Exercise 10.1: sudo

- It is very dangerous to run a **root shell** unless absolutely necessary: a single typo or other mistake can cause serious (even fatal) damage. Thus, the sensible procedure is to configure things such that single commands may be run with superuser privilege, by using the **sudo** mechanism. With **sudo** the user only needs to know their own password and never needs to know the root password.
- If you are using a distribution such as **Ubuntu**, you may not need to do this lab to get **sudo** configured properly for the course. However, you should still make sure you understand the procedure.
- To check if your system is already configured to let the user account you are using run **sudo**, just do a simple command like:

  ```
  $ sudo ls
  ```

  You should be prompted for your user password and then the command should execute. If instead, you get an error message you need to execute the following procedure.
- Launch a root shell by typing **su** and then giving the **root** password, not your user password.
- On all recent **Linux** distributions you should navigate to the `/etc/sudoers.d` subdirectory and create a file, usually with the name of the user to whom root wishes to grant **sudo** access. However, this convention is not actually necessary as **sudo** will scan all files in this directory as needed. The file can simply contain:

/etc/sudoers.d/student

```
student ALL=(ALL)  ALL
```

  if the user is `student`.
- An older practice (which certainly still works) is to add such a line at the end of the file `/etc/sudoers`. It is best to do so using the **visudo** program, which is careful about making sure you use the right syntax in your edit.

Text Editing Basics

- It may not be obvious to you how to create or edit such a file, especially if you are logged in as a normal user and then have used **su** to run as root.
- In particular, using a graphical editor such as **gedit** would require dealing with some complications because root does not own the display screen.
- If you are not yet familiar with using **vi** or **emacs** you can easily use **nano**, which is very simple to use and has information in the bottom panel about all needed commands. So you could do:

  ```
  # log in as root
  $ su
  ....
  # nano /etc/sudoers.d/student
  ```

  ```
  ....
  Ctr-X (to save file and exit)
  ```

- Or you can type at the command line:

```
cat << EOF > student
student ALL=(ALL)  ALL
EOF
```

- You probably also need to set proper permissions on the file by typing:

```
$ sudo chmod 440 /etc/sudoers.d/student
```

(Some **Linux** distributions may require `400` instead of `440` for the permissions.)

- After you have done these steps, exit the root shell by typing `exit` and then try to do `sudo ls` again.
- There are many other ways an administrator can configure **sudo**, including specifying only certain permissions for certain users, limiting searched paths etc. The `/etc/sudoers` file is very well self-documented.
- However, there is one more setting we highly recommend you do, even if your system already has **sudo** configured. Most distributions establish a different path for finding executables for normal users as compared to root users. In particular the directories `/sbin` and `/usr/sbin` are not searched, since **sudo** inherits the `PATH` of the user, not the full root user.
- Thus, in this course we would have to be constantly reminding you of the full path to many system administration utilities; any enhancement to security is probably not worth the extra typing and figuring out which directories these programs are in. Consequently, we suggest you add the following line to the `.bashrc` file in your home directory:

```
PATH=$PATH:/usr/sbin:/sbin
```

If you log out and then log in again (you do not have to reboot) this will be fully effective.