Rudimentary Sudo

Reeshabh Kumar Ranjan
2017086

1 ASSUMPTIONS

1. Basic error-detection related to unauthorised access to files due to insufficient permissions will be done by the OS itself.
2. User will provide the full path to the executable.
3. User will not misuse the program by executing binaries maliciously by switching to root (authentication is skipped).
4. User will provide space before and after the pipeline ("|") operator.
5. While using the pipeline operator, the user will put escape character, like the following.

   ```
   ./mysudo /bin/ls -l . \| /usr/bin/wc -l
   ```

6. If the user is using a pipeline operator, then the user must use only one pipeline operator.
7. For using pipeline operator, the user must put one command before and one command after the pipeline operator.
8. The whole command (including the command after the pipeline operator) will be run as the username provided (or root if the -u flag is skipped).
9. The write-permission for the command after the pipeline operator will be checked for the username requested via the -u flag or root if -u parameter is skipped.
10. User will follow the directions to use this program. (Described below).

2 Error handling

1. If the user enters a username which does not exists, the user will receive a suitable error message.
2. If the user enters a path which does not exist, the user will receive a suitable error message.
3. If the user tries to execute a file as a user having insufficient permissions, the user will get a permission denied message.
4. At the end, the euid is reset to the initial value.
5. There is a SIGINT handler. The work of the SIGINT handler is to restore the EUID of the calling user before exiting, in case the mysudo program's execution is interrupted unexpectedly.
6. For pipe operation, let say the command is the following.

   ```
   ./mysudo -u reeshabh /bin/ls -l . \| /usr/bin/wc -l
   ```

   The command will fail because the user named reeshabh does not have write permissions for the /usr/bin/wc file.
7. Any suspicious/erroneous operation will be detected and the execution of the whole program will be terminated immediately.

3 Using the mysudo executable

In order to generate the mysudo executable file, you just need to run the following command in the working directory containing the Makefile and the mysudo.c source code file.

```
make build
```

This command will do three things.

1. Compile the source file mysudo.c to generate the executable mysudo.
2. Call chown to change the ownership to root:root.
3. Call chmod to change the permissions so that it has setuid bit set and executable permissions enabled for the owner (root).

Hence, it will ask for your password for completion, because chmod and chown are called using sudo prefix.

First, change the working directory of your shell/terminal to the directory containing the mysudo executable. The command format for using the mysudo executable is as follows.

```
./mysudo -u [username] [path/to/executable --with-parameters]
```

You can include the pipeline ("|") operator with the escape character just preceding it (backslash), but please put spaces before and after it. The username can be any user on your system. To run a program, you should give the complete path of the executable. For example, for running the ls -la command in the present directory as reeshabh (sample user), the full command will be as follows (notice the dot in the end).

```
./mysudo -u reeshabh /bin/ls -la .
```

However, in case you omit the -u flag, then the command will be run by default as root. For example, the following command is a command that should successfully execute.

```
./mysudo /bin/ls . \| /usr/bin/wc - l
```

In case you enter an invalid input, you will receive a suitable error message.