Let us build the Linux Kernel

1 Installing Guest Additions on Ubuntu

The guest additions may be needed in case your installed VM has not completed all necessary updates during installation as well as for running your VM in full screen mode (which will be needed later during kernel compilation).

Follow these steps to install the Guest Additions on your Ubuntu virtual machine:

1. Login to your Ubuntu VM using suitable login and password.
2. Press Cltr+Alt+T to open the system terminal.
3. Update your APT database with `sudo apt-get update`, and typing your password, if requested.
4. Install the latest security updates with the command
   `sudo apt-get upgrade`.
5. Install required packages by running the command
   `sudo apt-get install build-essential module-assistant`.
6. Configure your system for building kernel modules by running the command
   `sudo m-a prepare`.
7. Look to the top menu bar for your VM while it is open/active. Click on Install Guest Additions... from the Devices menu, then choose to browse the content of the CD when requested. It may give errors about unmounting the related filesystem, sometimes. In which case, cancel this step and proceed to next step.
8. Change directory to /media/$< your - user - name >$/VBOXAdditions... and run the command
   `sudo sh VBoxLinuxAdditions.run`
   and follow the instructions on screen.
9. In case you have installed the VM along with installing all necessary updates, you may not follow the above steps [1]-[8] necessarily. You may only run the following command and it should give you the same result.
   `sudo apt-get install virtualbox-guest-dkms`

2 Setting the environment for Linux Kernal Compilation

To compile Linux Kernel the following are required to be installed.

- gcc latest version,
- ncurses development package and
- system packages should be upto date

To install the dependencies run following commands in terminal and type password for the user, when prompted.

1. For gcc installation (optional, in case you already have gcc on your VM), use the following command.

   `sudo apt-get install gcc`

2. For ncurses development package, use the following command.

   `sudo apt-get install libncurses5-dev`

3. After installing the above packages then update your system by running the following command.

   `sudo apt-get update && sudo apt-get upgrade`

4. Now download the Linux Kernel 4.2.5 from kernel.org or by using the below command.

   `wget https://www.kernel.org/pub/linux/kernel/v4.x/linux-4.2.5.tar.xz`

5. Once the download has completed move to the directory where you have downloaded the kernel package. In most cases your current working directory will be your home directory $/home/ < your - user - name >$, and if that was your directory where you executed the above `wget` command, your home directory is were the tar file for kernel package got downloaded. Now extract the tar file to the location `"/usr/src/"`. To move to the directory, for example if the downloaded package is in your "/home/user123" directory. Use the below command.

   `cd /home/user123`

6. To extract the tar file run the following command. Type the password for the user when prompted.

   `sudo tar -xvf linux4.2.5.tar.xz -C /usr/src/`

7. Now move to the directory where the extracted file is, or copy and paste the below command in terminal.

   `cd /usr/src/linux-4.2.5/`

8. Now you can configure, compile and install Linux Kernel 4.2.5 in your system. Run the commands one by one and type the password for the user, when prompted. To Configure (GUI mode):

   `sudo make menuconfig` or `sudo make xconfig` check - works only for older kernel versions 2.x

   The above command is used to configure the Linux kernel. Once you execute the command, you will get a pop up window with the list of menus and you can select the items for the new configuration. If you are unfamiliar with the configuration just check for the file systems menu and check whether ext4 is chosen or not, if not, select it and save the configuration. If you like to have your existing configuration then run (command-line mode - press enter whenever prompted) the below command.

   `sudo make oldconfig`

   There are other alternate configuration commands available and you can find them in README file under linux-4.2.5 directory.

3 Compiling Linux Kernel

1. To Compile, run the following command. What is CC, LD etc.

   `sudo make`

   The above command is used to compile the Linux Kernel. It will take some time to complete it, few minutes as large as 90 minutes on some machines, it depends on your system configuration.

   A close look at what your terminal screen shows while the above command is being executed, you will notice a few markings like CC, LD, CC (M) etc. which have some meaning in the context of ongoing kernel build. The different markings specify the following.

   CC Compiles the C file into an designated object file. The object file contains the archicture assembler code of that .c file. As it might also reference parts outside its scope. For example calling another function in another .c file. The function calls are left open within the object file, which is later included by the linker.

   LD is the proces of linking the compiled objects together, and wire up the function calls that has been left open by the compiler. However, many parts are linked together as the core part of the kernel, while some parts are left out.

   CC (M) for those parts which are compiled as points to be loaded into the kernel at runtime. But which are not linked together in the monolithic part of the kernel. But instead can be inserted when the kernel is booted.

   In summary, CC means that the file listed is being compiled from C by the C compiler. LD means that the file listed is being linked from a number of object files by the linker (ld); for e.g., `aacraid` is built from a number of files including src.o. Further, SHIPPED means that the file listed was shipped in the kernel source and is being copied as-is rather than rebuilt; it can be rebuilt if you really want to but doing that may require extra work. As indicated above, [M] means that the process is building a kernel module.

2. It is important to know which kernel version your Ubuntu installation is using at present. Use the command `uname -r` and keep note of the present kernel version.

3. To Install, run the following command.

   `sudo make modules_install install`

   The above command will install the Linux Kernel 4.2.5 into your system and this should replace the kernel image that was used by your Ubuntu VM. It will create some files under /boot directory and it will automatically make a entry in your grub.cfg.

   The files added under your /boot directory are on the Ubuntu VM will be as follows.

   `System.map-4.2.5`

   `vmlinuz-4.2.5`

   `initrd.img-4.2.5`

   `config-4.2.5`

   Also, check whether it made correct entry in the grub.cfg and check for the files it created. Also note that the /boot directory will continue to have all the files for your older kernel. The reason being that, in case your

newer kernel version was not installed properly, with the boot loader not able to start on the next boot, grub boot loader will continue to use your older kernel version.

View the grub.cfg file and you will find that entries for the newer kernel version (i.e., 4.2.5) have been made, and next time you restart your Ubuntu VM it will use this newer kernel, which you could verify using the same command mentioned in (2) above to check whether you correctly installed the linux kernel.

Restart your system by default it will boot in Ubuntu, you can able to see while booting.

Go to the command shell, and if you see 4.2.5 as the output of `uname -r`, be happy, you have successfully installed the new kernel version and replaced the kernel image on your Ubuntu VM.

It is to be noted that while you may be using Ubuntu 14.04 LTS 64-bit, which continues to use older kernel version 3.13x, you can build your system around newer kernel versions as and when they are released.