/etc/profile is executed only for login shells and the /etc/bashrc is executed for both login and non-login shells.

**If your login shell is bash, it runs several profile scripts before you get control. If /etc/profile exists, bash runs it first, along with any .sh files in /etc/profile.d. After bash runs the system scripts, it checks your home directory for the ~/.bash\_profile, ~/.bash\_login, and ~/.profile files, in that order, and runs the first one that it finds.**

**/etc/profile vs /etc/bashrc vs ~/.bash\_profile vs ~/.bashrc vs ~/.bash\_logout**

**/etc/profile is the global config file that affects all users’s environments if they use the bash shell. It’s sourced (read) every time a user performs a login shell.**

**/etc/profile is global, applies to all users.**

**~/.bash\_profile is local, applies to a specific user.**

**/etc/bashrc is global, applies to all users.**

**~/.bashrc is local, applies to a specific user.**

**/etc/profile sources all files in /etc/profile.d/\*.sh folder**

**~/.bashrc sources /etc/bashrc**

**Bash\_logout file is used to issue any commands to do before logging out. In fact, it is mostly useless since it is only used along with login shell.**

If you want to perform work any time bash exits (and whether it’s a login shell or not), use trap foo EXIT. The most convenient way to do this is to put your code in a shell function, e.g.,:

print\_goodbye () { echo Goodbye; }

trap print\_goodbye EXIT

When working with Linux, Unix, and Mac OS X, I always forget which bash config file to edit when I want to set my PATH and other environmental variables for my shell. Should you edit .bash\_profile or .bashrc in your home directory?

You can put configurations in either file, and you can create either if it doesn’t exist. But why two different files? What is the difference?

According to the [bash man page](http://linux.die.net/man/1/bash), .bash\_profile is executed for login shells, while .bashrc is executed for interactive non-login shells.

**What is a login or non-login shell?**

When you *login* (type username and password) via console, either sitting at the machine, or remotely via ssh: .bash\_profile is executed to configure your shell before the initial command prompt.

But, if you’ve already logged into your machine and open a new terminal window (xterm) inside Gnome or KDE, then .bashrc is executed before the window command prompt. .bashrc is also run when you start a new bash instance by typing /bin/bash in a terminal.

**Why two different files?**

Say, you’d like to print some lengthy diagnostic information about your machine each time you login (load average, memory usage, current users, etc). You *only* want to see it on login, so you *only* want to place this in your .bash\_profile. If you put it in your .bashrc, you’d see it every time you open a new terminal window.

**Mac OS X — an exception**

An exception to the terminal window guidelines is Mac OS X’s Terminal.app, which runs a login shell by default for each new terminal window, calling .bash\_profile instead of .bashrc. Other GUI terminal emulators may do the same, but most tend not to.

**Recommendation**

Most of the time you don’t want to maintain two separate config files for login and non-login shells — when you set a PATH, you want it to apply to both. You can fix this by sourcing .bashrc from your .bash\_profile file, then putting PATH and common settings in .bashrc.

To do this, add the following lines to .bash\_profile:

if [ -f ~/.bashrc ]; then

source ~/.bashrc

fi

Now when you login to your machine from a console .bashrc will be called.

If you tell the shell, e.g. bash, to read a file and execute the **commands** in the file, it's called sourcing. The main point is, the current process (shell) does this, not a new child process. In BASH you can use the **source command**