Lab 04 – Access Control Lists

Raymond Ng: JQG999
IS 3033-CY1 – Summer 2022
June 30, 2022

INTRODUCTION

The purpose of this lab was to allow users to explore the use of access control lists (ACL). Moreover, it allowed user to experiment with read(`r`), write(`w`), and execute(`x`) permissions using the `getfacl` and `setfacl` commands. Further, it allowed users to experiment with password permissions in a controlled environment via virtual machine.

PROCEDURE

4.3 Reviewing Existing File Permissions

```
[alice@acl ~]$ cd /shared_data
[alice@acl shared_data]$ ls -l
total 24
-rw-rw----+ 1 root  root    13 Jan 27  2020 accounting.txt
drwxr-xr-x  1 alice alice 4096 Jan 27  2020 alice
drwxr-xr-x  1 bob   bob   4096 Jan 27  2020 bob
[alice@acl shared_data]$ cat accounting.txt
some numbers
```

Followed initial instructions of the lab by logging in as three different users, *bob*, *alice*, and *harry*. In *Alice*, executed `cd /shared_data` and listed contents in that directory. Executed `cat accounting.txt` to display the contents of the file. From interpreting the read(`r`), write(`w`), execute(`x`) access controls, as the relate to user, group, and other respectively, both *Alice* and *Bob* should be able to view (or read(`r`)) `accounting.txt`. Further, when I executed `cat accounting.txt` in *Alice* terminal it listed the current content of `accounting.txt`, confirming that *Alice* can view the content.

```
[alice@acl shared_data]$ getfacl accounting.txt
# file: accounting.txt
# owner: root
# group: root
user::rw-
user:alice:r--
user:harry:rw-
group::r--
mask::rw-
other::---
```

Executing `getfacl accounting.txt`, I noticed that Harry has a write(w) permission.

```
[harry@acl ~]$ echo "more stuff" >> /shared_data/accounting.txt
[harry@acl ~]$
```

Executed `echo "more stuff" >> /shared_data/accounting.txt` in *Harry*'s terminal. *Permissions denied* did not result, validating *Harry*'s write(`w`) permission.

```
[alice@acl shared_data]$ echo "confirming alice has access" >> /shared_data/accounting.txt
-bash: /shared_data/accounting.txt: Permission denied
```

To confirm if *Alice* lacks write(w) access to modify `accounting.txt`. I executed `echo "confirming alice has access" >> /shared_data/accounting.txt`. A permission denied resulted indicating *Alice* lacks access.

4.2 Set an ACL on a Single File

```
[bob@acl ~]$ setfacl /shared_data/bob/bobstuff.txt
Usage: setfacl [-bkndRLP] { -m|-M|-x|-X ... } file ...
Try `setfacl --help' for more information.
[bob@acl ~]$ setfacl -m "u:alice:permissions" /shared_data/bob/bobstuff.txt
setfacl: Option -m: Invalid argument near character 9
[bob@acl ~]$ setfacl -m "u:alice:r--" /shared_data/bob/bobstuff.txt
```

I ran into issues executing step 4.2 so I referenced *https://wiki.archlinux.org/title/Access_Control_Lists* to remedy the issues. After reviewing the use of `setfacl`, thoroughly, I executed `setfacl -m u:alice:r—" /shared_data/bob/bobstuff.txt` to allow read(`r`) permission.

```
[alice@acl shared_data]$ cd /shared_data/bob/
[alice@acl bob]$ cat bobstuff.txt
bob's stuff
```

Executed `cat bobstuff.txt` in *Alice* terminal to verify/validate read(`r`) permission. Note: I ran into an error message previously when attempting to execute the aforementioned. I simply forgot to `cd` change directory.

```
[harry@acl ~]$ cd /shared_data/bob/
[harry@acl bob]$ cat bobstuff.txt
cat: bobstuff.txt: Permission denied
```

In *Harry* terminal, to validate if *Harry* had permission to read(`r`) `bobstuff.txt`. I changed directories first and then executed `cat bobstuff.txt`. Output indicated *Permission denied*, *Harry* does not have permission to read(`r`) `bobstuff.txt`.

4.3 Set a Default ACL for a Directory

```
[alice@acl alice]$ touch alicefile.txt
[alice@acl alice]$ echo "alice content" >> /shared_data/alicefile.txt
-bash: /shared_data/alicefile.txt: Permission denied
[alice@acl alice]$ echo "alice content" >> /shared_data/alice/alicefile.txt
[alice@acl alice]$ cat alicefile.txt
alice content
[alice@acl alice]$ setfacl -m "u:bob:r--" /shared_data/alice/
[alice@acl alice]$ setfacl -m "u:bob:r-x" /shared_data/alice/
[alice@acl alice]$ setfacl -m "u:bob:r--" /shared_data/alice/alicefile.txt
[alice@acl alice]$ touch alicefile2.txt
[alice@acl alice]$ setfacl -m "u:bob:r--" /shared_data/alice/alicefile2.txt
```

```
[bob@acl ~]$ cd /shared_data/alice/
-bash: cd: /shared_data/alice/: Permission denied
[bob@acl ~]$ cd /shared_data/alice/
[bob@acl alice]$ cat alicefile.txt
cat: alicefile.txt: Permission denied
[bob@acl alice]$ cat alicefile.txt
alice content
[bob@acl alice]$ cat alicefile2.txt
[bob@acl alice]$
```

In step 4.3, I created `alicefile.txt` in alice directory and added content in the new file, *alice content*. Gave access to *Bob* by executing `setfacl -m "u:bob:r-x" /shared_data/alice/` to allow *Bob* access into *Alice*'s directory. Then gave *Bob* access to alicefile.txt by executing `setfacl -m "u:bob:r--" /shared_data/alicefile.txt`. To ensure *Bob* had access I went into *Bob* terminal, changed directories into *Alice* and executed to `cat alicefile.txt` and it depicted its contents. Further, I created a new file, `alicefile2.txt`, gave read(`r`) permission to *Bob*, and tested to see if *Bob* could read the file. Based on the permissions I set, *Bob* was able to read(`r`) `alicefile2.txt`. I don't input any content into `alicefile2.txt`; therefore, working as expected.

4.4 Password Permissions

To change passwords, we would simply use `/usr/bin/passwd` in the root directory of each terminal. Before I executed any of the steps in Step 4.4 I executed `cd ~` to go back to the root directory of each terminal

```
[harry@acl ~]$ ls -l /etc/shadow /usr/bin/passwd
---------- 1 root root   945 Jan 27  2020 /etc/shadow
-rwsr-xr-x 1 root root 27832 Jun 10  2014 /usr/bin/passwd
[harry@acl ~]$ stat /usr/bin/passwd
  File: '/usr/bin/passwd'
  Size: 27832           Blocks: 56         IO Block: 4096   regular file
Device: 31h/49d Inode: 2103812     Links: 1
Access: (4755/-rwsr-xr-x)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2014-06-10 06:27:56.000000000 +0000
Modify: 2014-06-10 06:27:56.000000000 +0000
Change: 2022-01-11 18:02:35.045893998 +0000
 Birth: -
```

Executed `ls -l /etc/shadow /usr/bin/passwd` and `stat /usr/bin/passwd` to depict the file permissions in *Harry* terminal.

```
[alice@acl ~]$ ls -l /etc/shadow /usr/bin/passwd
---------- 1 root root   945 Jan 27  2020 /etc/shadow
-rwsr-xr-x 1 root root 27832 Jun 10  2014 /usr/bin/passwd
[alice@acl ~]$ stat /usr/bin/passwd
  File: '/usr/bin/passwd'
  Size: 27832           Blocks: 56         IO Block: 4096   regular file
Device: 31h/49d Inode: 2103812     Links: 1
Access: (4755/-rwsr-xr-x)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2014-06-10 06:27:56.000000000 +0000
Modify: 2014-06-10 06:27:56.000000000 +0000
Change: 2022-01-11 18:02:35.045893998 +0000
 Birth: -
```

Executed `ls -l /etc/shadow /usr/bin/passwd` and `stat /usr/bin/passwd` to depict the file permissions in *Alice* terminal.

```
[bob@acl ~]$ ls -l /etc/shadow /usr/bin/passwd
---------- 1 root root   945 Jan 27  2020 /etc/shadow
-rwsr-xr-x 1 root root 27832 Jun 10  2014 /usr/bin/passwd
[bob@acl ~]$ stat /usr/bin/passwd
  File: '/usr/bin/passwd'
  Size: 27832           Blocks: 56         IO Block: 4096   regular file
Device: 31h/49d Inode: 2103812     Links: 1
Access: (4755/-rwsr-xr-x)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2014-06-10 06:27:56.000000000 +0000
Modify: 2014-06-10 06:27:56.000000000 +0000
Change: 2022-01-11 18:02:35.045893998 +0000
 Birth: -
```

Executed `ls -l /etc/shadow /usr/bin/passwd` and `stat /usr/bin/passwd` to depict the file permissions in *Bob* terminal.

```
[harry@acl ~]$ /usr/bin/passwd
Changing password for user harry.
Changing password for harry.
(current) UNIX password:
New password:
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word
New password:
BAD PASSWORD: The password contains the user name in some form
New password:
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word
passwd: Have exhausted maximum number of retries for service
[harry@acl ~]$ /usr/bin/passwd
Changing password for user harry.
Changing password for harry.
(current) UNIX password:
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[harry@acl ~]$
```

I decided to change *Harry*'s password first. I ran into a few issues. First, was I kept trying to change the old one without entering the old password first. Figured that out. And then I ran into issues changing the password based on complexity. After a few attempts I could determine there was a moderately level of complexity to the password requirements.

```
[alice@acl ~]$ /usr/bin/passwd
Changing password for user alice.
Changing password for alice.
(current) UNIX password:
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

```
[bob@acl ~]$ /usr/bin/passwd
Changing password for user bob.
Changing password for bob.
(current) UNIX password:
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

I decided to change all three users to the same password as depicted in the table below

User	Password
bob	newpassword4terminal
alice	newpassword4terminal
harry	newpassword4terminal

CONCLUSION

The goals of this lab were to explore the use of ACLs. Additionally, it allowed users to experiment with permissions, read(`r`), write(`w`), execute(`x`) as they pertain to User, group, and Others. Further, users

were able to experiment with password permissions to secure and restrict access due to anomalous activity occurring in user directories. I would assess that these types of techniques used in this lab are practiced and implemented regularly/routinely to secure user systems in the real world.

From my rudimentary understanding of ACLs, these techniques seem to be pretty useful and likely effective to mitigate threats in the real world. Restricting access to the users maintains original users integrity and restricts access to potential malign actors.

The overall difficulty of lab was pretty simple. There were a few issues that I ran into, but resources provided by the instructor and access simple data-mining via the internet remedied the issues. I'm not really sure what other aspects of this lab could be explore more. Maybe more steps could have been included regarding password permissions.

REFERENCES

Internet Resources

https://www.cyberciti.biz/faq/understanding-etcshadow-file/

https://wiki.archlinux.org/title/Access_Control_Lists

http://linuxcommand.org/lc3_lts0090.php

Collaboration

The entirety of this lab was performed independently coupled with simple user data-mining via the internet. No additional collaboration.