Note from Brent: Stuff added by me is in all caps and I have bolded all the important commands to run according to the version of Ubuntu and the kernel that we have.

Important: Running RealSense Depth Cameras on Linux requires patching and inserting modified kernel drivers. Some OEM/Vendors choose to lock the kernel for modifications. Unlocking this capability may requires to modify BIOS settings

Make Ubuntu Up-to-date:

- Update Ubuntu distribution, including getting the latest stable kernel:
    - **sudo apt-get update && sudo apt-get upgrade && sudo apt-get dist-upgrade**

Note: On stock Ubuntu 14 LTS systems and Kernel prior to 4.4.0-04 the standard `apt-get upgrade` command is not sufficient to bring the distribution to the latest recommended baseline.

It is recommended to upgrade the distribution with: (WARNING FROM BRENT: THIS STEP IS UNNECESSARY IF YOU HAVE THE LATEST KERNEL. CHECK WITH THE COMMAND uname -r AND SEE THE ACCEPTABLE VERSIONS IN THE LAST BULLET)

- `sudo apt-get install --install-recommends linux-generic-lts-xenial xserver-xorg-core-lts-xenial xserver-xorg-lts-xenial xserver-xorg-video-all-lts-xenial xserver-xorg-input-all-lts-xenial libwayland-egl1-mesa-lts-xenial`
- Update OS Boot and reboot to enforce the correct kernel selection with
- `sudo update-grub && sudo reboot`
- Interrupt the boot process at Grub2 Boot Menu -> "Advanced Options for Ubuntu" and select the kernel version installed in the previous step. Press and hold SHIFT if the Boot menu is not presented.
- Complete the boot, login and verify that a supported kernel version (4.[4,8,10,13,15,16]]) is in place with **uname -r**

Download/Clone librealsense github repository: **git clone https://github.com/IntelRealSense/librealsense.git**

1. Navigate to *librealsense* root directory to run the following scripts.
2. Unplug any connected Intel RealSense camera.

3. Install the core packages required to build *librealsense* binaries and the affected kernel modules:
4. **sudo apt-get install git libssl-dev libusb-1.0-0-dev pkg-config libgtk-3-dev**
5. Distribution-specific packages:
    - Ubuntu 14 or when running of Ubuntu 16.04 live-disk:
    - sudo apt-get install
    - ./scripts/install_glfw3.sh
    - Ubuntu 16:
    - sudo apt-get install libglfw3-dev
    - Ubuntu 18: (WE WANT THESE)
    - **sudo apt-get install libglfw3-dev libgl1-mesa-dev libglu1-mesa-dev**

Cmake Note: certain librealsense CMAKE flags (e.g. CUDA) require version 3.8+ which is currently not made available via apt manager for Ubuntu LTS.

Go to the official CMake site to download and install the application

Note on graphic sub-system utilization:
glfw3, *mesa* and *gtk* packages are required if you plan to build the SDK's OpenGl-enabled examples. The *librealsense* core library and a range of demos/tools are designed for headless environment deployment.

1. Install Intel Realsense permission scripts located in librealsense source directory:
2. **sudo cp config/99-realsense-libusb.rules /etc/udev/rules.d/**
3. **sudo udevadm control --reload-rules && udevadm trigger**
4. Build and apply patched kernel modules for:
5. * Ubuntu 14/16/18 with LTS kernel script will download, patch and build realsense-affected kernel modules (drivers).
6. Then it will attempt to insert the patched module instead of the active one. If failed the original uvc modules will be restored.
7. **./scripts/patch-realsense-ubuntu-lts.sh**
8. * Ubuntu with Kernel 4.16
9. ./scripts/patch-ubuntu-kernel-4.16.sh
10.
11. * Intel® Joule™ with Ubuntu Based on the custom kernel provided by Canonical Ltd.
12. ./scripts/patch-realsense-ubuntu-xenial-joule.sh
13. * Arch-based distributions

    - You need to install the base-devel package group.
    - You also need to install the matching linux-headers as well (i.e.: linux-lts-headers for the linux-lts kernel).
        - Navigate to the scripts folder `cd ./scripts/`
        -
        - Then run the following script to patch the uvc module: `./patch-arch.sh`
        -
        - * Odroid XU4 with Ubuntu 16.04 4.14 image Based on the custom kernel provided by Hardkernel
14. `./scripts/patch-realsense-ubuntu-odroid.sh`
15. Some additional details on the Odroid installation can also be found in installation_odroid.md

Check the patched modules installation by examining the generated log as well as inspecting the latest entries in kernel log:

```
sudo dmesg | tail -n 50
```

The log should indicate that a new uvcvideo driver has been registered.

Refer to Troubleshooting in case of errors/warning reports.

1. TM1-specific:
    - Tracking Module requires *hid_sensor_custom* kernel module to operate properly. Due to TM1's power-up sequence constrains, this driver is required to be loaded during boot for the HW to be properly initialized.
2. In order to accomplish this add the driver's name *hid_sensor_custom* to `/etc/modules` file, eg:

```
echo 'hid_sensor_custom' | sudo tee -a /etc/modules
```

Building librealsense2 SDK

- On Ubuntu 14.04, update your build toolchain to *gcc-5*: THIS BLOCK OF STEPS IS NOT NECESSARY WE HAVE A NEWER VERSION OF UBUNTU
    - sudo add-apt-repository ppa:ubuntu-toolchain-r/test
    - sudo apt-get update
    - sudo apt-get install gcc-5 g++-5

    - sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-5 60 --slave /usr/bin/g++ g++ /usr/bin/g++-5
    - sudo update-alternatives --set gcc "/usr/bin/gcc-5"
- You can check the gcc version by typing: `gcc -v` If everything went fine you should see gcc 5.0.0.
- Navigate to *librealsense* root directory and run **mkdir build && cd build**
- Run CMake:
    - `cmake ../` - The default build is set to produce the core shared object and unit-tests binaries in Debug mode. Use `-DCMAKE_BUILD_TYPE=Release` to build with optimizations.
    - **cmake ../ -DBUILD_EXAMPLES=true** - Builds *librealsense* along with the demos and tutorials (THIS COMMAND BUILDS ALL EXAMPLES FOUND IN LIBREALSENSE. TO FIND THE EXECTUABLES NAVIGATE TO /usr/local/bin OR ENTER **cd /usr/local/bin**)

      SIDE NOTE: IF YOU WANT TO BUILD THE OPENCV EXAMPLES AS WELL, THEY CAN BE BUILT WITH THIS COMMAND **cmake ../ -DBUILD_EXAMPLES=true -DBUILD_CV_EXAMPLES=true**

      HOWEVER AS OF NOW I CANNOT GET CMAKE WITH THE CV EXAMPLES TO RUN PROPERLY.

    - `cmake ../ -DBUILD_EXAMPLES=true -DBUILD_GRAPHICAL_EXAMPLES=false` - For systems without OpenGL or X11 build only textual examples

- Recompile and install *librealsense* binaries:
-
- **sudo make uninstall && make clean && make && sudo make install**
-
- The shared object will be installed in `/usr/local/lib`, header files in `/usr/local/include`.
- The binary demos, tutorials and test files will be copied into `/usr/local/bin`
- Tip: Use *make -jX* for parallel compilation, where *X* stands for the number of CPU cores available:
- `sudo make uninstall && make clean && make **-j8** && sudo make install`
- This enhancement may significantly improve the build time. The side-effect, however, is that it may cause a low-end platform to hang randomly.

- Note: Linux build configuration is presently configured to use the V4L2 backend by default.
- Note: If you encounter the following error during compilation `gcc: internal compiler error` it might indicate that you do not have enough memory or swap space on your machine. Try closing memory consuming applications, and if you are running inside a VM increase available RAM to at least 2 GB.

1. Install IDE (Optional): We use QtCreator as an IDE for Linux development on Ubuntu * Follow the link for QtCreator5 installation