Ekos/Indi Pi4 Image Setup

(Using Ubuntu 22.04.1 LTS (64-bit) OS)

Christopher Kovacs - Janyary 11, 2023

Introduction

The installation/deploy scripts allow you to Install Ekos/Indi on a Raspberry Pi 4 (Rpi4) by adding two files to a newly created Ubuntu Raspberry Pi 22.04-1 Release image file. Once installed onto an SD card and system is brought online, a single script can be executed that will install Ekos/Indi with various options and display access. This is a semi-automated process and once started only needs user input during various stages of the installation process.

Required Hardware

- Raspberry Pi 4 with 4 Gb memory (2Gb should work, but not tested)
- Minimum 16GB Class 10 SD card or a SSD drive
- Ethernet cable and router/switch access
- Display/Monitor (should not be required)

Expert Installation Overview

Below is an installation overview for Raspberry Pi experts. This is a is a very short installation overview for people that have previously completed this install process, or are familiar with Raspberry Pi administration.

- Create SD card Image using Ubuntu 22.04.1 LTS 64-bit OS.
- Use settings to create ekos user, set the ekos user password, and initially setup the wifi connection.
- Download the distribution files from git hub:
    - https://github.com/w0anm/Ekos-Indi_RPi_Image (run_first.sh and scripts.tar)
- Mount system-boot partition of the SD card image with Ubuntu image. Copy over the "run_first.sh" and "scripts.tar" files onto the system-boot partition (FAT partition).
- Unmount the SD card and install in the Rpi4, connect ethernet cable and power on the Rpi4.
- Determine the IP of the Rpi4 and open an ssh session using "ubuntu" account and password "ubuntu.

- Once password is updated, re-log into the Rpi4. As the ubuntu user, change directories and execute the "run_first.sh" shell (cd /boot/firmware; sudo bash run_first.sh)
- Answer the prompts and wait until the installation is completed.
- Reboot, and login using Ekos user. Select to install PHD2 or not.
- Setup WIFI using "wifi_setup.sh"

Creating SD/SSD Image using the Raspberry Pi Imager

Use the Raspberry Pi Imager to create an image on the SD card or SSD drive. This software is available for Linux, MAC, and Window systems. (https://www.raspberrypi.org/software/)

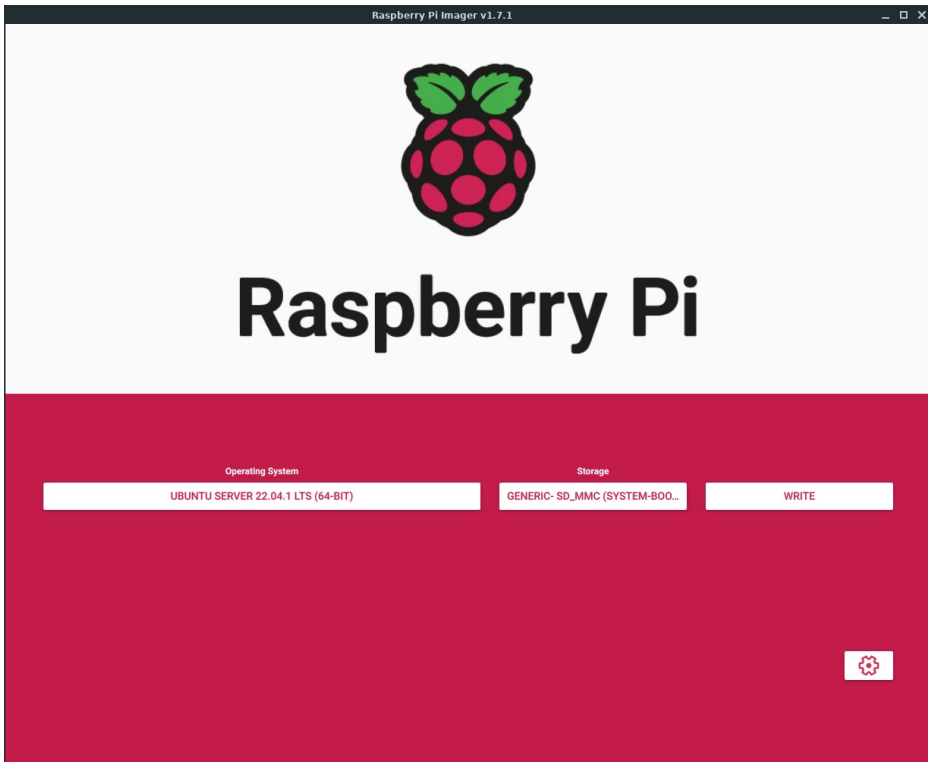
Raspberry Pi Imager is the quick and easy way to install Raspberry Pi OS and other operating systems on to a micro SD card or SSD drive, ready to use with your Raspberry Pi.

If you don't have an SSD drive you can still create the image on the SD card and use a the **`sd_to_ssd_conv.sh`** script described later in this document as well as what I recommend for SSD hardware.

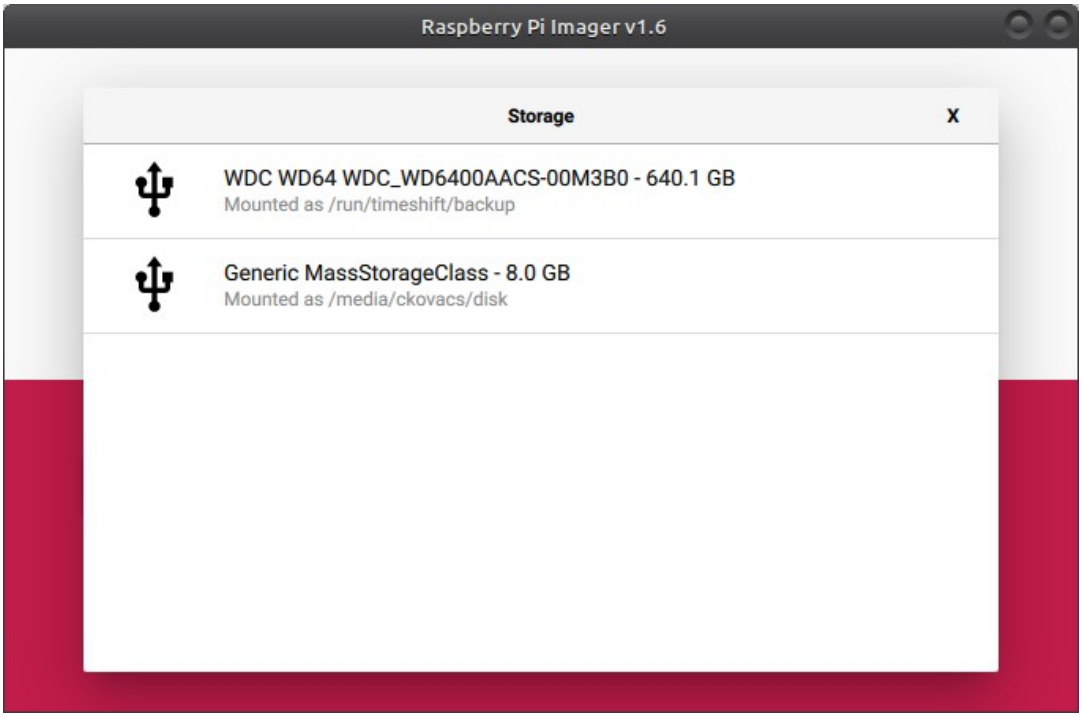
1. Download the Imaging software for your Operating System. This document describes installation using a Linux Desktop, but other operating systems version are similar in operation.

2. Insert an *vfat formatted* SD card in the SD Reader/Writer device.

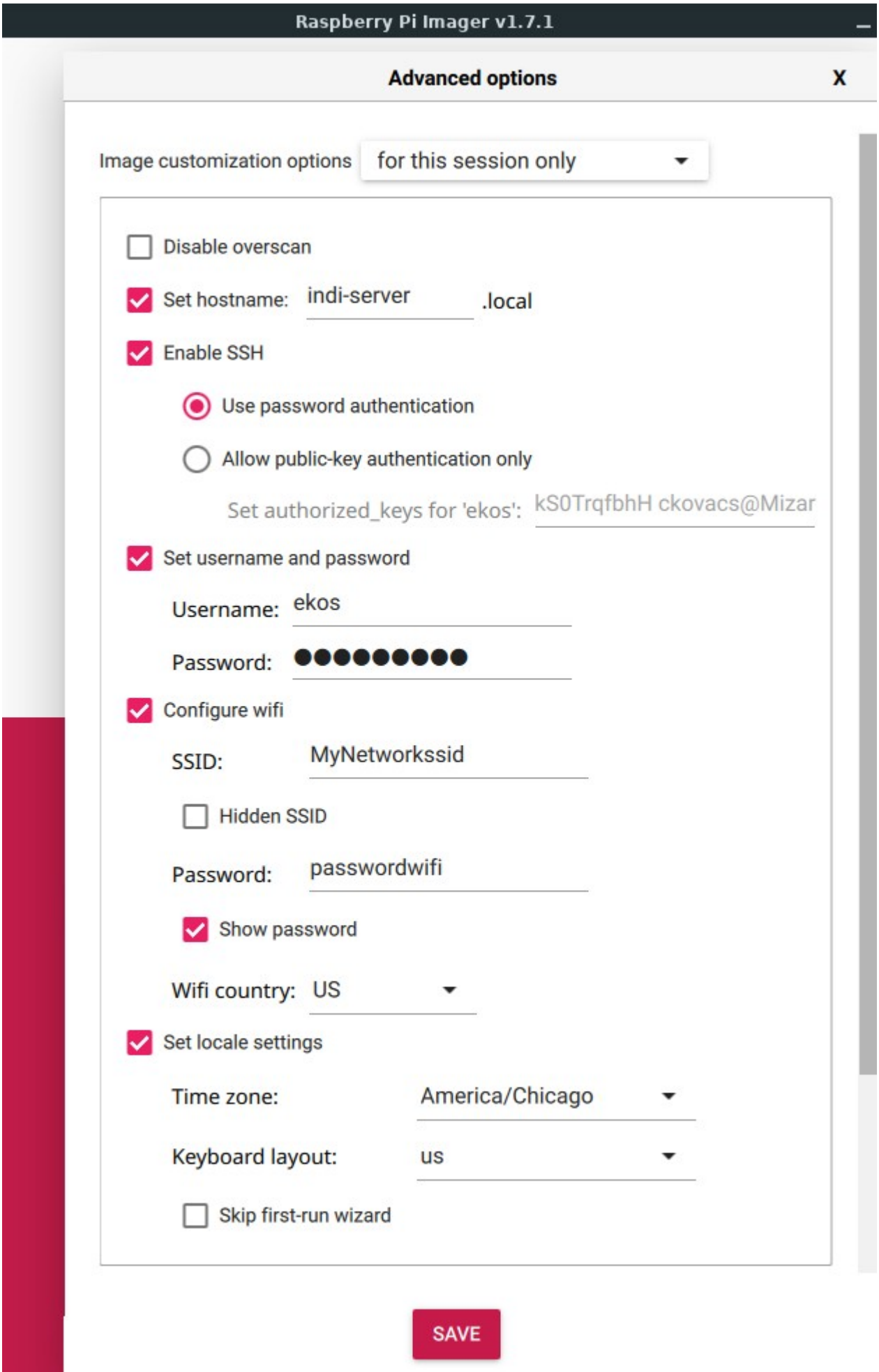
3. Select "CHOOSE OS", "Other general Purpose OS", "Ubuntu", and "Ubuntu Server 22.04.1 LTS 64-bit OS"
4.



5. Press "CHOOSE STORAGE" button, select your SD card:



6. Select the "Settings" Icon for the Advanced Settings. This will bring up an other menu as show here:


Raspberry Pi Imager v1.7.1
Advanced options
X
Image customization options
for this session only
Disable overscan
Set hostname:
indi-server
.local
Enable SSH
Use password authentication
Allow public-key authentication only
Set authorized_keys for 'ekos':
kS0TrqfbhH ckovacs@Mizar
Set username and password
Username:
ekos
Password:
Configure wifi
SSID:
MyNetworkssid
Hidden SSID
Password:
passwordwifi
Show password
Wifi country:
US
Set locale settings
Time zone:
America/Chicago
Keyboard layout:
us
Skip first-run wizard
SAVE

7. Check and fill out the following:
   - Desired Host Name
   - Enable ssh and use password authentication
   - If using wifi, Configure wifi and use the SSID/Password
   - Set up the locale settings (Timezone).
   - The rest can be left blank

8. Once reviewed, press "SAVE" button to save the settings.

IMPORTANT!
Make sure you select the create storage device. If you choose the wrong device, it will override the wrong device and may make your operating system non-functional.

9. PRESS "WRITE" button, verify that you have the correct device and then Select "YES"

Once verify, you can remove the SD card from reader or remove the USB driver for the SSD disk and then press Continue.

Adding Necessary Files to SD/SSD Image

Before you can use the newly created SD card, there are files that need to be installed in order to automatically install all necessary packages, files, and file modifications.

1. Download from Github the distribution zip file.

   https://github.com/w0anm/Ekos-Indi_RPi_Image/releases

2. Unzip the file in a temporary directory. There will be two files, "run_first.sh" and "scripts.tar"
3. Re-insert the SD card into the reader or re-connect the USB connector of the SSD drive. The Windows/Linux auto-mounter will mount the newly created system-boot filesystem (FAT Partition) in the image file system.
4. Using the file manager, copy the "run_first.sh" and "scripts.tar" files into this filesystem/folder (/media/<userid>/system-boot/).
5. Once added, eject and remove the SD card or disconnect the SSD drive.

Installation Process

The installation process for Ekos/Indi on a Rpi4 has been simplified by using scripts to automate the process. Once the software is installed, log into the Rpi4 server as "ekos" user and optionally install PHD2 software and setup WIFI network access.

Installation

1. Connect the Rpi4 directly to your router or switch via an Ethernet cable. Optionally, connect a monitor/keyboard to the Rpi4.
2. Install the newly created/modified SD card or connect the USB connector for the SSD drive in the Rpi4 and power on Rpi4.
3. After booting ssh into server or use Display/Monitor to login. Verify IP address and that system is on the network.
4. Insert the modified SD card image or the USB connector for the SSD drive into the Rpi4 and power the pi. Wait about 60 seconds, then using an ssh connection via an ssh client (or keyboard/monitor), login using user "***ekos***" and the password you entered when you created the image using Raspberry Pi Images Advanced Settings.
5. ~~You will immediately be prompted for a new password. Make sure you know/record this password. Once the ubuntu user account password is successfully updated, you will be terminated from the session and you will need to re-login using your new password. If you make a mistake either entering the current password, or new password, you can re-login to the Rpi4 and start the process over.~~
6. Once you've log into the Rpi4, change directories to /boot/firmware:

   ```
   cd /boot/firmware
   ```

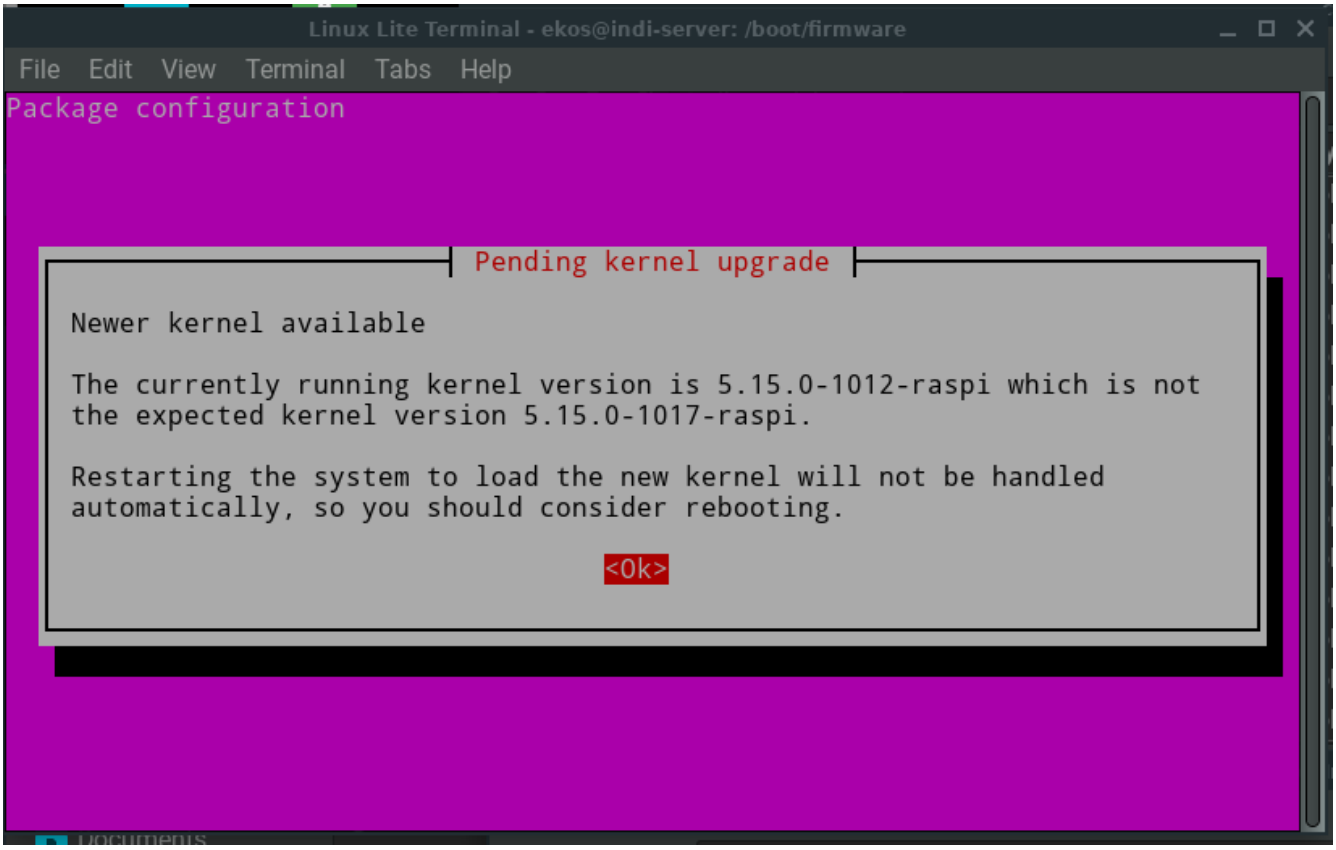
7. Now start the installation process by typing:

   ```
   sudo bash run_first.sh
   (enter password)
   ```

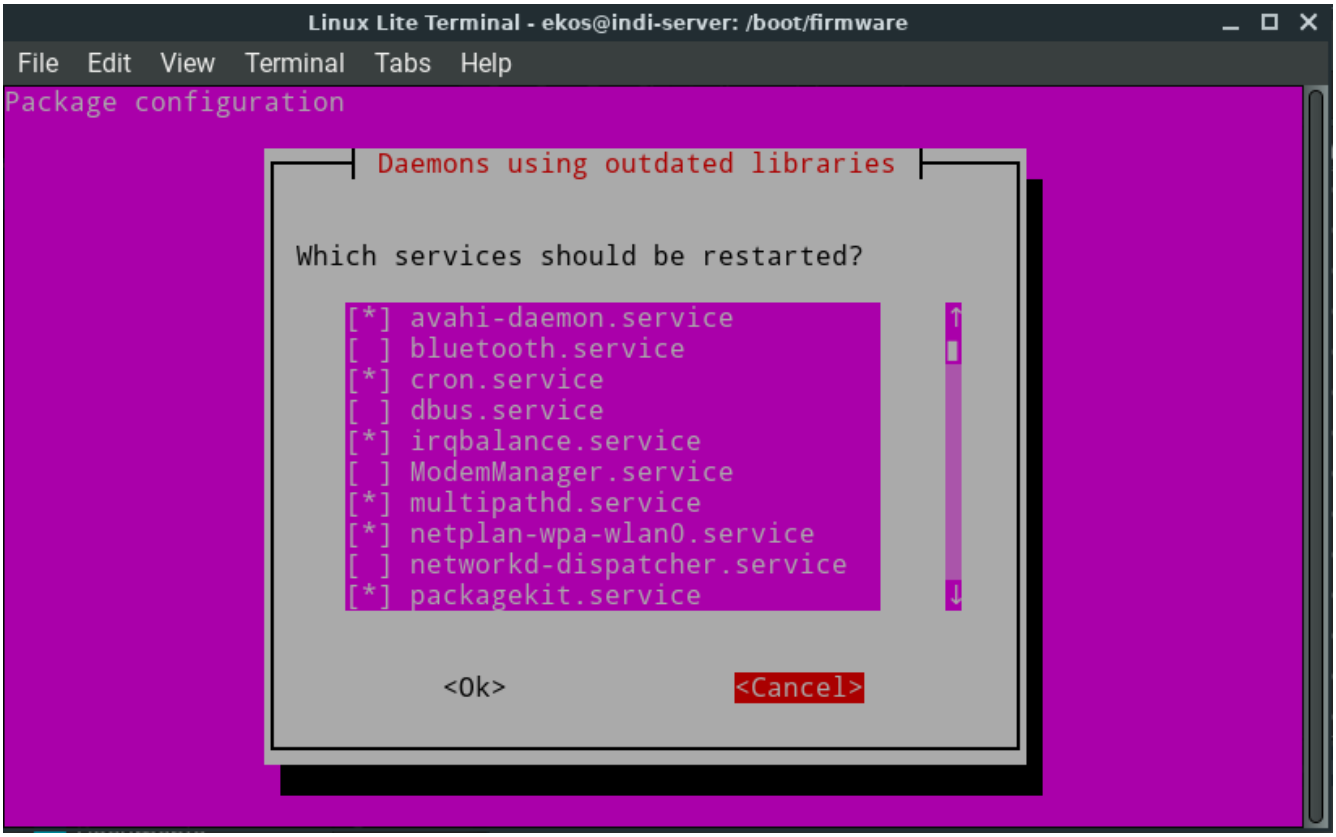
   This will start he installation process.

NOTE:

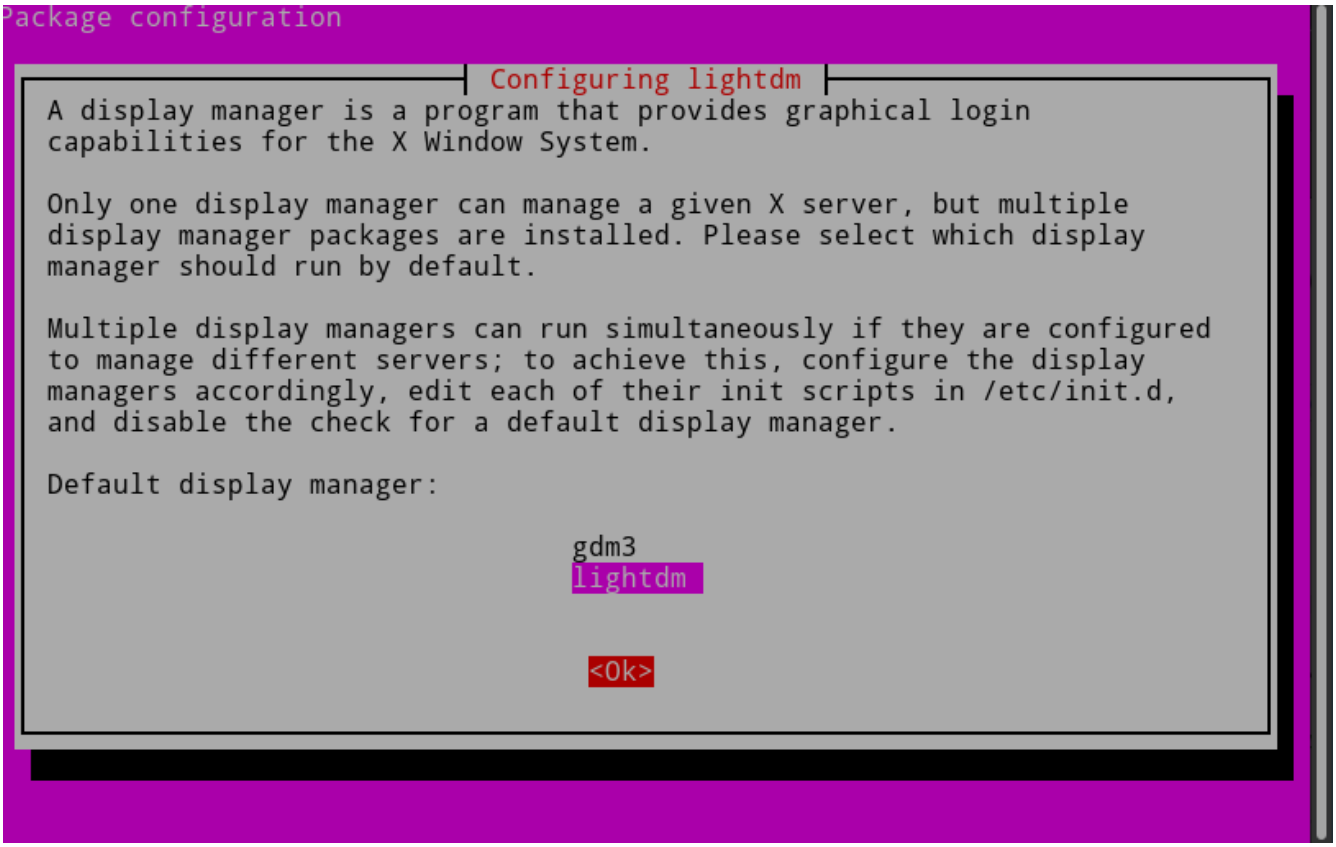
If you see a "Pending kernel upgrade" message, select "OK"



If you see a "Daemons using outdated libraries" message, select "Cancel" and continue.



You will be prompted for various information during this process. When prompted for the lightdm greeter, select "lightdm" as shown below.



Installation will start by copying over and extracting the script files and execute the start install script. The following will occur and can take up to an 45 minutes to complete:

- system update and upgrade
- Optionally change hostname
- package installation which includes:
  - desktop
  - indi packages
  - web manager (download/compile)
  - lightdm Greeter (**you will be prompted selecting display manager, choose lightdm**)
  - x11vnc
- unnecessary packages
- system file installs
- Setting up services and stopping unnecessary services. Y**ou will be prompted for vnc-server password as shown below:**

```
Setting x11vnc password...
Enter VNC password:
Verify password:
Write password to /root/.vnc/passwd?  [y]/n y
Password written to: /root/.vnc/passwd
```

- cleanup installation files
- System Reboot.

Once rebooted, you'll need to login using "***ekos***" user as outlined in the next section.

Ekos first login

Wait a minute or so for the system to reboot. Log back into the indi-server using the "ekos" user.

Once logged into the server, you will be prompted if you want to install PHD2. This will take some time to build and install. If you select "no" to the option, you can still build PHD2 by using the command:

```
_phd2_install.sh
```

WIFI Setup

I created a simple comand line script to configure an WIFI setup. This sets up the netplan for the WIFI interface for the desired SSID/Password. To use this script, type:

```
wifi_setup.sh
Then enter the SSID and password when prompted as shown below:
Wifi Module: wlan0
What is the wifi ssid you want to connect to (Case sensitive)? myssid
What is the wifi password? myssidpassword
Creating new file using:
    wifis:
        wlan0:
            dhcp4: true
            optional: true
```

```
            access-points:
                "myssid":
                    password: "myssidpassword"
Applying the new net plan…
NAME                    UUID                                  TYPE      DEVICE
netplan-eth0            626dd384-8b3d-3690-9511-192b2c79b3fd  ethernet  eth0
netplan-wlan0-Blackhole 0852fa06-55c4-3874-812e-08369cf7c82f  wifi      wlan0
```

Once entered, the script will generate a netplan and then apply. You can verify that you have access to the WIFI network by using:

```
ip a s

nmcli connection show

status-wifi
```

Below is an example output:

ip a s

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state
UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq
state DOWN group default qlen 1000
    link/ether dc:a6:32:94:3e:64 brd ff:ff:ff:ff:ff:ff
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
fq_codel state UP group default qlen 1000
    link/ether dc:a6:32:94:3e:66 brd ff:ff:ff:ff:ff:ff
    inet 192.168.15.62/24 brd 192.168.15.255 scope global
dynamic noprefixroute wlan0
       valid_lft 85668sec preferred_lft 85668sec
    inet6 fe80::dea6:32ff:fe94:3e66/64 scope link
       valid_lft forever preferred_lft forever
```

wifi-status

```
Every 1.0s: iw wlan0 info; ip add...  indi-server-test: Tue Oct
26 20:16:02 2021

Interface wlan0
        ifindex 3
        wdev 0x1
        addr dc:a6:32:94:3e:66
        ssid myssid
        type managed
        wiphy 0
        channel 40 (5200 MHz), width: 80 MHz, center1: 5210 MHz
        txpower 31.00 dBm
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
fq_codel state UP gro
up default qlen 1000
    link/ether dc:a6:32:94:3e:66 brd ff:ff:ff:ff:ff:ff
    inet 192.168.15.62/24 brd 192.168.15.255 scope global
dynamic noprefixroute
wlan0
       valid_lft 86125sec preferred_lft 86125sec
    inet6 fe80::dea6:32ff:fe94:3e66/64 scope link
       valid_lft forever preferred_lft forever
default via 192.168.15.1 dev eth0 proto dhcp metric 100
default via 192.168.15.1 dev wlan0 proto dhcp metric 600
169.254.0.0/16 dev eth0 scope link metric 1000
192.168.15.0/24 dev eth0 proto kernel scope link src
192.168.15.61 metric 100
192.168.15.0/24 dev wlan0 proto kernel scope link src
192.168.15.62 metric 600
```

Use “control-c” to stop the output.

Added Useful Programs

I have added extra programs/scripts. These programs/scripts include switching between x11vnc server access, creating a Rpi4 Hotspot network, and backup scripts. Below describe the programs and usage.

Using Hardwire Ethernet Connection

By default, you can use the hard-wire Ethernet connection (rj45 cable) with a router. This works great when you have a home network that leads out the your observing area, but doesn't work out remotely.

You could bring a router with you and setup the connection via the router, but that means more items to connect. Instead of a router, you can connect a Ethernet cable between the laptop device and the i. RPi4. Most modern interfaces will work without a crossover cable.

In order to run with a RJ45 cable, you need to setup static IP on the RPi4 and make sure you are using a static IP on your laptop device and on the RPi4. I have written a script to convert to a static IP for the RPi4. This script takes the standard configuration for networking and coverts it to allow static IP address without internet dependencies (dns,dhcp, etc). This allows to use a cable to cable connection.

The script does the following:

- Disables system-resolved services
- Sets static address
- Disable auto network configuration
- Configures resolv.conf (dns resolving)
- Adds host information to /etc/hosts

The script can be found in /root directory:

```
/root/conv_static_addr.sh
```

Before you can use this script, you need to edit the script and change the Network Settings and save, Once saved, you can then execute the script as root.

Using Lightdm and x11vnc Server

Normally, the Lightdm X Display Manager (Lightdm) and x11vnc is disabled. If you want to use these optional remote methods of access, you will need to run the following script:

vnc_greeter_control.sh

```
Window Management Utility

1 - Start/Stop X11vnc Service (with Window Manager)
2 - Start/Stop Window Manager Greeter
3 - Exit

Enter selection:
```

Option 1 allows you to startup a vnc session and optionally starting that service on reboot.

Hotspot Network

This scripts enables the Rpi4 to become a hotspot. The script defaults for the hotspot network name and password is:

```
HOTSPOTNAME=EkosHotSpot
```

PASSWORD=Ekos4192

Here is an example output of the program:

To disable the hot spot, type:

stop_hotspot.sh

This will disable the hotspot and resume the normal wifi connection if being used.

sysinfo

This is a simple program that shows basic information (was part of the initial login).

sysinfo

System information as of Fri Oct 15 12:04:10 CDT 2021

System load: 0.94 Temperature: 28.2 C
Usage of /: 20.7% of 218.54GB Processes: 163
Memory usage: 9% Users logged in: 1

Swap usage: 0% IPv4 address for wlan0: 192.168.15.62

Backup Script

I have added a backup scripts that you can use to create a backup either onto a nfs mounted filesystem, or a thumb-drive. I recommend using a SanDisk Ultra (r), USB 3.0 Flash Drive.

File	***Description***
/usr/local/sbin/image_backup.sh	Creates a backup image on a usb thumb drive formatted as vfat. (Wrapper for sys-imagebkup)
/usr/local/sbin/image_nfs_backup.sh	Creates a backup image on to an nfs mounted filesystem. (Wrapper for sys-imagebkup)
/usr/local/sbin/sys-imagebkup	Main imaging backup script
/usr/local/etc/image_backup.conf	Configuration file for backup. Make sure you review and update.

The scripts will take a current running system and create an image backup. This image backup can then be used to create another SD card image. SD cards do have limited write access counts and can fail. This is useful to backup and restore the system.

Formatting a USB drive for use with image_backup.sh script

You must partition and format the USB drive. This can be done using the Rpi4.

Note: If you are using an ssd device, the mount point will be device will be different and normally it will be device "sdb". Below shows a non-ssd mounted OS.

1. Insert the usb drive into the Rpi4 usb port.

2. Verify that the device can be see by the Rpi4 by using the following command:
   `sudo lsblk`

```
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda           8:0    1 28.7G  0 disk
└─sda1        8:1    1 28.7G  0 part
mmcblk0     179:0    0 14.9G  0 disk
├─mmcblk0p1 179:1    0  256M  0 part
└─mmcblk0p2 179:2    0 14.6G  0 part /
```

3. If the device is mounted, see above "MOUNTPOINT", you must un-mount the device:
   `sudo umount /dev/sda1`

4. Create partition on the device, so use the following commands to remove an existing partition and and create a new one:

   `sudo fdisk /dev/sda`

   a) Delete existing partitions by using selecting "**d**" and then enter at the prompt.

b) Create a new partition by entering "**n**" and then return at the prompt and when prompted for partition type, enter "**p**" for primary.

c) Then press "enter" at each prompt to except all of the defaults.

d) Finally, verify that the partition is created by entering "**p**" at the prompt and the device partition information will be displayed. and finally use the "w" enter, to write the changes to the device.

For example:

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p):p
Partition number (1-4, default 1): <enter>
First sector (2048-120176639, default 2048): <enter>
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-120176639, default
120176639): <enter>

Created a new partition 1 of type 'Linux' and of size 57.3 GiB.

Command (m for help): p
Disk /dev/sdf: 57.31 GiB, 61530439680 bytes, 120176640 sectors
Disk model:  SanDisk 3.2Gen1
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x0e9d9c79

Device     Boot Start       End   Sectors  Size Id Type
/dev/sdf1        2048 120176639 120174592 57.3G 83 Linux

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

5. Once the partition has been created, you can then use the following command to create an exfat file system:

sudo mkfs.exfat /dev/sda1
mkexfatfs 1.3.0
Creating... done.
Flushing... done.
File system created successfully.

6. Now label the newly created file system:

sudo fatlabel /dev/sda1 backup_imgs

You can ignore the warning about lowercase labels/

Image Backup Process

Before using the image_backup.sh script

If you are using an SSD drive, you must edit the following file: /usr/local/etc/image_backup.conf.

Comment out (add '#" to front of lines) the standard configuration entries:

```
# standard config
# DISKDEV=mmcblk0
# DEVPART=p
# BOOT_DISK_DEV=/dev/mmcblk0p1
# SYS_DISK_DEV=/dev/mmcblk0p2
```

and uncomment out (remove '#") the SSD drive entry:

```
# for ssd drive
DISKDEV=sda
DEVPART=
BOOT_DISK_DEV=/dev/sda1
SYS_DISK_DEV=/dev/sda2
```

You will need a exfat formatted usb thumb-drive (see above previous section).

1. mount usb drive:

   sudo udisksctl mount -b /dev/sda1

2. Make sure that /boot/firmware is mounted by entering the command:
   sudo mount /boot/firmware

3. Start the backup process by executing:

   sudo image_backup.sh

   At the prompt, you have to enter a selection. This selection can be an existing image file or a "New" file. If a "New" selection is used, it will create a new image. If you use an existing image, it will overwrite/resync the old image and will be faster.

   Also, you will be prompted for a image size. The default size is the size of the SD card/SSD Drive. You can reduce this value to save time or if the SD card is very large. The typical size for this image is about 9000MB (9Gb). This value must be larger than the actual overall filesystem size. You do not need to use the default value if the SD card in the Rpi4 is large, ie 32G, but can be created smaller.

4. When completed, umount usb drive:

```
sudo udisksctl unmount -b /dev/sda1
-or-
sudo umount /dev/sda1
```

Once the disk is unmounted, you can remove this driver and use the image file on the usb drive to create a new SD card image which is a snapshot of the mount OS on your Rpi4.

Below is an example execution:

```
sudo udisksctl mount -b /dev/sda1
sudo image_backup.sh

+=========================================================================+
| Please insert an exFAT  formatted usb thumb drive for                    |
| your backup.                                                            |
|                                                                         |
| Press any key when disk has been inserted, or  (Ctl-C) to abort..       |
|                                                                         |
+=========================================================================+

Checking for media, please wait...  Premount-/media/root/backup-imgs
found media.  [/media/root/backup-imgs]

+=========================================================================+
| Select an existing backup image file to update by entering the number   |
| next to the file.                                                       |
|                 -OR-                                                    |
| Select the number corresponding to "New" to create a new backup image.  |
+=========================================================================+
1) .
2) New
=== Please select:  2

executing --> /usr/local/sbin/sys-imagebkup -s 9000 -n /media/root/backup-imgs -
m /mnt/bkup -f indi-server_2021-10-19-1318.img -b /boot/firmware
Creating NEW Image File: indi-server_2021-10-19-1318.img

/media/root/backup-imgs
Creating image file.   Working [2724]... [/] 1+0 records in
1+0 records out
1048576 bytes (1.0 MB, 1.0 MiB) copied, 0.0968456 s, 10.8 MB/s
Image creation complete.
GNU Parted 3.3
Using /media/root/backup-imgs/indi-server_2021-10-19-1318.img
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) mktable msdos
(parted) unit s
(parted) mkpart primary fat32 2048 526335
(parted) mkpart primary ext4 526336 18431999
(parted) quit
mkfs.fat 4.1 (2017-01-24)
```

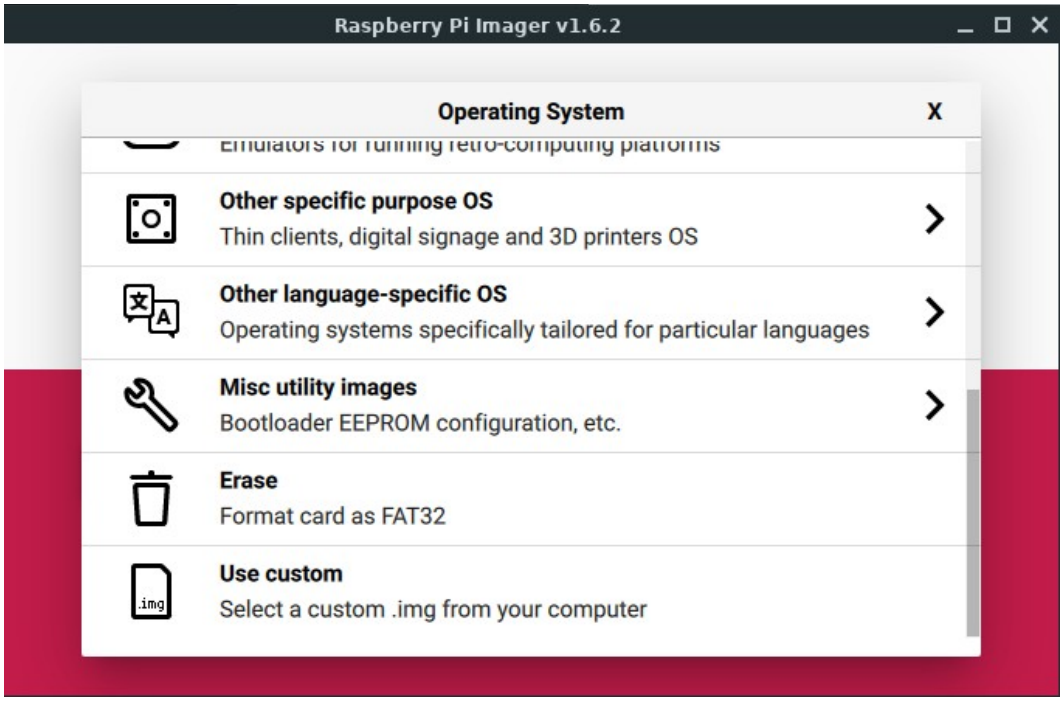
```
fatlabel: warning - lowercase labels might not work properly with DOS or Windows
mke2fs 1.45.5 (07-Jan-2020)
Creating filesystem with 2238208 4k blocks and 559728 inodes
Filesystem UUID: 59505467-ea81-49fa-bbac-e5d89e81353c
Superblock backups stored on blocks:
      32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

-----------------------------------

Running backup now...
    This could take up to 50 minutes to completed. Please wait..
===================================

     Working [4101]....
Update completed, umounting image, please wait..

Differences between existing /etc/fstab and new /etc/fstab:

Differences between existing /boot/firmware/cmdline.txt and
/mnt/bkup/boot/firmware/cmdline.txt:
    Finished.
-----------------------------------

    OK to remove USB drive ...

Completed.

sudo udisksctl unmount -b /dev/sda1
```

Restore Backup

To Restore, copy image to a system that has the tool to create SD cards from images such as Pi Imager (v1.6 or higher).

Select "***Use custom, Select a custom .img from your computer***", under "Choose OS" drop down . You will be prompted to select an image name. (See below).

Follow the same process as you did when creating the initial installation of Ubuntu. See above "Creating SD Image using the Raspberry Pi Imager".



Coping the OS from the SD card to SSD device

I have added a script to version v0.8 to copy over the contents of the SD card to an SSD device and setup the boot files/processes. I have been using the Kingston 240G SSD device with a StarTech.com SATA to USB Cable - USB 3.0 to 2.5” SATA III Hard Drive Adapter.

Kingston 240GB A400 SATA 3 2.5” Internal SSD SA400S37/240G:

https://www.amazon.com/StarTech-com-SATA-USB-Cable-USB3S2SAT3CB/dp/B00HJZJI84/ref=rvi_1/134-7439384-2707630?pd_rd_w=mnhoo&pf_rd_p=c0296674-5a83-4ad6-b035-0702d2b359df&pf_rd_r=VHMMQX3M07AR0EER4W1D&pd_rd_r=ad9a2f5d-c41e-40fc-915e-a354db568832&pd_rd_wg=lV9R3&pd_rd_i=B00HJZJI84&th=1

StarTech.com SATA to USB Cable - USB 3.0 to 2.5” SATA III Hard Drive Adapter - External Converter for SSD/HDD Data Transfer (USB3S2SAT3CB):

https://www.amazon.com/dp/B01N5IB20Q?ref=ppx_yo2_dt_b_product_details&th=1

Using the script

Once the SD image has been created, installed and booted, you can execute the script.

1. Install the USB to Sata cable and attach the SSD device.

2. Verify that the device is being seen by usng lsblk:
   `lsblk`

   You should see the device:

```
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda           8:0    0 223.6G  0 disk
├─sda1        8:1    0   512M  0 part
└─sda2        8:2    0   223G  0 part
mmcblk0     179:0    0  29.7G  0 disk
├─mmcblk0p1 179:1    0   256M  0 part /boot/firmware
└─mmcblk0p2 179:2    0  29.5G  0 part /
```

   Note that the number of partions may vary.

3. To execute the script, type:
   `sudo su -`
   `./sd_to_ssd_conv.sh`

4. You will be prompted to continue:

```
This script converts the Ekos/Ubuntu OS contents to an SSD device..

Hit any key to continue, or control-C to abort!
```

5. The process will start. You may get a message about system label, go ahead and type "y" to continue. If my mistake you type something else, just re-execute the script. You can also ignore the warning about lowercase labels.

6. If you get any messages about optomizing partitions, you can continue. However, I would like to know about this as I am reviewing if this optimazation can be automated. I am using a standard start sector that works with most SSD devices.

7. Once the script is complete, go ahead and shutdown the RPi4:
   `shutdown -h now`

8. Disconnect the power, and remove the SD card.

9. Re-attached the power cable and wait for the system to reboot.

10. Once rebooted, re-log back into the server and you now are using the SSD device.

Using the Script with SSD Device

Note: If you are using an ssd device, the mount point will be device will be different and normally it will be device "sdb". Below shows a non-ssd mounted OS.

The backup USB device will be something other than sda device as this is being used as the ssd mounted device. For example, there is my system using an ssd device with a mounted usb device:

```
root@indi-server:/# sudo udisksctl unmount -b /dev/sdb1
Unmounted /dev/sdb1.
root@indi-server:/# lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda      8:0    0 223.6G  0 disk
├─sda1   8:1    0   512M  0 part /boot/firmware
└─sda2   8:2    0   223G  0 part /
sdb      8:16   1  57.3G  0 disk
└─sdb1   8:17   1  57.3G  0 part
```

As you can see from above, the sda device is the ssd device and the /dev/sdb1 device is the partitioned usb device. When using the backup script, you must make sure you don't exceed the usb device size. You should review the backup configuration file /usr/local/etc/image_backup.conf file.

Here is an example:

```
# Configuraiton file for image_nfs_backup.sh

# Various Distribution Setups
# OSTYPE=UBUNTU              # mount is different with Ubuntu
# OSTYPE=RASPBIAN            # most Rpi2/3 using debian including
OSTYPE=UBUNTU

# BOOTCMDFILE=cmdline.txt
# This version uses btcmd.txt
# BOOTCMDFILE=btcmd.txt
BOOTCMDFILE=cmdline.txt

# exclude list
# EXCLUDE="--exclude /media/ --exclude /opt/media/"
EXCLUDE="--exclude /media/ --exclude /astrometry/ --exclude /home/"

# mount point on local system default
LOCAL_MOUNTPT=/nfs_image1

# nfs server mountpoint default
NFS_SERVER_NAME=rpi-backuppc
NFS_SERVER_MOUNTPT=/nfs_image1

# standard config
# DISKDEV=mmcblk0
# DEVPART=p
# BOOT_DISK_DEV=/dev/mmcblk0p1
# SYS_DISK_DEV=/dev/mmcblk0p2
```

```
# for ssd drive
DISKDEV=sda
DEVPART=
BOOT_DISK_DEV=/dev/sda1
SYS_DISK_DEV=/dev/sda2
```

The EXCLUDE variable but be keep in the same format when adding directories. In the above example, the /media (which is always excluded), the /astrometry and the /home directories have been excluded. I backup up the /home directory seperate from the main backup. Also, you will noticed at the end of the configuration, the standard config (using mmcblk0 devices) is commented out and the updated ssd device added.

Example Execution Output

```
root@indi-server:~# ./sd_to_ssd_conv.sh

This script converts the Ekos/Ubuntu SD contents to an SSD device..

Hit any key to continue, or control-C to abort!

Creating Partiions...

GNU Parted 3.3
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) rm 1
(parted) rm 2
(parted) mktable msdos
Warning: The existing disk label on /dev/sda will be destroyed and all data
on this disk will be lost. Do you want to continue?
(parted) unit s
(parted) mkpart primary fat32 65535 1114094
(parted) mkpart primary ext4 1114095 468862127
(parted) quit
Information: You may need to update /etc/fstab.

Creating File systems...

mkfs.fat 4.1 (2017-01-24)
mke2fs 1.45.5 (07-Jan-2020)
/dev/sda2 contains a ext4 file system labelled 'ssd-writable'
      last mounted on / on Tue Sep  7 13:37:23 2021
Proceed anyway? (y,N) y
Creating filesystem with 58468504 4k blocks and 14622720 inodes
Filesystem UUID: 1afde20c-5ce6-4192-bda2-a3374fef72cb
Superblock backups stored on blocks:
      32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
      4096000, 7962624, 11239424, 20480000, 23887872

Allocating group tables: done
Writing inode tables: done
Creating journal (262144 blocks): done
```

```
Writing superblocks and filesystem accounting information: done

Labeling new partitions...

fatlabel: warning - lowercase labels might not work properly with DOS or
Windows
Creating mount points and mounting newly created filesystems...

Starting Rsync...

Rsync completed...
updating /etc/fstab...

Updating cmdline.txt with new label...

Setup Boot files which includes kernal uncompression...

Found writable partition at /ssd/
Found boot partition at /ssd/boot/firmware
Decompressing kernel from vmlinuz to vmlinux...
Kernel decompressed
Updating config.txt with correct parameters...
Creating script to automatically decompress kernel...
Creating apt script to automatically decompress kernel...
MD5 generated Succesfully
Unmounting /ssd directory and cleaning up...

================================================================================
Completed...
  Now shutdown Rpi4 and remove SD card from Rpi4. Turn on Rpi4 and start
  the boot process using the SSD
================================================================================
```