What is actually happening when you install Arch Linux?

Tristan Miller

February 2026

Arch Linux

- Many distros provide a graphical installer to make the installation process easy
- Not Arch!
- https://wiki.archlinux.org/title/Installation_guide

Overall steps

- Prepare the installation medium
- Boot and connect to internet
- Partition the disk
- Install the base system
- Configure the system

1. Pre-installation

Acquire an installation image

- ISO: a disk image
  - Contains an entire Arch installation and some useful utilities
- Can be downloaded over HTTP(s) or Torrent
  - *Mirrors* volunteer to host ISOs and packages
  - Torrent talk some other time :)

Verify signature

- Why should I trust the people who host mirrors?
  - I shouldn't!
- Arch also releases *signed hashes* for each ISO
- Check that the signature on the hash is valid and that the hash matches the ISO

Boot the live environment

- By default your computer boots from its internal drive.
- Tell your computer to boot from the ISO instead:
  - Copy it to a drive (usually USB flash drive) [or use PXI]
  - Enter your BIOS and tell it to boot from the flash drive

Set the console keyboard layout and font

```
localectl list-keymaps
loadkeys <keymap>
setfont <font>
```

- Console keymap: keyboard layout for the TTY
- The default keymap is `US`, so you don't usually need to change it
- Changing the font may be useful for high DPI screens

Verify the boot mode

- BIOS: the first thing that gets run after your computer turns on
  - Initializes the system, tests hardware, loads and executes a boot loader
  - Allows for basic configuration (system time, boot order, performance settings, etc.)
- All modern systems use UEFI, a successor to BIOS
  - Most VMs use legacy BIOS by default

Verify the boot mode

```
cat /sys/firmware/efi/fw_platform_size
```

- 64 — 64-bit UEFI
- 32 — 32-bit UEFI (odd)
- Does not exist — legacy BIOS

Connect to the internet

- The main guide requires an internet connection to download packages
  - See https://wiki.archlinux.org/title/Offline_installation

```
ip link
```

- `ip`: ubiquitous networking utility across many Linux distros
- Shows and lets you edit network devices, interfaces, and routes
- `ip link`: shows all network devices
- Wired connections are easiest — should work automatically

Connect to the internet

Wi-Fi is a little trickier

```
iwctl device list
iwctl station <devname> scan
iwctl station <devname> get-networks
iwctl station <devname> connect <SSID>
```

- Arch installer uses `iwd` for Wi-Fi
  1. Find your device's name
  2. Scan for available networks
  3. List available networks
  4. Connect to a network (you'll be prompted for a password)

Connect to the internet

What about eduroam?

- **Avoid at all costs!** Use a mobile hotspot or RIT-WiFi instead
- It is doable, but you'd need to prepare your certs ahead of time and somehow get them into the installer
- https://wiki.ritlug.com/eduroam/iwd may be useful

Connect to the internet

Test your connection with `ping`

```
ping ping.archlinug.org
```

- ICMP: Internet Control Message Protocol
  - Network protocol like TCP or UDP
  - Used mostly for diagnostics and connectivity testing
- `ping` — sends ICMP Echo Request packets to a server
- Most servers respond with an ICMP Echo Reply automatically

Check the system clock

`timedatectl`

- NTP (Network Time Protocol) is used to synchronize computer's clocks
- Should happen automatically once connected to the internet
- `timedatectl` – systemd command to get and set the system time

Partition the disks

Here we go!

```
fdisk -l
```

- List available disks

  ```
  fdisk /dev/whichever_disk_i_want
  ```

Partition the disks

- One drive can contain multiple *partitions*
- Different types of data or file systems
  - Boot info
  - Swap
  - Operating system files
  - Data files
- Use `fdisk` or `cfdisk`

Partition the disks

What partitions do I need?

- Root, of course
- UEFI requires a boot partition
- Swap partition is recommended, required for hibernation

Each partition will get its own "file" in `/dev`

- eg. `/dev/sda` partitions would be `/dev/sda1`, `/dev/sda2`, etc.

Format the partitions

- Partitions start out with random data
- Root: use a Linux filesystem (ext4, btrfs, probably not bcachefs)
- UEFI partitions needs to be FAT32
- Swap partition needs to be initialized

```
mkfs.ext4 /dev/<root>
mkfs.fat -F 32 /dev/<efi>
mkswap /dev/<swap>
```

Mount the file systems

- The installer file system is stored in memory
  - all changes you make are temporary
- Mount the new system's partitions to `/mnt` so we can start setting it up
- Enable the swap partition

```
mount /dev/<root> /mnt
mkdir /mnt/boot
mount /dev/<efi> /mnt/boot
swapon /dev/<swap>
```

2. Installation

Select the mirrors

- Edit `/etc/pacman.d/mirrorlist` to choose some mirrors close to you
  - `https://mirrors.rit.edu/archlinux/$repo/os/$arch`

Install essential packages

```
pacstrap -K /mnt base linux linux-firmware
```

- `pacstrap` creates a new system by installing packages using the installer's `pacman`
- `-K`: create a keyring
- `/mnt`: the directory to install to
- Then a list of packages you want to install

Install essential packages

- `base`: the base Arch system, required
- `linux` or `linux-lts` or etc. (good to have a kernel)
- `linux-firmware`
- Can also install other things now, or wait until later

3. Configure the system

Fstab

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

- `/etc/fstab` tells the kernel where to mount partitions at boot
- `genfstab`: generate an fstab based on current mounts
- `/mnt`: treat this directory as root
- `-U`: use partition UUIDs

Chroot

```
arch-chroot /mnt
```

- `chroot`: run a command with a different root directory
  - Part of how containers work
  - We can interact with the new system *as if we've booted into it*
- We also need devfs, tmpfs, sysfs, procfs, etc to be available
- `arch-chroot` will mount those for us

Time

```
ln -sf /usr/share/zoneinfo/<area>/<location>
    /etc/localtime
```

- Set the system's time zone
- `/usr/share/zoneinfo` contains time zone files (binary format!)
- `ln -s` creates a *symlink*: special file that references another file by its path

Time

```
hwclock --systohc
```

- Set the hardware clock based on the system time
- Also updates `/etc/adjtime`, keeps track of clock drift

Localization

- Edit `/etc/locale.gen` to choose which locales you want
  - Text encoding, date format, currency, decimal separator
  - `en_US.UTF-8 UTF-8`

```
locale-gen
```

- Compiles the requested locales to a binary format `glibc` can read

Localization

Edit `/etc/locale.conf` to enable a locale

```
LANG=en_US.UTF-8
```

- `$LANG` controls the default locale
- `$LC_TIME`, `$LC_COLLATE`, `$LC_MONETARY`, etc. can also be set to override the default

Network configuration

- Edit `/etc/hostname` to change your system's hostname
- **Install networking software!**
  - NetworkManger is the most popular, `pacman -S networkmanager`

Root password

- Use `passwd` to change your root password
- Can add other users now too
  - `useradd -m <user>`, `passwd <user>`
  - `-m`: create a home directory

Bootloader

Install a bootloader! Otherwise you can't boot!

- Loaded by the BIOS/UEFI, takes care of the rest of the boot process
  - Load the kernel, set kernel parameters, load the initramfs
  - May allow you to select multiple boot options
- `GRUB` is most popular

```
# for UEFI systems
pacman -S grub efibootmgr
grub-mkconfig -o /boot/grub/grub.cfg
grub-install --target=x86_64-efi
    --efi-directory=/boot --bootloader-id=GRUB
```

Reboot

Press `Ctrl+D` to exit the chroot.

Run `reboot` to reboot — remember to remove your installation medium!

All done!

yay :)

4. After installing

After installing

- Log in and make sure everything works
- Connect to the internet
  - hope you remembered to install a networking package!
- Install some drivers/firmware
- Install a graphical environment
- Have fun!