Manual Installation - Arch Linux

Step-by-Step Linux Installation Guide & Walkthrough

Yaki, sukiyaki2116@gmail.com,
https://www.linkedin.com/in/sodunke-olasunkanmi/

2024-10-27

Contents

Step 1: Arch Linux ISO & Boot Media Setup

The first step will be to download the Archlinux ISO image from https://archlinux.org/download/

The image can be written to a Usb drive with a tool like `Rufus` or `USBImager`

Create free space with Windows Disk Management or Wipe the entire hard drive depending on your installation scenario(Dual/triple boot or Bare Metal Installation). If you are unfamiliar with the process, refer to the articles or YouTube video

https://www.hp.com/us-en/shop/tech-takes/how-to-partition-a-hard-drive

https://www.zdnet.com/article/how-to-disk-partitioning-for-linux-and-windows-dual-booting/

https://www.youtube.com/watch?v=7JBFJuA5QsM

Step 2: Create three partitions with cfdisk

```
cfdisk /dev/nvme0n1
```

Choose free space and create the partition based on that (Choose the type of filesystem) - Linux File system

For Arch, create one or two 1GB partition for both boot & EFI and use the rest of the partition space for the regular filesystem.

If you intend on using an existing partition, Select resize to make the space available.

When done make sure you write the changes to disk to confirm changes.

Step 3 - Format file system & Encrypt main disk

```
mkfs.fat -F32 /dev/nvme0n1p7
mkfs.ext4 /dev/nvme0n1p8

#Encrypting main disk
cryptsetup luksFormat /dev/nvme0n1p9
```

confirm with yes in all caps.

Then enter `passphrase` you will use to enter encrypted drive moving forwards.

Step 4: Set up LVM

```
cryptsetup open --type luks /dev/nvme0n1p9 lvm     //lvm = name of encrypted vuolume
```

Enter passphrase for the volume you just created.

Step 5: Create a physical Volume for the LVM

```
pvcreate /dev/mapper/lvm
```

This should create the physical volume for nvme0n1p9 volume.

Step 6: Create a volume group

```
vgcreate volgroup0 /dev/mapper/lvm
```

Step 7: Create Logical Volumes

```
lvcreate -L 21GB volgroup0 -n lv_root

lvcreate -L 21GB volgroup0 -n lv_root

vgdisplay   //verify volume groups created so far
lvdisplay   //verify logical groups created so far
```

Logical volumes works as a partition.

First logical volume would be for the `root file system`.

Second logical volume would be for the `file system where all our files` would be located

Step 8: Install a Kernel Module with modprobe

```
modprobe dm_mod
```

Step 9: Scan for Available Volume groups & Activate them

```
vgscan      //scan
vgchange -ay       //activate all volume groups
```

Step 10: Format the logical group we would be using for the root filesystem & Home file system

```
mkfs.ext4 /dev/volgroup0/lv_root     //root
mkfs.ext4 /dev/volgroup0/lv_home     //home
```

Step 11: Mounting the Partitions/logical volumes(Root filesystem and Boot)

```
mount /dev/volgroup0/lv_root /mnt

#Make directory for root and mount it
mkdir /mnt/boot
mount  /dev/nvme0n1p8 /mnt/boot/
```

Step 12: Mounting the Home Partition

```
#First maker directory and then mount

mkdir /mnt/home
mount /dev/volgroup0/lv_home /mnt/home
```

Step 13: Install some of the required Packages for Arch Linux

```
pacstrap -i /mnt base
```

Then select yes and iptable_nft for better firewall support and configuration.

Step 14: Create the FSTAB required for Mounting partitions at Boot time

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

cat /mnt/etc/fstab       //verifying it was created
```

Step 15: Log in to In-progress Archlinux Installation

This is to use chroot to access the partition we are installing archlinux unto and run some commands against it as well as install some packages.

```
arch-chroot /mnt
```

Step 16: Creating a root passwd for the root user

```
passwwd
```

Make sure you choose a strong password.

Step 17 : Create a new user as well

```
useradd -m -g users -G wheel yaki

passwd yaki
```

In this case, I am adding the user to the users group as well as the wheel which will give us access to Sudo which we wlil definitely need later down the line.

Make sure you target a username when setting up a password to avoid overriding the root password.

Step 18: install additional Packages

```
pacman -S vim base-devel dosfstools grub efibootmgr gnome gnome-tweaks lvm2 mtools nano
↪  mousepad networkmanager openssh os-prober sudo
```

Step 19 : Enable SSH when System starts (Optional)

```
systemctl enable sshd
```

Step 20 : Install Linux - Kernel

```
pacman -S linux linux-headers linux-lts linux-lts-headers
```

Linux is the kernel, Archlinux is the operating system.

Also install linux header and linux-lts(Long supported linux kernel) as a failsafe incase your primary linux fails for whatever reason.

Step 21: Install linux Firmware

```
pacman -S linux-firmware
```

Firmware package contains firmware for the linux kernel, you might need it for the proprietary hardware.

Step 22: Set up a driver for Video Card

```
lspci

pacman -S mesa

pacman -S nvidia nvidia-utils nvidia-lts


pacman -S intel-media-driver    //Intel hardware decoding
pacman -S libva-mesa-driver     //AMD hardware decoding
```

`lspci` lists all your PCI devices. Look for output that indicates what kind of GPU you have (Intel , AMD or NVidia).

if you have an intel or AMD GPU you would have to install the `mesa` package.

For Nvidia Users, you would install the nvidia package: `nvidia`, `nvidia-utils` and `nvidia-lts` for the linux-lts kernel.

For `Intel` users, you would have to install a package to enable hardware decoding to know how recent your chipset is

For AMD users, you would also have to install a package to enable hardware decoding to know how recent your chipset is.

Step 23: Edit a Special Config File - mkiniticpio.conf

```
nano /etc/mkinitcpio.conf


HOOKS=(base udev autodetect microcode modconf kms keyboard keymap consolefont block encrypt
↪  lvm2 filesystems fsck)
```

Add encrypt and lvm2 to the config file, add it specifically between bloc and filesystem.

This enables encryption and lvm2. Without this the kernel wouldn't know how to deal with the encrypted filesystem and lvm2.

Save the file amd exit out.

Step 24: Generating a few things for our linux kernel

```
mkinitcpio -p linux
mkinitcpio -p linux-lts
```

Run this for both linux kernels you installed.

Step 25: Set up the locale Information for our installation

```
nano /etc/locale.gen

Uncomment en_Us.UTF-8 UTF-8


locale-gen
```

locale-gen is used to generate the locale itself.

Step 26: Edit the GRUB file

```
nano /etc/default/grub


Add the following to the line
GRUB_CMDLINE_LINUX_DEFAULT="loglevel=3 cryptdevice=/dev/nvme0n1p9:volgroup0 quiet"
```

This is for booting up the OS.

Step 27: Set up the EFI partition

```
mkdir /boot/EFI
mount /dev/nvme0n1p7 /boot/EFI/
```

Step 28: Install Grub bootloader

```
grub-install --target=x86_64-efi --bootloader-id=grub_uefi --recheck
```

This is the bootloader that will allow the computer to even boot in the first place.

Step 29 : Copy file to Boot Directory

```
cp /usr/share/locale/en\@quot/LC_MESSAGES/grub.mo /boot/grub/locale/en.mo
```

Step 30: Generate Config file for the Grub bootloader

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

Saved it to the output file `/boot/grub/grub.cfg`

Step 31: Install GDM for a login screen when we boot our system

```
systemctl enable gdm
```

Step 32: Enable Networkl Manager

```
systemctl enable NetworkManager
```

Step 33: Finishing Steps

```
exit

umount -a

reboot
```

Exit the chroot environment

unmount everything and reboot. VOILA!! A brand new Linux installation.