Logotipo

Descrição gerada automaticamente com confiança baixa

Relatório de CTF

Hackfinity - Dark Encryptor – TryHackMe

|  |  |
| --- | --- |
| **Informações do documento** | |
| **Referência** | Hackfinity Dark Encryptor – Pedro Borges |
| **N° Revisão** | 1 |
| **Data de publicação** | 24/03/2025 |
| **Link** | https://tryhackme.com/room/HackfinityBattleEncore |

|  |  |  |
| --- | --- | --- |
| **Redação** | Pedro Borges | Estudante |
| **Revisão** | Nome do revisor | Orientador |
| **Aprovação** | Nome do aprovador | Diretor |

|  |  |  |
| --- | --- | --- |
| **Histórico de revisões** | | |
| **N°** | **Entregas** | **Descrição** |
| **0** | DD/MM/AAAA | Produção |
| **1** | DD/MM/AAAA | Revisão |
| **2** | DD/MM/AAAA | Aprovação |

|  |  |
| --- | --- |
| **Informações do CTF** | |
| **Nível de Dificuldade** | Fácil |
| **Tipo de acesso** | Gratuito |
| **Conceitos envolvidos** | RCE |
| **Plataforma** | Tryhackme, |
| **Área** | Red |

**Sumário**

[Contextualização 3](#_heading=h.gjdgxs)

[Subtítulo caso necessário 3](#_heading=h.30j0zll)

[Desenvolvimento 3](#_heading=h.1fob9te)

[Pergunta 1 3](#_heading=h.3znysh7)

[Pergunta 2 3](#_heading=h.2et92p0)

[Pergunta 3 3](#_heading=h.tyjcwt)

[Pergunta N 3](#_heading=h.3dy6vkm)

[Conclusão 3](#_heading=h.1t3h5sf)

[Referências 3](#_heading=h.4d34og8)

**Contextualização**

Void managed to hack into DarkMatter's internal network. I don't think they use it much, but we found this encryption tool hosted on a server. Let's see if we can find anything interesting lying around.

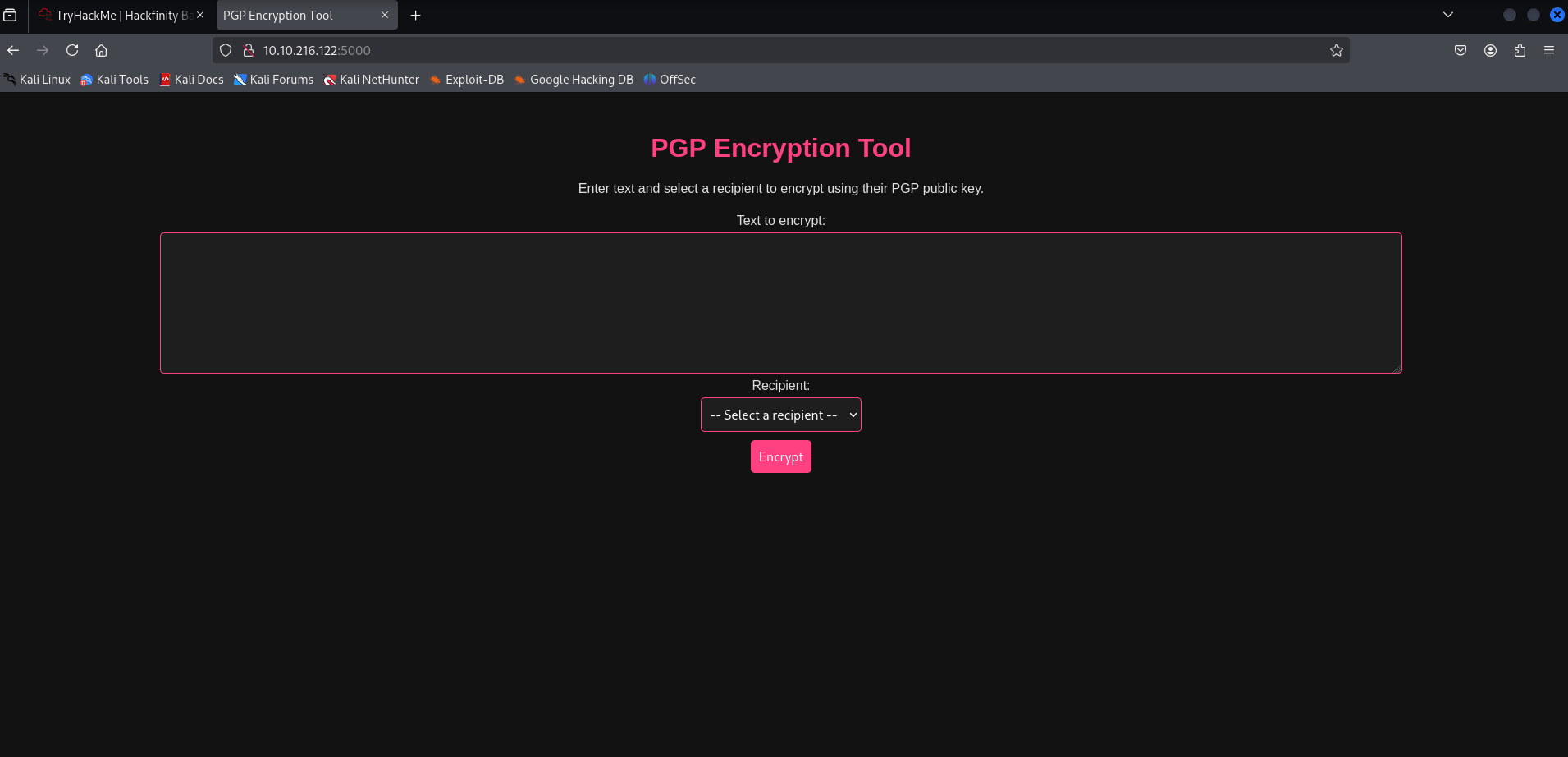
**Subtítulo caso necessário**

Espaço de tópico extra caso necessário

**Desenvolvimento**

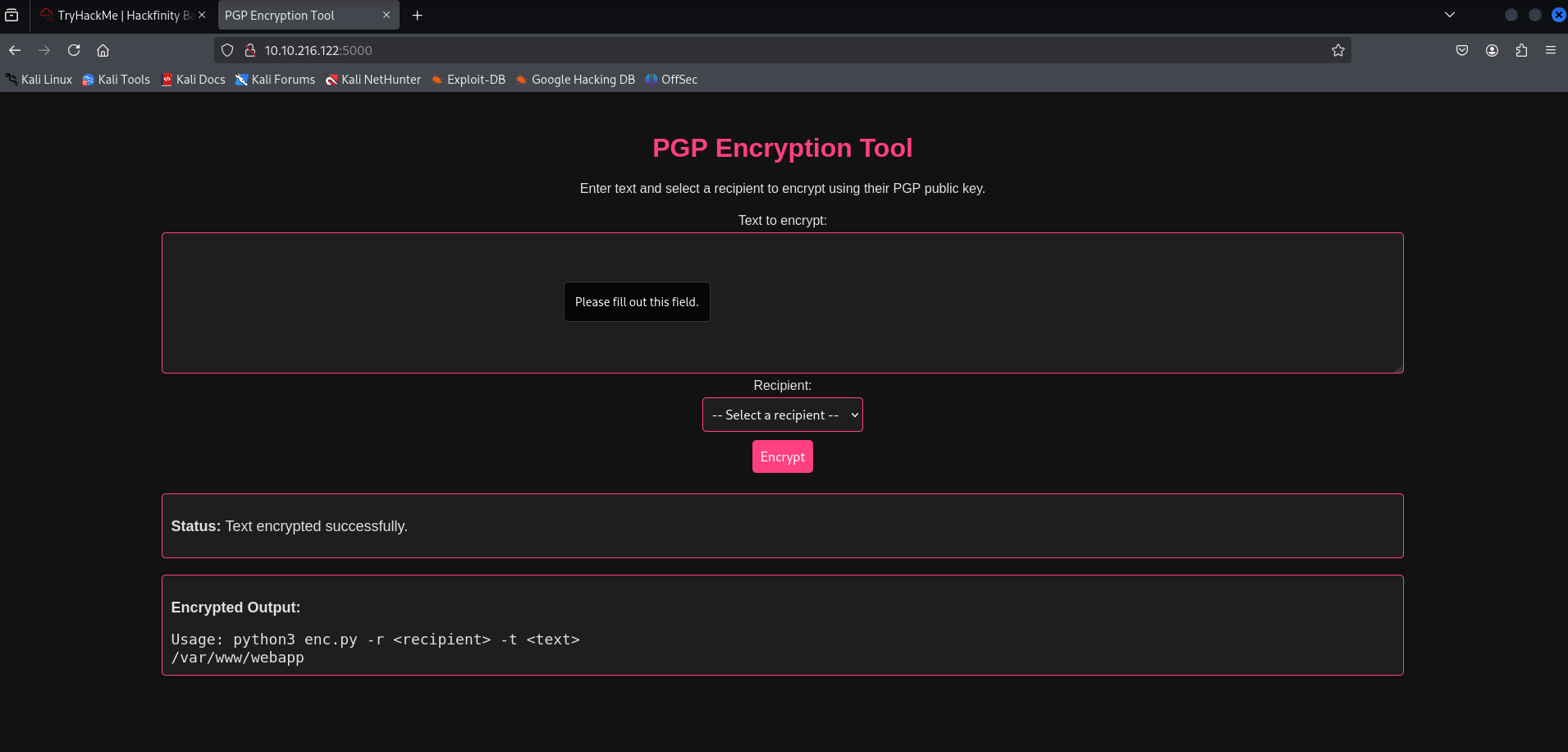
**What are the contents of flag.txt?**

Acessando o IP da máquina na porta fornecida (5000) caímos em uma webpage com um input (vide print abaixo)

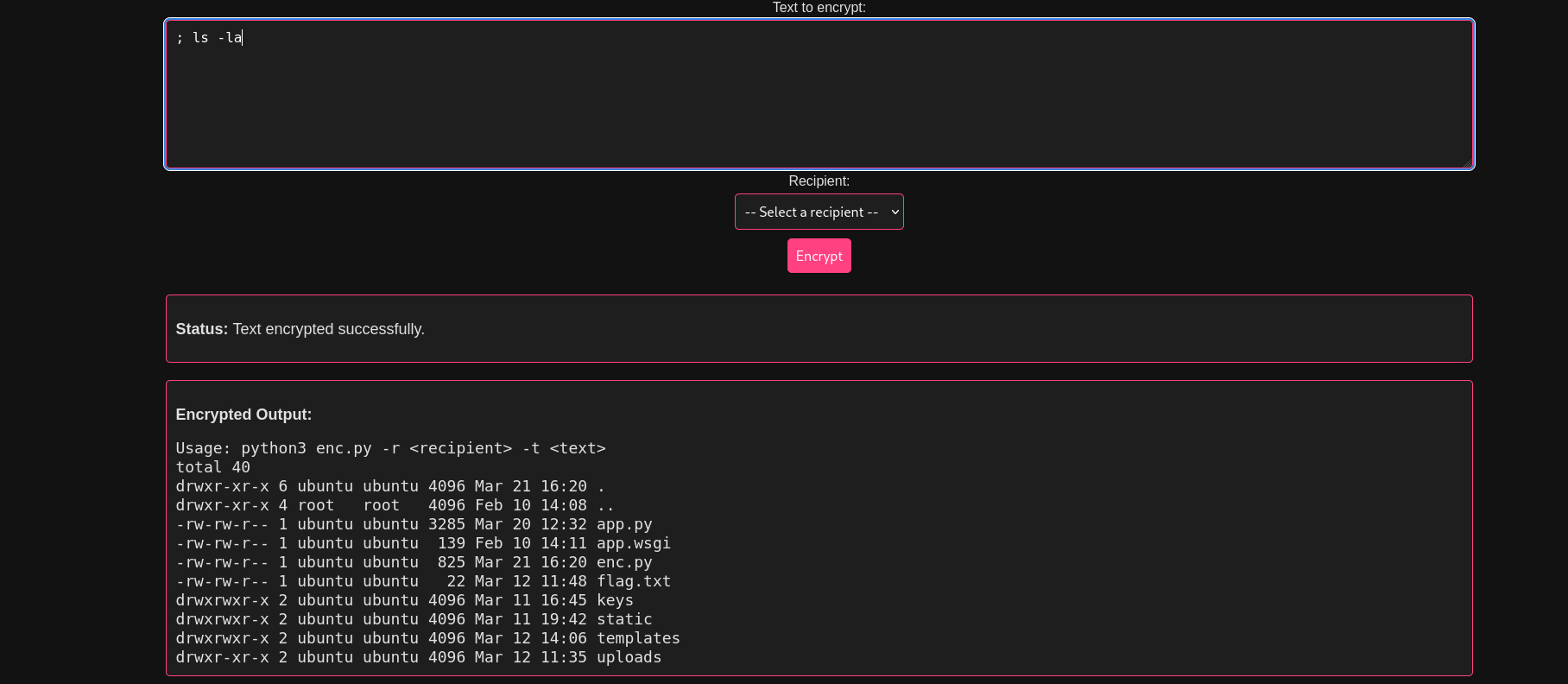


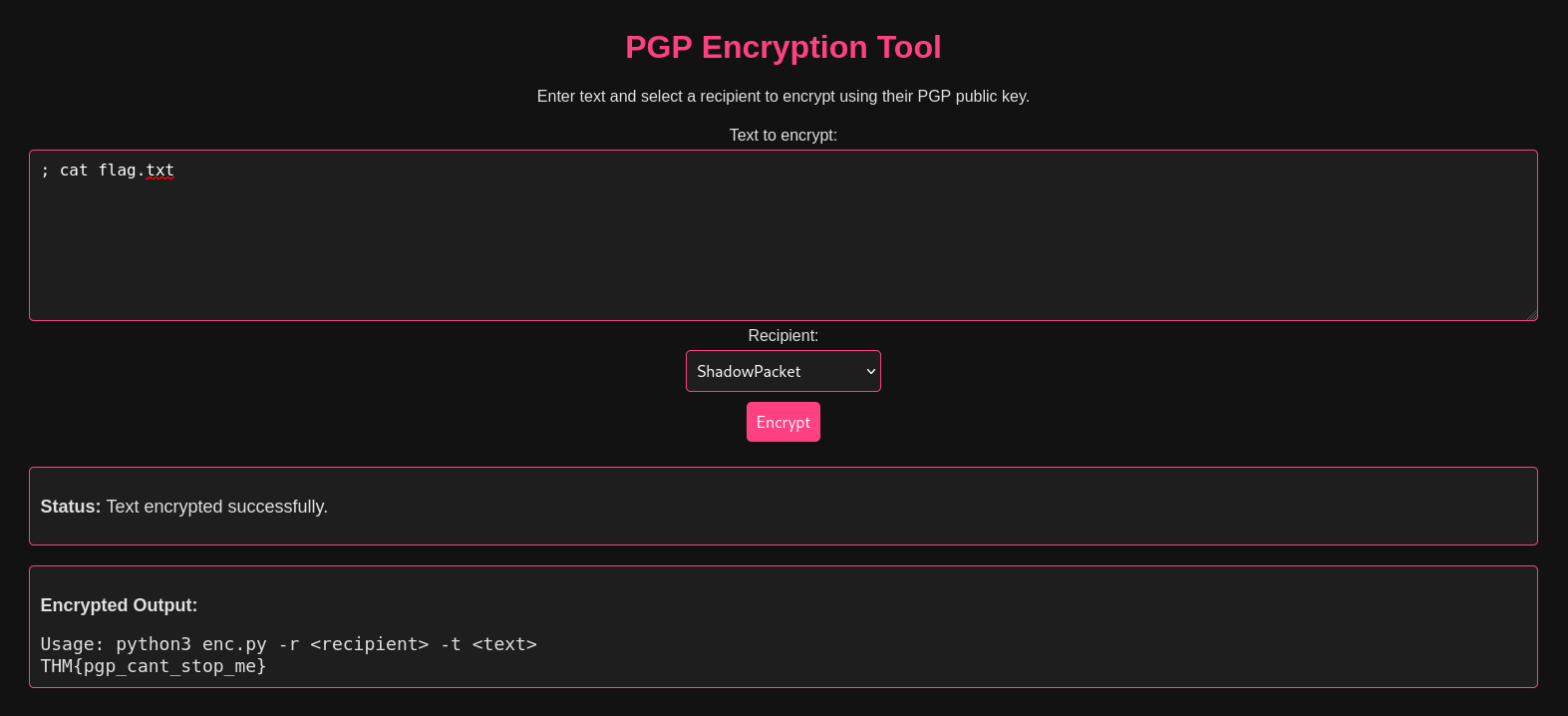
O sistema parece utilizar o utilitário gpg para criptografar mensagens com base nos parâmetros fornecidos pelo usuário. Como o GPG é normalmente executado via linha de comando no backend, uma implementação sem validação do frontend poderia concatenar os valores dos campos em um comando do shell.

Com essa hipótese em mente, utilizei a técnica de injeção de comandos, começando com uma abordagem simples: adicionei *; pwd* ao campo de input. O ponto e vírgula (;) no shell Unix separa múltiplos comandos, permitindo que o sistema execute um comando adicional após o original. Assim, se a aplicação estivesse vulnerável, o servidor executaria o comando gpg ... seguido de pwd, mostrando o caminho do diretório que eu estava.



Sabendo que existe essa vulnerabilidade, procedo listando os diretórios com *; ls –la*

A aplicação me retorna que existe um arquivo flag.txt, dando um cat nesse arquivo ele me retorna a flag do ctf.



**Conclusão**

Considerações finais a respeito do CTF.

**Referências**

Caminho dos materiais utilizados para a solução do CTF.