Arch Linux Install Notes

Ryan Nash

(If a section has an asterisk next to it, it is unfinished)

Boot Live Environment

Changing settings for live environment

- Set Keyboard to GB
  - `loadkeys UK`
- Check we are in `efi` mode
  - `efivar -l` (if this outputs text, we are in efi-mode
- Connect to internet (if using wifi)
  - `wifi-menu`
- Update clock
  - `timedatectl set-ntp true`

Partitioning

- List devices to find drive arch is to be installed on
  - `lsblk`
- Wipe partition table (GPT)
  - ‘gdisk /dev/*drive_name*/
  - `x` (go to advanced menu), `z` (destroy GPT structures), `y`, `y` (confirm twice)
- Create new GPT partition table and partitions
  - cgdisk /dev/*drive_name*/

EFI partition

- Press enter to initialise wiped drive to GPT
- Highlight `free space` and press enter on `new`
- Press enter to set starting sector as default
- Type 512MB and press enter for sector size, creating a partition of size 512MB
- Type `EF00` as the filesystem hex code (this is the code for an EFI system partition)
- Type *name_you_want* and press enter (this is the name/label for the drive)

Primary partition

(I am using a two partition table, just the efi and then one more for `root` and `home`, I will not be using a swap partition as I will be using a swap file instead)

- Highlight `free space` again and press enter on `new`
- Press enter 3 times to keep the defaults
- Name the drive and press enter (e.g. primary)
- Arrow over to write, press enter, and then quit

Partition File Systems

```
* Make our EFI partition `FAT32`
```

```
    * `mkfs.fat -F32 /dev/*efi_partition*/
    * `mkfs.ext4 /dev/*root_partition*/
```

Mount Partitions

```
* Mount root partition
    * `mount /dev/*root_partition*/ /mnt
* Create mount point for efi partition
    * `mkdir /mnt/boot`
* Mount efi partition
    * `mount /dev/*efi_partition*/ /mnt/boot
```

Mirrorlists*

- Make a copy of the mirrorlist (just in case)
    - ‘cp /etc/pacman.d/mirrorlist /etc/pacman.d/mirrorlist.backup
- Open mirrorlist and put UK mirrors at the top

Install

- Install arch base packages and development packages
    - `pacstrap -i /mnt base base-devel`
- Generate FSTAB file (what is this?)
    - `genfstab -U /mnt /mnt/etc/fstab`
- Check FSTAB is correct (need to look into this a bit more)
- Chroot into new system
    - `arch-chroot /mnt`

Configure Booting, Time & Language for the new system

- Create locale files
    - `vi /etc/locale.gen` then uncomment my language (en_GB.UTF-8)
    - `echo LANG=en_GB.UTF-8 > /etc/locale.conf`
    - `echo KEYMAP=uk > /etc/vconsole.conf`
- Time zone
    - `ln -sf /usr/share/zoneinfo/Europe/London /etc/localtime`
    - hwclock –systohc
- Hostname
    - `echo RN-T480 > /etc/hostname`
    - Add these lines to the hosts file `/etc/hosts`
        - 127.0.0.1 localhost
        - ::1 localhost
        - 127.0.1.1 RN-T480.localdomain RN-T480
- Set root password
    - `passwd` and choose a password
- Enable multilib and AUR repositories
    - `vi /etc/pacman.conf` and uncomment these repos
    - Update packages with `pacman -Sy`
- Install required packages for wifi-menu used when we reboot into our new OS
    - `pacman -S iw wpa_supplicant dialog`
- Add a new standard user
    - `useradd -m -g users -G wheel,storage,power -s /bin/bash ryan`

- Sudoers
  - `EDITOR=vi visudo` uncomment `%wheel ALL=(ALL) ALL`
  - Make sudo require root pw instead of user password
    - Add the line `Defaults rootpw`
- Install bash completion `pacman -S bash-completion`

Bootloader (I am using rEFInd)

- Mount EFI variables
  - `mount -t efivarfs efivarfs /sys/firmware/efi/efivars`
  - If it says already mounted that is fine as it is the point of the command
- Install rEFInd
  - `pacman -S refind-efi`
  - `refind-install`
- Configure boot options (COME BACK TO)
  - Open `/boot/refind_linx.conf`
  - Change second string of each line to `root=UUID*rest_of_UUID_of_drive`

Intel Processors*

- Install intel microcode package

Nvidia

(May need to set up before rebooting as some cards can freeze with Arch's default nouveau drivers)

- Install default linux kernel headers, so we can use the DKMS nvidia drivers (DKMS modules will be rebuilt if we decide to switch kernels, so we will not need to install these drivers again)
  - `sudo pacman -S linux-headers`
- Install nvidia drivers and related packages
  - # X-Server (GUI)
- Install touchpad support
  - `sudo pacman -S xf86-input-synaptics`
- Install 3D support
  - `sudo pacman -S mesa`
- Install X
  - `sudo pacman -S xorg-server xorg-apps xorg-xinit xorg-twm xorg-xclock xterm`
- Test X is working
  - `startx` we should see a few terminals and a clock
  - Type exit in a terminal to go back to a TTY

Desktop Environment

- Install KDE Plasma
  - `sudo pacman -S plasma sddm`
  - sudo systemctl enable sddm.service'
  - Install `firefox gvim konsole dolphin`
- Switch to network-manager instead of wifi-menu
  - Boot into KDE
  - `sudo pacman -S networkmanager network-manager-applet`
  - Find internet devices using `ip link`
  - Disable dhcpcd on any ethernet devices 'sudo systemctl disable dhcpcd@*ethernet_device*.service

    - Disable netctl on any wireless devices ‘sudo systemctl disable netctl-auto@*wireless_device*.service
    - Enable network manager `sudo systemctl enable NetworkManager.service`
    - Reboot

Post Install

Swap File

systemd-swap

- Install the `systemd-swap` package
- Set `swapfc_enabled=1` in the file `/etc/systemd/swap.conf`
- Check current configuration
    - `cat /proc/sys/vm/swappiness`
    - `cat /proc/sys/vm/vfs_cache_pressure`
- Configure using recommended settings
    - `echo vm.swappiness=5 | sudo tee -a /etc/sysctl.d/99-sysctl.conf`
    - `echo vm.vfs_cache_pressure=50 | sudo tee -a /etc/sysctl.d/99-sysctl.conf`
    - `sudo sysctl -p /etc/sysctl.d/99-sysctl.conf` this reloads the file
- `sudo systemctl enable systemd-swap.service`

Configure vim

```
* Create .vim folder in HOME
* Copy vim folder from dotfiles with plugins etc
* Copy vimrc to HOME/.vimrc
```

- Packages for R markdown
    - Firstly install LaTeX with the `texlive-core and texlive-latexextra` packages
    - `R pandoc-citeproc` maybe `gcc-fortran` too
- Install r packages using r
    - `sudo r` use sudo to install these packages system wide, means there won’t be an annoying r folder in my home directory
    - ‘install.packages(“rmarkdown”)
    -

Other

Silent Boot

- Install package `ntfs-3g` for reading and writing to NTFS formatted drives
- TRIM SSD support, a safe weekly TRIM service
    - `systemctl enable fstrim.timer`
- Install `redshift plasma5-applets-redshift-control`
- Sync Firefox
- Install `git`
- Copy git credentials and git config file to root of home
- Install `ruby`
- Install jekyll, `gem install jekyll bundler`
- Add `PATH="$PATH:$(ruby -e 'puts Gem.user_dir')/bin"` to ~/.profile and source it
- Add the line `PATH=PATH:$/home/ryan/.gem/ruby/2.6.0/bin` to .profile and source it
    - This line sets the PATH to its current path with the additional location‘
- Change bundler install path to local user dir ‘bundle config path ~/.gem
- Run bundle install or create a new jekyll site from cmd

KDE Specific Notes

- Change kb layout to UK

Firefox Specific Notes

- Adjust font settings in preferences

Fonts

- Directory `~/.local/share/fonts`
- Update with `fc-cache -v`

Nvidia Screen Tearing

-

Speeding up AUR compiling

-

References:

GloriousEggRoll Arch Install Guide