



American International University-Bangladesh

Advanced Operating System

Project Title: Media Server

Section: A

| No | Student Name | Student Id |
|----|--------------------------|------------|
| 1 | Hasan, S M Mahmudul | 17-35752-3 |
| 2 | Rafa, Humayara Chowdhury | 17-35413-3 |
| 3 | Mowmita, Nazia Afsan | 18-37787-2 |
| 4 | Uddin, Md. Sazid | 18-37712-1 |
| 5 | Islam, Asika | 18-37421-1 |
| 6 | Hasan, Md. Pial | 18-37671-1 |
| 7 | Das, Bidhan | 15-29014-1 |

GUIDED BY

ARIFUR RAHMAN

Department of Computer Science

Project Idea

The idea is to create a media server so that devices on the same network can play movies, music and other type of media from a central location. Goal is very simple, which is to create a fast and ready to stream across home network easily. This reduces any hassle of loading of media files in every device. With the help of a media server all the family members can enjoy their personal favorite shows at the same time while using a home server of their own.

Component List

1. Raspberry Pi 4 Model B x1
2. Micro SD Card 16GB x1
3. USB 3.0 cable x1
4. Ethernet cable x1

Project Implementation Description

We will create something similar to a NAS (Network Attached Storage) [1] and use Kodi [2] which is an open-source media server software, media player. On our Raspberry Pi 4 we will setup NAS by creating a Samba server [3]. We will make all our USB devices which are plugged into Raspberry Pi to mount in a common location. And share that location using Samba server. Then installing Kodi on other devices of the home network and adding our samba server on the Raspberry Pi will do the work. We will cover how to setup our Raspberry Pi for the job and also demonstrate how to play the media from the server on a Windows machine.

We will first setup a headless Raspberry Pi [4], that means without any monitor using ethernet connection. And later we will enable Wi-Fi so the Raspberry Pi can connect to the home network automatically.

We will execute a script to watch for new USB device insertion and the script will mount the devices to a common location. For our case, we will create a directory named `smb-devices` and the script will mount a USB device to `smb-devices/storage0`. If a new device is added then the new devices will be mounted on `smb-devices/storage1`. The script will keep records of each mounted device. And once there's at least one mounted device, the script will check the records and will automatically unmount an inactive mount-point. Because users will just plug and unplug their USB devices, pulling out the device without prior unmounting will make that device's mount location busy. So, the script will if there's any mount location busy but not actually used, it will free up that mount-point and mount any newly plugged device to that location.

Let's see how to setup things in our Raspberry Pi,

NB: Make sure you do not switch to other account; setup procedure must be done within the default `ubuntu` user as our python script has absolute path of different directories for the sake of consistency.

1. At first please download latest version of Ubuntu server LTS for Raspberry Pi. We are using Ubuntu 20.04.1 LTS.

2. Then please download Raspberry Pi Imager and write the Ubuntu image to a microSD card (16GB recommended).
3. After it's done, please remove the microSD card reader from your computer and plug it again.
4. Now once it's ready, create an empty file named 'ssh' with no file extension. [5]
5. Then eject the microSD card, insert it in your Raspberry Pi 4 and connect the Pi to your router using an ethernet cable.
6. Now, please connect Raspberry Pi to a power source and it will boot up.
7. Please go to your router settings from your web browser and check the IP address of your Raspberry Pi. [6]
8. Now please open a command prompt or terminal depending on your OS and execute like this,

```
ssh ubuntu@<your_ip>
```

Example, ssh ubuntu@172.16.15.5

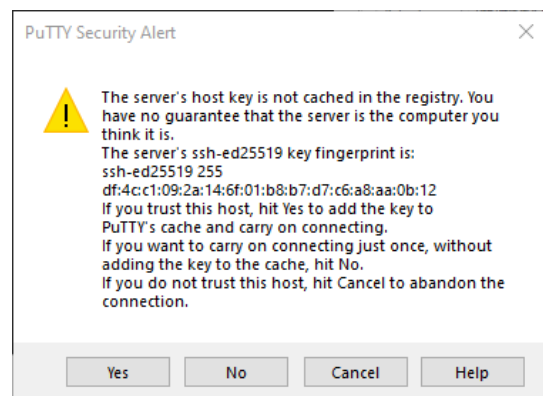
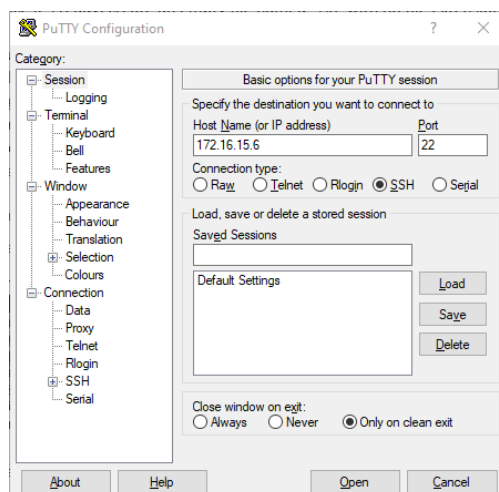
(demonstrated on Windows 10),

```
C:\Users\>ssh ubuntu@172.16.15.5
The authenticity of host '172.16.15.5 (172.16.15.5)' can't be established.
ECDSA key fingerprint is SHA256:ShstOC04L8N0avxr3o5FepfZynC0AZBZKpEc8qVbaPA.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.15.5' (ECDSA) to the list of known hosts.
ubuntu@172.16.15.5's password:
```

Please note that you have to enter 'yes' for the first time, it will add the device to your 'known_hosts' and won't ask for confirmation later.

Also, default account in ubuntu server is ubuntu and password is also 'ubuntu'. [7]

Or you can also use third party tools like Putty,



```
172.16.15.6 - PuTTY
login as: ubuntu
ubuntu@172.16.15.6's password: 
```

```
ubuntu@ubuntu: ~
* Support: https://ubuntu.com/advantage

System information as of Sun Sep 20 15:02:52 +06 2020

System load: 0.17          Temperature: 45.8 C
Usage of /: 20.1% of 14.23GB Processes: 140
Memory usage: 7%          Users logged in: 1
Swap usage: 0%           IPv4 address for wlan0: 172.16.15.6

* Kubernetes 1.19 is out! Get it in one command with:

    sudo snap install microk8s --channel=1.19 --classic

https://microk8s.io/ has docs and details.

1 update can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Last login: Sun Sep 20 15:00:27 2020 from 172.16.15.4
ubuntu@ubuntu:~$ 
```

9. Once you login, you will be asked to change the default password, please change it to whatever you like.
10. After you have changed your password, server will automatically log you out. Please login again.
11. Now before doing anything on the system, please check your time and fix if it doesn't match with your current time. Execute 'date' to see the current time and date. If wrong, please execute,
sudo dpkg-reconfigure tzdata

And select your appropriate region. If it still doesn't solve your issue, please execute,
sudo apt install ntp

sudo service ntp stop

sudo ntpd -gq

sudo service ntp start

This will install 'ntp' [8] and fix your time synchronize your time with Internet. Incorrect time will cause problem later.

12. Then please edit the 'sudoer' file so that executing sudo commands doesn't ask for password. This is crucial for the python script (will come later). Follow these steps,
A) 'sudo visudo' to open the file

B) add '\$USER ALL=(ALL) NOPASSWD: ALL' at the bottom of the file like this where \$USER is your username.

C) Please save (ctrl + s) and exit (ctrl + x). Now whatever you execute with sudo, it won't ask password for your user next time you execute command with 'sudo'.

13. Then please update and upgrade your system before we install and configure anything, please execute,
sudo apt update && sudo apt upgrade -y

14. Now, after the successful upgrade, most likely you will have 'python3' and 'tmux' installed. So, please just install pip (python3-pip) with,
sudo apt install python3-pip

You can verify whether you have a package installed or not with,
apt list --installed | grep '<package>'

Example, apt list --installed | grep 'tmux'

If you have the package installed you will see it appear on terminal. You can also check it with 'whereis tmux'.

15. After you have installed pip (pip3), which is a package manager for Python, please install 'pyudev',
sudo pip3 install pyudev

16. Then please create a folder on your home directory named 'smb-devices'.

17. Now, please install samba with,
sudo apt install samba

18. After that please open your samba configuration with,
sudo nano /etc/samba/smb.conf

19. Now let's add a new config, please add something similar to this at the end of the file,
[<your_server_name>]
comment = <your_comment>
path = /home/ubuntu/smb-devices
read only = yes
browsable = yes

Example,

```
[my-server]
comment = SMB on RPi
path = /home/ubuntu/smb-devices
read only = yes
browsable = yes
```

20. Then please restart samba service with,
sudo service smbd restart

21. Now, please let samba password for your user account, execute,
sudo smbpasswd -a ubuntu

22. Then please create a script with ``sudo nano mount_usb.sh`` and paste the following contents,

```
#!/bin/sh
```

```
PATH=/usr/bin/
```

```
tmux new-session -d -s usb_watcher '/usr/bin/python3.8 /home/ubuntu/watch_driver.py'
```

Hosted at GitHub: <https://l.iamlizu.com/mountUSBScript>.

23. After that, please execute the following command to add executable permission to the shell script,

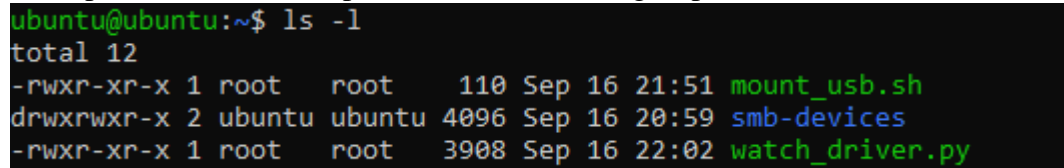
```
sudo chmod +x mount_usb.sh
```

24. Then please create a python script with `'sudo nano watch_driver.py'` and please copy the contents from <http://l.iamlizu.com/watchDriver>.

25. Again, we will have to give this script executable permission, please execute the following command to do so,

```
sudo chmod +x watch_driver.py
```

26. Please verify again that both of your script has executable permission, 'x' on the very left represents executable permission for owner, group and others,



```
ubuntu@ubuntu:~$ ls -l
total 12
-rwxr-xr-x 1 root  root   110 Sep 16 21:51 mount_usb.sh
drwxrwxr-x 2 ubuntu ubuntu 4096 Sep 16 20:59 smb-devices
-rwxr-xr-x 1 root  root  3908 Sep 16 22:02 watch_driver.py
```

27. Now you can run the script with the following command,

```
./mount_usb.sh
```

28. This script will create a tmux session named 'usb_watcher', you will see something like this if you execute 'tmux ls',

```
usb_watcher: 1 windows (created Wed Sep 16 22:06:47 2020)
```

You can attach to this session using the following command (however this won't be necessary),

```
tmux attach-session -t usb_watcher
```

[Initially you will see nothing there until you plug any USB]

To detach the session or get into your normal terminal, please press 'ctrl+b' then press 'd'.

29. Now, let's setup Wi-Fi so Raspberry Pi can automatically connect to your home network. For that, please execute the following command to check wireless network interface,

```
ls /sys/class/net
```

Your wireless interface will be something similar to 'wlan0'

30. After that, please find out your netplan, by executing `'ls /etc/netplan/'`, you will see something like '01-network-manager-all.yaml' or '50-cloud-init.yaml'. [9]

31. Now, please open your net configuration file like this,

```
sudo nano /etc/netplan/<your_config.yaml>
```

Example, `sudo nano /etc/netplan/50-cloud-init.yaml`

32. Then, please enter your Wi-Fi credentials, like this bellow,

```
network:
  ethernets:
    eth0:
      dhcp4: true
      optional: true
  version: 2
  wifis:
    wlan0:
      optional: true
      access-points:
        "SSID-NAME-HERE":
          password: "PASSWORD-HERE"
      dhcp4: true
```

33. Now, please save and close that file with ‘ctrl + s’, followed by ‘ctrl + x’.
34. After that, to make above changes to persistent on reboot, please create another file with,
`sudo nano /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg`

And write the following contents,
 network: {config: disabled}

35. Now please go to your router settings and bind your Raspberry Pi to a specific IP address [10]. So that, each time your Raspberry PI is connected to the network it gets same IP address. This is very crucial for later stages. And because binding procedure can vary from router to router, please search on google how to do it for your system. The following shows a configured setting in a NETGEAR WNR614,

The screenshot shows the NETGEAR genie WNR614 router settings page. The 'LAN Setup' tab is selected. Under 'LAN TCP/IP Setup', the 'Use Router as DHCP Server' checkbox is checked. The 'Starting IP Address' is 172.16.15.2 and the 'Ending IP Address' is 172.16.15.254. In the 'Address Reservation' table, a reservation is shown for device 'RPI 4' with IP address '172.16.15.6' and MAC address 'DC:A6:32:75:60:35'.

| # | IP Address | Device Name | MAC Address |
|---|-------------|-------------|-------------------|
| 1 | 172.16.15.6 | RPI 4 | DC:A6:32:75:60:35 |

36. You can now reboot your system.

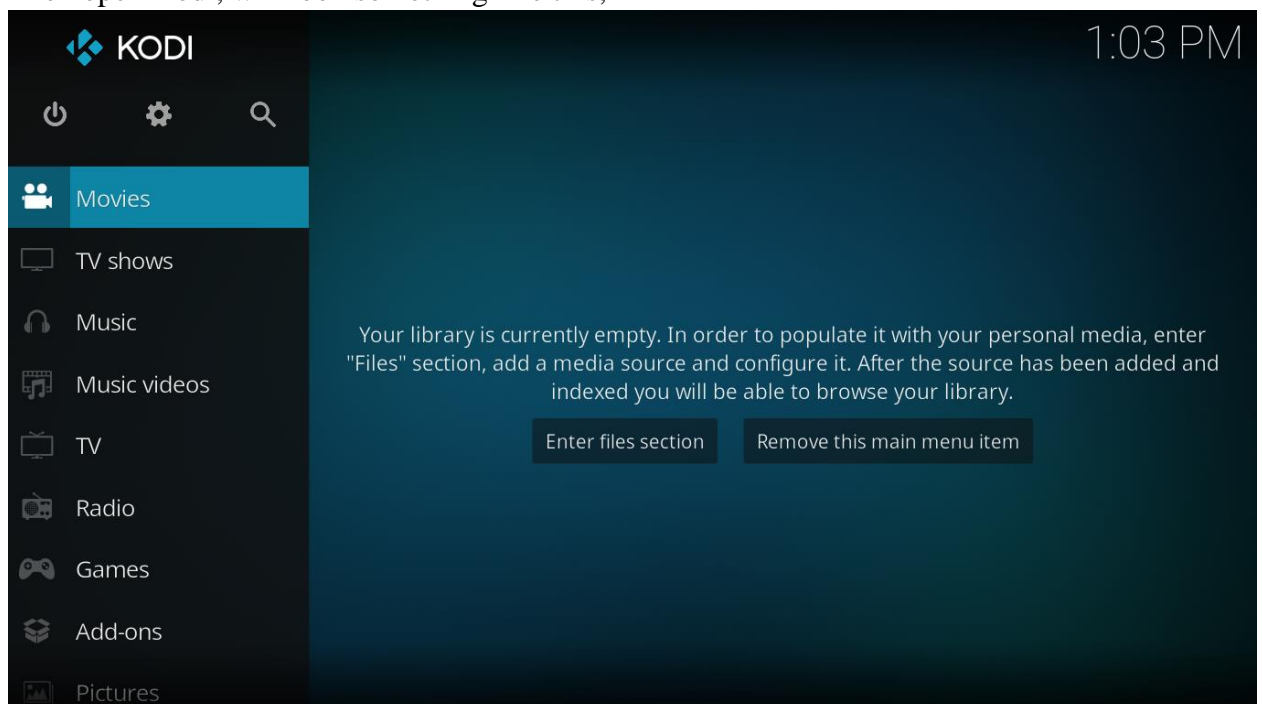
That's it! You are done setting up Raspberry Pi!

NB: You will have to run the script (./mount_usb.sh) manually every time you start or restart your Raspberry Pi. As of now, it has to auto start function implemented. So, just like we did a while ago, you will have to login your account using SSH and initiate the script before inserting any USB.

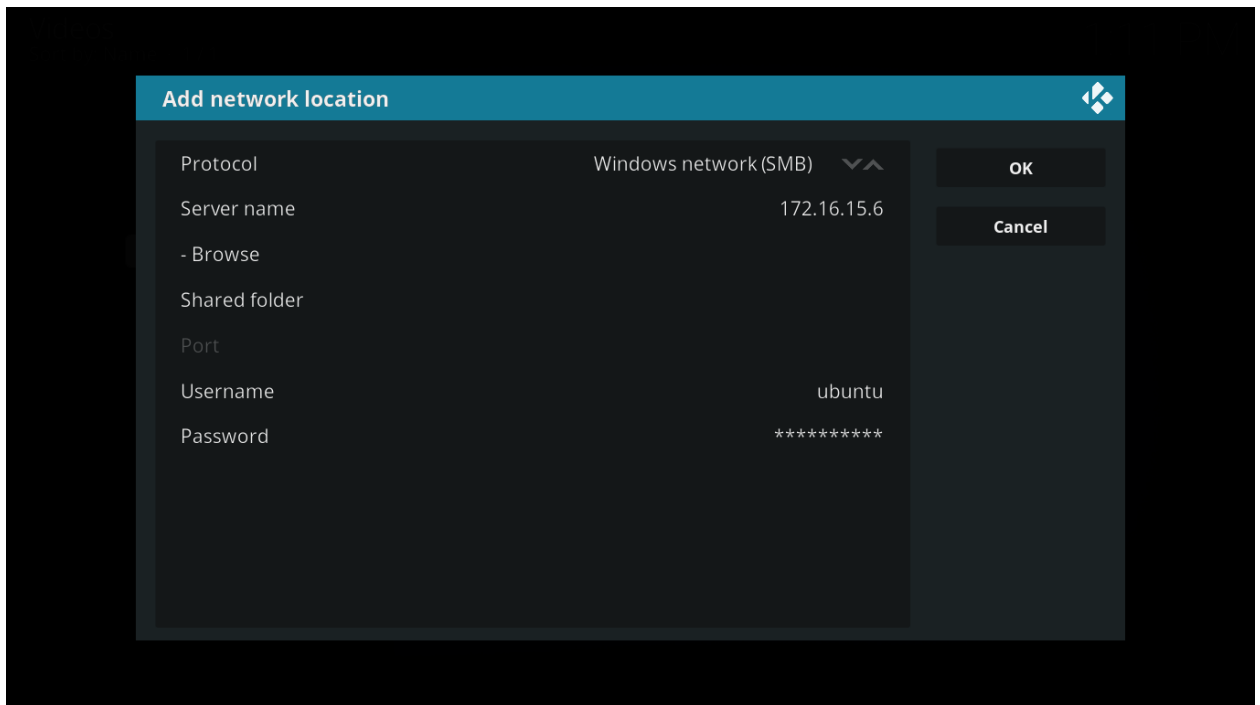
Now, please plug a USB device to the Raspberry Pi and follow us to see how to configure our samba server on a client device and play media.

Before getting started, Let's add some movies, plug your USB storage into Raspberry Pi and wait a while (15s). We have set a grace period to give the USB storage some time to get ready. Following steps are demonstrated on Windows 10,

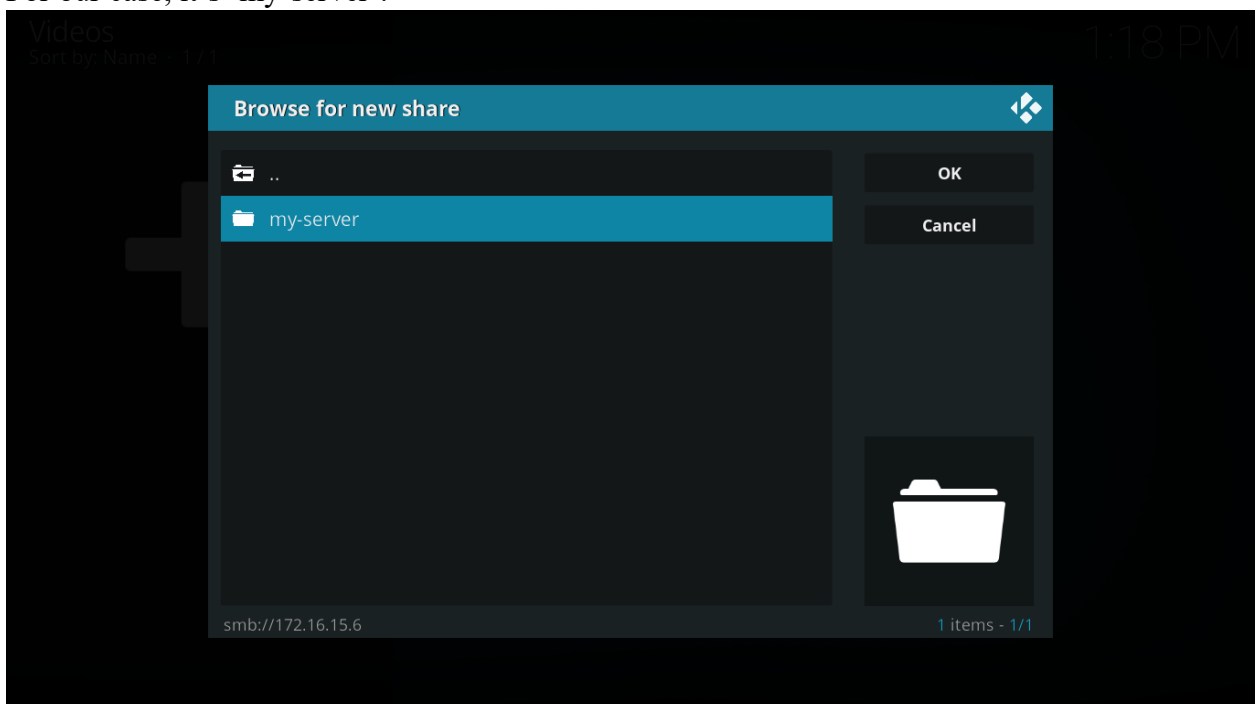
1. At first, please install latest version of Kodi.
2. Then open Kodi, will look something like this,



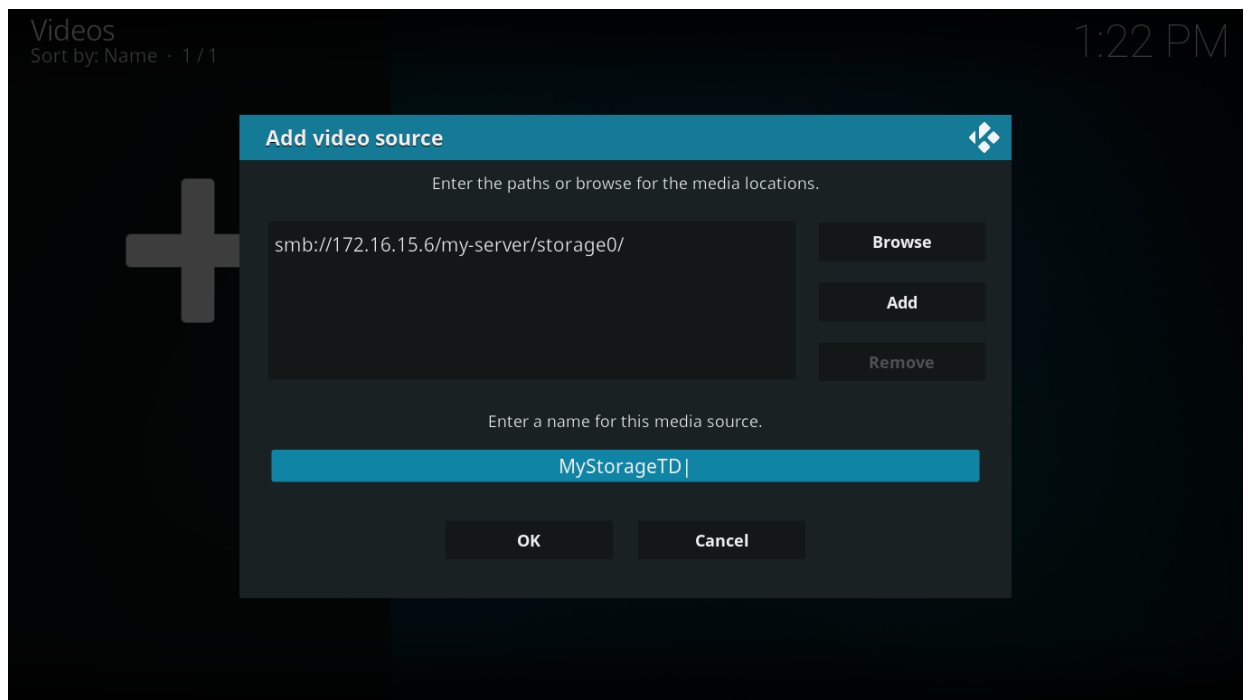
3. Then please click 'Enter files section' and select 'Add Videos', and from there click 'Browse'.
4. Now please scroll down till you see 'Add network location'.
5. After that, write your Raspberry Pi's IP address in the 'Server name' field which you already know or can check from your router settings, 'ubuntu' in the username field, and the smbpasswd in Password field, the on that you set while setting up samba. It will look something like this bellow and be sure the Protocol is 'Windows network (SMB)',



6. After that, please hit 'OK'.
7. Now, please select the network you just added from the list, you will see a directory appear, the directory name should be one you specified as the server name of samba. For our case, it's 'my-server'.



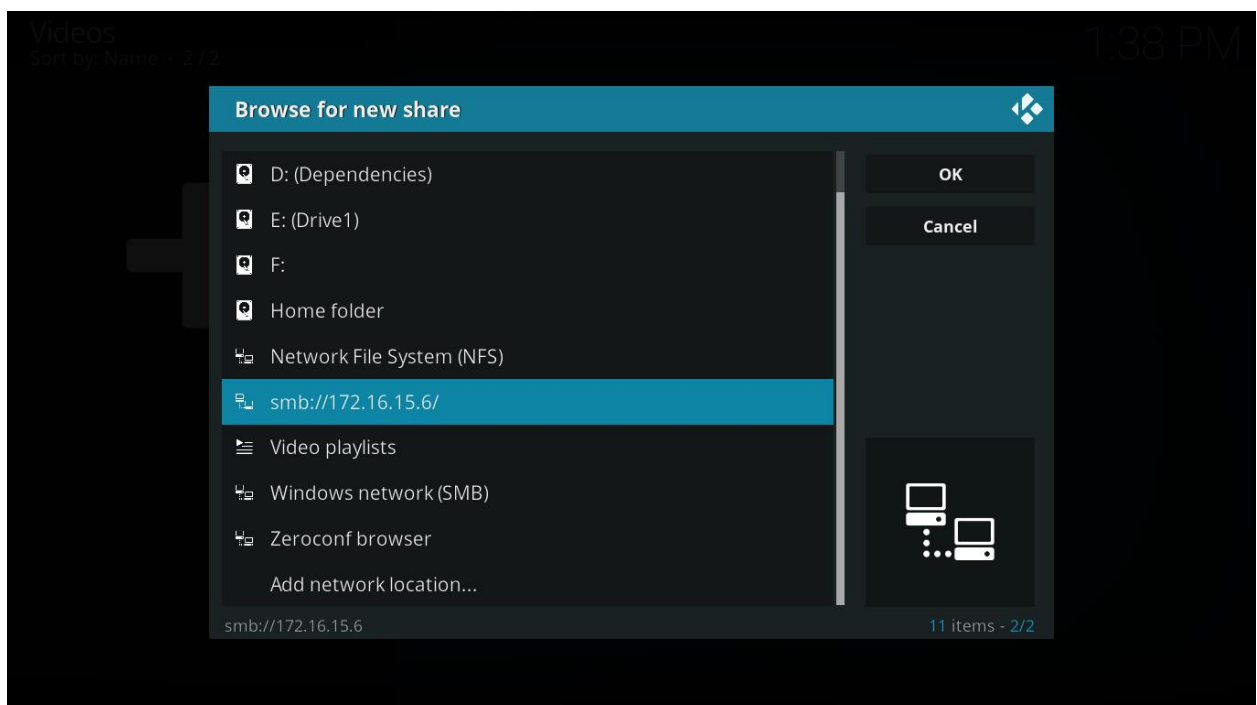
8. Then, after selecting your server, you will see storage 0. Please select that and press 'OK'.
9. Rename your storage to be saved as, as you like, and hit 'OK' again.



10. Please specify what the directory contains, and hit 'OK' again.
11. After that please confirm to refresh your library and your movies will be added!
12. Now you can watch and enjoy your movies without having to load them in your system.

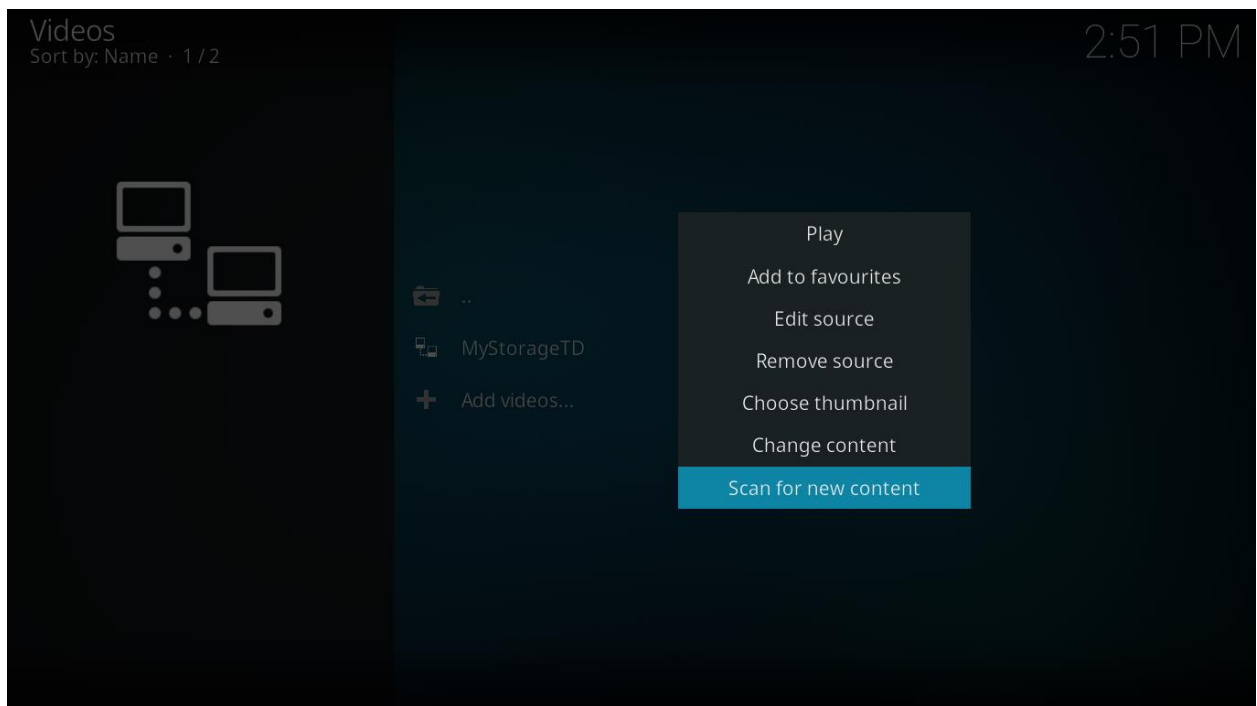
Please make sure to follow this process on each device you want to play media from the media server. You will only have to configure once.

If you plug any new storage device to Raspberry Pi, please click Movies from the main left bar menu and navigate to 'Files', then click 'Add videos', but this time instead of adding a new network location, select your preadded Raspberry Pi server like this bellow and select the new storage device.

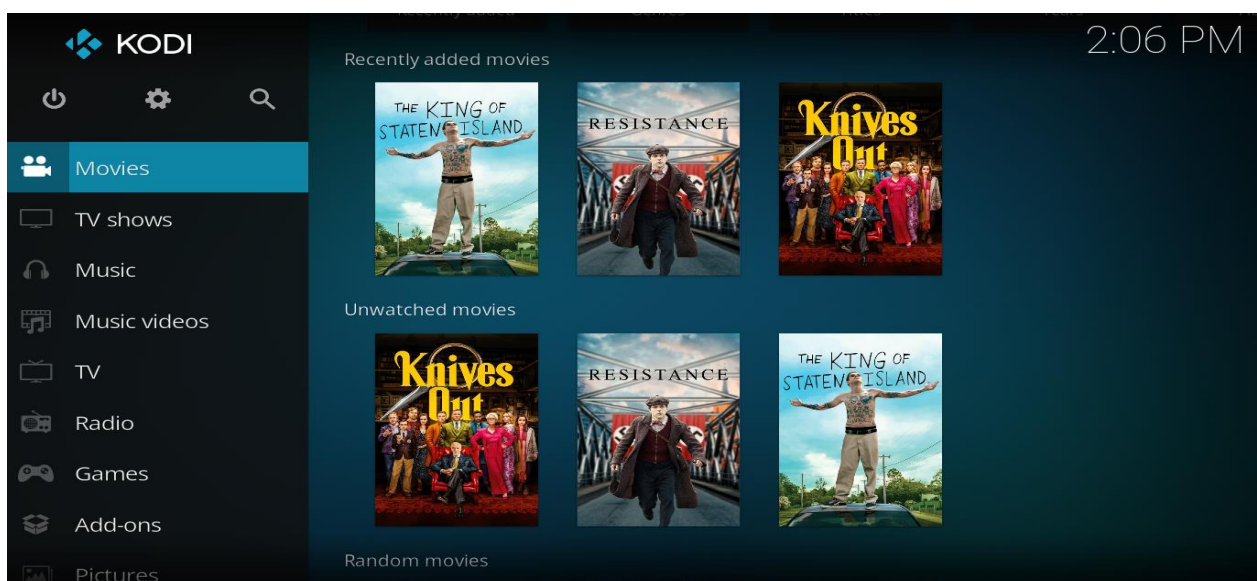


NB: You will just have to add whatever your storages appear as; you know the scripts will automatically mount them dynamically. For example, if you have two storage mounted one of your storage is mounted as storage0, and one of them gets unplugged and plugged again, then this might get mounted on a new location like storage2. So, please do not just unplug your storages, if you do, be sure to readd them and remove the old obsolete source. Hence, unnecessary removal and adding of storage is depreciated.

If you add additional content to your storage, navigate to the source and click ‘scan for new content’, this will add the new contents.



Now it's time to watch some movies, click whatever you like to play and enjoy! Following image shows example of the Kodi UI after adding movies,



Implemented Code

1. mount_usb.sh, used to create 'tmux' session that fires up the python script 'watch_driver.py' within.

```
#!/bin/sh
PATH=/usr/bin/
tmux new-session -d -s usb_watcher '/usr/bin/python3.8
/home/ubuntu/watch_driver.py'
```

Hosted at GitHub: <https://l.iamlizu.com/mountUSBScript>

2. watch_driver.py, watches for new USB devices and mount them in a common location. This script is smart enough to unmount any mount-point that is not used anymore (user unplugs a device).

Hosted at GitHub: <https://l.iamlizu.com/watchDriver>

```

import pyudev # python library for libudev which is api for enumerating and introspecting local devices
import time # helps playing with time
import os # helps playing with operating system itself
import subprocess # helps playing with processes

context = pyudev.Context()
monitor = pyudev.Monitor.from_netlink(context)
monitor.filter_by(subsystem='usb')

i = 0 # to count storage devices
mount_locs = [] # stores mountpoints of already mounted usb

for device in iter(monitor.poll, None):
    if device.action == 'add':
        time.sleep(15) # giving some time to get the device ready
        usb_devices = [] # stores inserted devices 'partitions'

        for device in context.list_devices(MAJOR='8', subsystem='block'): # looping through usb partitions
            if (device.device_type == 'partition'):
                usb_devices.append("{}".format(device.device_node)) # adding a partition to usb_devices list
                print("Found {}".format(device.device_node))

        if len(usb_devices) != 0:
            for stick in usb_devices:
                # finding mountpoint of a partition
                execute = 'lsblk -o name,mountpoint | grep {}'.format(stick[-4:])
                output = subprocess.check_output(execute, shell=True)

                # if partition not mounted, then,
                if b'/home/ubuntu/smb-devices/storage' not in output: # b'string' because what we have in
output is bytecode
                    if len(mount_locs) >= 1:
                        for m in mount_locs:
                            # checking for not used mountpoints
                            check = "ls -l {} || echo 'Input/output error'".format(m)

                            os.system(check + "> /home/ubuntu/mount_loc_output.txt")
                            if os.path.exists('/home/ubuntu/mount_loc_output.txt'):
                                dump = open('/home/ubuntu/mount_loc_output.txt')
                                error = dump.read()
                                dump.close()
                                os.remove('/home/ubuntu/mount_loc_output.txt')

                            # if unused mountpoint found, unmount that

```

References

- [1] Wikipedia, "Network-attached storage - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Network-attached_storage. [Accessed 17 September 2020].
- [2] "Open Source Home Theater Software," Kodi, [Online]. Available: <https://kodi.tv/>. [Accessed 17 September 2020].
- [3] Canonical, "Install and Configure Samba | Ubuntu," [Online]. Available: <https://ubuntu.com/tutorials/install-and-configure-samba#1-overview>. [Accessed 17 September 2020].
- [4] Raspberry Pi Foundation UK, "Setting up a Raspberry Pi headless - Raspberry Pi Documentation," [Online]. Available: <https://www.raspberrypi.org/documentation/configuration/wireless/headless.md>. [Accessed 17 September 2020].
- [5] Raspberry Pi Foundation UK, "SSH (Secure Shell) – Raspberry Pi Documentation," [Online]. Available: <https://www.raspberrypi.org/documentation/remote-access/ssh/README.md>. [Accessed 17 September 2020].
- [6] Raspberry Pi Foundation UK, "IP Address – Raspberry Pi Documentation," [Online]. Available: <https://www.raspberrypi.org/documentation/remote-access/ip-address.md>. [Accessed 17 September 2020].
- [7] Canonical, "ARM/Raspberry Pi – Ubuntu Wiki.," [Online]. Available: <https://wiki.ubuntu.com/ARM/RaspberryPi>. [Accessed 17 September 2020].
- [8] Wikipedia, "Network Time Protocol – Wikipedia.," [Online]. Available: https://en.wikipedia.org/wiki/Network_Time_Protocol. [Accessed 17 September 2020].
- [9] Canonical, "Ubuntu Manpage: netplan - YAML network configuration abstraction for various backends.," [Online]. Available: <https://wiki.ubuntu.com/ARM/RaspberryPi>. [Accessed 17 September 2020].
- [10] NETGEAR, "How do I reserve an IP address on my NETGEAR router? | NETGEAR Support.," [Online]. Available: <https://kb.netgear.com/25722/How-do-I-reserve-an-IP-address-on-my-NETGEAR-router>. [Accessed 17 September 2020].