

ARCH LINUX + GNOME + INTEL + LVM + SECURE BOOT + UEFI

1. UEFI & KEYBOARD & LOCALE

IMPORTANT: MAKE SURE THAT SECURE BOOT WITH NO KEYS ENROLLED IS SET

```
ls /sys/firmware/efi/efivars
efibootmgr
```

```
loadkeys br-abnt2
nano /etc/locale.gen ( unmark LANG=pt_BR.UTF-8 )
locale-gen
cat /etc/locale.conf
echo 'LANG=pt_BR.UTF-8' > /etc/locale.conf
```

2. NETWORK & CONNECTIVITY BY WI-FI

```
ip link
iwctl
device list
station wlan0 scan
station wlan0 get-networks
station wlan0 connect
exit
ping 1.1.1.1
timedatectl
```

```
ls /usr/share/kbd/consolefonts/ | grep ter-120b
if yes: setfont ter-120b
if not: pacman -S kbd terminus-font ttf-terminus-nerd
then : setfont ter-120b
```

3. PARTITIONING & FORMATTING & MOUNTING NVME LVM

```
parted /dev/nvme0n1 mklabel gpt
parted /dev/nvme0n1 mkpart ESP fat32 1MiB 1025MiB
parted /dev/nvme0n1 set 1 esp on
parted /dev/nvme0n1 mkpart CRYPTO 1025MiB 100%
```

```
mkfs.fat -F32 /dev/nvme0n1p1
```

```
cryptsetup luksFormat --type luks2 /dev/nvme0n1p2
cryptsetup open --allow-discards /dev/nvme0n1p2 cryptroot
```

```
pvcreeate /dev/mapper/cryptroot
vgcreate vg0 /dev/mapper/cryptroot
```

```
lvcreate -L 3G -n boot vg0
```

```
mkfs.ext4 /dev/vg0/boot
```

```
lvcreate -l 100%FREE -n root vg0
```

```
mkfs.btrfs -f -L ROOT /dev/vg0/root
```

```
mount /dev/vg0/root /mnt
```

```
btrfs subvolume create /mnt/@  
btrfs subvolume create /mnt/@home  
btrfs subvolume create /mnt/@log  
btrfs subvolume create /mnt/@pkg  
btrfs subvolume create /mnt/@snapshots
```

```
umount /mnt
```

```
mount -o subvol=@ /dev/vg0/root /mnt  
mkdir -p /mnt/{home,var/log,var/cache/pacman/pkg,var/snapshots}
```

```
mount -o subvol=@home /dev/vg0/root /mnt/home  
mount -o subvol=@log /dev/vg0/root /mnt/var/log  
mount -o subvol=@pkg /dev/vg0/root /mnt/var/cache/pacman/pkg  
mount -o subvol=@snapshots /dev/vg0/root /mnt/var/snapshots
```

```
mkdir -p /mnt/boot  
mount /dev/vg0/boot /mnt/boot
```

```
mkdir -p /mnt/boot/efi  
mount /dev/nvme0n1p1 /mnt/boot/efi
```

4. BASE SYSTEM INSTALLATION FOR INTEL CHIPSET

```
reflector --country CA,BR,PT,NO,CH --protocol https --latest 15 --score 10 --delay 1 --  
sort rate --save /etc/pacman.d/mirrorlist
```

```
pacstrap -K /mnt base intel-ucode linux-firmware linux-lts linux-lts-headers linux-zen  
linux-zen-headers vulkan-intel vulkan-tools sudo sbctl networkmanager nano lvm2  
efibootmgr dkms dracut cryptsetup btrfs-progs apparmor
```

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

5. CHROOT CONFIGURATION

```
arch-chroot /mnt
```

```
ln -sf /usr/share/zoneinfo/America/Araguaina /etc/localtime  
hwclock --systohc
```

```
nano /etc/locale.gen ( unmark pt_BR.UTF-8 UTF-8 )  
locale-gen  
echo 'LANG=pt_BR.UTF-8' > /etc/locale.conf  
echo 'KEYMAP=br-abnt2' > /etc/vconsole.conf  
echo 'sofos' > /etc/hostname
```

```
passwd ( root )  
useradd -m -g users -G wheel -s /bin/bash archer  
passwd archer
```

```
EDITOR=nano visudo
```

```
%wheel ALL=(ALL:ALL) ALL ( uncomment for enable sudo for "archer" user )
```

6. BOOTLOADER

a. bootctl config

```
blkid /dev/nvme0n1p2 ( catch PARTUUID code )
```

```
nano /boot/loader/entries/arch-zen.conf
```

```
title    Arch Linux ZEN
linux    /vmlinuz-linux-zen
initrd   /intel-ucode.img
initrd   /initramfs-linux-zen.img
options  rd.luks.name=<PARTUUID>=cryptroot rd.lvm.vg=vg0 root=/dev/vg0/root
rootfstype=btrfs rootflags=subvol=@ rw zswap.enabled=0 apparmor=1 security=apparmor
nvme_core.default_ps_max_latency_us=0 quiet loglevel=3
```

```
nano /boot/loader/entries/arch-lts.conf
```

```
title    Arch Linux LTS
linux    /vmlinuz-linux-lts
initrd   /intel-ucode.img
initrd   /initramfs-linux-lts.img
options  rd.luks.name=<PARTUUID>=cryptroot rd.lvm.vg=vg0 root=/dev/vg0/root
rootfstype=btrfs rootflags=subvol=@ rw zswap.enabled=0 apparmor=1 security=apparmor
nvme_core.default_ps_max_latency_us=0 quiet loglevel=3
```

IMPORTANT: dracut does not create or manage initramfs fallback

```
nano /boot/loader/loader.conf
```

```
default arch-zen.conf
timeout 2
console-mode max
editor no
```

b. sbctl config

```
sbctl status
sbctl create-keys
sbctl verify ( sign the EFI and Kernels according to example below )
```

```
sbctl sign -s /boot/EFI/BOOT/BOOTX64.EFI
sbctl sign -s /boot/EFI/systemd/systemd-bootx64.efi
sbctl sign -s /boot/vmlinuz-linux-lts
sbctl sign -s /boot/vmlinuz-linux-zen
```

```
sbctl verify
sbctl enroll-keys
sbctl list-enrolled-keys
sbctl status
```

c. hooks config

```
nano /etc/sysctl.d/99-sysctl.conf
```

```
kernel.kptr_restrict = 2
kernel.dmesg_restrict = 1
kernel.randomize_va_space = 2
fs.protected_hardlinks = 1
fs.protected_symlinks = 1
```

```
sysctl --system
```

```
mkdir -p /etc/pacman.d/hooks/
```

```
nano /etc/pacman.d/hooks/99-secureboot.hook
```

```
[Trigger]
```

```
Operation = Upgrade
```

```
Type = Package
```

```
Target = systemd
```

```
Target = linux-lts
```

```
Target = linux-zen
```

```
[Action]
```

```
Description = Signing EFI binaries and kernels for Secure Boot
```

```
When = PostTransaction
```

```
Exec = /bin/sh -c '(sbctl sign -s /boot/EFI/BOOT/BOOTX64.EFI 2>/dev/null || true) &&
(sbctl sign -s /boot/EFI/systemd/systemd-bootx64.efi 2>/dev/null || true) && (sbctl
sign -s /boot/vmlinuz-linux-lts 2>/dev/null || true) && (sbctl sign -s /boot/vmlinuz-
linux-zen 2>/dev/null || true)'
```

d. generate initramfs

```
ls /sys/firmware/efi/efivars
efibootmgr
```

```
lsblk -o NAME,SIZE,FSTYPE,TYPE,MOUNTPOINT
```

```
btrfs subvolume list /
```

```
sbctl verify
```

```
sbctl status
```

```
systemctl enable apparmor.service
```

```
systemctl enable NetworkManager.service
```

```
dracut -f -v
```

```
exit ( arch-chroot environment logoff )
```

```
umount -R /mnt
```

```
swapoff -a
```

reboot + F2 + Secure Boot + Key Management (verify keys or import keys if you did not has used sbctl enroll-keys) + F10

7. POST-INSTALLATION NETWORK CONFIGURATION

Login: "root"

```
systemctl start NetworkManager.service
```

```
nmcli general status
```

```
nmcli device status
```

```
nmcli device wifi list
```

```
nmcli device wifi connect "SSID" --ask
```

```
pacman -S kbd terminus-font ttf-terminus-nerd
```

```
setfont ter-120b
```

8. INSTALLING DESKTOP GNOME

```
pacman -S adwaita-icon-theme bluez bluez-libs bluez-obex bluez-utils colord eog gdm  
gnome gnome-backgrounds gnome-bluetooth gnome-boxes gnome-calendar gnome-characters  
gnome-clocks gnome-color-manager gnome-console gnome-control-center gnome-disk-utility  
gnome-firmware gnome-info-collect gnome-keyring gnome-logs gnome-menus gnome-music  
gnome-online-accounts gnome-power-manager gnome-session gnome-settings-daemon gnome-  
terminal gnome-text-editor gnome-themes-extra gnome-tweaks gnome-usage gnome-user-docs  
gnome-weather grilo-plugins gst-plugins-base gst-plugins-base-libs gvfs gvfs-afc gvfs-  
dnssd gvfs-goa gvfs-gphoto2 gvfs-mtp gvfs-nfs gvfs-smb gvfs-wsdd localsearch loupe  
mesa-utils mutter nautilus network-manager-applet orca pacman pacman-contrib reflector  
rygel seahorse system-config-printer tinysparql xdg-desktop-portal-gnome xdg-user-dirs-  
gtk xkeyboard-config yelp yelp-tools yelp-xsl
```

```
systemctl enable gdm.service
```

```
systemctl status gdm.service
```

```
reboot
```

Login: "user"

```
sudo systemctl start gdm.service ( if login was failed )
```

10. ESSENTIALS PACKAGES AND CONFIGURATIONS

Complementary applications and services to Desktop Gnome (Ctrl C + Ctrl V)

```
sudo pacman -S gnome-browser-connector gnome-shell-extension-appindicator gnome-shell-  
extension-arc-menu gnome-shell-extension-caffeine gnome-shell-extension-dash-to-panel  
gnome-shell-extension-desktop-icons-ng gnome-shell-extension-vitals gnome-shell-  
extension-weather-oclock gnome-shell-extensions
```

```
sudo pacman -S collision decibel chromium file-roller firewalld foliate font-manager  
fragments gedit gedit-plugins gparted gthumb eyedropper libreoffice-still libreoffice-  
still-pt-br mpv papers pavucontrol ptyxis qalculate-gtk shotwell showtime snapshot
```

```
sudo pacman -S audio-convert alsa-utils aspell aspell-en aspell-pt at-spi2-core avahi
bashtop bat bind-tools cronie cups cups-browsed cups-filters cups-pdf curl dconf
ethtool fail2ban fastfetch fd ffmpeg foomatic-db foomatic-db-engine foomatic-db-ppds
fzf fwupd git glances grc gutenprint hspell htop hunspell hwinfo imagemagick inxi
iproute2 kbd less libcamera libssh libssh2 libvncserver libvoikko libwireplumber lsd
lsof man-db man-pages meson ninja nmap nodejs npm nss-mdns ntfs-3g nuspell p7zip parted
pipewire pipewire-libcamera pipewire-pulse powertop pwgen python-pyqt5 python-pyqt6
qt5-wayland qt6-wayland ripgrep rpcbind smartmontools speech-dispatcher speedtest-cli
tecla tlp tree unzip upower v4l-utils v4l2loopback-utils wget wireless_tools
wireplumber zip zram-generator zsh zsh-autocomplete zsh-autosuggestions zsh-completions
zsh-history-substring-search zsh-lovers zsh-syntax-highlighting
```

```
sudo pacman -S gnu-free-fonts powerline-fonts terminus-font ttf-anonymous-pro ttf-
atkinson-hyperlegible ttf-bitstream-vera ttf-caladea ttf-carlito ttf-cascadia-code ttf-
crimson-pro ttf-crimson-pro-variable ttf-croscore ttf-dejavu ttf-doulos-sil ttf-droid
ttf-eurof ttf-fantasque-sans-mono ttf-fira-code ttf-fira-mono ttf-fira-sans ttf-hack
ttf-ibm-plex ttf-inconsolata ttf-input ttf-jetbrains-mono ttf-jetbrains-mono-nerd ttf-
junicode ttf-junicode-variable ttf-khmer ttf-lato ttf-liberation ttf-libertinus ttf-
linux-libertine ttf-linux-libertine-g ttf-material-icons ttf-material-symbols-variable
ttf-meslo-nerd ttf-mona-sans ttf-monospace-frozen ttf-monospace-variable ttf-monofur
ttf-monoid ttf-montserrat ttf-nunito ttf-opensans ttf-overpass ttf-roboto ttf-roboto-
mono ttf-terminus-nerd ttf-ubuntu-font-family
```

```
sudo wget -O /usr/local/bin/yt-dlp
https://github.com/yt-dlp/yt-dlp/releases/latest/download/yt-dlp && sudo chmod a+rx
/usr/local/bin/yt-dlp
```

```
sudo systemctl enable --now avahi-daemon.service
sudo systemctl enable --now bluetooth.service
sudo systemctl enable --now cronie.service
sudo systemctl enable --now cups.service
sudo systemctl enable --now cups-browsed.service
sudo systemctl enable --now firewalld.service
sudo systemctl enable --now fstrim.timer
sudo systemctl enable --now tlp.service
```

```
sudo systemctl mask systemd-rfkill.service systemd-rfkill.socket ( for tlp.service )
```

1. setup gnome-control-center
2. setup gnome-tweaker
3. setup extension-manager
4. setup fstab: sudo nano /etc/fstab
5. setup zram-generator
6. setup zsh: chsh -s /bin/zsh \$USER

reboot

* * *