PROPUESTA TFG MARIO A. 2DAM

masenper@myuax.com

9 de Febrero 2026

Idea: Plataforma de Pre-Explotación y Análisis Ofensivo de Binarios

Enfoque: Pre-Explotación (Reconocimiento técnico del binario y su comportamiento), sin explotación automática.

Qué es exactamente:

Una plataforma que, dado un binario (ELF / PE), ejecuta:

1. Análisis estático (sin ejecutar):
- Metadatos, secciones, imports/exports.
- Protecciones (PIE/NX/RELRO/canary/ASRL...).
- Strings interesantes y rutas.
- Heurísticas de riesgo (funciones peligrosas, símbolos, etc).
2. Análisis dinámico controlado (ejecución monitorizada):
- Trazado de syscalls / procesos.
- Accesos al sistema de ficheros.
- Conexiones de red registradas (básico).
- Resumen de comportamiento.
3. Sistema de puntaje ofensivo + priorización:
- Combina hallazgos estáticos y dinámicos.
- Genera un riesgo ofensivo (prioridad de revisión).
4. Informe final:
- HTML o PDF.
- Con secciones claras y “acciones sugeridas” (sin guías de explotación).

Problema real que resuelve:

1. En auditorías ofensivas reales, el inicio suele ser:
- Herramientas sueltas (strings, checksec, ldd, strace...).
- Resultados dispersos tras análisis.
- Sin trazabilidad ni priorización.
2. Mi plataforma:
- Centraliza el análisis inicial.
- Reduce el tiempo de reconocimiento.
- Prioriza qué revisar primero.
- Genera un informe consistente.

Objetivos:

Objetivo General:

Diseñar e implementar una plataforma modular para la fase de pre-explotación, capaz de realizar análisis estático y dinámico de binarios y generar un informe con priorización de riesgos orientada a una auditoría ofensiva.

Licencia:

Se publica bajo licencia MIT para facilitar revisión, reutilización académica y adopción, manteniendo atribución en el siguiente repositorio:

mario-asenjo/DAM-TFG-PPAOB: Repositorio de mi trabajo final de grado superior de Desarrollo de Aplicaciones Multiplataforma.

Tecnologías Utilizadas:

Núcleo / Bajo Nivel:

C: Capturador dinámico (ptrace / seccomp) y/o parser binario básico.
- Justificación: Control de syscalls, rendimiento y acceso a estructuras de bajo nivel.

Orquestación y Análisis:
Python: Pipeline de análisis, heurísticas, sistema de puntaje, generación de informe.
- Justificación: Rapidez de desarrollo y ecosistema de parsing / reporting.

Interfaz:
Java + JavaFX: UI Desktop multiplataforma.
- Justificación: Entrego un producto usable en cualquier sistema.

Almacenamiento:
Dudo entre SQLite o JSON persistente.
- Justificación: Trazabilidad, historial de análisis y reproducibilidad.

Capa Ética:

La plataforma se limita a la fase de pre-explotación (identificación y priorización de superficie de ataque), excluyendo funcionalidades de explotación automática o persistencia, enfocándose en auditoría y evaluación de seguridad.

Atentamente,

Mario Asenjo