Homework 1 Solutions

BIOL BC3308: Genomics and Bioinformatics

Due Date: 01/28/2016

1 General Notes

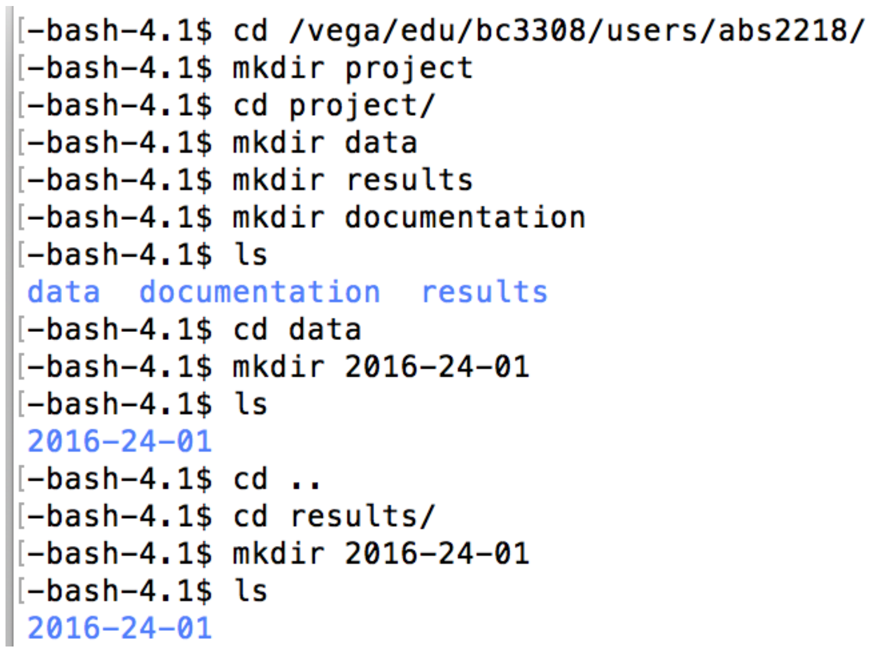
You should have submitted three files in a folder named "Homeworks" under your shared folder in yeti (the specific path would be /vega/edu/bc3308/UNI/Homeworks/).

- yourname_HW1_EX1.txt for the first exercise
- the list of passwords for the second exercise
- the script log for the second exercise

2 Exercise 1

The purpose of reading the paper written by William Stafford Noble was show you a systematic way of organizing bioinformatics projects. This organization scheme is what you will be using for your Qiime project and independent project.

In this exercise, you were expected to create 6 directories: project, data, results, documentation, and two dated directories, one in the data directory and one in the results directory, using the **mkdir** command. It should look something like this:



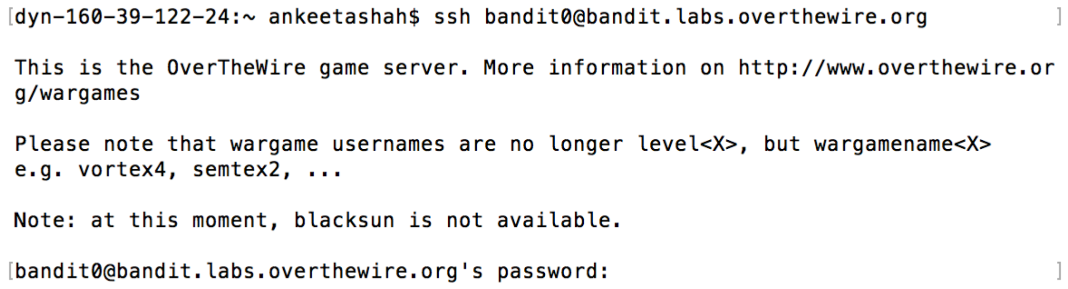
```
[-bash-4.1$ cd /vega/edu/bc3308/users/abs2218/
[-bash-4.1$ mkdir project
[-bash-4.1$ cd project/
[-bash-4.1$ mkdir data
[-bash-4.1$ mkdir results
[-bash-4.1$ mkdir documentation
[-bash-4.1$ ls
 data  documentation  results
[-bash-4.1$ cd data
[-bash-4.1$ mkdir 2016-24-01
[-bash-4.1$ ls
 2016-24-01
[-bash-4.1$ cd ..
[-bash-4.1$ cd results/
[-bash-4.1$ mkdir 2016-24-01
[-bash-4.1$ ls
 2016-24-01
```

3 Exercise 2

There are many ways that you could have beaten levels 0 through 13. What is outlined below is just an example. It is highly recommend that you work through certain levels again, particularly those you found tricky, this time maybe looking at the solutions.

3.1 Level 0

This level asked you to SSH into game's server with the provided username (bandit0) and password (bandit0).



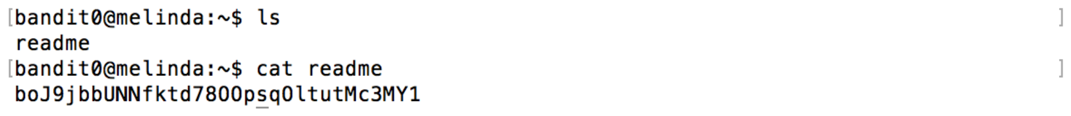
```
[dyn-160-39-122-24:~ ankeetashah$ ssh bandit0@bandit.labs.overthewire.org          ]

 This is the OverTheWire game server. More information on http://www.overthewire.or
 g/wargames

 Please note that wargame usernames are no longer level<X>, but wargamename<X>
 e.g. vortex4, semtex2, ...

 Note: at this moment, blacksun is not available.

[bandit0@bandit.labs.overthewire.org's password:                                    ]
```

3.2 Level 0 to 1

Notable commands:

ls to list the contents of the current directory.

cat to print the contents of the file "readme" into the terminal window. You also could have used the **head** command, the **more** command, or the **less** command instead - all allow you visualize the contents of a file.

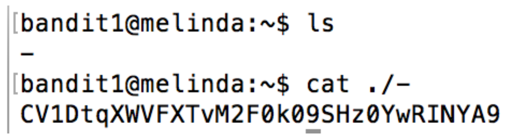


```
bandit0@melinda:~$ ls
readme
bandit0@melinda:~$ cat readme
boJ9jbbUNNfktd78OOpsqOltutMc3MY1
```

3.3 Level 1 to 2

Notable commands:

cat ./- to print the contents of the file "-" into the terminal window. Note that here you have to use ./ prior to entering the filename because a dash in Linux/Unix is a meta-character, meaning that it is normally reserved for flagging commands. In this case, you are using the ./ syntax to tell the operating system that you mean that dash for a filename, not a flag.



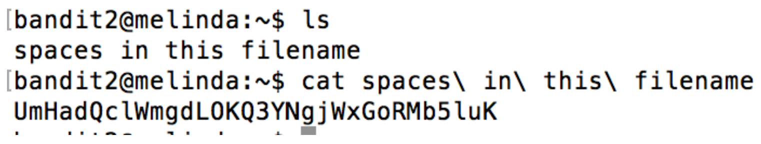
```
bandit1@melinda:~$ ls
-
bandit1@melinda:~$ cat ./-
CV1DtqXWVFXTvM2F0k09SHz0YwRINYA9
```

3.4 Level 2 to 3

Notable concept:

\ is used after every word in a filename that contains spaces. Just like a dash, a space in Linux/Unix is a meta-character.

Note also that an easy way to do this without having to type the backslash after every word is by using tab completion.

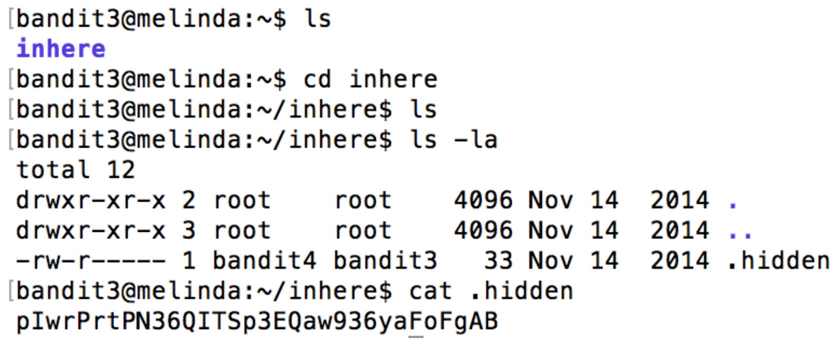


```
bandit2@melinda:~$ ls
spaces in this filename
bandit2@melinda:~$ cat spaces\ in\ this\ filename
UmHadQclWmgdLOKQ3YNgjWxGoRMb5luK
```

3.5 Level 3 to 4

Notable flags:

ls -la remember that the -la flag is great for seeing all files in a particular directory, even those that have names that start with period (hidden files).

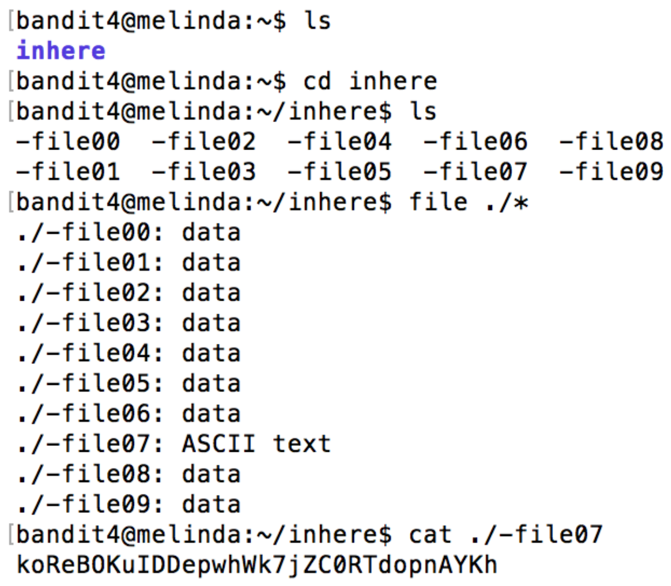


```
[bandit3@melinda:~$ ls
 inhere
[bandit3@melinda:~$ cd inhere
[bandit3@melinda:~/inhere$ ls
[bandit3@melinda:~/inhere$ ls -la
 total 12
 drwxr-xr-x 2 root    root    4096 Nov 14  2014 .
 drwxr-xr-x 3 root    root    4096 Nov 14  2014 ..
 -rw-r----- 1 bandit4 bandit3   33 Nov 14  2014 .hidden
[bandit3@melinda:~/inhere$ cat .hidden
 pIwrPrtPN36QITSp3EQaw936yaFoFgAB
```

3.6 Level 4 to 5

Notable commands:

file is a command that allows you to view the type of data contained within a file. ASCII stands for American Standard Code for Information Interchange. Computers can only understand numbers, so an ASCII code is the numerical representation of characters. As the instructions suggest, the password was located in the only human-readable file in this directory, which is the one that contains ASCII characters.

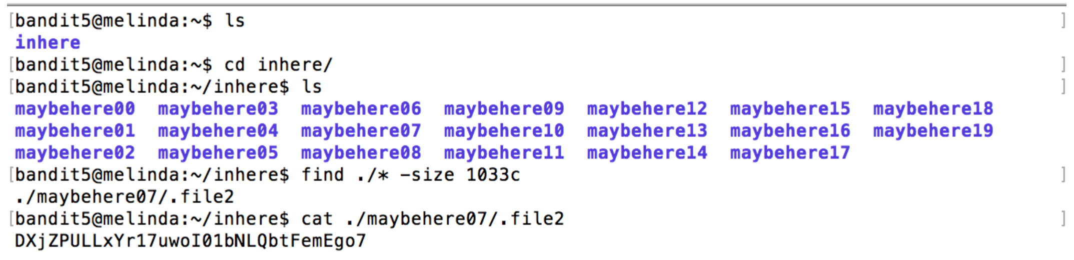


```
bandit4@melinda:~$ ls
inhere
bandit4@melinda:~$ cd inhere
bandit4@melinda:~/inhere$ ls
-file00  -file02  -file04  -file06  -file08
-file01  -file03  -file05  -file07  -file09
bandit4@melinda:~/inhere$ file ./*
./-file00: data
./-file01: data
./-file02: data
./-file03: data
./-file04: data
./-file05: data
./-file06: data
./-file07: ASCII text
./-file08: data
./-file09: data
bandit4@melinda:~/inhere$ cat ./-file07
koReBOKuIDDepwhWk7jZC0RTdopnAYKh
```

3.7 Level 5 to 6

Notable commands:

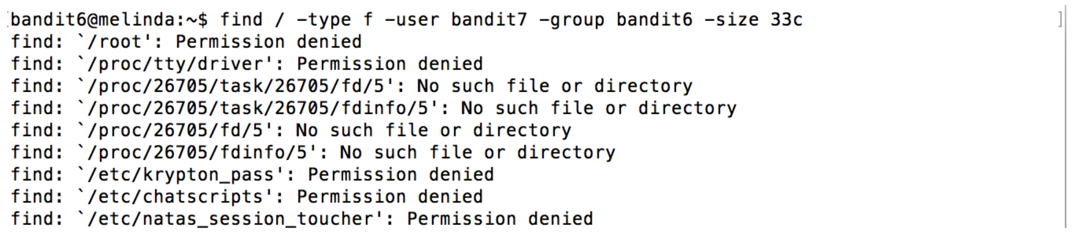
find is a command you should probably read the man page for. It has many flags. The one used here was for the -size flag and the c at the end of the number represents the number of bytes, given as hint in the game instructions.



```
bandit5@melinda:~$ ls
inhere
bandit5@melinda:~$ cd inhere/
bandit5@melinda:~/inhere$ ls
maybehere00  maybehere03  maybehere06  maybehere09  maybehere12  maybehere15  maybehere18
maybehere01  maybehere04  maybehere07  maybehere10  maybehere13  maybehere16  maybehere19
maybehere02  maybehere05  maybehere08  maybehere11  maybehere14  maybehere17
bandit5@melinda:~/inhere$ find ./* -size 1033c
./maybehere07/.file2
bandit5@melinda:~/inhere$ cat ./maybehere07/.file2
DXjZPULLxYr17uwoI01bNLQbtFemEgo7
```

3.8 Level 6 to 7

You probably did something like what is right below and got a bunch of permissions errors. This is what makes this level particularly tricky.



```
bandit6@melinda:~$ find / -type f -user bandit7 -group bandit6 -size 33c
find: `/root': Permission denied
find: `/proc/tty/driver': Permission denied
find: `/proc/26705/task/26705/fd/5': No such file or directory
find: `/proc/26705/task/26705/fdinfo/5': No such file or directory
find: `/proc/26705/fd/5': No such file or directory
find: `/proc/26705/fdinfo/5': No such file or directory
find: `/etc/krypton_pass': Permission denied
find: `/etc/chatscripts': Permission denied
find: `/etc/natas_session_toucher': Permission denied
```

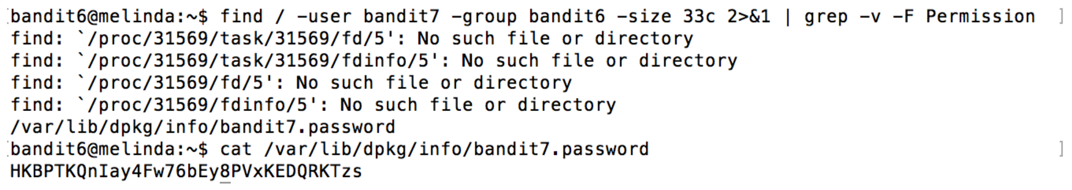
Notable commands/ideas:

find -type f lets you search for a something on the server that is of type file.

pipes to combine commands.

2 > &1 is a little strange and probably required a bit of web searching to figure out. 2 > &1 redirects standard error to standard output where it can be piped to grep to ignore all lines that contain the Permission denied error message.

grep is the key to solving this level. Remember, grep is used for searching for lines matching a regular expression.

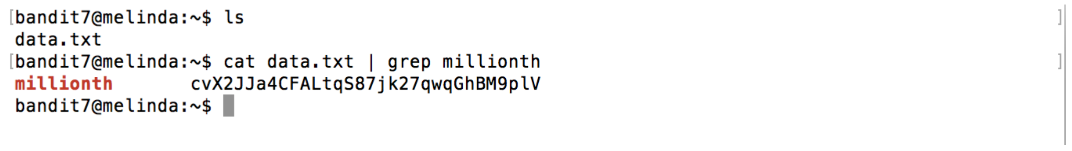


```
bandit6@melinda:~$ find / -user bandit7 -group bandit6 -size 33c 2>&1 | grep -v -F Permission
find: `/proc/31569/task/31569/fd/5': No such file or directory
find: `/proc/31569/task/31569/fdinfo/5': No such file or directory
find: `/proc/31569/fd/5': No such file or directory
find: `/proc/31569/fdinfo/5': No such file or directory
/var/lib/dpkg/info/bandit7.password
bandit6@melinda:~$ cat /var/lib/dpkg/info/bandit7.password
HKBPTKQnIay4Fw76bEy8PVxKEDQRKTzs
```

3.9 Level 7 to 8

Notable commands:

grep again to extract the line that contains the word "millionth".



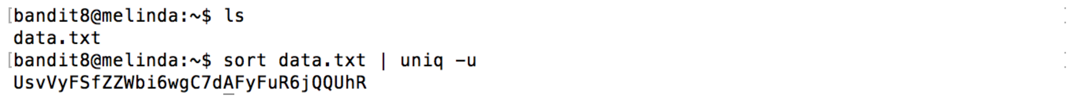
```
bandit7@melinda:~$ ls
data.txt
bandit7@melinda:~$ cat data.txt | grep millionth
millionth       cvX2JJa4CFALtqS87jk27qwqGhBM9plV
bandit7@melinda:~$
```

3.10 Level 8 to 9

Notable commands:

sort sorts the contents of a text file, line by line.

uniq reports or filters out repeated lines in a file.

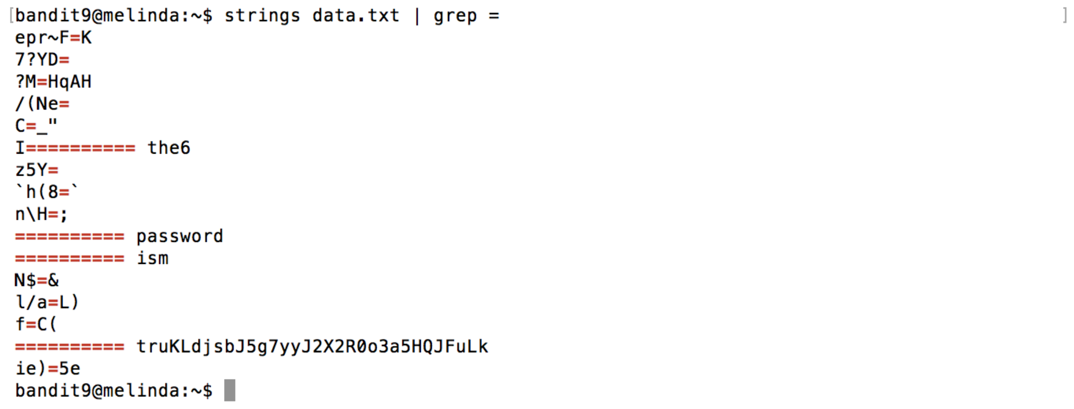


```
bandit8@melinda:~$ ls
data.txt
bandit8@melinda:~$ sort data.txt | uniq -u
UsvVyFSfZZWbi6wgC7dAFyFuR6jQQUhR
```

3.11 Level 9 to 10

Notable commands:

strings is command that prints strings, which is just a sequence characters (you will learn more about the idea of data types in the Python section of this course).

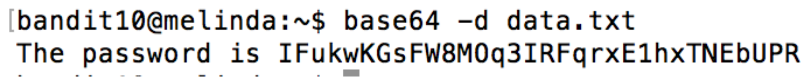


```
bandit9@melinda:~$ strings data.txt | grep =
epr~F=K
7?YD=
?M=HqAH
/(Ne=
C=_"
I========== the6
z5Y=
`h(8=`
n\H=;
========== password
========== ism
N$=&
l/a=L)
f=C(
========== truKLdjsbJ5g7yyJ2X2R0o3a5HQJFuLk
ie)=5e
bandit9@melinda:~$
```

3.12 Level 10 to 11

Notable commands:

base64 command with -d option to decode the password. If you tried this without the -d option, you would have gotten cipherytext (a result of encryption).



```
bandit10@melinda:~$ base64 -d data.txt
The password is IFukwKGsFW8MOq3IRFqrxE1hxTNEbUPR
```

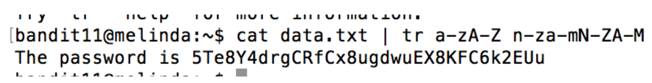
3.13 Level 11 to 12

This level is tricky, but it teaches you how to manipulate files.

Notable commands:

tr is used to translate or delete characters. The ”a-zA-Z n-za-mN-ZA-M” options are part of ”rotate by 13 places (ROT13)” that replace a letter with 13 letters after it in the alphabet. The table below shows you how ROT13 works generally.

Input	ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
Output	NOPQRSTUVWXYZABCDEFGHIJKLMnopqrstuvwxyzabcdefghijklm



```
bandit11@melinda:~$ cat data.txt | tr a-zA-Z n-za-mN-ZA-M
The password is 5Te8Y4drgCRfCx8ugdwuEX8KFC6k2EUu
```

3.14 Level 12 to 13

This was a tedious level. The general idea behind this level was that you had to repeatedly decompress the given hexdump file. This meant that you had to figure out what the type of file compression you had, give the correct extension for said file type, and decompress. The answer is shown partially below. You had to keep doing this three step process until you reached a file that contained ASCII text (for our purposes, it is human-readable).

Notable commands:

xxd un-makes a hexdump in this case.

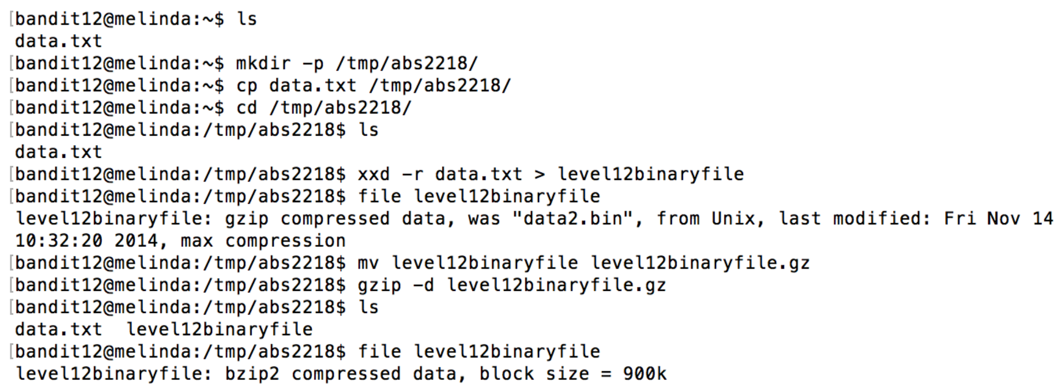
file will tell you what type of file you have.

mv is technically the move command. But if you don't give a path, you can use this command to rename a file. When you specify a single source file and the target is not a directory but just another file with a different name, mv just renames the file.

gzip -d gunzip with the -d flag can decompress files that have the .gz extension.

There were many other file types you had to decompress to beat this level. They included:

- **gzip -d**: decompress .gz files
- **bzip2 -d**: decompress .bz2 files
- **tar -xvf**: extract files from an archive



```
[bandit12@melinda:~$ ls
 data.txt
[bandit12@melinda:~$ mkdir -p /tmp/abs2218/
[bandit12@melinda:~$ cp data.txt /tmp/abs2218/
[bandit12@melinda:~$ cd /tmp/abs2218/
[bandit12@melinda:/tmp/abs2218$ ls
 data.txt
[bandit12@melinda:/tmp/abs2218$ xxd -r data.txt > level12binaryfile
[bandit12@melinda:/tmp/abs2218$ file level12binaryfile
 level12binaryfile: gzip compressed data, was "data2.bin", from Unix, last modified: Fri Nov 14
 10:32:20 2014, max compression
[bandit12@melinda:/tmp/abs2218$ mv level12binaryfile level12binaryfile.gz
[bandit12@melinda:/tmp/abs2218$ gzip -d level12binaryfile.gz
[bandit12@melinda:/tmp/abs2218$ ls
 data.txt  level12binaryfile
[bandit12@melinda:/tmp/abs2218$ file level12binaryfile
 level12binaryfile: bzip2 compressed data, block size = 900k
```

The full list of commands is not shown, but the password is:
8ZjyCRiBWFYkneahHwxCv3wb2a1ORpYL