

# Průvodce instalací Arch Linuxu

Průvodce instalací Arch Linuxu se šifrováním celého disku pomocí BTRFS na LUKS a šifrovaným spouštěcím oddílem (GRUB) pro systémy UEFI.

- [Úvod](#)
- [Připojení k internetu](#)
- [Příprava disku](#)
- [Instalace Arch Base, lokalizace,](#)
- [Konfigurace sítě, swap](#)
- [Initramfs](#)
- [Grafické ovladače a Xorg](#)
- [Heslo uživatele root, vytvoření uživatele](#)
- [Zavaděč \(bootloader\)](#)

# Úvod

Tato příručka obsahuje pokyny pro instalaci Arch Linuxu se šifrováním celého disku pomocí BTRFS na LUKS a šifrovaným spouštěcím oddílem (GRUB) pro systémy UEFI.

Po hlavní instalaci následují další pokyny, jak se bránit útokům Evil Maid prostřednictvím vlastní registrace klíče UEFI Secure Boot, jádra a bootloADERu s vlastním podpisem

## Předmluva

Většinu těchto informací najdete na [Arch Wiki](#) a dalších zdrojů na ně odkazovaných v dolní části článku.

*Poznámka:* Návod popisuje instalaci na moderní NVMe SSD disk. Pokud používáte jiný typ disků, proveďte všude v návodu substituci `/dev/nvme0nX` na `/dev/sdX`, popř. jiný typ zařízení podle potřeby. Video návod používá `/dev/sdX`, který je standardním nastavením ve Virtualboxu.

# Připojení k internetu

## Připojení k internetu

Zapojte ethernet a pokračujte dál, nebo pro bezdrátové připojení se podívejte na vševědoucí [Arch Wiki](#).

## Nastavení Wifi

```
iwctl
[iwd]# device list -> note your device
[iwd]# station <device> get-networks
[iwd]# station <device> connect <SSID>

# second variant
iwctl --passphrase <heslo> station <device> connect <SSID>

# test connection
ping archlinux.org
```

## Aktualizujte systémové hodiny

```
timedatectl set-ntp true
```

## Aktualizujte mirrorlist

```
pacman -Syy
pacman -S reflector
cp /etc/pacman.d/mirrorlist /etc/pacman.d/mirrorlist.bak

reflector --country 'Czechia' --age 24 --verbose --sort rate --save /etc/pacman.d/mirrorlist
# OR
reflector -c "CZ" -f 12 -l 10 -n 12 --save /etc/pacman.d/mirrorlist
```

## Nastavení rozlišení Virtualbox tty

```
pacman -S fbset terminus-font
fbset -g 2048 1080 2048 1080 32
setfont ter-132n
```

# Příprava disku

Aktualizujte btrfs-progs

```
pacman -Syy btrfs-progs
```

Zobrazení disků a oddílů

```
lsblk
```

Vytvořte oddíly EFI System a Linux LUKS

Number	Start (sector	End (sector)	Size	Code	Name
1	2048	1130495	256.0 MiB	EF00	EFI System
2	1130496	976773134	465.2 GiB	8309	Linux LUKS

```
gdisk /dev/nvme0n1

# alternative apps
cfdisk
fdisk
```

```
o
n
[Enter]
0
+256M
ef00
n
[Enter]
[Enter]
```

[Enter]

8309

w

Vytvořte šifrovaný kontejner LUKS1 na oddílu Linux LUKS (GRUB nepodporuje LUKS2 od května 2019)

```
cryptsetup luksFormat --type luks1 --use-random -S 1 -s 512 -h sha512 -i 1000 /dev/nvme0n1p2
```

Otevřete kontejner (dešifrujte jej a zpřístupněte na /dev/mapper/cryptbtrfs)

```
cryptsetup open /dev/nvme0n1p2 cryptbtrfs
```

## Příprava svazku/podsvazků BTRFS

Formátování btrfs, nyní již dešifrovaného disku

```
mkfs.btrfs -L "Arch Linux" /dev/mapper/cryptbtrfs
```

Připojit souborový systém (zatím bezparametrické)

```
mount /dev/mapper/cryptbtrfs /mnt
```

## Vytvoření podsvazků (subvolumes)

Toto schéma lze upravit podle vašich potřeb, navrhol bych alespoň jeden podsvazek pro root (@) a jeden pro snímky (.@snapshots). varlog a tmp jsou vytvořeny pro snadné zakázání kopírování při zápisu na /var/log a /tmp. pkg subvolume naopak pomáhá v organizaci pacman balíčků s možností jejich zálohy a vrácení se k předchozím verzím. docker subvolume umožňuje efektivní práci s dockrem, který má implementovanou efektivní spolupráci s btrfs souborovým systémem.

```
btrfs sub cr /mnt/@  
btrfs sub cr /mnt/@home  
btrfs sub cr /mnt/@tmp  
btrfs sub cr /mnt/@log  
btrfs sub cr /mnt/@pkg  
btrfs sub cr /mnt/@docker  
btrfs sub cr /mnt/.@snapshots
```

## Zakázat kopírování při zápisu do /var/log a /tmp

```
chattr +C /mnt/@log  
chattr +C /mnt/@tmp  
umount /mnt
```

## Připojení subvolumes BTRFS, nyní již s kompletní parametrizací pro stálé používání

```
mount -o defaults,noatime,discard,ssd,subvol=@ /dev/mapper/cryptbtrfs /mnt  
mkdir -p /mnt/{home,var/log,var/cache/pacman/pkg,var/lib/docker,tmp,.@snapshots}
```

```
# Discard and ssd options and are for ssd disks only
mount -o defaults,noatime,discard,ssd,subvol=@home /dev/mapper/cryptbtrfs /mnt/home
mount -o defaults,noatime,discard,ssd,subvol=@tmp /dev/mapper/cryptbtrfs /mnt/tmp
mount -o defaults,noatime,discard,ssd,subvol=@log /dev/mapper/cryptbtrfs /mnt/var/log
mount -o defaults,noatime,discard,ssd,subvol=@pkg /dev/mapper/cryptbtrfs /mnt/var/cache/pacman/pkg/
mount -o defaults,noatime,discard,ssd,subvol=@docker /dev/mapper/cryptbtrfs /mnt/var/lib/docker
mount -o defaults,noatime,discard,ssd,subvol=.@snapshots /dev/mapper/cryptbtrfs /mnt/.snapshots
```

## Příprava oddílu EFI

Vytvořte souborový systém FAT32 na systémovém oddílu EFI

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

Vytvořte bod připojení pro systémový oddíl EFI na /efi pro kompatibilitu s grub-install a připojte jej

```
mkdir /mnt/efi
mount /dev/nvme0n1p1 /mnt/efi
```



# Instalace Arch Base, lokalizace,

Nainstalujte potřebné balíčky

```
pacstrap /mnt/ base base-devel linux linux-headers linux-firmware polkit git btrfs-progs efibootmgr dhcpcd bash-cd
```

## Nakonfigurujte systém

Vygenerujte soubor /etc/fstab

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

Zkontrolujte, zda `fstab` je správně vytvořen s definovanými parametry btrfs z předchozí kapitoly s editorem neovim.

Vstupne do svého nového systému pomocí arch-chroot

```
arch-chroot /mnt
```

V tomto okamžiku byste měli mít následující oddíly a logické svazky:

```
lsblk
```

NAME	MAJ:MIN	RM	SIZE	RO	TYPE	MOUNTPOINT
nvme0n1	259:0	0	465.8G	0	disk	
└─nvme0n1p1	259:5	0	256M	0	part	/efi
└─nvme0n1p2	259:6	0	465.2G	0	part	/.snapshots
└─cryptbtrfs	254:0	0	465.2G	0	crypt	/var/lib/docker
						/var/cache/pacman/pkg
						/var/log
						/tmp
						/home
						/

## Časové pásmo

Nahradit `Europe/Prague` s vaším příslušným časovým pásmem nalezeným v `/usr/share/zoneinfo`

```
In -sf /usr/share/zoneinfo/Europe/Prague /etc/localtime
```

Běh `hwclock` vygeneruje `/etc/adjtime`

Hardwarové hodiny budou nastaveny na UTC a synchronizovány s aktuálním systémovým časem.

```
hwclock --systohc --utc
```

## Lokalizace

Odkomentovat řádku `en_US.UTF-8 UTF-8` v `/etc/locale.gen` a vygenerovat národní prostředí.

locale-gen

Vytvořit `locale.conf` a nastavte `LANG` proměnnou prostředí

```
echo LANG=en_US.UTF-8 > /etc/locale.conf  
export LANG=en_US.UTF-8
```

# Konfigurace sítě, swap

Vytvořte soubor `/etc/hostname` s názvem počítače

```
echo myhostname > /etc/hostname
```

Toto je jedinečný název pro identifikaci vašeho zařízení v síti.

Přidejte odpovídající záznamy do `/etc/hosts`

```
nvim /etc/hosts
```

```
127.0.0.1 localhost
::1 localhost
127.0.1.1 myhostname
```

## (Volitelné) Vytváření swap souboru

Nyní vytvořte prázdný (s velikostí 0) odkládací soubor. Vytvořte samostatný podsvazek pro odkládací soubor. Tento podsvazek je potřebný k tomu, abyste mohli vytvořit snímek `/`, což by nebylo možné s žádným souborem v něm s vypnutým CoW!

```
bftrfs su create /swap
chattr +C /swap
```

```
# Copy on Write should always be disabled on swap file, so it will be done in the next step
touch /swap/swapfile
```

# Check if C attribute is enabled (should be already if created in folder with disabled CoW attribute)

```
lsattr /swap/swapfile
```

# If not then disable CoW for swapfile manually

```
chattr +C /swap/swapfile
```

# Expanding empty file to 4GiB swap file

```
dd if=/dev/zero of=/swap/swapfile bs=1024K count=4096
```

```
chmod 600 /swap/swapfile
```

# Format the swap file.

```
mkswap /swap/swapfile
```

# Turn swap file on.

```
swapon /swap/swapfile
```

# You also need to update /etc/fstab to mount swapfile on boot

```
/swap/swapfile none swap sw 0 0
```

# Initramfs

Přidat `encrypt`, a `btrfs` háčky (hooks) na `/etc/mkinitcpio.conf`

*Poznámka:* Záleží na pořadí!

```
HOOKS=(base udev autodetect modconf kms keyboard keymap consolefont block encrypt btrfs filesystems fsck)

# Add btrfsck to binaries
BINARIES=(btrfsck)
```

## Znovu vytvořte obraz initramfs

```
mkinitcpio -P
```

## Instalace základního softwaru

```
pacman -S openssh networkmanager wpa_supplicant netctl
systemctl enable NetworkManager
systemctl enable sshd

pacman -S amd-ucode (for AMD), pacman -S intel-ucode (for INTEL)
mkinitcpio -p linux
```

# Grafické ovladače a Xorg

## Karty AMD

```
pacman -S xorg xorg-xinit  
pacman -S mesa
```

## Karty Nvidia

```
pacman -S xorg xorg-xinit  
pacman -S nvidia nvidia-utils  
  
sudo vim /etc/mkinitcpio.conf  
# edit Modules and Files  
MODULES=(nvidia nvidia_modeset nvidia_uvm nvidia_drm ...)  
FILES="/etc/modprobe.d/nvidia.conf"  
  
sudo mkinitcpio -P  
nvim /etc/modprobe.d/nvidia.conf  
# Add row to the file  
options nvidia_drm modeset=1
```

## Podpora Virtualboxu

```
pacman -S virtualbox-guest-utils
```

# Heslo uživatele root, vytvoření uživatele

Nastavte heslo uživatele root

## Přidat uživatele Linuxu

```
useradd -m -g users -G wheel,storage,power -s /bin/bash <user>  
passwd <user>  
pacman -S sudo  
EDITOR=nvim visudo, (uncomment) %wheel ALL=(ALL:ALL) ALL
```



# Zavaděč (bootloader)

## Nainstalujte GRUB

```
pacman -S grub efibootmgr os-prober dosfstools mtools
```

Nakonfigurujte GRUB tak, aby umožňoval spouštění z /boot na šifrovaném oddílu LUKS1

```
nvim /etc/default/grub
```

```
GRUB_ENABLE_CRYPTODISK=y
```

Nastavte parametr jádra pro odemknutí fyzického svazku BTRFS při spouštění pomocí `encrypt` hook

UUID je oddíl obsahující kontejner LUKS

```
blkid
```

```
/dev/nvme0n1p2: UUID="xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" TYPE="crypto_LUKS" PARTLABEL="Linux LUKS"
```

```
/etc/default/grub
```

```
# allow-discards is only for ssd to let trim work with encryption enabled
```

```
GRUB_CMDLINE_LINUX="... cryptdevice=UUID=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx:cryptbtrfs:allow-discards"
```

# Nainstalujte GRUB do připojeného ESP pro spouštění UEFI

```
grub-install --target=x86_64-efi --bootloader-id=ARCH --efi-directory=/efi --recheck
```

## Vygenerujte konfigurační soubor GRUB

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

## (doporučeno) Vložte soubor s klíčem do initramfs

To se provádí proto, abyste nemuseli zadávat heslo pro dešifrování dvakrát (jednou pro GRUB, jednou pro initramfs.)

## Vytvořte soubor s klíčem a přidejte jej jako klíč LUKS

```
mkdir /root/secrets && chmod 700 /root/secrets  
head -c 64 /dev/urandom > /root/secrets/crypto_keyfile.bin && chmod 600 /root/secrets/crypto_keyfile.bin  
cryptsetup -v luksAddKey -i 1 /dev/nvme0n1p2 /root/secrets/crypto_keyfile.bin
```

## Přidejte soubor s klíčem do obrazu initramfs

```
nvim /etc/mkinitcpio.conf
```

```
FILES=(/root/secrets/crypto_keyfile.bin)
```

## Znovu vytvořte obraz initramfs

```
mkinitcpio -P
```

Nastavte parametry jádra pro odemknutí oddílu LUKS pomocí souboru s klíčem v `encrypt` hook

```
/etc/default/grub
```

```
GRUB_CMDLINE_LINUX="... cryptkey=rootfs:/root/secrets/crypto_keyfile.bin"
```

## Znovu finálně vygenerujte konfigurační soubor GRUB

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

## Omezit `/boot` oprávnění

```
chmod 700 /boot
```

Instalace je nyní dokončena. Ukončete chroot a restartujte počítač.

```
exit  
reboot
```