

Diagrama de Proceso Completo - HybridSecScan

Proyecto: Sistema de Auditoria Automatizada Hibrida (SAST + DAST)

Autor: Oscar Isaac Laguna Santa Cruz

Universidad: UNMSM - Facultad de Ingenieria de Sistemas e Informatica

Fecha: Noviembre 2025

Version: 1.0

Descripción General

Este diagrama muestra el flujo de proceso completo de la aplicación HybridSecScan desde que el usuario inicia una auditoría hasta que obtiene el reporte final consolidado. Incluye todos los módulos del sistema y sus interacciones.

Diagrama de Proceso End-to-End (Horizontal)

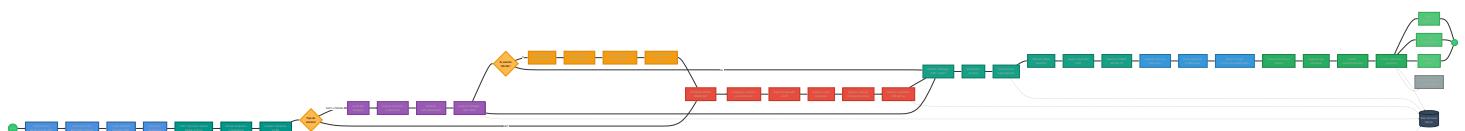
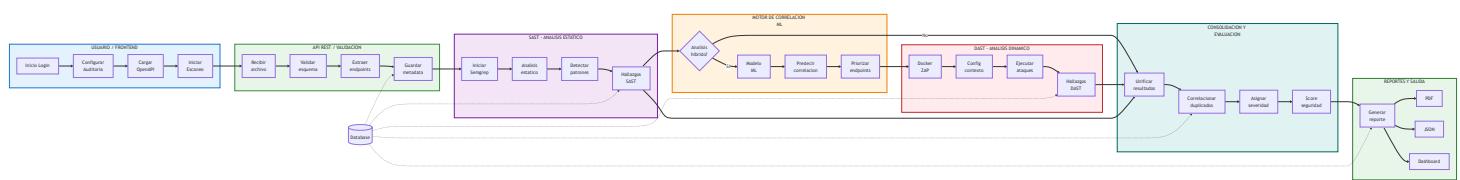


Diagrama con Swimlanes Horizontales



Descripcion Detallada del Flujo

Fase 1: Autenticacion y Configuracion

Actores: Usuario, Frontend, API

1. **Inicio de Sesion:** Usuario se autentica con credenciales (JWT token)
2. **Dashboard:** Visualiza proyectos y auditorias previas
3. **Seleccion de Analisis:** Elige tipo (SAST, DAST o Hibrido)
4. **Carga de Archivo:** Upload de especificacion OpenAPI/Swagger
5. **Configuracion:** Define parametros de auditoria (severidad, alcance)

Tiempo estimado: 2-3 minutos

Fase 2: Validacion de Entrada

Actores: API, Modulo de Validacion, Base de Datos

1. **Repcion:** API recibe archivo OpenAPI via `POST /api/scan/upload`
2. **Validacion de Schema:** Verifica formato JSON/YAML conforme a OpenAPI 3.0
3. **Extraccion de Endpoints:** Parsea paths, methods, parameters
4. **Identificacion de Parametros:** Detecta query, body, headers, path params
5. **Persistencia:** Guarda metadata en base de datos

Tiempo estimado: 10-30 segundos

Fase 3: Analisis Estatico (SAST)

Actores: Modulo SAST, Semgrep, Base de Datos

1. **Inicializacion:** Carga reglas de Semgrep para OWASP API Top 10
2. **Analisis Sintactico:** Valida estructura del esquema OpenAPI
3. **Analisis Semantico:** Busca patrones de vulnerabilidad en definiciones
4. **Deteccion de Vulnerabilidades:**
 - API1: Broken Object Level Authorization (BOLA)
 - API2: Broken Authentication
 - API3: Broken Object Property Level Authorization

- API5: Broken Function Level Authorization
- API8: Security Misconfiguration
- API9: Improper Inventory Management
- API10: Unsafe Consumption of APIs

5. Generacion de Hallazgos: Formato JSON estructurado con:

- Tipo de vulnerabilidad
- Endpoint afectado
- Severidad (Critical, High, Medium, Low)
- CWE ID
- Linea de codigo (si aplica)
- Recomendacion de mitigacion

Tiempo estimado: 1-2 minutos

Fase 4: Correlacion con Machine Learning

Actores: Motor de Correlacion, Modelo Random Forest

Condicion: Solo si el usuario selecciono analisis Hibrido

1. Carga del Modelo: Modelo pre-entrenado Random Forest (85% accuracy)

2. Extraccion de Features: 15 caracteristicas por hallazgo SAST:

- Tipo de vulnerabilidad
- Metodo HTTP
- Presencia de autenticacion
- Numero de parametros
- Tipo de datos (sensibles vs no sensibles)
- Profundidad del endpoint
- Uso de headers de seguridad
- Esquema de validacion

3. Prediccion: Probabilidad de confirmacion en DAST (0.0 - 1.0)

4. Calculo de Confianza: 4 factores ponderados:

- Similitud de endpoint (30%)
- Coincidencia de tipo de vulnerabilidad (40%)
- Severidad concordante (20%)
- Timestamp proximity (10%)

5. Priorizacion: Top N endpoints criticos para pruebas dinamicas

Tiempo estimado: 30-60 segundos

Fase 5: Analisis Dinamico (DAST)

Actores: Modulo DAST, OWASP ZAP, Docker

Condicion: Solo si el usuario selecciono DAST o Hibrido

1. **Inicializacion de Contenedor:** Docker ejecuta OWASP ZAP en modo daemon

2. **Configuracion de ZAP:**

- Modo API (headless)
- Context con URL base
- Authentication (si se proporciona)
- Politicas de escaneo (pasivo + activo)

3. **Importacion de Esquema:** ZAP carga OpenAPI para generar requests

4. **Generacion de Casos de Prueba:**

- Fuzzing de parametros
- Inyeccion SQL (SQLi)
- Cross-Site Scripting (XSS)
- BOLA (modificacion de IDs)
- Mass Assignment
- Security Misconfiguration

5. **Ejecucion de Ataques:** Requests maliciosos a endpoints reales

6. **Captura de Resultados:** Alertas, evidencias, screenshots, requests/responses

Tiempo estimado: 5-15 minutos (depende de cantidad de endpoints)

Fase 6: Consolidacion de Resultados

Actores: Modulo de Consolidacion, Motor de Correlacion

1. **Repcion:** Hallazgos de SAST y DAST (formato JSON)

2. **Normalizacion:** Esquema unificado para ambos tipos de hallazgos

3. **Correlacion de Duplicados:**

- ML: Modelo predice matches con 85% accuracy
- Reglas: Heuristicas para casos obvios (mismo endpoint + mismo CWE)

4. **Eliminacion de Falsos Positivos:** Validacion cruzada SAST-DAST

5. **Asignacion de Severidad:** CVSS v3.1 scoring

6. **Clasificacion OWASP:** Mapeo a OWASP API Security Top 10 (2023)

Tiempo estimado: 1-2 minutos

Fase 7: Sistema de Evaluacion

Actores: Sistema de Evaluacion

1. Calculo de Metricas:

- Total de vulnerabilidades encontradas
- Distribucion por severidad
- Distribucion por tipo OWASP

2. Score de Seguridad: Formula ponderada (0-100 puntos)

Score = 100 - (Critical*10 + High*5 + Medium*2 + Low*0.5)

3. Comparacion con Baseline: Historico de auditorias previas

4. Nivel de Riesgo:

- Critico: Score < 40
- Alto: Score 40-59
- Medio: Score 60-79
- Bajo: Score >= 80

Tiempo estimado: 10-20 segundos

Fase 8: Generacion de Reportes

Actores: Modulo de Reportes

1. Estructura del Reporte:

- Resumen ejecutivo
- Score de seguridad
- Distribucion de vulnerabilidades
- Hallazgos detallados (ordenados por severidad)
- Recomendaciones de mitigacion
- Referencias OWASP

2. **Organizacion por Severidad:** Critical > High > Medium > Low
3. **Recomendaciones:** Guias de mitigacion especificas por tipo de vulnerabilidad
4. **Evidencias:** Screenshots, requests HTTP, responses
5. **Graficos:** Charts de distribucion (Pie, Bar, Timeline)

Formatos de salida:

- **PDF:** Reporte ejecutivo para gerencia (6-20 paginas)
- **JSON:** Formato programatico para CI/CD
- **Dashboard Web:** Visualizacion interactiva con filtros

Tiempo estimado: 30-60 segundos

Fase 9: Notificaciones y Persistencia

Actores: Sistema de Notificaciones, Base de Datos

1. **Email:** Notificacion al usuario con link al reporte
2. **Webhook:** Callback a sistema CI/CD (Jenkins, GitLab CI)
3. **Logs:** Registro de eventos en sistema de auditoria
4. **Base de Datos:** Persistencia de todos los hallazgos y reportes

Tiempo estimado: 5-10 segundos

Tiempos Totales Estimados

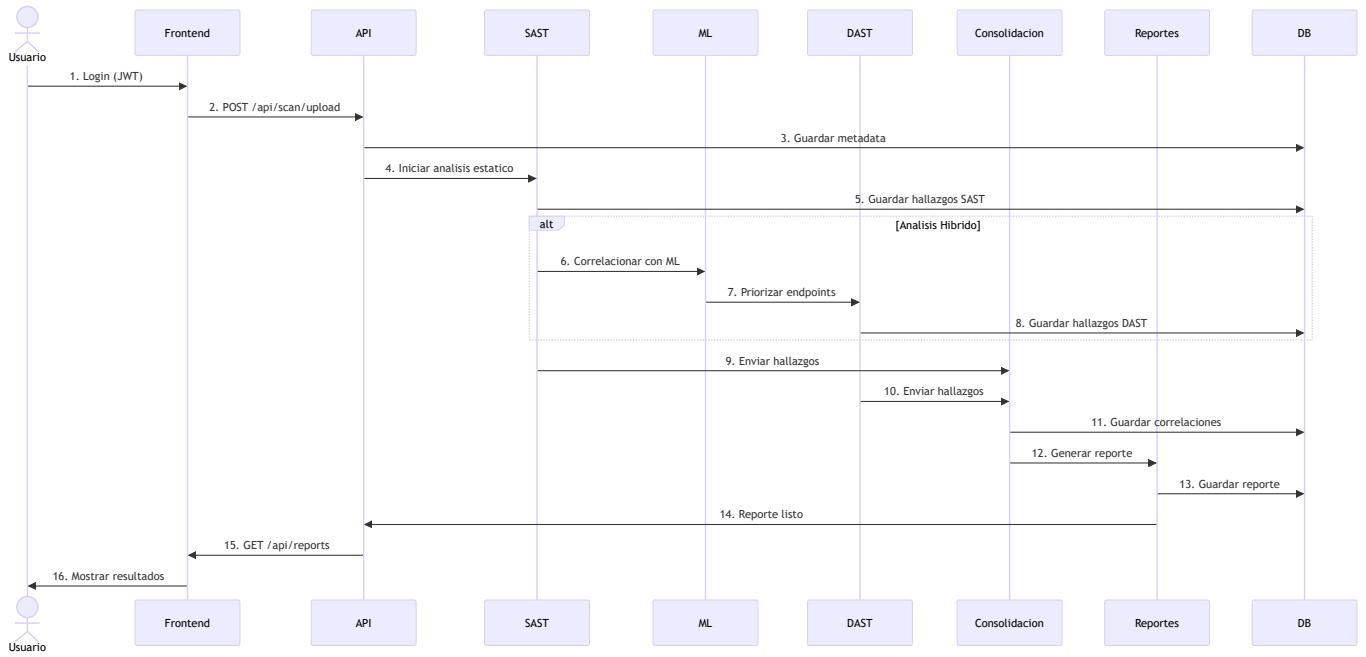
Tipo de Analisis	Tiempo Minimo	Tiempo Maximo
SAST Solamente	2 minutos	5 minutos
DAST Solamente	6 minutos	18 minutos
Hibrido (SAST + ML + DAST)	9 minutos	20 minutos

Factores que afectan el tiempo:

- Cantidad de endpoints en la API (10-100+)
- Complejidad de los endpoints (parametros, autenticacion)

- Tipo de escaneo DAST (pasivo vs activo)
- Recursos del servidor (CPU, RAM)
- Latencia de red (si API esta remota)

Diagrama de Secuencia Simplificado



Casos de Uso

Caso de Uso 1: Auditoria Rapida (Solo SAST)

Escenario: Desarrollador quiere validar su OpenAPI antes de commit

Flujo:

1. Login en dashboard
2. Upload de archivo `openapi.yaml`
3. Seleccionar "SAST Solamente"
4. Iniciar auditoria
5. Esperar 2 minutos
6. Descargar reporte PDF

Resultado esperado: Reporte con vulnerabilidades estaticas detectadas

Caso de Uso 2: Auditoria Completa (Hibrido)

Escenario: Equipo de seguridad audita API en produccion

Flujo:

1. Login en dashboard
2. Upload de archivo OpenAPI + URL base de API
3. Seleccionar "Analisis Hibrido"
4. Configurar nivel de severidad: "All"
5. Iniciar auditoria
6. Monitor de progreso en tiempo real
7. Esperar 12 minutos
8. Revisar dashboard interactivo
9. Descargar reporte PDF para gerencia
10. Descargar JSON para CI/CD

Resultado esperado: Reporte consolidado con vulnerabilidades confirmadas

Caso de Uso 3: Integracion CI/CD

Escenario: Pipeline automatizado de Jenkins

Flujo:

1. Commit en repositorio dispara webhook
2. Jenkins ejecuta curl a API HybridSecScan
3. Upload de OpenAPI automatico
4. Espera asincronica (polling cada 30s)
5. Descarga JSON con resultados
6. Parsea JSON y bloquea deploy si Critical > 0
7. Envia notificacion a Slack con resultados

Resultado esperado: Pipeline bloqueado si hay vulnerabilidades criticas

Patrones de Diseño Aplicados

1. **Strategy Pattern:** Seleccion de tipo de analisis (SAST/DAST/Hibrido)
2. **Factory Pattern:** Creacion de escaneres (Semgrep, ZAP)
3. **Observer Pattern:** Monitor de progreso en tiempo real
4. **Chain of Responsibility:** Pipeline de validacion y procesamiento
5. **Singleton Pattern:** Conexion a base de datos
6. **Facade Pattern:** API REST oculta complejidad interna

Consideraciones de Seguridad

Autenticacion y Autorizacion

- JWT tokens con expiracion (24h)
- Refresh tokens para sesiones largas
- RBAC: Admin, Auditor, Developer

Proteccion de Datos Sensibles

- Encriptacion de credenciales en BD (bcrypt)
- HTTPS obligatorio en produccion
- API keys para integraciones CI/CD

Rate Limiting

- 10 auditorias/hora por usuario
- 100 auditorias/dia por organizacion

Logs de Auditoria

- Quien ejecuto la auditoria
- Cuando se ejecuto
- Que endpoints se escanearon
- Resultados obtenidos

Metricas de Rendimiento

Recursos del Sistema

Componente	CPU	RAM	Disco
Backend FastAPI	10-30%	512MB	100MB
SAST (Semgrep)	20-40%	256MB	50MB
DAST (OWASP ZAP)	40-80%	2GB	500MB
ML Correlation	15-25%	512MB	200MB
Database (SQLite)	5-10%	128MB	1GB
Total	90-185%	3.4GB	1.85GB

Recomendacion: Servidor con 4 CPU cores, 8GB RAM, 10GB disco

Limitaciones y Trabajo Futuro

Limitaciones Actuales

- Solo soporta OpenAPI 3.0 (no soporta Swagger 2.0)
- DAST requiere API accesible via HTTP (no APIs internas)
- Modelo ML entrenado con 500 samples (necesita mas datos)
- No soporta autenticacion OAuth2 compleja en DAST

Mejoras Futuras

- Soporte para GraphQL APIs
- Integracion con Postman Collections
- Deteccion de API drift (cambios no documentados)
- Reentrenamiento continuo del modelo ML
- Soporte para WebSockets y gRPC

Referencias

Estandares

- OWASP API Security Top 10 (2023): <https://owasp.org/API-Security/>
- OpenAPI Specification 3.0: <https://spec.openapis.org/oas/v3.0.0>
- CVSS v3.1: <https://www.first.org/cvss/v3.1/specification-document>

Herramientas

- Semgrep: <https://semgrep.dev/>
- OWASP ZAP: <https://www.zaproxy.org/>
- FastAPI: <https://fastapi.tiangolo.com/>
- Scikit-learn: <https://scikit-learn.org/>

Contacto

Autor: Oscar Isaac Laguna Santa Cruz

Email: oscar.laguna@unmsm.edu.pe

Universidad: UNMSM - FISI

Proyecto: HybridSecScan

Repositorio: <https://github.com/OscarILS/HybridSecScan>

Ultima actualizacion: Noviembre 23, 2025