Linux Operations

Linux Environment Variables and Shell Variables

- In Linux and Unix based systems environment variables are a set of dynamic named values, stored within the system that are used by applications launched in shells or subshells. In simple words, an environment variable is a variable with a name and an associated value.
- Environment variables allow you to customize how the system works and the behavior of the applications on the system. For example, the environment variable can store information about the default text editor or browser, the path to executable files, or the system locale and keyboard layout settings.

In this guide, we will explain to read and set environment and shell variables.

- Variables format:

```
KEY=value
KEY="Some other value"
KEY=value1:value2
```

> Note:

- Names of the variables are case-sensitive. By Naming convention, environment variables should be UPPER CASE.
- There is no space around the equals = symbol.

Variables can be classified into two main categories, `environment variables, and shell variables`.

- Environment variables are variables that are available system-wide and are inherited by all spawned child processes and shells.
- Shell variables are variables that apply only to the current shell instance.
- Each shell such as `bash`, has its own set of internal shell variables.

Below commands that allows us to list and set environment variables in Linux:

- `printenv` – The command prints all or the specified environment variables.
- `export` – The command sets environment variables.
- `env` – The command allows you to run another program in a custom environment without modifying the current one. When used without an argument it will print a list of the current environment variables.

```
[cloudshell-user@ip-10-0-76-152 ~]$ whereis pwd
pwd: /usr/bin/pwd /usr/share/man/man1/pwd.1.gz
# Binaries present in /usr/bin/pwd do not require sudo permissions to execute.

[cloudshell-user@ip-10-0-76-152 ~]$ whereis useradd
useradd: /usr/sbin/useradd /usr/share/man/man8/useradd.8.gz

# Binaries present in /usr/sbin/useradd requires sudo permissions to execute.
```

- `set` – The command sets or unsets shell variables. When used without an argument it will print a list of all variables including environment and shell variables, and shell functions.
- `unset` – The command deletes shell and environment variables.

```
set
a=10
echo $a
unset a
echo $a
```

List Environment Variables

```
printenv
printenv HOME
```

- Commonly used environment variables:
  - `USER` - The current logged in user.
  - `HOME` - The home directory of the current user.
  - `SHELL` - The path of the current user's shell, such as bash or zsh.
  - `PATH` - A list of directories to be searched when executing commands. When you run a command the system will search those directories in this order and use the first found executable.
    - `echo $PATH`
      - `/sbin:/bin:/usr/sbin:/usr/bin`
        - These are directory path that are separted by colon `:`
  - `TERM` - The current terminal emulation.

Setting Environment Variables

```
export MYVAR="MyVar"
export MYPATH="/tmp"
export AWS_REGION="us-east-1"
```

Persistent Environment Variables

- To make the `Environment variables` persistent, we can those variables in the bash configuration files.
- Per-user shell specific configuration files, that gets executed whenever a user login happens. For example, if you are using Bash, you can declare the variables in the `~/.bashrc`:

```
export PATH="$HOME/bin:$PATH"
export MYUSERVAR="$USER"
export AWS_REGION="us-east-1"
```

- Execute the `~/.bashrc` using `source ~/.bashrc`.
- Or just exit the bash session and login again.
- In most Linux distributions when you start a new session/login terminal shell, environment variables are read from the following files:
- `/etc/environment` - Use this file to set up system-wide environment variables. Variables in this file are set in the following format:
- `/etc/profile` - Variables set in this file are loaded whenever a bash login shell is entered. When declaring environment variables in this file you need to use the export command:

```
export JAVA_HOME="/path/to/java/home"
export PATH=$PATH:$JAVA_HOME/bin
```

- To load the new environment variables into the current shell session use the source command:

```
source ~/.bashrc
```