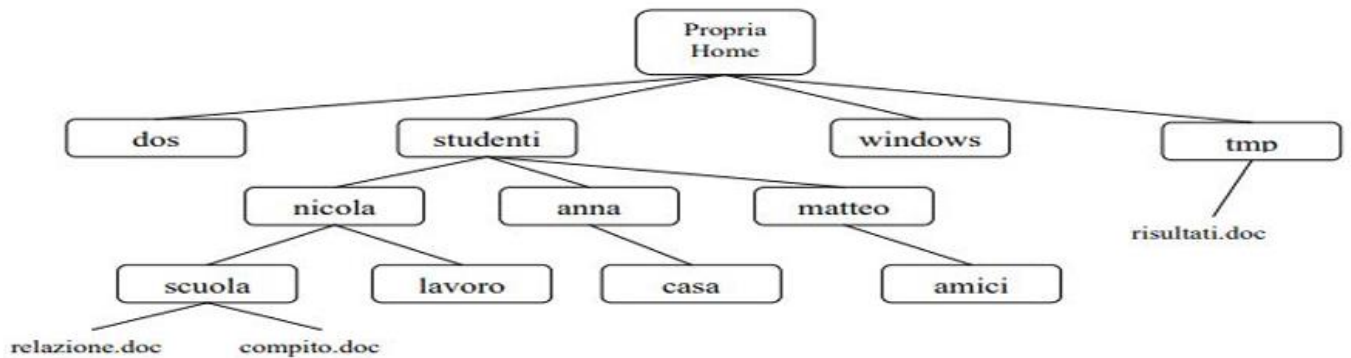


## ESERCIZIO W5D1

La seguente esercitazione ci metterà davanti all'uso del terminal delle sue keywords. Creiamo cartelle e sottocartelle come in figura adoperando il comando mkdir.



### Muoversi tra le cartelle

Cominciamo con le prime richieste. Partiamo dalla directory lavoro e mettiamo piedi nella directory casa, usando sia il percorso relativo che quello assoluto. Il procedimento è il seguente:

```
(kali@kali)-[~/studenti/nicola/lavoro]
$ .. / .. /anna/casa
(kali@kali)-[~/studenti/anna/casa]
$
```

Percorso relativo verso casa

```
(kali@kali)-[~]
$ cd studenti/nicola/lavoro
(kali@kali)-[~/studenti/nicola/lavoro]
$ cd /home/kali/studenti/anna/casa
(kali@kali)-[~/studenti/anna/casa]
$
```

spostamento in casa tramite percorso assoluto

### a) Copia di compito nella cartella correte

Una volta all'interno della sotto directory casa, copiamo il file compito per incollarlo al suo interno. Per farlo, affianchiamo al comando cp(copia) un percorso relativo/assoluto.

```
(kali㉿kali)-[~/studenti/anna/casa]
$ cp ../../nicola/scuola/compito .
(kali㉿kali)-[~/studenti/anna/casa]
$ ls
compito
```

copia di compito in casa

### b) Spostamento del file relazione nella cartella corrente

Sempre dentro casa, ci spostiamo adesso il file relazione proveniente dal percorso di nicola. Il comando giusto per riuscirci è mv, accompagnato dal percorso sorgente.

```
(kali㉿kali)-[~/studenti/anna/casa]
$ mv ../../nicola/scuola/relazione .
(kali㉿kali)-[~/studenti/anna/casa]
$ ls
compito relazione
```

file relazione spostato in casa

### c) Cancella la cartella tmp

Il prossimo obiettivo consisterà nel cancellare la cartella tmp assieme al suo contenuto risultati. Useremo rm, seguito dal percorso di destinazione.

```
(kali㉿kali)-[~/tmp]
$ cd /home/kali/studenti/anna/casa

(kali㉿kali)-[~/studenti/anna/casa]
$ rmdir /home/kali/tmp
```

eliminazione tmp

rm cancella file e directory completamente, senza mantenerle nella cache. Volendo, come nel mio caso, si potrà anche usare rmdir, a patto che la cartella risulti già vuota.

### d-g) Creare il file pippo nella cartella lavoro

Siamo alla quarta fase dell'esercitazione. Dopo aver scritto il testo appropriato con sudo nano scegliamo come destinazione, durante il salvataggio(ctrl+o), la cartella lavoro.

```
kali@kali: ~/studenti/nicola/lavoro
File Actions Edit View Help
GNU nano 7.2 pippo.txt
Mi chiamo PIPPO PIPPO PIPPO
|
```

testo pippo

```
(kali㉿kali)-[~/studenti/anna/casa]
$ sudo nano

(kali㉿kali)-[~/studenti/anna/casa]
$ ls
compito relazione

(kali㉿kali)-[~/studenti/anna/casa]
$ cd ../../studenti/nicola/lavoro
cd: no such file or directory: ../../studenti/nicola/lavoro

(kali㉿kali)-[~/studenti/anna/casa]
$ cd ../../nicola/lavoro

(kali㉿kali)-[~/studenti/nicola/lavoro]
$ ls
pippo.txt

(kali㉿kali)-[~/studenti/nicola/lavoro]
$
```

Creazione file pippo.txt

### e) Cambiare i privilegi del file pippo

Pippo necessita di un upgrade. Grazie al comando `chmod` rendiamo pippo leggibile e scrivibile per l'utente, mentre per gli altri gruppi solo leggibile. Nel terminale utilizziamo la combinazione numerica 644 per ottenere il risultato voluto.

```
(kali㉿kali)-[~/studenti/nicola/lavoro]
$ cd /home/kali/studenti/anna/casa

(kali㉿kali)-[~/studenti/anna/casa]
$ sudo chmod 644 /home/kali/studenti/nicola/lavoro/pippo.txt

(kali㉿kali)-[~/studenti/anna/casa]
$
```

cambio privilegi pippo

### f) Nascondere il contenuto della cartella Anna

Da qui in poi nascondiamo la sotto cartella di Anna, ovvero casa. Per farlo ci basterà il comando `mv`. Sovrascriviamo la cartella casa e la rinominiamo in `.casa`. La cartella non sarà identificabile con `ls`, a meno che non aggiungiamo `-l` (long list format).

```
(kali㉿kali)-[~/studenti/anna]
$ ls
casa

(kali㉿kali)-[~/studenti/anna]
$ mv casa .casa

(kali㉿kali)-[~/studenti/anna]
$ ls

(kali㉿kali)-[~/studenti/anna]
$ ls -a
.  ..  .casa

(kali㉿kali)-[~/studenti/anna]
$
```

Nascondino di casa

## h) Rimuovere la cartella amici

Prima o poi arriva il momento di augurare buon viaggio ai nostri amici, ed eccoci qua. Forse `rm -r` (recursive) è un po' troppo potente, ma dovevamo assicurarci di non lasciare nessun testimone.

```
(kali㉿kali)-[~/studenti/nicola/lavoro]
$ rm -r ../../matteo/amici

(kali㉿kali)-[~/studenti/nicola/lavoro]
$ cd ../../matteo

(kali㉿kali)-[~/studenti/matteo]
$ ls
addio, addio, amici addio

(kali㉿kali)-[~/studenti/matteo]
$
```

## i) Rimuovere tutte le cartelle create

Come ultima task eliminiamo tutte le cartelle che abbiamo creato per l'esercizio. Usiamo `rm -r` e confermiamo con `y` l'eventuale cancellazione di write protected files.

```
(kali㉿kali)-[~]
$ rm -r studenti
rm: remove write-protected regular file 'studenti/nicola/lavoro/pippo.txt'? y
rm: remove write-protected regular file 'studenti/nicola/scuola/compito'? y
rm: remove write-protected regular file 'studenti/anna/.casa/relazione'? y

(kali㉿kali)-[~]
$
```

## Esercizi processi

Dopo un veloce controllo delle funzionalità rese disponibili da `w`, `who`, e `whoami`, ci accingiamo adesso ad esplorare i comandi relativi ai processi.

```
(kali㉿kali)-[~]
$ w
09:31:15 up 14 min, 1 user, load average: 0.23, 0.23, 0.19
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU WHAT
kali      tty7          :0           09:17    14:21  10.72s  0.61s xfce4-sessio

(kali㉿kali)-[~]
$ who
kali      tty7          2023-11-22 09:17 (:0)

(kali㉿kali)-[~]
$ whoami
kali

(kali㉿kali)-[~]
$
```

w, who, whoami running test

## 1) Aprire un terminale

Come da indicazione apriamo un terminale usando l'interfaccia grafica, oppure lo shortcut Ctrl+Alt+T.

## 2) Leggere il manuale del comando job, ps e kill

Il comando man ci consente di visualizzare su terminale le istruzioni relative a questi comandi. Affianchiamo ad esso i comandi del quale vogliamo sapere di più, et voila.

```
PS(1)                                User Commands                                PS(1)

NAME
    ps - report a snapshot of the current processes.

SYNOPSIS
    ps [options]

DESCRIPTION
    ps displays information about a selection of the active processes.
    If you want a repetitive update of the selection and the displayed
    information, use top instead.

    This version of ps accepts several kinds of options:

    1  UNIX options, which may be grouped and must be preceded by a
       dash.
    2  BSD options, which may be grouped and must not be used with a
       dash.
    3  GNU long options, which are preceded by two dashes.

    Options of different types may be freely mixed, but conflicts can
    appear. There are some synonymous options, which are functionally
    identical, due to the many standards and ps implementations that
    this ps is compatible with.

    Note that ps -aux is distinct from ps aux. The POSIX and UNIX
    standards require that ps -aux print all processes owned by a user

Manual page ps(1) line 1 (press h for help or q to quit)
```

man ps - manuale

```
KILL(1)                                User Commands                                KILL(1)

NAME
    kill - send a signal to a process

SYNOPSIS
    kill [options] <pid> [ ... ]

DESCRIPTION
    The default signal for kill is TERM. Use -l or -L to list available signals. Particularly useful signals include HUP, INT, KILL, STOP, CONT, and 0. Alternate signals may be specified in three ways: -9, -SIGKILL or -KILL. Negative PID values may be used to choose whole process groups; see the PGID column in ps command output. A PID of -1 is special; it indicates all processes except the kill process itself and init.

OPTIONS
    <pid> [ ... ]
        Send signal to every <pid> listed.

    -<signal>
    -s <signal>
    --signal <signal>
        Specify the signal to be sent. The signal can be specified by using name or number. The behavior of signals is explained in signal(7) manual page.

Manual page kill(1) line 1 (press h for help or q to quit)
```

A parte i ps e kill, job non possiede un libretto di istruzioni richiamabile, per questo dovremo usare builtins. Ci toccherà scorrere la gerarchia alfabetica fino a che non troveremo job.

```
jobs [-lnprs] [ jobspec ... ]
jobs -x command [ args ... ]
    The first form lists the active jobs. The options have the following meanings:
    -l      List process IDs in addition to the normal information.
    -n      Display information only about jobs that have changed status since the user was last notified of their status.
    -p      List only the process ID of the job's process group leader.
    -r      Display only running jobs.
    -s      Display only stopped jobs.

    If jobspec is given, output is restricted to information about that job. The return status is 0 unless an invalid option is encountered or an invalid jobspec is supplied.

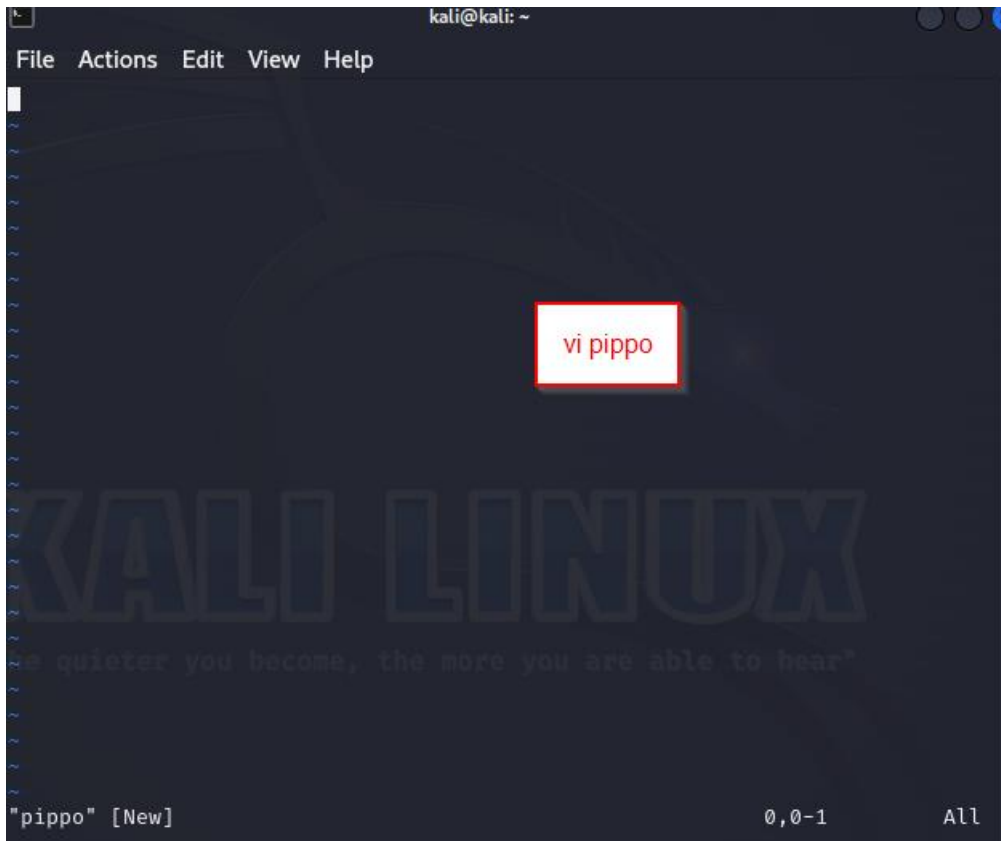
    If the -x option is supplied, jobs replaces any jobspec found in command or args with the corresponding process group ID, and executes command passing it args, returning its exit status.

    kill [-s sigspec | -n signum | -sigspec] [pid | jobspec] ...

Manual page builtins(7) line 823 (press h for help or q to quit)
```

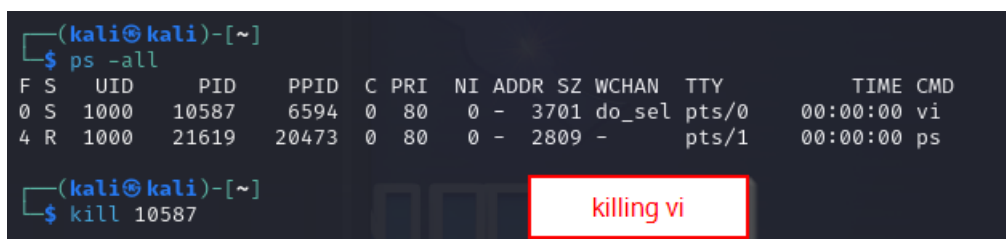
### 3) Lanciare il comando vi pippo

Andiamo ad aprire l'editor di testo vi chiamato pippo.



### 4) Aprire un nuovo terminale e visualizzare tutti i processi

Apriamo il terminale ed inseriamo il comando `ps -all` per avere una panoramica di tutti i processi attivi.





## 5) Cercare di terminare il processo vi per sbloccare il terminale

Se tutto procede bene il comando kill, seguito dal codice identificativo del processo, darà in output un'immagine come questa.

```
(kali㉿kali)-[~]  
$ vi pippo  
Vim: Caught deadly signal TERM  
Vim: Finished.  
  
zsh: terminated vi pippo
```

Pippo era una  
brava persona

## 6-7) Lanciare il comando firefox in background

Eseguire in bg un processo ci permette di renderlo indipendente da noi mentre utilizziamo il terminale. Il comando per farlo sarà `firefox &`.

```
(kali㉿kali)-[~]  
$ firefox &  
[1] 24399  
  
(kali㉿kali)-[~]  
$  
[1] + done          firefox  
(kali㉿kali)-[~]  
$
```

running  
firefox on bg

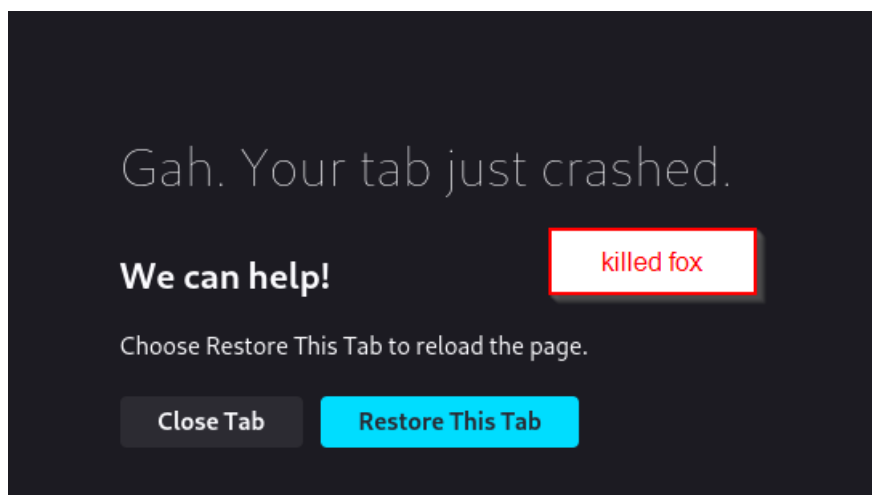
## 8) Cercare di terminare il processo firefox

Ci aiutiamo con `ps -eaf` per cercare tra i processi in funzione. Sicuramente esistono modi migliori per uccidere un processo, ma mi sono trovato più a mio agio con `ps -eaf`. Basta cercare il percorso il percorso giusto. Di solito lo troveremo proprio in fondo alla lista. Se la terminazione riesce, la schermata successiva a questa mostrerà un messaggio di errore.

```
(kali@kali)-[~]
$ ps -eaf
```

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	1	0	0	09:16	?	00:00:01	/sbin/init splash
root	2	0	0	09:16	?	00:00:00	[kthreadd]
root	3	2	0	09:16	?	00:00:00	[rcu_gp]
root	4	2	0	09:16	?	00:00:00	[rcu_par_gp]
root	5	2	0	09:16	?	00:00:00	[slub_flushwq]
root	6	2	0	09:16	?	00:00:00	[netns]
root	11	2	0	09:16	?	00:00:00	[mm_percpu_wq]
root	12	2	0	09:16	?	00:00:00	[rcu_tasks_kthread]
root	13	2	0	09:16	?	00:00:00	[rcu_tasks_rude_kthread]
root	14	2	0	09:16	?	00:00:00	[rcu_tasks_trace_kthread]
root	15	2	0	09:16	?	00:00:00	[ksoftirqd/0]
root	16	2	0	09:16	?	00:00:01	[rcu_preempt]
root	17	2	0	09:16	?	00:00:00	[migration/0]
root	18	2	0	09:16	?	00:00:00	[idle_inject/0]
root	19	2	0	09:16	?	00:00:00	[cpuhp/0]
root	20	2	0	09:16	?	00:00:00	[cpuhp/1]
root	21	2	0	09:16	?	00:00:00	[idle_inject/1]
root	22	2	0	09:16	?	00:00:00	[migration/1]
root	23	2	0	09:16	?	00:00:00	[ksoftirqd/1]
root	25	2	0	09:16	?	00:00:00	[kworker/1:0H-events_high]
root	26	2	0	09:16	?	00:00:00	[cpuhp/2]
root	27	2	0	09:16	?	00:00:00	[idle_inject/2]
root	28	2	0	09:16	?	00:00:00	[migration/2]
root	29	2	0	09:16	?	00:00:00	[ksoftirqd/2]

attempt to kill fox



## 9) Verificare quanto spazio si sta occupando su disco

Per ultimo, visto che abbiamo a cuore la salute del nostro Mushu chiamato Kali Linux, diamo un'occhiata alla memoria libera sul disco. Usiamo df accompagnato da -h per vedere dei numeri più comprensibili.

```
(kali@kali)-[~]  
$ df -h  
Filesystem      Size  Used Avail Use% Mounted on  
udev            2.4G   0    2.4G   0% /dev  
tmpfs           499M  960K  498M   1% /run  
/dev/sda1       79G   17G   58G  23% /  
tmpfs           2.5G   0    2.5G   0% /dev/shm  
tmpfs           5.0M   0    5.0M   0% /run/lock  
tmpfs           499M  76K   499M   1% /run/user/1000
```

