UNIVERSITAT DE BARCELONA

FACULTAT DE MATEMÀTIQUES I INFORMÀTICA

DEPARTAMENT D'ENGINYERIA INFORMÀTICA

Pràctica 1: Chroot jail

Autors:
Carla Morral
Martí Pedemonte

Professor:
Jordi José Bazán

Grup A - Parella 01

Sistemes Operatius II

Octubre de 2020

Índex

1 Introducció

Aquesta primera pràctica consisteix a crear una gàbia, de manera que les comandes que s'hi executin tinguin accés restringit a fitxers i directoris. L'objectiu és entendre la utilitat i el seu funcionament, així també com la seva construcció. Això ho hem fet amb la comanda de bash `chroot`, passant com a primer argument el directori que farà d'arrel de la gàbia, que en el nostre cas es diu *gabia*, i com a segon paràmetre la comanda a executar. Primer ho hem fet amb l'executable `statistics` provinent del codi proporcionat `statistics.c`, i després hem estès la gàbia a poder realitzar comandes més complexes, com són `bash`, `ls`, `cp` i `rm`.

2 Qüestions

En aquest apartat resoldrem les qüestions plantejades al llarg de la pràctica.

1. **Pot l'aplicació llegir el fitxer que hi ha a l'interior del directori "data"? En cas que no pugui, per què no pot?**
   L'aplicació no pot llegir el fitxer a l'interior del directori "`data`", i retorna l'error `Could not open file 'file.txt'`. Això és degut al fet que, en el codi original, la variable FILE té el valor "`file.txt`", és a dir, suposa que el fitxer de dades és al mateix directori en què s'executa el programa. En canvi això no és cert, ja que el fitxer es troba al directori "`data/file.txt`".

2. **Quin valor ha de tenir la variable FILE al codi C perquè es llegeixi correctament el fitxer? Ho aconseguiu fer posant una ruta completa al fitxer?**
   Per a solucionar l'error obtingut a la qüestió anterior el que hem de fer és, en efecte, canviar la ruta del fitxer. És a dir, si canviem el valor de la variable FILE de "`file.txt`" a "`data/file.txt`" la comanda s'executa correctament, com es pot veure a continuació:

```
$ sudo chroot gabia /bin/statistics
Summary:
 Vowels: 1345529
 Consonants: 2047454
 Digits: 1214
 Space chars: 1060666
 Puntuacion chars: 196974
```

3 Construcció de la gàbia i proves realitzades

En aquesta secció s'exposen en detall els passos que hem seguit per a construir la gàbia demanada, així com diverses proves que hem cregut adients per a assegurar el bon funcionament d'aquesta.

3.1 Passos duts a derme

En primer lloc el que hem de fer per poder executar una comanda és copiar-la del directori `/bin`. Per a fer-ho hem utilitzat sempre la versatilitat del terminal, executant la comanda `cp`. En el cas de la comanda `bash`, la instrucció completa és la següent:

```
$ cp /bin/bash gabia/bin
```

D'aquesta manera ja tenim l'executable desitjat dins la gàbia. No obstant això, si intentéssim executar la gàbia (`sudo chroot gabia /bin/bash`) obtindríem el següent missatge d'error:

```
chroot: failed to run command ‘/bin/bash’: No such file or directory
```

Això és perquè no hem carregat les llibreries dinàmiques necessàries per a poder executar la instrucció. Per a solucionar aquest problema, tal com ens diu el guió, hem copiat les llibreries dins la gàbia. Per a saber quines ens fan falta només cal executar el següent:

```
$ ldd gabia/bin/bash
linux-vdso.so.1 (0x00007ffc9c379000)
libtinfo.so.6 => /lib/x86_64-linux-gnu/libtinfo.so.6 (0x00007f775aa2b000)
libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007f775aa25000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f775a833000)
/lib64/ld-linux-x86-64.so.2 (0x00007f775ab98000)
```

Ara, simplement hem de copiar-les dins la gàbia:

```
$ cp /lib/x86_64-linux-gnu/libtinfo.so.6 gabia/lib/x86_64-linux-gnu/
$ cp /lib/x86_64-linux-gnu/libdl.so.2 gabia/lib/x86_64-linux-gnu/
$ cp /lib/x86_64-linux-gnu/libc.so.6 gabia/lib/x86_64-linux-gnu/
$ cp /lib64/ld-linux-x86-64.so.2 gabia/lib64/
```

Si ara provem d'executar la gàbia amb la comanda `bash` veiem que ja funciona correctament:

```
$ sudo chroot gabia /bin/bash
bash-5.0#
```

Per a construir la gàbia per a la resta de comandes el procediment és anàleg: primer copiem la comanda de `/bin`, mirem quines llibreries dinàmiques es necessiten en cada cas (`ldd`) i després les copiem on toca.

3.2 Proves i comprovacions realitzades

Ja hem vist a la segona qüestió que l'executable `statistics` funciona correctament quan l'executem dins la gàbia. És per això que mostrarem només les proves que hem fet amb les altres comandes (`bash`, `ls`, `cp` i `rm`).

Proves amb la comanda `bash`: Per a testejar aquesta comanda hem mirat si podíem fer servir el terminal bash-5.0. Hem mirat quins directoris i fitxers hi ha a la gàbia, i hem comprovat que no podem accedir fora d'ella:

```
$ sudo chroot gabia /bin/bash
bash-5.0# ls
bin  data  lib lib64
bash-5.0# cd data
bash-5.0# ls
file.txt
bash-5.0# cd ..
bash-5.0# ls
bin  data  lib lib64
bash-5.0# cd ..
bash-5.0# ls
bin  data  lib lib64
bash-5.0# exit
exit
```

Proves amb la comanda `ls`: Per a testejar aquesta comanda hem mirat si podíem veure fitxers i directoris dins la gàbia, i hem comprovat que no podem accedir fora d'ella:

```
$ sudo chroot gabia /bin/ls
bin  data  lib lib64
$ sudo chroot gabia /bin/ls data
file.txt
$ sudo chroot gabia /bin/ls -l
total 0
drwxrwx--- 1 0 998 0 Oct  8 14:39 bin
drwxrwx--- 1 0 998 0 Oct  5 10:46 data
drwxrwx--- 1 0 998 0 Oct  5 17:20 lib
drwxrwx--- 1 0 998 0 Oct  8 14:54 lib64
$ sudo chroot gabia /bin/ls ..
bin  data  lib lib64
```

Proves amb la comanda `cp`: Per a testejar aquesta comanda hem mirat si podíem copiar fitxers dins la gàbia:

```
$ ls gabia/
bin  data  lib  lib64
$ sudo chroot gabia /bin/cp data/file.txt .
$ ls gabia/
bin  data  file.txt  lib  lib64
```

Proves amb la comanda `rm`: Per a testejar aquesta comanda hem mirat si podíem eliminar fitxers dins la gàbia. Eliminem el fitxer que havíem copiat a l'anterior prova:

```
$ ls gabia/
bin  data  file.txt  lib  lib64
$ sudo chroot gabia /bin/rm file.txt
$ ls gabia/
bin  data  lib  lib64
```

Com hem pogut comprovar, totes les comandes funcionen de la manera esperada, sempre restringint totes les accions a dins la gàbia.

4 Conclusions

Durant la realització d'aquesta pràctica hem pogut veure que l'ús de la comanda `chroot` és molt útil, ja que el fet de poder canviar el directori arrel implica que podem aïllar les dependències de qualsevol projecte en què vulguem treballar i així poder testejar-lo de manera més segura i fiable. En resum, ens ha semblat una eina molt interessant.