**# IOC Analyzer - Blue Team**

Sistema profissional para análise e gerenciamento de Indicadores de Comprometimento (IOCs) voltado a times de Segurança Cibernética e Blue Team.

**## ✨ Funcionalidades**

- Análise automática de logs com detecção de IPs, domínios, URLs e hashes maliciosos

- Verificação de reputação automática via AbuseIPDB (IPs) e VirusTotal (domínios, URLs, hashes)

- Classificação instantânea dos IOCs (Malicioso, Suspeito, Não Malicioso)

- CRUD completo e interface web moderna para gestão dos IOCs

- Exportação de IOCs em JSON

- Dados e modificações persistentes (não perde dados ao reiniciar)

- Suporte à exportação e revisão rápida dos indicadores

**## 💻 Demonstração**

***### Tela de Upload e Análise de Logs***

![Upload de Logs](docs/img/upload-analyzer.png)

***### Painel CRUD - Gerenciamento de IOCs***

![CRUD dos IOCs](docs/img/painel-crud.png)

***### Painel CRUD - Visualização Detalhada***

![CRUD Detalhado](docs/img/painel-crud-1.png)

**## 📋 Requisitos**

- Python 3.8+

- Flask

- Requests

- python-dotenv

- AbuseIPDB API Key

- VirusTotal API Key

**## 🛠 Instalação**

1. Clone o repositório:

git clone <https://github.com/pablonoliveira/ioc_analyzer.git>  
cd ioc\_analyzer

2. Crie e ative um ambiente virtual (opcional e recomendado):

python -m venv env

**Ative no Windows:**

.\env\Scripts\activate

**Ou no Linux/Mac:**

source env/bin/activate

3. Instale as dependências:

pip install -r requirements.txt

4. Configure as chaves de API:

- Crie um arquivo `.env` na raiz do projeto, usando `ioc/.env.example` como modelo:

```

ABUSEIPDB\_KEY=sua\_chave\_abuseipdb

VIRUSTOTAL\_API\_KEY=sua\_chave\_virustotal

```

- \*\*Nunca faça commit do seu .env!\*\* Ele já está protegido no `.gitignore`.

**## 🚀 Executando a aplicação**

python webapp.py

- Upload de Logs: [http://127.0.0.1:5000/](http://127.0.0.1:5000/)

- Gerenciar IOCs: [http://127.0.0.1:5000/crud](http://127.0.0.1:5000/crud)

**## 📁 Estrutura do Projeto**

ioc\_analyzer/  
├── ioc/ # Integrações com AbuseIPDB/VirusTotal  
├── parsers/ # Parser de logs para extração de IOCs  
├── templates/ # Templates HTML (painel web CRUD)  
├── utils/ # Utilidades e logger  
├── data/ # Banco de dados dos IOCs (persistente)  
├── docs/img/ # Imagens e prints de demonstração  
├── requirements.txt # Dependências do projeto  
├── webapp.py # Servidor Flask principal  
└── README.md

**## 🔒 Segurança**

- Suas chaves de API ficam sempre no `.env` (excluído do controle de versão).

- As chaves devem ser obtidas em:

- [AbuseIPDB](https://www.abuseipdb.com/)

- [VirusTotal](https://www.virustotal.com/)

**## 📝 Licença**

GPL-3.0. Projeto aberto para fins educacionais e profissionais de Blue Team.

**## 🤝 Contribuições**

Contribuições são bem-vindas! Abra issues ou pull requests para sugerir melhorias.

**## 👤 Autor**

\*\*Pablo Nunes de Oliveira\*\* |

Analista de Segurança da Informação | Blue Team

[LinkedIn](https://www.linkedin.com/in/pabloliveira/) | [Email](mailto:pabloliveir@gmail.com)

---

\*\*Desenvolvido com foco em profissionais de Cibersegurança e operações de Defesa (Blue Team).\*\*

**IOC Analyzer - Blue Team v2 🔍🛡️**

Sistema profissional para análise e gerenciamento de Indicadores de Comprometimento (IOCs) e Vulnerabilidades (CVEs) voltado a times de Segurança Cibernética e Blue Team.

**✨ Funcionalidades**

**📤 Análise de Logs**

* Análise automática de logs com detecção de IPs, domínios, URLs e hashes maliciosos
* Verificação de reputação automática via **AbuseIPDB** (IPs) e **VirusTotal** (domínios, URLs, hashes)
* Classificação instantânea dos IOCs (Malicioso, Suspeito, Não Malicioso)
* Salvamento automático de IOCs detectados

**🔍 IOC Panel**

* CRUD completo para gerenciamento de IOCs
* Busca rápida via API (AbuseIPDB, VirusTotal)
* Filtros avançados por severidade, status e tipo
* Descrições detalhadas: país, ISP, total de reports
* Exportação de IOCs em JSON

**🛡️ CVE Panel (NOVO v2)**

* Gerenciamento completo de CVEs
* Busca em APIs externas: **CISA KEV** e **CIRCL CVE**
* **Tradução automática** de descrições para português
* Informações detalhadas: severidade, exploitação, referências
* Filtros por severidade e status de exploitação

**📊 Dashboard Interativo (NOVO v2)**

* **6 Gráficos em tempo real** usando Chart.js:
  + IOCs por Severidade (pizza com porcentagens)
  + CVEs por Severidade (pizza com porcentagens)
  + IOCs por Status (rosca)
  + CVEs Exploitadas (rosca)
  + IOCs por Tipo (barras)
  + Distribuição Geral (barras combinado)
* Cards de estatísticas com totais consolidados
* Interface moderna com tema escuro profissional
* Navegação integrada entre todas as páginas

**🌐 Recursos Adicionais v2**

* **Tradução automática** (português) via Google Translator API
* **Porcentagens automáticas** nos gráficos
* **Design responsivo** e animações suaves
* Dados e modificações persistentes (não perde dados ao reiniciar)

**💻 Demonstração**

**Dashboard Principal**

*Dashboard com 6 gráficos interativos mostrando estatísticas consolidadas de IOCs e CVEs*

**Painel IOC - Gerenciamento**

*Painel de gerenciamento de Indicadores de Comprometimento com busca via API*

**Painel CVE - Gerenciamento de Vulnerabilidades**

*Painel de gerenciamento de Vulnerabilidades com consulta em APIs externas e tradução automática*

**Tela de Upload e Análise de Logs**

*Interface de upload de logs com análise automática e extração de IOCs*

**Painel CRUD - Gerenciamento Completo**

*CRUD completo para gerenciamento de IOCs e CVEs*

**📋 Requisitos**

* Python 3.8+
* Flask 3.0
* Requests
* python-dotenv
* googletrans 4.0.0rc1
* AbuseIPDB API Key
* VirusTotal API Key

**🛠 Instalação**

**1️⃣ Clone o repositório:**

bash

git clone https://github.com/pablonoliveira/ioc\_analyzer.git

cd ioc\_analyzer

**2️⃣ Crie e ative um ambiente virtual (recomendado):**

bash

python -m venv env

*# Ative no Windows:*

.\env\Scripts\activate

*# Ou no Linux/Mac:*

source env/bin/activate

**3️⃣ Instale as dependências:**

bash

pip install -r requirements.txt

**4️⃣ Configure as chaves de API:**

Crie um arquivo .env na raiz do projeto, usando ioc/.env.example como modelo:

text

ABUSEIPDB\_KEY=sua\_chave\_abuseipdb

VIRUSTOTAL\_API\_KEY=sua\_chave\_virustotal

**⚠️ Nunca faça commit do seu .env!** Ele já está protegido no .gitignore.

**Como obter as chaves:**

* **AbuseIPDB**: <https://www.abuseipdb.com/api>
* **VirusTotal**: <https://www.virustotal.com/gui/my-apikey>

**🚀 Executando a aplicação**

bash

python webapp.py

**Acesse as páginas:**

* **Upload de Logs**: <http://127.0.0.1:5000/>
* **IOC Panel**: <http://127.0.0.1:5000/ioc_panel>
* **CVE Panel**: <http://127.0.0.1:5000/cve_panel>
* **Dashboard**: <http://127.0.0.1:5000/dashboard>

**📁 Estrutura do Projeto**

text

ioc\_analyzer/

├── data/

│ ├── ioc\_database.json # Banco de dados de IOCs (persistente)

│ └── cve\_database.json # Banco de dados de CVEs (persistente)

├── docs/

│ └── img/ # Imagens e prints de demonstração

│ ├── painel-principal.png

│ ├── painel-ioc-1.png

│ ├── painel-ioc-2.png

│ ├── painel-cve-1.png

│ ├── painel-crud.png

│ ├── painel-crud-1.png

│ ├── painel-upload.png

│ └── upload-analyzer.png

├── ioc/ # Integrações com AbuseIPDB/VirusTotal/CISA/CIRCL

│ ├── abuseipdb\_client.py

│ ├── virustotal\_client.py

│ ├── cisa\_kev\_client.py

│ ├── circl\_cve\_client.py

│ └── url\_checker.py

├── parsers/ # Parser de logs para extração de IOCs

│ └── log\_parser.py

├── templates/ # Templates HTML (painéis web)

│ ├── ioc\_panel.html

│ ├── cve\_panel.html

│ └── dashboard.html

├── utils/ # Utilidades e logger

├── requirements.txt # Dependências do projeto

├── webapp.py # Servidor Flask principal

└── README.md

**🔌 APIs Integradas**

| **API** | **Funcionalidade** | **Endpoint** |
| --- | --- | --- |
| **AbuseIPDB** | Análise de reputação de IPs | https://api.abuseipdb.com/api/v2/check |
| **VirusTotal** | Análise de URLs, domínios e hashes | https://www.virustotal.com/api/v3/ |
| **CISA KEV** | Known Exploited Vulnerabilities | https://www.cisa.gov/known-exploited-vulnerabilities |
| **CIRCL CVE** | Base de dados de CVEs | https://cve.circl.lu/api/cve/ |

**🔒 Segurança**

* Suas chaves de API ficam sempre no .env (excluído do controle de versão)
* As chaves devem ser obtidas em:
  + [AbuseIPDB](https://www.abuseipdb.com/)
  + [VirusTotal](https://www.virustotal.com/)
* **Nunca compartilhe suas chaves** publicamente

**🗺️ Roadmap**

**Versão 3.0 (Planejado)**

* Integração com MISP (Malware Information Sharing Platform)
* Exportação de relatórios em PDF
* Sistema de alertas via e-mail/Telegram
* Autenticação de usuários
* API REST completa
* Análise de correlação automática entre IOCs e CVEs
* Suporte a múltiplos idiomas

**📝 Licença**

GPL-3.0. Projeto aberto para fins educacionais e profissionais de Blue Team.

**🤝 Contribuições**

Contribuições são bem-vindas! Abra issues ou pull requests para sugerir melhorias.

**Como contribuir:**

1. Fork o projeto
2. Crie uma branch para sua feature (git checkout -b feature/NovaFuncionalidade)
3. Commit suas mudanças (git commit -m 'Adiciona nova funcionalidade')
4. Push para a branch (git push origin feature/NovaFuncionalidade)
5. Abra um Pull Request

**👤 Autor**

**Pablo Nunes de Oliveira**  
Analista de Segurança da Informação | Blue Team

**📊 Estatísticas do Projeto**

**📝 Changelog**

**v2.0 (Atual)**

* ✨ Dashboard interativo com 6 gráficos
* ✨ Porcentagens automáticas nos gráficos
* ✨ CVE Panel com busca em APIs externas
* ✨ Tradução automática de descrições
* ✨ Interface renovada com tema escuro
* ✨ Navegação integrada entre páginas

**v1.0**

* 🎉 Release inicial
* 📤 Upload e análise de logs
* 🔍 IOC Panel básico
* 🔌 Integração com AbuseIPDB e VirusTotal

**⭐ Agradecimentos**

* **Chart.js** - Biblioteca de gráficos interativos
* **Flask** - Framework web minimalista
* **AbuseIPDB** - API de reputação de IPs
* **VirusTotal** - Análise de malware
* **CISA** - Base de vulnerabilidades conhecidas
* **CIRCL** - Base de dados CVE

**Desenvolvido com ❤️ para profissionais de Cibersegurança e operações de Defesa**