Maturitní práce
Script pro instalaci operačního systému Gentoo

Studijní obor: 18-20-M/01
Informační technologie

Autor: Halamka David
Třída: IT4/A
Vedoucí práce: Petr Martínek

Prohlašuji, že jsem předkládanou práci vypracoval sám za použití zdrojů a literatury v ní uvedených. Souhlasím s tím, aby moje maturitní práce byla využívána pro potřeby Střední školy spojů a informatiky, Tábor, Bydlinského 2474 nebo jiných subjektů, které se podílely na zadání práce.

……. ……..
datum podpis

Obsah

1 Anotace

Teoretická část maturitní práce popisuje proces instalace operačního systému Gentoo a její vliv na tvorbu scriptu. V práci je také zároveň rozvedena problematika stěžejních kroků instalace a částí operačního systému. Cílem práce bylo usnadnit novým uživatelům instalaci a konfiguraci systému pro jednodušší přechod z jiných operačních systémů. Z toho důvodu je script volně dostupný na webových stránkách Github.

Klíčová slova:
Keywords:

2 Příprava disků

Příprava disků je nutným krokem každé instalace. Hlavním důvodem je, že stávající schéma rozdělení a souborové systémy nemusí být v souladu s novým operačním systémem. Z toho důvodu nestačí disky pouze smazat. Použité schéma rozdělení je GPT. Mezi hlavní důvody pro výběr GPT patří: lepší kompatibilita s UEFI systémy, vyšší maximální počet oddílů, podpora disků větších než 2TB a kontrola integrity. Oddíly a jejích nastavení mají velký vliv na paměť i výkon systému. Proto je potřeba, aby script i samotná instalace přizpůsobily rozdělení specifikacím systému.

První je EFI oddíl. Slouží pro zavádění UEFI systémů. Bude obsahovat Bootloader. Velikost je pevně nastavena na 1GB, jelikož není potřeba škálovat se systémem. Oddíl bude naformátován jako fat32.

Druhý je Swap oddíl. Swap obsahuje časti operační paměti, vyhodnocené jako nejméně používané. Rozbor samotného algoritmu pro správu paměti je mimo záběr maturitní práce. Je třeba, aby velikost Swap oddílu rostla s kapacitou paměti RAM. Tento vztah je způsoben vlastnostmi hibernace, která nahraje celý obsah operační paměti na Swap oddíl. Má-li tedy systém využívat hibernaci, je nutné aby velikost oddílu byla ostře větší, než velikost paměti RAM.

Třetí je Root oddíl. Obsahuje komplement EFI oddílu. Je to zároveň také jediný oddíl z trojice, u kterého lze účině zvolit souborový systém. Zabírá zbytek místa na disku.

Script vytváří oddíly pomocí nástroje `fdisk`. `fdisk` je interaktivní nástroj na tvorbu oddílu, proto je nutné aby script dodával instrukce přímo do vstupu nástroje, ne shellu. Přesměrování provádí Here document, který dokáže přesměrovat dávkovaný vstup přímo do programu.
Výběr disku:

```
   select diskname in /dev/*; do
if [[ -n $diskname ]]; then
    echo "Zvolen: $diskname"
    break
else
    echo "Vyberte zařízení"
fi
done
```

`select` umožňuje vybrat soubor ze složky `/dev`, kde se nacházejí bloková zařízení.

Pro škálování Swapu je potřeba zjistit informace o velikosti paměti RAM. Ty jsou uložené v souboru `/proc/meminfo`. Pro `fdisk` je potřeba, aby jsme dostali pouze numerický údaj. Přečtení a zpracování tedy vypadá následovně:

```
swapsize=$(sudo cat /proc/meminfo | head -n 1 | sed 's/[^0-9]//g')
```

Výsledné číslo pak zvětšíme o 50 %:

```
swapsize=$((($swapsize/1000000)*3/2))
```

2.1 Souborový systém

Manuální instalace i script dávají uživateli možnost vyběru souborového systému pro kořenový oddíl. Souborový systém slouží k interpretaci dat ve formě souborů, adresářů a metadat. Dělí se na dvě hlavní skupiny: Žurnálovací a Copy on Write.
Žurnálovací systémy uchovávají záznam prováděných operací ve speciální datové struktuře v rámci jejich oddílu. Záznam slouží k obnově konzistence souborového systému v případě přerušení zápisu, způsobeným například odpojením zdrojem energie.
Copy on Write systémy nevytváří kopii souboru pro úpravu, pouze předají odkaz na již existující soubor a to klidně více procesům najednou. Kopii dat vytvoří až v případě, že se nějaký z procesů pokusí přepsat data. Tento přístup umožňuje integraci snapshotů a RAIDů. Script volí mezi třemi souborovými systémy: XFS, btrfs a ext4.
Je vhodné zmínit, že formátování odstraní veškerá data z oddílu. Formátování oddílů pro souborový systém ve scriptu provádí příkaz mkfs:

```
mkfs.vfat -F 32 /dev/sda1
mkfs.$filesystem /dev/sda3
```

Speciální případ je Swap, který je potřeba prvně inicializovat:

```
mkswap /dev/sda2
```

A poté aktivovat:

```
swapon /dev/sda2
```

Nyní stačí namontovat oddíly na jejich adresáře:

```
mount /dev/sda3 /mnt/gentoo
mount /dev/sda1 /mnt/gentoo/efi
```

3 Stage Tarball

Pro instalaci základního systému je třeba tzv. Stage 3 Tarball, neboli archiv obsahující minimální systém pro začátek instalace. Archivů je mnoho a dělí se podle profilů.

Každý profil blíže specifikuje budoucí záměr a funkce systému. Například profil 'Desktop' obsahuje balíčky, respektive jejich USE přepínače, usnadňující zprovoznění uživatelského rozhraní. Profilů existuje velké množství, čehož důsledkem je více funkčních kombinací. Z tohoto důvodu script umožňuje zvolit archiv jak automaticky, tak i manuálně pomocí URL adresy.

Pro výchozí výběr byl zvolen archiv z FTP serveru Fakulty informatiky Masarykovy Univerzity v Brně. Script řeší výběr archivu pomocí jednoduché logiky, která upřednostní manuální výběr, je-li dostupný:

```
   read -p stage3 2>&1
if [ -z "$stage3" ]; then
   wget https://ftp.fi.muni.cz/pub/linux/gentoo/...
else
   wget $stage3
fi
```

Nesprávně zvolený profil lze po dokončení instalace změnit pomocí:

```
eselect profile list
eselect profile set #
```

Uživatel by však měl mít na paměti, že změna profilu může, v některých případech vést až k překompilování celého systému.

V tuto chvíli se doporučuje, aby uživatel provedl kontrolu archivu pomocí checksumů. Ty lze získat v souboru .DIGESTS z jakéhokoliv zrcadla. Pro druhý, vlastní checksum je doporučený nástroj sha256sum, který vytváří checksum pomocí algoritmu sha256. Použití:

```
sha256sum nazevSouboru
```

Z bezpečnostních důvodů není doporučeno starší algoritmy, například: md5, sha1...

Script, z důvodu minimalizace rizik a skrytých paradoxů, tuto kontrolu neprovádí.

3.1 Instalace archivu

Před instalací je vhodné nastavit systémový čas. K tomu slouží chrony:
`chrony -q`
anebo, není-li dostupné RTC, pomocí:
`date mmddhhmmyy`.

V případě jakýchkoliv pochybností je možné datum zkontrolovat pomocí `date`. Archiv je potřeba extrahovat do cílového adresáře nového systému. Proto je nutné, aby byl adresář již předem vytvořený. Ve velké většině případů `/mnt/gentoo`. Tarball instalujeme pomocí nástroje tar v příkazu:

```
tar xpvf stage3-*.tar.xz --xattrs-include='*.*' --numeric-owner
-C /mnt/gentoo
```

Pro lepší orientaci bude význam přepínačů a argumentů rozepsán:

`x` - soubor bude extrahován.
`p` - ponechat oprávnění. *
`v` - zapíná výpis průběhu extrakce, volitelný.
`f` - určuje jaký soubor má být extrahován a cestu k němu.
`--xattrs-include='*.*'` - zachová rozšiřující atributy. (např.:SELinux štítky)
`--numeric-owner` - zachová vlastnictví podle číselných ID uživatelů a skupin.
`-C /mnt/gentoo` - extrahuje obsah do uvedeného adresáře.

*Ačkoliv tar již nemění oprávnění při extrakci, může dojít ke ztrátě speciálních oprávnění, například: setuid, setgid

3.2 Kompilace

Součástí základního systému je Portage. Portage je nativní package manager pro Gentoo. Obstarává kompilace, instalace, závislosti, synchronizace a mnoho dalšího. V této fázi instalace je doporučené optimalizovat kompilace. Veškerá globální konfigurace je součástí /etc/portage/make.conf. Zároveň obsahuje přepínače pro překladače GCC, C++ a Rust. Jejich správnému nastavení bude věnována následující podkapitola.

3.2.1 Flags

Pro nastavení GCC slouží proměnná CFlags ve tvaru `CFLAGS="flag1 flag2..."`. Nastavení C++ je totožné, mění se pouze proměná a to na `'CXXFLAGS'`. Uvádíme-li, že nastavení jsou totožná, je to myšleno doslovně. Je doporučeno, aby přepínače u překladačů byli při instalaci stejné. Níže je uvedena vzorová konfigurace:

```
   CFLAGS="-march=native -O2 -pipe"
CXXFLAGS="-march=native -O2 -pipe"
```

`-march=native` - použije nastavení a instrukční sady pro daný procesor
`-O2` - překladač se pokusí kompilovat rychleji na úkor paměti.
`-pipe` - zrychluje kompilaci díky 'rourám', ty překladač vytvoří místo dočasných souborů.

Proměnná pro Rust je `RUSTFLAGS`.

```
RUSTFLAGS="$-C target-cpu=native opt-level=2"
```

`target-cpu=native` - přizpůsobí kompilaci pro instalovaný procesor.
`opt-level=2` - optimalizuje strojový kód, podobné -O2

3.3 MAKE Operace

Chceme-li, aby systém zůstal responzivní i během kompilace, je potřeba omezit maximální zatížení systému. Přepínače make omezující zátěž se zadávají do proměnné `MAKEOPTS`.

```
MAKEOPTS="-j5 -l5"
```

`-j` - určuje maximální počet paralelních procesů.
`-l` - nastavuje maximální vytížení procesoru.

Důležitým faktorem pro volbu parametrů je, že každý proces zatěžuje mimo procesoru také paměť.

4 Nový systém

Všechny následující kroky se již budou provádět v novém systému. Pro vstup do nového systému, jelikož se nejedná o virtualizaci, je nutné zajistit přístup k fyzickým prostředkům. Toho docílíme přikazem mount adresářů `/sys`, `/proc`, `/dev` a `/run` na jejich `/mnt/gentoo` protějšky. Chceme-li v novém systému stejné DNS servery, zkopírujeme ještě jejich záznam:
`cp /etc/resolv.conf /mnt/gentoo/etc`
Pokud se jedná o symlink je potřeba přidat přepínač `--dereference`,
abychom zkopírovali soubor, ne odkaz na něj.

4.1 Chroot

Do nového prostředí se dostaneme pomocí nástroje chroot ve tvaru:

```
chroot /mnt/gentoo chroot.sh
```

`/mnt/gentoo` - adresář s novým kořenem
`/bin/bash` - program/příkaz, který chceme v novém prostředí spustit

Zároveň je nyní vhodné namontovat EFI oddíl na `/efi` adresář.

4.2 Portage

Oficiální package manager pro Gentoo je Portage. Kromě instalací zároveň řeší závislosti, opravy, správu a metadata balíčků.

4.2.1 Základní příkazy

Software se instaluje pomocí příkazu:

```
emerge kategorie/nazev
```

Nevíme-li, některý z výše uvedených parametrů, můžeme využít příkaz:

```
emerge --search regulerni_vyraz
```

Příkaz pro jednoduchý update celého systému je:

```
emerge -avuDN @world
```

Uživatel by měl mít na paměti, že update může trvat až nižší desítky hodin. Pro lepší časový odhad je vhodné si přečíst obsah výstupu přepínače `-a`.

4.2.2 USE

Mezi hlavní výhody Portage patří USE Flags. Ty slouží ke konfiguraci sestavení a instalace balíčků. Určují jaké funkce mají programy obsahovat a podporovat.
Použijeme-li například vlajku `X`, bude všechen software instalován s podporou display serveru Xorg. Víme-li, že některé funkce nebudeme potřebovat, můžeme je odebrat pomocí znaménka mínus(pomlčky). Například: `-kde` odebere podporu grafických prostředí KDE. Ne všechny balíčky přepínače USE podporují. Podporu přepínačů určují vývojáři. Všechny použité přepínače se nachází ve složce `/etc/portage/make.conf`. Jejich seznam si můžeme vypsat pomocí příkazu:

```
cat /etc/portage/make.conf | grep USE
```

Seznam všech přepínačů se nachází na webu:

```
https://www.gentoo.org/support/use-flags/
```

Existují i přepínače procesor a grafický čip. Přepínače pro procesor se nastavují pomocí programu cpuid2cpuflags:

```
echo "*/* $(cpuid2flags)" > /etc/portage/package.use/00cpu-flags
```

Přepínače pro grafický čip, v případě potřeby, lze nastavit pomocí příkazu:

```
echo "*/* VIDEO_CARDS: *gpu*" > /etc/portage/package.use/00video-cards
```

gpu nahradíme dle následující tabulky:

Výrobce	Přepínač	poznámka
Nvidia	nvidia	-
Intel	intel	od 4. Generace
AMD	amdgpu radeonsi	od roku 2013/14
Raspberry Pi	vc4	-
QEMU	virgl	-
KVM	virgl	-
WSL	d3d12	-
Nvidia	nouveau	Open Source*

*Všechny modely kromě architektur Maxwell, Pascal, Volta

4.2.3 Licence

Od roku 2014 obsahují repositáře Gentoo štítky s licencemi balíčků. Pro lepší orientaci byla také zavedena proměnná `ACCEPT_LICENSE`. Licence byly rozděleny do skupin, které se deklarují v `/etc/portage/make.conf`:

```
ACCEPT_LICENSE="group"
```

Základní skupiny jsou:

Pro software	Popis	Svobodné
@GPL-COMPATIBLE	GPL kompatibilní	ANO
@FSF-APPROVED	Všechny licence schválené FSF	ANO
@MISC-FREE	Momentálně neschválené	ANO
@EULA	Licence pro koncového uživatele	NE
@OSI-APPROVED	Schválené OSI	NE
@OSI-APPROVED-FREE	Schválené OSI	ANO
@OSI-APPROVED-NONFREE	Schválené OSI	NE
@BINARY-REDISTRUTABLE	Neschválené	- - -

Pro dokumenty	Popis
@FSF-APPROVED-OTHER	FSF schválené non-Software Licence
@MISC-FREE-DOCS	Neschválená

Nadskupiny	Popis
@FREE	@FREE-SOFTWARE a @FREE-DOCUMENTS
@FREE-SOFTWARE	@FSF-APPROVED a @OSI-APPROVED-FREE
@FREE-DOCUMENTS	@FSF-APPROVED-OTHER @MISC-FREE-DOCS

Pokud uživatel chce mít možnost nainstalovat software pod jakoukoliv licencí, může zvolit přepínač `*`.

Jelikož se jedná o limitující nastavení, script nastaví hodnotu parametru na `*`. Důvodem je velké množství proprietárních ovladačů, nutných pro funkci fyzických prostředků.

V případě, že uživatel chce použít licenci pouze v rámci balíčku, lze toho docílit pomocí záznamu v `/etc/portage/package.use/` ve tvaru:

```
nazev/balicku licence
```

K propsání změn v `ACCEPT_LICENSE` je nutné systém aktualizovat pomocí:

```
emerge --ask --verbose --update --deep --newuse @world
```

5 Kernel

Jádro, kernel, je součástí většiny operačních systémů. Zajišťuje přístup k fyzickým prostředkům počítače. Mezi hlavní funkce patří správa: procesorů, procesů, pamětí a periferií.

Program nikdy nekomunikuje přímo s komponentou, místo toho komunikuje s jádrem pomocí systémových volání, které pak předá instrukce komponentě. Vyhodou je, že jádro může takto sbírat instrukce od více procesů a následně je řadit podle plánovacího algoritmu, tzv. scheduler.

5.1 Scheduler

Scheduler pro linuxové jádro se nazývá **C**ompletely **F**air **S**cheduler (**CFS**). Účelem je poskytnout všem entitám stejný čas na procesoru. Entity zaštiťují jeden proces či skupinu procesů. Nejmenší možnou entitou je jeden jednovláknový proces, největší entita (skupina) může obsahovat všechny běžící procesy, taková entita je označena `root_task_group`.

5.1.1 Group scheduling

Skupiny mají přidělenou hodnotu `cpu.shares`. Poměr hodnot `cpu.shares` jednotlivých skupin určuje jejich čas na procesoru. Obsahuje-li skupina podskupiny, postup se opakuje. Dělení na skupiny není teoreticky nutné, avšak je žádoucí například na systémech s více uživateli.

5.1.2 Task scheduling

Po přidělení času (pod)skupinám je potřeba naplánovat jednotlivé procesy skupiny. Plánování se provádí pomocí hodnoty `vruntime`. `vruntime` je odvozen z doby strávené na procesoru a priority(NICE). Platí, že rostoucí priorita (klesající NICE) zpomaluje růst `vruntime`. Procesy s vyšší prioritou dostanou více prostoru. Procesy jsou podle `vruntime` seřazeny do Red-black tree. Jedná se o upravený binární strom s barevně označenými uzly. Strom vždy řadí menší hodnotu (v tomto případě `vruntime`) vlevo. Barvy uzlů se průbežně mění, vždy však splňují čtyři podmínky:

1. Vkládaný uzel je červený.
2. Každý uzel je černý nebo červený
3. Nikdy nesmí být dva červené uzly bezprostředně nad sebou.
4. Cesty z libovolných černých uzlů na konec obsahují stejný počet černých bodů.

Uvedené podmínky zaručují že, jakýkoliv takový strom s n uzly bude obsahovat maximálně $2\log(n+1)$ vrstev. Asymptotická složitost operací je tedy $O(\log n)$.

5.2 Firmware

Firmware je kód běžící přímo na fyzické komponentě. Často je uložen v paměti, která je součástí komponenty. V případech kdy tomu tak není je potřeba jej nahrát z kernelu operačního systému. Nejčastěji se jedná o grafické karty, síťové a zvukové karty. Firmware Linuxu uložen v adresáři `/usr/lib/firmware`, odkud je při zavádění načten do zařízení. Velkou část potřebného firmwaru obsahuje `sys-kernel/linux-firmware`.

`emerge sys-kernel/linux-firmware¨`

Pro procesory od výrobce Intel je potřeba stáhnout ještě mikrokód:

`emerge sys-firmware/intel-microcode`.

Procesory od AMD mají mikrokód z `sys-kernel/linux-firmware`.

5.3 Initramfs

Initial **ram**-based **f**ile **s**ystem je malý filesystem. Cílem initramfs je připravit minimální prostředí pro spuštění Init systému. Provádí načtení kernel modulů a driverů pro zavedení důležitých filesystémů. Jakmile je vše potřebné připraveno, předá kontrolu Init systému. Initramfs je plnohodnotné uživatelské prostředí, je tedy možné v něm spouštět procesy. Standartním využitím této vlastnosti je záchranný systém nebo zadávání hesla pro odšifrování disku. Méně standartním využitím je hraní hry Tetris.

Generaci initramfs provádí Installkernel najde-li dracut parametr v souboru adresáře /etc/portage/package.use/.

Script parametr vkládá do souboru installkernel:

`echo "sys-kernel/installkernel dracut" | tee -a /…/installkernel`

Úprava initramfs vyžaduje velké množství znalostí a schopností, proto ji nebudeme více rozebírat. Podrobnější informace obsahuje Gentoo wiki.

5.4 Linux Kernel

5.4.1 Architektura

Architektura linuxového jádra je monolitická, modulární. Je tedy možné dynamicky přidávat nebo odebírat moduly. Linuxové jádro může tedy teoreticky obsahovat jen momentálně potřebné moduly. Jádro je sestaveno jako jeden blok, proto není potřeba mezi procesova komunikace jako u mikrojádra.

5.4.2 Kompilace