**Universidade Comunitária da Região de Chapecó**
**Curso:** Ciência da Computação
**Disciplina:** Tópicos em Redes de Computadores
**Professor:** Lucas Antunes da Rocha Volfe

# TRABALHO FINAL - EXPLORAÇÃO DE MÁQUINA NO TRYHACKME

Acadêmicos: Bernardo Zanetti, Gustavo Bones, João Vitor Lussi, Mariana Wagner, Murilo Debortoli, Naieli Loeblein e Pedro Knebel

Chapecó, 14 de dezembro de 2025

**PASSOS PARA RESOLUÇÃO DA BOX:**

**1º - Configuração do ambiente:**

Para a configuração do ambiente do TryHackMe, é necessário iniciar a VPN com o arquivo de configuração adquirido diretamente na plataforma. Utilizando o comando *sudo openvpn <caminho para o arquivo .ovpn>*, iniciamos a VPN e, posteriormente, a máquina do desafio.

**2º - Exploração e reconhecimento do ambiente:**

Primeiramente, para encontrarmos a primeira resposta, utilizamos o comando *nmap* para mapear todas as portas abertas no IP da máquina, por meio do comando *nmap -sV <ip da máquina>*. A flag *-sV* foi utilizada para descobrir o serviço em execução em cada porta e sua respectiva versão, respondendo assim às três primeiras perguntas.

Após isso, para encontrarmos os diretórios ocultos no IP da máquina, utilizamos o Gobuster, que realiza um de força bruta para descobrir diretórios com base em uma lista de palavras contida em um arquivo .txt. Foi utilizado o comando *gobuster dir -u http://<ip da máquina>/ -w <caminho para a lista de palavras>*. A lista de palavras foi obtida no repositório https://github.com/danielmiessler/SecLists.git. Após a execução do Gobuster, alguns diretórios foram descobertos.

**3º - Aplicando o Reverse Shell:**

Para aplicar o shell reverso, encontramos um repositório no GitHub que possui um exemplo de arquivo para shell reverso em .php. O PHP foi escolhido pois é compatível com servidores Apache. O repositório utilizado foi: https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php.

Após isso, alteramos o IP presente no script para o IP da máquina do TryHackMe e tentamos realizar o upload do arquivo. No entanto, o diretório */panel* bloqueava a importação de arquivos .php. Para contornar esse problema, alteramos a extensão do arquivo para .php5, o que possibilitou o upload. Para confirmar que a importação ocorreu com sucesso, retornamos ao diretório */upload,* onde foi possível verificar que o script havia sido executado.

Para obter acesso ao shell, utilizamos o Netcat, que é um listener de rede usado para ler e escrever dados por meio de conexões TCP e UDP, executando o comando *nc -lvpn 1234*. Em seguida, executamos o arquivo .php5 que foi importado na máquina virtual. Após isso, já estávamos dentro da máquina, o que pôde ser confirmado com o comando *whoami*. Para

encontrar a resposta desta etapa, executamos o comando find para localizar o arquivo user.txt e obter a chave correta. O comando completo utilizado foi *find / -name user.txt.*

**4º - Escalando privilégio:**

Para realizar o escalonamento de privilégios, utilizamos novamente o comando find, porém desta vez com as flags *-user root*, para localizar arquivos pertencentes ao usuário root, *-perm /4000*, para identificar arquivos que possuem o bit SUID ativado (ou seja, são executados com privilégios de root), e *2>/dev/null*, que evita a exibição de mensagens de erro relacionadas à falta de permissão de acesso. O comando completo utilizado foi find / -user root -perm /4000 2>/dev/null.

Após analisar o retorno do comando, identificamos a existência de um arquivo Python, que respondeu à primeira pergunta dessa etapa, pois poderia ser utilizado para o escalonamento de privilégios.

Em seguida, acessamos o site https://gtfobins.github.io/gtfobins/python/#suid, onde encontramos um comando que permite a execução de um shell com privilégios de root. O comando utilizado foi *usr/bin/./python2.7 -c 'import os; os.execl("/bin/sh", "sh", "-p")'*. Dessa forma, seguindo as orientações do site, foi possível abrir um shell com privilégios de administrador.

Confirmamos o acesso com o comando whoami, que retornou root, e ao executar o comando ls, localizamos o arquivo root.txt, que continha a última chave necessária para concluir a sala.

## AVALIAÇÃO DE RISCO E IMPACTO

**- Visão geral do risco:**

A cadeia de falhas identificada permite que um atacante evolua de acesso externo não autenticado para execução remota de código e, por fim, controle total do sistema (root). Isso caracteriza um cenário de alto risco, pois compromete completamente a confidencialidade, integridade e disponibilidade do ambiente.

## PRINCIPAIS ACHADOS E IMPACTO POTENCIAL

**Exposição de serviços/versões (Nmap -sV)**

- **Risco:** facilita mapeamento do alvo e escolha de exploits compatíveis com versões conhecidas.

- **Impacto:** aumenta a probabilidade de exploração bem-sucedida (fase de preparação do ataque).
- **Severidade:** Média (sozinha), mas amplifica o risco das demais falhas.

**Enumeração de diretórios revelando áreas sensíveis (Gobuster)**

- **Risco:** descoberta de endpoints funcionais como painéis e áreas de upload/admin.
- **Impacto:** permite encontrar superfícies de ataque "escondidas" e caminhos para exploração.
- **Severidade:** Média, com impacto alto quando combinado com upload inseguro.

**Upload inseguro resultando em execução de código (shell em PHP/PHP5)**

- **Risco:** possibilidade de RCE (Remote Code Execution) a partir de funcionalidade web.
- **Impacto:** O invasor pode executar comandos no servidor, acessar arquivos, capturar credenciais, pivotar para outros ativos da rede, manter persistência.
- **Severidade:** Alta/Crítica, por permitir comprometimento direto do host.

**Escalonamento de privilégio via SUID (python com bit SUID)**

- **Risco:** elevação de permissões de usuário comum para root.
- **Impacto:** controle total: leitura/alteração/exclusão de qualquer arquivo, instalação de backdoors, desativação de logs, criação de usuários, comprometimento permanente.
- **Severidade:** Crítica (pior cenário possível).

**Impacto por dimensão:**

- **Confidencialidade:** Crítica (acesso a user.txt, root.txt e potencialmente quaisquer dados do sistema).
- **Integridade:** Crítica (capacidade de alterar arquivos, binários, configs e aplicações).
- **Disponibilidade:** Alta/Crítica (possibilidade de derrubar serviços, apagar dados ou criptografar arquivos).

**Conclusão de risco:**

O risco global é Crítico, pois existe uma cadeia explorável que culmina em privilégio root, tornando o servidor completamente comprometido.

## RECOMENDAÇÕES DE CORREÇÃO/MITIGAÇÃO

### Mitigações prioritárias (alto impacto, rápida aplicação)

1. **Corrigir o upload de arquivos**
   - Bloquear upload de extensões executáveis e variações (.php, .php5, .phtml, etc.).
   - Validar tipo real do arquivo (MIME + assinatura) e não apenas a extensão.
   - Renomear arquivos no servidor (gerar nomes aleatórios) e impedir path traversal.
   - Aplicar tamanho máximo e quarentena/varredura (antimalware), se aplicável.

2. **Impedir execução no diretório de upload**
   - Configurar o servidor web para não executar scripts em /upload (apenas servir como arquivos estáticos).
   - Ajuste de permissões (owner/group), e política "sem execução" em diretórios de dados.

3. **Remover SUID indevido**
   - Remover o bit SUID de binários que não deveriam ter privilégio elevado (ex.: interpretadores como Python).
   - Revisar periodicamente: find / -perm /4000 -type f em auditorias internas.

### Melhorias estruturais

4. **Princípio do menor privilégio**
   - Garantir que serviços web rodem com usuário restrito.
   - Reduzir permissões de escrita do usuário do servidor, evitando upload em diretórios sensíveis.

5. **Hardening e atualização**
   - Atualizar pacotes/serviços expostos e reduzir banner/version disclosure quando possível.
   - Desabilitar módulos/funções desnecessárias no servidor (redução da superfície de ataque).

6. **Monitoramento e detecção**
    - Registrar tentativas de upload, erros 4xx/5xx, e atividades em endpoints sensíveis.
    - Alertas para padrões de enumeração (brute force de diretórios) e uploads suspeitos.

7. **Segmentação e contenção**
    - Isolar o servidor web de dados críticos e outros serviços internos.
    - Aplicar regras de firewall (permitir somente portas necessárias) e restringir acesso administrativo.

## LIÇÕES APRENDIDAS

- **Enumeração bem-feita "paga juros compostos":** o Nmap mostrou serviços e versões, e o Gobuster revelou diretórios relevantes. Sem essa base, a exploração seria tentativa e erro.
- **Funcionalidades "simples" viram porta de entrada:** um endpoint de upload mal configurado pode virar RCE com facilidade. Segurança precisa ser pensada no detalhe.
- **Defesa em profundidade é essencial:** mesmo que o upload falhe, um segundo controle (como impedir execução em /upload) poderia quebrar a cadeia de ataque.
- **SUID mal aplicado é um "atalho para o root":** binários com SUID devem ser extremamente raros e justificados. Interpretadores com SUID são especialmente perigosos.
- **Cadeias de vulnerabilidades são mais reais do que falhas isoladas:** cada etapa (descoberta → execução → privilégio) pode parecer "pequena", mas juntas resultam em comprometimento total.
- **Relatório técnico precisa conectar falha → exploração → impacto → mitigação:** não basta dizer "foi possível invadir"; é crucial mostrar por que é crítico e como corrigir de forma prática.