Security Engineering
Assignment 2

Date – Feb 12, 2016

Implementing ACLs Using Mediated Access

- Deepak Kumar Sood
MT15013

Assumptions:

1. 4 users are hardcoded into the program with one of them being fakeroot.
2. Fakeroot is made a secondary root.
3. If there is anytime conflict between the implemented acl and unix own permission then user must use chmod command with 777 attribute to grant access to the files (depends on system configuration).
4. Permission of the owner of the file cannot be changed by anyone including root because it can lead to file with no owner.
5. Every command has a specific format that needs to be followed and can be seen when a wrong command is entered.
6. Users cannot access outside the users directories (i.e. home, slash_root or anywhere from there) using read commands.
7. Root has access for every file and folder.
8. The effective user id (euid) is changed back to the original user id (uid) before performing any operation and after checking the permissions from DAC and ACL.
9. Only the user present in the users list at home folder can access the programs.
10. If a new user is added then its entry should also be done to users.txt file inside home directory and it will not be done automatically.
11. For accessing a folder permission.acl is used to check for access.
12. For accessing a file, file's acl is used to check permission, anyone can access file if it has read permission for file even if the users has no read permission for the folder.
13. The ACL and data in a file is seperated using “::” which marks the boundary between the acl and data.
14. ACL can be seen through ./getacl and not by ./fget
15. ACL of a file is modified using ./setacl and not by ./chmod.
16. ACL for a folder can be modified using ./chmod and not by ./setacl.
17. When a directory is created, parent's permission is copied along and will be used by default.
18. When a recursive directory is created, all the created directory will use the permission of its underlying parent.
19. Whenever path is not specified the operation will be performed at fakeroot directory.

Running Procedure:

1. Goto to the directory where makefile exists, using cd command.

2. Run “make setup” for setting up the environment.
    - This command will setup 4 users including fakeroot with the use of adduser command.
    - The users are namely bill, bob, fakeroot and harry.
    - Their home directories will be set to /simple_slash/home/<username>. This is done by using the --home attribute with adduser command.
    - A group will be added name simple_slash using addgroup command.
    - Owner for home and simple_slash directories will be changed to root using chown command.
    - The fakeroot user will be made secondary root using useradd command and changing its userid and groupid to 0.
    - The users directories is given full access to each other using chmod 777 command so the unix DAC would not conflict with our program.
    - A users.txt file is created at home directory which will contain the username of all users added.
    - Permission.acl file will be created at every user's directory which will by default have rwx permission for the owning user.

3. Now run “make compile” for compiling all the commands. This will compile commands ls, create_dir, fget, fput, chmod, getacl, setacl. Also this command will set the setuid bit using chmod with u+s attribute.

4. Finally run “make clean” to remove all the binary files.

Commands List:

1. ./ls
2. ./ls <path-to-directory>
3. ./create_dir <new-directory>
4. ./create_dir <path-to-new-directory>
5. ./chmod <access-code> <username>
6. ./chmod <access-code> <username> <path-to-directory>
7. ./fget <path-to-file>
8. ./fput <filename> <text-to-write-to-file>
9. ./fput <path-to-newfile> <NULL>
10. ./setacl <file-name> <username> <access-code>
11. ./setacl <path-to-file> <username> <access-code>
12. ./getacl
13. ./getacl <filename>
14. ./getacl <path-to-file>
15. ./getacl <path-to-directory>

Examples:

1. ./ls /simple_slash/home/bill/
2. ./create_dir /simple_slash/home/bill/test
3. ./chmod 0 bill
4. ./chmod rwx bill /simple_slash/home/harry/
5. ./fput /simple_slash/home/bill/test.txt Hello! World!
6. ./fget /simple_slash/home/bill/test.txt
7. ./setacl /simple_slash/home/bill/test.txt harry rwx
8. ./getacl /simple_slash/home/bill/
9. ./getacl /simple_slash/home/bill/test.txt

Inputs used for Testing:

1. Checks have been performed on all the commands with fakeroot as original user on all other directories.
2. Tests have been performed using all commands with bill as user on all other users with or without permission.
3. Wrong inputs are used for testing.
4. Out of bound inputs are used for testing access permission.
5. Wrong access codes were used to check for acl.
6. Tried to change the owner of a file which was handled gracefully by the program.
7. Wrong usernames and path have been tried.
8. Every possibility of command on every possible permission is checked.

Errors Handled:

1. Non users access checked and handled.
2. Wrong commands entered by user handled.
3. Principle of Least Privelege implemented, only the most basic permission will be used to get the work done.
4. If command incomplete, then correct implementation of the command is displayed to user.
5. For every error, error message is displayed.
6. Users trying to access root folder handled gracefully with appropriate error message.
7. Program never crashes the computer.

Attacks Handled:

1. Access to root files restricted.
2. Attack to try to give path for accessing other user files handled.
3. Data cannot be written over acl to corrupt acl file.
4. Owner permission cannot be changed.
5. Root also cannot change owner permission.
6. Null pointer exception handled.