**Informe de Viabilidad y Plan de Ejecución para SentinelCore**

**📌 Análisis de Viabilidad**

**1. Fortalezas del Proyecto**

**✅ Innovación técnica:**

* **El "toquen de antesala" es un concepto único que diferencia el proyecto de soluciones existentes (Nessus, OpenVAS).**
* **Arquitectura modular basada en microservicios cooperativos ("toquens").**

**✅ Alcance realista para un TFG:**

* **Combina tecnologías maduras (OWASP ZAP, APIs cloud) con desarrollo propio (toquens).**
* **No requiere infraestructura compleja (puede desplegarse en Docker/Local).**

**✅ Valor educativo:**

* **Cubre análisis de vulnerabilidades, hardening, mitigación y seguridad proactiva.**

**2. Riesgos y Mitigación**

| **Riesgo** | **Mitigación** |
| --- | --- |
| **Complejidad del "toquen de antesala"** | **Priorizar solo PDFs y Office en MVP** |
| **Integración con múltiples APIs (AWS, ZAP, etc.)** | **Usar SDKs oficiales (boto3, python-owasp-zap)** |
| **Tiempo limitado del equipo** | **Enfoque en 3 módulos clave: Toquen, Escáner Web, Protección Subdominios** |

**📅 Plan de Ejecución (Equipo de 4 Personas)**

**Fase 1: Diseño y Setup (2 semanas)**

| **Tarea** | **Responsable** | **Entregable** |
| --- | --- | --- |
| **Definir arquitectura técnica detallada** | **Todos** | **Diagrama UML/Flujo de Toquens** |
| **Configurar entorno dev (Python+Docker)** | **Dev 1** | **Repo GitHub con Dockerfile** |
| **Investigar APIs (OWASP ZAP, AWS Security Hub)** | **Dev 2** | **Documentación de integración** |
| **Diseñar interfaz mínima (CLI o Web)** | **Dev 3** | **Mockups/Wireframes** |

**Fase 2: Desarrollo Núcleo (4 semanas)**

| **Módulo** | **Tareas Clave** | **Tecnologías** |
| --- | --- | --- |
| **Toquen de Antesala** | **- Análisis de magic numbers - Sanitización básica de PDF/Office** | **Python + python-magic + PyMuPDF** |
| **Escáner Web** | **- Integración OWASP ZAP API - Parser de resultados** | **FastAPI + OWASP ZAP Docker** |
| **Protección Subdominios** | **- Config Nginx/Apache (noindex) - Auth básica** | **Nginx + .htpasswd** |

**Fase 3: Integración y Pruebas (2 semanas)**

| **Tarea** | **Criterio de Éxito** |
| --- | --- |
| **Toquen detecta y bloquea PDF con JS embebido** | **100% de bloqueo en pruebas** |
| **Escáner reporta vulnerabilidades OWASP Top 10** | **SQLi/XSS detectados** |
| **Subdominio no aparece en Google** | **Test con Google Search Console** |

**Fase 4: Documentación y Presentación (1 semana)**

* **Video demostrativo (3 min) mostrando:**
  1. **Flujo del toquen con un PDF malicioso.**
  2. **Escaneo de una web vulnerable.**
  3. **Subdominio protegido (auth + noindex).**
* **Memoria técnica (focus en innovación del "toquen").**

**✔ Checklist de Trabajos**

**Módulo Toquen de Antesala**

* **Clase MasterToken (identificación tipo archivo).**
* **PDFSanitizer (elimina JS/embeds con pdf-redact-tools).**
* **OfficeAnalyzer (detecta macros con oletools).**
* **Pruebas con archivos maliciosos de muestra.**

**Módulo Escáner Web**

* **Dockerizar OWASP ZAP.**
* **API REST (FastAPI) para lanzar escaneos.**
* **Parser de resultados (JSON → informe simple).**

**Protección de Subdominios**

* **Configurar Nginx/Apache con:**
  + **X-Robots-Tag: noindex.**
  + **Autenticación básica.**
* **Verificar con curl y Google Search Console.**

**Extra (si hay tiempo)**

* **Dashboard mínimo con Vue.js.**
* **Integración AWS Security Hub (solo lectura).**

**🔧 Mejoras Propuestas**

1. **Simplificar Toquens:**
   * **Empezar solo con PDF y Office (los más críticos).**
   * **Usar librerías existentes (oletools, pdfid) en lugar de desarrollo desde cero.**
2. **Automatizar Protección Subdominios:**
   * **Script Python que genere configs Nginx automáticamente.**
   * **Ejemplo:**

**python**

**def generate\_nginx\_config(subdomain):**

**config = f"""**

**server {{**

**listen 80;**

**server\_name {subdomain};**

**add\_header X-Robots-Tag "noindex";**

**auth\_basic "Restricted";**

**auth\_basic\_user\_file /etc/nginx/.htpasswd;**

**}}"""**

**with open(f"/etc/nginx/sites-available/{subdomain}", "w") as f:**

**f.write(config)**

1. **Mitigación Guiada:**
   * **Base de datos SQLite con vulnerabilidades y soluciones (ejemplo):**

**sql**

**CREATE TABLE mitigations (**

**vuln\_id TEXT PRIMARY KEY, -- Ej: "CVE-2021-1234"**

**description TEXT, -- "SQL Injection en formulario X"**

**solution TEXT -- "Usar parámetros preparados"**

**);**

**🚀 Conclusión**

**Viabilidad Alta: El proyecto es realizable en 8-10 semanas con foco en:**

1. **Toquen de antesala (PDF/Office).**
2. **Escáner web con OWASP ZAP.**
3. **Protección de subdominios (noindex + auth).**

**Recomendación final:**

* **Priorizar funcionalidades "diferenciales" (toquen) sobre integraciones complejas.**
* **Usar herramientas ya existentes (ZAP, oletools) para ahorrar tiempo.**