

Install Arch Linux on a Macbook Air

A set of tips for installation and post install

Ronald (Ryy) Glenn Thomas

2024-08-03

Table of contents

1	Introduction	2
2	Download	2
3	Install Arch on a Macbook Air	2
4	Install Core Components	3
4.1	Connect to target machine with ssh	3
4.2	Begin install process	4
5	for intel processors	6
6	Setup configuration	8
7	Additional Software setup	9
8	Appendix 1. Script to set up links from local Home to Dropbox	11
9	Appendix 2. Copy files to new Mint machine	11
10	Practice	12
11	Appendix	12



1 Introduction

The specific goal of this “DIY” project is to refurbish a 2017 macbook pro laptop with a contemporary linux operating system.

The OS we’ll focus on for this post is **Arch Linux** a rolling **Arch Linux** distribution. Why **Arch**?

Well, we’re looking for a lightweight, fast installing distro that has access to the **AUR** repository of apps. There are many other considerations that can go into choosing a linux distribution, but for our purposes, this is the main one.

2 Download

To get started, acquire a copy of the Arch distribution. The simplest way to do this is to download the latest **ISO** image file and “burn” it onto a **USB** drive. (We’ll use an apple laptop running **macos Sonoma** to facilitate the download and writing onto a 60 GB **USB** drive.) At this writing the latest **iso** file is version **2024.08.01** which is based on linux kernel: **6.10.2**. The **ISO** Size is **1.1 GB**. We’ll use a torrent file **archlinux-2024.07.08-x86_64.iso.torrent** to download the **iso** file using the torrent client **Transmission**

3 Install Arch on a Macbook Air

The target machine is a 2016 13-inch MacBook Air with one Thunderbolt 3 port.

Install the mac app [Transmission](#) and add the torrent file.

Also download the associated **sha256sum.txt** file.

To check the integrity of your local ISO file, generate its SHA256 checksum and compare it to the content of the `sha256sum.txt` file:

```
> sha256sum archlinux-2024.07.01-x86_64.iso
```

compare to SHA256 sums from the download site. In our case:

```
398dceea2d04767fbb8b61a9e824f2c8f5eacf62b2cb5006fd63321d978d48bc
```

We can transfer the `iso` file to a USB flash drive using one of several methods. On `macos` we suggest using the app `balanaEtcher`. You can download `balanaEtcher` [here](#).

Insert the bootable USB flash drive into the target macbook and reboot. Hold the `ALT` key while the machine reboots and you'll be presented with a screen offering boot drive options. Select the icon for the USB drive. A `grub` menu will appear.¹

From the `Grub` menu choose `Arch Linux install medium (x86_64, UEFI)` and the `arch` install program will start.

¹ **GNU GRand Unified Boot-loader (GRUB)**. “When your Linux operating system starts up, GRUB is the first program that runs. It loads the kernel of the operating system, and then the kernel loads the rest of the operating system, including the shell, the desktop environment, and other operating system features.” [codecademy.com](https://www.gnu.org/software/grub/)

4 Install Core Components

To allow cut and paste from `macos` to target laptop connect on your local network via `ssh`.

4.1 Connect to target machine with `ssh`

configure WIFI

- `bash> iwctl`
-
- `iwd> device list` (assume device is `wlan0`)
- `idw> station wlan0 scan`

- `idw> station wlan0 connect rgtnet2`
- passphrase for `rgtnet2`
- `idw> exit`
- start ssh and determine target machine IP address

```
* bash> systemctl enable sshd
* bash> systemctl start sshd
* bash> ip route | grep default
```

- (assume the local IP address is 10.0.1.176)
- Set a password for root user. You'll need it to log in.
- `bash> passwd`
- (enter `z` password)
- New password: z
- Retype new password: z

Now switch over to the mac

```
* zsh> ssh -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no root@10.0.1.176
* ssh> root@10.0.1.176's password:
* z
```

- ssh returns prompt on arch target machine

```
root@archiso ~ >
```

4.2 Begin install process

First step:

- partition harddisk:

```
bash> cfdisk /dev/nvme0n1
```

Use interface to create two partitions:

1. EFI type of size 1gb
2. root of size entire rest of disk.

3. write partition to disk

check the partition:

- `bash> fdisk /dev/nvme0n1 -l`
- Format the partitions.
 - EFI disk is fat32
 - Root is ext4
- `bash> mkfs.fat -F32 /dev/nvme0n1p1`
- `bash> mkfs.ext4 /dev/nvme0n1p2`
- set the keymap to mac us
 - `bash> loadkeys mac-us`
- Set system time and date
- `bash> timedatectl set-ntp true`

“The `/mnt` mount point in Linux is for mounting a storage device temporarily. As we only need to mount the partition for installing Arch Linux on it, the `/mnt` mount point is perfect.”

[The Arch Linux Handbook – Learn Arch Linux for Beginners](#)

- `bash> mount /dev/nvme0n1p2 /mnt`
- `bash> mount -mkdir /dev/efi_system_partition /mnt/boot`
- Find best mirror:
-
- `bash> reflector --download-timeout 5 --country “United States” --age 12 --protocol https --sort rate --save /etc/pacman.d/mirrorlist`
-
- `bash> pacstrap /mnt base base-devel linux linux-firmware sudo networkmanager`
- `bash> genfstab -U /mnt » /mnt/etc/fstab`
- `bash> arch-chroot /mnt`
- `bash> ln -sf /usr/share/zoneinfo/America/Los_Angeles /etc/localtime`

- edit /etc/locale.gen to set locale
- search for en_US and uncomment first row
- bash> vim /etc/locale.gen
- bash> locale.gen
- bash> vim /etc/locale.conf
- pacman -S networkmanager
- Avahi provides local hostname resolution using a “host-name.local” naming scheme. To enable it, install the nss-mdns package and start/enable avahi-daemon.service.
- pacman -S nss-mdns
- systemctl start avahi-daemon.service
- systemctl enable NetworkManager (add root passwd)
- passwd
- useradd -m -G wheel zenn
- passwd zenn
-
-

5 for intel processors

- pacman -S intel-ucode
-
- pacman -S grub efibootmgr
-
- mkdir /boot/efi
- mount /dev/nvme0n1p1 /boot/efi
- grub-install --target=x86_64-efi --bootloader-id=grub
- grub-mkconfig -o /boot/grub/grub.cfg

-
- pacman -S xorg-server
- check on what type of video card is in place on target
- lspci -v | grep -A1 -e VGA -e 3D
-
- pacman -S xf86-video-intel
-
- pacman -S gnome
-
- pacman -S cinnamon
-
- systemctl enable gdm
-
- exit
-
- root@archiso ~ # umount -R /mnt

Put the whole thing together for post disk partitioning ...

```
mkfs.fat -F32 /dev/nvme0n1p1
mkfs.ext4 /dev/nvme0n1p2
mount /dev/nvme0n1p2 /mnt
reflector --download-timeout 5 --country "United States" --age 12 --protocol https --sort rate
pacman -Sy
pacstrap /mnt base base-devel linux linux-firmware sudo networkmanager
genfstab -U /mnt >> /mnt/etc/fstab
arch-chroot /mnt

passwd
useradd -m -G wheel z
passwd z
pacman -S vim
vim /etc/sudoers
```

```

uncomment # %wheel ALL=(ALL) ALL
ln -sf /usr/share/zoneinfo/America/Los_Angeles /etc/localtime
vim /etc/locale.gen
    uncomment en_US line
locale-gen
vim /etc/locale.conf
    LANG=en_US.UTF-8
vim /etc/hostname
    enter zz
pacman -S networkmanager
systemctl enable NetworkManager
pacman -S intel-ucode
pacman -S grub efibootmgr
mkdir /boot/efi
mount /dev/nvme0n1p1 /boot/efi
grub-install --target=x86_64-efi --bootloader-id=grub
grub-mkconfig -o /boot/grub/grub.cfg
pacman -S --noconfirm xorg-server
pacman -S --noconfirm xf86-video-intel
pacman -S --noconfirm gnome
pacman -S --noconfirm cinnamon
systemctl enable gdm
exit
umount -R /mnt

```

The final hardware related step is to add a second monitor, if available, via HDMI or “USB-C”.

Thats it. The base system is ready to go. Reboot and login with the admin username and password you provided earlier.

6 Setup configuration

Set keyboard and trackpad preferences:

- * Open `Mouse and Touchpad` in settings. Turn on `Reverse scroll`.
- * Open `Keyboard` > `Layouts` > `Options` > `Caps Lock behavior` and select

`Swap Esc` and `Caps-Lock`. This is an important setting for `vim` use.

- * Open `Shortcuts` > `Windows`.
 - * Set `Maximize window` to `Alt-f`
 - * Set `Unmaximize window` to `Alt-g`
 - * Set `Close window` to `Alt-q`

Next configure the displays.

1. On a two monitor system open Display menu (press `command` key to open menu and search for “display”). Select the macbook as the primary monitor with 2560x1440 resolution. Set Monitor scale at 200% to increase default font size in apps. Second monitor (e.g. Dell ?) should be set at 3840x2160 (200%)

7 Additional Software setup

Start with Dropbox to transfer working environment

```
sudo apt install nautilus-dropbox
dropbox autostart y
dropbox start -i
```

Dropbox startup process will launch a “Sign in” web page. Login with Dropbox credentials through web page.

Next

- Add a PPA for R packages,
- Update the `apt` “listings” and
- install basic utilities `fzf`, `ripgrep`, `ssh`, `git`, `wget`, `curl`, `zsh` and plugins, as well as
- major applications `R`, `vim`, `qutebrowser`, `firefox` `dropbox` and `zathura` with the following commands:

```
# R contributing PPA
sudo add-apt-repository ppa:c2d4u.team/c2d4u4.0+ -y
sudo apt update
sudo apt upgrade -y
sudo apt install \
terminator tree ssh zsh curl git vim fzf ripgrep \
autojump zsh-syntax-highlighting zsh-autosuggestions \
r-base-core r-cran-tidyverse \
r-cran-kableextra r-cran-styler \
r-cran-shiny r-cran-rmarkdown r-cran-tidyverse r-cran-knitr \
texlive-science zathura qutebrowser firefox -y
```

Run bash shell script `~/Dropbox/dotfiles/set_up_links.sh` to set up symbolic links (e.g. `ln -s ~/Dropbox/prj ~/prj`). See Appendix 1 below for details.

Make `zsh` the default shell.

```
> chsh -s $(which zsh)
```

Set up the shell (`zsh`) per the post [\[link to set up terminal post\]](#)

Install `zotero` using software manager and set up syncing (login: `rgthomas`)

add `vimium` extension to `firefox`

Testing: Should be able to render both `Rmd` and `qmd` files.

```
cd c176
vm
ZT

po
cd setupmint
quarto render index.qmd --to pdf
```

8 Appendix 1. Script to set up links from local Home to Dropbox

set_up_links.sh

```
#!/bin/zsh

# since the install process creates a .config directory move it temporarily
mv ~/.config ~/.config.tmp

# create links to hidden files from ~/Dropbox/dotfiles directories
ff=".zshrc" ".viminfo" ".vimrc" ".local" ".vim" \
    ".vimplugins" ".config" ".Rprofile")
for P in "${ff[@]}"
do
    echo "create a link for Dropbox/dotfiles version of $P in Home"
    ln -v -s "$HOME/Dropbox/dotfiles/$P" "$HOME/$P"
done

# copy the original ".config" files into new linked .config
cp -R ~/.config.tmp/* ~/.config

# create new directories (links) for working files from Dropbox
dd=("sandbox" "bin" "docs" "prj" "work" "ssh" "shr")
for P in "${dd[@]}"
do
    echo "create a link for Dropbox/dotfiles version of $P in Home"
    ln -v -s "$HOME/Dropbox/$P" "$HOME/$P"
done
```

9 Appendix 2. Copy files to new Mint machine

Connect to new machine via ssh from mac laptop

First on the new machine (zz)

```
zz> sudo apt install ssh
zz> ifconfig
```

get IP for target, say 10.0.1.196

Either shell in to linux mint machine, or secure copy files over.

```
mac> ssh z@10.0.1.196
mac> scp install_apps.sh z@10.0.1.196:~
mac> scp set_up_links.sh z@10.0.1.196:~
```

Possible Shortcut

Install dropbox first. You could wait for Dropbox to finish installing or you could use scp to copy and run the two shell scripts: `install_app.sh` and `set_up_links.sh` from `~/Dropbox/dotfiles`. These two shells can run in parallel with Dropbox installing.

10 Practice

11 Appendix

alternatively using fdisk:

- - fdisk /dev/sda
- -n create new partition
- -p make it a primary partition
- -1 first of four partitions
- RET accept default first sector
- RET accept default last sector (one big partition)
- a toggle bootable flag
- w write to disk
- add file system to partition

- `* mount root partition on /mnt`
- `> mount /dev/sda1 /mnt`

```
# Install Docker sudo pacman update sudo pacman -S docker  
sudo systemctl start docker.service
```

12 Run peng1.Rmd

```
bash> sudo pacman -S r bash> R -e "library(rmarkdown); ren-  
der('peng1.Rmd')"
```