ESERCIZIO 1

Un file regolare di nome "pippo" è eseguibile da tutti, leggibile ed eseguibile dal gruppo, leggibile, modificabile ed eseguibile dal proprietario.

a) riportare la notazione Unix che rappresenta correttamente i permessi e le protezioni associate a pippo

b) indicate il comando necessario per togliere l'eseguibilità di pippo per il gruppo e tutti gli altri utenti del sistema (l'eseguibilità deve rimanere per il proprietario del file)

1

ESERCIZIO 1 (cont.)

c) indicate il comando necessario per permettere al gruppo di modificare il file

d) riportare la notazione Unix che rappresenta correttamente i permessi e le protezioni associate a pippo dopo le operazioni dei punti b) e c)

e) qual è il comando Unix normalmente usato per visualizzare i permessi e le protezioni associate ad un file?

2

ESERCIZIO 2

Nel seguito, assumiamo che tutti i comandi indicati vengano correttamente eseguiti senza errore

a) All'interno di una directory vuota (di nome dir1), viene dato il comando "mkdir new-folder". Disegnare la struttura interna nella cartella dir1 dopo questa operazione. (scegliete voi dei valori adeguati per gli i-node)

3

ESERCIZIO 2 (cont.)

b) Sempre all'interno di dir1, viene data la seguente sequenza di comandi:

```
cp ../pippo .
ln pippo pluto
rm pippo
```

come è fatto il file "dir1" a questo punto?

4

ESERCIZIO 3

a) Qual è l'effetto del seguente comando Unix?

ps agx | grep gcc | wc -l > xyz

b) In questo contesto, qual è la funzione dei "caratteri" "|" e ">"?

5

ESERCIZIO 4

a) qual è la differenza tra i seguenti due comandi?

1) gcc myfile.c
2) gcc myfile.c &

b) come si può "trasformare" il comando 1), mentre questo è in esecuzione, nel comando 2)?

6

ESERCIZIO 5

Scrivete uno script shell che ha tre argomenti di input. I primi due sono file che esistono. Il terzo argomento è il nome di un file che deve essere creato e deve essere formato dalle prime righe del primo file (ad esempio le prime 10) e dalle ultime righe del secondo file (ad esempio le ultime 10). Non devono essere usati file temporanei.

7

ESERCIZIO 6

si consideri il seguente programma C:

```
1 main()
2  {
3  int n, i;
4  i = 5;
5  n = fork();
6  if ( n != 0) { i = i+1;
7                 printf("%d",i);}
8  else printf("%d",i);
9  }
```

a) qual è il valore della variabile i stampato alle righe 7 e 8? Perché?

8

ESERCIZIO 6 (cont.)

b) che cosa fa la system call fork? In che senso possiamo dire che "restituisce due valori"? Che valori?

c) nel codice dato, possiamo dire in che ordine verranno eseguite le stampe delle righe 7 e 8?

9

ESERCIZIO 7

si considerino i seguenti programmi C:

P1:

```
void pippo()
{printf("ciao!");
  exit()}

1 main()
2  {
3  int i,n;
4  for (i=0; i<5; i++) sleep(1);
5  n = signal(SIGUSR1,pippo);
6  printf("bye bye");
7 for (;;) sleep(1);}
```

P2:

```
1 main(int argc, char *argv[])
2  {
3  if (argc > 1) kill(PID,SIGUSR1)
4  else kill(PID,SIGKILL)
5  }
```

dove PID è il process-id di P1 (si assuma che sia noto a P2 quando questo viene eseguito)

a) a cosa serve la syscall signal?

10

ESERCIZIO 7 (cont.)

b) a cosa serve la syscall kill?

c) A quali condizioni P1 stampa la scritta "ciao!"?

d) A quali condizioni P1 stampa la scritta "bye bye"?

e) A quali condizioni P1 stampa sia "ciao!" che "bye bye"?

11

ESERCIZIO 8

una cartella "mydir" contiene come unico file l'eseguibile "myproc", di cui questo è il corrispondente sorgente:

myproc.c:

```
1 main()
2  {
3  printf("ciao\n");
4  execl("/bin/ls","/bin/ls",0);
5  printf("bye bye\n");
6  }
```

a) qual è l'output prodotto dall'esecuzione di myproc nella cartella mydir se la execl viene correttamente eseguita?

12

ESERCIZIO 8 (cont.)

myproc.c:

```
1 main()
2  {
3  printf("ciao\n");
4  execl("/bin/ls","/bin/ls",0);
5  printf("bye bye\n");
6  }
```

b) e se la execl per qualche ragione fallisce (ad esempio ls non è presente nella directory /bin)?

c) cosa fa la syscall execl?

13

ESERCIZIO 9

a) Spiegate in che contesto vengono usati e quali sono i vantaggi e gli svantaggi nell'uso dei blocchi marcati "delayed write"

b) Poichè il buffer cache ruba spazio in ram, potrebbe essere conveniente implementarlo usando la memoria virtuale?

14

ESERCIZIO 10

a) Che cosa indica il link counter di un i-node? Se leggiamo il link counter di un qualsiasi i-node di un file system, qual è il valore più piccolo che possiamo trovare? E il valore più grande?

b) Viene dato il comando "rm A". In quale caso i blocchi dei dati del file A vengono efffettivamente rimossi insieme all-i-node di A? (supponendo ovviamente che si abbiano i permessi per rimuovere A)

15

ESERCIZIO 11

a) In un sistema sappiamo che alcuni utenti stanno attualmente lanciando il compilatore gcc. Cos'è che ci dice quante istanze del compilatore sono attive in un dato istante? Quante copie dell'eseguibile gcc saranno presenti in RAM? Perché?

b) In quale caso il codice di gcc non viene rimosso dalla RAM anche se nessuno sta compilando?

16

ESERCIZIO 12

a) Nell-i-node di un file A il link-counter vale 3. Viene dato il comando: ln -s A B (B diviene un link simbolico ad A). Che valore assume il link-counter del file?

b) Viene dato il comando "rm A". Cosa succede di B?

17

ESERCIZIO 13

In un sistema Unix i blocchi dell'hard disk sono grandi 512 byte, e vengono usati due byte per scrivere il numero di un blocco. Un file A è lungo 5140 byte.

a) La lettura del byte 1000 richiede più, meno, o lo stesso tempo della lettura del byte 5135 di A (motivate la risposta, assumendo che non sia presente il buffer cache).

18

ESERCIZIO 13(cont.)

b) Qual è la frammentazione interna prodotta da A?

c) Quanti blocchi diversi possono essere indirizzati usando 2 byte?

d) Qual è la massima dimensione di un file in questo sistema?

19

IN DEFINITIVA:

a) se avete fatto e capito gli esercizi sui comandi unix, e se vi siete abituati ad usarli per lavorare sul sistema quando sviluppate i programmi in C,

b) Se avete capito come funzionano le principali system call (fork, wait, signal, kill, execl) e avete fatto gli esercizi che abbiamo dato durante il corso,

non avrete problemi a risolvere gli esercizi dell'esame.

ATTENZIONE: non ci saranno domande/esercizi sull'IPC, ma ci potranno essere domande sulle cose viste in aula sul kernel Unix, ad esempio: come funziona un file di tipo pipe, che cosa è una user file table, cosa è il set-uid bit, quali regioni di un processo possono essere condivise, come viene calcolata la priorità di un processo unix, e così via.

20