nishantnadkarni.tech

Arch Linux with BTRFS Installation (Base)

Introduction

In this guide, I will show steps to install Arch Linux with the BTRFS file system with several different subvolumes.

What is Arch Linux?

Arch Linux is a rolling release linux distribution for x86-64 computers. It is widely popular due to it's ease of use and simplicity. You can read more about Arch Linux here : https://wiki.archlinux.org/index.php/Arch_Linux

Why Btrfs?

Btrfs is a modern copy on write (CoW) filesystem for Linux aimed at implementing advanced features while also focusing on fault tolerance, repair and easy administration. It's biggest boon is enabling users to take system snapshots that do not take time to create or restore and barely take up any space on your system.

Since Arch is soo bleeding edge, it is possible that once in a while, due to an update your system breaks. Whenever such faults happen, you can simply roll back on the last snapshot in seconds if you have a BTRFs file system in place.

Step 7: Creating Filesystems

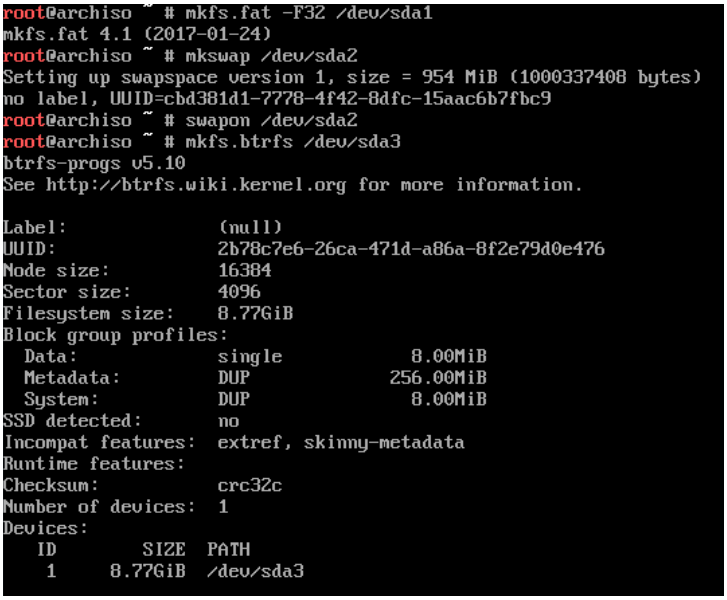
Now we must format the partitons with the respective file systems.

We need FAT32 file system for /boot:

For /swap partition, we need to make the partition and activate swap so:

```
mkswap /dev/sda2
swapon /dev/sda2
```

For /(root), we need to make with the btrfs file system:



Step 8: Mounting the partitions and subvolumes

Now we must mount the partitions that we just created (except swap as it is not used to store static files).

Now that we have mounted the root subvolume, we must create subvolumes for btrfs.

We create subvolumes to better organize our data and to exclude them from btrfs snapshots. Also, if you're using multiple disks for a single OS (eg. Windows C: and D: drives are on different disks), subvolumes enable you to store even system files on another directory. On my personal setup, I have the @var and @tmp subvolumes on my HDD so as to save space on my SSD where Arch is installed.

```
btrfs subvolume create /mnt/@
btrfs subvolume create /mnt/@home
btrfs subvolume create /mnt/@var
btrfs subvolume create /mnt/@opt
btrfs subvolume create /mnt/@tmp
btrfs subvolume create /mnt/@.snapshots
umount /mnt
```

These subvolumes are mainly named after system directories which have specific functions:

- @ - This is the main root subvolume on top of which all subvolumes will be mounted.
- @home - This is the home directory. This consists of most of your data including Desktop and Downloads.
- @var - Contains logs, temp. files, caches, games, etc.
- @opt - Contains third party products
- @temp - Contains certain temperory files and caches
- @.snapshots - Directory to store snapshots for snapper (Can exclude this if you plan on using Timeshift)

Now to mount these partitions:

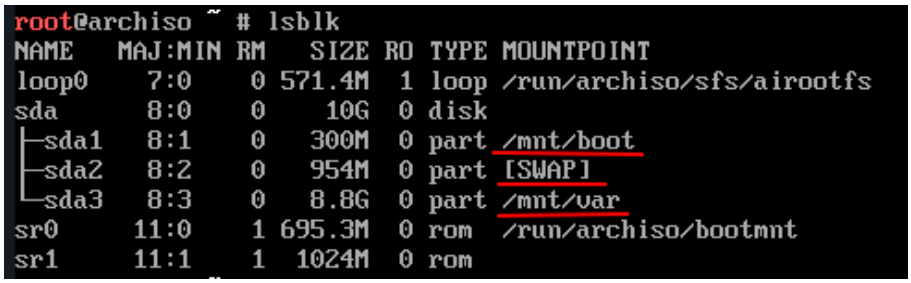
```
mount -o noatime,commit=120,compress=zstd,space_cache,subvol=@ /dev/sda3 /mnt
# You need to manually create folder to mount the other subvolumes at
mkdir /mnt/{boot,home,var,opt,tmp,.snapshots}

mount -o noatime,commit=120,compress=zstd,space_cache,subvol=@home /dev/sda3
/mnt/home

mount -o noatime,commit=120,compress=zstd,space_cache,subvol=@opt /dev/sda3
/mnt/opt

mount -o noatime,commit=120,compress=zstd,space_cache,subvol=@tmp /dev/sda3
/mnt/tmp

mount -o noatime,commit=120,compress=zstd,space_cache,subvol=@.snapshots
/dev/sda3 /mnt/.snapshots

mount -o subvol=@var /dev/sda3 /mnt/var
# Mounting the boot partition at /boot folder
mount /dev/sda1 /mnt/boot
```

Btrfs options:

- noatime - No access time. Improves system performace by not writing time when thefile was accessed.
- commit - Peridoic interval (in sec) in which data is synchronized to permanent storage.
- compress - Choosing the algorithm for compress. I have set zstd as it has good compression level and speed.
- space_cache - Enables kernel to know where block of free space is on a disk to enable it to write data immediately after file creation.
- subvol - Choosing the subvol to mount.

You can read more about btrfs mount options here:
https://btrfs.wiki.kernel.org/index.php/Manpage/btrfs(5)

Verify that you have mounted everything correctly:



The mountpoints show the last subvolume that you mounted.

Step 9: Installing the base system

For intel CPUs:

```
pacstrap /mnt base linux linux-firmware nano intel-ucode btrfs-progs
```

For AMD CPUs:

```
pacstrap /mnt base linux linux-firmware nano amd-ucode btrfs-progs
```

For VMs:

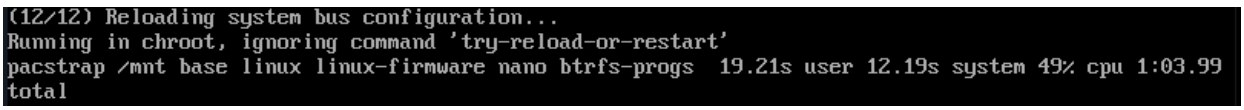
```
pacstrap /mnt base linux linux-firmware nano btrfs-progs
```

Type ‘y’ when asked for confirmation.

pacstrap will install the packages mentioned on the newly made root partition. Packages installed:

- base - Base linux system
- linux - Latest linux kernal and modules(You can replace with with linux-lts if you want a more stable kernel)
- linux-firmware - Firmware files for linux (You can skip this in a vm)
- nano - A simple terminal based text editor
- intel-ucode - Microcode update files for Intel CPUs
- amd-ucode - Microcode update image for AMD CPUs
- btrfs-progs - Btrfs filesystem utilities

Depending on your internet speed, this step might take time. You should see something like this when installation is done:

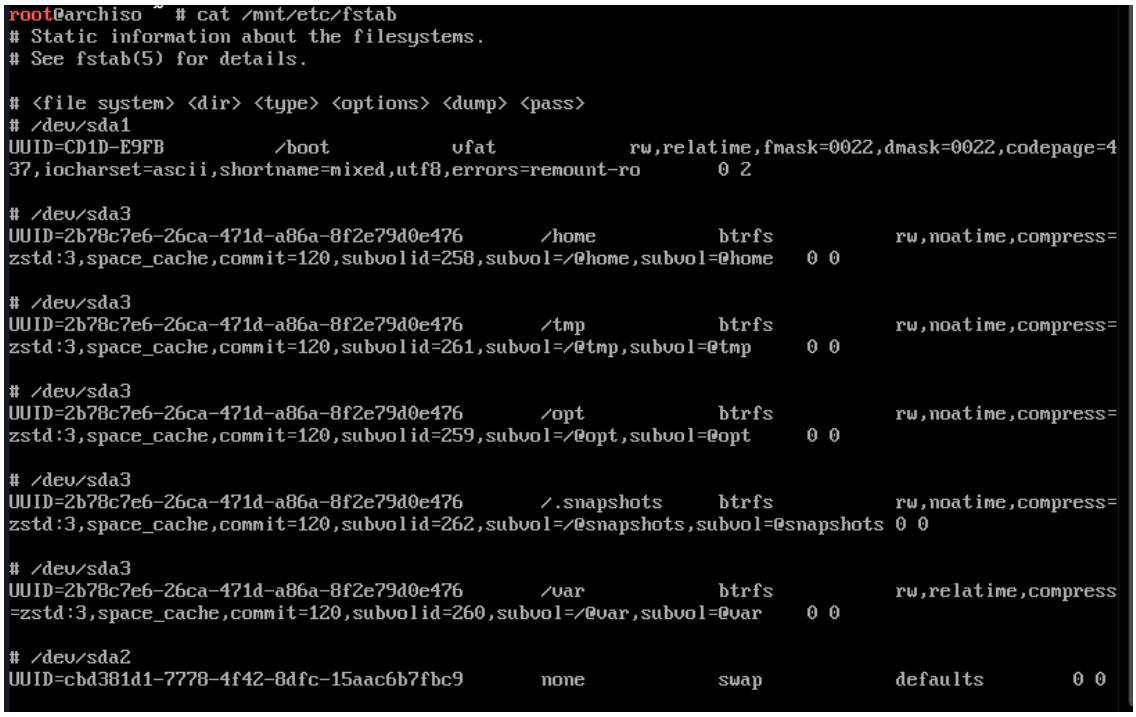


Step 10: Generate fstab

After installation of all packages is done, we need to now generate the fstab. The fstab file is used to define how disk partitions, various other block devices, or remote filesystems should be mounted into the filesystem. Generate it using:

```
genfstab -U /mnt >> /mnt/etc/fstab
```

Verify fstab entries by, It should look something like this:



Step 11: Chroot into install

Now you must enter your Arch install to set it up:

Step 12: Seting timezone

You set timezone using:

```
ln -sf /usr/share/zoneinfo/Region/City /etc/localtime
```

Replace region and city with your own timezone. In my case it will be:

```
ln -sf /usr/share/zoneinfo/Asia/Kolkata /etc/localtime
```

You can list all timezones with:

```
timedatectl list-timezones
```

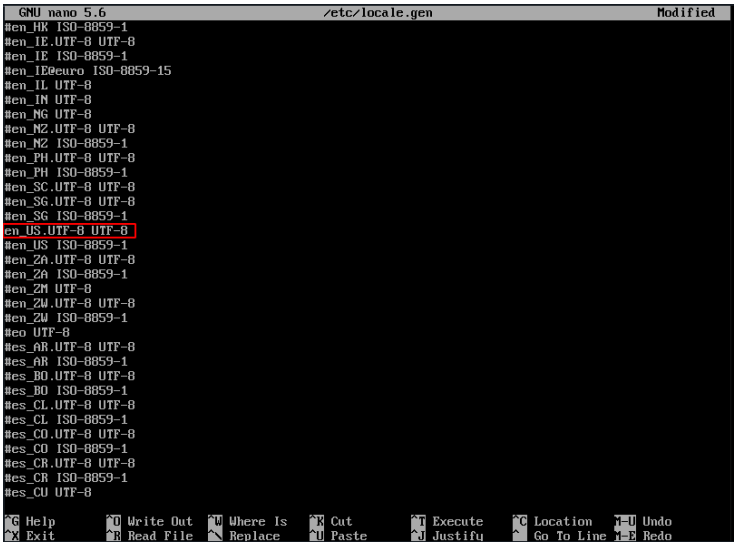
Press ‘q’ to quit list

Now to sync hardware and system clock:

Step 13: Setting System Locale

You need to manually edit a file for this:

You need to scroll down and uncomment the language you want. For me, since I want English US, I will scroll down and uncomment that:



After you uncomment, press Ctrl+O and then Enter to save and Ctrl+X to exit.

Now generate locales:

Now we set locale in locale.conf file:

```
echo LANG=en_US.UTF-8 >> /etc/locale.conf
```

If you choose a different language, replace en_US.UTF-8 with your language.

Step 14: Setting Keymap (Only if you did Step 3)

If you changed your keymap in step 3, you need to add it here also:

```
echo KEYMAP=[keymap] >> /etc/vconsole.conf
```

Replace [keymap] with your specific keymap.

Step 15: Network Configuration

We now need to set our Hostname

```
echo Arch-VM >> /etc/hostname
```

Replace Arch-VM with whatever hostname you wish to set.

Now for the hostfiles:

Arch Wiki states the format for this:

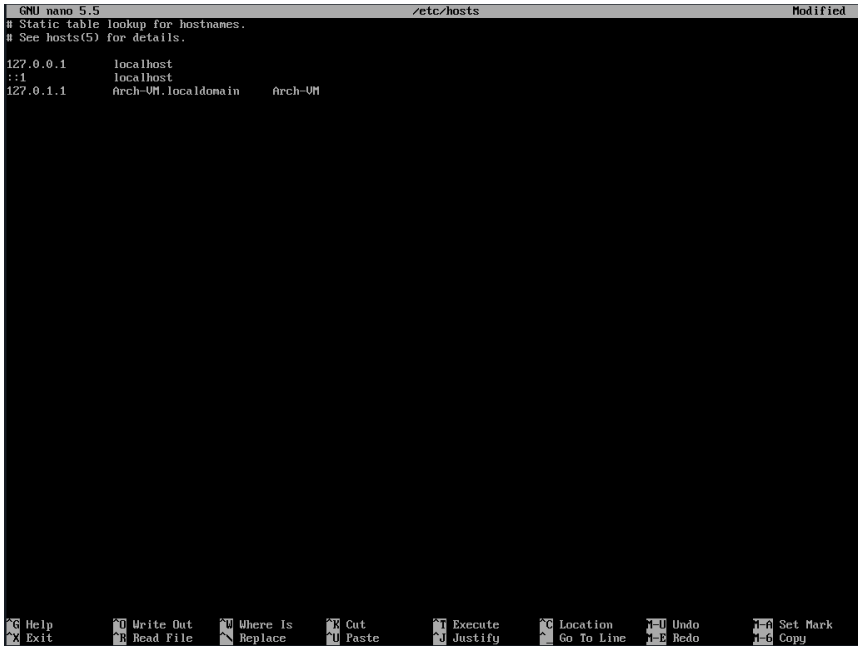
```
127.0.0.1       localhost
::1             localhost
127.0.1.1       myhostname.localdomain  myhostname
```

So in my case, I will add: After you add, press Ctrl+O and then Enter to save and Ctrl+X to exit.

Step 16: Setting password for root user

Enter your password twice to set root password.

> **Note**: In linux, visual feedback for passwords is disabled for better security.



Step 17: Installing remaining essential packages

```
pacman -S grub grub-btrfs efibootmgr base-devel linux-headers networkmanager network-manager-applet wpa_supplicant dialog os-prober mtools dosfstools reflector git
```

These are some basic sets of packages you will need if you plan to use Arch in the long run. I would recommend that you google all packages to understand what they do.

Additional things you can add:

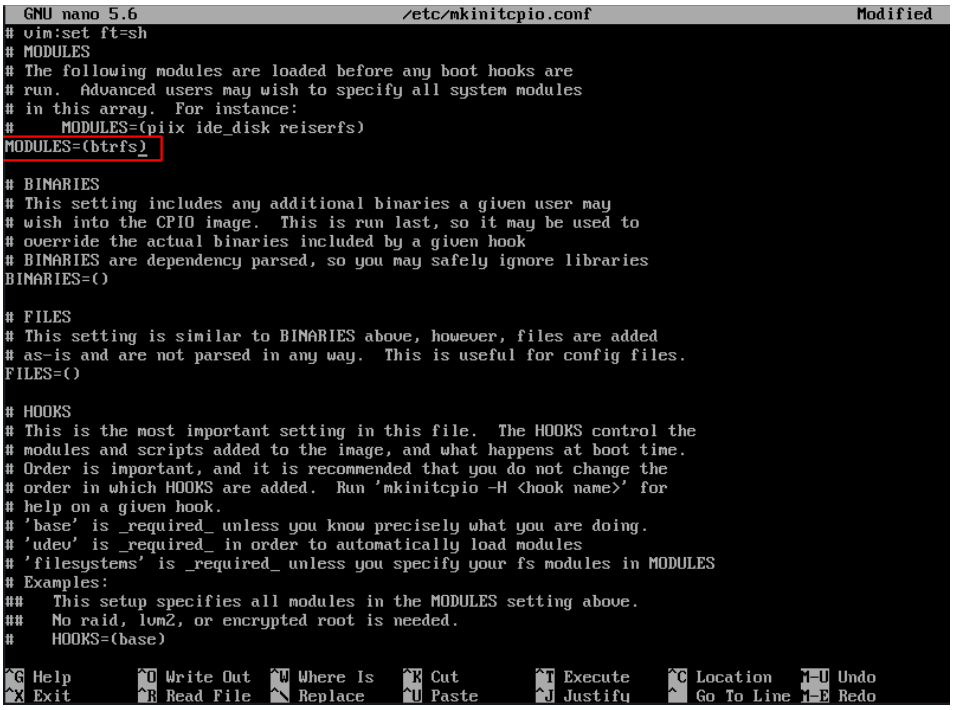
Package Name	Use
bluez & bluez-utils	Bluetooth support
cups	Printing support
xdg-utils & xdg-user-dirs	Better integration with desktop environments

After entering the command, press Enter to select all of the base-devel packages to install.

Then wait for the installation to finish.

Step 18: Adding btrfs module to mkinitcpio

```
nano /etc/mkinitcpio.conf
```



Add btrfs in MODULES=() Save and exit nano.

Now to recreate the image:

Replace linux with linux-lts if you installed the lts kernel

Step 19: Installing GRUB

Installing grub:

```
grub-install --target=x86_64-efi --efi-directory=/boot --bootloader-id = Arch
```

Now to generate the configuration file:

```
grub-mkconfig -o /boot/grub/grub.cfg
```

Step 20: Creating a User

Adding a user:

```
useradd -mG wheel nishantn
```

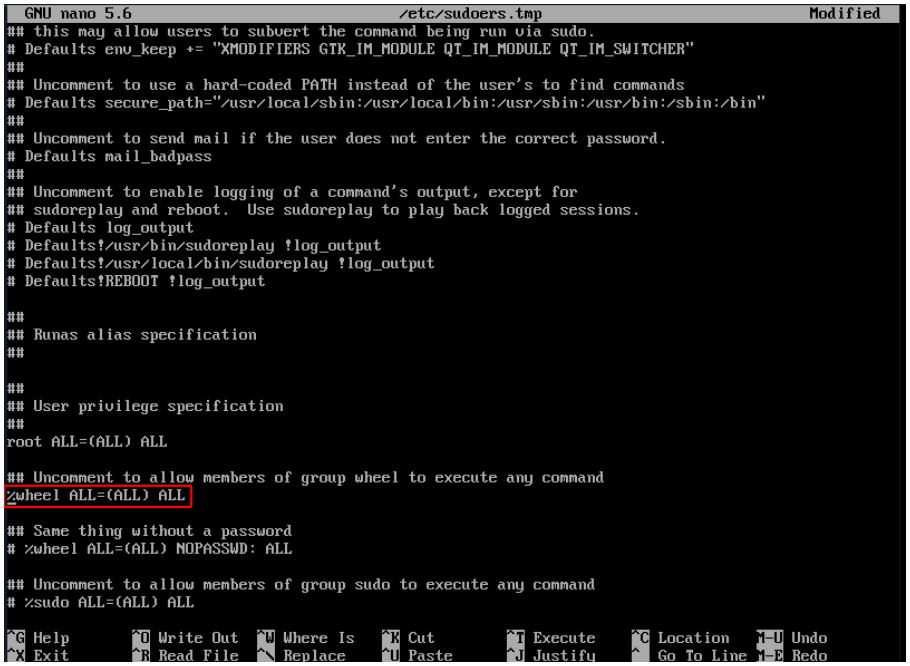
Above command adds a user with name nishantn and gives it access to wheel group (for sudo privilages). Replace nishantn with whatever name you want.

Giving a password to the user:

Enter password twice

Now to give usersfrom the wheel group full sudo access:

Uncomment the line which says **%wheel ALL=(ALL) ALL** Save and exit nano

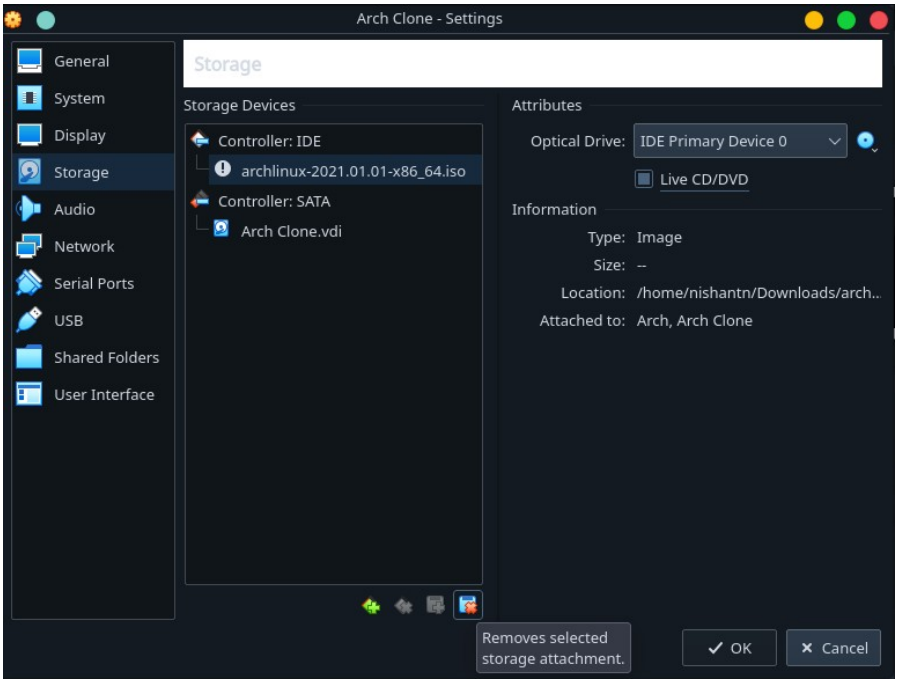


Step 21: Enabling services

```
systemctl enable NetworkManager
## If you installed bluez
systemctl enable bluetooth
## If you installed cups
systemctl enable org.cups.cupsd
```

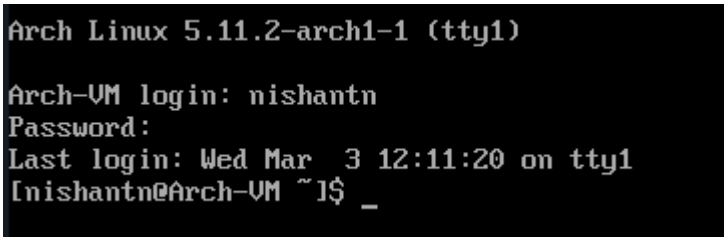
Step 22: Restarting into Arch

```
## Exiting the installation
exit
## Unmounting all drives
umount -l /mnt
## If you're installing Arch on VM
shutdown now
## If you're dual booting/installing on a device reboot
```



Deleting Arch iso (VM users only)

After shutting down, go to Storage settings of your VM, select the iso file and click on remove selected. After restarting and logging in, it should look something like this:



After Install

Congratulations you have installed Arch Linux successfully! You would still need to go ahead and install a desktop environment or window manager on top if you want but the hard part is over. After completing the base install, you are now eligible to make an account on the prestigious Arch Linux website which contains the Wiki, Forums, etc.

Refer this to see post install guide for Arch:
https://wiki.archlinux.org/index.php/General_recommendations

Thank you!