

**UNIVERSIDAD AUTÓNOMA DE OCCIDENTE**

**PROGRAMA ESPECIALIZACIÓN EN SEGURIDAD INFORMATICA**



**SEGURIDAD EN REDES**

**Aplicación para detectar servidores con el servicio DNS expuesto con la ayuda de IA  
(ChatGPT ) y Shodan.**

**Presentado por:**

**Roni El Barkachi**

**Código:**

**22502223**

**Docente:**

**Andres Felipe Gonzalez Coronado**

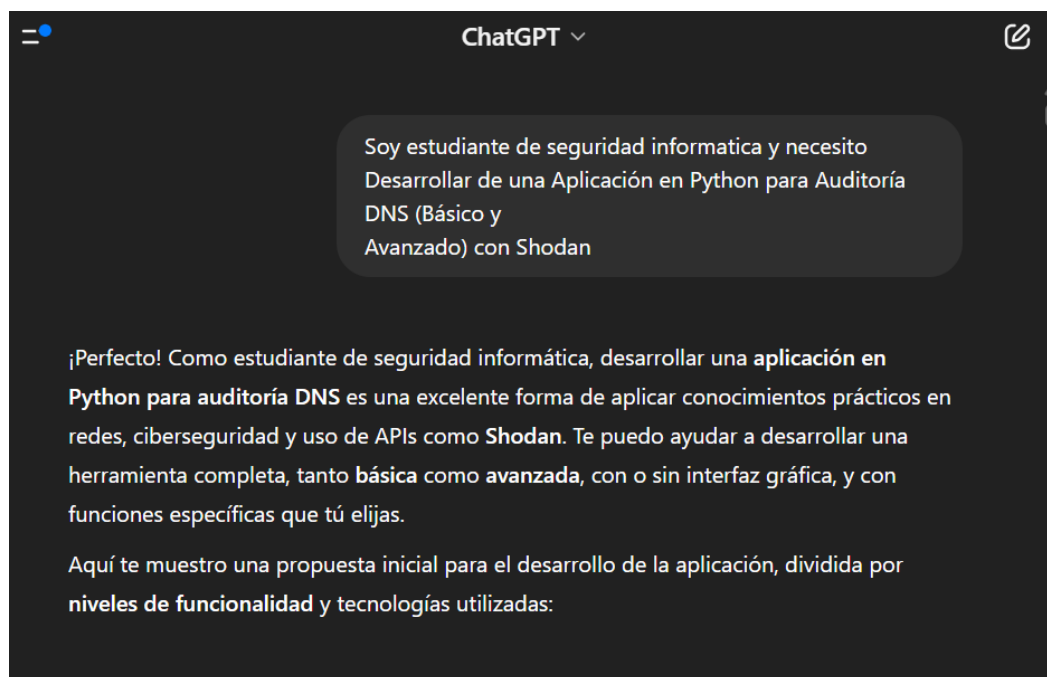
**CALI**

**ABRIL 2025**

## Fabricación de la aplicación

Para comenzar con el desarrollo del proyecto, utilizaremos una herramienta de inteligencia artificial que nos ayude a generar los scripts en el lenguaje de programación Python. En este caso, emplearemos ChatGPT. Todo el proyecto será implementado dentro de una máquina virtual que ejecuta Kali Linux como sistema operativo. También es fundamental contar con una cuenta en SHODAN, ya que esta plataforma será la que usaremos para localizar direcciones IP públicas que tengan el puerto 53 expuesto y potencialmente vulnerable.

El primer prompt que se utilizó para comenzar la creación de la aplicación fue el siguiente:



*Imagen #1. Prompt inicial.*

En este prompt le he especificado que la aplicación se hará con fines educativos al decirle que soy estudiante de seguridad informática con la intención de crear una aplicación en Python para auditoría DNS con requerimientos básicos y avanzados utilizando shodan que nos ayudará a buscar los puertos expuestos "53"

## Aplicación para Auditoría DNS con Python y Shodan

### Tecnologías:

- Python 3
- Shodan API
- Tkinter (para interfaz gráfica, si se desea)
- socket y dnspython (para pruebas DNS locales)
- CSV / logging (para reportes)

*Imagen #2. Respuesta al prompt 1*

## Modo Básico

**Objetivo:** Detectar si una IP tiene el puerto 53 (TCP o UDP) abierto, sin resolver dominios.

### Funcionalidades:

- Ingreso de palabra clave o filtro de búsqueda para Shodan.
- Búsqueda de IPs con el puerto 53 abierto (usando Shodan).
- Identificación si es TCP o UDP.
- Mostrar resultados en consola o GUI.
- Exportar a CSV/log.

*Imagen #3. Sugerencia de Modo Basico*

## Modo Avanzado

**Objetivo:** Analizar comportamiento DNS de servidores detectados.

### Funcionalidades extra:

- Comprobación de resolución recursiva.
- Envío de solicitudes DNS a dominios seleccionados.
- Verificación de listas negras (desde archivo `.txt`).
- Medición de tiempo de respuesta DNS.
- Manejo de múltiples páginas de resultados en Shodan.
- Reporte detallado de cada IP (puerto, recursividad, tiempo, blacklist).
- Exportación organizada de resultados.

*Imagen #4. Sugerencia de Modo Avanzado*

### ⚠ Advertencia Ética

Antes de ejecutar, la herramienta muestra una advertencia sobre el uso ético y educativo de las pruebas realizadas.

### 🖥 GUI (Opcional)

Una interfaz gráfica sencilla usando Tkinter con:

- Campos para API Key, filtros, dominios de prueba.
- Selección entre modo básico o avanzado.
- Botón para iniciar auditoría.
- Ventana de resultados y exportación.

Imagen #5. Advertencia Ética y GUI Aplicacion

### 📁 Estructura del Proyecto

```
bash Copiar Editar

auditor_dns/
|
├─ main.py           # Punto de inicio
├─ shodan_scanner.py # Funciones con Shodan
├─ dns_checker.py    # Comprobaciones DNS
├─ gui.py            # Interfaz gráfica (opcional)
├─ blacklist.txt     # Dominios/IPS a bloquear
├─ report.csv        # Resultados exportados
└─ utils.py          # Funciones auxiliares
```

Imagen #6. Sugerencia de Estructura del proyecto

En este aplicacion necesito cumplir con las siguientes funciones basicas y avanzads

- Funciones Básicas:
  - o Búsqueda de IPs con DNS expuesto (básico).
  - o Verificación de resolución DNS (por ejemplo, a un dominio específico).
- Funciones Avanzadas (al menos dos de las siguientes):
  - o Verificación de recursividad.
  - o Detección de amplificación DNS.
  - o Paginación en Shodan (o herramienta alternativa).
  - o Múltiples dominios de prueba.
  - o Integración de lista negra (blacklist).

Perfecto, según lo que me comentas y lo que ya venimos trabajando con tu herramienta de auditoría DNS en Python con GUI (Tkinter), te detallo cómo podrías estructurar y asegurar que cumples tanto con las funciones básicas como avanzadas que necesitas implementar:

Imagen #7. Prompt 2

En este prompt yo le pido a Chatgpt que tenga en cuenta las funciones avanzadas y básicas de la aplicación la cual nos pide en la practica

### Estructura de Archivos Recomendados

```
bash
auditor_dns/
├── main.py           # Lógica principal y flujo
├── shodan_module.py  # Conexión a Shodan y paginación
├── dns_module.py     # Funciones de resolución, recursividad, amplificación
├── blacklist.txt     # Lista de dominios/IPs a evitar
├── dominios.txt      # Lista de dominios de prueba
├── results.csv       # Resultados exportados
└── config.py         # API keys y configuraciones
```

Imagen #8. Estructura de archivos recomendados del prompt 2

### shodan\_module.py (búsqueda básica y paginación)

```
python
import shodan

def buscar_dns_expuestos(api_key, query="port:53", max_results=100):
    api = shodan.Shodan(api_key)
    resultados = []

    try:
        for resultado in api.search_cursor(query):
            ip = resultado.get('ip_str')
            port = resultado.get('port')
            if ip:
                resultados.append((ip, port))
                if len(resultados) >= max_results:
                    break
    except shodan.APIError as e:
        print(f"[!] Error de Shodan: {e}")
    return resultados
```

### dns\_module.py (resolución DNS básica)

```
python
import dns.resolver

def probar_resolucion(ip, dominio='google.com'):
    resolver = dns.resolver.Resolver()
    resolver.nameservers = [ip]

    try:
        respuesta = resolver.resolve(dominio, lifetime=3)
        return True, [r.to_text() for r in respuesta]
    except Exception as e:
        return False, str(e)
```

### main.py (ejecución básica)

```
python
from shodan_module import buscar_dns_expuestos
from dns_module import probar_resolucion

SHODAN_API_KEY = "TU_API_KEY_AQUI"

def main():
    print("[*] Buscando IPs con DNS expuesto...")
    resultados = buscar_dns_expuestos(SHODAN_API_KEY, max_results=20)

    print("[*] Verificando resolución DNS...")
    for ip, port in resultados:
        ok, resultado = probar_resolucion(ip)
        estado = "✅" if ok else "❌"
        print(f"{estado} {ip}:{port} -> {resultado}")

if __name__ == "__main__":
    main()
```

Imagen #9,10,11. Código de archivos iniciales sugeridos

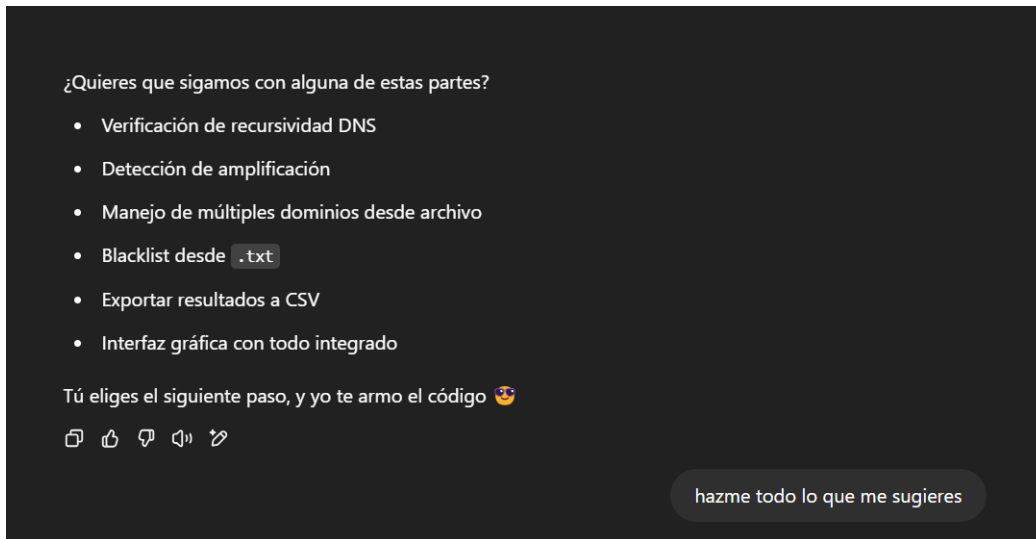


Imagen #12. Prompt 3

El prompt 3 fue muy corto porque chatgpt ya tenia en cuenta el siguiente paso de la aplicación para complementar lo últimos detalles en ese orden de ideas se logro lo siguiente:

#### Funciones Básicas

- **Búsqueda de IPs con puerto 53 abierto** usando Shodan.
- **Prueba de resolución DNS** a dominios conocidos (google.com, example.com).

#### Funciones Avanzadas

- **Verificación de recursividad DNS** (responde a consultas fuera de su zona).
- **Detección de amplificación DNS** (respuesta desproporcionada vs solicitud).
- **Paginación Shodan** con search\_cursor.
- **Soporte para múltiples dominios de prueba** desde una lista interna.
- **Carga y verificación de lista negra** desde archivo blacklist.txt

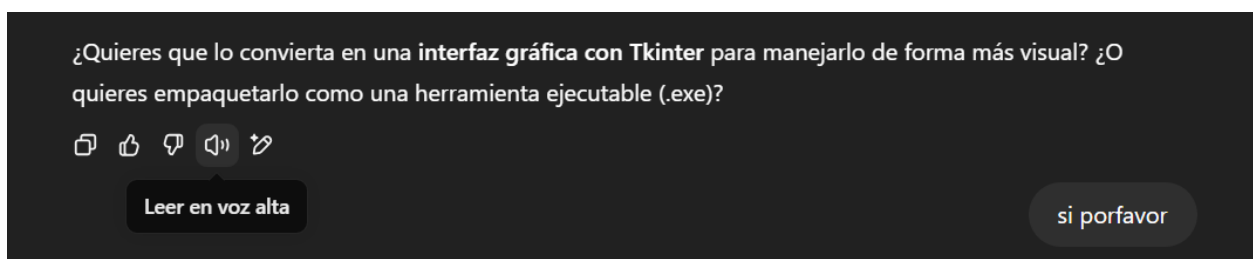


Imagen #13. Prompt 4

Otra vez chatgpt me sugiere convertir la aplicación en una interface grafica para poder manejarlo de forma visual entonces el prompt fue decirle que si acepto la propuesta.

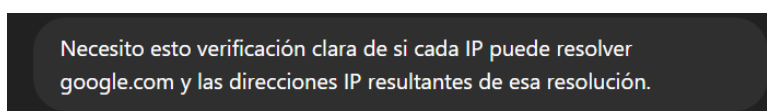


Imagen #14. Prompt 5

Después de haber creado la interfaz gráfica le pedí que por ejemplo cuando se ejecute Google.com de si cada ip puede resolver Google.com con las direcciones ip resultantes .

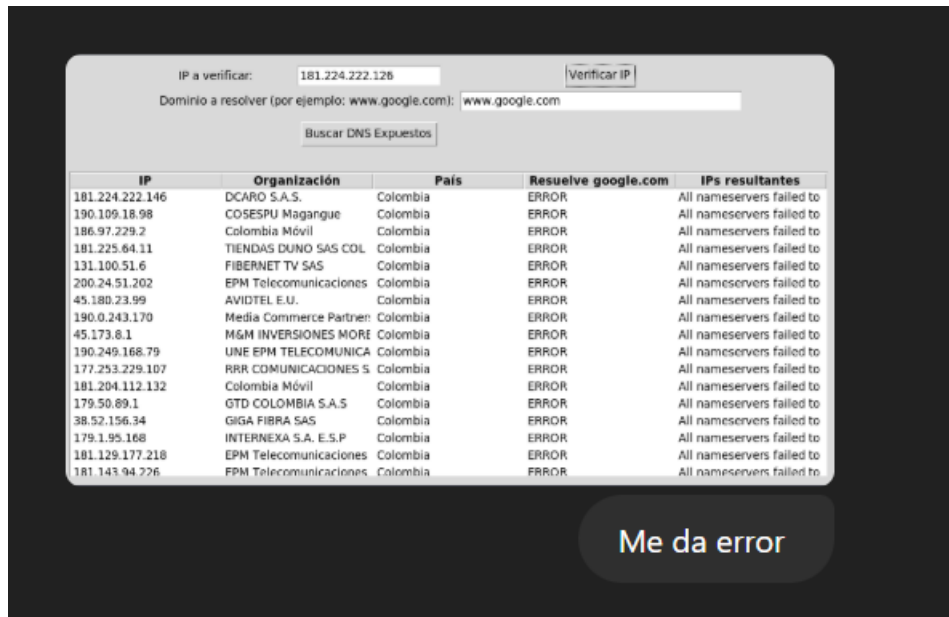


Imagen #15. Prompt 6

En este caso note que cuando buscaba el dominio de Google encontraba ips de organizaciones pero no los resolvía, entonces le comente a chatgpt que me daba un error entonces la IA me sugiere que quizás hay un problema al verificar el puerto 53 porque la maquina estaba en puente y tocaba configurarla para que se conecte la maquina virtual Kali en modo NAT.

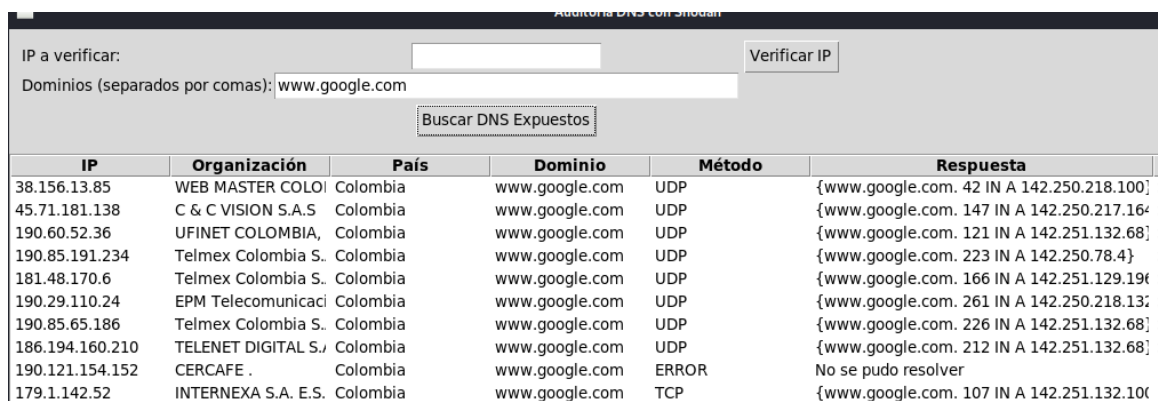


Imagen #16. Corrección modo de conexión

En esta imagen podemos ver que al buscar dominio de Google ya tenemos una respuesta clara de la ip de algunas organizaciones

Verificación de recursividad.  
Paginación en Shodan (o herramienta alternativa).  
quiero que se pueda integrar con comas un dominio al otro

Imagen #17. Prompt 7

Luego le pedí añadir la verificación de recursividad, Paginacion de shodan y poder integrar con comas un domino al lado del otro





```
(kali㉿kali)-[~/ProyectoDNS]
$ python3 principal11.py -s -d google.com,wikipedia.org

=====
AVISO LEGAL:
Esta herramienta es solo para fines educativos y de auditoría autorizada.
El uso inadecuado puede ser ilegal. El autor no se hace responsable del uso indebido.
=====

[*] Buscando servidores DNS públicos con Shodan...

[+] IP: 38.52.158.69
    Organización: GIGA FIBRA SAS
    País: Colombia
    ¿Recurso?: Si
    Dominio: google.com | Método: TCP | Respuesta: ['google.com. 263 IN A 142.251.133.110']
    Dominio: wikipedia.org | Método: UDP | Respuesta: ['wikipedia.org. 290 IN A 208.80.154.224']

[+] IP: 190.60.43.54
    Organización: UFINET COLOMBIA, S. A.
    País: Colombia
    ¿Recurso?: Si
    Dominio: google.com | Método: UDP | Respuesta: ['google.com. 156 IN A 172.217.172.14']
    Dominio: wikipedia.org | Método: UDP | Respuesta: ['wikipedia.org. 295 IN A 208.80.154.224']

[+] IP: 152.231.27.245
    Organización: COLOMBIA TEL TELECOMUNICACIONES
    País: Colombia
```

Imagen #22. Herramienta hecha en CLI

En esta imagen podemos observar que ejecutamos el archivo principal11.py mediante el comando `python3 principal11.py` en donde está el código en CLI y nos muestra primero el aviso legal sobre el uso de esta herramienta. Luego busca las IPs encontradas con su nombre, origen, si es recursivo o no y su dominio.

```
(kali㉿kali)-[~/ProyectoDNS]
$ shodan info
Query credits available: 26
Scan credits available: 100
```

Imagen #23. consumo de credenciales API Shodan

En esta imagen podemos ver cuántas credenciales quedaron después del uso de la aplicación con varias búsquedas realizadas.

```
File Actions Edit View Help
GNU nano 8.3 principal10.py

import shodan
import dns.resolver
import dns.message
import dns.query
import csv
import time
import tkinter as tk
from tkinter import ttk, messagebox

# ===== CONFIG =====
SHODAN_API_KEY = "UduhN5TASmISl0a9C0sa1aC0K3TvCnNL"
MAX_RESULTADOS = 40
RESULTADOS_POR_PAGINA = 10

def buscar_dns_expuestos(api_key, query="port:53 country:CO", max_results=30):
    api = shodan.Shodan(api_key)
    resultados = []
    try:
        for resultado in api.search_cursor(query):
            ip = resultado.get('ip_str')
            org = resultado.get('org', 'Desconocida')
            pais = resultado.get('location', {}).get('country_name', 'Desconocido')
            if ip:
                resultados.append((ip, org, pais))
    except Exception as e:
        messagebox.showerror("Error", str(e))

# =====
```

Imagen #24. Credenciales de shodan en código Python

Tu código sí cumple con las funciones básicas y varias funciones avanzadas requeridas. Aquí te detallo punto por punto para que lo veas claro:

#### ✓ Funciones Básicas

- **Búsqueda de IPs con DNS expuesto:**
  - ✓ Usas `shodan.search_cursor("port:53 country:co")` para encontrar IPs con el puerto 53 abierto.
- **Verificación de resolución DNS (a dominios específicos):**
  - ✓ La función `probar_resolucion` usa `dns.query.udp` y `dns.query.tcp` para probar si una IP puede resolver los dominios proporcionados.

*Imagen #25. Funcionalidades Básicas*

En esta imagen observamos el cumplimiento de las funcionalidades básicas de nuestra aplicación como la búsqueda de IPs con DNS expuesto y la verificación de resolución DNS a dominios específicos

#### ✓ Funciones Avanzadas (cumples con varias)

- **Verificación de recursividad:**
  - ✓ La función `verificar_rekursividad` consulta `www.google.com` para ver si la IP realiza resolución recursiva.
- **Paginación en Shodan (o herramienta alternativa):**
  - ✓ Tienes paginación implementada en la interfaz con `pagina_actual`, `mostrar_pagina`, `siguiente_pagina`, etc.
- **Múltiples dominios de prueba:**
  - ✓ Permites ingresar múltiples dominios separados por comas, que se prueban contra cada IP.

*Imagen #26. Funcionalidades Básicas*

Aquí podemos observar el cumplimiento de las funciones avanzadas como la verificación de recursividad, paginación en shodan y múltiples dominios de prueba

## Conclusiones sobre la herramienta de auditoría DNS

1. *Búsqueda de IPs con DNS expuesto (Shodan)*  
La herramienta permite identificar servidores que tienen el puerto 53 abierto (protocolo DNS), lo cual es fundamental para detectar posibles configuraciones inseguras. Esta búsqueda se realiza mediante la API de Shodan, lo que agiliza y automatiza el proceso de recolección de objetivos.
2. *Verificación de resolución DNS*  
Se incluye una prueba directa para verificar si un servidor DNS resuelve un dominio específico. Esto permite comprobar si el servicio está funcionando correctamente y puede utilizarse para validar configuraciones o detectar servidores abiertos que podrían ser abusados para ataques.
3. *Verificación de recursividad*  
Esta característica identifica si el servidor permite consultas recursivas, lo cual representa una vulnerabilidad potencial si no está restringido. Un servidor recursivo mal configurado puede ser explotado en ataques de amplificación DNS.
4. *Paginación en Shodan*  
Se implementó la paginación para consultar múltiples páginas de resultados, permitiendo una búsqueda más exhaustiva y un mejor aprovechamiento de la cuota de resultados en la API de Shodan. Esto es esencial en auditorías a gran escala.
5. *Múltiples dominios de prueba*  
La herramienta permite especificar varios dominios a la vez, facilitando pruebas paralelas y más completas. Esto mejora la eficiencia del análisis y permite detectar inconsistencias o configuraciones específicas por dominio.

### Sobre el uso de NAT para el puerto 53

#### Importancia de la conexión NAT al habilitar el puerto 53

El uso de NAT (Network Address Translation) permite controlar qué servicios están expuestos desde una red interna hacia el exterior. Al abrir el puerto 53, es importante aplicar NAT de forma segura para evitar:

- Exposición involuntaria de servicios internos.
- Abuso del servidor como DNS abierto.
- Posibles ataques de amplificación si no hay control de consultas recursivas.

NAT permite limitar las conexiones entrantes a IPs específicas o definir reglas de redirección para reforzar la seguridad. Es una capa crítica de control en entornos reales.

### Aviso legal y uso responsable

#### Advertencia sobre el uso ético y legal de la herramienta

La herramienta se ha desarrollado exclusivamente con fines educativos, de auditoría y análisis de redes autorizadas. Su uso en redes sin permiso puede ser ilegal y violar leyes de ciberseguridad y privacidad.

#### Importancia del aviso legal

Incluir un aviso legal visible al inicio de la aplicación (en GUI o CLI) es una buena práctica que protege tanto al desarrollador como a los usuarios, y subraya la ética del uso responsable de herramientas de ciberseguridad.