Cognoms: _____ Nom: _____ DNI: _____

CONCEPTES AVANÇATS DE SISTEMES OPERATIUS (CASO)

Facultat d'Informàtica de Barcelona, Dept. d'Arquitectura de Computadors, curs 2019/2020 – 2Q

Control de Laboratori 27/5/20

L'examen és individual
Responeu en l'espai assignat
Indiqueu COGNOMS, NOM i DNI (per aquest ordre)
Podeu consultar llibres, apunts i Internet.
No podeu parlar amb ningú dels temes d'aquest control mentre duri el control
Justifiqueu totes les respostes
Temps: 1 hora 30'

1. Introducció

En aquest control de laboratori farem dues pràctiques sobre la gestió que fa Linux de les taules de descriptors de fitxers i de la gestió de memòria. També es veuran aspectes d'avaluació del rendiment. Comencem:

- Desempaqueta el fitxer de suport **ctrl-lab-1920.tar.bz2** que hauràs obtingut juntament amb aquest enunciat.
- **Entra en el directori CASO1920_xx**. (xx serà el grup on fas el control)

2. Taules de canals (4 punts)

Analitzarem la informació que podem obtenir del sistema usant el shellscript fd-top.sh, proporcionat en el paquet de software.

Ja teniu obert un terminal a la màquina en la que esteu fent el control (aquest terminal l'anomenarem "terminal 1").

* En el "terminal 1" executeu l'script fd-top.sh, demant-li l'ajuda

```
$ ./fd-top.sh -h
```

i estudieu la sortida per saber les opcions que pot rebre l'script i quines tecles poden canviar el seu comportament en runtime. Feu proves de la seva execució amb diferents paràmetres.

Mireu també el fitxer README amb les instruccions sobre el seu funcionament, que trobareu acompanyant-lo en el mateix directori.

Feu diverses proves més d'execució amb diferents paràmetres per entendre millor com funciona i què fa.

* Obriu un altre terminal on executarem les comandes que volem examinar i determineu quin és. L'anomenarem "terminal en observació".

* En el “terminal 1”, executeu l'fd-top.sh , indicant-li que observi només els processos del "terminal en observació".

* Compileu el fitxer copiar.cpp amb

```
$ make
```

En el "terminal en observació" executeu un

```
$ ./copiar /dev/tty fitxer.out
```

2a) Expliqueu detalladament quina informació mostra l'script fd-top.sh de cada procés que està executant-se en el "terminal en observació. Haurien de ser 2 processos com a molt, un dels quals el “copiar”. Quin és l'altre?

En acabar aquesta pregunta, podeu interrompre l'execució de ./copiar amb ctrl-c (^C).

No useu ctrl-z (^Z) perquè es quedarà encara aturat i el veurem associat al terminal en observació.

2b) Ara executeu aquesta comanda en el "terminal en observació"

```
$ dd if=/dev/tty of=fitxer.out
```

i explica les diferències que veus amb el fd-top.sh entre el que feia la comanda "copiar" per llegir del fitxer "/dev/tty" i com ho fa la comanda "dd":

2c) Executa el següent pipeline de comandes en el "terminal en observació" i mentre estigui en execució prem un ctrl-z per aturar-lo (no fer-lo acabar, ara sí que el volem deixar parat):

```
$ od -x /dev/tty | wc -l |more
```

Explica la sortida que veus del fd-top.sh i com pots determinar com flueix la informació per les pipes.

2d) Mireu l'script fd-top.sh, i indiqueu de quins fitxers es serveix per obtenir la informació que ens dóna. Què és el directori "/proc" i per a què serveix?

3. "Locked memory" (3 punts)

Els processos poden indicar al sistema que volen disposar d'una certa quantitat de dades que siguin sempre presents a memòria. És el que anomenem "locked memory".

3a) Podeu determinar quin límit tenen habitualment els processos d'usuaris no privilegiats respecte a la quantitat de memòria que poden usar "locked"? Expliqueu com ho feu:

3b) Trobareu el programa pipes-bandwidth en el directori de treball, compileu-lo i executeu-lo. Mireu-lo i expliqueu breument què fa. Pista: observeu que està relativament ben comentat:

Apunteu aquí el rendiment (bandwidth en Mbytes/s) que obté a la vostra màquina:

3c) Volem que els buffers que usa el programa pipes-bandwidth estiguin "locked" a memòria. Com podeu aconseguir-ho? Feu-ho:

Pista: dins el fitxer pipes-bandwidth.cpp trobareu les línies:

```
// mlock the buffers, checking for errors appropriately    ←Només podeu afegir codi a partir d'aquí

// end of your code   ← No hauríeu de canviar codi més avall d'aquesta línia
```

3d) Un cop canviat el programa, haureu de fer un canvi perquè funcioni correctament. Quin canvi suggeriu? Pista: relacionat amb la mida del buffer (BUFFER_SIZE)?

3e) Finalment, executeu-lo un cop els dos buffers (buffer i recvbuf) estiguin “mlocked”, i apunteu aquí el rendiment que obté:

Apunteu aquí el nou rendiment (bandwidth en Mbytes/s) que obté a la vostra màquina:

Quina conclusió en traieu? Ha millorat? (Ja entenem que si l’entorn on esteu executant és en màquina virtual o uniprocessador, serà difícil que hagi millorat):

4. Relació entre els processos i els fitxers (3 punts)

La comanda “lsof” permet veure quins processos tenen obert un determinat fitxer (per lectura/escriptura o update) i també per veure quins processos tenen un directori com a “current working directory”.

4a) En el "terminal 1", useu la comanda lsof per obtenir informació dels processos que hi ha al directori actual:

```
$ lsof  “.”
```

Expliqueu cada camp de la informació que us dóna:

4b)

En el "terminal en observació" executeu:

```
$ cat >fitxer
```

i en el "terminal 1" feu un

```
$ lsof fitxer
```

I també:

"terminal en observació: cat 1<>fitxer

"terminal 1" : lsof fitxer

I indiqueu la diferència principal entre el que fa la comanda

```
$ cat >fitxer
```

i el que fa la comanda

```
$ cat 1<>fitxer
```

Pista: man bash i busqueu <> a la pàgina de manual

Fi del control de lab.