

Actividad 28-03-25
José Camilo García Ponce

Primero creamos un nuevo grupo de control, esto lo hacemos creando un directorio nuevo en “/sys/fs/cgroup” con el comando “sudo mkdir /sys/fs/cgroup/mi_grupo”

```
patito@debian:~$ sudo mkdir /sys/fs/cgroup/mi_grupo
[sudo] contraseña para patito:
patito@debian:~$ ls /sys/fs/cgroup/
cgroup.controllers      cgroup.threads          init.scope              memory.reclaim          sys-kernel-debug.mount
cgroup.max.depth        cpu.pressure             io.cost.model           memory.stat             sys-kernel-tracing.mount
cgroup.max.descendants    cpuset.cpus.effective   io.cost.qos             mi_grupo               system.slice
cgroup.pressure          cpuset.mems.effective   io.pressure             misc.capacity          user.slice
cgroup.procs            cpu.stat                io.stat                proc-sys-fs-binfmt_misc.mount
cgroup.stat             dev-hugepages.mount     memory.numa_stat        sys-fs-fuse-connections.mount
cgroup.subtree_control  dev-mqueue.mount        memory.pressure         sys-kernel-config.mount
patito@debian:~$
```

Luego establecemos el límite de procesos del grupo con el comando “echo 100 | sudo tee /sys/fs/cgroup/mi_grupo/pids.max”

```
patito@debian:~$ echo 100 | sudo tee /sys/fs/cgroup/mi_grupo/pids.max
100
patito@debian:~$ cat /sys/fs/cgroup/mi_grupo/pids.max
100
```

Después metemos el proceso de nuestra terminal al grupo con el comando “echo \$\$ | sudo tee /sys/fs/cgroup/mi_grupo/cgroup.procs”

```
patito@debian:~$ echo $$ | sudo tee /sys/fs/cgroup/mi_grupo/cgroup.procs
1008
patito@debian:~$
```

Posteriormente creamos el script de la bomba fork y le damos permiso de ejecución

```
GNU nano 7.2                                bomba.sh *
#!/bin/bash
:(){ :|:& };:

patito@debian:~$ nano bomba.sh
patito@debian:~$ chmod +x bomba.sh
patito@debian:~$
```

Luego instalamos el comando “cgexec” mediante el comando “sudo apt install cgroup-tools -y”

```

patito@debian:~$ sudo apt install cgroup-tools -y
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  libcgrou2
Se instalarán los siguientes paquetes NUEVOS:
  cgroup-tools libcgrou2
0 actualizados, 2 nuevos se instalarán, 0 para eliminar y 28 no actualizados.
Se necesita descargar 128 kB de archivos.
Se utilizarán 480 kB de espacio de disco adicional después de esta operación.
Des:1 http://deb.debian.org/debian bookworm/main amd64 libcgrou2 amd64 2.0.2-2 [51.7 kB]
Des:2 http://deb.debian.org/debian bookworm/main amd64 cgroup-tools amd64 2.0.2-2 [76.3 kB]
Descargados 128 kB en 1s (170 kB/s)
Seleccionando el paquete libcgrou2:amd64 previamente no seleccionado.
(Leyendo la base de datos ... 31782 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../libcgrou2_2.0.2-2_amd64.deb ...
Desempaquetando libcgrou2:amd64 (2.0.2-2) ...
Seleccionando el paquete cgroup-tools previamente no seleccionado.
Preparando para desempaquetar .../cgroup-tools_2.0.2-2_amd64.deb ...
Desempaquetando cgroup-tools (2.0.2-2) ...
Configurando libcgrou2:amd64 (2.0.2-2) ...
Configurando cgroup-tools (2.0.2-2) ...
Procesando disparadores para libc-bin (2.36-9+deb12u9) ...
Procesando disparadores para man-db (2.11.2-2) ...
patito@debian:~$

```

Después ejecutamos la bomba fork con el comando “sudo cgexec -g pids:mi_grupo bash ./bomba.sh”

```

patito@debian:~$ sudo cgexec -g pids:mi_grupo bash ./bomba.sh
patito@debian:~$

```

```

patito@debian:~$ ps -aux | grep "bash ./bomba.sh"
root      2287  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh
root      2296  0.0  0.0   7064  1968 ?        S   14:43   0:00 bash ./bomba.sh
root      2349  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh
root      2361  0.0  0.0   7064  1972 ?        S   14:43   0:00 bash ./bomba.sh
root      2379  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh
root      2392  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh
root      2393  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh
root      2401  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh
root      2408  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh
root      2413  0.0  0.0   7064  1968 ?        S   14:43   0:00 bash ./bomba.sh
root      2423  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh
root      2430  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh
root      2432  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh
root      2434  0.0  0.0   7064  1964 ?        S   14:43   0:00 bash ./bomba.sh

```

```

patito@debian:~$ ps -aux | grep "bash ./bomba.sh" | wc -l
100
patito@debian:~$

```

Posteriormente matamos los procesos que están ejecutando la bomba fork con el comando “sudo kill -9 \$(ps aux | grep "bash ./bomba.sh" | grep -v grep | awk '{print \$2}')

```

patito@debian:~$ sudo kill -9 $(ps aux | grep "bash ./bomba.sh" | grep -v grep | awk '{print $2}')
patito@debian:~$

```

Y por último revisamos que todo esté bien, todo funcionó bien ya que el único proceso que queda no crea ningunos más

```
patito@debian:~$ ps -aux | grep "bash ./bomba.sh"
patito      25683  0.0  0.1   6492  2148 pts/1    S+   14:54   0:00 grep bash ./bomba.sh
patito@debian:~$ ps -aux | grep "bash ./bomba.sh" | wc -l
1
patito@debian:~$
```

Todo funcionó bien ya que no tuvimos problemas de que la computadora se alentara o no funcionara, entonces la creación del grupo de control y la ejecución tuvo éxito