Contents

1. **Project Introduction** 1
2. **Executive Summary** 1
3. **Problem Statement** 1
4. **Project Objectives** 2
5. **Requirements:** 2
3. **Connect components** 3
4. **Flash Operating System (OS) on SD card:** 4
4. **Establish ssh connection** 6
5. **Server Configuration** 7
6. **Disk Mount** 8
7. **CasaOS Installation** 9
8. **Helpful References** 10

1. Project Introduction

Project Title: NiCloud (Local NAS)
Nirmal Thapa, Minnesota State University Mankato
Date: September 5, 2024

2. Executive Summary

Brief Overview:
At present cloud services are widely used by individuals and organizations for effective, secure, and easy data management solutions. This research proposal aims to develop and implement a local NAS (Network Attached Storage) for individuals looking for solutions to store their data without having to pay other external vendors like Google Cloud, Apple Cloud, or OneDrive. This research addresses data storage, performance, power consumption, and data access reliability. NiCloud is a personal storage server that helps to store files, videos, music etc.

3. Problem Statement

Each day a large amount of digital data is collected (Pictures, videos and in other forms). This data needs designated solutions for data retention. Devices, for example, iPhones, run out of

storage capacity eventually. Users are recommended to back up the data either by purchasing more storage in the cloud or transferring it to different traditional storage devices like pen drives and hard drives. The process of collecting data never stops but instead increases significantly. Cloud subscribers have no option but to buy more storage by paying more money each month. of traditional devices like pen drives poses a huge risk to data integrity and security as those devices can easily be lost or accessed by other people. This project will provide data storage solution by developing a local NAS that stores digital data at an individual level.

The research aims to provide storage capacity up to 1 TB in the initial phase and continue to make the NAS expandable in the later phases. After the project's completion, individuals will no longer have to keep paying for cloud services or carry traditional storage devices.

4. Project Objectives

- Development and implementation of a local NAS system: A functioning NAS will be developed that has storage capacity of 1 TB; download and upload speed of 100mbps.
- Increase reliability and performance: Make sure NAS will be accessible for authorized users for data storage whenever needed.
- Evaluate cost effectiveness: Comparative Analysis on initial cost to set up, maintenance and power consumption will be carried out.
- Storage expansion capability: It will also allow us to expand storage capacity by adding more HDD or SSD.

5. Requirements:

1.1. Raspberry pi
1.2. Power Adapter 5V/5A
1.3. SD card (Preferred 64 GB)
1.4. SD card reader
1.4. Penta SATA HAT (4 SATA + 1 eSATA interface)
1.5. Power adapter for SATA HAT
1.6. Hard Drive (1-5)
1.7. FPC

3. Connect components

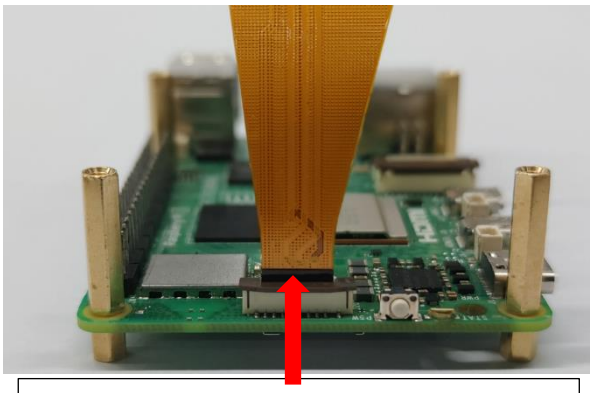


Fig 2.1 While connecting FPC with Raspberry Pi, make sure the black side is facing you.

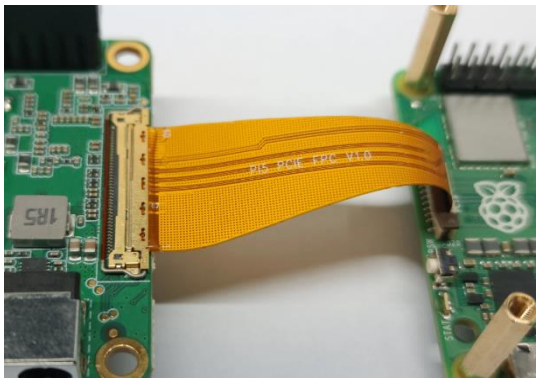


Fig 2.2 Connect with SATA HAT

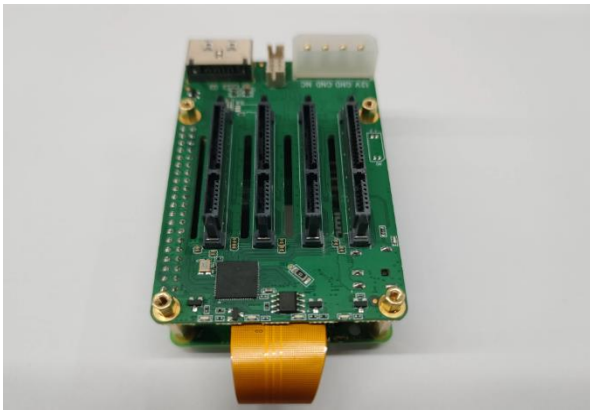


Fig 2.3 Stack SATA HAT on top of Raspberry pi with screws.

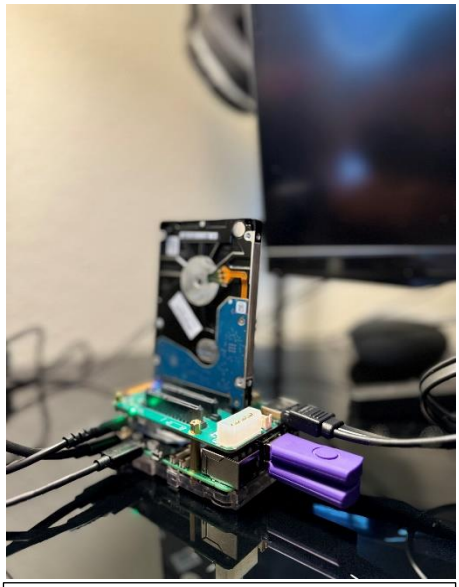
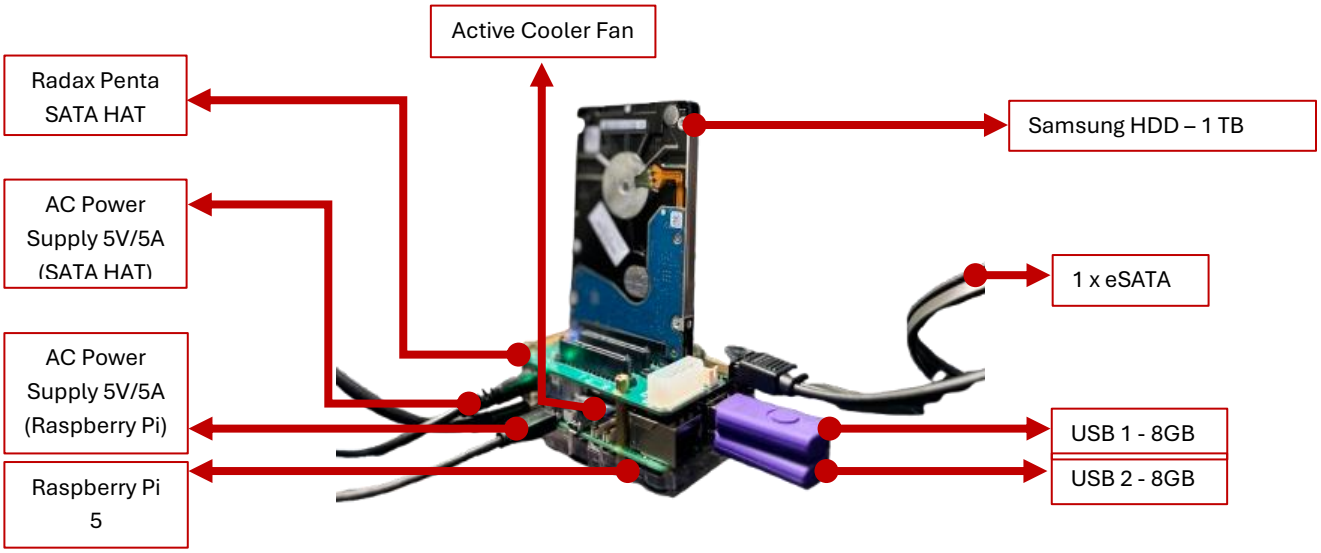


Fig 2.4 Install storage device on SATA HAT

Fig 2.1, 2.2, 2.3, Source: https://docs.radxa.com/en/accessories/penta-sata-hat/penta-for-rpi5

Note: If you are using old drives, please make sure data is transferred to the safe location and hard drive is formatted for easy mounting process in the later steps.



4. Flash Operating System (OS) on SD card:

Before inserting SD card into Raspberry pi, flash an image of an Operating System in the SD card. BalenaEtcher (https://etcher.balena.io/) or Raspberry Pi imager (https://www.raspberrypi.com/software/) can be used to flash OS. I will be using Raspberry pi Imager here. To do so, follow the steps below:

3.1. Download and Install Raspberry pi imager from https://www.raspberrypi.com/software/.

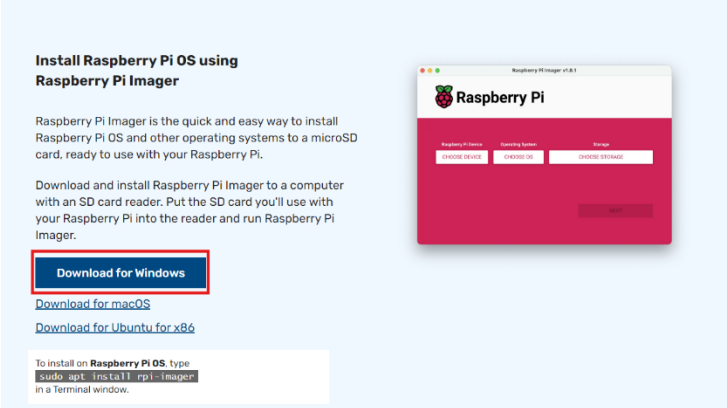


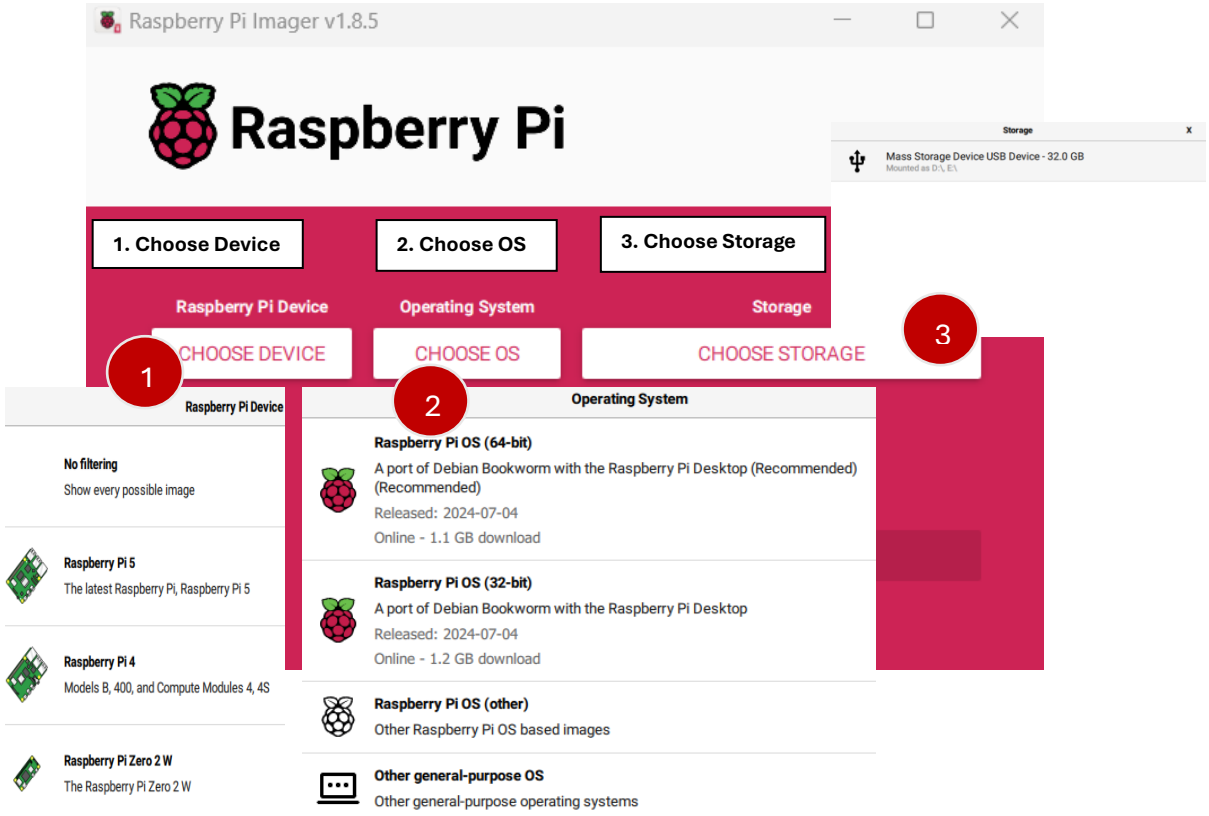
Figure Source: https://www.raspberrypi.com/software/

3.2. Open Raspberry Pi imager.

3.3. Choose a device that you are installing OS on for example (Raspberry pi 5, Raspberry Pi 4 etc.). Here, I will be using Raspberry Pi 5.

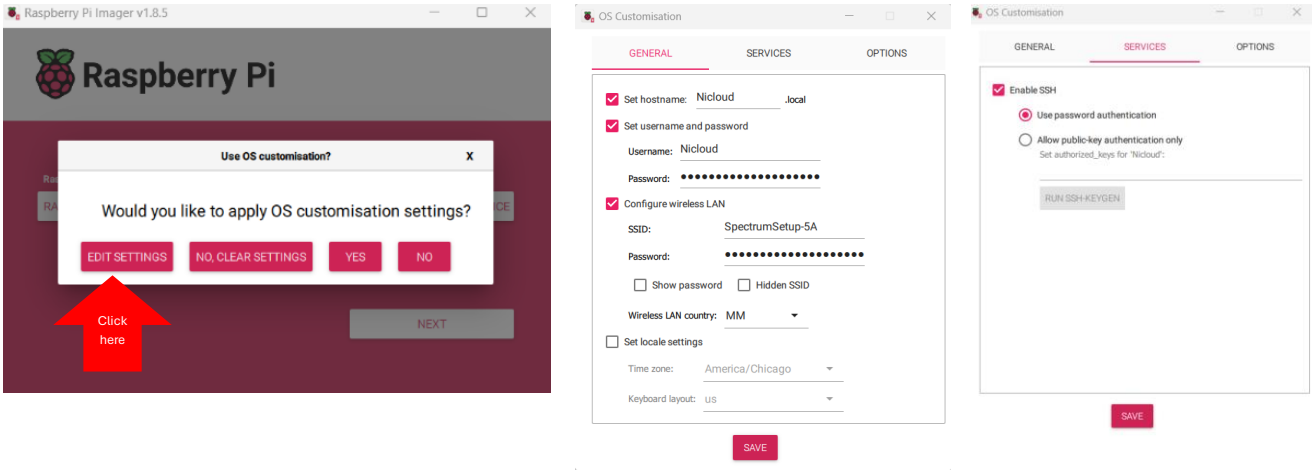
3.4. Choose which operating system you want to install (I will be installing Ubuntu Server 24.04 LTS)

3.5. Select storage device to flash OS.



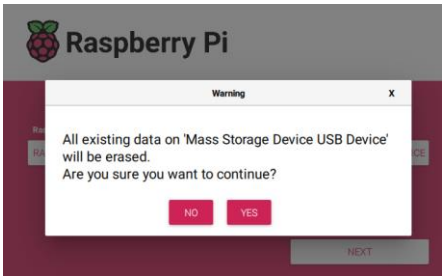
3.6. Click on 'Next'.

3.7. Click on Edit Settings to create host name, Username, configure wireless network and enable ssh.



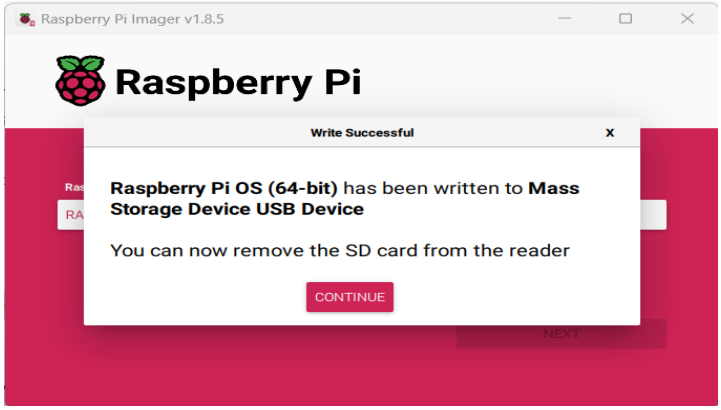
Note: Please make sure to SAVE and NOTE down edited information somewhere, we will be needing it later to establish ssh remote connection.

3.8. Start write



Note: It takes from 2-5 minutes to flash OS in the SD card depending on internet speed and device performances.

3.9. Once flashing is completed, remove the SD card from computer and insert into the Raspberry Pi



4. Establish ssh connection

4.1. Power on Raspberry pi, make sure SATA HAT is also connected to power source via Power Adapter.

4.2. To open ssh terminal from command prompt in windows, type *ssh [host name]@[user name or IP address of raspberry pi]*, hit *Enter*. Enter password and hit *Enter* again. This is where you use password created earlier while flashing OS.

```
C:\Users\nirma>ssh Nicloud@192.168.1.204
Nicloud@192.168.1.204's password:
```

4.3. OS will start installing automatically.

5. Server Configuration

5.1. Update Ubuntu Server: *sudo app update*

5.2. Upgrade apps*: sudo app upgrade*

5.3. Enable PCIe (scroll to the bottom add three lines command): *sudo nano /boot/firmware/config.txt*

Note: Dtparam=pciex1 –this command helps to enable PCIe port in SATA HAT, so the external storage is detected.

Dtparam=PCIex1_gen=3 – this command helps to enable gen 3 speed of connected storage device.

```
 GNU nano 7.2                                                    /boot/firmware/config.txt
dtparam=audio=on
dtparam=i2c_arm=on
dtparam=spi=on

# Comment out the following line if the edges of the desktop appear outside
# the edges of your display
disable_overscan=1

# If you have issues with audio, you may try uncommenting the following line
# which forces the HDMI output into HDMI mode instead of DVI (which doesn't
# support audio output)
#hdmi_drive=2

# Enable the KMS ("full" KMS) graphics overlay, leaving GPU memory as the
# default (the kernel is in control of graphics memory with full KMS)
dtoverlay=vc4-kms-v3d
disable_fw_kms_setup=1

# Enable the serial pins
enable_uart=1

# Autoload overlays for any recognized cameras or displays that are attached
# to the CSI/DSI ports. Please note this is for libcamera support, *not* for
# the legacy camera stack
camera_auto_detect=1
display_auto_detect=1

# Config settings specific to arm64
dtoverlay=dwc2

[pi4]
max_framebuffers=2
arm_boost=1

[pi3+]
# Use a smaller contiguous memory area, specifically on the 3A+ to avoid an
# OOM oops on boot. The 3B+ is also affected by this section, but it shouldn't
# cause any issues on that board
dtoverlay=vc4-kms-v3d,cma-128

[pi02]
# The Zero 2W is another 512MB board which is occasionally affected by the same
# OOM oops on boot.
dtoverlay=vc4-kms-v3d,cma-128

[cm4]
# Enable the USB2 outputs on the IO board (assuming your CM4 is plugged into
# such a board)
dtoverlay=dwc2,dr_mode=host

[all]
dtparam=pciex1
dtparam=pciex1_gen=3
```

Add these lines:
[all]
dtparam=pciex1
dtparam=pciex1_gen=3

5.4. Check for available free space on mounted disk: *df -h, (Note: my external Hard drive is /dev/sda1, yours could be different like sdb, sdc)*. It is listed because it is already partitioned, filesystem is created and is mounted. If it is not listed, start mounting process.

6. Disk Mount

6.1. First, make a directory to mount the storage device: *sudo mkdir /mnt/Storage1* (Note: my folder name is Storage1, feel free to create your own folder name.)

6.2. To mount sda1 (External Drive) to Storage1 (folder), enter command: *sudo mount /dev/sda1 /mnt/Storage1*

6.3. To auto mount sda1 every time Ubuntu Server is rebooted, we need to edit fstab, for that we will need UUID of sda1, use command: *sudo blkid* to get UUID.

```
nicloud@Nicloud:~$ sudo blkid
[sudo] password for Nicloud:
/dev/mmcblk0p1: LABEL_FATBOOT="system-boot" LABEL="system-boot" UUID="306B-9693" BLOCK_SIZE="512" TYPE="vfat" PARTUUID="9b94b926-01"
/dev/mmcblk0p2: LABEL="writable" UUID="5c5a44bd-f1a2-4f16-a2e8-89a3f10f2eb5" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="9b94b926-02"
/dev/sda1: UUID="7a8066f4-8d26-4612-b4bf-1a70d9e996c3" BLOCK_SIZE="4096" TYPE="ext4" PARTLABEL="primary" PARTUUID="25c58a13-433a-4e5f-9229-444e168852c0"
/dev/sdb1: UUID="cdc9ab0a-5678-4407-b066-1564d2121b7f" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="01ea8a2f-01"
/dev/loop0: BLOCK_SIZE="131072" TYPE="squashfs"
/dev/sdc1: UUID="d5381dd7-37f4-4b77-9507-7c4d5c2c2a9e" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="01b7654a-01"
nicloud@Nicloud:~$
```

Note: COPY the entire line of sda1 and save it somewhere (notepad)

```
/dev/sda1: UUID="7a8066f4-8d26-4612-b4bf-1a70d9e996c3" BLOCK_SIZE="4096" TYPE="ext4"
PARTLABEL="primary" PARTUUID="25c58a13-433a-4e5f-9229-444e168852c0"
```

COPY to notepad

6.4. *Open fstab with the command: sudo nano /etc/fstab* – edit fstab so the external device is auto mounted every time server reboots.

```
 GNU nano 7.2                                                                    /etc/fstab
LABEL=writable  /       ext4    defaults        0       1
LABEL=system-boot       /boot/firmware  vfat    defaults        0       1
UUID=cdc9ab0a-5678-4407-b066-1564d2121b7f       /mnt/nicloud    ext4    defaults        0       0
UUID=d5381dd7-37f4-4b77-9507-7c4d5c2c2a9e       /mnt/nicloud2   ext4    defaults        0       0
UUID=7a8066f4-8d26-4612-b4bf-1a70d9e996c3       /mnt/Storage1   ext4    defaults        0       0
```

Copy and paste UUID from notepad to here | Storage directory | Keep as it is

6.5. Reboot system: *sudo reboot*

6.6. *Now, type: df -h again to verify that the external drive is mounted and listed.*

7. Nextcloud Installation

7.1. Refer to Phase 2 project documentation.

8. Helpful References

8.1. To create Filesystem in the drive, use command: *sudo mkfs.ext4 /dev/sda1*

8.2. To handle partitions, use command: *sudo fdisk /dev/sda*

```
Nicloud@Nicloud:~$ sudo fdisk /dev/sdb

Welcome to fdisk (util-linux 2.39.3).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help):
```

Enter *m* to see help options:

```
Nicloud@Nicloud: ~
Generic
 d   delete a partition
 F   list free unpartitioned space
 l   list known partition types
 n   add a new partition
 p   print the partition table
 t   change a partition type
 v   verify the partition table
 i   print information about a partition

Misc
 m   print this menu
 x   extra functionality (experts only)

Script
 I   load disk layout from sfdisk script file
 O   dump disk layout to sfdisk script file

Save & Exit
 w   write table to disk and exit
 q   quit without saving changes

Create a new label
 g   create a new empty GPT partition table
 G   create a new empty SGI (IRIX) partition table
 o   create a new empty MBR (DOS) partition table
 s   create a new empty Sun partition table


Command (m for help):
```

8.2.1. To delete partitions, enter *d*. Then, enter the partition number that needs to be deleted.
Enter *w* to write changes.

8.2.2. To create new partitions, enter *n*

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p):
```

- Enter *p* to create primary partitions

```
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):
```

- Assign partition number. Just hit enter to assign default partition number.

```
Partition number (1-4, default 1): 1
First sector (2048-976773167, default 2048):
```

- First and Last sector: Just hit enter to assign default value

```
First sector (2048-976773167, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-976773167, default 976773167):

Created a new partition 1 of type 'Linux' and of size 465.8 GiB.

Command (m for help):
```

- Hit enter again, new partition is created.
- Enter *w* to write partition.