

Esercizio S3L2 - Gameshell Livelli 1 > 10

L'esercizio di oggi verteva sull'utilizzo dei comandi base per spostarsi all'interno delle directories di Kali: Kali possiede una user interface assolutamente utilizzabile, ma per l'utilizzo i comandi dal terminale risultano molto più immediati e veloci una volta che ne si comprende appieno l'utilizzo. Per comprenderli in maniera interattiva, abbiamo installato Gameshell sul nostro terminale Kali, con l'obiettivo di completare i primi 10 livelli.

```

Mission goal
-----
Go to the top of the main tower of the castle.

Useful commands
-----
cd LOCATION
  Move to the given location.
  Remark: "cd" is an abbreviation for "change directory".

pwd
  Show the path to your current location.
  Remark: "pwd" is an abbreviation for "print working directory".

ls
  Show a list of locations that are currently accessible.
  Remark: "ls" is an abbreviation of "list".

gsh check
  Check if the mission objective has been achieved.

gsh reset
  Restart the mission from the beginning.

Remarks
-----
UPPERCASE words appearing in commands are meta-variables: you need to replace them by
appropriate (string) values.
Most filesystems treat uppercase and lowercase characters differently. Make sure you use the
correct path.
-----
(0)-----(0)

[use 'gsh help' to get a list of available commands]
[mission 1] $ cd Castle
[use 'gsh help' to get a list of available commands]
[mission 1] $ ls
Cellar Great_hall Main_building Main_tower Observatory
[use 'gsh help' to get a list of available commands]
[mission 1] $ cd Main_tower/First_floor/Second_floor
[use 'gsh help' to get a list of available commands]
[mission 1] $ ls
Top_of_the_tower
[use 'gsh help' to get a list of available commands]
[mission 1] $ cd Top_of_the_tower
[use 'gsh help' to get a list of available commands]
[mission 1] $ gsh check
Congratulations, mission 1 has been successfully completed!

```

```
( )=( )
( )=( )
Mission goal
Go the castle's cellar.
Secondary objective
Understand the difference between ``cd -`` and ``cd ..``.
Useful commands
cd -
Jump back to the location you were in prior to your last move.
cd ..
Move to the parent directory (one step back along the path to your current location).
pwd
See the path to your current location.
( )=( )
( )=( )
[use 'gsh help' to get a list of available commands]
[mission 2] $ pwd
/home/kali/gameshell.1/World/Castle/Main_tower/First_floor/Second_floor/Top_of_the_tower
[use 'gsh help' to get a list of available commands]
[mission 2] $ cd ..
cd..: command not found
[use 'gsh help' to get a list of available commands]
[mission 2] $ cd ..
[use 'gsh help' to get a list of available commands]
[mission 2] $ cd ..
[use 'gsh help' to get a list of available commands]
[mission 2] $ cd ..
[use 'gsh help' to get a list of available commands]
[mission 2] $ pwd
/home/kali/gameshell.1/World/Castle/Main_tower
[use 'gsh help' to get a list of available commands]
[mission 2] $ cd ..
[use 'gsh help' to get a list of available commands]
[mission 2] $ cd Cellar
[use 'gsh help' to get a list of available commands]
[mission 2] $ gsh check
Congratulations, Mission 2 has been successfully completed!
```

```
| |
--+-----+--
| Use the command |
|   $ gsh help   |
| to get the list of "gsh" commands. |
--+-----+--
| |

[mission 3] $ cd

[use 'gsh help' to get a list of available commands]
[mission 3] $ cd Castle/Main_building/Throne_room

[use 'gsh help' to get a list of available commands]
[mission 3] $ gsh check

Congratulations, mission 3 has been successfully completed!
```

I primi tre livelli ci hanno insegnato il comando “cd”, forse il più importante e fondamentale fra i comandi visti oggi: “cd + directoryname” ci permette di muoverci all’interno delle directories di Kali, con solo il comando base e il nome della directory dobbiamo trovarci già all’interno della cartella principale contenente la sottocartella che interessa a noi, alternativamente possiamo tornare alla home scrivendo solamente cd, mentre separando directories e subdirectories con degli slash possiamo fare balzi di più cartelle come suggeritoci nel livello 3. Solo muovendoci all’interno delle cartelle potremo fare un check di tutto ciò che è contenuto (salvo i file nascosti) con il comando “ls”. Questo ci aiuta a muoverci con i nomi precisi delle directories.

```

Mission goal
Build a "Hut" in the forest, and then build a "Chest" in the hut.

Useful commands

mkdir DIRECTORY
Create a new directory inside the current directory.
Remark: "mkdir" is an abbreviation for "make directory".

~/Castle/Main_building/Throne_room
[mission 4] $ cd
~
[mission 4] $ cd Forest
~/Forest
[mission 4] $ mkdir Hut
~/Forest
[mission 4] $ cd Hut
~/Forest/Hut
[mission 4] $ mkdir Chest
~/Forest/Hut
[mission 4] $ gsh check

Congratulations, mission 4 has been successfully completed!

```

```

~/Forest/Hut
[mission 5] $ gsh goal

Mission goal
Go back to the cellar and get rid of all the spiders. Leave the bats alone: they appear on
the castle's coat of arms and are said to confer luck.

Useful commands

rm FILE1 FILE2 ... FILEn
Delete the files (permanently).
Remark: "rm" is an abbreviation for "remove".

~/Forest/Hut
[mission 5] $ cd
~
[mission 5] $ cd Castle/Cellar
~/Castle/Cellar
[mission 5] $ rm spider_1 spider_2 spider_3
~/Castle/Cellar
[mission 5] $ ls
barrel_of_apples bat_1 bat_2
~/Castle/Cellar
[mission 5] $ gsh check

```

```

Mission goal
Collect all the coins that you can find in the garden in front of the castle, and put them in
your chest in your hut in the forest.

Useful commands
mv FILE1 FILE2 ... FILEn DIRECTORY
Move the files to the directory.
Remark: "mv" is an abbreviation of "move".

~
The "~" symbol is an abbreviation for the initial directory.
Example: wherever you are, "~/Tavern" denotes the directory (or file) "Tavern" in the
initial directory.

~/Castle/Cellar
[mission 6] $ cd ~
/home/kali/gameshell.1/World

[mission 6] $ cd Castle

~/Castle
[mission 6] $ ls
Cellar Great_hall Main_building Main_tower Observatory

~/Castle
[mission 6] $ cd

[mission 6] $ ls
Castle Forest Garden Mountain Stall

[mission 6] $ cd Garden

~/Garden
[mission 6] $ ls
coin_1 coin_2 coin_3 Flower_garden Maze Shed

~/Garden
[mission 6] $ mv coin_1 coin_2 coin_3 ~/Forest/Hut/Chest

~/Garden
[mission 6] $ cd ..

[mission 6] $ cd Forest/Hut/Chest

~/Forest/Hut/Chest
[mission 6] $ ls
coin_1 coin_2 coin_3

~/Forest/Hut/Chest
[mission 6] $ gsh check

```

```

~/Forest/Hut/Chest
[mission 7] $ gsh goal

()=(
Mission goal
Collect all the coins hidden in the garden in front of the castle, and put them in your chest
(in your hut in the forest).

Secondary objective
Learn how to use the "Tab" key to go faster.

Useful commands
ls -A
List all the files of the current directory, including hidden files. (A file is "hidden"
when its name starts with a dot.)

Tab
The tabulation key "completes" the name of a file or directory once you have typed the
beginning of its name. This only works
if there is only one possible completion.

Tab-Tab
Pressing tabulation twice successively shows a list of possible completions.

()=(

~/Forest/Hut/Chest
[mission 7] $ cd ~/Garden

~/Garden
[mission 7] $ ls -A
.20703_coin_2 .25666_coin_1 .3928_coin_3 Flower_garden Maze Shed

~/Garden
[mission 7] $ mv .20703_coin_2 .25666_coin_1 .3928_coin_3 ~/Forest/Hut/Chest

~/Garden
[mission 7] $ cd ~/Forest/Hut/Chest

~/Forest/Hut/Chest
[mission 7] $ ls
.20703_coin_2 .25666_coin_1 .3928_coin_3 coin_1 coin_2 coin_3

~/Forest/Hut/Chest
[mission 7] $ ls
.20703_coin_2 .25666_coin_1 .3928_coin_3 coin_1 coin_2 coin_3

~/Forest/Hut/Chest
[mission 7] $

~/Forest/Hut/Chest
[mission 7] $ gsh check

```

Gli esercizi 4, 5, 6 e 7 vertevano sui comandi “rm”, “mv” e “mkdir”; questi comandi rispettivamente servono a rimuovere e spostare file o subdirectories da una directory, o per creare una nuova directory: “cp file_name directory_name” copia il file_name alla directory_name. I comandi di movimento combinato di cd valgono anche per il path delle subdirectories di questi comandi. “rm file_name” rimuove file_name dalla directory in cui ci troviamo al momento. “mv file_name directory_name” è come se fosse un taglia incolla, dove file_name è l’oggetto e directory_name è la destinazione. “mkdir directory_name” crea

invece una cartella con il nome scelto al posto della variabile "directory_name". Nell'ultimo esercizio vediamo come individuare invece anche i file nascosti che non troveremmo col semplice comando "ls": ls -A ci dà una visione di tutti i file all'interno di una directory, compresi quelli nascosti.

```

~/Forest/Hut/Chest
[mission 8] $ gsh goal

Mission goal

Get rid of all the spiders that are crawling in the cellar. Again, do not do disturb the bats.

Shell patterns

*
The "*" character stands in for any sequence of characters
(including an empty sequence).

?
The "?" character stands in for any single character.

Those wildcards can be used to denote lists of existing files / directories in the current
working directory.

For example: if the current folder contains
file-1 Folder-1 file-14 potato
then
*      → file-1 Folder-1 file-14 potato
*1     → file-1 Folder-1
*0*    → Folder-1 potato
*x*    → error, no matching file
*?     → file-1 Folder-1
*-*?   → file-14

~/Forest/Hut/Chest
[mission 8] $ cd ~/Castle/Cellar

~/Castle/Cellar
[mission 8] $ ls -A
10157_bat_2  1432_bat_3  19290_spider_25  22469_spider_43  26065_spider_49  30999_spider_16  3830_spider_35
10413_bat_1  1632_bat_5  19440_spider_16  2309_spider_30  26882_spider_12  32491_spider_11  6779_spider_46
10694_spider_5  16581_spider_32  19832_spider_44  23589_spider_20  26133_spider_36  32580_spider_4  7052_spider_18
11472_spider_33  17283_spider_31  19944_spider_3  2375_spider_7  26498_spider_27  3874_spider_30  7492_spider_42
12008_spider_2  17635_spider_47  20427_spider_24  24387_spider_22  27006_spider_48  4076_spider_9  8004_spider_23
13116_spider_41  17942_spider_13  21087_spider_19  24791_bat_4  27489_spider_17  4746_spider_20  8111_spider_15
13145_spider_6  17957_spider_34  21234_bat_5  24888_spider_37  29622_spider_10  540_spider_21  897_spider_29
14177_spider_38  186_spider_50  22171_spider_40  25877_spider_1  29861_spider_45  5558_spider_8  barrel_of_apples

~/Castle/Cellar
[mission 8] $ rm *spider*

~/Castle/Cellar
[mission 8] $ ls -A
10157_bat_2  10413_bat_1  1432_bat_3  21234_bat_5  24791_bat_4  barrel_of_apples

~/Castle/Cellar
[mission 8] $ gsh check

```

```

~/Castle/Cellar
[mission 9] $ gsh goal

Mission goal

The spiders are getting clever: they found a way to hide.
Get rid of all the spiders that are hiding in the cellar without disturbing the bats.

Shell patterns

*
The "*" character stands in for any sequence of characters (including an empty sequence).

?
The "?" character stands in for any single character.

Remark

The wildcards "*" and "?" don't see hidden files, you need to add an explicit dot at the
start of the pattern.

~/Castle/Cellar
[mission 9] $ ls -A
10157_bat_2  1432_bat_3  .21008_spider_29  .20191_spider_36  .32_spider_7  .8068_spider_49
10413_bat_1  14779_spider_11  .22023_bat_1  .20437_spider_42  .3313_spider_22  8103_spider_17
10990_spider_38  .15852_spider_3  .22208_spider_32  .30000_spider_14  .3952_spider_25  .8437_spider_21
10413_bat_1  .16672_spider_39  .23176_spider_45  .30173_spider_9  .551_bat_5  .9609_spider_8
11335_spider_37  .17484_spider_19  24791_bat_4  .304_spider_30  .5569_spider_28  .9974_spider_20
11910_spider_31  .17171_spider_26  .25238_spider_27  .31269_spider_6  .6115_spider_23  barrel_of_apples
12593_spider_47  .18419_spider_24  .26201_bat_3  .31360_spider_34  .6236_spider_5
12946_spider_48  .18451_spider_30  .26267_spider_50  .31380_spider_2  .6253_spider_35
13265_spider_40  .19725_spider_13  .26463_spider_15  .31149_spider_4  .6385_spider_44
13342_spider_1  21234_bat_5  .27414_spider_16  .32616_spider_43  .7004_spider_12
13774_spider_18  .21726_spider_46  .28430_bat_4  .32736_bat_2  .7947_spider_41

~/Castle/Cellar
[mission 9] $ rm ?spider?
rm: cannot remove '?spider?': No such file or directory

~/Castle/Cellar
[mission 9] $ rm *.21*
rm: cannot remove '*.21*': No such file or directory

~/Castle/Cellar
[mission 9] $ rm *.21
rm: cannot remove '*.21': No such file or directory

~/Castle/Cellar
[mission 9] $ rm .*.spider.*

~/Castle/Cellar
[mission 9] $ ls -A
10157_bat_2  1432_bat_3  .22023_bat_1  .26201_bat_3  .32736_bat_2  barrel_of_apples
10413_bat_1  21234_bat_5  24791_bat_4  .28430_bat_4  .551_bat_5

~/Castle/Cellar
[mission 9] $ gsh check

```

```

~/Castle/Cellar
[mission 10] $ gsh goal

()=(
    Mission goal
    =====
    You have taken a fancy to the four standards in the great hall of the castle. As stealing
    them would not go unnoticed, put a copy (same name, same content) of each in your chest.

    Useful commands
    =====
    cp FILE DIRNAME
    Copy the file to the directory.
    Remark: ``cp`` is an abbreviation of "copy".
()=(

~/Castle/Cellar
[mission 10] $ cd ..

~/Castle
[mission 10] $ cd Great_hall

~/Castle/Great_hall
[mission 10] $ ls
22187_decorative_shield 30446_suit_of_armour 45161_stag_head standard_1 standard_2 standard_3 standard_4

~/Castle/Great_hall
[mission 10] $ cp standard_1 standard_2 standard_3 standard_4 ^C

~/Castle/Great_hall
[mission 10] $ cp standard_1 standard_2 standard_3 standard_4 $home/Forest/Hut/Chest
cp: target '/Forest/Hut/Chest': No such file or directory

~/Castle/Great_hall
[mission 10] $ cp standard_1 standard_2 standard_3 standard_4 ~/Forest/Hut/Chest

~/Castle/Great_hall
[mission 10] $ gsh check

```

I livelli 8, 9 e 10 invece spiegavano come utilizzare i pattern gli shell pattern, che permettono di raggruppare con delle condizionali file con nomi simili per non doverli cancellare uno alla volta o scrivere l'intera lista. Ad esempio, se vogliamo cancellare tutti i file "spider", ci basterà scrivere "rm *spider*" e verranno rimossi tutti i file che contengono spider: il carattere ? invece sostituisce un qualsiasi carattere all'interno del nome di un file. L'ultimo esercizio ci ha anche insegnato un ultimo comando, ovvero il comando "cp": proprio come il comando mv è un taglia e incolla a cui siamo abituati, cp è la versione a terminale del copia e incolla. "cp file_name directory_name" copia il file_name fino alla directory definita dalla variabile directory_name".