Administering your Linux system

Security issues

Linux is secure, or so they say. Take a Windows box, it will get hacked and compromised in no time. Linux, that is a different league.
Of course, you have to watch out with dogmatic ideas like that. Linux is inherently more secure because of the way it is built, while Windows carries with it the legacy of the DOS age. The fact that Windows platforms must maintain compatibility with their older cousins to a large degree does no good to their levels of security. But even so, a Linux box is as safe as the weakest link.That weakest link of course being you, the user!

How to prevent very bad things from happening: limit the use of and access to the *root* account.

Don't login as root

An often-shouted comment at newbies: *do not login as root unless you have to!*
It is completely unnecessary, and potentially dangerous, to run your box under the root account. Any and every function of the operating system related to desktop work is available to ordinary user accounts, and obtainig root rights is easy from a normal account using the `su` and `sudo` commands. Running as root all the time, will most certainly cause you grief some future day: either because you accidentally type a space where you should not have and run `rm -rf / tmp/*` instead of `rm -rf /tmp/*` (to give an arbitrary example stemming from own experience); or because you manage to run a program containing malicious code that uses your root permissions to compromise the system.

Wheel

The *wheel* group is a legacy from UNIX. When a server had to be maintained at a higher level than the day-to-day system administrator, root rights were often required. The 'wheel' group was used to create a pool of user accounts that were allowed to get that level of access to the server. If you weren't in the 'wheel' group, you were denied access to root. I'll show a couple of ways to use membership of 'wheel' to limit the amount of havoc you can wreck on your system.
Most modern-day Linux distro's still use this concept of grouping people to assign different levels of administrative access, but to my knowledge the 'wheel' group is not necessarily used to that purpose. I like to be old-fashioned from time to time, and so I resurrected the use of 'wheel'. Add yourself to wheel when creating your account (use 'wheel' as your primary group) or use `vigr` to edit the file `/etc/group` and put your name to the end of the line starting with wheel - like this:

```
wheel::10:root,alien
```

The `vigr` command is a safe way of editing the group file in a multi-user environment. Of course, if there's no one working on your box except yourself, you might just as well use plain `vi`.

Su

Or rather, *su* and *sudo*. The usual way to obtain root permissions is to run

```
sudo <some_command>
```

to execute a single command as root, or

```
su -
```

to open a root shell and work as root for a longer period of time. The `sudo` command requires you to enter your *own* password, whereas running `su -` requires the *root* password.
There is an easy way to prevent ordinary users from getting root access this way. The command `visudo` allows you to tailor the `/etc/sudoers` file which determines who may run what commands using `sudo`. My sudoers file looks like this:

```
# sudoers file.
# This file MUST be edited with the 'visudo' command as root.
# See the sudoers man page for the details on how to write a sudoers file.

# User privilege specification
root    ALL=(ALL) ALL
# Uncomment to allow people in group wheel to run all commands
%wheel  ALL=(ALL)       ALL
# Same thing without a password
# %wheel        ALL=(ALL)       NOPASSWD: ALL

# Samples
%users  ALL=/sbin/mount /mnt/cdrom,/sbin/umount /mnt/cdrom
# %users  localhost=/sbin/shutdown -h now
```

You will notice that user root as well as 'wheel' members are allowed to run *any* command with root privileges using `sudo`. You could opt for the alternative to allow the 'wheel' members to not even enter their own password when runniing `sudo`, but I think entering the password gives you just that little time to re-think what you're doing before it is too late 🙂

Limiting the invocation of `su` is possible too, by writing a file called `/etc/suauth`. There is a man page for that: `man suauth`. My file always looks like this (do not forget to set it accessible to root only by running `chmod 600 /etc/suauth`):

```
# sample /etc/suauth file
#
# A couple of privileged usernames may
# su to root with their own password.
#
root:alien:OWNPASS
#
# Anyone else may not su to root unless in
# group wheel. This is how BSD does things.
#
root:ALL EXCEPT GROUP wheel:DENY
```

You will notice that not being member of 'wheel' means you will not be able to use `su` to get a root shell. The command `sudo bash -l` will of course get you that shell just as easily, but the use of `sudo` is logged (or *should* be logged if you're security minded). An interesting line is "*root:alien:OWNPASS*" which allows user 'alien' to become root using his own password. This is a nice way of not having to share the sensitive root password with others and still alowing those others to run a root shell without having their actions logged.