Setting Up Arch Linux for Home Use

Sean Craig

Last revised: April 17, 2019

Contents

1 License information

2 Foreword

The following text is a collection of things that have "worked for me" over the course of my time maintaing Arch Linux systems in my home. Arch has a reputation for being a particularly technically-demanding flavor of Linux. This point is subject to debate, a debate I have no desire to participate in. It is undeniable, however, that Arch users benefit from tested code, particularly during the distribution's command-line setup process. While the ArchWiki's Installation Guide and General Recommendations provide sufficient resources to set up a machine, finding exactly what you need can take undue time, especially for otherwise knowledgeable users who *just need the code*.

I am to provide a one-stop shop for setting up a functional Arch Linux system suitable for everyday use in the home. The instructions are intended to produce a machine capable of getting technical work done, but not to be a programmer's or system administrator's daily driver. Experienced Linux users will find a "vanilla" system build. My default system uses UEFI, runs Xorg rather than Wayland, typically prefers GTK applications to Qt, uses large portions of the Gnome stack (but not the Gnome desktop environment), is oriented around a small number of users, and considers that the user might want to dual-boot Arch with Windows.

More than anything, the text aims to minimize the effort needed to prepare a system that I would want to use. It is not an educational text. I use these instructions when setting up my home machines and, as such, it is subject to my own biases. I do not apologize for them. I hope, however, that other users will find the text useful.

The text assumes that the reader is "Linux-literate" and comfortable using the command line; in other words, I assume the reader possesses the skills of the typical *potential* Arch user. I do not assume, however, that the user has used Arch before.

The code encluded herein is a mix of snippets from the ArchWiki, Arch Linux Forums, various other websites, and my own creations. Nothing is novel or particularly clever. My contribution is, predominately, collecting the information in one place and presenting it in roughly the order it will be needed.

Code appears in indented blocks. These instructions do not indicate whether the code is to executed as the root user. This is because we will enter commands as root only during the Arch live session. At that time we will act *exclusively* as root. During that session, we will set up user-level privilege escalation and authorize the our user-level profile to execute commands using `sudo`. The rest of system configuration will take place in a user-level account, using `sudo` where necessary.

3 Materials

To use these instructions, you will need the following:

1. An **internet connection**, preferably wired.
2. A **bootable Arch image**. I typically use a USB stick for this. There is no need for the disk image to be particularly current. An old image will typically do just fine.
3. A **bootable GParted Live image**. GParted Live is a minimal Debian live distribution that includes just enough of an interface to run the GParted disk partitioning utility. While it perfectly reasonable to partition the disk(s) during the Arch live session, being able to visualize the partition table tends to result in fewer mistakes.
4. An hour or more of spare time.

4 The GParted Live Session

Boot from the **GParted Live** medium. GParted will launch automatically. Edit the partition table(s) on the disk(s) as needed. If you are installing Arch directly on top of another Linux distribution, simply make a note of the drive numbers (`/dev/sdXY`) associated with the partitions you will be using.

I typically use either 3 or 4 partitions for my Arch system.

1. An EFI System Partition ("ESP"), identified as `FAT32`
2. A Linux / (root) parition identified as `ext4`
3. (optional) A Linux /home partition, identified as `ext4`
4. A Linux swap partition

I use a separate home partition on machines that will be running *only* Arch Linux as a way to protect locally-stored files in the event something goes wrong. On systems running multiple operating systems (e.g., Arch and Windows), I use a separate, shared "data" partition, usually formatted `ntfs`, that serves the same purpose.

The ESP should be 512 MiB (about 540 MB). On a system without a separate `/home` partition, the root partition should be at least 30 GiB, just to be safe. Where a separate `/home` partition is used, 15 GiB will suffice for the root partition with home getting as much of the remaining space as can be spared. Swap space is a contentious issue. On modern systems, I find the maxim that swap space equals system memory to be wasteful. On systems with less than 16 GB of memory, 6-8 GB will be plenty, *if not planning to use hibernation*. On systems with both a solid state drive *and* a conventional hard disk, I always put the swap partition on the conventional drive.

Once the disks have been prepared, reboot and remove the GParted live medium.

5 The Arch Live Session

Boot from the Arch live medium. You should see a plain black-and-white boot loader with chunky text (i.e., systemd-boot). If you see a fancy boot loader with the Arch Linux logo (i.e., GRUB 2), you are booting in MBR/BIOS mode, rather than UEFI mode. While the live medium will successfully load in MBR/BIOS mode on a computer that uses UEFI, the resulting installation will not be bootable.

Note: These instructions cover only UEFI machines, though most of the code will be applicable to both.

5.1 Connect to the Internet

Wired networks should connect automatically. If connecting via wifi:

```
wifi-menu
```

Test the connection

```
ping google.com
```

5.2 Update the System Clocks

We will potentially do more with the system clock later, when we set the time zone. For now, just update the system clock.

```
timedatectl set-ntp true
```

5.3 Format the Partitions

Format the partitions with the appropriate filesystem. Replace X and Y with the approprate letter and number that you identified earlier.

```
mkfs.fat -F32 /dev/sdXY
```

Format the Linux partitions as appropriate.

```
mkfs.ext4 /dev/sdXY
```

Format and initialize the swap partition.

```
mkswap /dev/sdXY
swapon /dev/sdXY
```

5.4 Mount the File Systems

We need to mount the filesystems so that the files we install go to the appropriate disk partitions.

Mount the root file system first. Make sure that X and Y reflect the root partition.

```
mount /dev/sdXY /mnt
```

Create mount points for any other partitions (other than swap) within the root file system. Then mount those partitions to the appropriate mount point. Mount the ESP to `/boot/efi`. If you have no `/home` partition, skip the corresponding lines. Do **not** try to mount the swap partition.

```
mkdir /mnt/boot
mkdir /mnt/boot/efi
mkdir /mnt/home
mount /dev/sdXY /mnt/boot/efi
mount /dev/sdXY /mnt/home
```

5.5 Install core packages

In this section, we will download and install a minimal Arch system onto the mounted filesystem.

Make sure the installation medium PGP keys are up to date.

```
pacman -Sy archlinux-keyring
pacman-key --populate archlinux
```

Install the base group. Other packages can be installed at this time, but I find it more convenient to save this for after we change root into the new system. This makes it a little easier to track what is going on if things go wrong, since there is a limit on the number of screens you can page-up in a true interface like this one.

```
pacstrap /mnt base
```

5.6 Create a File System Table

Generate the file system table (fstab) for the new file system, defining partitions by UUIDs.

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

5.7 Chroot into `/mnt`

Change root into the new file system.

```
arch-chroot /mnt
```

5.8 Configure System Time

Set the time zone with a symbolic link. For U.S. Eastern Time, use:

```
ln -sf /usr/share/zoneinfo/America/New_York \
/etc/localtime
```

For other time zones, replace the region (`America`) and zone (`New_York`) with the appropriate values.
To see a list of regions and zones (replace REGION with your desired region):

```
ls /usr/share/zoneinfo
ls /usr/share/zoneinfo/REGION
```

If dual-booting with Windows or Ubuntu, set the hardware clock to use local time rather than UTC. This will prevent a situation where one OS always has the incorrect time.

```
timedatectl set-local-rtc 1
```

Generate `/etc/adjtime`

```
hwclock --systohc
```

5.9 Localization

Open `/etc/locale.gen` and uncomment "`en_US.UTF-8 UTF-8`" for the standard, American locale, and any other desired locales.

```
nano /etc/locale.gen
```

Alternatively, simply append the line for the appropriate locale to the end of the file.

```
cat en_US.UTF-8 UTF-8 >> /etc/locale.gen
```

Generate the specified locales.

```
locale-gen
```

Define the corresponding language variable. For our default, American locale:

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

5.10 Network Configuration

Set the host name, replacing MYHOSTNAME as desired.

```
cat MYHOSTNAME >> /etc/hostname
```

Write the /etc/hosts file. I do this line by line, but one can also use nano

```
cat 127.0.0.1 localhost >> /etc/hosts
cat ::1 localhost >> /etc/hosts
cat 127.0.1.1 MYHOSTNAME.localdomain MYHOSTNAME \
>> /etc/hosts
```

5.11 Users and Privileges

Set the root password.

```
passwd
```

After we reboot, we will do all additional configuration as a normal user with access root privileges when we want them. To get there, we need to create a user profile, set up privilege escalation, and authorize the user profile to escalate.

Create a new user, replacing MYUSERNAME as desired. I usually place this user in the wheel group to more easily grant it privileges later. Repeat for each administrator user. Omit "-G wheel" for any user who you wish to refuse these privileges, such as a young child or a person prone to breaking things.

```
useradd -m -G wheel -s /bin/bash MYUSERNAME
```

Set passwords for the new user(s)

```
passwd MYUSERNAME
```

Install the base-devel group to access privilege escalation via sudo, among other things. Installing base-devel now, instead of just sudo will save time later because the group also includes (almost) everything we need to build packages from source code.

```
pacman -Sy base-devel
```

Open the visudo file in a text editor to set which users can use sudo. Since we have defined our administrator user(s) as belonging to the wheel group, uncomment one of the lines allowing members of that group to execute any command. On single-user machines that do not travel, such as a desktop computer, I normally choose the option that omits the need for a password.

```
EDITOR=nano visudo
```

5.12 Boot Loader

Download the GRUB boot loader and auxiliary packages for working with UEFI.

```
pacman -Sy grub efibootmgr
```

If dual-booting, install `os-prober`. If dual-booting with Windows, install the package for working with Microsoft filesystems.

```
pacman -Sy os-prober
pacman -Sy ntfs-3g
```

Finally, if the system has an Intel CPU, install microcode updates.

```
pacman -Sy intel-ucode.
```

Install GRUB to the system.

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

Initialize the GRUB configuration file.

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

If dual-booting, hopefully GRUB will automatically recognize the other OS's.

5.13 Additional Packages

Install Git to enable use of the Arch User Repository on reboot.

```
pacman -Sy git
```

If using wifi, install command-line tools

```
pacman -Sy iw wireless_tools wpa_supplicant dialog
```

5.14 Reboot

Exit the `chroot` session and reboot the computer. Be sure to remove the installation medium.

```
exit
reboot
```

6 Booting for the First Time

After the system restarts, GRUB should boot into an Arch session. Log in using your **username** and password. We want to be a non-root user to install packages from the user repository safely. We'll use `sudo` to become root as needed.

6.1 Connect to the Internet

If connecting wireless, run `wifi-menu`

```
sudo wifi-menu
```

Verify the connection.

```
ping google.com
```

If using a wired connection and there is no connection, the most likely issue is an inactive `dhcpcd`.

```
sudo systemctl enable dhcpcd.service
sudo systemctl start dhcpcd.service
```

Wait about 30 seconds, then attempt to ping again.

6.2 Install an AUR Helper

We will install a package from the Arch User Repository (AUR) manually exactly once. This package is a program that will act as a package manager for the AUR in the same way that pacman acts as a package manager for the main repository. Whether manually or with a helper, we must execute the commands as a normal user.

First, clone the source for `pakku`, my favorite AUR helper.

```
git clone https://aur.archlinux.org/pakku.git
```

Change into the cloned directory.

```
cd pakku
```

Compile and install the package from the PKGBUILD

```
makepkg -sic
```

Change back to the parent directory and delete the cloned source code.

```
cd ..
rm -Rf pakku
```

6.3 The Display Server

Install the Xorg display server.

```
sudo pacman -Sy xorg
```

Install the appropriate graphics drivers. If using an Nvidia graphics card, this should be:

```
sudo pacman -Sy xf86-video-nouveau mesa
```

If you would rather use the proprietary driver, instead use:

```
sudo pacman -Sy nvidia nvidia-utils
```

If, instead, the computer has integrated Intel graphics, use:

```
sudo pacman -Sy xf86-video-intel mesa
```

6.4 Install the Desktop Environment

The appropriate instructions will vary somehwta depending on the environment you choose. I like Budgie because it is simple, familiar, and reasonably snappy.

```
sudo pacman -Sy budgie-desktop budgie-extras
```

Also install packages for GUI network management, especially if connecting wirelessly.

```
sudo pacman -Sy networkmanager network-manager-applet
```

Install other optional dependencies.

```
sudo pacman -Sy gnome-backgrounds gnome-control-center \
gnome-screensaver
```

Finally, install a terminal emulator (and possibly) a web browser). Other major applications, like and office suite or media player, can wait until after we boot into the GUI environment. Without a terminal emulator, we will not be able to install other programs. I like Tilix.

```
sudo pacman -Sy tilix
```

If installing a web browser, install the package for Adobe Flash at the same time. I always forget otherwise. If using Chromium, the package is `pepper-flash`. Since I use Firefox, I use the following:

```
sudo pacman -Sy firefox flashplugin
```

6.5 The Display Manager

The Display Manager controls logging in to the GUI environment. With Budgie, I prefer LightDM.

```
sudo pacman -Sy lightdm
```

-p
LightDM requires a separate "greeter" package. I prefer this one.

```
sudo pacman -Sy lightdm-webkit2-greeter
```

Enable LightDM so it starts at boot.

```
sudo systemctl enable lightdm.service
```

By default, LightDM wants to use the GTK greeter. We didn't install that, so we have to specify a different one. Open the general LightDM configuration file.

```
sudo nano /etc/lightdm/lightdm.conf
```

Change the `[Seat:*]` section of this file to read:

```
[Seat:*]
...
greeter-session=lightdm-webkit2-greeter
```

6.6 Install Essential Applications

Create the standard user directories.

```
sudo pacman -Sy xdg-user-dirs-gtk
```

Install a web broswer and flash plugin if you did not do so earlier. I use Firefox.

```
sudo pacman -Sy firefox flashplugin
```

Install a graphical file manager. `nautilus` integrates well with Budgie.

```
sudo pacman -Sy nautilus
```

Install an assortment of basic fonts.

```
sudo pacman -Sy ttf-liberation ttf-dejavu ttf-roboto
```

6.7 Reboot

Reboot the system.

```
reboot
```

7 Configuring Applications

In this section, I include specific instructions for setting up and/or configuring various applications. I also include a list of recommended software packages, with descriptions. By the end of this section, you should have a polished, fully-functioned system.

UNDER CONSTRUCTION