



UNIVERSIDADE  
LUSÓFONA

# Trabalho Final de Curso

**Guia de Instalação API Monitorizador**

**Miguel Lourenço**

**Vasco Pereira**

27/04/2025

[www.ulusofona.pt](http://www.ulusofona.pt)

# Índice

Índice .....	ii
Lista de Figuras .....	iii
Resumo .....	4
Guia de Instalação do Monitorizador .....	5
Inicialização da API .....	8
Requests à API .....	10
Endpoints da API.....	12

## Lista de Figuras

Figura 1 .....	6
Figura 2 .....	7
Figura 3.....	7
Figura 4 .....	9
Figura 5 .....	10
Figura 6 .....	11
Figura 7 .....	11
Figura 8 .....	12
Figura 9 .....	<b>Erro! Marcador não definido.</b>

## **Resumo**

Será disponibilizado um guia de instalação detalhado, que orienta o utilizador ao longo de todas as etapas necessárias para a configuração da API. Este guia inclui a lista de requisitos prévios, instruções para a instalação de dependências, definições de ambiente e todas as configurações necessárias para garantir o correto funcionamento da aplicação.

# Guia de Instalação do Monitorizador

## Pré-Requisitos

Para executar o projeto localmente, é necessário garantir que o ambiente cumpre os seguintes requisitos, de modo a permitir a replicação completa deste guia.

### Sistemas Operativos:

1. *Linux* (Ubuntu 22.04 ou superior 22.04 ou 24.04 LTS)
2. *macOS*

Todos os sistemas operativos recomendados foram devidamente testados embora seja tecnicamente possível utilizar outra distribuição Linux, não se pode garantir o funcionamento integral do sistema conforme o esperado.

### É necessário que a máquina cumpra os seguintes requisitos:

1. *Python* 3.10
2. *Git*
3. Uma IDE à escolha (recomenda-se o *VSCode* ou o *PyCharm*)
4. *Postman* (opcional)
5. *ExploitDB*
6. *Masscan*

## Instalação dos principais requisitos

### Python

Para instalar o *Python*, é necessário aceder ao terminal da máquina e executar os seguintes comandos:

- `sudo apt update`
- `sudo apt install python3`

### Git

Para instalar o *Git*, é necessário aceder ao terminal da máquina e executar o seguinte comando:

- `sudo apt install git`

### Exploitdb

Para instalar o *ExploitDB*, recomenda-se consultar a documentação oficial, de forma a garantir uma instalação completa e atualizada. A documentação está disponível em:

<https://www.exploit-db.com/searchsploit>

## Masscan

Para instalar o *Masscan*, é necessário aceder ao terminal da máquina e executar os seguintes comandos:

- `sudo apt update`
- `sudo apt install masscan`

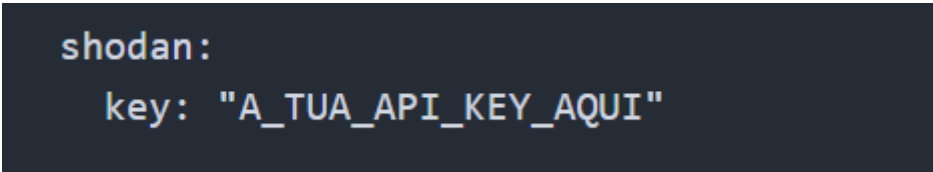
Para além das ferramentas mencionadas acima, é também necessário requerer e configurar as chaves de API, de forma a garantir o funcionamento adequado dos serviços utilizados. Esta configuração deve ser efetuada no ficheiro: `api_keys.yaml`

## Keys:

- *Shodan*
- *HackerTarget*
- *SecurityTrails*
- *urlSan.Io*
- *OxSi\_f33d*
- *Lookup*
- *Blacklist Checker*
- *Opencti*

## Figura 1

*Exemplo de como configurar as keys*



```
shodan:  
  key: "A_TUA_API_KEY_AQUI"
```

É necessário configurar o ficheiro `configs.yaml` com a interface de rede correta. Para identificar a interface em uso, pode utilizar o seguinte comando no terminal:

- `ip a`

Figura 2

Exemplo de requisição da interface de internet ens33

```
migu@migu-VMware-Virtual-Platform:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 q
    link/loopback 00:00:00:00:00:00 brd 0
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft fo
    inet6 ::1/128 scope host noprefixrout
        valid_lft forever preferred_lft fo
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_U
    link/ether 00:0c:29:84:6b:26 brd ff:f
    altname enp2s1
    inet 192.168.1.104/24 brd 192.168.1
        valid_lft 1685sec preferred_lft 16
    inet6 fe80::20c:29ff:fe84:6b26/64 sco
        valid_lft forever preferred_lft fo
```

- masscan\_interface: "ens33" (exemplo)

É ainda necessário configurar no mesmo ficheiro o url da máquina que contem o opencti de modo

Figura 3

Exemplo de url Opencti

```
opencti_url: 'http://139.59.163.170:8080/'
```

Após garantir que todas as dependências mencionadas foram corretamente instaladas, podemos proceder com a instalação do projeto.

# Inicialização da API

## Clonar repositório

Para clonar o repositório para a máquina local, deve-se executar o seguinte comando no terminal:

- `git clone https://github.com/duke-the-1998/TFC-Lusofona-API`

## Criar e Ativar um ambiente Virtual

O *Python* utiliza ambientes virtuais, nos quais é possível instalar as bibliotecas necessárias para a execução do projeto. Para criar e ativar o ambiente virtual, devem ser utilizados os seguintes comandos:

- `sudo apt install python3.12-venv`
- `python -m venv venv`
- `source venv/bin/activate`

## Instalar dependências

O repositório já inclui as dependências necessárias para o funcionamento do projeto. Para instalá-las, deve-se executar o seguinte comando:

- `pip install -r requirements.txt`

## Como correr API

Após concluir todos os procedimentos acima descritos, podemos proceder à execução do projeto. O mesmo deve ser executado a partir do terminal. Como no Linux não é possível utilizar o `sudo` fora do ambiente virtual, e como o `sudo` é necessário para a execução de certos componentes no diretório principal do projeto, deve-se executar o seguinte comando:

- `which python`

Deste modo, será possível obter o caminho até ao *Python* dentro do ambiente virtual. Após isso, podemos executar o seguinte comando:

- `sudo /home/user/PycharmProjects/TFC-Lusofona-API/.venv/bin/python monitorizador.py`

Este comando irá iniciar o servidor *Flask* local na porta 5000, acessível em <http://127.0.0.1:5000/>. No entanto, a porta pode ser alterada no ficheiro `monitorizador.py`.



*Figura 4*

*Porta padrão flask*

```
if __name__ == "__main__":  
    app.run(port=5000)
```

# Requests à API

Existem três formas principais de interagir com os endpoints da API:

## Postman

Pode utilizar o Postman para testar os endpoints expostos pela API.

Exemplo de URL de request:

- <http://127.0.0.1:5000/monitorizador/DOM>

Neste exemplo:

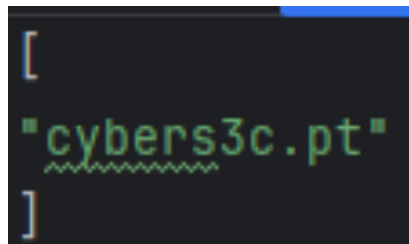
- 5000 que é a porta definida no servidor *Flask*
- monitorizador/DOM representa o *endpoint* que queremos fazer um *request*

O método HTTP a ser utilizado deve ser o POST. No *Postman*, essa opção pode ser selecionada no menu suspenso à esquerda do campo onde é inserido o URL da requisição.

Após definir o método HTTP e o URL, é necessário inserir no corpo da requisição (*body*) uma lista de IPs ou domínios que se deseja consultar, no formato de lista, como exemplificado abaixo:

*Figura 5*

*Exemplo de requisição*



```
[  
  "cybers3c.pt"  
]
```

## script teste\_conexao\_api.py

Foi também desenvolvido um script para facilitar a realização de testes automáticos através da linha de comandos.

Exemplo de execução:

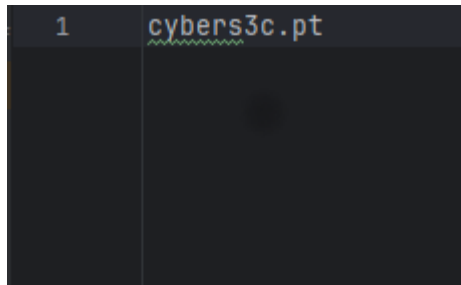
- `python teste_conexao_api.py monitorizador -t DOM -a test.txt`

Neste exemplo:

- DOM é o tipo de scan que se quer realizar
- test.txt é onde vai estar a lista dos alvos que queremos analisar

Figura 6

Exemplo de requisição

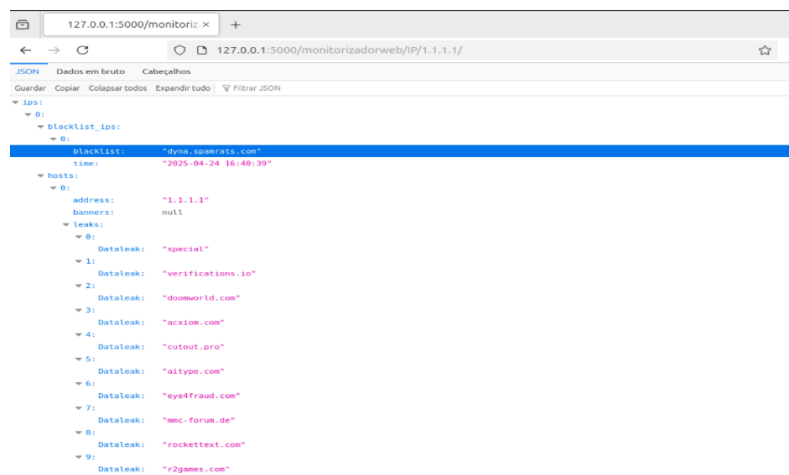


## Request browser

É possível requisitar alguns endpoints diretamente através do *browser*; no entanto, essa abordagem está limitada aos endpoints que utilizam o método GET, como é o caso de monitorizadorweb e outros semelhantes.

Figura 7

Exemplo de Requisição



# Endpoints da API

## Monitorização de Ips

Este *endpoint* retorna informações detalhadas sobre um determinado IP fornecido, bem como sobre os IPs associados. Foi desenvolvido com o método POST, permitindo receber um ficheiro JSON contendo uma lista de IPs.

Parâmetros:

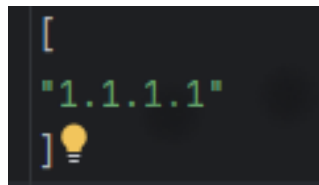
- **Ip's (obrigatório):** Os endereços Ip's a ser analisados. Exemplo: ["8.8.8.8"]

Exemplo de Requisição:

- <http://127.0.0.1:5000/monitorizador/IP>

*Figura 8*

*Exemplo de Requisição Post*



## Monitorizador de IPS via Web

Este *endpoint* retorna informações detalhadas sobre um determinado IP fornecido e os IPs a ele associados. Ao contrário do anterior, foi desenvolvido com o método GET, sendo, por isso, possível analisar apenas um endereço IP de cada vez.

Parâmetros

- **IP (obrigatório):** IP a ser analisado, como no exemplo: /1.1.1.1/
- 

Exemplo de Requisição:

- <http://127.0.0.1:5000/monitorizadorweb/IP/1.1.1.1/>

## Monitorização de Domínios

Este *endpoint* retorna informações detalhadas sobre um determinado domínio fornecido, bem como sobre todos os seus subdomínios. Foi desenvolvido com o método POST, permitindo o envio de um ficheiro JSON contendo uma lista de domínios ou subdomínios.

Parâmetros

- **domínios(obrigatório):** Domínios a analisar. Exemplo: ["cybers3c.pt", "teste.pt"]

Exemplo de Requisição:

- <http://127.0.0.1:5000/monitorizador/DOM>

### Monitorizador de Domínios via Web

Este *endpoint* retorna informações detalhadas sobre um determinado domínio fornecido, bem como os seus subdomínios associados. Ao contrário do anterior, foi desenvolvido com o método GET, sendo por isso possível analisar apenas um domínio de cada vez.

Parâmetros:

- **domínio (obrigatório):** Domínio a ser analisado, como no exemplo: /**exemple.com/**

Exemplo de Requisição:

- <http://127.0.0.1:5000/monitorizadorweb/DOM/exemple.com/>

### LOOKUP

Este *endpoint* retorna os *data leaks* associados a um determinado IP, domínio, nome de utilizador (*username*) ou endereço de e-mail. Foi desenvolvido com o método POST, permitindo o envio de um ficheiro JSON com uma lista de IPs, domínios, *usernames* e e-mails para os quais se pretende obter essa informação.

Parâmetros:

- **IP, Domínio, Username ou mail (obrigatório)** a ser analisado

Exemplo de requisição

- <http://127.0.0.1:5000/LOOKUP/>

### CVE

Este *endpoint* retorna os scripts e *exploits* associados a um determinado CVE pesquisado. Foi desenvolvido com o método POST, permitindo o envio de um ficheiro JSON com uma lista de CVEs para consulta.

Parâmetros:

- **Cves (Obrigatório):** CVE a ser analisado como no exemplo: ["cve-202154"]

Exemplo de requisição:

- <http://127.0.0.1:5000/CVE/>

## pesquisar-cve

Este *endpoint* retorna todos os CVEs associados a uma determinada palavra-chave ou tecnologia. Foi desenvolvido com o método POST, permitindo o envio de um ficheiro JSON contendo uma lista de palavras-chave para procurar os CVEs correspondentes.

Parâmetros:

- **Palavras (Obrigatório):** Palavra a ser analisada como no exemplo: ["Ubuntu "]

Exemplo de requisição:

- <http://127.0.0.1:5000/pesquisa-cve/>

## OxSI\_feed

Este *endpoint* retorna a consulta do OxSI\_f33d da segurança informática em formato JSON. Foi desenvolvido com o método POST.

Exemplo de Requisição:

- [http://127.0.0.1:5000/OxSI\\_feed/](http://127.0.0.1:5000/OxSI_feed/)

## ctipais

Este *endpoint* retorna a consulta sobre o *feed* do OpenCTI relacionado a ataques direcionados a um país durante um período definido.

Parâmetros:

- **/ctipais/<typeScan>/<typeScan2>/<typeScan3>/**

O primeiro parâmetro (<typeScan>) é obrigatório e corresponde ao nome do país.

Exemplo:

- "United Kingdom of Great Britain and Northern Ireland"

O segundo (<typeScan2>) e terceiro parâmetro (<typeScan3>) correspondem à data do período de procura. Ambos devem seguir o formato MM-AAAA.

Exemplo:

- "03-2025"

Exemplos de Requisição:

- <http://127.0.0.1:5000/ctipais/Portugal/03-2025/04-2025/>

- <http://127.0.0.1:5000/ctipais/Portugal/None/03-2025/>
- <http://127.0.0.1:5000/ctipais/Portugal/03-2025/>

### ctiTOP10

Este *endpoint* retorna a consulta sobre o *feed* do *OpenCTI* relacionado aos 10 principais ataques direcionados a um determinado país. Foi desenvolvido com o método GET.

Parâmetros:

- **ctiTOP10/<typeScan>/ (Obrigatório)** como no exemplo – Portugal

Exemplos de Requisição:

- <http://127.0.0.1:5000/ctiTOP10/Portugal/>

### ctiweb

Este *endpoint* retorna a consulta sobre os *feeds* do *OpenCTI* relacionados aos *hashes*, domínios e IPs do último mês. Foi desenvolvido com o método GET.

Parâmetros:

Este endpoint oferece **três subopções** para retornar o feed de **hashes**, **IPs** ou **domínios**:

- O parâmetro **<typeScan>** é **obrigatório** e define o tipo de feed a ser retornado. Os valores possíveis são:
  - hashes
  - DOM (domínios)
  - IPS

Exemplos de Requisição:

- <http://127.0.0.1:5000/ctiweb/DOM/>
- <http://127.0.0.1:5000/ctiweb/IPS/>
- <http://127.0.0.1:5000/ctiweb/hashes/>