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"
}
```

Revision #1
Created 2023-01-01 15:49:22 UTC by archos
Updated 2023-01-01 15:51:24 UTC by archos