

## PRÁCTICA 4

### TAREA 1

Para limitar el número máximo de logins simultáneos de un usuario procederemos a modificar el fichero “/etc/security/limits.conf” que utiliza el módulo pam\_limits.

En el escribiremos la siguiente línea:

```
# End of file
user005          -          maxlogins          4
```

Y ahora si ejecutamos 5 conexiones SSH obtendremos el siguiente error en el último:

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
There were too many logins for 'user005'.
Last login: Tue Oct 19 22:35:19 2021 from ::1
Connection to localhost closed.
$
```

### TAREA 2

Primero descargamos la imagen y vemos cuanta CPU nos exige:

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
110631	root	20	0	2775164	285252	135544	S	59.7	2.1	0:12.11	firefox+

Una vez hecho esto ejecutamos el CPULIMIT con el proceso correspondiente y ya podemos ver como cambia el uso de CPU en el navegador.

root@fso-2021-22:~# fg	PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
cpulimit --pid 110631 -l 20	110631	root	20	0	2880740	300168	150672	S	16.9	2.2	0:47.54	firefox+
	1165	root	20	0	8572	4036	2016	S	7.0	0.0	1:06.17	dbus-da

## TAREA 3

Para crear un cgroup vamos al directorio `/sys/fs/cgroup` y creamos un directorio:

```
root@fso-2021-22:/sys/fs/cgroup# mkdir p4
```

## TAREA 4

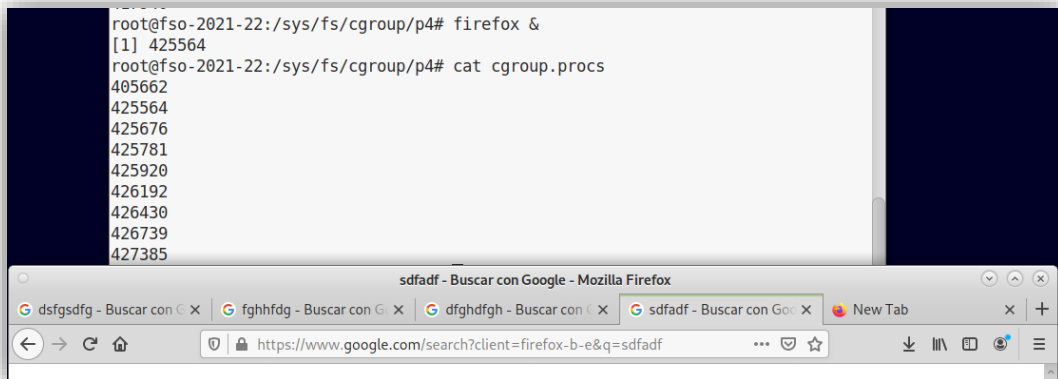
Accedemos al cgroup creado mediante el comando “`cd`” y localizamos el PID del terminal para añadirlo al cgroup:

```
root@fso-2021-22:/sys/fs/cgroup/p4# cat cgroup.procs
root@fso-2021-22:/sys/fs/cgroup/p4# ps -l
F S  UID      PID     PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S   0    405662   405641  0  80   0 -  2604 -      pts/0    00:00:00 bash
4 R   0    414920   405662  0  80   0 -  3207 -      pts/0    00:00:00 ps
root@fso-2021-22:/sys/fs/cgroup/p4# echo 405662 > cgroup.procs
root@fso-2021-22:/sys/fs/cgroup/p4# cat cgroup.procs
405662
```

## TAREA 5

Ejecutamos Firefox en ese mismo terminal y vemos lo que se añade en el fichero `cgroup.procs`.

```
root@fso-2021-22:/sys/fs/cgroup/p4# firefox &
[1] 425564
root@fso-2021-22:/sys/fs/cgroup/p4# cat cgroup.procs
405662
425564
425676
425781
425920
426192
426430
426739
427385
```



## TAREA 6

Vemos que algunos de los procesos que se han ejecutado anteriormente corresponden con el navegador abierto:

```
1 S   0    425920   425564  0  80   0 -  618218 -      pts/0    00:00:01 Web Conten
1 S   0    426192   425564  0  80   0 -  618651 -      pts/0    00:00:01 Web Conten
1 S   0    426430   425564  1  80   0 -  647610 -      pts/0    00:00:01 Web Conten
1 S   0    426739   425564  1  80   0 -  622853 -      pts/0    00:00:01 Web Conten
```

## TAREA 7

Para ver cuánta memoria están utilizando los procesos albergados en el cgroup:

```
root@fso-2021-22:/sys/fs/cgroup/p4# cat memory.current
534573056
```

Calculamos el tercio de la memoria que están utilizando los procesos y lo ponemos en el fichero de memory.high que logrará establecer un límite superior:

```
root@fso-2021-22:/sys/fs/cgroup/p4# echo 178191018 > memory.high
```

```
root@fso-2021-22:/sys/fs/cgroup/p4# cat memory.current
177098752
```

## TAREA 8

Ahora, con el límite de memoria establecido podemos observar como el navegador tiene un rendimiento pésimo en comparación con la primera vez sin límites.

## TAREA 9

Para congelar los procesos del cgroup bastan con ejecutar la siguiente sentencia modificando así el valor del archivo cgroup.freeze:

```
root@fso-2021-22:/sys/fs/cgroup/p4# echo 1 > cgroup.freeze
root@fso-2021-22:/sys/fs/cgroup/p4# echo 0 > cgroup.freeze
```

## TAREA 10

Para crear un contenedor de tipo debian basta con utilizar la herramienta LXC mediante el comando “lxc-create”

```
root@fso-2021-22:~# lxc-create -t debian -n containerP4
debootstrap is /usr/sbin/debootstrap
Checking cache download in /var/cache/lxc/debian/rootfs-stable-amd64 ...
Downloading debian minimal ...
I: Target architecture can be executed
I: Retrieving InRelease
I: Checking Release signature
I: Valid Release signature (key id A4285295FC7B1A81600062A9605C66F00D6C9793)
I: Retrieving Packages
I: Validating Packages
I: Resolving dependencies of required packages...
I: Resolving dependencies of base packages...
```

Para arrancar el contenedor basta con utilizar el comando "lxc-start" con los parámetros -F que indica que se inicia en primer plano y -n que permite indicar el nombre del contenedor que se desea iniciar.

```
Debian GNU/Linux 11 containerP4 console

containerP4 login: root
Password:

Login incorrect
containerP4 login: root
Password:
```

No podremos acceder debido a que no sabemos la contraseña del root, pero podemos eliminar la contraseña del mismo debido a que el fichero "/etc/shadow" del contenedor se guarda en la máquina host en la carpeta "/var/lib/lxc/containerP4/rootfs/etc/"

```
GNU nano 5.4 shadow
root::18919:0:99999:7:::
daemon:*:18919:0:00000:7:::
```

Ahora con el usuario root no nos pedirá la contraseña;

```
containerP4 login: root
Linux containerP4 5.10.0-8-amd64 #1 SMP Debian 5.10.46-4 (2021-08-03) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@containerP4:~#
```

Instalaremos el servidor web apache2:

```
root@containerP4:~# apt-get install apache2
Reading package lists... Done
Building dependency tree... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils bzip2 ca-certificates file libapr1
  libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap libbrotli1 libcurl4
  libexpat1 libgdbm-compat4 libgdbm6 libicu67 libjansson4 libldap-2.4-2
  libldap-common liblua5.3-0 libmagic-mgc libmagic1 libnghttp2-14 libperl5.32
  libpsl5 librtmp1 libsasl2-2 libsasl2-modules libsasl2-modules-db
  libsqlite3-0 libssh2-1 libxml2 mailcap media-types mime-support openssl perl
  perl-modules-5.32 publicsuffix ssl-cert xz-utils
```

Y SSH:

```
root@containerP4:~# apt-get install ssh
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  ssh
```

## TAREA 11

Creamos los usuarios en el contenedor:

```
root@containerP4:~# useradd -m usuario02
root@containerP4:~# useradd -m usuario01
root@containerP4:~# ls -l /home/
total 8
drwxr-xr-x 2 usuario01 usuario01 4096 Oct 19 21:53 usuario01
drwxr-xr-x 2 usuario02 usuario02 4096 Oct 19 21:53 usuario02
```

Bien, sabiendo que el sistema de ficheros es el mismo no cabría dudar de que los dueños de esos mismos ficheros en la máquina host tendrían otro dueño, pues los usuarios no se comparten ni nada por el estilo:

```
root@fso-2021-22:/var/lib/lxc/containerP4/rootfs# ls -l home/
total 8
drwxr-xr-x 2 user001 user001 4096 Oct 19 23:53 usuario01
drwxr-xr-x 2 user000 user000 4096 Oct 19 23:53 usuario02
root@fso-2021-22:/var/lib/lxc/containerP4/rootfs#
root@containerP4:~# ls -l /home/
total 8
drwxr-xr-x 2 usuario01 usuario01 4096 Oct 19 21:53 usuario01
drwxr-xr-x 2 usuario02 usuario02 4096 Oct 19 21:53 usuario02
```

Como podemos ver el usuario001 de la máquina host es el propietario del usuario 01 de la invitada. Esto es por que comparten el mismo identificador. El 1001:

```
root@fso-2021-22:/var/lib/lxc/containerP4/rootfs# id user001
uid=1001(user001) gid=1022(privilegiados) groups=1022(privilegiados)
root@fso-2021-22:/var/lib/lxc/containerP4/rootfs#
root@containerP4:~# id usuario01
uid=1001(usuario01) gid=1001(usuario01) groups=1001(usuario01)
root@containerP4:~#
```

Por último configuramos en el contenedor una password para usuario01 y para root de la siguiente manera utilizando “passwd”:

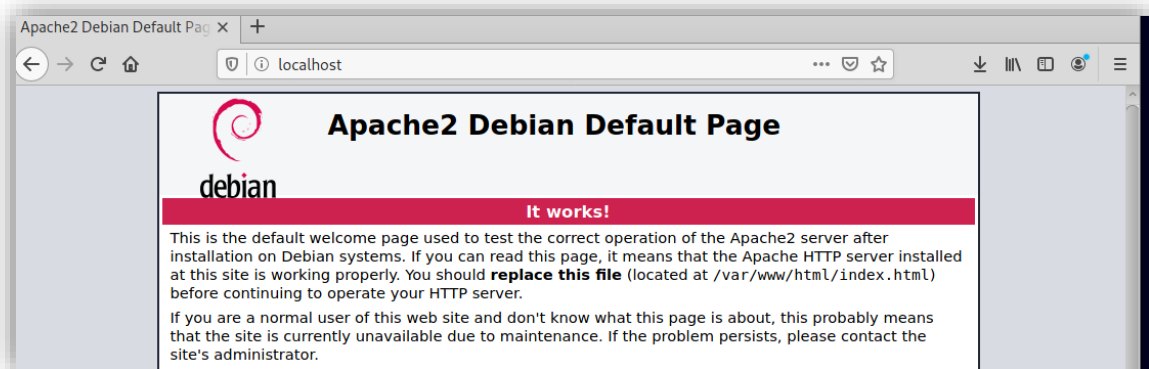
```
root@containerP4:~# passwd
New password:
Retype new password:
passwd: password updated successfully
```

Además localizamos la IP del contenedor que se obtiene mediante DHCP:

```
inet 10.0.3.138/24
```

## TAREA 12

Para arrancar el servidor apache de la máquina host basta con ejecutar “service apache2 start”. Una vez hecho esto tendremos acceso al servidor web de apache.



## TAREA 13

Lanzamos ahora el contenedor en segundo plano:

```
root@fso-2021-22:/var/lib/lxc/containerP4/rootfs# lxc-start -n containerP4
root@fso-2021-22:/var/lib/lxc/containerP4/rootfs#
```

## TAREA 14

Y accedemos al usuario root mediante SSH para modificar el contenido de la página web:

```
root@fso-2021-22:~# ssh usuario01@10.0.3.138
usuario01@10.0.3.138's password:
Linux containerP4 5.10.0-8-amd64 #1 SMP Debian 5.10.46-4 (2021-08-03) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
$ su
Password:
root@containerP4:/home/usuario01# cd
root@containerP4:~#
```

Modificamos a nuestro antojo la página web:

```
root@containerP4:~# echo "CONTENEDOR" > /var/www/html/index.html
```

## TAREA 15

Como podemos observar en la siguiente imagen, si accedemos al servidor web desplegado en el contenedor a través de su IP podemos observar que hemos cambiado el contenido del fichero y lo podemos observar desde la máquina host.

