Installation

It is possible to install any init system (openrc, runit, s6) from any ISO.

Contents

- Fresh Artix installation from bootable media
  - Set the keyboard layout
  - Partition your disk (BIOS)
  - Format partitions
  - Mount Partitions
  - Connect to the internet
  - Install base system
    - Install a kernel
- Configure the base system
  - Set system clock
  - Localization
  - Boot Loader
  - Add user(s)
  - Network configuration
    - OpenRC
    - Runit
    - s6
  - Reboot the system
- Post installation configuration
  - Desktop Environment
  - Display Login Manager

Fresh Artix installation from bootable media

Artix can either be installed through the console or the GUI installer. The GUI install is quite straightforward, we'll focus on the console installation procedure here. The installation images are confirmed to work on both BIOS and UEFI systems.

Set the keyboard layout

To check the available layout types:

```
ls -R /usr/share/kbd/keymaps
```

Then type the name of the layout without the extension. For example, to set the Spanish(Spain) layout, type:

```
loadkeys es
```

This only sets the selected keyboard layout in the current tty and only until reboot. To make the setting permanent, you have to edit `/etc/conf.d/keymaps` in the installed system.

Partition your disk (BIOS)

Partition your hard drive (`/dev/sda` will be used in this guide) with **fdisk** or **cfdisk**, the partition numbers and order are at your discretion:

```
cfdisk /dev/sda
```

If you want to install side-by-side with other operating systems, you must make some space on the disk by resizing the existing partitions. You may use **gparted** for this, however detailed instructions are out of the scope of this guide. See the ArchWiki

NOTE: The BIOS boot partition is necessary on UEFI systems with a GPT-partitioned disk. EFI system partition has to be created and mounted at `/mnt/boot` and the suggested size is around 512 MiB.

Format partitions

Next, format the new partitions, we will use ext4 in this example:

```
mkfs.ext4 -L ROOT /dev/sda2        <- root partition
mkfs.ext4 -L HOME /dev/sda3        <- home partition, optional
mkfs.ext4 -L BOOT /dev/sda4        <- boot partition, optional
mkswap -L SWAP /dev/sda1           <- swap partition
```

The -L switch assigns labels to the partitions, which helps referring to them later through `/dev/disk/by-label` without having to remember their numbers

If you are doing a UEFI installation, the boot partition is not optional and needs to be formatted as fat32.

```
mkfs.fat -F 32 /dev/sda4
```

Mount Partitions

Now, activate your swap space and mount your partitions:

```
swapon /dev/disk/by-label/SWAP             (if created)
mount /dev/disk/by-label/ROOT /mnt
mkdir /mnt/boot
mkdir /mnt/home
mount /dev/disk/by-label/HOME /mnt/home    (if created)
mount /dev/disk/by-label/BOOT /mnt/boot    (if created)
```

Connect to the internet

A working internet connection is required and assumed. A wired connection is setup automatically, if found. Wireless ones must be configured by the user. Verify your connection before you proceed. If you want connect through a wireless interface you should use wpa_supplicant and dhcpcd to set it up.

```
ping artixlinux.org
```

Install base system

Use **basestrap** to install the **base** and optionally the **base-devel** package groups and your preferred init (currently available: `openrc`, `runit`, and `s6`):

```
basestrap /mnt base base-devel openrc
```

or

```
basestrap /mnt base base-devel runit elogind-runit
```

or

```
basestrap /mnt base base-devel s6 elogind-s6
```

If you encounter errors, you can use the **-i** flag of **basestrap** ('interactive'). Example:

```
basestrap -i /mnt base
```

and you will be prompted to choose the respective **elogind**.

Install a kernel

Artix provides two kernels `linux-lts` and `linux`, but you can use any other kernel you like ('-ck, -pf, -zen' etc). While optional, it is recommended to install `linux-firmware`. You can try not installing it, but some devices such as network cards may not work.

```
basestrap /mnt linux linux-firmware
```

or

```
basestrap /mnt linux-lts linux-firmware
```

Use **fstabgen** to generate `/etc/fstab`, use -U for UUIDs and -L for partition labels:

```
fstabgen -U /mnt >> /mnt/etc/fstab        <- edit and verify, also set
root, swap, home and etc..
```

Check the resulting fstab for errors before rebooting. Now, you can chroot into your new Artix system with:

```
artools-chroot /mnt
```

Configure the base system

Set system clock

Set the time zone:

```
ln -sf /usr/share/zoneinfo/Region/City /etc/localtime
```

Run hwclock to generate /etc/adjtime:

```
hwclock --systohc
```

Note that this will default to UTC. If you use Windows and you want the time to be synchronized in both Artix and Windows, follow System_time#UTC_in_Windows:ArchWiki for instructions to enable UTC in there also.

Localization

Install a text editor of your choice (let's use **nano** here) and edit `/etc/locale.gen`, uncommenting the locales you desire:

```
pacman -S nano
nano /etc/locale.gen
```

Generate your desired locales running:

```
locale-gen
```

To set the locale systemwide, create or edit `/etc/locale.conf` (which is sourced by `/etc/profile`) or `/etc/bash/bashrc.d/artix.bashrc` or

`/etc/bash/bashrc.d/local.bashrc` ; user-specific changes may be made to their respective `~/.bashrc`, for example:

```
export LANG="en_US.UTF-8"     <-- localize in your languages
export LC_COLLATE="C"
```

Boot Loader

First, install **grub** and **os-prober** (for detecting other installed operating systems):

```
pacman -S grub os-prober efibootmgr
grub-install --recheck /dev/sda (for BIOS systems)
grub-install --target=x86_64-efi --efi-directory=/boot --bootloader-id=grub
(for UEFI systems)
grub-mkconfig -o /boot/grub/grub.cfg
```

NOTE: On GPT-partitioned drives, refer to ArchWiki:Grub article.

Alternatively, you can use some other bootloaders or even the own kernel using EFISTUB. Bear in mind that we won't cover them here since most are not supported officially and the Arch Wiki already provides enough documentation. See the ArchWiki:Bootloader article if you want to view the alternatives.

Add user(s)

First, set the root passwd:

```
passwd
```

Second, create a regular user and password:

```
useradd -m user
passwd user
```

Network configuration

Create the hostname file and install a DHCP client:

```
nano /etc/hostname
coolhostname
pacman -S dhcpcd
```

If you want to use a wireless connection, be sure to also install **wpa_supplicant**.

Note: In Runit and and s6, enabling a service by default at this step requires a

different command than the normal one because both init systems rely on a /run (a tmpfs) to be created. That creation accours when you actually boot into the new system. You can choose to skip these steps and and enable the services *after* a reboot if you wish using the commands listed on the Runit and s6 pages. Just be sure you have an internet daemon and its respective service script installed first. In this example, we'll use *connman*.

OpenRC

Install **connman** and optionally a front-end:

```
pacman -S connman-openrc connman-gtk (or cmst for Qt-based DEs)
rc-update add connmand
```

Runit

Install **connman** and optionally a front-end:

```
pacman -S connman-runit connman-gtk (or cmst for Qt-based DEs)
ln -s /etc/runit/sv/connmand /etc/runit/runsvdir/default
```

s6

Install **connman** and optionally a front-end:

```
pacman -S connman-s6 connman-gtk (or cmst for Qt-based DEs)
s6-rc-bundle -c /etc/s6/rc/compiled add default connmand
```

Alternatively, if you will use `openrc`, Gentoo's **netifrc** modules can be used, these are located in /etc/init.d/ and work on a script-per-interface basis.

```
ip -s link                <- Get the exact name of your interface
nano /etc/conf.d/net        <- Add config_<interface>="dhcp"
```

Now the parent script `/etc/init.d/net.lo` should be symlinked to create additional scripts for each network interface and then loaded into an openrc runlevel.

```
ln -s /etc/init.d/net.lo /etc/init.d/net.<interface>
rc-update add net.<interface> default
```

Reboot the system

Now, you can reboot and enter into your new installation:

```
exit                            <- exit chroot environment
umount -R /mnt
```

```
reboot
```

Post installation configuration

Once shutdown is complete, remove your installation media. If all went well, you should boot into your new system. Log in as your root to complete the post-installation configuration. To get a graphical environment you need to install the xorg group:

```
pacman -S xorg
```

Choose your packages, or just install all of them. For the closed-source nvidia drivers you can use the **nvidia** package:

```
pacman -S nvidia
```

If you want to run a kernel other than **linux**, you can install the respective nvidia dkms package which ensures all installed kernels get their nvidia modules. AMD and Intel cards enjoy excellent (or near-excellent) 3D support with open-source drivers. Read the Arch wiki, for information on how Xorg chooses the best available video driver and which one is optimal for your hardware.

Desktop Environment

Install a desktop environment, for example **MATE**, **XFCE4** or **LXQt**:

```
pacman -S mate mate-extra system-config-printer blueman connman-gtk

pacman -S xfce4 xfce4-goodies

pacman -S lxqt
```

Display Login Manager

A display manager, like **LightDM**, **LXDM** or **SDDM** may be desirable:

```
pacman -S displaymanager-openrc lightdm (or lxdm or sddm)
```

If using OpenRC:

```
rc-update add xdm default
nano /etc/conf.d/xdm        <- edit and set DISPLAYMANAGER="lightdm"
```

Or you can use .xinitrc to launch your DE manually; edit (or create) `~/.xinitrc` and add **exec mate-session**. Notice: **mate-session** and quite a few other packages from

the Arch repositories are compiled against systemd even if they don't actually use it, at least not as PID1; to satisfy the library link you may install **elogind** and its services files **elogind-openrc**, **elogind-runit**, or **elogind-s6**.

```
pacman -S elogind
```
