Archlinux Install

This page explains how to install Arch Linux on a USB flash drive. The end result is a persistent installation identical to that on a normal hard drive along with several performance optimizations aimed at running Linux on removable flash media.

The only packages explicitly installed besides `linux`, `linux-firmware`, and `base` are:

- `efibootmgr`
- `grub`
- `iwd`
- `polkit`
- `sudo`
- `vim`

Basic services, such as networking and time sync, are all handled by systemd.

1. Preparations

1.1 Acquiring an ISO and prepare an installation medium

Visit the Archlinux Download page and, depending on how you want to boot, acquire the ISO file or a netboot image. Personally I have never tried the netboot image, and strongly suggest you use the ISO to boot from USB or CD.

The installation image (ISO) can be supplied to the target machine via a USB flash drive, an optical disc or a network with PXE. Usually installing with a USB flash drive is the prefferable option.

1.2 Boot the live environment

Point the current boot device to the one which has the Arch Linux installation medium. Typically it is achieved by pressing a key during the POST phase, as indicated on the splash screen. Refer to your motherboard's manual for details.

When the installation medium's boot loader menu appears, select `Arch Linux install medium` and press `Enter` to enter the installation environment. The installation image uses systemd-boot for `x64 UEFI`, GRUB for `IA32 UEFI` and syslinux for `BIOS` booting, recognizing your system automatically.

You will be logged in on the first virtual console as the root user, and presented with a Zsh shell prompt.

1.3 Set the console keyboard layout

The default console keymap is US. Available layouts can be listed with:

```
# ls /usr/share/kbd/keymaps/**/*.map.gz
```

To set the keyboard layout, pass a corresponding file name to `loadkeys`, omitting path and file extension. For example, to set a Swiss German keyboard layout:

```
# loadkeys de_CH-latin1
```

2 Internet

To set up a network connection in the live environment, go through the following steps:

- Ensure your network interface is listed and enabled, for example with `ip-link`:

```
# ip link
```

- Connect to the network:
  - Ethernet – plug in the cable.
  - Wi-Fi – authenticate to the wireless network using iwctl. Follow the instructions in the next section.
- The connection may be verified with ping:

```
# ping archlinux.org
```

If you have a working internet connection at this point you can skip the next sections. Most probably though, you will get the response `ping: google.com: Name or service not known.`, meaning you are not connected to the internet.

2.1 Start Wireless Service

Start the INet Wireless Daemon (iwd). This wireless package comes pre-installed on most laptops.

To get iwd running do the following:

1. Enable the `iwd.service`

   ```
   # systemctl enable iwd.service
   ```

2. Start the service

   ```
   # systemctl start iwd.service
   ```

3. Verify it has started

   ```
   # systemctl status iwd.service
   ```

2.2 Connect to a Wi-Fi network

After verifying the iwd service is running in the previous step, type `iwctl` to get an interactive prompt.

The prompt will be displayed as `[iwd]#`.Here you are interacting with the client program 'iwctl'.

1. Get you device name

   ```
   [iwd]# device list
   ```

Note: wireless devices usually start with letter 'w'. e.g `wlan0` or `wlp3s0`

2. List all available networks

   ```
   [iwd]# station wlan0 get-networks
   ```

3. Connect to your network

   ```
   [iwd]# station wlan0 connect <your_network_name>
   ```

Substitute the above command with your network name, angle brackets excluded to connect to it. When you type the above command, for instance `station wlan0 connect guy_next_door` you will be prompted for passphrase. This is nothing but you network password. Type it and press enter.

Exit the iwctl client by typing `exit`

The connection may again be verified with ping:

```
# ping archlinux.org
```

3 Disk Partitioning

When recognized by the live system, disks are assigned to a block device such as `/dev/sda`, `/dev/nvme0n1` or `/dev/mmcblk0`. To identify the name of your devices, use:

```
# fdisk -l
```

Results ending in rom, loop or airoot may be ignored.

3.1 Wipe

The process of zeroing out or wiping the hard disk is almost always optional. Overwriting the Hard disk with zeros may be needed in rare cases when previous data aligns with the newly created partition table. If you have trouble installing a bootloader further on in the installation, you may have to come back to this point and wipe your USB before continuing.

Use dd to write the USB with all zeros, permanently erasing all data:

```
# dd if=/dev/zero of=/dev/sdX status=progress
```

Expect this to take a relatively long time (hour+) depending on the size of the hard disk. You can skip this step if you don't feel like waiting.

3.2 Partition

The following partitions are recommended for a chosen device:

- For booting in UEFI mode: an EFI system partition.
- One partition for the root directory `/`.
- One partition for the home directory `/home`.
- If you want to create any stacked block devices for LVM, system encryption or RAID, do it now.

Use `sgdisk` to create a 500M EFI partition, a linux root partition (in our case 40 GB), and a linux partition with the rest of the hard disks space:

```
# sgdisk -o -n 1:0:+500M -t 1:EF02 -n 2:0:+40G -t 2:EF00 -n 3:0:0 -t 3:8300 /dev/sdX
```

3.3 Format

View the new block layout of the target USB device:

```
# lsblk /dev/sdX
```

You should now have three blocks on your target USB device: a 500MB block `/dev/sdX1` for your EFI partition, a 40GB block `/dev/sdX2` for your root directory, and a block for all the remaining memory `/dev/sdX3` which will be your home directory.

Format the 500MB EFI system partition with a `FAT32` filesystem:

```
# mkfs.fat -F32 /dev/sdX1
```

Format the Linux partitions `sdX2` and `sdX3` with an `ext4` filesystem:

```
# mkfs.ext4 /dev/sdX3
# mkfs.ext4 /dev/sdX2
```

3.4 Mount

Create a mountpoint for the root partition:

```
# mkdir /mnt
```

- Mount the `ext4` formatted **root** partition first; this is very important:

  ```
  # mount /dev/sdX2 /mnt
  ```

Create two mountpoints for the remaining partitions:

```
# mkdir /mnt/home /mnt/efi
```

- Mount the `FAT32` formatted **EFI** partition:

  ```
  # mount /dev/sdX1 /mnt/efi
  ```

- Mount the `ext4` formatted **home** partition:

  ```
  # mount /dev/sdX3 /mnt/home
  ```

Following the Arch Way, this guide intends to install the minimum number of packages necessary to create a portable working Linux system.

4 Configure the Base System

Download and install the Arch Linux base packages as well as some additional packages to support wifi interfaces, and sudo users.

```
# pacstrap /mnt linux linux-firmware base vim grub efibootmgr iwd sudo polkit
```

4.1 Fstab

Generate a new `fstab` using `UUIDs` as source identifiers for root and home:

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

Add the proper path to the entries in the resulting `/mnt/etc/fstab` file if they are missing:

```
# vim /mnt/etc/fstab
```

```
# /dev/nvme0n1p2
UUID=81xxxxx-xxxx-xxx   /       ext4    rw,relatime     0 1

# /dev/sdX3
UUID=66xxxxx-xxxx-xxx   /home   ext4    rw,relatime     0 2

# /dev/sdX1
UUID=05xx-xxxx          /efi    vfat    rw,relatime,fmask=0022,dmask=0022,
codepage=437,iocharset=ascii,shortname=mixed,utf8,errors=remount-ro   0 2
```

4.2 Chroot

All configuration can be performed within a `chroot` environment.

Begin by chrooting into the new system:

```
# arch-chroot /mnt
```

Inside a chroot the names of block devices may differ from the host system. View the block device list with `lsblk` and note the new name of device `/dev/sdY`:

4.3 Locale

Use tab-completion to discover the appropriate entries for region and city:

```
# ln -sf /usr/share/zoneinfo/region/city /etc/localtime
```

Generate /etc/adjtime:

```
# hwclock --systohc
```

Edit /etc/locale.gen and uncomment your desired language (for US English, uncomment en_US.UTF-8 UTF-8):

```
# vim /etc/locale.gen
```

Generate the locale information:

```
# locale-gen
```

Set the LANG variable in /etc/locale.conf:

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

If you set the console keyboard layout, make the changes persistent in /etc/vconsole.conf:

```
# echo KEYMAP=de_CH-latin1 > /etc/vconsole.conf
```

4.4 Hostname

Create the /etc/hostname file containing your desired valid hostname on a single line:

```
# echo hostname > /etc/hostname
```

Edit /etc/hosts to contain only the following content:

```
# vim /etc/hosts
```

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

Check everything has been installed properly.

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

4.5 Bootloader

The GRUB bootloader along with efibootmgr will enable the USB to boot in both BIOS and UEFI boot modes.

Install the grub and efibootmgr packages:

```
# pacman -S grub efibootmgr
```

Install GRUB for MBR/BIOS booting mode:

```
# grub-install --target=i386-pc --boot-directory /boot --removable /dev/sdY
```

Install GRUB for UEFI booting mode:

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

Generate a GRUB configuration:

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

Network interface names

To ensure that the main ethernet and wifi interfaces will always be named `eth0` and `wlan0`, revert to traditional device naming:

```
# ln -s /dev/null /etc/udev/rules.d/80-net-setup-link.rules
```

4.6 Networking

Create a new `systemd-networkd` configuration file with the following content to automatically establish wired network connections:

```
# vim /etc/systemd/network/20-ethernet.network
```

```
[Match]
Name=en*
Name=eth*

[Network]
DHCP=yes
IPv6PrivacyExtensions=yes

[DHCPv4]
RouteMetric=100

[IPv6AcceptRA]
RouteMetric=100
```

Enable networkd and resolved:

```
# systemctl enable systemd-networkd.service
# systemctl enable systemd-resolved.service
```

Enable `iwd` to allow user control over wireless interfaces:

```
# systemctl enable iwd.service
```

Copy the wired network configuration file and edit it to the following to handle wifi interfaces:

```
# cp /etc/systemd/network/20-ethernet.network /etc/systemd/network/20-wlan.network
# vim /etc/systemd/network/20-wlan.network
```

```
[Match]
Name=wl*

[Network]
DHCP=yes
IPv6PrivacyExtensions=yes

[DHCPv4]
RouteMetric=600

[IPv6AcceptRA]
RouteMetric=600
```

Enable network time synchronization:

```
# systemctl enable systemd-timesyncd.service
```

4.7 Account Passwords

Set the root password:

```
# passwd
```

Create a new user account and set its password:

```
# useradd -m -G sys,wheel,users,adm,log -s /bin/bash user
# passwd user
```

Sudo

Enable sudo for the wheel group by umcommenting:

```
# EDITOR=vim visudo

    %wheel ALL=(ALL) ALL
```

Logout

Logout of the chroot:

```
# exit
```

Unmount the USB:

```
# umount /mnt/boot /mnt/home /mnt && sync
```

5 Installation Complete!

Welcome to Arch Linux! First time users should probably checkout the ArchWiki's general recommendations.