CS 5750 Programming Assignment #1: `get`
Due Date: *Monday, February 4, 2019 @ 8am*

In this project, you will implementing a portion of access control lists for files. For any file, you will be able to specify a list of users that have read access to that file and provide access to those files through an SUID binary.

Requirements

This project will utilize SUID programming to implement an interface to one's files. There are two subjects: USER and OWNER. Thus, you will need two accounts to test your code. OWNER is a subject that owns a set of files protected by access-control lists. USER is a subject that wishes to access OWNER's files. Here is a scenario illustrating how this system works.

1. OWNER puts a copy of the SUID binary `get` in `~owner/getbin`.
2. OWNER sets the Linux protections bits on any files he/she wishes to protect (*e.g.*, `basename.ext`) so that there is no group or world access to `basename.ext`. These files should be stored in the directory `~owner/files`.
3. OWNER creates the file `basename.ext.acl` with no group or world access in `~owner/files` and uses the format detailed below to specify a list of users that may access that file.
4. USER runs the binary `owner/getbin/get` to get a copy of any file in `~owner/files` to which USER has been granted access an access-control list (`.acl` file).

Specifying Access Control. Access to the protected file `basename.ext` is determined by the contents of the ACL file named `basename.ext.acl`. The file will contain a comma separated list of users. The file may be empty.

Function of `get`. USER attempts to get a file by executing the command

```
get <source> <destination>
```

where `get` is an SUID binary owned by OWNER. `<source>` is a full path specification of a file that must be in `~owner/files` and `<destination>` is a full-path specification of a file that must be in the home directory of USER. A successful `get` should copy the contents of `<source>` to `<destination>`. If `<destination>` exists, the user should be prompted to confirm that an overwrite is desired. If `<destination>` is overwritten, the owner and protections of the file are not changed by the write. If `<destination>` does not exist, it is created with the owner and group corresponding to USER and his default group. (See the manual page for `getpwnam().`) The file protection is set to 400.

A number of errors may occur that must prevent the `get` from succeeding. In each case, `get` should emit a meaningful error message.[1]

[1] Meaningful error messages may give useful information to a hacker. However, I need these messages to grade the assignment properly.

1. A command with improper syntax.
2. A `get` command where `<destination>` is not writable by USER. `get` should emit the error `get: <destination> not writable by` USER. Note that the file may or may not exist and existence is not required for the command to succeed.
3. `<source>` does not exist.
4. OWNER does not own `<source>`.
5. OWNER does not have read access to `<source>`.
6. There is no `<source>.acl` file.
7. `<source>.acl` is malformed.
8. `<source>.acl` is not owned by OWNER.
9. `<source>.acl` has group or world permissions set.
10. Either `<source>` or `<destination>` is a symbolic link.
11. USER does not have an entry in `<source>.acl`.
12. `<source>` is in the wrong directory or is not a full path.
13. `<destination>` is in the wrong directory or is not a full path.

If any of the above errors occur, the contents of `<source>` are NOT written to `<destination>`.

Miscellaneous. You need not worry about file locking for this assignment. The project must be coded in C or C++.

You will be graded on how well you use the secure coding principles discussed in class. **Follow these rules if you wish to receive a grade better than a D.**

Collaboration Rules

This project is to be performed individually. An individual may neither show any code nor look at the code of another individual. (This policy extends to any external resource, including code found on the web or individuals who are not enrolled in the course.)

Linux

Use Ubuntu 18.04 LTS downloaded from `ubuntu.com`. If you need help, ask the CS system administrators in C-218.

Submissions

You must prepare a `makefile` and all necessary source files so that I can simply do a `make` to build `get`. Submissions will be made via *eLearning*.