1. **Environmental Variables and User Profiles**

**Environment variables** are a set of dynamic named values that can affect the way running processes will behave on a computer. They can be said in some sense to create the operating environment in which a process runs.

In order to view environmental variables in Linux you can use 3 commands **set**, **env**, **declare**. Variables within Linux operating system ca be identified by a $ sign that precedes their name. The most common environment variable in Linux is the PATH variable which permits a user to execute a binary file without an absolute path. Here is an example to change the $PATH:

[root@Oma.lan:/tmp]# echo $PATH

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin

[root@Oma.lan:/tmp]#

[root@Oma.lan:/tmp]# export PATH=$PATH:/tmp

[root@Oma.lan:/tmp]# myscript.sh

Hello World

Notice that $PATH represent the value of the variable with the name PATH. The command **export** is used to declare this variable as a global one, so it’s value will be the same for all shells. If we had used command **set** instead of export we would have declared this variable as a local one, so only for the current shell ( the one in which we had logged in).

In the following example we have 2 string variables. Notice that to defined variable local with set and variable global with export. Even though we moved to korn shell at a moment we could notice that variable global was still in memory.

[root@Oma.lan:/tmp]# **local="i am only here :("**

[root@Oma.lan:/tmp]# **global="i am everywhere :)"**

[root@Oma.lan:/tmp]# **set local**

[root@Oma.lan:/tmp]# **export global**

[root@Oma.lan:/tmp]# set | awk '/local/ && /global/'

[root@Oma.lan:/tmp]# set | awk '/local/ || /global/'

global='i am everywhere :)'

local='i am only here :('

[root@Oma.lan:/tmp]# ps

PID TTY TIME CMD

1342 pts/3 00:00:00 ps

32418 pts/3 00:00:00 su

32419 pts/3 00:00:00 bash

[root@Oma.lan:/tmp]# ksh

# ps

PID TTY TIME CMD

1344 pts/3 00:00:00 ksh

1345 pts/3 00:00:00 ps

32418 pts/3 00:00:00 su

32419 pts/3 00:00:00 bash

# echo $$

1344

# set | grep local

LD\_LIBRARY\_PATH=.:/opt/lib:/opt/usr/lib:/opt/usr/local/lib

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/tmp

# set | grep global

global='i am everywhere :)'

Here are some fragments from the output of the set command:

[root@Oma.lan:/tmp]# set

BASH=/bin/bash - **default bash absolute path**

HISTFILE=/root/.bash\_history - **default history file for this user**

HISTFILESIZE=500

HISTSIZE=500

HISTTIMEFORMAT='%h/%d - %H:%M:%S '

HOME=/root - **default root directory for this user**

HOSTNAME=Oma.lan

LANG=en\_US.UTF-8

PS1='[$LOGNAME@`hostname`:$PWD]# ' - **default prompt**

PWD=/tmp

SHELL=/bin/bash

TERM=xterm

TZ=Europe/Bucharest

UID=0

USER=root

\_=myscript.sh - **the last argument of the last command processed**

The user profile information is kept in files that can be global and local. When a user logs in the global profile is loaded and then a customized profile is loaded if this user has one. The main purpose of the profile files is to establish the default and if desired customized environmental varibles for users.

The global profile file **/etc/profile** and **/etc/bashrc**

The local profile file (the customized one) is placed within the user’s home directory under the same names profile and bashrc.

When switching between users it is very important if you use su or su - .

**NON–LOGIN SHELL -> su <user>** – only /etc/bashrc file is read and environmental variables are inherited from the previous user

**LOGIN SHELL -> Su – <user>** – only /etc/profile file is read and environmental variables are not inherited

Here is an example of customizing the $PS1 environmental variable for user root:

[student@Grendel ~]$ cat /etc/bashrc | grep -i PS1

if [ "$PS1" ]; then

[ "$PS1" = "\\s-\\v\\\$ " ] && PS1="[\u@\h \W]\\$ "

if [ "$PS1" ]; then

[student@Grendel ~]$ su root

Password:

my prompt#

my prompt#

my prompt#id

uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)

my prompt#

my prompt#

my prompt#pwd

/home/student

my prompt#id

uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)

my prompt#echo $PATH

/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/student/bin

my prompt#

my prompt#cat /root/.bashrc

# .bashrc

# User specific aliases and functions

alias rm='rm -i'

alias cp='cp -i'

alias mv='mv -i'

# Source global definitions

if [ -f /etc/bashrc ]; then

. /etc/bashrc

fi

export PS1="my prompt#"

my prompt#

User student does not have a customized profile, so the /etc/bashrc global file is read when he connected to the system, but when he switches to user root for a NON-LOGIN Shell he finds out that user root has a customized $PS1 variable (the prompt) which was read from the .bashrc file located in user’s root home directory.

Also notice that when user stundent switched to user root via a NON LOGIN Shell the working directory wasn’t changed to /root also the $PATH variable remained the same and the $PS1 variable was changed because it was customized for user root.