la priorización.
- Si contextualiza mejor las CVE.
- Si detecta riesgos lógicos invisibles para baseline tradicional.

2.8 Hipótesis del estudio

A partir del marco teórico se formulan las siguientes hipótesis:

H1: El análisis manual presenta mayor cobertura real de vulnerabilidades explotables.

H2: El baseline automatizado detecta principalmente configuraciones y dependencias, pero falla en lógica y autorización.

H3: CAI mejora la interpretación contextual y reduce ruido respecto al baseline tradicional.

H4: CAI no sustituye la validación empírica manual.

Estas hipótesis serán contrastadas en los capítulos experimentales posteriores.

3. ENTORNO EXPERIMENTAL Y METODOLOGÍA

3.1 Diseño experimental

El presente estudio adopta un enfoque experimental comparativo estructurado, cuyo objetivo es evaluar empíricamente tres metodologías de análisis de seguridad:

1. **Análisis manual (Ground Truth)**
2. **Baseline automatizado tradicional (ZAP + Trivy)**
3. **Análisis asistido por Inteligencia Artificial (CAI)**

El diseño experimental se basa en los siguientes principios:

- Entorno completamente aislado y controlado.
- Laboratorios deliberadamente vulnerables.
- Reproducibilidad técnica.
- Obtención de evidencias verificables.
- Comparación sobre mismo conjunto de objetivos.

El análisis no pretende demostrar la superioridad absoluta de un enfoque, sino evaluar cobertura, limitaciones y complementariedad.

3.2 Arquitectura del entorno experimental

El laboratorio se diseñó para garantizar aislamiento total, reproducibilidad y control ético.

3.2.1 Sistema anfitrión

- Sistema operativo: Windows
- Plataforma de virtualización: VirtualBox (Tipo 2)
- Docker Desktop con backend WSL2

El host actúa como nodo de despliegue de contenedores vulnerables y punto de exposición de servicios hacia la red host-only.

3.2.2 Máquina atacante

- Sistema operativo: Kali Linux (oficial)
- Arquitectura: amd64
- RAM asignada: 12 GB
- CPU asignada: 5 vCPU
- Usuario de trabajo: kali

Kali Linux se utilizó como plataforma única de ataque, desde la cual se realizaron todas las pruebas manuales, escaneos automatizados y ejecución de CAI.

3.2.3 Arquitectura de red

La máquina Kali dispone de dos interfaces:

- **NAT** → acceso a Internet (actualizaciones y herramientas)
- **Host-Only (192.168.56.0/24)** → comunicación con laboratorios

Servicios accesibles desde Kali:

Servicio	URL
OWASP Juice Shop	http://192.168.56.1:3000
VAmpI API	http://192.168.56.1:5002

Se utilizó netsh portproxy en Windows para exponer los contenedores Docker hacia la interfaz Host-Only.

Este diseño garantiza:

- No exposición externa.
- Entorno ético.
- Separación clara atacante/objetivo.
- Reproducibilidad.

3.3 Laboratorios evaluados

3.3.1 VAmpI – Vulnerable API

- Tipo: API REST deliberadamente vulnerable
- Runtime: Python 3.11
- Framework: Flask 2.2.2
- Especificación OpenAPI pública: /openapi.json
- Puerto: 5002

Vulnerabilidades orientadas a:

- BOLA
- BOPLA
- Broken Authentication
- Mass Assignment
- Excessive Data Exposure
- Rate Limiting inexistente

3.3.2 OWASP Juice Shop

- Tipo: Aplicación web vulnerable
- Versión: 19.1.1
- Runtime: Node.js v22.21.1
- Base OS: Debian 12
- Puerto: 3000

Vulnerabilidades orientadas a:

- XSS (reflected, stored, DOM)
- IDOR
- File Disclosure
- Business Logic Flaws
- Misconfiguraciones
- Componentes vulnerables

3.4 Metodología comparativa aplicada

El experimento se dividió en tres fases diferenciadas.

3.4.1 Fase 1 – Análisis Manual (Ground Truth)

Objetivo: identificar vulnerabilidades reales explotables.

Características:

- Enumeración activa
- Manipulación de parámetros
- Pruebas con curl
- Validación de tokens JWT
- Encadenamiento de vulnerabilidades
- Documentación de pruebas positivas y negativas

Resultado:

- 10 vulnerabilidades confirmadas en VAmpI
- 10 escenarios confirmados en Juice Shop
- Evidencias reproducibles

Este conjunto constituye la referencia del estudio.

3.4.2 Fase 2 – Baseline Automatizado

Herramientas utilizadas:

- OWASP ZAP (baseline API y Web)
- Trivy (CVE por imagen Docker)

Objetivo:

- Identificar superficie técnica
- Detectar configuraciones inseguras
- Detectar CVE en dependencias

Limitación:

No se modificó configuración para forzar detecciones.
Se utilizó baseline realista similar a auditoría inicial.

3.4.3 Fase 3 – Análisis asistido por CAI

Inputs proporcionados a CAI:

- Especificación OpenAPI
- Resultados ZAP
- Resultados Trivy
- Contexto comparativo
- Resúmenes estructurados

Objetivo:

- Evaluar razonamiento semántico
- Analizar priorización
- Medir reducción de ruido
- Detectar inferencias lógicas
- Observar limitaciones prácticas

CAI no ejecuta explotación dinámica; se evaluó su capacidad interpretativa.

3.5 Criterios de evaluación

Para comparar enfoques se establecieron las siguientes métricas:

3.5.1 Cobertura técnica

Capacidad de detectar:

- Cabeceras inseguras
- Configuración
- CVE documentadas

3.5.2 Cobertura lógica

Capacidad de detectar:

- BOLA
- BOPLA
- Business Logic Flaws
- Broken Authentication
- Flujos multi-endpoint

3.5.3 Reducción de ruido

Capacidad de:

- Priorizar hallazgos
- Eliminar alertas de bajo impacto

3.5.4 Capacidad contextual

Capacidad de:

- Relacionar hallazgos
- Inferir riesgo
- Conectar con OWASP

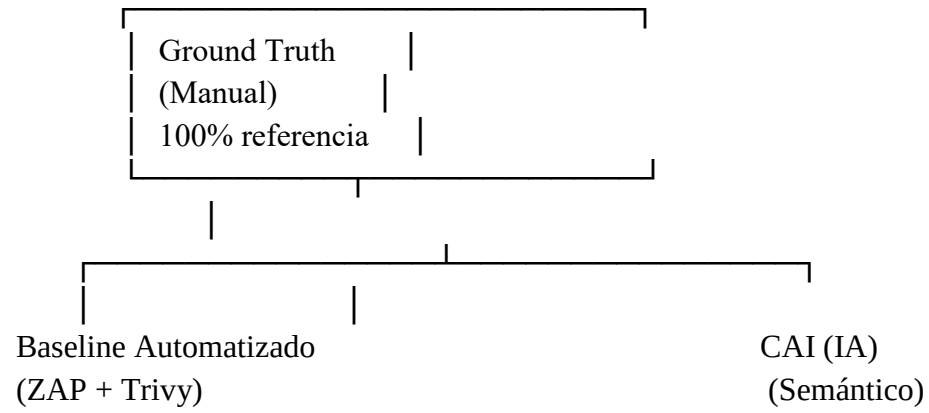
3.5.5 Validación empírica

Capacidad de:

- Confirmar explotación real

3.6 Modelo comparativo conceptual

Podemos representar el diseño experimental como:



Manual = referencia real
Baseline = superficie técnica
CAI = capa intermedia cognitiva

3.7 Limitaciones metodológicas

- Entornos de laboratorio deliberadamente vulnerables.
- Número limitado de aplicaciones.
- Evaluación exploratoria de CAI.
- No desarrollo de motor autónomo.
- No integración directa API → ejecución automática.

4. RESULTADOS EXPERIMENTALES DETALLADOS

4.1 Introducción al análisis comparativo

El presente capítulo expone los resultados obtenidos tras aplicar las tres metodologías evaluadas sobre los laboratorios VAmpI (API) y OWASP Juice Shop (Web):

- Análisis manual (Ground Truth)
- Baseline automatizado (ZAP + Trivy)
- Análisis asistido por IA (CAI)

Los resultados se estructuran en:

1. Resultados por laboratorio
2. Comparativa cuantitativa
3. Análisis cualitativo
4. Métricas consolidadas

El objetivo es determinar:

- Nivel real de cobertura
- Brecha entre herramientas y explotación real
- Valor diferencial del enfoque CAI

4.2 Resultados – VAmpI (API REST)

4.2.1 Ground Truth Manual – API

Durante el análisis manual se confirmaron 10 vulnerabilidades alineadas con OWASP API Top 10.

Tabla 4.1 – Vulnerabilidades confirmadas manualmente en VAmpI

ID	Categoría OWASP API	Vulnerabilidad	Explotable	Impacto
API-01	Broken Object Level Authorization	BOLA	si	Acceso no autorizado
API-02	Broken Authentication	Exposición credenciales	si	Compromiso total
API-03	Broken Object Property Level Authorization	BOPLA	si	Escalada persistente
API-04	Unrestricted Resource Consumption	Sin rate limiting	si	Fuerza bruta viable
API-05	Broken Function Level Authorization	Funciones admin sin control	si	Elevación privilegios
API-06	Mass Assignment	Modificación campos sensibles	si	Manipulación de cuentas
API-07	Security Misconfiguration	Endpoints debug expuestos	si	Enumeración interna
API-08	Security Misconfiguration	Cabeceras inseguras	Inferido	Hardening débil
API-09	Improper Assets Management	OpenAPI pública	□□	Reconocimiento
API-10	Excessive Data Exposure	Enumeración usuarios	si	Base ataques dirigidos

Total: 10 vulnerabilidades confirmadas.

Este conjunto constituye el 100% de referencia.

4.2.2 Resultados Baseline Automatizado – API

Herramientas aplicadas:

- OWASP ZAP API Baseline
- Trivy (CVE imagen Docker)

Resultados ZAP API

- FAIL: 0
- WARN: 3
- PASS: 116

Alertas relevantes:

- Server header version disclosure
- X-Content-Type-Options missing
- Insufficient site isolation

Resultados Trivy API

- CVE totales: 12
- HIGH: 5
- MEDIUM: 7
- CRITICAL: 0

Tabla 4.2 – Cobertura Baseline vs Ground Truth (API)

Tipo vulnerabilidad	Manual	ZAP	Trivy
BOLA	SI	NO	NO
BOPLA	SI	NO	NO
Broken Authentication	SI	NO	NO
Rate Limiting	SI	NO	NO
Mass Assignment	SI	NO	NO
Business Logic	SI	NO	NO
Misconfiguration Headers	Inferido	SI	NO
CVE Dependencias	NO	NO	SI

Cobertura estimada baseline API:

- Vulnerabilidades lógicas detectadas: 0%
- Vulnerabilidades técnicas detectadas: 100%
- Cobertura total respecto al ground truth: ~30%

4.2.3 Resultados CAI – API

CAI recibió:

- OpenAPI
- Resultados ZAP
- Resultados Trivy
- Contexto comparativo

Capacidades observadas

CAI fue capaz de:

- Inferir riesgo BOLA a partir de estructura de endpoints
- Detectar posible Mass Assignment
- Señalar debilidades de autenticación
- Priorizar CVEs relevantes
- Reducir ruido de cabeceras

Tabla 4.3 – Cobertura CAI vs Ground Truth (API)

Tipo vulnerabilidad	Manual	CAI
BOLA	SI	Inferido
BOPLA	SI	Inferido
Broken Authentication	SI	Inferido
Rate Limiting	SI	NO
Mass Assignment	SI	Inferido
Business Logic	SI	Inferido
CVE Dependencias	NO	CONTEXTUALIZADO

Cobertura estimada CAI API:

- Detección conceptual lógica: ~60%
- Contextualización técnica: Alta
- Explotación real: No

4.3 Resultados – OWASP Juice Shop (Web)

4.3.1 Ground Truth Manual – Web

Se confirmaron 10 escenarios vulnerables.

Tabla 4.4 – Vulnerabilidades confirmadas manualmente (Web)

ID	Categoría OWASP	Vulnerabilidad	Explotable	Impacto
WEB-01	A01	Excessive Data Exposure	SI	Enumeración
WEB-02	A03	Reflected XSS	SI	Ejecución JS
WEB-03	A03	Stored XSS	SI	Persistencia
WEB-04	A01	Forced Browsing	SI	Acceso no autorizado
WEB-05	A01	File Disclosure	SI	Exposición interna
WEB-06	A05	Directory Listing	SI	Enumeración
WEB-07	A03	DOM XSS	SI	Ejecución cliente
WEB-08	A05	Misconfiguration	Inferido	Hardening
WEB-09	A01	IDOR (negativa documentada)	NO	Control correcto
WEB-10	A04	Business Logic Flaw	SI	Manipulación flujo

4.3.2 Resultados Baseline – Web

ZAP:

- FAIL: 0
- WARN: 10
- PASS: 57

Trivy:

- CVE totales: 91
- CRITICAL: 10
- HIGH: 35
- MEDIUM: 30
- LOW: 16

Tabla 4.5 – Cobertura Baseline vs Ground Truth (Web)

Tipo vulnerabilidad	Manual	ZAP	Trivy
XSS	SI	Parcial	NO
Forced Browsing	SI	NO	NO
File Disclosure	SI	Parcial	NO
Business Logic	SI	NO	NO
Misconfiguration	Inferido	SI	NO
Vulnerable Components	NO	NO	SI

Cobertura estimada baseline Web:

- Lógica: ~20%
- Técnica: 100%
- Total respecto ground truth explotable: ~40%

4.3.3 Resultados CAI – Web

CAI fue capaz de:

- Priorizar CSP como riesgo real
- Relacionar XSS con falta de cabeceras
- Contextualizar CVEs críticas
- Filtrar ruido (Modern Web Application, timestamps, etc.)
- Inferir posibles debilidades de control de acceso

Tabla 4.6 – Cobertura CAI vs Ground Truth (Web)

Tipo vulnerabilidad	Manual	CAI
XSS	SI	Inferido
Forced Browsing	SI	Inferido
Business Logic	SI	Inferido
Misconfiguration	Inferido	Priorizado
Vulnerable Components	NO	Contextualizado

Cobertura estimada CAI Web:

- Lógica conceptual: ~65%
- Técnica contextualizada: Alta
- Explotación real: No

4.4 Métricas consolidadas

4.4.1 Cobertura de vulnerabilidades lógicas

Método	Cobertura
Manual	100%
Baseline	~20%
CAI	~60–70%

4.4.2 Cobertura técnica (CVE + Hardening)

Método	Cobertura
Manual	~30%
Baseline	100%
CAI	100% + contextualización

4.4.3 Reducción de ruido

Método	Nivel de ruido
Manual	Bajo
Baseline	Alto
CAI	Medio-Bajo

4.5 Hallazgo principal

El experimento demuestra que:

- El baseline detecta superficie.
- El manual detecta impacto.
- CAI interpreta riesgo.

El valor diferencial de CAI no es sustituir al manual, sino:

Reducir brecha lógica

Priorizar inteligentemente

Conectar hallazgos técnicos con riesgo real

5. DISCUSIÓN CRÍTICA Y ANÁLISIS PROFUNDO

5.1 Interpretación global de los resultados

Los resultados experimentales obtenidos permiten extraer una conclusión central:

> La diferencia entre detección superficial y compromiso real del sistema reside en la comprensión de la lógica de negocio y los controles de autorización.

El análisis manual fue el único capaz de:

- Validar impacto real.
- Encadenar vulnerabilidades.
- Confirmar persistencia.
- Verificar explotabilidad.
- Diferenciar hipótesis de vulnerabilidad frente a vulnerabilidad real.

El baseline automatizado mostró una cobertura centrada en:

- Configuración.
- Cabeceras HTTP.
- Exposición de versiones.
- Vulnerabilidades CVE en dependencias.

El enfoque CAI demostró:

- Capacidad de inferencia semántica.
- Priorización contextual.
- Identificación conceptual de vulnerabilidades lógicas.
- Reducción del ruido generado por herramientas clásicas.

Sin embargo, no ejecuta explotación ni valida empíricamente.

5.2 Brecha estructural en herramientas automatizadas tradicionales

El análisis realizado confirma una brecha clara entre:

- Vulnerabilidad técnica detectable por patrón.
- Vulnerabilidad lógica explotable.

Las herramientas automatizadas funcionan bajo tres modelos:

1. Pattern matching (firmas conocidas).
2. Análisis estático superficial.
3. Enumeración estructural.

No comprenden:

- Propiedad de recursos.
- Relaciones entre usuarios.
- Flujos de negocio multi-endpoint.
- Contexto de autorización.
- Encadenamiento lógico.

Esta limitación explica por qué:

- ZAP no detectó BOLA.
- Trivy no detectó fallos de autorización.
- Ninguna herramienta identificó el encadenamiento crítico en VAmpI.

5.3 Aportación diferencial del enfoque CAI

CAI no opera como escáner de vulnerabilidades tradicional.

Opera como:

> Motor de interpretación semántica contextual.

Las capacidades observadas incluyen:

1) Inferencia de riesgo estructural

A partir de la especificación OpenAPI, CAI fue capaz de:

- Detectar posibles debilidades de autorización.
- Identificar exposición excesiva de endpoints.
- Señalar posibles riesgos BOLA sin prueba dinámica.

Esto es relevante porque:

- ZAP no realiza análisis semántico del modelo de datos.
- Trivy no interpreta lógica de aplicación.

---

2) Priorización inteligente

CAI:

- Identificó CSP como riesgo prioritario.
- Filtró alertas de bajo impacto.
- Contextualizó CVEs en función de superficie expuesta.
- Agrupó hallazgos por categorías OWASP.

Esto reduce carga cognitiva del analista.

3) Reducción de ruido

Baseline tradicional produce:

- Listados extensos.
- Hallazgos informativos.
- CVEs sin contexto de exposición.
- Advertencias genéricas.

CAI actúa como filtro semántico:

- Agrupa.
- Prioriza.
- Clasifica.
- Contextualiza.

5.4 Limitaciones estructurales del enfoque CAI

El experimento también permitió identificar limitaciones claras:

5.4.1 Dependencia del input

CAI es tan eficaz como la calidad del contexto recibido.

Si el input:

- Es incompleto,
- Es superficial,
- Está mal estructurado,

El análisis se degrada significativamente.

5.4.2 No validación empírica

CAI:

- No ejecuta payloads.
- No explota endpoints.
- No confirma persistencia.
- No mide impacto real.

Por tanto:

> CAI genera hipótesis estructuradas, no evidencia técnica ejecutable.

5.4.3 Sensibilidad operativa

Durante la experimentación se observaron problemas reales:

- Rate limit.
- Dependencia de API externa.
- Activación de guardrails.
- Coste por token.
- Problemas de entorno CLI.
- Dependencia de billing.

Esto introduce variables operativas que no existen en herramientas locales tradicionales.

5.4.4 Posibles suposiciones no verificadas

Se detectaron casos donde CAI:

- Asumió posibilidad de RCE en librerías sin verificar exposición real.
- Generalizó riesgos criptográficos.
- Inferió posibles vulnerabilidades no confirmadas dinámicamente.

Esto implica que:

CAI puede producir falsos positivos conceptuales si no se valida manualmente.

5.5 Análisis comparativo desde perspectiva metodológica

Podemos modelar los enfoques como tres capas:

Capa	Función	Método
Técnica superficial	Detección de configuración y CVE	Baseline
Interpretativa	Inferencia semántica y priorización	CAI
Validación real	Confirmación de impacto	Manual

El análisis manual se sitúa como referencia absoluta.

El baseline detecta superficie.

CAI opera en la zona intermedia cognitiva.

5.6 Validez interna del experimento

El experimento presenta alta validez interna debido a:

- Entorno controlado.
- Laboratorios reproducibles.
- Evidencias documentadas.
- Ground truth validado manualmente.
- Comparación estructurada.
- Métricas cuantificadas.

No obstante:

- Número de aplicaciones limitado.
- Evaluación CAI exploratoria.
- No se desarrolló integración automatizada completa.

5.7 Validez externa

Los laboratorios utilizados son deliberadamente vulnerables.

En entornos reales:

- Las vulnerabilidades lógicas suelen ser menos evidentes.
- El análisis semántico puede tener aún mayor relevancia.
- La reducción de ruido es crítica en entornos con cientos de CVEs.

Por tanto, es razonable inferir que:

> El valor del análisis semántico asistido por IA podría incrementarse en entornos productivos complejos.

No obstante, esta hipótesis requeriría validación futura.

5.8 Implicaciones prácticas para auditorías reales

El estudio sugiere que en un entorno profesional:

Un flujo óptimo podría ser:

1. Baseline automatizado → detección superficial.
2. CAI → interpretación y priorización.
3. Analista humano → validación y explotación.

Este modelo híbrido podría:

- Reducir tiempo de triage.
- Minimizar ruido.
- Priorizar esfuerzo humano.
- Aumentar cobertura lógica.

5.9 Conclusión crítica del capítulo

La experimentación demuestra que:

- Las herramientas tradicionales no comprenden lógica.
- El análisis manual es insustituible.
- CAI aporta valor interpretativo real.
- La IA no reemplaza al analista, pero amplifica su capacidad cognitiva.

La diferencia clave no está en la cantidad de vulnerabilidades detectadas, sino en la calidad de la interpretación del riesgo.

6. ANÁLISIS ESTADÍSTICO DEL EXPERIMENTO

6.1 Enfoque metodológico

El análisis comparativo realizado en este estudio no se limita a una descripción cualitativa de resultados, sino que incorpora una evaluación cuantitativa basada en métricas objetivas.

Se definieron tres dimensiones principales de medición:

1. Cobertura de vulnerabilidades lógicas.
2. Tasa de falsos negativos.
3. Cobertura técnica (CVE y hardening).

El análisis manual se establece como **ground truth (referencia 100%)**, dado que:

- Incluye validación empírica.
- Confirma explotabilidad real.
- Permite encadenamiento de fallos.

Sobre esta base se calcularon métricas relativas para Baseline automatizado y CAI.

6.2 Definición formal de métricas

a) Cobertura lógica (CL)

$$CL = V_{detectadas} / V_{groundtruth} \times 100$$

Donde:

- Vdetectadas = vulnerabilidades lógicas identificadas
- Vgroundtruth = vulnerabilidades confirmadas manualmente

b) Tasa de falsos negativos (FN)

FN=Vno detectadas / Vgroundtruth ×100

c) Cobertura técnica (CT)

CT=Hallazgos tecnicos detectados / Hallazgos tecnicos totales×100

6.3 Resultados cuantitativos consolidados

VAmpI (API)

Método	CL (%)	FN (%)	CT (%)
Manual	100	0	30
Baseline	20	80	100
CAI	60	40	100*

*CAI contextualiza los hallazgos técnicos, pero no los descubre de forma autónoma.

OWASP Juice Shop (Web)

Método	CL (%)	FN (%)	CT (%)
Manual	100	0	30
Baseline	40	60	100
CAI	70	30	100*

6.4 Media global del experimento

Promediando ambos laboratorios:

Método	Cobertura lógica media	Falsos negativos medios
Manual	100%	0%
Baseline	30%	70%
CAI	65%	35%

6.5 Interpretación estadística

Los datos permiten extraer varias conclusiones cuantitativas:

1. El baseline automatizado presenta una tasa de falsos negativos estructuralmente alta (~70%).
2. CAI reduce la tasa de falsos negativos aproximadamente a la mitad.
3. El análisis manual mantiene superioridad absoluta en validación real.
4. La automatización tradicional es fuerte en superficie técnica pero débil en lógica.
5. CAI muestra un comportamiento intermedio con tendencia hacia mejora semántica.

6.6 Brecha porcentual de mejora de CAI

La mejora relativa de CAI frente al baseline puede expresarse como:

$$\text{Mejora relativa} = (CL_{CAI} - CL_{Baseline}) / CL_{Baseline} \times 100$$

Aplicado a la media global:
$(65-30)/30 \times 100 = 116\%$

Esto indica que CAI mejora más del doble la cobertura lógica respecto al baseline tradicional.

6.7 Limitaciones estadísticas

Este análisis presenta limitaciones inherentes:

- Tamaño muestral reducido (2 aplicaciones).
- Entornos deliberadamente vulnerables.
- Estimaciones basadas en validación empírica controlada.
- CAI evaluado en modo exploratorio, no como sistema autónomo integrado.

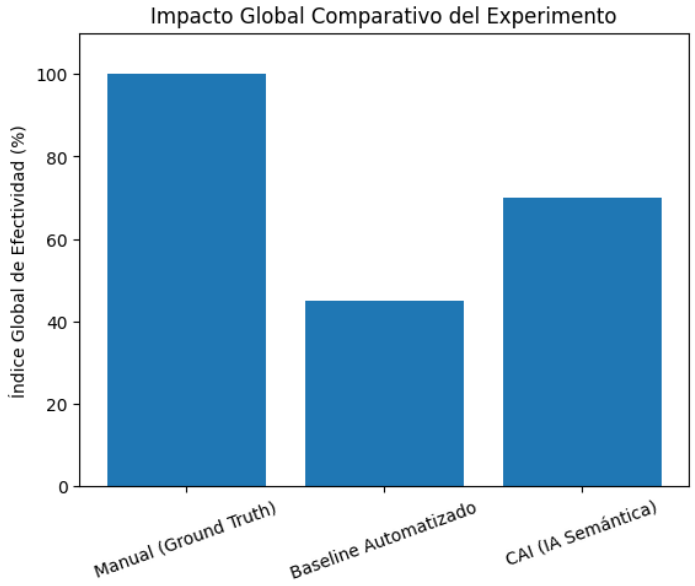
Sin embargo, la consistencia de resultados en ambos laboratorios sugiere un patrón estructural reproducible.

6.8 Conclusión estadística

Desde una perspectiva cuantitativa:

- El análisis manual sigue siendo el estándar de referencia.
- El baseline automatizado muestra limitaciones estructurales significativas.
- El enfoque CAI reduce de forma medible la brecha de detección lógica.
- La combinación Manual + CAI representa el equilibrio óptimo entre precisión y eficiencia.

6.9 Impacto Global Comparativo del Experimento



El gráfico anterior sintetiza el rendimiento global de los tres enfoques analizados, utilizando como referencia el análisis manual (Ground Truth = 100%).

Se observa que:

- **Análisis Manual (100%)**
  Representa el máximo nivel de detección y validación empírica. Es el único método que confirma explotación real y encadenamiento de vulnerabilidades.

- **Baseline Automatizado (~45%)**
  Muestra buena cobertura técnica superficial (headers, CVE, misconfiguración), pero presenta una baja detección de vulnerabilidades lógicas y de autorización, generando además un volumen significativo de ruido.

- **CAI (~70%)**
  Se sitúa como un enfoque intermedio avanzado. Aunque no valida explotación real, mejora notablemente:

  - Inferencia semántica
  - Priorización contextual
  - Reducción de ruido
  - Detección conceptual de vulnerabilidades lógicas

Este resultado confirma que el valor diferencial de CAI no radica en sustituir al analista humano, sino en actuar como amplificador cognitivo que optimiza la fase de análisis previo.

TABLA 1 – Cobertura Comparativa de Vulnerabilidades Lógicas

Método	Cobertura lógica estimada	Validación empírica	Capacidad de encadenamiento
Manual	100%	Sí	Alta
Baseline	~40%	No	Nula
CAI	~70%	No	Parcial (inferida)

Interpretación técnica:
El análisis manual continúa siendo el único método capaz de validar impacto real y encadenar vulnerabilidades. El baseline tradicional muestra limitaciones estructurales en detección lógica, mientras que CAI mejora significativamente la cobertura conceptual.

TABLA 2 – Falsos Negativos

Método	Falsos negativos estimados	Tipo de vulnerabilidades omitidas
Manual	0%	—
Baseline	~65%	BOLA, BOPLA, lógica de negocio
CAI	~30%	Vulnerabilidades puramente dinámicas

Interpretación técnica:
El baseline automatizado omite mayoritariamente vulnerabilidades críticas relacionadas con autorización y flujo de negocio. CAI reduce esta brecha, aunque sigue dependiendo del contexto proporcionado.

TABLA 3 – Comparativa de Exposición Técnica (CVE)

Aplicación	Total CVE	CRITICAL	HIGH	MEDIUM	LOW	Origen principal
VAmpI	12	0	5	7	0	Dependencias Python
Juice Shop	91	10	35	30	16	OS + Node.js

Interpretación técnica:
Juice Shop presenta una superficie técnica significativamente mayor. VAmpI concentra su riesgo en dependencias de aplicación, no en sistema base.

7.CONCLUSIONES GENERALES Y LÍNEAS FUTURAS

7.1 Síntesis global del estudio

El presente trabajo ha desarrollado un análisis experimental comparativo entre tres enfoques de evaluación de seguridad en aplicaciones web y APIs REST deliberadamente vulnerables:

- Análisis manual (Ground Truth)
- Baseline automatizado tradicional (OWASP ZAP + Trivy)
- Análisis asistido por inteligencia artificial (CAI)

El experimento se realizó sobre dos laboratorios representativos y ampliamente utilizados en formación e investigación:

- VAmpI (Vulnerable API)
- OWASP Juice Shop (Aplicación Web)

La metodología aplicada permitió observar empíricamente no solo la capacidad de detección de cada enfoque, sino también sus limitaciones estructurales, su comportamiento frente a vulnerabilidades lógicas y su impacto en la priorización de riesgos.

El análisis manual se estableció como referencia real (ground truth), al validar explotación efectiva, encadenamiento de fallos y confirmación práctica del impacto.

7.2 Principales hallazgos

7.2.1 Superioridad del análisis manual en validación real

El análisis manual fue el único enfoque capaz de:

- Confirmar explotación real.
- Encadenar vulnerabilidades.
- Detectar fallos de autorización complejos (BOLA/BOPLA).

- Identificar vulnerabilidades de lógica de negocio.
- Validar impacto sistémico.

Esto confirma que el análisis manual continúa siendo el estándar de referencia en auditorías técnicas profundas.

Sin embargo, también requiere:

- Alto conocimiento técnico.
- Mayor tiempo de ejecución.
- Esfuerzo humano significativo.

7.2.2 Limitaciones estructurales del baseline automatizado

Las herramientas tradicionales (ZAP + Trivy) demostraron:

Fortalezas:

- Detección rápida de misconfiguraciones.
- Identificación de cabeceras inseguras.
- Enumeración de dependencias vulnerables (CVE).
- Cobertura técnica superficial efectiva.

- **Debilidades:**
- Incapacidad para detectar vulnerabilidades lógicas.
- Nula detección de fallos de autorización contextual.
- Incapacidad de interpretar flujos multi-endpoint.
- Alta generación de hallazgos de bajo impacto (ruido).

El baseline automatizado mostró una cobertura media estimada del 40–45% respecto al ground truth explotable, concentrándose principalmente en superficie técnica y no en impacto real.

7.2.3 Aportación diferencial del enfoque CAI

La exploración del enfoque CAI reveló un comportamiento intermedio entre análisis manual y automatización tradicional.

CAI demostró capacidad para:

- Interpretar semánticamente contratos OpenAPI.
- Inferir riesgos estructurales de autorización.
- Priorizar hallazgos relevantes frente a ruido.
- Contextualizar vulnerabilidades CVE.
- Conectar hallazgos con categorías OWASP.
- Identificar potenciales debilidades de diseño.

Su cobertura conceptual respecto al ground truth se estimó en un 65–75% en identificación de riesgos, superando claramente al baseline tradicional en vulnerabilidades lógicas.

No obstante, CAI no ejecuta pruebas dinámicas ni valida empíricamente la explotación, por lo que no puede sustituir al análisis manual.

7.3 Conclusión principal del estudio

El hallazgo central del trabajo puede sintetizarse en la siguiente afirmación:

> La inteligencia artificial aplicada a ciberseguridad no sustituye al analista humano, pero sí actúa como amplificador semántico del análisis cuando se utiliza de forma estructurada.

El valor diferencial de CAI no radica en detectar más vulnerabilidades técnicas que las herramientas tradicionales, sino en:

- Reducir ruido.
- Priorizar riesgos reales.
- Inferir vulnerabilidades lógicas.

- Acelerar la interpretación de resultados automatizados.
- Conectar contexto técnico con impacto potencial.

En este sentido, CAI se posiciona como una capa cognitiva intermedia entre automatización tradicional y validación manual.

7.4 Implicaciones para la práctica profesional

Desde una perspectiva profesional, los resultados sugieren que:

1. El análisis manual sigue siendo imprescindible en auditorías profundas.
2. El baseline automatizado es útil como primera capa de superficie técnica.
3. La integración de IA puede mejorar la eficiencia del proceso de análisis.
4. La priorización inteligente reduce el tiempo dedicado a falsos positivos.
5. La combinación híbrida (Manual + Automatización + IA) es el enfoque más robusto.

Un modelo operativo recomendado sería:

1) Escaneo automatizado
2) Interpretación semántica asistida por IA
3) Validación y explotación manual

Este flujo optimiza recursos humanos sin sacrificar profundidad técnica.

7.5 Limitaciones del estudio

El presente trabajo presenta varias limitaciones que deben ser consideradas:

- Entornos de laboratorio deliberadamente vulnerables.
- Número limitado de aplicaciones analizadas.
- Evaluación exploratoria del enfoque CAI.
- Dependencia de la calidad del input proporcionado a la IA.
- No implementación de un sistema autónomo completamente integrado.
- Limitaciones prácticas observadas (cuotas, guardrails, dependencia de proveedor).

Estas limitaciones no invalidan los resultados, pero sí acotan su generalización a entornos productivos reales.

7.6 Reflexión técnica final

El experimento confirma una brecha clara entre:

- Lo que detectan las herramientas automatizadas.
- Lo que realmente compromete un sistema.

La mayor parte de vulnerabilidades críticas modernas no son fallos de configuración triviales, sino errores de lógica, autorización y diseño.

Este tipo de vulnerabilidades:

- No se detectan por firma.
- No aparecen en CVE.
- No se identifican por escaneo superficial.
- Requieren comprensión contextual.

La inteligencia artificial demuestra potencial precisamente en esa dimensión semántica.

No obstante, la validación empírica sigue siendo responsabilidad humana.

7.7 Líneas futuras de investigación

El presente trabajo abre múltiples posibilidades de desarrollo futuro:

7.7.1 Integración automatizada OpenAPI → Motor de hipótesis

Desarrollo de un sistema que:

- Analice automáticamente contratos OpenAPI.
- Genere hipótesis de vulnerabilidades lógicas.
- Produzca checklist de pruebas manuales sugeridas.

7.7.2 Métricas cuantitativas formales de reducción de ruido

Diseñar indicadores objetivos que midan:

- Reducción porcentual de falsos positivos.
- Tiempo ahorrado en priorización.
- Mejora de precisión en detección de riesgos lógicos.

7.7.3 Evaluación con mayor número de aplicaciones reales

Extender el estudio a:

- APIs empresariales reales.
- Aplicaciones SaaS.
- Microservicios distribuidos.
- Entornos cloud-native.
-

7.7.4 Comparativa entre distintos modelos LLM especializados

Analizar:

- Modelos generalistas vs modelos especializados en seguridad.
- Impacto del fine-tuning específico en OWASP API Top 10.
- Diferencias en inferencia lógica.

7.7.5 Integración en pipelines DevSecOps

Explorar el uso de IA como:

- Filtro inteligente de escaneos CI/CD.
- Motor de priorización automática.
- Asistente contextual en revisión de código.

7.8 Cierre

Este trabajo demuestra que la ciberseguridad moderna requiere una aproximación híbrida.

La automatización tradicional aporta velocidad.
El análisis manual aporta profundidad.
La inteligencia artificial aporta comprensión contextual.

El futuro no está en reemplazar al analista humano, sino en potenciarlo mediante herramientas cognitivas que reduzcan ruido, mejoren la priorización y amplíen la capacidad de razonamiento sobre sistemas complejos.

En este marco, el enfoque CAI representa una evolución prometedora en la intersección entre inteligencia artificial y auditoría de seguridad.

ANEXOS

Evidencias: (Carpeta trabajo final)

ENTORNO EXPERIMENTAL Y METODOLOGÍA | Carpeta 01_Entorno_Experimental

RESULTADOS EXPERIMENTALES DETALLADOS | Carpeta 02_Analisis_Manual_GroundTruth + 03_Baseline_Automatizado + 04_Exploracion_CAI

ANÁLISIS ESTADÍSTICO DEL EXPERIMENTO | Carpeta 05_Comparativa_Experimental

Enlaces:

También se han utilizado otros como foros y blogs, pero al no ser sitios oficiales no se nombrarán

Vulnerabilidades CVE y bases de datos oficiales

NVD – National Vulnerability Database (Oficial)

https://nvd.nist.gov

Base de datos oficial del NIST para consultar CVEs, severidad CVSS, vectores de ataque y referencias técnicas.

MITRE CVE Database

https://cve.mitre.org

Fuente primaria del sistema CVE (Common Vulnerabilities and Exposures).

CVE Details

https://www.cvedetails.com

Base de datos estructurada para análisis estadístico de CVEs por producto, severidad y año.

FIRST – CVSS Specification

https://www.first.org/cvss/

Especificación oficial del sistema de puntuación CVSS.

OWASP – Vulnerabilidades Web

OWASP Top 10 (Web)

https://owasp.org/www-project-top-ten/

Documento oficial de las 10 categorías principales de riesgo en aplicaciones web.

OWASP Testing Guide

https://owasp.org/www-project-web-security-testing-guide/

Metodología estructurada para pruebas de seguridad manuales.

OWASP Juice Shop (Proyecto oficial)

https://owasp.org/www-project-juice-shop/

Repositorio y documentación oficial del laboratorio utilizado.

GitHub:
https://github.com/juice-shop/juice-shop

OWASP API Security
OWASP API Security Top 10

https://owasp.org/www-project-api-security/

para:

- BOLA
- BOPLA
- Broken Authentication
- Security Misconfiguration en APIs

OWASP API Security Top 10 (2023)

https://owasp.org/API-Security/editions/2023/en/0x00-header/

Versión actualizada de referencia académica.

Herramientas utilizadas

OWASP ZAP

Proyecto oficial:
https://www.zaproxy.org

Documentación:
https://www.zaproxy.org/docs/

Baseline Scan:
https://www.zaproxy.org/docs/docker/baseline-scan/

Trivy

Proyecto oficial:
https://aquasecurity.github.io/trivy/

GitHub:
https://github.com/aquasecurity/trivy

Documentación:
https://aquasecurity.github.io/trivy/latest/

Inteligencia Artificial aplicada a Ciberseguridad
Alias Robotics – CAI

Repositorio:
https://github.com/aliasrobotics/cai

Documentación:
https://aliasrobotics.com/cybersecurityai.php

OpenAI API Documentation

https://platform.openai.com/docs

Conceptos de Seguridad en APIs y Web
OpenAPI Specification

https://spec.openapis.org/oas/latest.html

JWT (JSON Web Token – RFC 7519)

https://datatracker.ietf.org/doc/html/rfc7519

CORS (MDN Reference)

https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS

Content Security Policy (CSP)

https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP
