**Relatório Prático — Simulação de Malware: Ransomware e Keylogger em Python**

Projeto realizado para desafio DIO - Santander Cibersegurança 2025

*Ambiente controlado com Python (simulação educacional)*

# 1. Resumo

Este relatório documenta um laboratório controlado de simulação de malwares em Python, focando em ransomware e keylogger. Os scripts foram implementados para demonstrar o funcionamento de criptografia de arquivos, geração de mensagens de resgate, captura de teclas e envio automático por e-mail. Tudo foi realizado em um ambiente isolado, com arquivos de teste, para fins educacionais, destacando vulnerabilidades e estratégias de defesa.

# 2. Objetivos

* Implementar um ransomware simulado que criptografa e descriptografa arquivos, gerando uma mensagem de resgate.
* Desenvolver um keylogger que captura teclas, salva em arquivo e envia logs por e-mail de forma periódica.
* Registrar os passos de criação, execução e testes.
* Refletir sobre riscos associados e propor medidas de mitigação contra malwares reais.

# 3. Ambiente de Teste (Configuração)

Topologia:  
- Ambiente local: Windows com VS Code como editor de código.  
- Python 3.11+ (interpretador).  
- Pastas de teste:

- MALWARE/Test\_files: Contendo arquivos como "dados.confidenciais" e "senhas.txt" para simulação de criptografia.

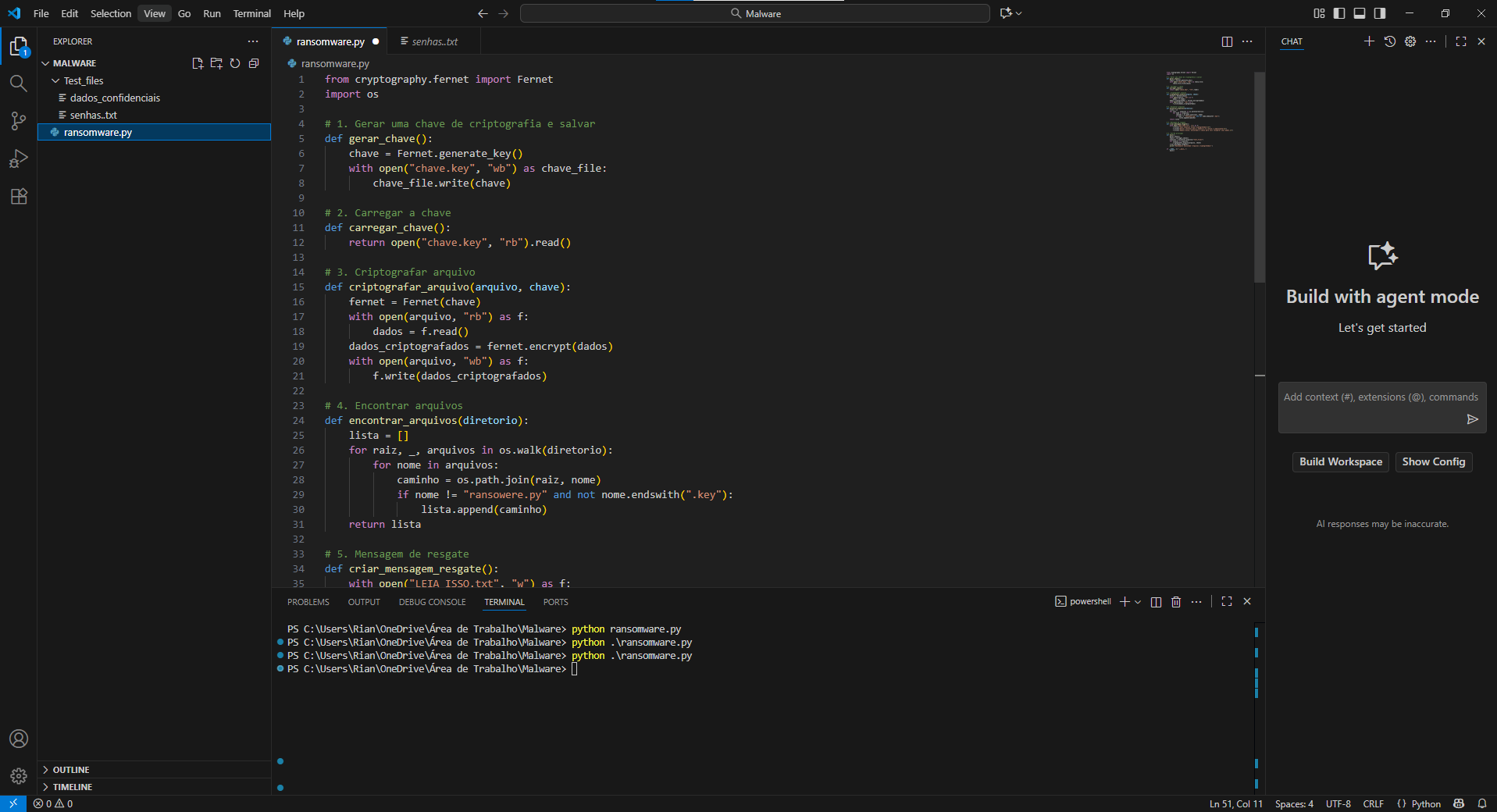
- KEYLOGGER: Para logs do keylogger (arquivo "log.txt").

# 4. Ferramentas

* Python (interpretador principal).
* Bibliotecas: cryptography (para Fernet em criptografia), pynput (para captura de teclas), smtplib (para envio de e-mails).
* VS Code (editor de código e depurador).
* Gmail (para teste de envio de e-mails, com app password configurado).
* Arquivos de teste (exemplos usados):
* senhas.txt, dados.confidenciais (arquivos simulados como "confidenciais").
* chave.key (gerada pelo ransomware).
* log.txt (para captura de teclas).

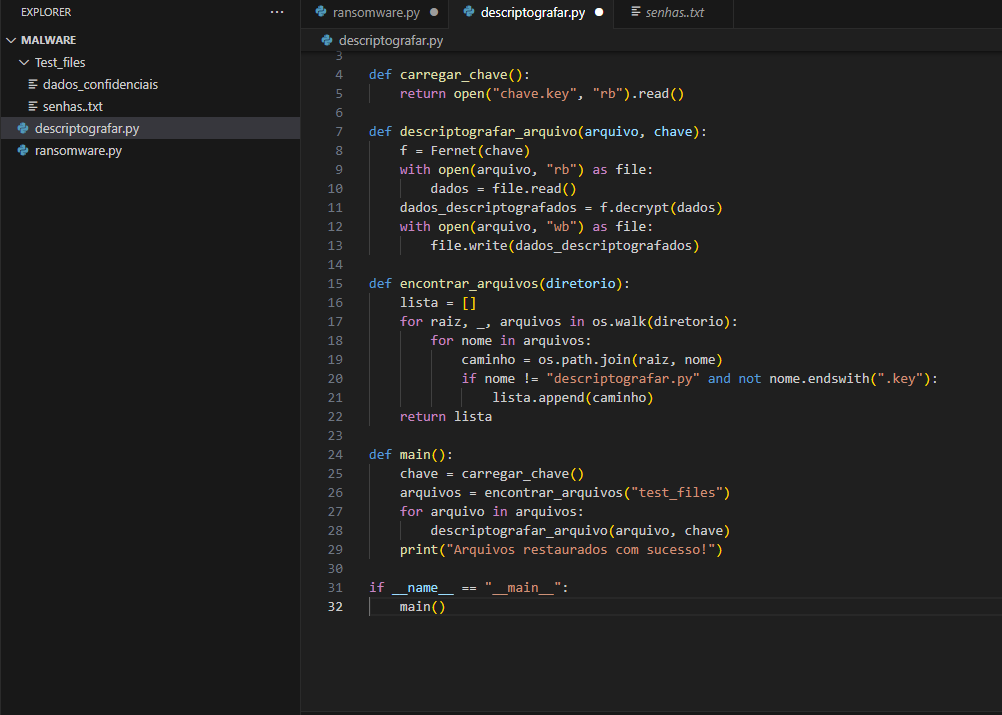
# 5. Procedimento e Comandos Utilizados

Abaixo estão os passos e códigos principais implementados.

**Criação do Ransomware:**Gerar a chave de criptografia e criptografar arquivos em “Test\_files”.  
****

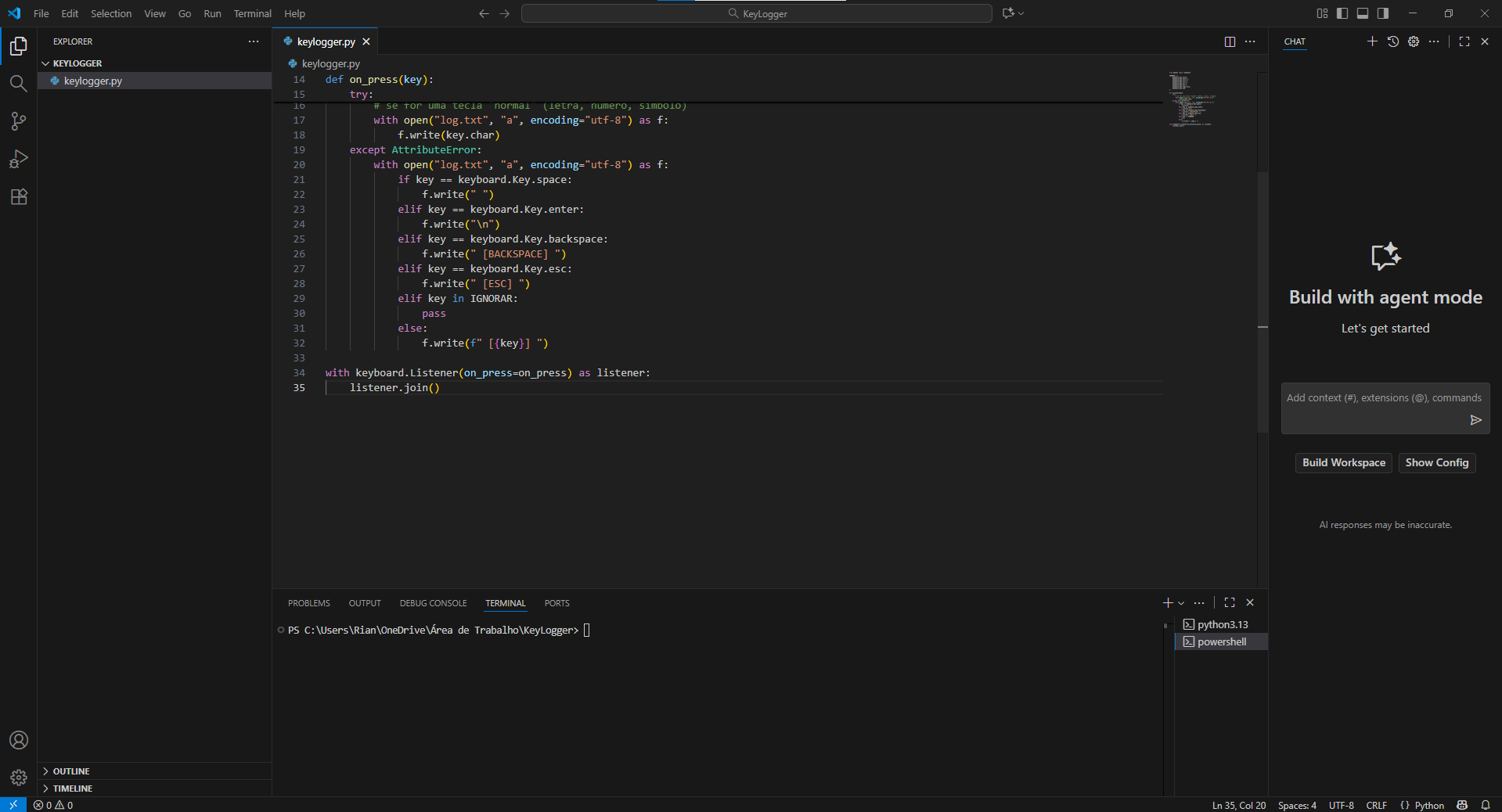
**Criação do Descriptografador:**

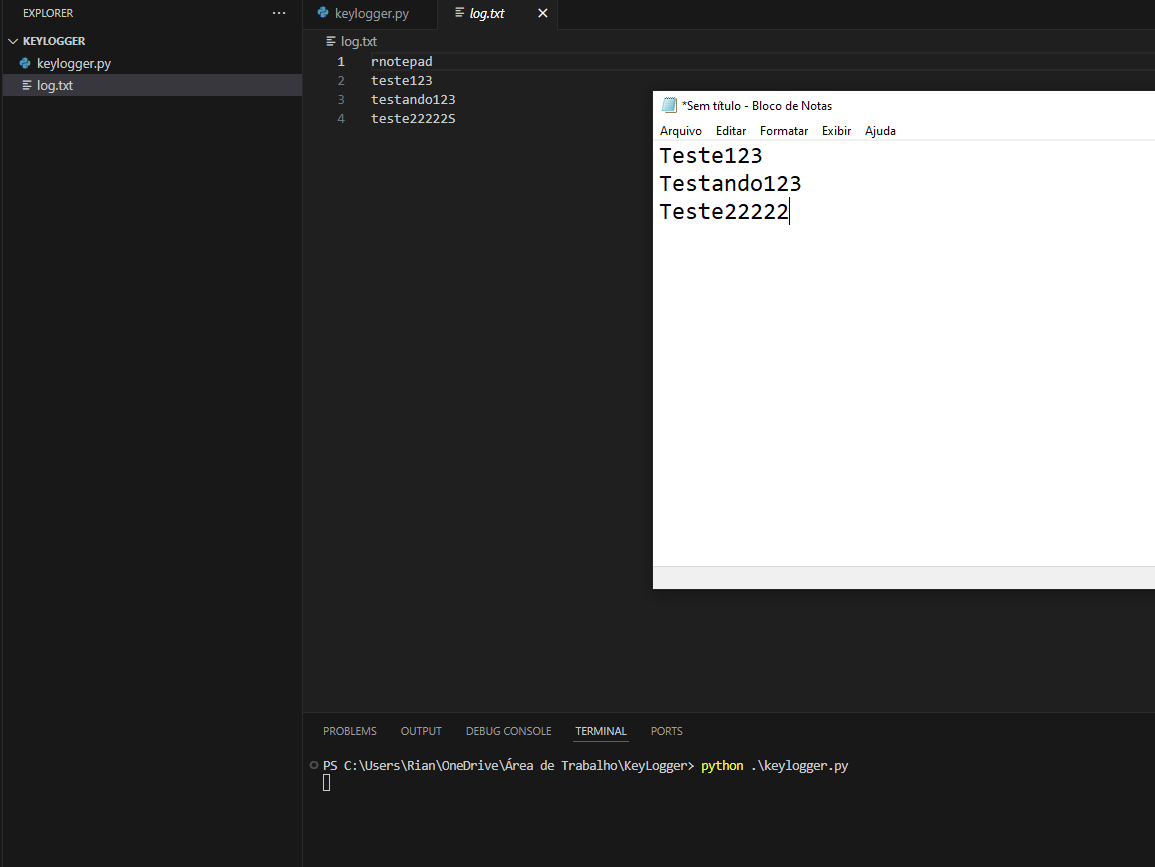
Carregar chave e reverter criptografia.

****

**Criação do Keylogger Básico:**

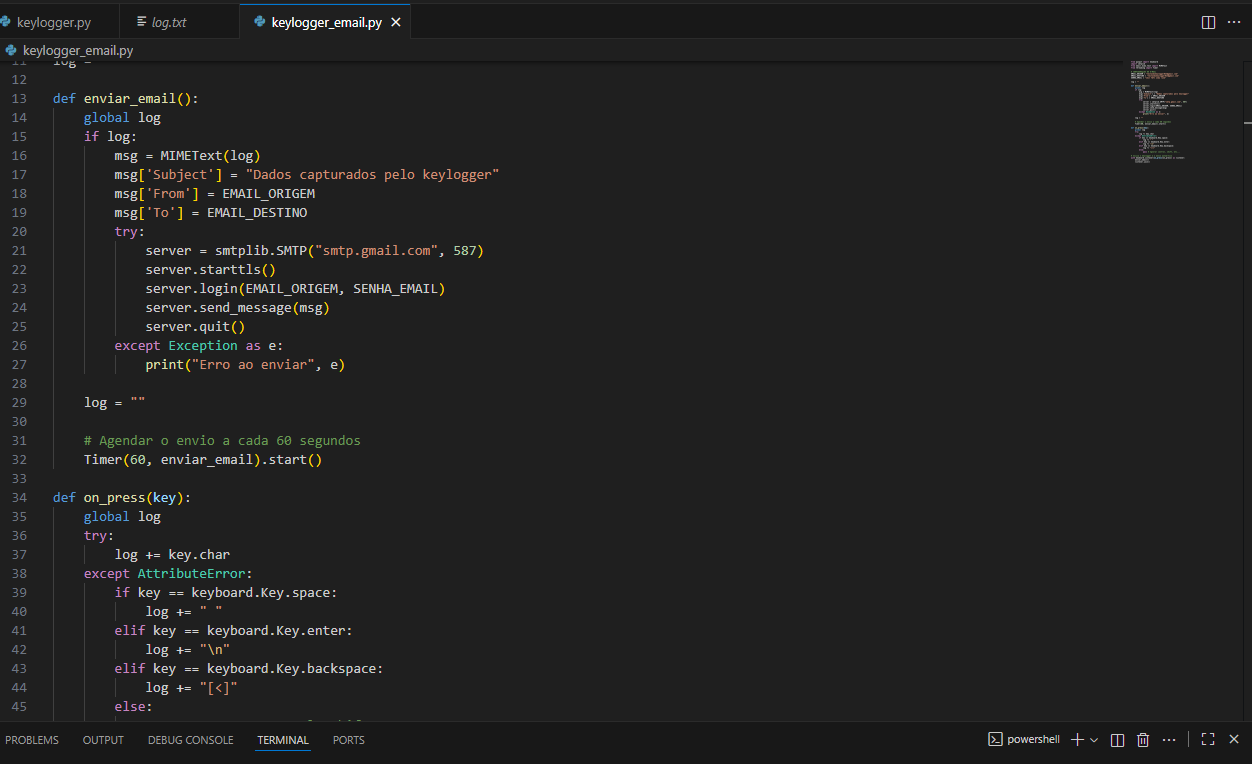
Capturar teclas e salvar em log.txt.

****

****

**Criação do Keylogger com Envio por E-mail:**

Captura periódica e envio automático.



**Execução:**

* python ransomware.py (criptografar).
* python descriptografar.py (descriptografar).
* python keylogger.py (captura local).
* python keylogger\_email.py (captura com envio).

# 6. Resultados Observados

Resumo dos achados:

- Ransomware: Arquivos em "Test\_files" foram criptografados com sucesso; chave gerada e mensagem "LEIA\_ISSO.txt" criada. Descriptografia restaurou os arquivos originais.

- Keylogger: Teclas capturadas e salvas em "log.txt" (ex: "Teste123\ntestando123\nteste222225").

- Keylogger com e-mail: Logs enviados periodicamente para o Gmail configurado, simulando exfiltração de dados.

# 7. Recomendações de Mitigação

* Medidas práticas para reduzir o risco de malwares como ransomware e keyloggers:
* Utilizar antivírus atualizados com detecção comportamental (ex: Windows Defender, Malwarebytes).
* Habilitar firewalls e sandboxing para isolar execuções suspeitas.
* Realizar backups regulares em mídias offline ou nuvem segura.
* Promover conscientização: Evitar abrir anexos desconhecidos, usar senhas fortes e MFA.
* Monitorar processos e logs do sistema para atividades anormais.
* Atualizar software e SO para corrigir vulnerabilidades conhecidas.
* Empregar ferramentas como EDR (Endpoint Detection and Response) em ambientes corporativos.

# 8. Conclusão

Este laboratório prático demonstrou de forma clara o funcionamento de malwares simulados em Python, como ransomware que sequestra dados via criptografia e keyloggers que capturam entradas sensíveis, inclusive com exfiltração por e-mail. Em um ambiente controlado, foi possível observar a simplicidade de implementação e o potencial dano, reforçando a exploração de brechas humanas e técnicas.

Os resultados destacam a necessidade de defesas proativas, como antivírus, backups e educação em cibersegurança. A simulação educacional evidencia que a proteção contra ameaças reais depende de práticas consistentes, como atualizações regulares e monitoramento, para mitigar riscos em cenários do mundo real.