Arch Linux průvodce instalací základního šifrování celého disku

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

- Průvodce instalací Arch Linuxu
  - Úvod
  - Přípojení k internetu
  - Příprava disku
  - Instalace Arch Base, lokalizace,
  - Konfigurace sítě, swap
  - Initramfs
  - Grafické ovladače a Xorg
  - Heslo uživatele root, vytvoření uživatele
  - Zavaděč (bootloader)
- Po instalaci (post-instalační kroky)
- Kompletní video návod

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

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.

Přípojení 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, navrhoval 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 subolumes 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 efiboot
```

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`

```
ln -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!

```
btrfs 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 pořádí!

```
HOOKS=(base udev autodetect modconf kms keyboard keymap consolefont block encrypt btrfs filesyst

# 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

```
passwd
```

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
```

```
/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-d
```

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_key
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
```

Po instalaci (post-instalační kroky)

Po instalaci (post-instalační kroky)

Váš systém by nyní měl být plně nainstalován, spustitelný a plně zašifrovaný.

Pokud jste soubor s klíčem vložili do obrazu initramfs, měl by k odemknutí do systému počítač vyžadovat vaše heslo pro dešifrování pouze jednou **před GRUB tabulkou**. Narozdíl od archinstall skriptu, toto je daleko více bezpečnostní zabezpečení.

Pro další standardní kroky po instalaci Arch Linuxu RTFM .

Zrychlení dešifrování LUKS v GRUB po zadání hesla

Upozornění: Před pokračováním v této části se ujistěte, že rozumíte důležitosti hesel s vysokou entropií. Informace o tom, jak generovat bezpečná hesla, naleznete na Wikipedii: Síla hesla.

Při bootování může GRUB v některých případech trvat dlouho, než bude ověřeno heslo. To může být způsobeno vysokým počtem iterací PBKDF, který můžete zkontrolovat následovně:

```
cryptsetup luksDump /dev/nvme0n1p2
```

Problém je v tom, že iterace byly vypočteny pro daný `keyslot` při normálním běhu počítače (nikoliv při bootu), když byl klíč přidán, aby byla zajištěna rovnováha mezi dostatečně vysokou úrovní pro

ochranu před útoky hrubou silou a dostatečně nízkou, aby umožnila rychlé odvození klíče pomocí odhadu schopností vašeho počítače. Když se však GRUB spustí, **nemusí mít** k dispozici stejné výpočetní zdroje, a proto je **výrazně pomalejší** .

Pokud vaše **heslo poskytuje dostatečnou entropii,** aby samo čelilo běžným útokům, můžete toto číslo snížit:

```
cryptsetup luksChangeKey --key-slot 1 --pbkdf-force-iterations 1000 /dev/nvme0n1p2
```

Podle RFC 2898 se doporučuje minimálně 1000 iterací, ale pokud je to možné, měli byste se zaměřit na vyšší hodnoty (náklady pro útočníka a také čas na odvození klíče lineárně škálují).

Tip: GRUB zkouší aktivované klíčové sloty postupně. Při přidávání klíčů se `--key-slot` lze použít k explicitnímu zadání klíčového slotu.

(doporučeno) Secure Boot – Odolávání proti útokům Evil Maid

Se zašifrovaným zaváděcím oddílem nikdo nemůže vidět ani upravit váš obraz jádra nebo initramfs, ale stále byste byli zranitelní vůči útokům Evil Maid .

Jedním z možných řešení je použití UEFI Secure Boot. Zbavte se předinstalovaných klíčů Secure Boot (opravdu nechcete věřit Microsoftu a OEM), zaregistrujte si své vlastní klíče Secure Boot a podepište zavaděč GRUB svými klíči. Evil Maid by nemohla zavést upravený zavaděč (nepodepsaný vašimi klíči) a útoku je zabráněno.

Vytváření klíčů

Následující kroky by měly být provedeny pod `root` uživatelem s doprovodnými soubory uloženými v `/root` adresáři.

Nainstalujte `efitools`

```
pacman -S efitools
```

Vytvořte GUID pro identifikaci vlastníka

```
uuidgen --random > GUID.txt
```

Platformový klíč

CN je běžné jméno, které lze zapsat jako cokoliv.

```
openssl req -newkey rsa:4096 -nodes -keyout PK.key -new -x509 -sha256 -days 3650 -subj "/CN=my P
openssl x509 -outform DER -in PK.crt -out PK.cer
cert-to-efi-sig-list -g "$(< GUID.txt)" PK.crt PK.esl
sign-efi-sig-list -g "$(< GUID.txt)" -k PK.key -c PK.crt PK PK.esl PK.auth
```

Podepište prázdný soubor, abyste umožnili odebrání klíče platformy v „uživatelském režimu“

```
sign-efi-sig-list -g "$(< GUID.txt)" -c PK.crt -k PK.key PK /dev/null rm_PK.auth
```

Klíč pro výměnu klíčů

```
openssl req -newkey rsa:4096 -nodes -keyout KEK.key -new -x509 -sha256 -days 3650 -subj "/CN=my
openssl x509 -outform DER -in KEK.crt -out KEK.cer
cert-to-efi-sig-list -g "$(< GUID.txt)" KEK.crt KEK.esl
sign-efi-sig-list -g "$(< GUID.txt)" -k PK.key -c PK.crt KEK KEK.esl KEK.auth
```

Podpisový klíč databáze

```
openssl req -newkey rsa:4096 -nodes -keyout db.key -new -x509 -sha256 -days 3650 -subj "/CN=my S
openssl x509 -outform DER -in db.crt -out db.cer
cert-to-efi-sig-list -g "$(< GUID.txt)" db.crt db.esl
sign-efi-sig-list -g "$(< GUID.txt)" -k KEK.key -c KEK.crt db db.esl db.auth
```

Podepisování bootloaderu a jádra

Když je Secure Boot aktivní (tj. v „Uživatelském režimu“), budete moci spouštět pouze podepsané binární soubory, takže musíte podepsat své jádro a zavaděč.

Nainstalujte `sbsigntools`

```
pacman -S sbsigntools
```

```
sbsign --key db.key --cert db.crt --output /boot/vmlinuz-linux /boot/vmlinuz-linux
sbsign --key db.key --cert db.crt --output /efi/EFI/arch/grubx64.efi /efi/EFI/arch/grubx64.efi
```

Automaticky podepsat bootloader a jádro při instalaci a aktualizacích s pacman hookem

Je nutné podepsat GRUB pomocí klíčů UEFI Secure Boot pokaždé, když je systém aktualizován prostřednictvím `pacman`. Toho lze dosáhnout pomocí pacman hook .

Vytvořte adresář `hooks`

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

Vytvořte hooks pro oba `linux` a `grub` balíčky

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

```
[Trigger]
Operation = Install
Operation = Upgrade
Type = Package
Target = linux

[Action]
Description = Signing Kernel for SecureBoot
When = PostTransaction
Exec = /usr/bin/find /boot/ -maxdepth 1 -name 'vmlinuz-*' -exec /usr/bin/sh -c 'if ! /usr/bin/sb
Depends = sbsigntools
Depends = findutils
Depends = grep
```

```
/etc/pacman.d/hooks/98-secureboot-grub.hook
```

```
[Trigger]
Operation = Install
Operation = Upgrade
```

```
Type = Package
Target = grub

[Action]
Description = Signing GRUB for SecureBoot
When = PostTransaction
Exec = /usr/bin/find /efi/ -name 'grubx64*' -exec /usr/bin/sh -c 'if ! /usr/bin/sbverify --list
Depends = sbsigntools
Depends = findutils
Depends = grep
```

Zaregistrujte klíče do firmwaru

Zkopírujte vše `*.cer`, `*.esl`, `*.auth` do systémového oddílu EFI

```
cp /root/*.cer /root/*.esl /root/*.auth /efi/
```

Spusťte nástroj pro nastavení firmwaru UEFI (často nesprávně označovaný jako „BIOS“)

```
systemctl reboot --firmware
```

Firmware má různá rozhraní, viz Výměna klíčů pomocí nástroje pro nastavení firmwaru, pokud jsou následující pokyny nejasné nebo neúspěšné.

Nastavte Typ OS na `Windows UEFI mode`

Najděte možnosti Secure Boot a nastavte typ OS na `Windows UEFI mode` (Ano, i když nejsme na Windows.) To může být nezbytné pro funkci Secure Boot.

Vymažte předinstalované klíče Secure Boot

Pomocí Key Management vymažte všechny předinstalované klíče Secure Boot (Microsoft a OEM).

Vymazáním všech klíčů Secure Boot vstoupíte do režimu nastavení (takže si můžete zaregistrovat své vlastní klíče Secure Boot).

Nastavte nebo připojte nové klíče

Klíče musí být nastaveny v následujícím pořadí:

```
db => KEK => PK
```

To je způsobeno tím, že některé systémy ukončí režim nastavení, jakmile `PK` je zadáno.

Nenahrávejte výchozí tovární nastavení, místo toho procházejte dostupné souborové systémy při hledání souborů dříve zkopírovaných do systémového oddílu EFI.

Vyberte některý z formátů. Firmware by vás měl vyzvat k zadání typu (*Poznámka:* názvy typů se mohou mírně lišit.)

```
*.cer is a Public Key Certificate
*.esl is a UEFI Secure Variable
*.auth is an Authenticated Variable
```

Určitý firmware (například můj vlastní) vyžaduje použití souborů *.auth. Vyzkoušejte různé, dokud nebudou fungovat.

Nastavte heslo správce (administrátora) UEFI

V nastavení zabezpečení musíte také nastavit heslo správce (administrátora) firmwaru UEFI, takže nikdo nemůže jednoduše spustit nástroj pro nastavení UEFI a vypnout zabezpečené spouštění.

Nikdy byste neměli používat stejné heslo správce firmwaru UEFI jako heslo pro šifrování, protože na některých starých přenosných počítačích lze heslo správce obnovit jako prostý text z čipu EEPROM.

Ukončete a uložte změny

Jakmile načtete všechny tři klíče a nastavíte heslo správce, stiskněte F10 pro ukončení a uložení změn.

Pokud bylo vše provedeno správně, měl by se při restartu objevit váš zavaděč.

Zkontrolujte, zda byl povolen Secure Boot

```
od -An -t u1 /sys/firmware/efi/efivars/SecureBoot-XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
```

Znaky označené XXXX se liší stroj od stroje. Chcete-li s tím pomoci, můžete použít tab doplňování nebo seznam proměnných EFI.

Pokud je povoleno Secure Boot, tento příkaz vrátí 1 jako poslední celé číslo v seznamu pěti hodnot, například:

```
6  0  0  0  1
```

Pokud bylo aktivováno Secure Boot a bylo nastaveno heslo správce UEFI, můžete se nyní považovat za chráněného před útokům Evil Maid!

Docker používaný s BTRFS souborovým systémem

Seznámení se s dockerem na Arch Linuxu na wiki Docker on Arch.

BTRFS je idální pro správu docker snapshotů. Pro nastavení postupujte podle originálního článku.

Celé prostředí máme pro dockerizaci již nachystané. Mělo by stačit nastavit pouze v `/etc/docker/daemon.json`

```
{
  "storage-driver": "btrfs"
}
```

Kompletní video návod

Kompletní video návod