Arch-Installation-And-Setup-LVM

Legend:

| Red text - optional things
| *Yellow Italic text* - placeholder
| Green text - Important note
| Blue text - unnecessary command that checks if another (usually the previous) command has executed correctly, or to simply check things on the system
| ~~Striketrough~~ - I have no idea why this is here but it might be important so i am not deleting it
---

NOTE: This instalation guide isn't perfect and although it can be used on its own I strongly recomend having the arch wiki open at all times.
The information shown bellow is compiled from the wiki, some youtube videos and other random resources found across the web.
All necessary links will be provided at the end of the document!
The document is split in two parts: installation and setup, the installation is everything you do while the computer is booted from the usb. The setup is everything after and can be completely ignored

Instalation:

1. Verify boot mode:
$ ls /sys/firmware/efi/efivars

Desired output:

```
root@archiso ~ # ls /sys/firmware/efi/efivars
BackgroundClear-4d1ede05-38c7-4a6a-9cc6-4bcca8b38c14       LoaderEntries-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
Boot0000-8be4df61-93ca-11d2-aa0d-00e098032b8c              LoaderEntrySelected-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
Boot0001-8be4df61-93ca-11d2-aa0d-00e098032b8c              LoaderFeatures-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
Boot0002-8be4df61-93ca-11d2-aa0d-00e098032b8c              LoaderFirmwareInfo-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
Boot0003-8be4df61-93ca-11d2-aa0d-00e098032b8c              LoaderFirmwareType-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
BootCurrent-8be4df61-93ca-11d2-aa0d-00e098032b8c           LoaderImageIdentifier-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
BootOptionSupport-8be4df61-93ca-11d2-aa0d-00e098032b8c     LoaderInfo-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
BootOrder-8be4df61-93ca-11d2-aa0d-00e098032b8c             LoaderTimeExecUSec-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
ConInDev-8be4df61-93ca-11d2-aa0d-00e098032b8c              LoaderTimeInitUSec-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
ConIn-8be4df61-93ca-11d2-aa0d-00e098032b8c                 LoaderTimeMenuUSec-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
ConOutDev-8be4df61-93ca-11d2-aa0d-00e098032b8c             MTC-eb704011-1402-11d3-8e77-00a0c969723b
ConOut-8be4df61-93ca-11d2-aa0d-00e098032b8c                OsIndicationsSupported-8be4df61-93ca-11d2-aa0d-00e098032b8c
FirmwareFeaturesMask-4d1ede05-38c7-4a6a-9cc6-4bcca8b38c14  PlatformLangCodes-8be4df61-93ca-11d2-aa0d-00e098032b8c
FirmwareFeatures-4d1ede05-38c7-4a6a-9cc6-4bcca8b38c14      PlatformLang-8be4df61-93ca-11d2-aa0d-00e098032b8c
Key0000-8be4df61-93ca-11d2-aa0d-00e098032b8c               PlatformRecovery0000-8be4df61-93ca-11d2-aa0d-00e098032b8c
Key0001-8be4df61-93ca-11d2-aa0d-00e098032b8c               Timeout-8be4df61-93ca-11d2-aa0d-00e098032b8c
LangCodes-8be4df61-93ca-11d2-aa0d-00e098032b8c             VarErrorFlag-04b37fe8-f6ae-480b-bdd5-37d98c5e89aa
Lang-8be4df61-93ca-11d2-aa0d-00e098032b8c                  boot-args-7c436110-ab2a-4bbb-a880-fe41995c9f82
root@archiso ~ # _
```

2. Verify internet connection:
$ ping -c3 archlinux.org

Desired output:

```
root@archiso ~ # ping -c3 archlinux.org
PING archlinux.org (138.201.81.199) 56(84) bytes of data.
64 bytes from apollo.archlinux.org (138.201.81.199): icmp_seq=1 ttl=63 time=35.0 ms
64 bytes from apollo.archlinux.org (138.201.81.199): icmp_seq=2 ttl=63 time=35.8 ms
64 bytes from apollo.archlinux.org (138.201.81.199): icmp_seq=3 ttl=63 time=35.8 ms

--- archlinux.org ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 34.978/35.534/35.823/0.393 ms
root@archiso ~ # _
```

3. Update the system clock:

$ timedatectl set-ntp true

4. Partition disks:
$ cfdisk
 - gtp

| The scheme we will use will have a boot, root, swap and home partitions.
| boot - 512MB; type - EFI partition

| root - 5G; type - Linux filesystem/Linux LVM
| swap - twice the ram; type - Linux swap
| home (the rest); type - Linux filesystem !The home partition is optional

---

Encryption:

| $ modprobe dm-crypt
| $ modprobe dm-mod
|
| $ cryptsetup -v luksFormat device
| device examples: /dev/sda1 , /dev/nvme0n1p1
|
| To be able to work with the device:
| $ cryptsetup open --type luks *device name*
| !Device and name are two different things

! If you have a seperate home partion, you will have to repeat the same steps for that partiotion

Configure LVM:

| Create a physical volume:
| $ pvcreate /dev/mapper/*name*
|
| Create a volume group:
| $ vgcreate *groupName* /dev/mapper/name
| example $ vgcreate vg1 /dev/mapper/lvm
|
| Creating the logical volumes (root, swap home):
| $ lvcreate -L *G *groupName* -n *volumeName*
| example $ lvcreate -L 15G vg1 -n root
| example $ lvcreate -L 100%FREE vg1 -n home

---

5. Adding filesystems to these partitions:

Encryption LVM:

| $ mkfs.ext4 /dev/*groupName* / *volumeName*
| - Repeat for home
| - For swap use mkswap /dev/*groupName* / *volumeName*
|
| For the boot partition:
| $ mkfs.fat -F32 /dev/sda*

Encryption NO lvm

| $ mkfs.ext4 /dev/mapper/*name*
|
| For the boot partition:
| $ mkfs.fat -F32 /dev/sda*

No encryption no lvm:

| For the root partition:
| $ mkfs.ext4 /dev/sda*
|
| For the swap partition:
| $ mkswap /dev/sda*
| $ swapon /dev/sda*
|
| For the home partition:
| $mkfs.ext4 /dev/sda*

---

6. Mounting the filesystems:

Encryption LVM:

| $ mount /dev/*groupName* / *volumeName* /mnt
|
| To mount home:
| $ mkdir /mnt/home
|
| Mounting the efi partition:
| $ mkdir /mnt/boot
| $ mkdir /mnt/boot/efi
| $ mount /dev/sda* /mnt/boot/efi
|
| Here is where we activate the swap:

| $ swapon /dev/*groupName*/*volumeName*

Encryption no lvm:

| Mounting root:
| $ mount /dev/mapper/*name* /mnt
|
| Mounting the efi partition:
| $ mkdir /mnt/boot
| $ mkdir /mnt/boot/efi
| $ mount /dev/sda* /mnt/boot/efi

No encryption no lvm:

| Mounting the root partition:
| $ mount /dev/sda* /mnt
|
| Mounting the efi partition:
| $ mkdir /mnt/boot
| $ mkdir /mnt/boot/efi
| $ mount /dev/sda* /mnt/boot/efi
| Moutning the home partition:
| $ mkdir /mnt/home
| $ mount /dev/sda* /mnt/home
---

7. Picking the mirrors, which are located in `/etc/pacman.d/mirrorlist`:
Mirror generator: https://www.archlinux.org/mirrorlist/
The mirrors for Bulgaria: https://github.com/kub4e/Arch-Linux-Files

Using github (In this case mine):
$ git clone https://github.com/kub4e/Arch-Linux-Installation-And-Config-Files
$ cd Arch-Linux-Files
$ cat /dev/null > /etc/pacman.d/mirrorlist
$ cat Arch-Linux-Mirrors-Bulgaria > /etc/pacman.d/mirrorlist

8. The pacstrap command:
$ pacstrap /mnt base base-devel linux linux-firmware efibootmgr networkmanager zsh/bash texinfo intel-ucode/amd-ucode opendoas vim git dhcpcd dhclient man-db man-pages openssh parted wget
- For lvm add lvm2

9. Generate the filesystem table
$ genfstab -U /mnt >> /mnt/etc/fstab

10. Chroot into the system:
$ arch-chroot /mnt

Making a swap file:

| $ fallocate -l *GB /swapfile
| $ chmod 600 /swapfile
| $ mkswap /swapfile
| $ swapon /swapfile
| $ vim /etc/fstab
| - In the empy space at the end of the file add:
| /swapfile none swap defaults 0 0

Localization stuff:

| $ ln -sf /usr/share/zoneinfo/Europe/Sofia /etc/localtime
|
| Setting the hardware clock:
| $ hwclock --systohc
|
| Setting up locales:
| Edit `/etc/locale.gen` and uncomment `en_US.UTF-8 UTF-8` and other needed locales.
| $ echo “LANG=en_US.UTF-8” > /etc/locale.conf
| $ locale-gen

User managment:

| Creating a root password:
| $ passwd
|
| Add another user:

| $ useradd -m *username*
|
| Setting the password for that user:
| $ passwd *username*
|
| The steps bellow apply only if you want to use sudo (Bellow you can find an alternative):
| Check if sudo is installed (If the output is a path, than sudo is installed):
| $ whereis sudo
|
| Make the user an admin:
| Check the groups of the user:
| $ usermod -aG wheel,audio,video,optical,storage username
| $ groups *username*
| Add the user to the sudoers file:
| $ visudo
| Uncomment this line:

```
##
## User privilege specification
##
root ALL=(ALL) ALL

## Uncomment to allow members of group wheel to execute any command
# %wheel ALL=(ALL) ALL
```

Network stuff:

| Setting a hostname:
| $ echo *hostname* > /etc//hostname
| $ vim /etc/hosts

```
# Static table lookup for hostnames.
# See hosts(5) for details.
#
127.0.0.1       localhost
::1             localhost
127.0.1.1       domainname.localdomain domainname_
```

| the last line can also be with 127.0.0.1 (i guess)

| Making sure we have internet on reboot (Capitalization is important!!!):
| $ systemctl enable NetworkManager
| Desired output:

```
[root@archiso /]# systemctl enable NetworkManager
Created symlink /etc/systemd/system/multi-user.target.wants/NetworkManager.service → /usr/lib/systemd/system/NetworkManager.serv
ice.
Created symlink /etc/systemd/system/dbus-org.freedesktop.nm-dispatcher.service → /usr/lib/systemd/system/NetworkManager-dispatch
er.service.
Created symlink /etc/systemd/system/network-online.target.wants/NetworkManager-wait-online.service → /usr/lib/systemd/system/Net
workManager-wait-online.service.
[root@archiso /]# _
```

Encryption:

| Modify mkinitcpio.conf:
| $ vim /etc/mkinitcpio.conf
|
| - The HOOKS line should look like this:
| HOOKS=(base **udev** autodetect **keyboard keymap** consolefont modconf block **encrypt** lvm2 filesystems fsck)
|
| - Should probably add encrypt keyboard and keymap, keymap is not needed if the default keyboard layout is us
|
| Recreate the image:
| $ mkinitcpio -p linux

Bootloader (GRUB):

| Install and configure grub:

| $ pacman -S grub
|
| $ grub-install --target=x86_64-efi --efi-directory=/boot/efi --bootloader-id=GRUB
|
| Encryption:
| ---
| | Getting the encypted disk UUID:
| | $ ls -l /dev/disk/by-uuid | grep -o '(?<=25) [^]*'
| |
| | Edit the GRUB_CMDLINE_LINUX line in /etc/default/grub as shown bellow:
| | GRUB_CMDLINE_LINUX="cryptdevice=UUID=*UUID*:*name* root=/mapper/*name*" or root=/dev/*groupName*/*volumeName* for lvm
  example: `GRUB_CMDLINE_LINUX="cryptdevice=UUID=c15b50fc-06cd-4c16-8943-8b478ac09b55:cp_root root=/dev/mapper/cp_root"`
  $ grub-mkconfig -o /boot/grub/grub.cfg

Configuring doas:
$ vim /etc/doas.conf
- Add:
permit *username* as root

Setup:

! If you have read to here you should have a working arch installation, everything shown bellow is my personal choice, I would recommend ignoring it and customizing the system however you like !

Installing yay:
$ git clone https://aur.archlinux.org/yay.git
$ cd yay
$ makepkg -is PKGBUILD

Check for updates:
$ sudo pacman -Syu

Install xorg:
$ sudo pacman -S xorg-server xorg-xinit
- xorg-xinit is the package that allows usage of startx

Install fonts:
- There are many ways to install fonts, the easiest is to go to https://github.com/kub4e/Arch-Linux-Files and choose a font package after that:
$ doas pacman -S *packagename*
- The fonts config file is ~/.config/fontconfig/fonts.conf

Random Notes:

! This section contains random, but useful notes !

Shells:
To list all available shells:
$ chsh -l
Changing the default shell:
$ chsh -s *full-path-to-shell*
Example:
$ chsh -s /usr/bin/zsh
Links:
https://wiki.archlinux.org/index.php/Zsh

Fonts:
$ doas pacman -S fontconfig

Archives:
extracting:
tar:
$ tar xvf *archive*
gzip:
$ gzip -d *archive*
bzip:
$ bzip2 -d *archive*
zip:
uzip *archive*

St:
$ wget https://dl.suckless.org/st/st-0.8.4.tar.gz

$ tar xvf st-0.8.4.tar.gz
$ cd st-0.8.4.tar.gz
$ doas make install

Theming:

- Useful webiste: https://terminal.sexy

Dwm:

$ wget https://dl.suckless.org/dwm/dwm-6.2.tar.gz
$ tar xvf dwm-6.2.tar.gz
$ cd dwm-6.2.tar.gz
$ doas make install

Surf:

Dependencies:

$ pacman -S gtk3
$ pacman -S gcr
$ pacman -s webkit2gtk

Install:

$ wget https://dl.suckless.org/surf/surf-2.0.tar.gz
$ tar xvf surf-2.0.tar.gz
$ cd surf-2.0.tar.gz
$ sudo make clean install

Dmenu:

$ wget https://dl.suckless.org/tools/dmenu-4.9.tar.gz
$ tar xvf dmenu-4.9.tar.gz
$ cd dmenu-4.9.tar.gz
$ sudo make clean install

Feh:

$ sudo pacman -S feh
$ feh --bg-scale /path/to/image.file
$ vim .xinitrc
- Add:
~/.fehbg &

Change Resolution:

$ xrandr
$ xrandr --output *type-from-xrandr-command* --mode *resolution*

Picom:

Link: https://wiki.archlinux.org/index.php/Picom#Configuration
$ sudo pacman -S picom
$ cp /etc/xdg/picom.conf ~/.config/picom/
$ picom --config ~/.config/picom/