

CPRE 431

M03 HW

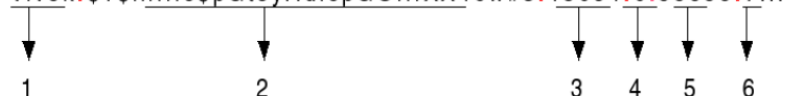
Assignments will be submitted in PDF format via Canvas.

Please submit your homework online through Canvas. Late homework will not be accepted.

Please ensure that you support all your answers with the correct screenshots showing your solutions.

1. In this “lab” problem, you will be working on a Linux Server Virtual Machine (VM). An image of this VM is available on Canvas (attached with the HW). The VM is having an administrator and 5 users, as shown in the figure below. You don’t have access to any of the users of the Server, to be able to access the Server, you will need to perform password cracking! You were given a line of password hash from (/etc/shadow) for the administrator (admin1) of the Server (attached with the HW).
 - a. Determine the used hash type of the password.

vivek:\$1\$fnfffc\$pGteyHdicpGOfffXX4ow#5:13064:0:99999:7:::



/etc/shadow file format (click to enlarge)

As with the [/etc/passwd](#), each field in the shadow file is also separated with “:” colon characters as follows:

1. **Username** : A valid account name, which exist on the system.
2. **Password** : Your encrypted password is in hash format. The password should be minimum 15-20 characters long including special characters, digits, lower case alphabetic and more. Usually password format is set to `idsalt$hashed`, The `$id` is the algorithm prefix used On GNU/Linux as follows
 1. `1` is MD5
 2. `$2a$` is Blowfish
 3. `$2y$` is Blowfish
 4. `5` is SHA-256
 5. `6` is SHA-512

From this website, we find out that the first part is the username, the first number between \$ sign is the algorithm, followed by the salt, which is followed by the password hash.

In our case, we have;

```
admin1:$6$xlS35S6$2UjEq.dUhICPw9zgDVJXcQYQp/9iLLPQt/8Zgu0uwngI5mVvB1eKQG9SnVLjm0OfkB4Jjb5VSAXGxjY4Cf5k90:18169:0:99999:7:::
```

So we know from **\$6\$**, that the type of hash is **SHA-512**.

- b. Determine the salt value of the password.

Since the format for `/etc/shadow` is `idsalt$hash`, in our case, the salt would be **xlS35S6**.

- c. Crack the passwords of all users using OpenSSL tool.

So the way to crack the password is to generate a hash using the salt from the password contents of a shadow file. To do this we write a bash script that checks each line and finds the correct sequence by brute force.

This is the bash script I wrote:

```
nzaheer@cpre587-f23-07:~/c  X + v
#!/bin/bash

filename="100k-most-used-passwords-NIST.txt"
output_hash="hashes.txt"
#password_hash='$6$xlS35S6$2UjEq.dUhICPw9zgDVJXcQYQp/9iLLPQt/8Zgu0uwngI5mVvB1eKQG9SnVLjm0OfkB4Jjb5VSAXGxjY4Cf5k90'
#password_hash='$6$2Ff.cblR$QwoNF0Ame5xy5/anjAsZVIDrZdBKZ.hZ6UIIdLU9D4DDEs30.CbRsICaVxxdQ0TG2TOHYSHDwfdsrG0WGsXVB/'
#password_hash='$6$zZKc4n0X$Rac9mB17TLEfGE/TOH0gTgRnAmaNk67ezuZo4fQA0SkuleKrrW6sum0uELlvmAeBqhF0k/jCYn2dddJCC0QzI1'
#password_hash='$6$dCPTizMy$b8Fiueet0w08JR66mI3UPg.U7ertejWDHTDCAyqbVSjhLgTu8N2/51R496408q356m.gmJSjG.u89L.3K8HH.'
#password_hash='$6$0Ptm7uW6$9cY0Hvx3S6dJBgK4ZhVq.bPHJLaMH.KV/59bsX2UYVSbp6RUit6KKFLnuoKz5L5yHMH75YZymLci19uBhV4XW.'
password_hash='$6$QpU0v3n/$Z5BKWAku6SsZMI4KStZmLR/IZuhE9Ts.cezqBca3iApKmbT/GSBC1GUHf0I0mmyt0dmqzcLHKt47idGnrmHoe0'

> "$output_hash"

while IFS= read -r line
do
    #hash=$(openssl passwd -6 -salt xlS35S6 $line)
    #hash=$(openssl passwd -6 -salt zZKc4n0X $line)
    #hash=$(openssl passwd -6 -salt dCPTizMy $line)
    #hash=$(openssl passwd -6 -salt 0Ptm7uW6 $line)
    hash=$(openssl passwd -6 -salt QpU0v3n/ $line)
    #echo "$line $hash"
    #echo "$hash" >> "$output_hash"
    echo $hash
    #echo $password_hash

    if [ "$hash" = "$password_hash" ]; then
        #output_hash="$line"
        echo "$line" >> "$output_hash"
        break
    fi
done < "$filename"
```

```
MINGW64:/c:/Users/nihaa/OneDrive - Iowa State University/#Subjects/Sem 7/Cpre 431/Labs/Lab 3
$6$xlS35S6$hcfrkPar7mgEBEQlNNF158YIiEu1jNT3UYhZNuTvmaGipwBSqK.sUOhizkCuUy8UWmd.JBhzypUXPo0NqGyaJ1
$6$xlS35S6$ySn9yWLa00FrL3P.6.b.aXXA.67RwA9Ckg15fWx.203mOw7Qj99bvwk8QuHHRQmrpPnfdwWkui1RdKk/Kp3ij0
$6$xlS35S6$/FKCxe1lU5mtXgqL5l10YeKyGXqQejeHdV6Emg059v5D3DfxAFe/swIjpJ6DNT6VctFcvIMQ0JmVni0MUMy0
$6$xlS35S6$jmTFgdawgovIMzpyJ0/7VuqGDL9YXytsnz9lHnGkqHPXuSwMsN.UtLq0YeHt8Rk/hoeKC445VHpXAQo.dv1w20
$6$xlS35S6$9AaIjav68zCqSFw.VgrZqYFE2myr1JNPfuQBm1PNyHLcLmR/mFukP9XtqzLyrWq7Q1YUotsrMGayfVh6Se12c/
$6$xlS35S6$KwvAtuMBHLKPBoQrymddjU3FC6NcH.Jos4JaCTQ/ZLKON1SDVtwXNjp69PPVjcP2lXh8I9k2/maSXweZyAnsi.
$6$xlS35S6$jkBdwPBv036vI3NdZKABCQ6LMn5XBGNQ2DTKmNXH.aBbfWty4tM60Bx6u9gg6vznayZG3MUJT3pDIa562QNE/
$6$xlS35S6$gm5ptIzq5Yg/Xrvs1pTtEMkzF/2Q.vnS032q04LsUyEwI4zsHAVknr5BoJ/ATcvFkrefPh7b4Yf6Xi2SHf14z/
$6$xlS35S6$ygc4qJks2lw9Z26Wuef1T/3iSME1NB0B7CQ4vmJkGeDNmAQISp4FQeKNFf4Ta.AL0oQe4PPIVFcTDY73gp2Dx.
$6$xlS35S6$7Ue4Cnk3/TW5Q6t7fu5eS5Kkc2QbJXCWu67fWvcZd.cBNMyZUwIXwJIwum9q59U1c42as.g6YosXYu5hk0m0.
$6$xlS35S6$0Jn2VJSKlihrOysSkQsSN9mmhJi1KUmmfalaJm00IKncDQBQ6g2BwJp4v137kK/gToFL.oRg41oH80LbUquTf.
$6$xlS35S6$tMRm5/GFW/G2ICdNVj8r2U1v.C7lAmn3mJd/bpew40e36/lrGbxzMoruchVxrKDFg.OLQ1CoglxSbrfEitN/R.
$6$xlS35S6$yARY8lNMqWcJhknyZdZFR14RlswTziwWZ0rUDVfFtbEXlEckIzIIY4l1pRs1GF3paCp06UrJ20e4E86nJEZY0
$6$xlS35S6$j9G4jjGiTNix/q1Cm.nnxuy1s6X.THVSe1y.AiBJwKMGsXQ2.ed5RtCd1rwjXgm3MBbd7pOE0.s0QGVXRpkPm/
$6$xlS35S6$/K.Bm1TLSSrHNYq4VL5vZNk7qsqAHUyJn.nkZYgzmsX08uClMnZhxR090B9zXp4BZFBshdT/yDnVrOs8tq03/
$6$xlS35S6$heaIpD9A7SKEVlgwTEwrI8zXbsok003CfNjL4av0WpflcBM1baUxj1a7dLPK0cc1UBWhu6DU00vqx.p/Gop3U/
$6$xlS35S6$GtUS43hBwMLXq78LgFwYxeDUyWqzVPP46m8x0jokmRASEgoSLtGgAMzvKYr8MDd.xiu7yAv/bEouq4FE8e8.
$6$xlS35S6$Vv.4qh5Ba.rKMHjy4qQLroPSFX2I.bgh/RDnfiwd.1ItI0fIK95rfzvKmoEBdgPwcz06fm4NzN.f5p13KzQik/
$6$xlS35S6$CESje1Rm.H83UeSNa3YqB8Arv1vb5Ycj3Rmh6Euquz5o7FP0f2ICvrSosNZiTe16FlxG2mtxz1dHN57EebqD91
$6$xlS35S6$uuyCwHqL3Jj/55pRPSJ3Tq9tGgdJmZp7LiKRRZamVncmzeEyHRQFpG1.SDBwozZWmoP01J9FYWwu8x7gLekV0
$6$xlS35S6$nwJh7wv7VL0N7N9lmoChk.1J05kA7S0cVHHjKCQdow3xgfc2J6kppqDCvJZMIbe08Bp0GIXSNDZQ57zBAPEhR/
```

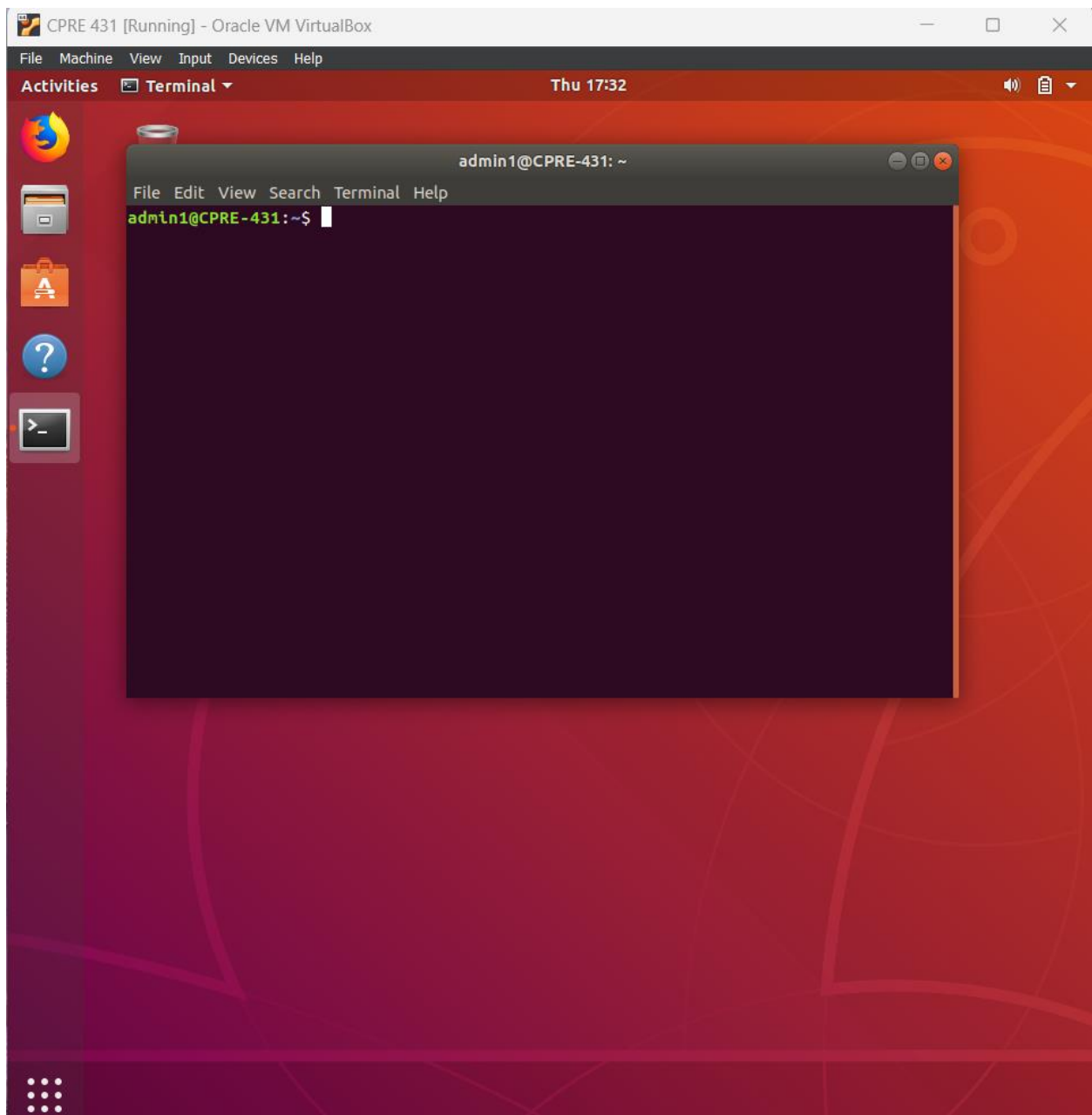
This is a snippet of the code running and inspecting the hash of each line.

I had been running it for more than 10 minutes, so I decided to use the GPU servers (cpre587). This ended up giving me a password after matching the hash to the line from the text file '100-most-used-password-NIST.txt'.

```
$6$xlS35S6$2UjEq.dUhICPw9zgDVJXcQYQp/9iLLPQt/8Zgu0uwngI5mVvB1eKQG9SnVLjm00f
kB4Jjb5VSAXGXjY4Cf5k90
[nzaheer@cpre587-f23-07 cpre431]$ ^C
[nzaheer@cpre587-f23-07 cpre431]$ ls
100k-most-used-passwords-NIST.txt  hash_finder.sh  hashes.txt
[nzaheer@cpre587-f23-07 cpre431]$ cat hashes.txt
P@ssw0rd
```

Above is the output that I got after partially running my script.

*The password is: **P@ssw0rd***



This is a screenshot of the unlocked admin1 profile.

Hints:

- It would be useful if you search for Linux shadow file password format.
- You must use the latest OpenSSL version 1.1.1x for this problem. It would be helpful to read about creating Linux password hashes using OpenSSL.
- You can use a password list for your cracking. There is a password list of most used 100K passwords, according to NIST attached with the HW.
- You will need to write some code to iterate through the password list (feel free to use any language you prefer).
- After cracking the password, ensure that you can access the Server.



2. The given Server users are managed as follows:

- There are 5 users: user1, user2, ..., user5. The first four users are staff members, but user5 is an external consultant.
 - User1 and user2 are programmers only, user4 is a manager only, and user3 is both programmer and manager.
- a. You are now the administrator of the server, and you are responsible for managing users and groups. Create 3 groups named “allstaff”, “prog” (short for programmers) and “mgmt” (short for management). Add users to the corresponding groups. List the users and groups to ensure the correct previous setup of system users and groups. Explain with screenshots how you applied this.

To make new group, we call the groupadd function.

```
admin1@CPRE-431:~$ sudo groupadd prog
admin1@CPRE-431:~$ sudo groupadd mgmt
admin1@CPRE-431:~$ cat /etc/group
```

Below is a snippet of /etc/group which shows new added groups

```
allstaff:x:1006:
prog:x:1007:
mgmt:x:1008:
admin1@CPRE-431:~$
```

We then add users to existing groups by using the usermod -aG command.

```
admin1@CPRE-431:~$ groups user1
user1 : user1 allstaff
admin1@CPRE-431:~$ groups user2
user2 : user2
admin1@CPRE-431:~$ sudo usermod -aG allstaff user2
admin1@CPRE-431:~$ sudo usermod -aG allstaff user3
admin1@CPRE-431:~$ sudo usermod -aG allstaff user4
admin1@CPRE-431:~$ groups user2
user2 : user2 allstaff
admin1@CPRE-431:~$
```

Finally after adding all users into their respective groups, we analyze the /etc/group file, and this is the result we get

```
allstaff:x:1006:user1,user2,user3,user4
prog:x:1007:user1,user2,user3
mgmt:x:1008:user4,user3
admin1@CPRE-431:~$ cat /etc/group
```

- Inside their home directory, each programmer has a directory called code and a directory called documentation. Inside the code directory, there is a file called source_code.txt, as well as one application called myapp.exe. Inside the documentation directory, there is a file called notes.txt.
 - Each manager has a directory called finance (for financial information) including a confidential business.txt file.
 - Each user also has a file called schedule.txt in their home directory.
- b. Configure file access controls so that it explicitly applies only the following:
- All users can view each other's schedules, but not other files in their home directory (except for as stated in the following).
 - All staff can view files in the documentation directory of other staff.
 - Programmers can view and edit each other's source code files, create new files in any code directory, as well as run each other's myapp.exe files.
 - Financial information (in the finance directory) is only viewable by the manager that owns the files, not by any other user.

Hint

- Take screenshots of setting the access and listing it using (ls -l). Also, take screenshots of testing that the access control works by logging in as each user and checking they can(not) access the specified files/directories.
- Use only the basic Linux permissions. Do NOT use advanced permissions such as setfacl or getfacl.
- Use the ["Introduction to Linux" Page](#) to help you in the needed commands for this Homework.

*Below are the hash for each user found in the shadow file, **"/etc/shadow"**.*

```
user1:$6$2Ff.cblr$QwoNFOAme5xy5/anjAsZVIDrZdBKZ.hZ6UIIdLU9D4DDEs30.CbRsICaVxxdQ0
TG2TOHYSHDwfdsrG0WGsXVB/:18169:0:99999:7:::
user2:$6$zZKc4nOX$Rac9mB17TLeFgE/TOH0gTgRnAmaNk67ezuZo4fQA0SkuLEKrrW6sum0uELLvMA
eBqhf0k/jCYn2dddJCC0QzI1:18169:0:99999:7:::
user3:$6$dCPTizMy$b8Fieet0w08JR66mI3UPg.U7ertejWDHTDCAyqbVSjhhLgTu8N2/51R496408
q356m.gmJSjG.u89L.3K8HH.:18169:0:99999:7:::
user4:$6$0Ptm7uW6$9cYOHvx3S6dJBgK4ZhVq.bPHJlaMH.KV/59bsX2UYVSBp6RUit6KKFLnuoKz5L
5yHMH75YZmLc1l9uBhV4XW.:18169:0:99999:7:::
user5:$6$QpU0v3n/$Z5BKWAKu6SsZMI4KstZmlR/IZuhE9Ts.cezqBca3iApKmbT/GSBC1GUHF0I0mm
yt0dmqzclHKT47idGnpmHoe0:18169:0:99999:7:::
admin1@CPRE-431:~$
```

password output with new hash and new salt:P@

user1:

**\$6\$2Ff.cblr\$QwoNFOAme5xy5/anjAsZVIDrZdBKZ.hZ6UIIdLU9D4DDEs30.CbRsICaVxxdQ0TG
2TOHYSHDwfdsrG0WGsXVB/:18169:0:99999:7:::**

```
[nzaheer@cpre587-f23-07 cpre431]$ cat hashes.txt  
trustno1
```

user2:

*\$6\$zZKc4nOX\$Rac9mB17TLeFgE/TOH0gTgRnAmaNk67ezuZo4fQAOSkulEKrrW6sum0uEl
LvmAeBqhf0k/jCYn2dddJCC0QzI1:18169:0:99999:7:::*

```
[nzaheer@cpre587-f23-07 cpre431]$ cat hashes.txt  
basketball
```

user3:

*\$6\$dCPTizMy\$b8Fiueet0w08JR66mI3UPg.U7ertejWDHTDCAyqbVSjhhLgTu8N2/51R496408
q356m.gmJSjG.u89L.3K8HH.:18169:0:99999:7:::*

```
[nzaheer@cpre587-f23-07 cpre431]$ cat hashes.txt  
pokemon
```

user4:

*\$6\$0Ptm7uW6\$9cYOHvx3S6dJBgK4ZhVq.bPHJlaMH.KV/59bsX2UYVSBp6RUit6KKFLnuoKz5L
5yHMH75YZymLcil9uBhV4XW.:18169:0:99999:7:::*

```
[nzaheer@cpre587-f23-07 cpre431]$ cat hashes.txt  
batman
```

user5:

*\$6\$QpU0v3n/\$Z5BKWAKu6SsZMI4KStZmlR/lZuhE9Ts.cezqBca3iApKmbT/GSBC1GUHf0I0m
mytOdmqzclHkT47idGnpmHoe0:18169:0:99999:7:::*

```
[nzaheer@cpre587-f23-07 cpre431]$ cat hashes.txt  
qwerty12345
```

Note: Common Testing can be done using User3 and User5, since user3 is a part of all groups and user5 is a part of no groups.

USER 1:

sudo chmod +r schedule.txt

sudo chgrp allstaff documentation

sudo g+rx documentation

sudo chgrp allstaff documentation/notes.txt

sudo g+r documentation/notes.txt

sudo chgrp prog code

`sudo chmod g+rx code`

`sudo chgrp prog code/source_code.txt`

`sudo chmod g+rw code/source_code.txt`

`sudo chgrp prog code/myapp.exe`

`sudo chmod 750 code/myapp.exe`

```
admin1@CPRE-431:/home/user1/documentation$ sudo chgrp allstaff notes.txt
admin1@CPRE-431:/home/user1/documentation$ ls -l
total 0
-rw-r----- 1 user1 allstaff 0 Sep 30 2019 notes.txt
admin1@CPRE-431:/home/user1/documentation$
```

According to permission set, notes.txt of user1 under documentation directory can only be read by users in group allstaff, which means that user5 won't be able to read it.

```
user1@CPRE-431:~$ ls -l
total 40
drwxrwx--- 2 user1 prog 4096 Sep 30 2019 code
drw----- 2 user1 user1 4096 Sep 30 2019 Desktop
drwxr-x--- 2 user1 allstaff 4096 Sep 22 15:07 documentation
drw----- 2 user1 user1 4096 Sep 30 2019 Documents
drw----- 2 user1 user1 4096 Sep 30 2019 Downloads
drw----- 2 user1 user1 4096 Sep 30 2019 Music
drw----- 2 user1 user1 4096 Sep 30 2019 Pictures
drw----- 2 user1 user1 4096 Sep 30 2019 Public
-rw-r--r-- 1 user1 user1 0 Sep 30 2019 schedule.txt
drw----- 2 user1 user1 4096 Sep 30 2019 Templates
drw----- 2 user1 user1 4096 Sep 30 2019 Videos
user1@CPRE-431:~$ ls -l documentation/
total 0
-rw-r----- 1 user1 allstaff 0 Sep 30 2019 notes.txt
user1@CPRE-431:~$ ls -l code/
total 0
-rw--x--- 1 user1 prog 0 Sep 30 2019 myapp.exe
-rw-rw--- 1 user1 prog 0 Sep 30 2019 source_code.txt
user1@CPRE-431:~$
```

```
user1@CPRE-431:~/code$ ./myapp.exe
user1@CPRE-431:~/code$ ls -l
total 0
-rwx--x--- 1 user1 prog 0 Sep 30 2019 myapp.exe
-rw-rw--- 1 user1 prog 0 Sep 30 2019 source_code.txt
user1@CPRE-431:~/code$ chmod g+r myapp.exe
user1@CPRE-431:~/code$ ls -l
total 0
-rwxr-x--- 1 user1 prog 0 Sep 30 2019 myapp.exe
-rw-rw--- 1 user1 prog 0 Sep 30 2019 source_code.txt
```

I set read permissions also for myapp.exe since it wasn't executing without it.


```
user3@CPRE-431:/home/user1/code$ ./myapp.exe
bash: ./myapp.exe: Permission denied
user3@CPRE-431:/home/user1/code$ ./myapp.exe
user3@CPRE-431:/home/user1/code$
```

Testing:

```
user3@CPRE-431:/home/user1$ cd documentation/
user3@CPRE-431:/home/user1/documentation$ cat notes.txt
user3@CPRE-431:/home/user1/documentation$ cd ../code/
user3@CPRE-431:/home/user1/code$ ./myapp.exe
user3@CPRE-431:/home/user1/code$ cat source_code.txt
user3@CPRE-431:/home/user1/code$
```

```
user5@CPRE-431:/home/user1$ cat schedule.txt
user5@CPRE-431:/home/user1$ cd code/
bash: cd: code/: Permission denied
user5@CPRE-431:/home/user1$ cd documentation/
bash: cd: documentation/: Permission denied
user5@CPRE-431:/home/user1$ cd Desktop/
bash: cd: Desktop/: Permission denied
user5@CPRE-431:/home/user1$
```

From the snippet of code, we see that the commands ran from user3 for specific files with group permission, run without any issue. Meanwhile, since user5 is not part of allstaff, the only file accessible is schedule.txt

USER 2:

```
sudo chmod +r schedule.txt

sudo chgrp allstaff documentation

sudo g+rx documentation

sudo chgrp allstaff documentation/notes.txt

sudo g+r documentation/notes.txt

sudo chgrp prog code

sudo chmod g+rw code

sudo chgrp prog code/source_code.txt

sudo chmod g+rw code/source_code.txt

sudo chgrp prog code/myapp.exe

sudo chmod 750 code/myapp.exe
```

```

user2@CPRE-431:~$ ls -l
total 40
drwxrwx--- 2 user2 prog      4096 Sep 30  2019 code
drwx----- 2 user2 user2    4096 Sep 30  2019 Desktop
drwxr-x--- 2 user2 allstaff  4096 Sep 30  2019 documentation
drwx----- 2 user2 user2    4096 Sep 30  2019 Documents
drwx----- 2 user2 user2    4096 Sep 30  2019 Downloads
drwx----- 2 user2 user2    4096 Sep 30  2019 Music
drwx----- 2 user2 user2    4096 Sep 30  2019 Pictures
drwx----- 2 user2 user2    4096 Sep 30  2019 Public
-rw-r--r-- 1 user2 user2       0 Sep 30  2019 schedule.txt
drwx----- 2 user2 user2    4096 Sep 30  2019 Templates
drwx----- 2 user2 user2    4096 Sep 30  2019 Videos
user2@CPRE-431:~$ ls -l documentation/
total 0
-rw-r----- 1 user2 allstaff 0 Sep 30  2019 notes.txt
user2@CPRE-431:~$ ls -l code/
total 0
-rwxr-x--- 1 user2 prog 0 Sep 30  2019 myapp.exe
-rw-rw---- 1 user2 prog 0 Sep 30  2019 source_code.txt
user2@CPRE-431:~$

```

Testing:

```

user5@CPRE-431:/home/user2$ cat schedule.txt
user5@CPRE-431:/home/user2$ cd code/
bash: cd: code/: Permission denied
user5@CPRE-431:/home/user2$ cd documentation/
bash: cd: documentation/: Permission denied
user5@CPRE-431:/home/user2$ cd Videos/
bash: cd: Videos/: Permission denied
user5@CPRE-431:/home/user2$

```

```

user1@CPRE-431:/home/user2$ cat schedule.txt
user1@CPRE-431:/home/user2$ cd documentation/
user1@CPRE-431:/home/user2/documentation$ cat notes.txt
user1@CPRE-431:/home/user2/documentation$ cd ../code/
user1@CPRE-431:/home/user2/code$ cat source_code.txt
user1@CPRE-431:/home/user2/code$ ./myapp.exe
user1@CPRE-431:/home/user2/code$

```

Similar to what we saw in User1's testing, this has the same permissions and are part of the same groups. Above is the execution as we expected.

USER 3:

```
sudo chmod +r schedule.txt
```

```
sudo chgrp allstaff documentation
```

```
sudo g+rx documentation
```

```
sudo chgrp allstaff documentation/notes.txt
```

sudo g+r documentation/notes.txt

sudo chgrp prog code

sudo chmod g+rw code

sudo chgrp prog code/source_code.txt

sudo chmod g+rw code/source_code.txt

sudo chgrp prog code/myapp.exe

sudo chmod 750 code/myapp.exe

sudo chmod 700 finance

sudo chmod 400 finance/business.txt

```
user3@CPRE-431:~$ ls -l
total 44
drwxrwx--- 2 user3 prog      4096 Sep 30  2019 code
drwx----- 2 user3 user3    4096 Sep 30  2019 Desktop
drwxr-x--- 2 user3 allstaff  4096 Sep 30  2019 documentation
drwx----- 2 user3 user3    4096 Sep 30  2019 Documents
drwx----- 2 user3 user3    4096 Sep 30  2019 Downloads
drwx----- 2 user3 user3    4096 Sep 24 19:58 finance
drwx----- 2 user3 user3    4096 Sep 30  2019 Music
drwx----- 2 user3 user3    4096 Sep 30  2019 Pictures
drwx----- 2 user3 user3    4096 Sep 30  2019 Public
-rw-r--r-- 1 user3 user3       0 Sep 30  2019 schedule.txt
drwx----- 2 user3 user3    4096 Sep 30  2019 Templates
drwx----- 2 user3 user3    4096 Sep 30  2019 Videos
user3@CPRE-431:~$ ls -l code/
total 0
-rwxr-x--- 1 user3 prog 0 Sep 30  2019 myapp.exe
-rw-rw---- 1 user3 prog 0 Sep 30  2019 source_code.txt
user3@CPRE-431:~$ ls -l documentation/
total 0
-rw-r----- 1 user3 allstaff 0 Sep 30  2019 notes.txt
user3@CPRE-431:~$ ls -l finance/
total 0
-r----- 1 user3 user3 0 Sep 30  2019 business.txt
```

Testing:

```
user4@CPRE-431:/home/user3$ cd documentation/
user4@CPRE-431:/home/user3/documentation$ cat notes.txt
user4@CPRE-431:/home/user3/documentation$ cd ../code/
bash: cd: ../code/: Permission denied
user4@CPRE-431:/home/user3/documentation$ cd ../finance/
bash: cd: ../finance/: Permission denied
user4@CPRE-431:/home/user3/documentation$ su user2
Password:
user2@CPRE-431:/home/user3/documentation$ cd ../code/
user2@CPRE-431:/home/user3/code$ ./myapp.exe
user2@CPRE-431:/home/user3/code$
```

Above we see that user4, who is in the mgmt group trying to access files. Documentation works. Code and finance directories are denied of permission even though user4 is in the mgmt group.

We go further by changing users and testing with a programmers account, we see that the programmer has access to the code directory and can read write and execute in the code directory. The permission was set this way so for cd, ls and writing new files.

USER 4:

```
sudo chmod +r schedule.txt
```

```
sudo chmod 700 finance
```

```
sudo chmod 400 finance/business.txt
```

```
user4@CPRE-431:~$ ls -l
total 36
drwx----- 2 user4 user4 4096 Sep 30 2019 Desktop
drwx----- 2 user4 user4 4096 Sep 30 2019 Documents
drwx----- 2 user4 user4 4096 Sep 30 2019 Downloads
drwx----- 2 user4 user4 4096 Sep 30 2019 finance
drwx----- 2 user4 user4 4096 Sep 30 2019 Music
drwx----- 2 user4 user4 4096 Sep 30 2019 Pictures
drwx----- 2 user4 user4 4096 Sep 30 2019 Public
-rw-r--r-- 1 user4 user4 0 Sep 30 2019 schedule.txt
drwx----- 2 user4 user4 4096 Sep 30 2019 Templates
drwx----- 2 user4 user4 4096 Sep 30 2019 Videos
user4@CPRE-431:~$ ls -l finance/
total 0
-r----- 1 user4 user4 0 Sep 30 2019 business.txt
user4@CPRE-431:~$ cat finance/business.txt
user4@CPRE-431:~$
```

```
user3@CPRE-431:/home/user4$ cd finance/
bash: cd: finance/: Permission denied
```

Above we see that, the finance directory can only be opened by the user and not even anyone from the same group (user4).

USER 5:

```
user5@CPRE-431:~$ ls -l
total 32
drwx----- 2 user5 user5 4096 Sep 30 2019 Desktop
drwx----- 2 user5 user5 4096 Sep 30 2019 Documents
drwx----- 2 user5 user5 4096 Sep 30 2019 Downloads
drwx----- 2 user5 user5 4096 Sep 30 2019 Music
drwx----- 2 user5 user5 4096 Sep 30 2019 Pictures
drwx----- 2 user5 user5 4096 Sep 30 2019 Public
drwx----- 2 user5 user5 4096 Sep 30 2019 Templates
drwx----- 2 user5 user5 4096 Sep 30 2019 Videos
user5@CPRE-431:~$
```

```

user5@CPRE-431:/home/user3$ cd code/
bash: cd: code/: Permission denied
user5@CPRE-431:/home/user3$ cd documentation/
bash: cd: documentation/: Permission denied
user5@CPRE-431:/home/user3$ cd Pictures/
bash: cd: Pictures/: Permission denied
user5@CPRE-431:/home/user3$ cat schedule.txt
user5@CPRE-431:/home/user3$

```

Above, we see that, user5 only has permission to see schedule.txt and no other file or directory.

Additional Tests:

We test users of the prog group being able to write and edit files in the code directory.

```

user1@CPRE-431:/home/user3/code$ ls -l
total 0
-rwxr-x--- 1 user3 prog 0 Sep 30  2019 myapp.exe
-rw-rw---- 1 user3 prog 0 Sep 30  2019 source_code.txt

```

As we see, only source_code.txt is editbale by users of the same group. This means that myapp.exe can only be executed, but not edited.



```

GNU nano 2.9.3 myapp.exe Modified
ww
[ Error writing myapp.exe: Permission denied ]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line

user1@CPRE-431:/home/user3/code$ nano source_code.txt
user1@CPRE-431:/home/user3/code$ cat source_code.txt
Testing to see if it's writable
user1@CPRE-431:/home/user3/code$

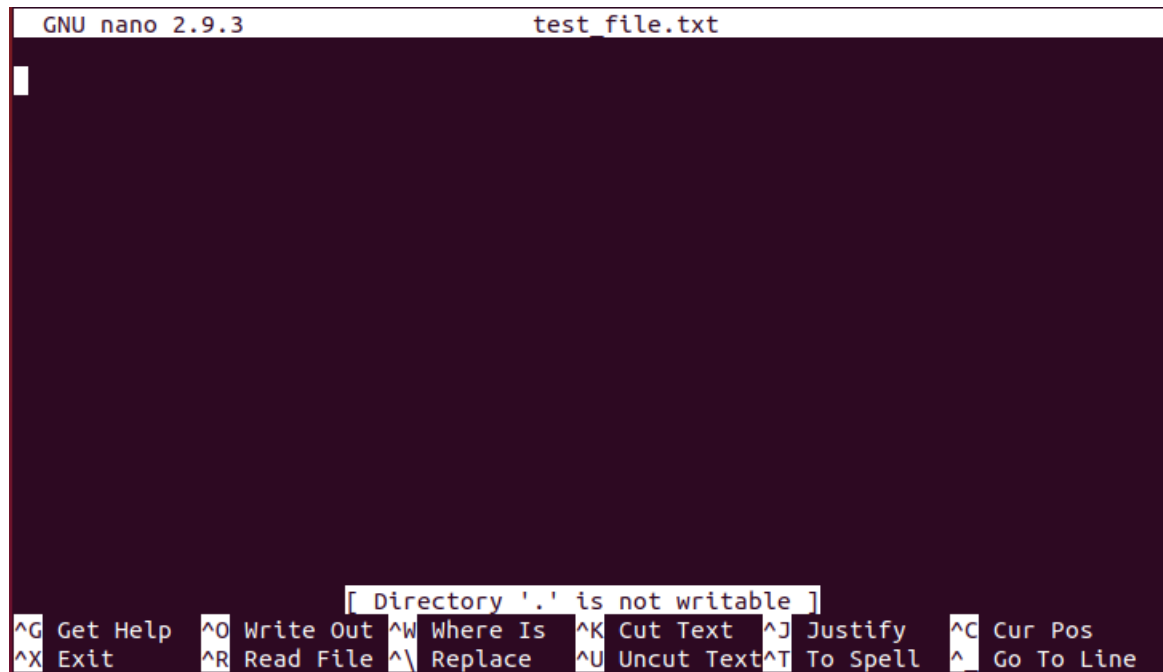
```

Above we see that source_code.txt was editable by user1.

Another condition was that users of prog group would be able to create new files only in the code directory and no where else. Below shows the implementation that proves the correctness.

```
user1@CPRE-431:/home/user3/code$ nano test_file.txt
user1@CPRE-431:/home/user3/code$ ls -l
total 8
-rwxr-x--- 1 user3 prog  0 Sep 30  2019 myapp.exe
-rw-rw---- 1 user3 prog 32 Sep 24 21:19 source_code.txt
-rw----- 1 user1 user1  8 Sep 24 21:20 test_file.txt
user1@CPRE-431:/home/user3/code$
```

Below is test_file.txt that was attempted to be created in the documentation directory.



```
GNU nano 2.9.3 test_file.txt

[ Directory '.' is not writable ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace  ^U Uncut Text ^T To Spell  ^ Go To Line
```

Conclusion:

All the tests and screenshots proves that everything works as expected.