**Ícone

O conteúdo gerado por IA pode estar incorreto.**

**Firewall\_ai - Scanner de Rede com IA para Infraestrutura On-Premises**

Sumário

[**Firewall\_ai - Scanner de Rede com IA para Infraestrutura On-Premises** 3](#_Toc190717417)

[**Funcionalidades** 3](#_Toc190717418)

[**Pré-Requisitos** 3](#_Toc190717419)

[**Instalação** 4](#_Toc190717420)

[**Executando o Sistema** 4](#_Toc190717421)

[**Estrutura do Projeto** 5](#_Toc190717422)

[**Funções Principais** 5](#_Toc190717423)

[**Conclusão** 12](#_Toc190717424)

# **Firewall\_ai - Scanner de Rede com IA para Infraestrutura On-Premises**

O firewall\_ai é uma aplicação desenvolvida com Flask e OpenAI para realizar varreduras de rede em um ambiente on-premises e gerar explicações sobre vulnerabilidades encontradas. Através de um simples formulário, é possível escanear portas e serviços de uma rede local e obter explicações geradas pela IA sobre os riscos e como mitigá-los.

Este sistema usa nmap para escanear a rede e OpenAI para fornecer informações detalhadas sobre vulnerabilidades, ajudando a melhorar a segurança da infraestrutura de TI.

## **Funcionalidades**

* Varredura de Rede: Realiza varredura completa de portas de um IP ou domínio utilizando o nmap.
* Análise de Vulnerabilidades: Para cada serviço encontrado, a IA explica potenciais riscos e sugere formas de mitigação.
* Interface Web: O sistema fornece uma interface web simples e interativa para facilitar o uso e a visualização dos resultados.

## **Pré-Requisitos**

Certifique-se de ter as seguintes dependências instaladas:

* Python 3.6+
* nmap: Utilizado para realizar a varredura de rede.
* Flask: Framework web utilizado para a construção da interface.
* openai: Biblioteca para integração com a API da OpenAI para análise de vulnerabilidades.

Você pode instalar as dependências do projeto utilizando o requirements.txt.

## **Instalação**

1. **Clone o repositório:**

git clone https://github.com/LeviLucena/Firewall\_ai.git

cd firewall\_ai

1. **Crie um ambiente virtual (opcional, mas recomendado):**

python -m venv venv

source venv/bin/activate # Para sistemas Unix

venv\Scripts\activate # Para sistemas Windows

1. **Instale as dependências:**

pip install -r requirements.txt

1. **Configure a chave da API da OpenAI**:

Abra o arquivo ai\_helper.py e substitua "SUA\_CHAVE\_OPENAI" pela sua chave de API da OpenAI:

openai.api\_key = "SUA\_CHAVE\_OPENAI"

Caso não tenha a chave, você pode obter uma [aqui](https://beta.openai.com/signup/)

**Uso**

## **Executando o Sistema**

Para iniciar a aplicação, execute o seguinte comando no terminal:

python app.py

A aplicação estará disponível em http://127.0.0.1:5000.

**Interface Web**

1. Acesse o sistema no navegador.
2. Insira o IP ou domínio que deseja escanear.
3. Clique no botão **Escanear** para iniciar a varredura.
4. Após a varredura, os resultados serão apresentados com as explicações geradas pela IA sobre as vulnerabilidades encontradas.

## **Estrutura do Projeto**

Interface gráfica do usuário, Texto, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

### **Funções Principais**

* **app.py**: Responsável pela configuração da aplicação Flask, pelas rotas e pela interação com os scripts de varredura e IA.
* **scanner.py**: Realiza a varredura das portas usando o nmap.
* **ai\_helper.py**: Interage com a API da OpenAI para gerar explicações sobre vulnerabilidades e como mitigá-las.
* **index.html**: Interface web que recebe os dados do usuário e exibe os resultados.
* **style.css**: Estilos personalizados para a interface web, melhorando a usabilidade.

**1️. Arquivo app.py**

O app.py será responsável por servir a aplicação Flask e lidar com a varredura e explicações de vulnerabilidades.

from flask import Flask, render\_template, request

from scanner import scan\_network

from ai\_helper import explain\_vulnerability

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

"""Carrega a página principal."""

return render\_template('index.html')

@app.route('/scan', methods=['POST'])

def scan():

"""Recebe um IP e retorna a varredura de portas + explicação da IA."""

target = request.form.get("target", "192.168.1.1")

results = scan\_network(target)

for host, ports in results.items():

for port, info in ports.items():

service = info.get("name", "desconhecido")

results[host][port]["ai\_explanation"] = explain\_vulnerability(port, service)

return render\_template('index.html', results=results)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**2️. Arquivo scanner.py**

Este script será responsável pela varredura de rede usando o nmap.

import nmap

def scan\_network(target="192.168.1.1"):

"""Faz uma varredura nas portas de um IP alvo."""

scanner = nmap.PortScanner()

scanner.scan(hosts=target, arguments='-p 1-65535 -T4')

results = {}

for host in scanner.all\_hosts():

results[host] = scanner[host]['tcp']

return results

**3. Arquivo ai\_helper.py**

Este script interage com a **OpenAI** para fornecer explicações sobre vulnerabilidades.

import openai

openai.api\_key = "SUA\_CHAVE\_OPENAI"

def explain\_vulnerability(port, service):

"""Gera uma explicação sobre os riscos e mitigação usando IA."""

prompt = f"O serviço {service} na porta {port} pode ter vulnerabilidades. Explique os riscos e como mitigar."

response = openai.ChatCompletion.create(

model="gpt-4",

messages=[{"role": "system", "content": prompt}]

)

return response["choices"][0]["message"]["content"]

**4. Arquivo templates/index.html**

Aqui estará o front-end simples em HTML para que o usuário insira o endereço IP e visualize os resultados da varredura.

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Infra IA - Scanner de Rede</title>

<link rel="stylesheet" href="{{ url\_for('static', filename='style.css') }}">

</head>

<body>

<h1>Infra IA - Scanner de Rede</h1>

<form action="/scan" method="post">

<label for="target">IP ou Domínio:</label>

<input type="text" id="target" name="target" placeholder="192.168.1.1" required>

<button type="submit">Escanear</button>

</form>

{% if results %}

<h2>Resultados</h2>

<pre>{{ results }}</pre>

{% endif %}

</body>

</html>

**5. Arquivo static/style.css**

Aqui está o estilo básico para o front-end.

body {

font-family: Arial, sans-serif;

text-align: center;

margin: 20px;

}

form {

margin-top: 20px;

}

input {

padding: 8px;

margin: 5px;

}

button {

padding: 10px;

background-color: #007BFF;

color: white;

border: none;

cursor: pointer;

}

button:hover {

background-color: #0056b3;

}

pre {

text-align: left;

background: #f4f4f4;

padding: 10px;

border-radius: 5px;

display: inline-block;

}

**6. Instalando Dependências**

Crie um arquivo requirements.txt com as dependências necessárias:

Flask

openai

nmap

Depois, instale as dependências:

pip install -r requirements.txt

**7. Rodando o Sistema**

Agora, execute o **app.py** para rodar o sistema Flask:

python app.py

Acesse no navegador:

http://127.0.0.1:5000/

**8. Melhorias**

Para melhorar o visual da sua página index.html utilizando o Bootstrap 3, ícones e aprimorar o layout para torná-lo mais profissional. Abaixo está o código atualizado para o index.html com essas melhorias.

Novo index.html com Bootstrap 3

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Infra IA - Scanner de Rede</title>

<!-- Incluindo Bootstrap 3 e Font Awesome (para ícones) -->

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.3/css/all.min.css">

<link rel="stylesheet" href="{{ url\_for('static', filename='style.css') }}">

</head>

<body>

<div class="container">

<header class="text-center">

<h1><i class="fas fa-shield-alt"></i> Firewall IA - Scanner de Rede</h1>

<p>Realize a varredura em sua rede e descubra vulnerabilidades com a ajuda de IA.</p>

</header>

<div class="row">

<div class="col-md-6 col-md-offset-3">

<!-- Formulário para entrada de IP -->

<form action="/scan" method="post" class="well well-lg">

<div class="form-group">

<label for="target"><i class="fas fa-laptop"></i> IP ou Domínio:</label>

<input type="text" id="target" name="target" class="form-control" placeholder="192.168.1.1" required>

</div>

<button type="submit" class="btn btn-primary btn-lg btn-block">

<i class="fas fa-search"></i> Escanear

</button>

</form>

</div>

</div>

{% if results %}

<div class="row">

<div class="col-md-12">

<h2><i class="fas fa-check-circle"></i> Resultados da Varredura</h2>

<pre class="well well-lg" style="background-color: #f8f9fa;">{{ results }}</pre>

</div>

</div>

{% endif %}

</div>

<!-- Incluindo jQuery, Bootstrap 3 e Font Awesome -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</body>

</html>

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

**O que mudou e melhorias feitas:**

1. **Uso do Bootstrap 3**:
   * A página agora usa o Bootstrap 3 para um layout mais moderno e responsivo.
   * A estrutura de containers e colunas melhora a disposição dos elementos na tela.
2. **Font Awesome**:
   * Ícones foram adicionados para tornar a interface mais amigável e visualmente agradável, como ícones de rede, busca e resultados.
3. **Formulário estilizado**:
   * O formulário foi colocado dentro de uma caixa (well well-lg) para dar um aspecto mais limpo e profissional.
   * O botão de envio agora possui o ícone de busca, além de uma estilização mais moderna.
4. **Melhor apresentação dos resultados**:
   * Quando os resultados são apresentados, eles estão dentro de uma caixa estilizada para melhorar a leitura e tornar a interface mais limpa.
5. **Responsividade**:
   * O layout foi ajustado para garantir que a página seja responsiva, ou seja, adaptada para diferentes tamanhos de tela, o que ajuda na usabilidade em dispositivos móveis.

**Exemplo de Como Ficará o Visual:**

* **Cabeçalho**: Título com ícone de rede e uma descrição curta explicando o propósito do sistema.
* **Formulário**: Com uma caixa de entrada para IP ou domínio e um botão grande e destacado para iniciar a varredura.
* **Resultado da Varredura**: Apresentação dos resultados com uma caixa que destaca os dados, incluindo as explicações da IA.

**Próximos Passos**

1. **Testar no Navegador**: Assim que você implementar o novo index.html no seu projeto, o layout será melhorado, ficando mais organizado e profissional.
2. **Customizar o Estilo**: Caso queira mais personalização, você pode mexer no arquivo style.css que está na pasta static para modificar cores, fontes ou adicionar animações.

# **Conclusão**

Com esse sistema, você consegue realizar uma **varredura de rede** dentro de um ambiente **on-premises**, analisar as **vulnerabilidades** encontradas e obter explicações sobre as falhas diretamente da **OpenAI**. A interface web permite que você insira um IP ou domínio e visualize os resultados da análise de segurança.