UNIVERSITAT DE BARCELONA

Treball de Fi de Grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Metodología para Aplicación del ENS 2022 Nivel ALTO en PYMES

Daniel Gonzalez Braza

Director: Raúl Roca Canovas
Realitzat a: Departament de
Matemàtiques i Informàtica

Barcelona, 10 de junio de 2025

Agradecimientos

Quiero dedicar estas líneas a todas las personas que, de una u otra forma, han hecho posible la realización de este Trabajo de Fin de Grado.

En primer lugar, a mi familia, por su amor incondicional, su paciencia y sus constantes ánimos. Gracias a vosotros he encontrado la tranquilidad y el respaldo necesarios para afrontar cada reto.

A mis amigos, por compartir conmigo risas y desahogos, especialmente en esos momentos en que el progreso se ralentizaba o cuando un error de código parecía no tener solución. Vuestra compañía, vuestras sugerencias y vuestro sentido del humor hicieron más llevadero este proceso y me ayudaron a recargar energías cada vez que lo necesitaba.

Por último, quiero agradecer al profesor Raúl Roca Canovas, quien propuso el tema de este TFG y llevó a cabo la planificación inicial del proyecto. Aunque no hayamos coincidido con frecuencia, también ha sido importante para sacar adelante este trabajo.

A todos vosotros, mil gracias.

Abstract

Implementing ENS 2022 at the High level requires SMEs to adopt numerous security measures, a task that is often costly and complex due to limited specialized staff and resources. This Final Degree Project introduces a software-assisted methodology, delivered through a containerized platform, that guides organizations through the entire compliance process. The solution integrates eight scripts covering key stages: initial compliance assessment, diagnostic report generation with recommendations, security policy creation, permissions auditing, incident monitoring, attack simulation, resource optimization, and executive report production. Each script automates tasks that would otherwise demand extensive manual effort. The platform was validated in a Windows/Linux laboratory environment, demonstrating the ability to automate the majority of programmable controls, significantly reduce operational workload, and generate audit-ready documentation.

Resumen

La adopción del ENS 2022 en su nivel Alto exige a las PYMEs implantar numerosas medidas de seguridad, lo cual suele resultar costoso y complejo dada su falta de personal especializado y recursos limitados. Este Trabajo de Fin de Grado presenta una metodología asistida por software, implementada en una plataforma Dockerizada, que guía a las empresas a lo largo de todo el proceso de adecuación. La solución integra ocho scripts que cubren fases clave: evaluación inicial de cumplimiento, generación de diagnóstico con recomendaciones, creación de políticas de seguridad, auditoría de permisos, monitorización de incidentes, simulación de ataques, optimización de recursos y elaboración de informes ejecutivos. Cada script automatiza tareas que, de otro modo, requerirían horas de trabajo manual. La plataforma se validó en un entorno de laboratorio Windows/Linux, demostrando que es posible automatizar gran parte de los controles programables, reducir significativamente el esfuerzo operativo y generar la documentación necesaria para auditorías de conformidad.

Resum

L'adopció de l'ENS 2022 en el seu nivell Alt obliga les PIMEs a implantar nombroses mesures de seguretat, sovint costoses i complexes atès la manca de personal especialitzat i recursos reduïts. Aquest Treball de Fi de Grau presenta una metodologia assistida per programari, desplegada en una plataforma Dockeritzada, que guia les empreses en tot el procés d'adequació. La solució integra vuit scripts que cobreixen fases claus: avaluació inicial de compliment, generació de diagnòstic amb recomanacions, creació de polítiques de seguretat, auditoria de permisos, monitorització d'incidents, simulació d'atacs, optimització de recursos i elaboració d'informes executius. Cada script automatitza tasques que, d'altra banda, requereixen hores de treball manual. La plataforma es va validar en un entorn de laboratori Windows/Linux, demostrant la capacitat d'automatitzar gran part dels controls programables, reduir de manera significativa l'esforç operatiu i generar la documentació necessària per a auditories de conformitat.

Índice

1. Introducción

1.1 Contexto y estado del arte del ENS 2022 en PYMEs

En España, el Esquema Nacional de Seguridad (ENS) se configura como el marco normativo principal para garantizar la protección de los sistemas de información de las administraciones públicas y de las entidades que prestan servicios o manejan datos para ellas. Su última versión está recogida en el Real Decreto 311/2022, que establece principios, requisitos mínimos y una catalogación de controles según tres niveles de seguridad, Bajo, Medio y Alto, en función del impacto que tendría un incidente sobre la organización, sus activos o las personas atendidas.

El nivel Alto del ENS se reserva para aquellos sistemas cuya categoría de riesgo exige un grado de protección muy riguroso. En este nivel, prácticamente la totalidad de los controles técnicos, organizativos y operacionales definidos por la normativa debe estar implantada, cubriendo ámbitos tan diversos como la gestión de riesgos, el control de accesos, la monitorización de eventos, la continuidad de servicio o la protección de la información. Según la última versión del ENS y los listados oficiales, son alrededor de 62 controles aplicables a nivel Alto, repartidos en familias que incluyen desde la definición de políticas y procedimientos hasta la configuración detallada de cortafuegos, sistemas de respaldo y vigilancia de logs.

Las guías del CCN-CERT y otros recursos oficiales describen recomendaciones para adaptar el ENS a organizaciones con menos recursos (perfiles simplificados, perfiles de cumplimiento), pero en la práctica las PYMEs siguen encontrando dificultades para llevar estas directrices a la operativa diaria de sus sistemas.

Por ello, en los últimos años han surgido propuestas de automatización de tareas clave de seguridad: auditorías de configuración, generación de evidencias, gestión de incidentes y elaboración de informes. Estas aproximaciones buscan reducir la carga manual y el nivel de conocimiento técnico necesario, permitiendo avanzar hacia el cumplimiento del ENS Alt de forma más eficiente y sistemática.

1.2 Retos de la implantación en empresas de pequeño/mediano tamaño

La adopción del ENS 2022 en nivel Alto presenta múltiples desafíos para las pequeñas y medianas empresas, que difícilmente disponen de los recursos y la experiencia requeridos:

- **Falta de personal especializado**
  La implantación de controles como el análisis de riesgos formales, la configuración de sistemas de gestión de incidentes o la supervisión continua de logs exige perfiles con conocimientos avanzados en ciberseguridad. En la mayoría de PYMEs, el equipo de TI suele ser pequeño y sus responsabilidades se centran en mantener la operativa diaria, por lo que no pueden asumir tareas tan técnicas ni dedicar el tiempo necesario a ellas.

- **Restricciones presupuestarias**
  Muchas medidas del ENS, como disponer de soluciones SIEM, sistemas de respaldo avanzados o auditorías externas periódicas, conllevan costos de licencias, infraestructuras y servicios profesionales. Estos gastos, habitualmente elevados, superan con frecuencia el presupuesto de una PYME y dificultan la inversión en seguridad hasta niveles aceptables para una certificación de nivel Alto.
- **Complejidad operativa y de mantenimiento**
  Aun cuando se adquieren las herramientas adecuadas, configurarlas correctamente según las directrices del ENS y asegurar su continuidad requiere procesos robustos de gobernanza y revisión permanente. La gestión de parches, el ajuste de cortafuegos, la actualización de políticas de acceso o la revisión diaria de alertas son actividades que demandan tiempo y disciplina, contraponiéndose a las prioridades de negocio habituales de una empresa pequeña.
- **Evidencia y trazabilidad para auditorías**
  El ENS de nivel Alto obliga a generar y conservar evidencias detalladas de cada control (registros de eventos, informes de auditoría, plan de continuidad probado, etc.). Organizar y documentar esta información de forma estructurada y fiable supone un esfuerzo adicional de gestión documental, que muchas PYMEs no están preparadas para asumir sin apoyo externo.
- **Adaptación a entornos heterogéneos**
  Las PYMEs suelen operar con infraestructuras mixtas (Windows, Linux, soluciones cloud o locales) y, a menudo, carecen de inventarios actualizados de sus activos. Esto complica la aplicación uniforme de controles y obliga a personal sin experiencia a familiarizarse con plataformas diversas, multiplicando el riesgo de errores de configuración.

Estos retos evidencian la necesidad de apoyos tecnológicos que alivien la carga manual y de conocimiento especializado. La automatización de tareas, la generación guiada de evidencias y la centralización de la gestión de controles se perfilan como soluciones clave para acercar el cumplimiento del ENS Alto a las PYMEs, optimizando recursos y reduciendo la probabilidad de incumplimientos por simple omisión o error humano.

1.3 Motivación y relevancia del proyecto

La motivación de este proyecto surge de la necesidad de proporcionar a las PYMEs un apoyo efectivo para afrontar el cada vez más exigente entorno normativo en ciberseguridad. Ante la creciente digitalización de los procesos de negocio y el papel crítico que juegan los datos en la competitividad, resulta fundamental que las pequeñas y medianas empresas cuenten con herramientas asequibles que les permitan implantar de forma ágil y fiable los controles requeridos por el ENS 2022 en su nivel Alto.

Asimismo, el proyecto persigue cerrar la brecha entre la complejidad técnica de la normativa y la realidad operativa de las PYMEs, facilitando la generación de evidencias y la automatización de tareas repetitivas. De este modo, se pretende contribuir a elevar el nivel de seguridad global del tejido empresarial, reduciendo tanto el riesgo de incidentes como las barreras de entrada a la contratación con el sector público.

2. Planificación

2.1 Cronograma inicial y fases del proyecto

La planificación estableció cuatro grandes fases distribuidas entre el 15 de marzo y el 2 de junio de 2025, con el siguiente detalle:

- **Fase 1: Planificación y fundamentos (15 mar – 28 mar)**
  Se definieron el marco de referencia del ENS 2022 nivel Alto, se identificaron los controles aplicables a PYMEs y se elaboró el borrador del índice de la memoria. También se perfilaron los ocho scripts que conformarían la solución.
- **Fase 2: Desarrollo de la metodología y automatización (29 mar – 9 may)**
  Se dividió el desarrollo en subfases semanales para abordar progresivamente cada módulo:
  - **Semana 3 (29 mar – 4 abr):** diseño del checklist de controles y planificación del Script 1 (Evaluador de Cumplimiento).
  - **Semana 4 (5 abr – 11 abr):** definición de la estructura de datos y desarrollo inicial del Script 2 (Generador de Diagnóstico) y del Script 3 (Generador de Políticas).
  - **Semana 5 (12 abr – 18 abr):** desarrollo del Script 4 (Auditor de Permisos y Accesos) y primeras pruebas en entornos Windows/Linux.
  - **Semana 6 (19 abr – 25 abr):** implementación del Script 5 (Detector de Incidentes) y del Script 6 (Simulador de Ataques).
  - **Semana 7 (26 abr – 2 may):** creación del Script 7 (Optimizador de Recursos) y del Script 8 (Generador de Informes).
- **Fase 3: Validación, optimización y pruebas (10 may – 23 may)**
  Se reservó este periodo para montar el laboratorio seguro con máquinas virtuales, ejecutar todos los scripts en escenarios controlados, recopilar resultados y ajustar reglas y recomendaciones según el feedback recibido.
- **Fase 4: Documentación y entrega (24 may – 2 jun)**
  Se planificó dedicar la última semana a la redacción y revisión de la memoria, incluyendo capturas, tablas, referencias y formato final, para completar la entrega el 2 de junio de 2025

2.2 Ajustes en la planificación

Durante el desarrollo se produjeron dos desvíos respecto al cronograma inicial:

- **Extensión de la fase de Desarrollo de la metodología y automatización**
  Inicialmente se había previsto completar los ocho scripts en seis semanas, distribuidos desde el 29 de marzo al 9 de mayo. Sin embargo, surgieron desafíos técnicos durante la implementación de los módulos de monitorización e integración con Flask, lo que llevó a extender esta fase una semana adicional. En concreto, la verificación de incidencias y la configuración de la red interna requirieron pruebas iterativas y ajustes en las reglas de alertas, por lo que el desarrollo efectivo de todos los scripts concluyó el 16 de mayo en lugar del 9 de mayo planificado.

- **Retraso en la redacción de la memoria**
  A raíz de esta extensión, la fase final de documentación experimentó también un retraso de una semana. La redacción completa de la memoria finalizó el 8 de junio en lugar del 2 de junio establecido inicialmente, con el fin de incorporar las pruebas adicionales, actualizar capturas de pantalla y ajustar los apartados de resultados según los últimos refinamientos del código.

Estos ajustes permitieron asegurar la calidad y fiabilidad de los scripts de automatización, aunque supusieron un desplazamiento de siete días en la entrega final.

2.3 Diagrama de Gantt

Ilustración 1: Planificación temporal del proyecto en un diagrama de Gantt

- **Semanas 1–2 (15–28 mar):** Análisis del ENS 2022 y definición del alcance. Durante este bloque se identificaron los 62 controles técnicos, se filtraron los automatizables, se volcó la información en Excel y se redactó el índice provisional de la memoria.
- **Semanas 3–8 (29 mar–9 may):** Desarrollo e integración de los ocho scripts. En este intervalo se implementaron sucesivamente los módulos de diagnóstico, políticas, auditoría, monitorización, simulador de ataques, optimizador de recursos e informe final, dentro de la plataforma Docker/Flask.
- **Semanas 9–11 (10 may–30 may):** Pruebas en laboratorio y ajustes. Se montaron las máquinas virtuales Windows y Linux, se configuró la red interna y Beats, y se validó cada script en escenarios controlados, ajustando las reglas de detección y corrigiendo errores.
- **Semana 12 (31 may–2 jun):** Redacción de la memoria, captura de resultados y revisión final. Se incorporaron capturas, tablas y referencias, y se completaron los últimos retoques de estilo y formato antes de la entrega.

Este diagrama sintetiza la planificación original facilitando la visualización de los solapamientos entre fases y el flujo de trabajo semanal.

2.4 Distribución del esfuerzo

La dedicación total del proyecto se repartió aproximadamente según los siguientes porcentajes, reflejando las prioridades temporales en cada fase:

- 15 % Análisis y definición de requisitos
  Incluye la lectura y cribado de los 62 controles ENS, la elaboración del Excel de referencia y el borrador inicial del índice de la memoria.
- 15 % Diseño de la metodología y plataforma
  Cubre la definición de la arquitectura contenerizada, el mapeo de controles a módulos, el boceto de la interfaz web y la planificación de flujos de usuario.
- 40 % Desarrollo e integración de scripts
  Abarca la codificación de los ocho módulos (diagnóstico, políticas, auditoría, monitorización, simulador de ataques, optimizador y generador de informes), así como su integración en Flask y Docker.
- 20 % Pruebas y validación en laboratorio
  Comprende el montaje de las máquinas virtuales Windows/Linux, la configuración de Beats y Logstash, y la ejecución iterativa de pruebas para ajustar reglas y corregir errores.
- 10 % Documentación y entrega
  Engloba la redacción de la memoria, la captura de resultados, la maquetación final y la revisión de estilo y formato antes de la entrega oficial.

3. Objetivos

3.1 Objetivo general

El objetivo general de este Trabajo de Fin de Grado es desarrollar una plataforma contenerizada que asista a las pequeñas y medianas empresas en la implantación del ENS 2022 en su nivel Alto, automatizando las tareas técnicas necesarias para el cumplimiento normativo. Dicha plataforma integrará un conjunto de ocho scripts especializados que guiarán al usuario desde la evaluación inicial de su grado de conformidad hasta la generación de la documentación requerida para auditorías, reduciendo de forma significativa el esfuerzo manual y el nivel de conocimiento técnico necesarios. En última instancia, se pretende demostrar que, mediante esta solución, una PYME puede avanzar de manera ágil y sistemática hacia la conformidad con los controles técnicos del ENS 2022, mejorando su postura de seguridad sin incurrir en elevados costes de personal o infraestructura.

3.2 Objetivos específicos

3.2.1 Análisis de requisitos ENS y selección de controles

Se analizará en detalle el listado de controles técnicos del ENS 2022 nivel Alto para determinar cuáles son pertinentes en el contexto de una PYME y susceptibles de ser automatizados. A partir de ese análisis, se establecerá un criterio de prioridad que tenga en cuenta el impacto de cada control y la facilidad de automatización, con el objetivo de enfocar el desarrollo de scripts en primer lugar en los controles de mayor criticidad y viabilidad técnica.

3.2.2 Diseño de la metodología y plataforma automatitzada

Se definirá la arquitectura modular de la solución, basado en Docker, que permita orquestar los diferentes componentes (scripts) de forma aislada pero coordinada. Esto incluirá el diseño de la estructura de contenedores, la interfaz web de gestión, los volúmenes de datos para persistencia de resultados, y los mecanismos de comunicación entre módulos.

3.2.3 Desarrollo e integración de los 8 scripts

Se implementarán e integrarán ocho scripts especializados, cada uno encargado de una fase distinta del proceso de adecuación (diagnóstico inicial, generación de políticas, auditoría de configuración, monitorización, simulación de ataques, optimización de recursos e informes). Se procurará emplear buenas prácticas de programación, documentar el código con comentarios y facilitar la parametrización para adaptarse a diferentes entornos.

3.2.4 Validación en caso de estudio simulado

Se montará un entorno simulado compuesto por máquinas virtuales Windows y Linux que representan la infraestructura típica de una PYME. En él, se desplegará la plataforma y se ejecutarán los scripts para medir métricas clave: cobertura de controles automatizados, tiempo de ejecución, detección de incidentes simulados y calidad de las evidencias generadas. Los resultados servirán para evaluar la eficacia y robustez de la solución.

3.2.5 Documentación y evidencias de cumplimiento

A partir de los resultados de los scripts y las pruebas de laboratorio, se elaborarán informes ejecutivos y documentación de soporte (tablas de cumplimiento, logs de incidentes, capturas de pantalla y PDFs) listos para presentarse en auditorías de conformidad con el ENS. Además, se documentará el proceso de despliegue y uso de la plataforma para facilitar su replicación en otras PYMEs.

4. Metodología y Desarrollo

4.1 Análisis de requisitos y selección de controles automatizables

Para definir el alcance de los scripts, se llevó a cabo un análisis exhaustivo de los 62 controles técnicos del ENS 2022 nivel Alto. Todos ellos se volcaron en un Excel con tres columnas: código del control, nombre y descripción. A continuación, cada control fue evaluado según dos criterios:

- **Aplicabilidad a PYMEs:** se identificaron aquellos controles pertinentes en el contexto de una pequeña o mediana empresa, descartando los de carácter puramente organizativo o de gobernanza que requieren intervención humana directa (por ejemplo, la creación de comités formales o la formación masiva de personal).
- **Viabilidad de soporte automatizado:** se identificó qué controles podían beneficiarse directamente de un script. Por ejemplo, los controles de gestión de accesos (OP.ACC) quedaron asignados al Script 4: Auditor de Permisos y Accesos, mientras que los relativos al registro de eventos (OP.EXP.9) y detección de incidentes se canalizaron al Script 5: Detector de Incidentes ENS.

Con este cribado, se definieron dos grandes bloques funcionales para la plataforma:

- **Detección y diagnóstico inicial (cubiertos por los Scripts 1 y 2):**
  Se implementa una fase de evaluación automática para comprobar el estado de cumplimiento de cada control y generar un informe estructurado con recomendaciones específicas.
- **Soporte a la implementación y verificación continua (cubierto por los restantes scripts):**
  Se diseñaron módulos que automatizan comprobaciones técnicas, simulan escenarios de ataque, optimizan la asignación de recursos y generan la documentación necesaria, asegurando así la verificación de los controles seleccionados.

De este modo, la metodología garantiza que todos los controles técnicos aplicables al ENS nivel Alto queden cubiertos, empezando por un diagnóstico sólido y siguiendo con un respaldo automatizado para facilitar la adecuación y la generación de evidencias.

4.2 Arquitectura y despliegue contenerizado

4.2.1 Arquitectura lógica de la plataforma

La plataforma se estructura como un único contenedor Docker que reúne todos los componentes necesarios para automatizar los controles técnicos del ENS 2022 nivel Alto en una PYME. Partiendo de una imagen base ligera, por ejemplo, python:3.10-slim se instalan en ella todas las dependencias: bibliotecas de análisis de datos (Pandas), generación de gráficos (Matplotlib), servicios web (Flask), lectura de YAML (PyYAML), y herramientas de red como Scapy. Dentro del contenedor, la carpeta /app/scripts/ alberga los ocho módulos de automatización, cada uno aislado en su propio subdirectorio con sus requisitos específicos. La interfaz de usuario, desarrollada en Flask y ubicada en /app/web/, ofrece un panel web desde el que el usuario puede lanzar cada script, ajustar parámetros y consultar resultados.

Para mantener la persistencia de datos entre ejecuciones, se definen tres volúmenes montados desde el host:

- /data/inputs, donde se depositan los ficheros de entrada (el Excel de controles, archivos CSV/JSON de checklist o config.yaml con parámetros de umbral);
- /data/results, que recoge las salidas de los scripts (informes, archivos CSV/JSON, gráficos en PNG y PDF finales);
- /data/logs, destinado a almacenar tanto los registros de la plataforma como los eventos capturados por el módulo de monitorización.

La comunicación interna se realiza por invocación directa de los scripts desde la aplicación Flask, que los ejecuta como procesos hijos o mediante llamadas a sus funciones principales. De este modo, no se exponen más puertos que el que atiende la interfaz web (por defecto el 8000), reduciendo la superficie de ataque.

4.2.2 Despliegue en Docker y requisitos del entorno

Para simplificar la instalación y asegurar un entorno consistente, toda la plataforma se distribuye como una única imagen Docker basada en Python 3.10. Dicha imagen incluye todas las dependencias necesarias para los scripts las cuales están definidas en el archivo requirements.txt.

La imagen se construye partiendo de python:3.10-slim, instalando las dependencias definidas en requirements.txt, copiando los directorios web/, scripts/ y templates/, y finalizando con la inclusión de config.yaml.Una vez construida, el contenedor se ejecuta con un comando único que monta los tres volúmenes locales que he mencionado antes.

Al arrancar, el contenedor levanta Flask en el puerto 8000 (http://localhost:8000). Gracias a Docker, no es necesario instalar Python ni librerías en el host; cualquier actualización basta con reconstruir la imagen, garantizando entornos idénticos en desarrollo, pruebas y producción.

4.3 Diseño de la interfaz web y flujo de Trabajo

La plataforma ofrece en su página principal un panel único con un botón dedicado para cada uno de los ocho scripts. El flujo de interacción varía según el tipo de script:

- **Scripts 1, 2, 3, 7 y 8 (ejecución in-app):**
  Al pulsar el botón de cualquiera de estos módulos, la propia web despliega un formulario o asistente paso a paso integrado en la misma página. Por ejemplo, el Script 1 (Evaluador de Cumplimiento) muestra un formulario con las preguntas derivadas de los controles del ENS; tras completar los campos, la web calcula el porcentaje de controles cumplidos y no cumplidos, lo presenta en un gráfico sencillo y ofrece un botón adicional para descargar el informe (CSV o JSON). Los scripts 2 y 3 funcionan de forma análoga: recogen datos de entrada, procesan la información en segundo plano y muestran los resultados directamente en pantalla, junto a opciones para exportarlos.
- **Scripts 4, 5 y 6 (ejecución local):**
  Dado que estas comprobaciones requieren interacción con la red o el sistema local (auditoría de permisos, monitorización de logs y simulación de ataques), la web

proporciona un enlace de descarga para cada script, acompañado de un README con instrucciones detalladas (dependencias, montaje de VM Windows/Linux, comandos de ejecución). Tras ejecutar el script en el entorno local, el usuario regresa a la web y utiliza la sección de “Subir resultados” para cargar el fichero de salida, ya sea un log, un CSV o un JSON. La plataforma analiza entonces ese archivo, extrae las recomendaciones pertinentes y las integra en el sistema de resultados para su posterior consolidación.

- **Integración y unificación de resultados:**
  Todos los resultados parciales, los informes generados por scripts in-app y las recomendaciones derivadas del análisis de los outputs locales, se almacenan en la base de datos interna del contenedor. Esto permite que, al ejecutar el Script 8 (Generador de Informes), la web compile toda la información en un informe ejecutivo consolidado, que incluye gráficas, tablas de cumplimiento y comentarios globales, y lo ofrezca como PDF descargable.

Con este diseño, la plataforma equilibra la comodidad de uso (para los scripts que pueden correr enteramente en la web) con la necesaria flexibilidad local (para aquellos que requieren acceso a sistemas específicos), manteniendo siempre una interfaz coherente y minimizando las barreras técnicas para el usuario.

4.4 Estructura del proyecto practico

Todo el código de la plataforma se encuentra bajo el directorio web/, organizado para separar claramente los distintos componentes:

- **web/app.py**
  El punto de entrada de Flask que define rutas, configura la aplicación y enlaza cada endpoint con su lógica correspondiente. Esta única clase orquesta la recepción de formularios, la ejecución de scripts y la devolución de resultados al usuario.

- **web/scripts/**
  Contiene ocho módulos Python (script1_evaluador.py, script2_diagnostico.py, ..., script8_generar_informe.py) que encapsulan la lógica de cada funcionalidad in-app: diagnóstico, generación de plantillas, optimización y creación de informes. Además, en este mismo directorio reside el fichero ens_controls.py, que carga la lista de controles desde el Excel static/_data/ENS2022_nivel_ALTO.xlsx y la expone como estructura de datos para los primeros scripts.

- **web/parsers/**
  Este subdirectorio incluye tres clases especializadas (auditor_parser.py, detector_parser.py y simulador_parser.py) encargadas de leer y transformar los ficheros de salida de los Scripts 4, 5 y 6 (logs en texto plano o JSON) en objetos Python normalizados. Cada parser identifica secciones, extrae las alertas originales y mapea los campos a las recomendaciones configuradas en web/scripts/script4_map_recomendaciones.py, script5_map_recomendaciones.py y script6_map_recomendaciones.py.

- **web/templates/**
  Carpeta de plantillas Jinja2 dividida en:
    - templates/politicas/: YAML de base para cada política (e.g. politica_autenticacion_externa.yml, etc.).
    - vistas HTML: archivos como index.html, evaluar.html, diagnostico.html, politicas.html, auditor_permisos.html, detector_incidentes.html, simulador_ataques.html, optimizador_form.html, informe_final.html que heredan de base.html y definen el formulario, botones de descarga, subida de archivos y visualización de resultados.
- **web/static/**
    - static/_data/: almacena el Excel de controles ENS.
    - static/imagenes/: capturas y gráficos que ilustran la memoria y la interfaz.
    - static/informes/: guarda los PDF generados por Script 8 y las plantillas de políticas ya procesadas.
    - static/scripts/: versiones descargables de los Scripts 4, 5 y 6 (auditor_linux.sh, auditor_windows.ps1, detector_unificado.py, simulador_ataques.py) junto a sus README_*.md.
- **requirements.txt, Dockerfile y docker-compose.yml**
  Ubicados en la raíz, habilitan el despliegue contenerizado, instalando Flask, Pandas, Scapy, PyYAML, ReportLab y demás librerías necesarias.
- **uploads/**
  Directorio donde Flask almacena temporalmente los ficheros subidos por el usuario antes de pasarlos a los parsers.

Esta organización modular facilita la extensión o sustitución de cada componente (scripts, parsers, plantillas) sin afectar al resto de la plataforma, y promueve la separación de responsabilidades: la lógica de back-end queda separada de la presentación y de los ficheros estáticos.

4.5 Detalle de implementación de los 8 scripts

4.5.1 Script 1: Evaluador de Cumplimiento

El primer módulo de la plataforma, denominado Evaluador de Cumplimiento ENS, automatiza la fase inicial de diagnóstico del grado de conformidad de la PYME con los controles técnicos del ENS 2022 nivel Alto. Al acceder a la sección “Diagnóstico” de la interfaz web, el usuario carga el fichero Excel donde se ha volcado previamente la lista de 62 controles (con su código, nombre y descripción). El script lee esta hoja de cálculo mediante Pandas, valida la presencia de las columnas esenciales y genera dinámicamente un formulario HTML en el que, para cada control, el administrador puede indicar si cumple o no cumple la medida correspondiente.

Tras enviar el formulario, el Evaluador agrupa las respuestas y calcula tanto el porcentaje global de controles conformes como la distribución por familias de control (por ejemplo,

accesos, registros de eventos o protección de comunicaciones). Además, la página ofrece un botón de descarga que permite exportar todos los resultados en formato CSV o JSON, incluyendo para cada control su código, estado y un campo de comentarios opcionales.

Este módulo se integra completamente en la aplicación Flask, ejecutándose en el mismo contenedor Docker y sin necesidad de ficheros temporales en disco. De este modo, la experiencia de usuario resulta fluida y evita dependencias adicionales. Asimismo, el código está organizado de forma que cualquier modificación en el Excel de controles (incluir, eliminar o renombrar controles) se traduzca automáticamente en cambios en el formulario, sin requerir ajustes manuales en la plantilla HTML.

Con el Evaluador de Cumplimiento ENS se consigue ofrecer a las PYMEs un diagnóstico rápido, intuitivo y cuantitativo de su situación actual frente al ENS, estableciendo una base objetiva sobre la cual planificar las siguientes fases de la metodología automatizada.

4.5.2 Script 2: Generador de Diagnóstico

El Generador de Diagnóstico ENS toma como punto de partida la información obtenida por el Evaluador de Cumplimiento (Script 1) y la transforma en un informe estructurado y accionable para la PYME. Tras recopilar las respuestas al checklist, este módulo carga los datos en un DataFrame de Pandas, donde cada fila corresponde a un control ENS y las columnas incluyen el código, la descripción, el estado (“Cumple” o “No cumple”) y una recomendación predeterminada en caso de incumplimiento. Estas recomendaciones se almacenan en un diccionario interno que asocia a cada control un breve texto con la acción sugerida (por ejemplo, “Habilitar autenticación multifactor para accesos externos”).

Una vez creada la tabla, el script calcula métricas adicionales, como el número total de controles no conformes o el porcentaje de incumplimientos por categoría, y las añade al mismo objeto DataFrame. A continuación, ofrece dos opciones de exportación: un fichero CSV, que mantiene la estructura tabular y es directamente navegable en Excel, y un fichero JSON, útil para integraciones con otras herramientas o APIs. La descarga de ambos formatos se realiza desde la interfaz web mediante enlaces generados dinámicamente; al pulsarlos, Flask sirve los archivos en memoria, sin escribirlos previamente en disco.

En cuanto a la implementación, el Generador de Diagnóstico está encapsulado en una función generate_diagnostic(input_data) que recibe un diccionario o DataFrame con los resultados del primer script. El uso de Pandas simplifica operaciones como filtrado de filas (df[df['Estado']=="No cumple"]) o agrupación por categoría (df.groupby('Categoría').count()). Además, el módulo incluye validaciones para asegurar que los datos de entrada cumplen el formato esperado, si falta alguna columna obligatoria, muestra un mensaje de error claro en la web.

Gracias a este script, la PYME dispone de un informe detallado donde no solo se identifican los controles pendientes, sino que cada uno de ellos viene acompañado de una recomendación práctica, facilitando la planificación de acciones y sirviendo de insumo directo para el resto de las fases automatizadas de la plataforma.

4.5.3 Script 3: Generador de Políticas de Seguridad

El Generador de Políticas de Seguridad automatiza la creación de documentos base que describen las políticas necesarias para cumplir con los controles organizativos y técnicos del ENS. Este módulo utiliza plantillas en formato YAML almacenadas en el directorio /app/templates/, donde cada archivo representa una política estándar (por ejemplo, “Política de Control de Accesos”, “Política de Copias de Seguridad” o “Política de Gestión de Incidentes”) con campos variables como el nombre de la organización, las responsabilidades asignadas o los plazos de revisión.

Al ejecutar el script desde la interfaz web, el usuario selecciona qué políticas desea generar y completa un pequeño formulario con parámetros clave, como nombre de la empresa, roles de usuario o caducidad de contraseñas. El módulo carga la plantilla correspondiente con PyYAML, reemplaza las variables definidas ({{company_name}}, {{password_expiry_days}}, etc.) y volca el resultado en un fichero YAML descargable o en un documento markdown.

Internamente, la función principal generate_policy(policy_type, params) abstrae el proceso de lectura, parametrización y escritura de archivos, de manera que añadir nuevas plantillas o modificar las existentes no requiere cambios en el código. El script también valida que los valores proporcionados por el usuario respetan formatos básicos (por ejemplo, que los días de expiración sean un número positivo) y alerta al usuario en caso de discrepancias.

De este modo, la PYME obtiene rápidamente un conjunto de documentos de política coherentes con el ENS 2022, listos para ser adaptados y aprobados en su entorno interno, lo que reduce significativamente el esfuerzo manual asociado a la redacción de estos textos y asegura que ningún apartado crítico quede olvidado.

4.5.4 Script 4: Auditor de Permisos y Accesos

El Auditor de Permisos y Accesos se encarga de inspeccionar la configuración de cuentas y derechos en los sistemas objetivo para detectar posibles vulnerabilidades relacionadas con el acceso no autorizado o la sobreexposición de recursos. Al pulsar el botón correspondiente en la interfaz web, el usuario recibe un enlace de descarga que incluye dos variantes del script, una en PowerShell para entornos Windows y otra en Bash para sistemas Linux, junto con un archivo README que detalla los prerequisitos, las dependencias necesarias y las instrucciones de ejecución paso a paso.

Una vez ejecutado localmente, el script recopila información clave: en Windows utiliza cmdlets como Get-LocalUser, Get-LocalGroupMember y Get-Acl para identificar cuentas con contraseñas sin expiración, miembros no autorizados en grupos privilegiados y permisos excesivos en rutas críticas; mientras que la versión Bash recorre /etc/passwd y /etc/shadow, busca archivos con permisos de escritura global (find / -perm -o+w) y revisa servicios habilitados con herramientas como systemctl. El resultado se vuelca en un fichero de salida (normalmente un CSV o log estructurado) que el usuario debe cargar de nuevo en la sección de “Subir resultados” de la web.

Al recibir este fichero, la plataforma parsea automáticamente las entradas y presenta en pantalla las alertas detectadas. Por ejemplo, una entrada podría indicar “Usuario

‘invitado’ con privilegios de Administrador” o “Archivo /etc/shadow accesible por usuarios no root”. Además, el script web asocia a cada alerta una recomendación genérica para corregir la configuración y permite descargar un informe consolidado con todas las incidencias halladas.

Este enfoque mixto, ejecución local para acceder a información sensible del sistema y análisis centralizado en la web, combina la potencia de los entornos nativos (sin necesidad de instalar agentes adicionales) con la conveniencia de recibir los resultados en un formato homogéneo y accesible, contribuyendo así al cumplimiento de los controles de acceso definidos en el ENS.

4.5.5 Script 5: Detector de Incidentes

El Detector de Incidentes ENS proporciona la capacidad de vigilancia continúa requerida para observar eventos sospechosos y anomalías en los sistemas de la PYME. Este script, similar al Auditor de Permisos, se distribuye como un paquete descargable junto con un README detallado que explica cómo instalar dependencias, por ejemplo, Winlogbeat para Windows o Filebeat para Linux, y configurar la recolección de logs hacia un destino accesible (un fichero JSON o un endpoint HTTP interno).

Una vez desplegado el agente correspondiente en la máquina local, el Detector puede ejecutarse en modo demonio para leer en tiempo real los registros del sistema operativo y de aplicaciones críticas. En Linux, procesa archivos como /var/log/auth.log o /var/log/syslog, mientras que en Windows consume los eventos de seguridad del Visor de Eventos a través de pywin32. El script aplica reglas de detección predefinidas, por ejemplo, múltiples intentos fallidos de login, creación de cuentas administrativas no autorizadas o cambios en configuraciones clave, y ante cada coincidencia genera un registro de incidente que incluye timestamp, descripción del evento y gravedad.

Estos registros se vuelcan en un fichero estructurado (CSV o JSON) que el usuario sube posteriormente a la plataforma web mediante la misma sección de “Subir resultados” empleada por el script de auditoría. La aplicación Flask parsea entonces los incidentes y presenta un panel donde se listan los eventos detectados, agrupados por tipo y nivel de gravedad, y se ofrecen recomendaciones para el análisis forense o la respuesta inicial (por ejemplo, “Revisar logs de autenticación y cambiar contraseñas de cuentas comprometidas”).

El Detector de Incidentes ENS cumple así los requisitos de los controles de monitorización y respuesta del ENS 2022, proporcionando una herramienta sencilla de desplegar que centraliza la supervisión de logs y facilita la generación de evidencias para auditorías y revisiones periódicas.

4.5.6 Script 6: Simulador de Ataques

El Simulador de Ataques ENS está diseñado para evaluar de manera proactiva la resiliencia de los sistemas de la PYME frente a amenazas comunes de red y aplicaciones. Este módulo se distribuye, al igual que los scripts de auditoría y detección, como un paquete descargable que incluye el código fuente Python, un archivo requirements.txt con las dependencias (principalmente Scapy) y un README con ejemplos de ejecución.

Una vez instalado en un entorno de laboratorio o red local controlada, el simulador se lanza desde la línea de comandos con parámetros de destino (dirección IP o rango de subred) y tipo de prueba. Entre las funcionalidades implementadas destacan:

- **Escaneo de puertos TCP/UDP:** envía paquetes SYN a una lista configurable de puertos y analiza las respuestas SYN-ACK o RST para identificar servicios abiertos.
- **Pruebas de servicios específicos:** al detectar servicios como FTP o Telnet, el script intenta conexiones anónimas o realiza banner grabbing para comprobar configuraciones por defecto.
- **Ataques ligeros de denegación de servicio:** envía ráfagas de paquetes ICMP o TCP para comprobar la capacidad de respuesta y los límites de rate-limiting.
- **Simulación de MITM básico:** en Linux, emplea ARP spoofing con Scapy para inyectar tráfico y verificar si se detectan cambios inesperados en los logs de red.

Durante la ejecución, cada prueba genera un registro estructurado con el objetivo, el resultado (vulnerable/seguro) y detalles técnicos (puerto, banner, cantidad de paquetes enviados). El usuario descarga el fichero de salida y lo sube a la plataforma web, donde, mediante la sección de “Subir resultados”, el sistema interpreta los hallazgos y los integra en el informe general. De ese modo, se comprueba no solo que los controles de configuración están correctamente aplicados, sino que también resisten ataques reales de baja complejidad.

4.5.7 Script 7: Optimizador de Recursos

El Optimizador de Recursos ENS facilita la planificación estratégica al estimar el esfuerzo y los costes asociados a la implementación de los controles del ENS. A diferencia de los módulos anteriores, este script se ejecuta completamente desde la interfaz web, sin necesidad de desembarcarlo localmente. Al acceder a la sección “Optimización de Recursos”, el usuario completa un formulario con datos como el número de empleados, el presupuesto disponible y el nivel de madurez inicial obtenido en el diagnóstico.

Internamente, el script emplea un conjunto de reglas heurísticas, definidas en un fichero de configuración, para realizar cálculos sencillos: por ejemplo, establece que cada usuario adicional requiere X horas de formación o que un análisis de riesgos completo demanda Y horas de consultoría externa. Con Pandas y NumPy, consolida estos valores en un DataFrame, añade márgenes de contingencia y genera una tabla con estimaciones desglosadas por categoría de control (organizativos, operacionales, protección).

La salida se presenta directamente en la misma vista web: una tabla donde el usuario puede ver el coste total y ordenado por prioridad, y un botón para descargar el plan en CSV o PDF. De esta manera, la PYME dispone de un plan de acción detallado, que sirve tanto para asignar recursos internos como para justificar inversiones ante la dirección o posibles proveedores externos. Este script garantiza que el proyecto se mantenga alineado con la capacidad de la organización, cerrando la brecha entre los requisitos técnicos y la viabilidad operativa.

4.5.8 Script 8: Generador de Informes

El Generador de Informes ENS es el módulo final que reúne toda la información producida por los demás scripts y la presenta en un documento profesional listo para auditoría. Desde la sección “Informes”, el usuario puede elegir el rango de fechas o fases específicas a incluir; el script entonces carga los ficheros CSV/JSON de los diagnósticos, auditorías, incidentes, simulaciones y planes de recursos desde el directorio de resultados.

Utilizando Matplotlib, crea gráficos de barras, pastel y líneas que ilustran la evolución del cumplimiento, el número de incidentes detectados y la comparación de estados antes y después de aplicar mejoras. Con ReportLab (o FPDF), inserta estos gráficos junto con textos generados dinámicamente, resúmenes ejecutivos, tablas de controles cumplidos/no cumplidos y recomendaciones finales, en un PDF multi-página. El informe resultante combina elementos visuales y narrativos, ofreciendo tanto una visión global de la adecuación al ENS como datos detallados para cada control.

El PDF se ofrece como descarga directa en la interfaz, y además se almacena automáticamente en el volúmen de resultados para futuras consultas. De este modo, la plataforma cierra el ciclo de automatización, proporcionando a la PYME no solo herramientas de diagnóstico y verificación, sino también la documentación formal necesaria para demostrar el cumplimiento del ENS 2022 nivel Alto ante auditores externos.

5. Pruebas, Evaluación y Resultados

5.1 Descripción del entorno de prueba simulado (infraestructura y datos)

Para validar los scripts que interactúan directamente con los sistemas (Scripts 4, 5 y 6) se montó un laboratorio seguro compuesto por dos máquinas virtuales aisladas y conectadas mediante una red interna denominada labnet, de modo que cualquier prueba queda confinada y puede repetirse con exactitud sin afectar al anfitrión. La primera VM, llamada Win-Audit, corre Windows 11 (64 bit) sobre 4 GB de RAM, 2 vCPUs y un disco virtual de 40 GB. En ella se instaló PowerShell 7, las Guest Additions de VirtualBox (para carpetas compartidas) y Winlogbeat, configurado para enviar eventos al servidor Logstash de la segunda VM. La segunda, Linux-Audit, ejecuta Ubuntu 22.04 Server sin interfaz gráfica, con 2 GB de RAM, 2 vCPUs y un disco de 20 GB; en ella residen Python 3.10, Filebeat, Logstash, Scapy y OpenSSH, actuando como detector central de incidentes y nodo objetivo de ataques.

Ambas VM comparten carpetas a través de VirtualBox: en Windows vuelcan los logs y CSV de PowerShell en C:\shared, mientras que en Linux usan /home/audit/shared, facilitando la transferencia de resultados al sistema host para cargarlos luego en la plataforma web. La red interna “labnet” conecta exclusivamente Win-Audit y Linux-Audit: en Windows se configuró un adaptador “labnet” para la comunicación con Linux y un NAT para actualizaciones ocasionales; en Linux, únicamente el “labnet” y SSH.

Antes de comenzar las pruebas, se instaló Guest Additions en ambas VM, se creó la red interna y se verificó la conectividad con ping. Winlogbeat en Windows apuntó a IP_Linux:5044, mientras Filebeat en Linux se configuró en /etc/filebeat/filebeat.yml para enviar /var/log/auth.log y /var/log/syslog a Logstash, cuyo pipeline básico reside en

/etc/logstash/conf.d/. Una prueba de verificación consistió en generar un evento de cinco intentos fallidos de inicio de sesión en menos de dos minutos en Win-Audit y comprobar, mediante curl http://localhost:9600/_node/pipelines, que Logstash lo procesaba correctamente.

Dentro de este entorno, el Auditor de Permisos y Accesos (Script 4) se ejecutó en Win-Audit y, tras por ejemplo crear y eliminar cuentas sin contraseña o modificar ACLs en carpetas críticas, se confirmó que identificaba cada vulnerabilidad. El Detector de Incidentes (Script 5) se puso en marcha en Linux-Audit, recogiendo los beats de Windows y Linux para registrar eventos como los bloqueos de login. Finalmente, el Simulador de Ataques (Script 6) empleó Scapy en Linux-Audit para enviar paquetes SYN a puertos TCP tanto a Win-Audit como a la propia VM, comprobando la detección de servicios abiertos. Gracias a esta configuración, cada script pudo validarse de forma aislada y coherente, produciendo los ficheros de salida necesarios para su posterior análisis en la plataforma y garantizando un entorno controlado y reproducible.

5.2 Resultados de la evaluación inicial (Script 1 y 2)

En esta fase se valida el diagnóstico automático de cumplimiento a través de los Scripts 1 y 2, mostrando al usuario un recorrido desde el formulario inicial hasta el informe detallado. Para ello, la plataforma web sigue este flujo:

- **Formulario dinámico de controles**
  Al acceder a la sección “Evaluar Cumplimiento ENS Nivel Alto”, el usuario encuentra un listado de los 62 controles técnicos (código, descripción) importados del fichero Excel. Cada control permite tres respuestas:
  1. Cumple completamente
  2. Cumple parcialmente (con porcentaje)
  3. No cumple

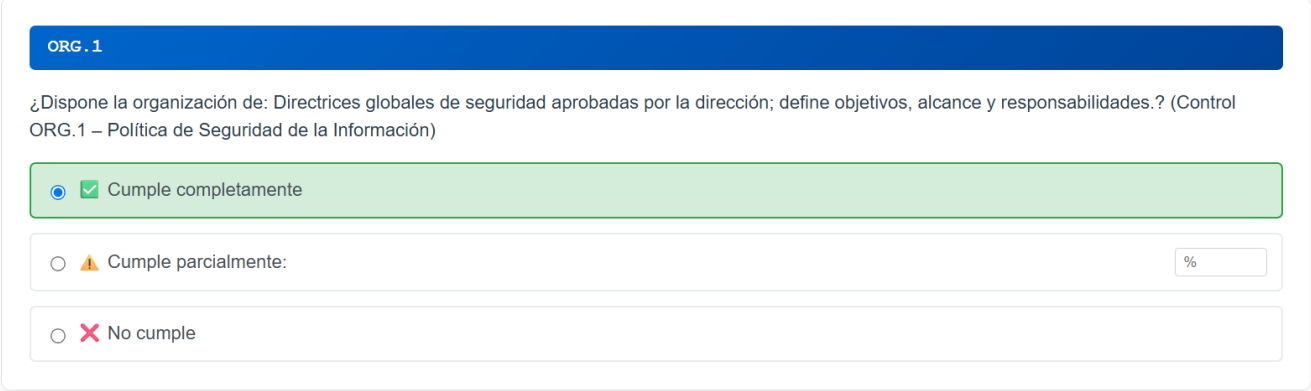

Ilustración 2: Formulario dinámico de evaluación de cumplimiento ENS.

- **Resumen agregado de cumplimiento**
  Tras enviar el formulario, el Evaluador (Script 1) calcula:
  1. Porcentaje de controles que cumplen totalmente

2. Porcentaje de controles que cumplen parcialmente
3. Porcentaje de controles no cumplidos

Estadísticas generales:
Controles que cumplen totalmente: 40%
Controles que cumplen parcialmente: 13%
Controles que no cumplen: 47%

Tabla 1: Resumen de porcentaje de controles cumplidos, parciales y no cumplidos

A continuación, el sistema desglosa estas métricas por familia de control (OP.ACC, OP.EXP, ORG, etc.), tal y como muestra la Figura 5.2:

Familia	**Total controles**	**Cumple (%)**	**Parcial (%)**	**No cumple (%)**
MP.COM	4	0%	24%	76%
MP.EQ	4	75%	0%	25%
MP.IF	7	29%	24%	47%
MP.INFO	6	50%	17%	33%
MP.PER	4	0%	30%	70%
MP.S	4	75%	0%	25%
OP.ACC	6	67%	6%	27%
OP.CONT	4	75%	0%	25%
OP.EXP	9	22%	14%	64%
OP.EXT	4	25%	12%	63%
OP.NUB	1	100%	0%	0%
OP.PL	5	40%	11%	49%
ORG	4	50%	25%	25%

Tabla 2: Extracto de estados por control

- **Detalle de cada control**
  Para quienes deseen el análisis fino, el script lista cada control junto con su estado y porcentaje:

Código	Estado
ORG.1	Cumple completamente
ORG.2	Cumple parcialmente (45 %)
ORG.3	No cumple
ORG.4	No cumple
…	…

Tabla 3: estado de cumplimiento control por control.

En la web esta información aparece como lista con colores (verde/ámbar/rojo), pero para la memoria mostramos aquí un fragmento tabular que facilita la lectura y comparación.

- **Informe estructurado y recomendaciones**
  A continuación, el Generador de Diagnóstico ENS (Script 2) toma estos resultados y, gracias a un diccionario interno de recomendaciones, construye un informe detallado. Cada fila incluye:

  1. Código ENS
  2. Estado (“Cumple”, “Parcial (X %)” o “No cumple”)
  3. Recomendación asociada

Código ENS	Estado	Recomendación
MP.COM.1	Cumple	Excelente. La integridad de las comunicaciones está garantizada.
MP.COM.2	No cumple	CRÍTICO: Implemente cifrado en comunicaciones (TLS 1.2+, VPN, túneles seguros). Use el Script 6 para detectar servicios no cifrados expuestos.
MP.IF.2	Cumple	Correcto. La protección física de instalaciones es adecuada.
OP.ACC.3	No cumple	ALTO: Implemente segregación de funciones para evitar conflictos de interés. Separe roles de administración, operación y control. Use el Script 4 para identificar usuarios con privilegios excesivos.
OP.EXP.6	No cumple	ALTO: Implemente gestión centralizada de logs (SIEM, syslog, retención). Use el Script 5 para detectar

		eventos de seguridad y el Script 7 para estimar costos de SIEM.
MP.EQ.2	Parcial (15%)	Acción necesaria: Complete el cifrado de discos y políticas de uso de equipos portátiles.
OP.EXP.5	Cumple	Correcto. La gestión de vulnerabilidades funciona adecuadamente.
…	…	…

Tabla 4: Fragmento de diagnóstico con recomendaciones

- **Descarga de resultados**
  Finalmente, la interfaz ofrece botones para exportar el diagnóstico en CSV o JSON. Estos archivos contienen la misma estructura tabular y pueden abrirse en herramientas de análisis o integrarse en otros sistemas de gestión.

5.3 Cobertura de controles ENS lograda con cada script

La metodología automatizada cubre de manera integral las principales familias de controles técnicos del ENS 2022 nivel Alto, asignando a cada script un conjunto de medidas sobre las que actúa directamente y dejando solamente aquellas que por su naturaleza organizativa o de gobernanza requieren intervención manual.

En primer lugar, los Scripts 1 y 2 (Evaluador de Cumplimiento y Generador de Diagnóstico) proporcionan un diagnóstico completo de los 62 controles técnicos. Gracias a ellos, la plataforma detecta de forma automática si cada medida está cumplida, parcialmente cumplida o no cumplida, y genera un informe estructurado con recomendaciones para abordar las carencias. Esta cobertura inicial abarca todos los controles definidos, sirviendo como base para priorizar acciones.

A continuación, el Script 3 (Generador de Políticas de Seguridad) acelera la creación de la documentación necesaria para los controles organizativos (familia ORG). Mediante plantillas YAML parametrizables, se generan políticas de control de accesos, gestión de incidentes, copias de seguridad y otras normas internas, garantizando que cada aspecto organizativo cuente con un borrador aprobado por la dirección.

Los aspectos puramente técnicos quedan cubiertos por tres módulos especializados:

- Script 4 (Auditor de Permisos y Accesos) examina la configuración de cuentas y directorios en Windows y Linux, ayudando a implementar y verificar los controles de la familia OP.ACC (gestión de identidades, privilegios y segregación de funciones).
- Script 5 (Detector de Incidentes ENS) ofrece monitorización continua de eventos y registros, alineándose con los controles de la familia OP.EXP (registro y seguimiento de actividad). Al procesar logs de sistemas y aplicaciones críticas en tiempo real, cumple con los requisitos de detección y respuesta rápida ante incidentes.
- Script 6 (Simulador de Ataques ENS) pone a prueba la resiliencia de la red y servicios mediante escaneos de puertos, pruebas de autenticación anónima y ataques ligeros

de denegación de servicio. Con ello, cubre los controles de pruebas de penetración y robustez de la familia OP.NET/IP, asegurando que las defensas técnicas resistan escenarios reales de amenaza.

Por último, el Script 7 (Optimizador de Recursos ENS) se encarga de la familia OP.PL (planificación y dimensionamiento), calculando el esfuerzo en horas de formación, consultoría y adquisición de herramientas necesario para implementar los controles detectados como pendientes. Y el Script 8 (Generador de Informes ENS) unifica evidencias de todas las fases, diagnóstico, políticas, auditorías, monitorización, simulaciones y planificación, en un informe PDF profesional, cumpliendo con los requisitos de documentación y presentación exigidos por el Anexo III del real decreto.

En conjunto, esta metodologia de ocho scripts automatiza la gran mayoría de los controles técnicos del ENS nivel Alto, desde el diagnóstico inicial hasta la entrega de evidencias formales, reduciendo drásticamente el esfuerzo manual y facilitando la auditoría externa. Sólo los controles más organizativos o de gestión humana (por ejemplo, la creación de comités de seguridad o formación presencial de todo el personal) quedan fuera del ámbito de automatización, pero se incorporan como recomendaciones en los informes finales.

5.4 Generación de políticas de seguridad (Script 3)

El módulo de Generación de Políticas de Seguridad está integrado en la sección “Políticas” de la plataforma web. Su propósito es agilizar la creación de documentos base alineados con los controles organizativos del ENS (familia ORG), permitiendo al usuario parametrizar las plantillas y descargar los resultados en formatos listos para revisión.

Flujo de trabajo en la interfaz

- **Selección de políticas**
  Al entrar en la pantalla, el usuario se encuentra con un listado de casillas en las que puede marcar cuáles de las políticas desea generar. Cada entrada muestra el título de la política y el código ENS asociado (por ejemplo, “Política de Seguridad (ORG.1)”, “Normativa de Seguridad (ORG.2)”, “Autenticación Usuarios Externos (OP.ACC.5)”, etc.).

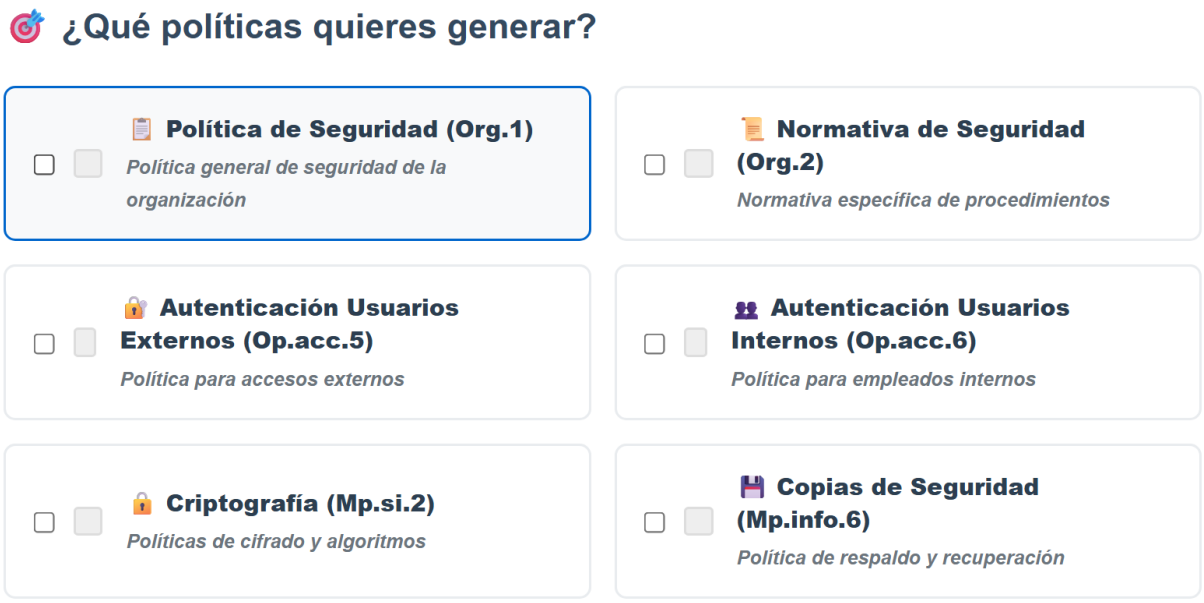

Ilustración 3: Selección de políticas a generar en la interfaz.

- **Parámetros comunes**
  Justo debajo, aparece un bloque de “Parámetros comunes”, con dos campos obligatorios:
  1. Nombre de la PYME
  2. Responsable de Seguridad

Estos valores se incorporan automáticamente en todas las plantillas seleccionadas, garantizando coherencia en la documentación.

- **Parámetros específicos por política**
  Tras seleccionar una política, la plataforma despliega dinámicamente los campos de parametrización propios de ese documento. Por ejemplo, al elegir “Autenticación Usuarios Externos (OP.ACC.5)”, se muestran:
  1. Longitud mínima de contraseña (por defecto “12”)
  2. Rotación de contraseña en días (por defecto “90”)
  3. Método de 2FA (desplegable: SMS, App, Email)

Ilustración 4: Parámetros específicos para la política de autenticación externa y copias de Seguridad.

De forma similar, al activar “Criptografía (MP.SI.2)” o “Copias de Seguridad (MP.INFO.6)”, aparecen nuevos controles de algoritmo, frecuencia de backups y rutas de destino.

- **Generación y descarga**
  Una vez completados los parámetros, el usuario pulsa “Generar políticas”. La aplicación procesa las plantillas YAML con PyYAML, reemplaza variables y presenta en pantalla una lista de cada política con enlaces de descarga en dos formatos:
  1. YAML (.yml)
  2. Texto plano (.txt)

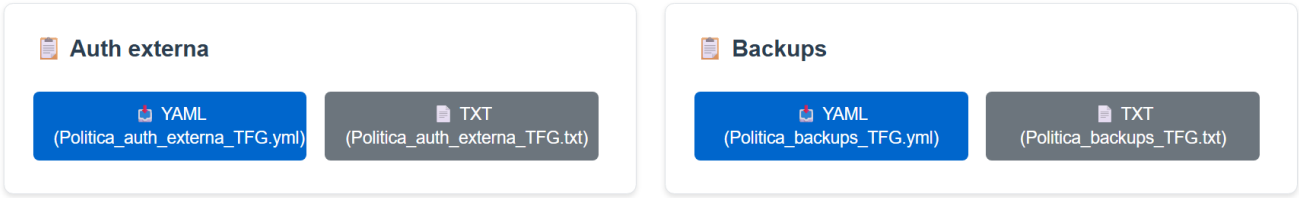

Ilustración 5: Enlaces de descarga de las políticas generadas.

Con este enfoque, el Script 3 ahorra horas de redacción manual al proporcionar un punto de partida estructurado para cada política requerida por el ENS. Al centralizar los parámetros en la web y automatizar la generación, se asegura que ningún apartado crítico quede olvidado y se facilita la revisión y aprobación por parte de la dirección.

5.5 Auditoría de Permisos y Accesos (Script 4)

El módulo de Auditoría de Permisos y Accesos es clave para garantizar el principio de mínimo privilegio en una organización. A diferencia de otros scripts que se ejecutan totalmente en la web, este requiere interacción local con los sistemas objetivo. A continuación se detalla todo el proceso de pruebas, los tipos de vulnerabilidades que cubre y cómo el usuario interactúa con la plataforma para obtener un informe claro y accionable.

Para maximizar la seguridad y evitar impactos en sistemas productivos, las pruebas de este script se llevaron a cabo en dos máquinas virtuales:

- **Win-Audit (Windows 11)**
    1. Se descargó el script auditor_windows.ps1 desde la sección correspondiente de la web.
    2. Se abrió PowerShell 7 como Administrador (requisito indispensable para poder inspeccionar cuentas y ACL).
    3. El script emitió mensajes en consola a medida que comprobaba:
        1. Op.acc.1 Identificación: detección de cuentas sin contraseña, configuraciones de expiración.
        2. Op.acc.2 Requisitos de acceso: listados de miembros de grupos privilegiados (Administradores, Escritorio Remoto, Operadores de Backup).
        3. Op.acc.4 Gestión de derechos: revisión de cuentas deshabilitadas vs. no eliminadas, expiración de cuentas.
        4. Op.acc.5–6 Autenticación: estado de RDP, políticas de contraseña, bloqueo por intentos fallidos.
- **Linux-Audit (Ubuntu 22.04)**
    1. Se descargó auditor_linux.sh y se otorgó permiso de ejecución (chmod +x).
    2. Se ejecutó con privilegios de root:
    3. El script recorrió:
        1. Op.acc.1: UID duplicados, shells no válidas.
        2. Op.acc.2: usuarios en sudo, adm, wheel.
        3. Op.acc.4: usuarios bloqueados y expirados.
        4. Op.acc.5–6: configuración de sshd_config y login.defs, verificaciones PAM.

Ambas ejecuciones produjeron un fichero de texto con secciones claramente delimitadas por control ENS, lo que permitió una posterior carga de resultados sin ambigüedades.

Ejemplo de logs al ejecutar el script en Windows:

```
==============================
Control Op.acc.1 – Identificación
==============================
Verificar si hay usuarios habilitados sin requerir contraseña
Todos los usuarios habilitados requieren contraseña.
Verificar si hay usuarios con configuración sospechosa (contraseña que nunca expira o expiración definida)
Usuario con contraseña que nunca expira: dagon
==============================
Control Op.acc.2 – Requisitos de acceso
==============================
Verificar si hay usuarios en grupos privilegiados (Administradores, Usuarios de escritorio remoto, Operadores de copia de seguridad)
Usuario con privilegios en grupo Administradores – DESKTOP-GFMP174\pruebaWin Usuario con privilegios en grupo Administradores – DESKTOP-GFMP174\dagon
No hay miembros en el grupo Usuarios de escritorio remoto.
No hay miembros en el grupo Operadores de copia de seguridad.
==============================
Control Op.acc.4 – Proceso de gestión de derechos de acceso
==============================
Verificar cuentas deshabilitadas (bloqueadas pero no eliminadas)
Cuenta deshabilitada: Invitado
Cuenta deshabilitada: pruebaDisabled
Verificar usuarios con cuenta expirada
(usando WMI)
No hay cuentas locales expiradas.
…
```

Ejemplo de logs al ejecutar el script en Linux:

```
==============================
Control Op.acc.1 – Identificación
==============================
Verificar si hay usuarios duplicados (mismo UID para distintos nombres)
UID duplicado: 0 pertenece a: root,eviladmin
Verificar si hay usuarios con shell no válido (por ejemplo /bin/false o /usr/sbin/nologin sin justificación)
Usuario con shell sospechosa: strangeuser -> /bin/false
==============================
Control Op.acc.2 – Requisitos de acceso
==============================
Verificar si hay usuarios en grupos con privilegios elevados (sudo, adm, wheel, shadow)
Usuario con privilegios sudo: danigonzalezbraza
Usuario con privilegios sudo: overpowered
Usuario con privilegios elevados en grupo adm: syslog
Usuario con privilegios elevados en grupo adm: danigonzalezbraza
No hay usuarios en el grupo shadow
==============================
Control Op.acc.4 – Proceso de gestión de derechos de acceso
==============================
Verificar si existen usuarios bloqueados (cuentas deshabilitadas pero no eliminadas)
Cuenta bloqueada: strangeuser
Cuenta bloqueada: overpowered
Cuenta bloqueada: blockeduser
...
```

Una vez obtenidos los logs locales, el usuario accede de nuevo a la plataforma y, en la misma sección de “Auditoría de Permisos y Accesos”, utiliza el formulario de “Subir resultado de auditoría” para enviar el archivo (auditoria_permisos_win.txt o auditoria_permisos_linux.txt). La web realiza los siguientes pasos:

- **Validación de formato**
    1. Comprueba que el archivo contiene bloques identificables de controles (p. ej. encabezados con “Control Op.acc.X”).
    2. En caso de error, muestra un mensaje explicativo (“El fichero no contiene la sección ‘Control Op.acc.’”).
- **Parseo y extracción de alertas**
    1. Divide el contenido por secciones, cada una correspondiente a un control ENS.

2. Extrae las líneas de vulnerabilidad (por ejemplo, “Usuario con contraseña que nunca expira: Administrador”).
3. Mapea cada sección al control ENS correspondiente.

- **Enriquecimiento con contexto**
  1. Para cada alerta, añade una sección “¿Qué significa?” explicando la importancia del control.
  2. Agrega recomendaciones prácticas (por ejemplo, “Forzar expiración de contraseñas en próximo inicio”, “Revocar usuarios innecesarios”).
- **Presentación interactiva**
  1. La plataforma muestra un listado de alertas formateadas en tarjetas: el título es “Op.acc.X – Descripción breve”, seguido de la alerta original, la explicación y recomendaciones.

Ejemplos de casos de prueba:

- **Escenario Windows estándar**
  Tras crear cuentas de prueba en el grupo Administradores y dejar la contraseña de “dagon” sin expirar, el auditor local detectó múltiples miembros privilegiados. Al subir auditoria_permisos_win.txt, la web produjo tarjetas como:

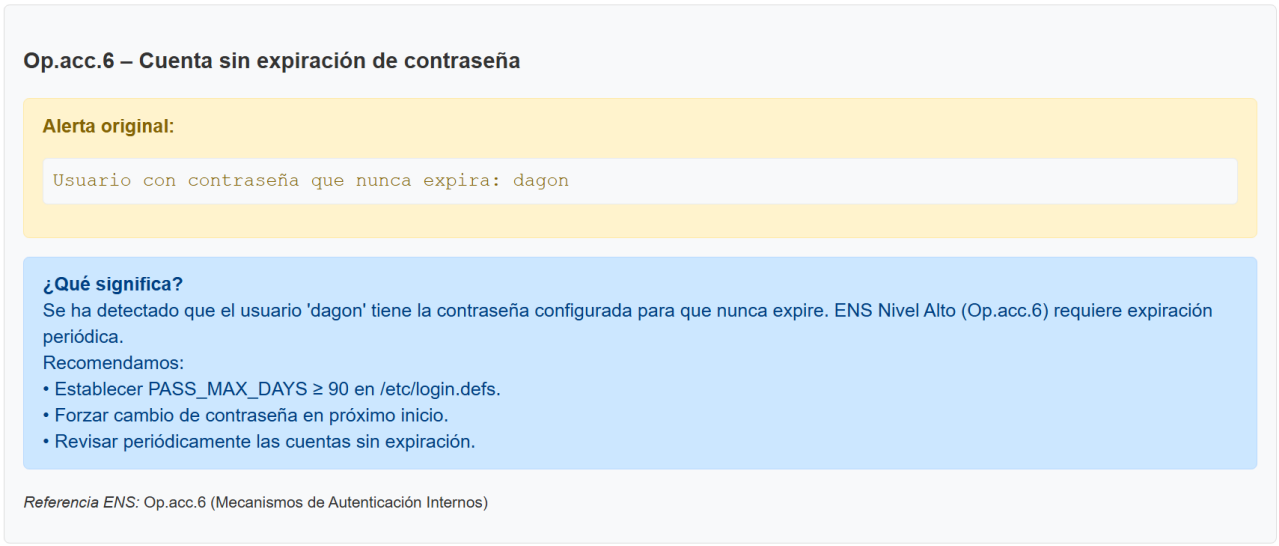

Ilustración 6: Captura del proceso de subida de resultados de la auditoria de permisos y accesos, cuenta sin expiración de contraseña.

- **Escenario Linux vulnerado**
  Con un UID duplicado y una cuenta “strangeuser” con shell /bin/false, el archivo auditoria_permisos_linux.txt incluyó:

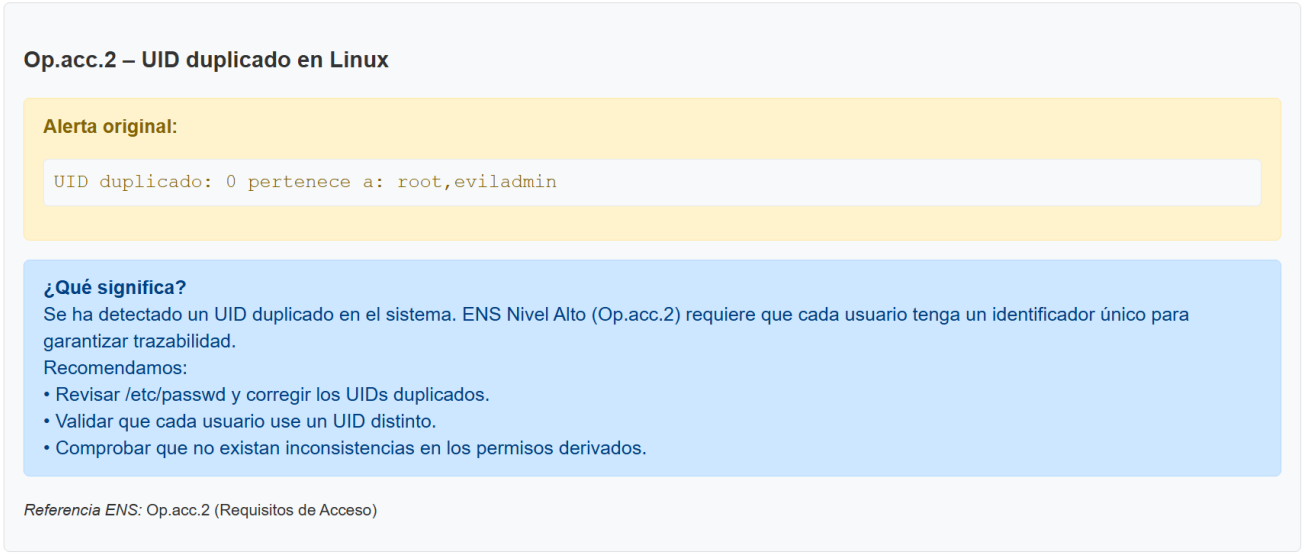

Ilustración 7: Captura del proceso de subida de resultados de la auditoria de permisos y accessos, UID duplicado en Linux.

Este enfoque completo, ejecución en VM, subida de logs y análisis en web, garantiza que el Script 4 no solo comprueba la existencia de vulnerabilidades, sino que traduce resultados técnicos en acciones concretas accesibles para responsables de TI con distintos niveles de experiencia.

5.6 Validación de monitorización e incidentes (Script 5)

El Detector de Incidentes ENS es el módulo encargado de la vigilancia continua de los sistemas, recopilando y analizando eventos de seguridad para alertar de anomalías en tiempo real. Su integración con la plataforma web sigue el mismo patrón de descarga local y subida de resultados:

- **Descarga y configuración**
  Desde la sección “Monitorización” de la web, el usuario obtiene un único paquete detector_unificado.py junto a un README que detalla:
  1. La instalación de Winlogbeat en Windows y Filebeat en Linux.
  2. La configuración de cada beat para enviar logs a Logstash.
  3. El comando para iniciar Logstash en Linux, preparando la cercanía de incidentes.json.
- **Ejecución en laboratorio**
  1. Windows: Winlogbeat recoge eventos del Visor de Eventos y los envía, en formato JSON, al fichero /var/log/incidentes.json de la VM Linux.
  2. Linux: Filebeat monitoriza /var/log/auth.log y otros logs críticos, enviándolos al mismo JSON.

3. En Linux-Audit, se lanza el script detector_unificado.py en modo demonio (nohup python3 detector_unificado.py &), que permanece “tail -f” sobre incidentes.json y, además de incidentes Windows/Linux, verifica integridad de archivos críticos (p. ej. /etc/passwd).

- **Subida y visualización de alertas**
  Al cabo de unos minutos de ejecución, el Detetor genera múltiples alertas, cada una impresa en consola en rojo y guardada en /var/log/incidentes_alertas.log junto al JSON subyacente. Ejemplos de salidas capturadas:

```
[ALERTA] [2025-06-02 23:06:34] Intento SSH con usuario inválido 'ghost' desde 127.0.0.1 detectado en Linux.
{
  "tipo": "Linux_InvalidUser",
  "usuario": "ghost",
  "origen": "127.0.0.1",
  "timestamp": "2025-06-02 23:06:34"
}
[ALERTA] [2025-06-02 23:06:51] Creación de cuenta local Windows 'TFGprueba' detectada → INCIDENTE (4720).
{
  "tipo": "Windows_CreacionCuenta",
  "event_id": 4720,
  "usuario": "TFGprueba",
  "timestamp": "2025-06-02 23:06:51"
}
[ALERTA] [2025-06-02 23:07:09] 5 fallos de logon Windows para usuario 'Administrador' desde ::1 en 2 minutos → POSIBLE FUERZA BRUTA (4625)
{
  "tipo": "Windows_FuerzaBruta",
  "event_id": 4625,
  "usuario": "Administrador",
  "origen": "::1",
  "num_fallos": 5,
  "ventana_min": 2,
  "timestamp": "2025-06-02 23:07:09"
}
```

El usuario descarga el fichero incidentes_alertas.log o una porción filtrada de incidentes.json, y lo sube mediante el formulario de “Subir resultados de monitorización”. La web procesa cada JSON, extrae las claves (tipo, event_id, usuario, etc.), y presenta en pantalla una lista de tarjetas de alerta:

Ilustración 8: Subida y visualizacion de alertas de incidentes, fuerza bruta Windows.

Ilustración 9: Subida y visualizacion de alertas de incidentes, SSH con usuario invalido.

- **Cobertura de controles OP.EXP**
  Gracias a esta validación, el Detector cumple los controles de la familia OP.EXP (registros y respuesta a incidentes), incluyendo:
  1. Detección de intentos de fuerza bruta (4625, Accepted Password).
  2. Creación y habilitación de cuentas (4720, 4722).
  3. Eventos de firewall (5152, 5156).
  4. Fallos SSH e intentos de usuario inválido.

5. Integridad de archivos críticos (alertas de modificación).

Este enfoque demuestra cómo el Script 5 no solo identifica incidentes relevantes en tiempo real, sino que convierte un flujo crudo de JSON en un conjunto de alertas accionables, un requisito esencial para auditorías ENS y para mantener la seguridad operativa de la PYME.

5.7 Eficacia del Simulador de Ataques (Script 6)

El Simulador de Ataques ENS permite a una PYME evaluar de manera proactiva la exposición de sus servicios de red y la resistencia frente a ataques comunes, integrándose con la plataforma web del mismo modo que los scripts de auditoría y monitorización. A continuación, se describen en detalle sus capacidades, la forma de interacción y los resultados obtenidos en laboratorio.

Descarga y parámetros de ejecución
En la sección “Simulación de Ataques” de la web, el usuario encuentra un único enlace para descargar simulador_ataques.py junto al README con instrucciones completas. El script acepta los siguientes parámetros:

- IP de destino (obligatorio): dirección del host o rango de subred a analizar.
- --tcp-ports p1 p2 …: puertos TCP concretos (p. ej. 22 80 443).
- --full-tcp: escaneo completo de puertos TCP 1–1024.
- --udp-ports u1 u2 …: puertos UDP específicos (p. ej. 53 67 123).
- --icmp-flood N: envío de N paquetes ICMP para medir la respuesta de ping.
- Opciones adicionales como --ftp-anon, --telnet-banner o --http-banner permiten probar servicios específicos si los puertos correspondientes están abiertos.

Flujo de prueba y entornos
Las pruebas se realizaron en dos escenarios dentro de la red interna labnet:

1. Objetivo Windows (192.168.100.2)

*sudo python3 simulador_ataques.py 192.168.100.2 *

--tcp-ports 445 80 --udp-ports 53 67 123 --icmp-flood 8

El simulador ejecuta sucesivamente:

```
[2025-06-4 00:10:15] Iniciando simulador contra 192.168.100.2

[2025-06-4 00:10:15] Ejecutando TCP SYN scan en [445, 80]
→ Puertos TCP abiertos: [445]

[2025-06-4 00:10:16] Iniciando UDP scan en [53, 67, 123]
→ Puertos UDP abiertos/filtrados: [53, 67, 123]

[2025-06-4 00:10:22] Informe guardado en
/home/danigonzalezbraza/simulador_ataques/informe_simulador.json

[2025-06-4 00:10:22] Informe guardado en
/home/danigonzalezbraza/simulador_ataques/informe_simulador.html

[2025-06-4 00:10:22] Fin del simulador.
```

2. Objetivo Linux (192.168.100.10)
   Con un comando similar, el script reveló puertos distintos ([] en TCP y [67, 123] en UDP) y generó informes análogos.

Integración con la plataforma web
Tras la ejecución local, el usuario sube el archivo JSON resultante mediante el formulario “Subir resultados de simulación”. La web parsea automáticamente cada sección del JSON y presenta en pantalla tarjetas de alerta estructuradas:

Ilustración 10: Captura del proceso de subida de resultados del Simulador de Ataques.

Este enfoque integral demuestra que el Script 6 no solo ejecuta pruebas técnicas avanzadas, sino que integra sus resultados en la plataforma web, proporcionando recomendaciones claras y ayudando a la PYME a fortalecer su perímetro de red de manera documentada y repetible.

5.8 Plan de mejora de recursos (Script 7)

El Optimizador de Recursos ENS (Script 7) ayuda a las PYMEs a traducir los hallazgos de diagnóstico en un plan de inversión y esfuerzo concreto, adaptado a su tamaño, capacidad interna y presupuesto disponible. Se ejecuta directamente en la interfaz web, donde el usuario completa un sencillo formulario con los siguientes parámetros de entrada:

- Número total de empleados: determina las horas de formación de usuarios.
- Número de técnicos TI dedicados: define el mínimo de formación avanzada y soporte.
- Número de sedes o sucursales: impacta en las horas de auditoría externa anual.
- Nivel de madurez actual: (‘Bajo’, ‘Medio’, ‘Alto’) opcionalmente ajusta prioridades.
- Presupuesto disponible (€): tope económico para el plan de prioridades.
- Estado de controles ENS: un diccionario interno con el resultado booleano de cada control (obtenido de Scripts 1 y 2).

Tras pulsar “Calcular estimación”, el script aplica las siguientes fórmulas y reglas heurísticas:

- **Formación de usuarios**
    1. Horas = nº_empleados × 2 h / empleado
    2. Coste = horas × €100 / h
       Este bloque cubre el tiempo necesario para capacitar a todo el personal en políticas y procedimientos básicos.
- **Formación de técnicos**
    1. Se usa al menos 1 técnico si la PYME no aporta ninguno.
    2. Horas = nº_tecnicos × 8 h / técnico
    3. Coste = horas × €120 / h
       Orientada a personal TI para desplegar y mantener las soluciones.
- **Redacción de políticas**
    1. Se consideran todos los controles de tipo “política” no cumplidos (por ejemplo ORG.2, OP.ACC.1, OP.EXP.3).
    2. Horas = 5 h / control
    3. Coste = horas × €90 / h
       Cada política pendiente genera una partida de redacción.
- **Auditoría externa**
    1. Horas = 20 h / sede
    2. Coste = horas × €150 / h
       Cobertura para la auditoría formal exigida.
- **Herramientas técnicas**
    1. Antivirus: €30 / usuario si no se dispone.
    2. Firewall perimetral: coste fijo €3000 si falta.
    3. SIEM inicial: coste fijo €2000 si no hay plataforma de logs.
- **Oficial de seguridad**
    1. Si no existe perfil designado, se añade €6000 / año (20 % de un salario promedio) como partida anual.

Con estos cálculos, el script obtiene un desglose de partidas (horas y coste) y genera un objeto JSON similar a:

"partidas": {
  "formacion_usuarios": {"horas":60,"coste":6000.0,"texto":"30×2h a €100/h ⇒ €6000"},

"formacion_tecnicos":{"horas":8,"coste":960.0,"texto":"1×8h a €120/h ⇒ €960"},

"politicas":{"horas":50,"coste":4500.0,"texto":"10 controles ×5h a €90/h ⇒ €4500"},

…

}

A continuación, el script calcula los totales:

- Horas totales: suma de todas las partidas con horas.
- Coste total inicial: suma de todos los costes.
- Coste recurrente anual: antivirus + auditoría + oficial de seguridad + mantenimiento SIEM (20 % del coste inicial de SIEM).

Por último, y de manera distintiva, el Optimizador genera un plan de prioridades en función del presupuesto disponible. Ordena todas las partidas con coste superior a cero de menor a mayor, y las acumula hasta alcanzar el tope presupuestario. De esta forma, la interfaz muestra:

1. Antivirus – €900 (acumulado €900).
2. Formación técnicos – €960 (acumulado €1860).
3. SIEM inicial – €2000 (acumulado €3860).
4. Auditoría externa – €3000 (acumulado €6860).
5. Firewall perimetral – €3000 (acumulado €9860).

Si el presupuesto fuese inferior al coste total, las partidas sobrantes (por ejemplo, redacción de políticas o contratación de oficial) quedarían fuera del plan inmediato. Esto permite a la PYME decidir qué acciones priorizar y justificar ante la dirección o inversores.

Resumen de resultados:
Para un caso de prueba con 30 empleados, 1 técnico, 1 sede, madurez baja y presupuesto de €10 000, el script devolvió:

- Horas estimadas: 138
- Coste inicial: €26 360
- Coste anual recurrente: €10 300
- Plan de prioridades: Antivirus, Formación técnicos, SIEM, Auditoría y Firewall (acumulado €9 860).

Desglose de partidas
Formacion usuarios: 30 empleados × 2 h = 60 h; a €100/h ⇒ €6000.00. Horas: 60 h Coste: €6000.00
Formacion tecnicos: 1 técnicos × 8 h = 8 h; a €120/h ⇒ €960.00. Horas: 8 h Coste: €960.00
Politicas: Controles ENS “política” sin cumplir: - ORG.2: 5 h a €90/h ⇒ €450.00 - ORG.3: 5 h a €90/h ⇒ €450.00 - ORG.4: 5 h a €90/h ⇒ €450.00 - OP.PL.3: 5 h a €90/h ⇒ €450.00 - OP.PL.4: 5 h a €90/h ⇒ €450.00 - OP.PL.5: 5 h a €90/h ⇒ €450.00 - OP.ACC.1: 5 h a €90/h ⇒ €450.00 - OP.ACC.4: 5 h a €90/h ⇒ €450.00 - OP.CONT.1: 5 h a €90/h ⇒ €450.00 - OP.EXP.3: 5 h a €90/h ⇒ €450.00 Horas: 50 h Coste: €4500.00
Auditoria externa: 1 sedes × 20 h = 20 h; a €150/h ⇒ €3000.00. Horas: 20 h Coste: €3000.00
Antivirus: 30 usuarios × €30 = €900.00 (antivirus corporativo). Coste: €900.00
Firewall: Instalación de firewall perimetral: €3000.00. Coste: €3000.00
Siem: Instalación/licencia inicial de SIEM o plataforma de logs: €2000.00. Coste: €2000.00
Plan continuidad: El control 'Op.cont.1 – Copias de seguridad / continuidad' ya se incluyó en 'políticas'. Coste: €0.00
Gestion incidentes: El control 'Op.exp.3 – Gestión de Incidentes' ya se incluyó en 'políticas'. Coste: €0.00
Oficial seguridad: Contratación de Oficial de Seguridad (20 % dedicación): €6000.00/año. Coste: €6000.00
Totales
Horas totales estimadas: 138 h Coste total inicial: €26360.00 Coste recurrente anual estimado: €10300.00

Tabla 5: Ejemplo de salida del optimizador de recursos.

Plan de prioridades (presupuesto €10000):

1. **Antivirus** – €900.00 (acumulado €900.00)
   30 usuarios × €30 = €900.00 (antivirus corporativo).
2. **Formacion tecnicos** – €960.00 (acumulado €1860.00)
   1 técnicos × 8 h = 8 h; a €120/h ⇒ €960.00.
3. **Siem** – €2000.00 (acumulado €3860.00)
   Instalación/licencia inicial de SIEM o plataforma de logs: €2000.00.
4. **Auditoria externa** – €3000.00 (acumulado €6860.00)
   1 sedes × 20 h = 20 h; a €150/h ⇒ €3000.00.
5. **Firewall** – €3000.00 (acumulado €9860.00)
   Instalación de firewall perimetral: €3000.00.

Tabla 6: Plan de prioridades según presupuesto.

Este módulo cierra el ciclo automatizado, proporcionando no solo un diagnóstico, sino un plan de acción financiable, alineado con la capacidad real de la PYME.

5.9 Informe final de cumplimiento (Script 8)

El Script 8 cierra el ciclo completo de la plataforma convirtiendo en un único documento PDF todo el conocimiento generado por las fases anteriores. Su propósito no es únicamente agrupar datos, sino ofrecer un informe profesional, estructurado y visualmente atractivo que facilite la presentación de resultados ante la dirección de la PYME o frente a un auditor externo.

Para ello, al pulsar “Generar Informe” la aplicación:

1. Recopila los datos intermedios desde la carpeta static/:
   - Estado de cumplimiento global y por familia (salida de Scripts 1 y 2).
   - Alertas de seguridad en tiempo real (Script 5).
   - Hallazgos de la simulación de ataques (Script 6).
   - Estimación de recursos y costes (Script 7).
2. Agrega y tabula la información:
   - Calcula métricas clave, como el porcentaje medio de controles cumplidos, el número total de alertas por tipo y las horas/coste totales.
3. Genera gráficos con Matplotlib:

- **Barras apiladas**: muestra para cada familia de controles (OP.ACC, MP.SI, OP.MON, OP.CONT, OP.EXP) el porcentaje de “Cumple”, “Parcial” y “No cumple”, ayudando a visualizar de un vistazo las áreas críticas donde la PYME debe concentrar sus esfuerzos.
- **Diagrama de pastel**: resume la distribución de alertas por tipo (fuerza bruta, creación de cuentas, cambios de archivos críticos), ilustrando los vectores de ataque más frecuentes.
- **Barras simples**: compara horas de formación, auditoría y costes de consultoría/herramientas, facilitando la evaluación presupuestaria.

4. Compone el PDF con FPDF:
    - Portada con título, nombre de la PYME y fecha.
    - Índice automático de secciones.
    - Resumen Ejecutivo (texto introductorio).
    - Secciones detalladas, cada una agrupando tabla, gráfico y un párrafo narrativo que explica y contextualiza los resultados.
    - Conclusiones y recomendaciones finales.

El informe resultante, de unas 9 páginas, combina tablas, gráficos e interpretación experta. Simplemente se introduce el nombre de la empresa y segenera el pdf, que sirve como evidencia documentada del cumplimiento del ENS 2022 nivel Alto, lista para presentar en auditorías o reuniones de dirección.

Ejemplos de graficos generados por el informe:

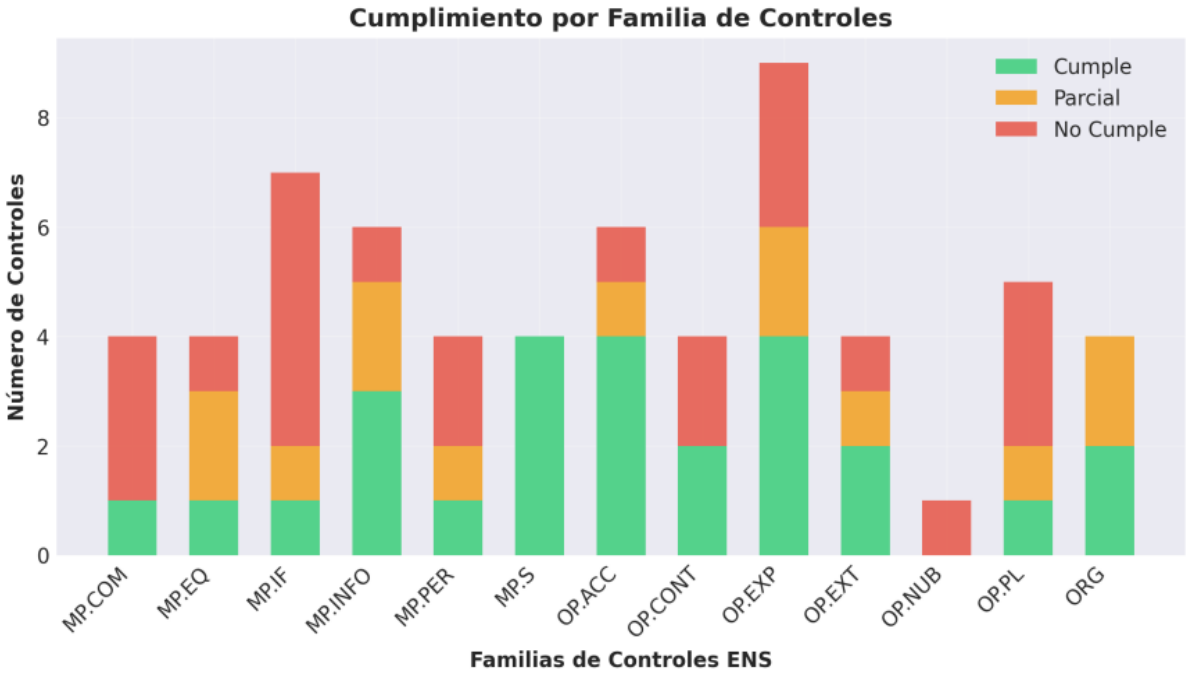

Ilustración 11: Porcentaje de controles ENS nivel Alto cumplidos, parciales y no cumplidos, desglosado por familia de medidas.

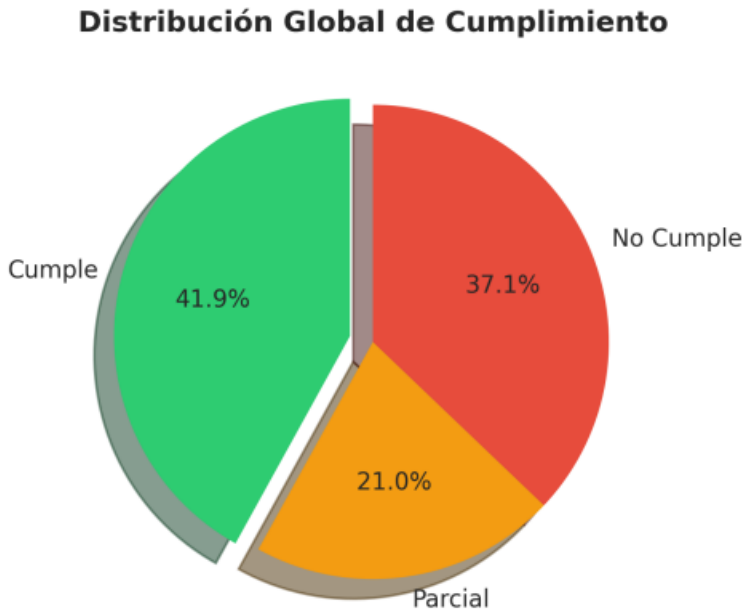

Ilustración 12: Porcentaje de controles ENS nivel Alto cumplidos, parciales y no cumplidos.

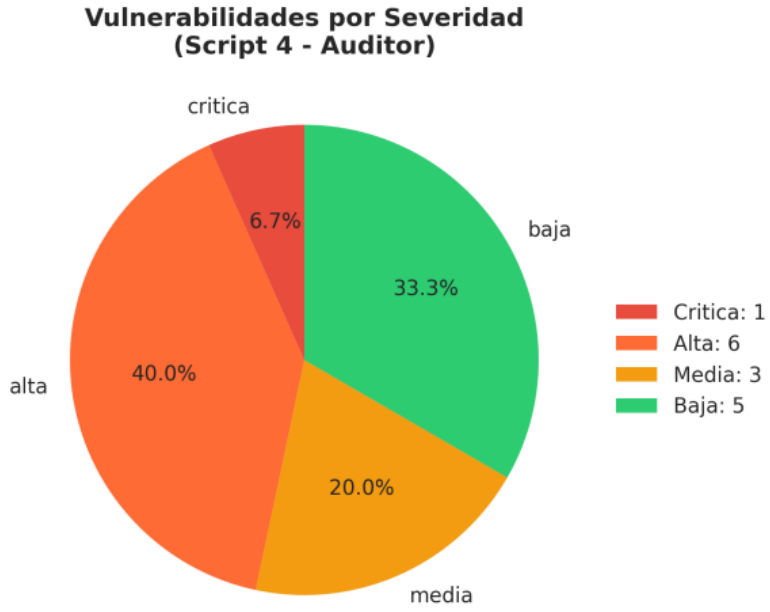

Ilustración 13: Distribución de vulnerabilidades detectadas en la auditoría de permisos, agrupadas por severidad.

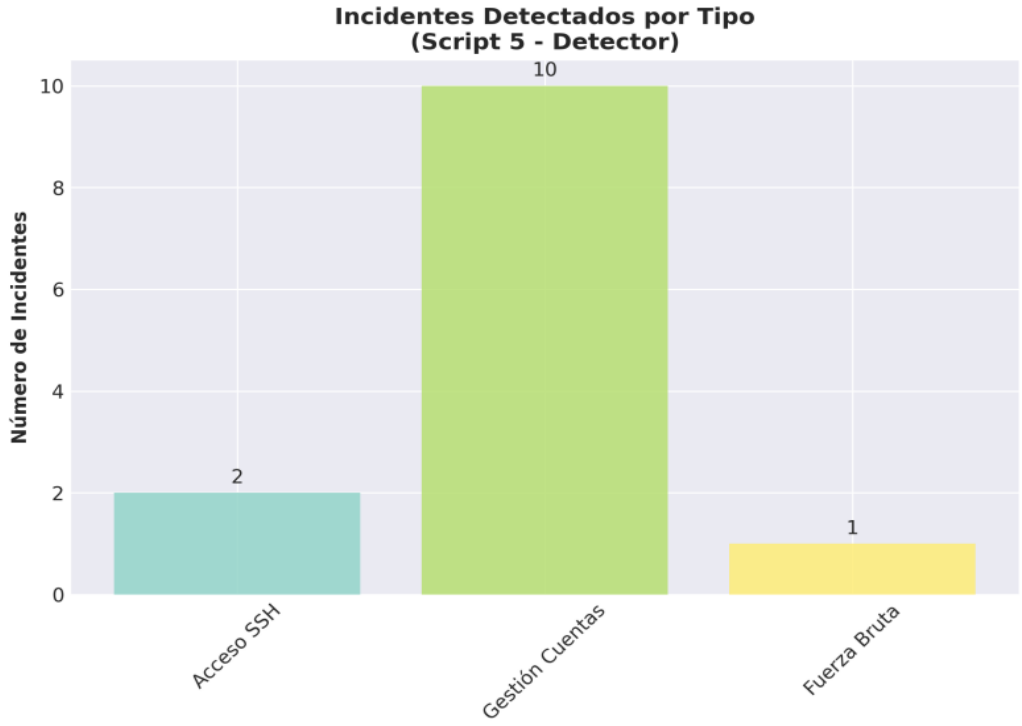

Ilustración 14: Número de alertas de seguridad generadas en tiempo real, clasificadas por tipo de incidente.

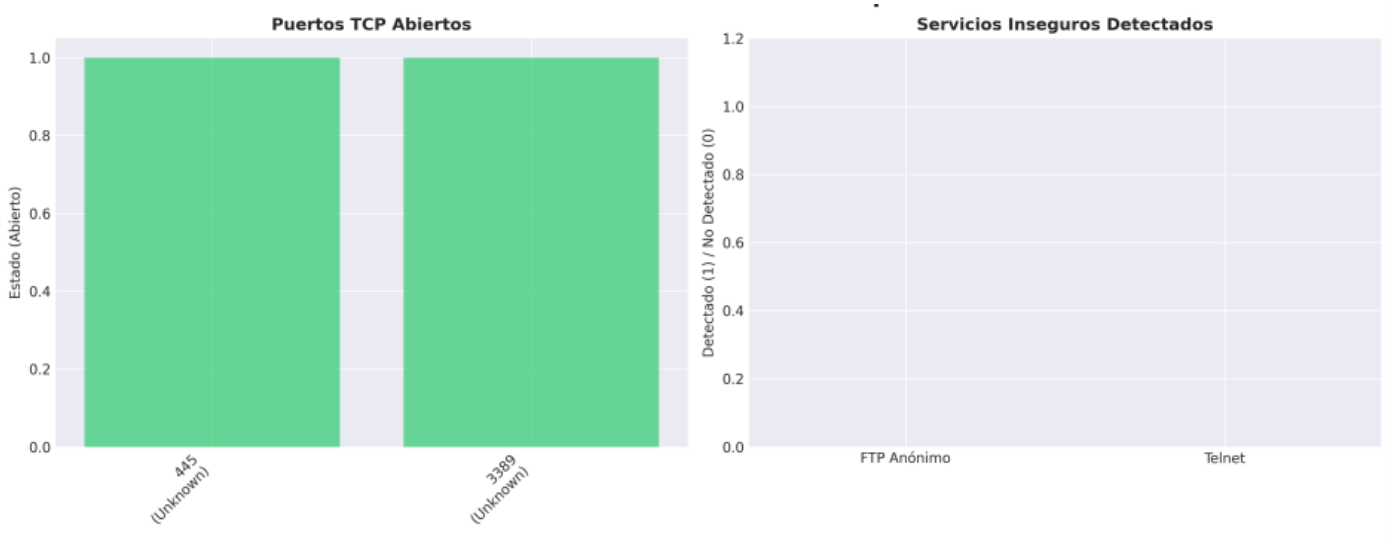

Ilustración 15: Número de puertos TCP y UDP abiertos identificados durante la simulación de ataques

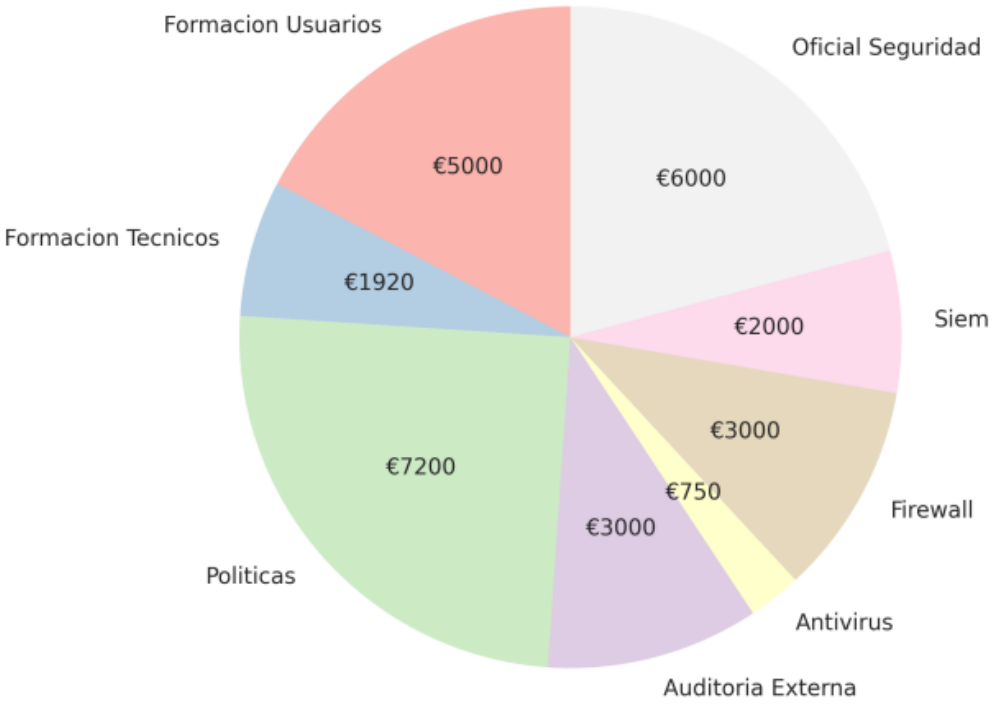

Ilustración 16: Porcentaje del coste de las optimizaciones.

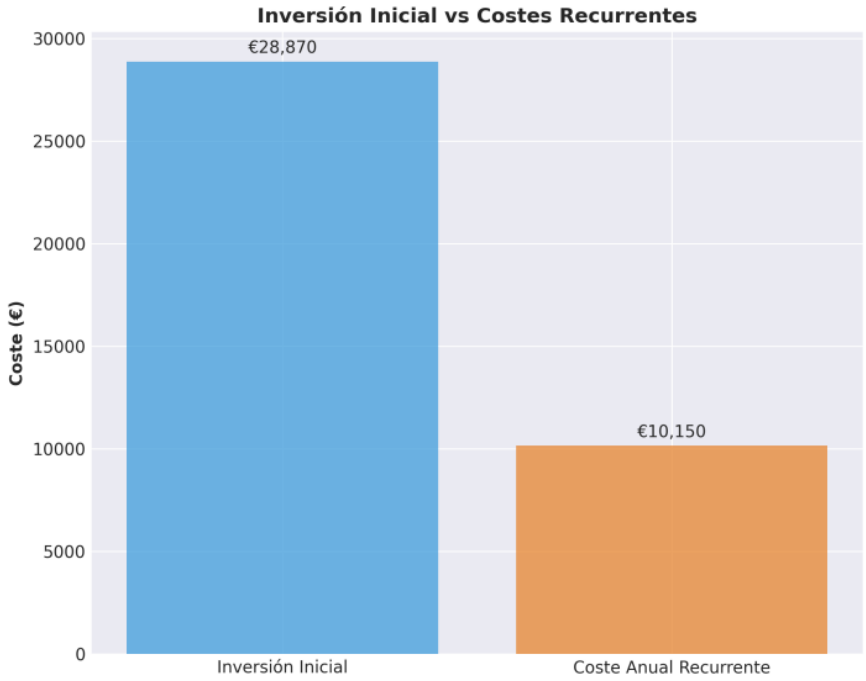

Ilustración 17: Comparativa del coste inicial y recurrente anual estimados para implementar las medidas del ENS.

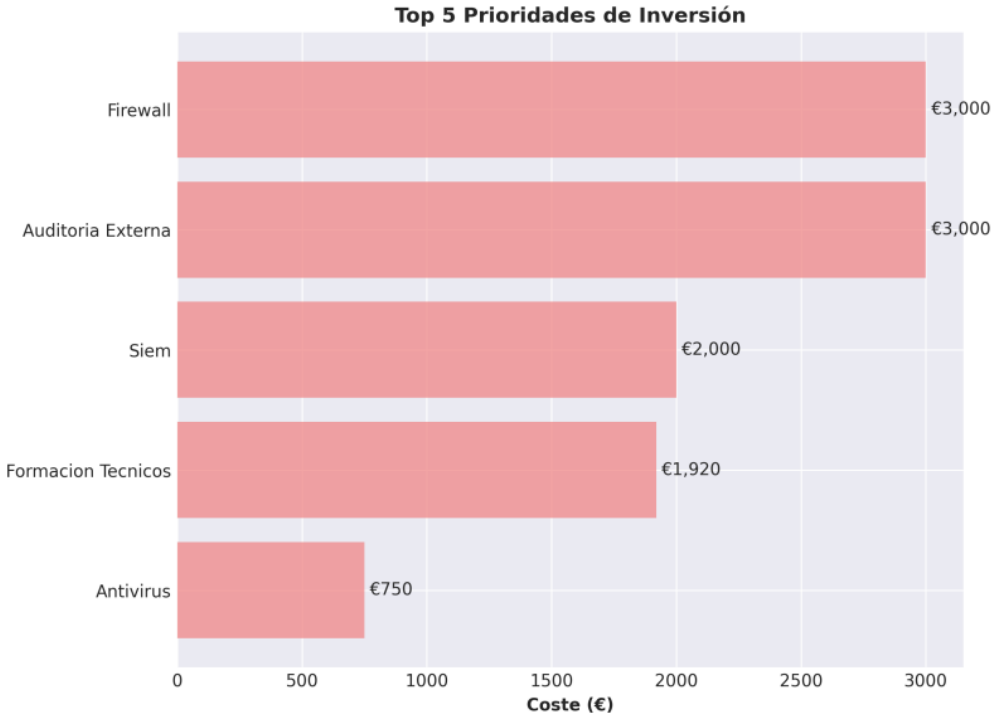

Ilustración 18: Grafico por prioridad de las cinco primeres optimizaciones.

6. Conclusiones y Trabajo Futuro

6.1 Resumen de las aportaciones y grado de cumplimiento ENS

Este TFG ha desarrollado una plataforma contenerizada con ocho scripts que automatizan desde el diagnóstico inicial hasta la generación de evidencias para auditorías ENS 2022 nivel Alto. En entornos Windows/Linux de laboratorio se comprobó que la herramienta cubre la gran mayoría de los controles técnicos, reduce drásticamente el esfuerzo manual (de horas a minutos) y facilita la producción de informes estructurados. Quedan pendientes únicamente controles organizativos y de formación, que se integran como recomendaciones finales.

6.2 Impacto de la automatización en PYMEs

La automatización de procesos clave del ENS a través de la plataforma propuesta supone un cambio de paradigma para las PYMEs, que hasta ahora dependían en gran medida de auditorías manuales, consultorías externas y la experiencia puntual de su personal de TI. Los principales beneficios detectados son:

- **Reducción drástica de esfuerzo y tiempo**: tareas que podían requerir decenas de horas de revisión manual o una semana de trabajo de un consultor externo pasan a completarse en cuestión de minutos, liberando recursos para actividades de valor añadido.
- **Consistencia y fiabilidad**: al basarse en scripts estandarizados, las comprobaciones se realizan siempre bajo los mismos criterios, evitando omisiones o errores humanos derivados de la fatiga o la variabilidad en los equipos.
- **Accesibilidad del cumplimiento ENS**: la herramienta convierte un proceso tradicionalmente complejo y costoso en un servicio web accesible para cualquier PYME con conocimiento básico de informática, democratizando el acceso a estándares de seguridad avanzados.
- **Visión continua y mejora iterativa**: la plataforma no se limita a una auditoría puntual, sino que puede reutilizarse periódicamente para monitorizar evoluciones en la infraestructura y detectar desviaciones en tiempo real, facilitando un enfoque de mejora continua.
- **Generación de evidencias estructuradas**: la consolidación automática de informes, gráficas y recomendaciones ofrece a las PYMEs documentación lista para presentar en procesos de homologación o licitación pública, reduciendo barreras administrativas.

En conjunto, esta automatización contribuye a elevar el nivel de madurez en ciberseguridad de las PYMEs, mejorando su competitividad y capacidad para acceder a contratos con el sector público, al tiempo que fortalece su resiliencia frente a amenazas digitales.

6.3 Limitaciones del prototipo y aspectos a mejorar

A pesar de los resultados prometedores, la versión actual del prototipo presenta varias limitaciones que conviene considerar:

- **Cobertura organizativa incompleta**
  El foco en automatización técnica deja fuera controles puramente organizativos o de formación (por ejemplo, constitución de comités de seguridad, evaluaciones de riesgo por personal externo y formación masiva de usuarios). Estas actividades requieren intervención humana y deberían complementarse con plantillas de procesos o módulos de gestión documental más avanzados.

- **Dependencia de entornos homogéneos**
  Los scripts suponen configuraciones estándares (Windows 11, Ubuntu 22.04, rutas fijas, formatos de log). Entornos heterogéneos o distribuciones distintas pueden requerir adaptaciones de rutas, nombres de servicios o estructuras de Syslog, lo que limita la portabilidad sin ajustes manuales.

- **Interacción semi-manual para scripts locales**
  Los módulos 4, 5 y 6 requieren descarga y ejecución local, así como subida manual de resultados. Esto introduce fricción y posibilidad de error (ficheros olvidados, cargas incorrectas). Una versión futura podría integrar agentes ligeros que se comuniquen automáticamente con la plataforma central.

- **Rudimentaria interfaz de usuario**
  Aunque funcional, la UI actual carece de un estilo homogéneo, carece de indicadores de progreso detallados y no agrupa resultados históricos para facilitar el seguimiento longitudinal. Un rediseño con un framework de componentes o template base mejoraría la experiencia y la claridad.

- **Generación de recomendaciones genéricas**
  Las recomendaciones se basan en un diccionario interno de texto estático. Para aumentar su relevancia, se podrían enriquecer con enlaces a guías CCN-CERT, FAQs o ejemplos de configuración concretos, así como ofrecer rutas de aprendizaje o integraciones con repositorios de scripts.

Atender estas limitaciones permitirá evolucionar el prototipo hacia una solución más robusta, escalable y adaptable a la diversidad real de infraestructuras y necesidades de las PYMEs.

6.4 Líneas de trabajo futuro.

Para convertir este prototipo en una solución madura y ampliamente aplicable a las necesidades de las PYMEs, se identifican las siguientes líneas de mejora y ampliación:

- **Cobertura de controles organizativos**
  Ampliar el Generador de Políticas (Script 3) con plantillas para comités de seguridad, gestión de riesgos formales y formación de usuarios, y añadir un módulo que gestione flujos de aprobación y firmas electrónicas de directivos.

- **UI/UX y dashboard unificado**
  Rediseñar la interfaz utilizando un framework de componentes (por ejemplo, Bootstrap o Tailwind) y añadir un dashboard central con métricas históricas, líneas de tendencia de cumplimiento y alertas activas.

- **Enriquecimiento de recomendaciones**
  Integrar enlaces a guías CCN-CERT, scripts de ejemplo en GitHub o fragmentos de

configuración automáticos que permitan aplicar las recomendaciones de forma guiada.

Estas líneas de trabajo futuro enriquecerán la plataforma, pasando de un prototipo académico a una herramienta empresarial capaz de adaptarse a diversos escenarios y de acompañar a las PYMEs en su mejora continua de ciberseguridad.

7. Referencias

AMETIC. (2022). *Guía de implantación del ENS 2022 para PYMEs*. AMETIC.

CCN-CERT. (s. f.-a). *Guía CCN-STIC 809: Adecuación al Esquema Nacional de Seguridad*. Centro Criptológico Nacional. Recuperado el 12 de junio de 2025, de https://www.ccn-cert.cni.es/es/guias.html

CCN-CERT. (s. f.-b). *Guía CCN-STIC 817: Medidas de seguridad de nivel alto*. Centro Criptológico Nacional. Recuperado el 12 de junio de 2025, de https://www.ccn-cert.cni.es/es/guias.html

CCN-CERT. (s. f.-c). *Guía CCN-STIC 818: Medidas de seguridad de nivel medio*. Centro Criptológico Nacional. Recuperado el 12 de junio de 2025, de https://www.ccn-cert.cni.es/es/guias.html

CCN-CERT. (s. f.-d). *Guía CCN-STIC 820: Gestión de incidentes de seguridad*. Centro Criptológico Nacional. Recuperado el 12 de junio de 2025, de https://www.ccn-cert.cni.es/es/guias.html

CCN-CERT. (s. f.-e). *Guía CCN-STIC 824: Análisis y gestión de riesgos de los sistemas de información*. Centro Criptológico Nacional. Recuperado el 12 de junio de 2025, de https://www.ccn-cert.cni.es/es/guias.html

CIS. (2021). *CIS Controls v8*. Center for Internet Security.

DocuSign. (s. f.). *Ciberseguridad en las pymes* [Blog]. DocuSign. Recuperado el 12 de junio de 2025, de https://www.docusign.com/es-mx/blog/desarrolladores/ciberseguridad-en-las-pymes

Elastic NV. (2025). *Logstash Reference Guide*. Elastic.

Gobierno de España. (2022, 3 de mayo). *Real Decreto 311/2022, por el que se regula el Esquema Nacional de Seguridad (ENS)*. *Boletín Oficial del Estado*. Recuperado el 12 de junio de 2025, de https://www.boe.es/buscar/doc.php?id=BOE-A-2022-7191

INCIBE. (s. f.). *Políticas de seguridad para pymes*. Instituto Nacional de Ciberseguridad. Recuperado el 12 de junio de 2025, de https://www.incibe.es

ISO. (2013). *ISO/IEC 27001:2013 Information technology – Security techniques – Information security management systems – Requirements*. International Organization for Standardization.

MITRE. (s. f.). *ATT&CK™: Adversarial Tactics, Techniques, and Common Knowledge*. The MITRE Corporation. Recuperado el 12 de junio de 2025, de https://attack.mitre.org/

NIST. (2018). *Framework for Improving Critical Infrastructure Cybersecurity* (versión 1.1). National Institute of Standards and Technology.

Oliver Wyman. (2023, abril). *Digitalización de pymes en España* [Oliver Wyman Insights]. Recuperado el 12 de junio de 2025, de https://www.oliverwyman.es/es/nuestra-experiencia/insights/2023/apr/digitalizacion-de-pymes-en-espana.html

Pandas Development Team. (2023). *pandas: Powerful Python data analysis toolkit* (v 1.5.1). Recuperado de https://pandas.pydata.org/

Scapy Team. (2023). *Scapy: Packet manipulation program* (v 2.4.5). Recuperado de https://scapy.net/

VirtualBox. (2023). *Oracle VM VirtualBox User Manual* (v 7.0). Recuperado de https://www.virtualbox.org/manual/UserManual.html

8. Anexo

Script 4: Auditor de Permisos y Accesos

Objetivo:
Recorrer los registros de usuarios y configuraciones de acceso para detectar incumplimientos de controles del ENS y generar un informe de incidencias.

```
// Inicio del script
Inicializar LOGFILE
Redirigir salida estándar y errores a LOGFILE

// 1. Comprobar privilegios de ejecución
Si no es administrador/root Entonces
  Mostrar mensaje de “ejecutar como administrador”
  Terminar ejecución
FinSi

// 2. Identificación de usuarios duplicados
Obtener lista de UIDs duplicados
Para cada UID duplicado:
  Obtener nombres de usuario asociados
  Registrar “UID duplicado: <UID> → <usuarios>”
FinPara
Si no hubo duplicados Entonces
  Registrar “No se han encontrado UIDs duplicados”
FinSi

// 3. Usuarios con shell o login inválido
Para cada usuario con UID >= 1000 y shell en {false, nologin}:
  Registrar “Usuario con shell inválido: <usuario> → <shell>”
FinPara
Si no hubo usuarios inválidos Entonces
  Registrar “Todos los usuarios tienen shells válidas”
FinSi

// 4. Usuarios en grupos privilegiados
```

```
Definir grupos críticos = {sudo, adm, wheel, shadow}
Para cada grupo en grupos críticos:
  Si el grupo existe Entonces
    Obtener usuarios del grupo
    Si la lista no está vacía Entonces
      Para cada usuario en la lista:
        Registrar “Usuario con privilegios en <grupo>: <usuario>”
      FinPara
    Else
      Registrar “No hay usuarios en <grupo>”
    FinSi
  FinSi
FinPara

// 5. Cuentas bloqueadas o expiradas
Para cada entrada en /etc/shadow (o equivalente):
  Si la cuenta está bloqueada Entonces
    Registrar “Cuenta bloqueada: <usuario>”
  Si la cuenta está expirada y fecha < hoy Entonces
    Registrar “Usuario expirado: <usuario> (expiró <fecha>)”
  FinSi
FinPara

// 6. Comprobaciones de configuración SSH
Si existe archivo de configuración SSH Entonces
  Verificar PermitRootLogin
  Verificar PasswordAuthentication
  Verificar PubkeyAuthentication
  Registrar cada hallazgo
Else
  Registrar “No se ha encontrado el archivo de configuración SSH”
FinSi

// 7. Políticas de contraseñas locales
Leer /etc/login.defs y /etc/pam.d/common-password
Para cada directiva (PASS_MAX_DAYS, PASS_MIN_LEN, etc.):
```

```
  Comparar valor actual vs. esperado
  Registrar “<clave> = <valor> (correcto/revisar)”
FinPara

// 8. Mensaje final
Si se detectaron vulnerabilidades Entonces
  Registrar “Se detectaron vulnerabilidades. Informe en LOGFILE”
Else
  Registrar “No se detectaron vulnerabilidades. Informe en LOGFILE”
FinSi

// Fin del script
```

Script 5: Detector de Incidentes

Objetivo:
Supervisar en tiempo real el fichero JSON de incidentes (incidentes.json) para Windows y Linux, detectar patrones de ataque (fuerza bruta, creación de cuentas, cambios en archivos críticos) y generar alertas tanto en pantalla como en un log dedicado.

```
// Inicio del script
Inicializar hashes de integridad de archivos
Abrir INCIDENTS_JSON en modo lectura y situar puntero al final

Mientras True:
  // 1. Comprobación periódica de integridad (Linux)
  Si han pasado ≥ FILE_CHECK_INTERVAL segundos desde último chequeo:
    Para cada ruta en FILES_TO_MONITOR:
      Calcular SHA256(ruta)
      Si hash_nuevo ≠ hash_anterior:
        log_alert("Archivo crítico modificado: " + ruta)
        print_json_with_timestamp({ tipo: "Linux_FileModificado", ruta, hash_anterior, hash_nuevo })
        Actualizar hash_anterior

  // 2. Lectura de línea nueva
  Leer linea de INCIDENTS_JSON
  Si linea vacía:
    Esperar 0.2s y continuar
  Intentar parsear linea como JSON → parsed
  Si falla parseo:
    Mostrar aviso y continuar

  // 3. Procesar evento Windows
  Si parsed contiene clave "winlog":
    event_id ← parsed["winlog"]["event_id"]
    Según event_id:
      4625: // fallo de logon → posible fuerza bruta
        Actualizar contador y deque de failed_logins[(usuario, ip)]
        Si deque alcanza FAIL_THRESHOLD en WINDOW_MINUTES:
```

```
        log_alert("…POSIBLE FUERZA BRUTA (4625)")
        print_json_with_timestamp({ tipo: "Windows_FuerzaBruta", event_id, usuario, ip,
num_fallos })
        Resetear contador/deque
      4624: // logon exitoso tras fallos
       Si failed_count_win[(usuario, ip)] ≥ umbral:
        log_alert("…POSIBLE FUERZA BRUTA CON ÉXITO (4624)")
        print_json_with_timestamp({ tipo: "Windows_FuerzaBrutaConExito", ... })
       Resetear contador/deque
      4720, 4722, 4728, 4732: // creación/habilitación/elevación
       log_alert("…INCIDENTE (" + event_id + ")")
       print_json_with_timestamp({ tipo: "Windows_…", event_id, usuario_o_grupo })
      5152, 5156: // Firewall DROP/ALLOW
       log_alert("…Firewall " + (DROP/ALLOW) + " (" + event_id + ")")
       print_json_with_timestamp({ tipo: "Windows_Firewall_" + (DROP/ALLOW), event_id,
ip, puerto })
    Continuar al siguiente bucle


  // 4. Procesar evento Linux
  msg ← lowercase(parsed.get("message",""))
  Si "invalid user" en msg:
    Extraer usuario e ip
    log_alert("Intento SSH con usuario inválido")
    print_json_with_timestamp({ tipo: "Linux_InvalidUser", usuario, origen })
    Continuar
  Si "authentication failure" en msg:
    Actualizar deque y contador ssh_failed[(usuario, ip)]
    Si deque alcanza SSH_THRESHOLD en SSH_WINDOW_MINUTES:
      log_alert("…POSIBLE FUERZA BRUTA SSH")
      print_json_with_timestamp({ tipo: "Linux_FuerzaBrutaSSH", usuario, origen,
num_fallos })
      Resetear contador/deque
    Continuar
  Si "accepted password" en msg:
    Si ssh_failed_count[(usuario, ip)] ≥ umbral:
      log_alert("…POSIBLE FUERZA BRUTA CON ÉXITO SSH")
```

print_json_with_timestamp({ tipo: "Linux_FuerzaBrutaConExitoSSH", usuario, origen, num_fallos_previos })

Resetear contador/deque

FinWhile

// Fin del script

Script 6: Simulador de Ataques

Objetivo:
Simular distintos tipos de ataques (TCP SYN, FTP anónimo, Telnet, HTTP, ICMP flood, UDP y SNMP) contra un objetivo, recopilar los resultados y generar un informe final en formato JSON y HTML.

// Inicio del script

// 1. Parsear argumentos de entrada
Leer argumento “target” (IP/hostname)
Leer “--tcp-ports” o bandera “--full-tcp”
Leer “--udp-ports”
Leer “--icmp-flood” (número de paquetes)

// 2. Determinar listas de puertos y conteo ICMP
Si “full-tcp” está activo:
  tcp_ports ← [1 … 1024]
Sino si “tcp-ports” especificados:
  tcp_ports ← lista de args.tcp_ports
Sino:
  tcp_ports ← [21,22,23,80,139,445,3389]
udp_ports ← args.udp_ports (por defecto [53,67,123])
icmp_count ← args.icmp_flood

// 3. Inicializar estructura de resultados
resultados ← {
  target,
  timestamp_actual,

```
  tcp_abiertos: [],
  ftp_anonimo: False,
  telnet_banner: None,
  http_banner: None,
  icmp_enviados: icmp_count,
  udp_abiertos: [],
  snmp_info: None
}

// 4. Mostrar mensaje de inicio

// 5. Escaneo TCP SYN
resultados.tcp_abiertos ← tcp_syn_scan(target, tcp_ports)
Mostrar “Puertos TCP abiertos: ” + resultados.tcp_abiertos

// 6. Prueba de FTP anónimo (puerto 21)
Si 21 ∈ resultados.tcp_abiertos:
  resultados.ftp_anonimo ← prueba_ftp_anonimo(target, 21)
  Mostrar “FTP anónimo: ” + (Sí/No)

// 7. Obtención de banner Telnet (puerto 23)
Si 23 ∈ resultados.tcp_abiertos:
  resultados.telnet_banner ← telnet_banner(target, 23)
  Mostrar “Banner Telnet: ” + (banner o “N/A”)

// 8. Obtención de banner HTTP (puerto 80)
Si 80 ∈ resultados.tcp_abiertos:
  resultados.http_banner ← http_banner(target, 80)
  Mostrar “Banner HTTP: ” + (banner o “N/A”)

// 9. ICMP flood
Si icmp_count > 0:
  icmp_flood(target, icmp_count)
  Mostrar “ICMP flood de ” + icmp_count + “ paquetes finalizado.”

// 10. Escaneo UDP
```

```
Si udp_ports no está vacío:
  resultados.udp_abiertos ← udp_scan(target, udp_ports)
  Mostrar “Puertos UDP abiertos/filtrados: ” + resultados.udp_abiertos

// 11. Verificación SNMP (puerto 161)
Si 161 ∈ udp_ports:
  resultados.snmp_info ← snmp_check(target, 161, "public")
  Mostrar “SNMP sysDescr: ” + (info o “N/A”)

// 12. Generar informe JSON
Escribir `resultados` en “informe_simulador.json”
Mostrar ruta o error

// 13. Generar informe HTML
Construir HTML básico con:
  - Target y timestamp
  - Listado de hallazgos para cada prueba
Escribir en “informe_simulador.html”
Mostrar ruta o error

// 14. Mostrar mensaje de fin de simulación

// FIN DEL SCRIPT

// — Funciones auxiliares empleadas —
// scan_tcp_port, tcp_syn_scan, full_tcp_scan
// prueba_ftp_anonimo, telnet_banner, http_banner
// icmp_flood, udp_scan, snmp_check
```