

Ubuntu Debian správce serveru díl II

Libor Jelínek

aktualizováno pro
Ubuntu 18.04 LTS



Umíte základy Linuxu, ale chcete vědět, jak
profesionálně spravovat linuxový server

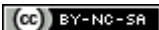
Vítá vás příručka Ubuntu a Debian správce serveru II!

Vítáme budoucí i současné administrátory v příručce „Ubuntu Debian správce serveru II“! Tato publikace může být skvělým doplňkem k našim [kurzům Linuxu](#), ale je koncipována jako zcela samostatná. Nabízíme ji zdarma všem návštěvníkům a samozřejmě našim studentům. Budeme rádi, když vám pomůže naučit se Ubuntu, Debian nebo dokonce i jinou distribuci Linuxu.

Tip

Líbí se vám tato knížka? Přijďte na [školení od autorů této příručky](#) na [Vacademy.cz](#)!

Licenční ujednání

Copyright © 2017 Virtage Software. Tato příručka je publikována pod  licencí [CC BY-NC-SA 4.0](#) (Uved'te původ-Neužívejte dílo komerčně-Zachovejte licenci 4.0 Mezinárodní). Tato licence dovoluje text sdílet a distribuovat v jakémkoli formátu nebo médiu *kromě použití pro výdělečné účely*. Text můžete upravovat a pozměňovat, pokud zachováte stejnou licenci. Vystavitel licence může tyto podmínky v budoucnu upravovat. Při sdílení a šíření je nutné uvést původ např. URL odkazem.

1. Disky a zařízení

„V Linuxu je vše soubor. Pokud to není soubor, je to proces.“

—Kamenné pravidlo souborů v Linuxu

1.1. Druhy souborů

Soubor v Linuxu má velmi široký význam a „běžný“ soubor s textem, obrázkem ap. je jen jedním z druhů souborů, které existují. První zvláštností je, že mezi soubory v Linuxu zahrnujeme tzv. souborová zařízení reprezentující různé hardwarové a jiné komponenty. Rovněž složka je zvláštním druhem souboru, který obsahuje seznam jiných souborů.

Přístup „vše je soubor“ má velké množství výhod, které postupně objevíme sami.

Typ souboru zobrazuje notoricky známý příkaz `ls` s volbou `-l` se kterou před oprávněním souboru uvede jeho typ (první znak prvního sloupce). Např. ve výpisu domovské složky `ls -l ~` najdeme obyčejný soubor (-) a složky (d):

```
...
-rwxr-xr-x  2 sandy sandy    1053 Oct 25 23:40 pgadmin.log
drwxr-xr-x  2 sandy sandy    4096 Oct 25 23:40 Pictures/
drwxrwxr-x  4 sandy sandy    4096 Oct 13 10:00 snap/
drwxrwxr-x  9 sandy sandy    4096 Oct 18 13:12 workspace/
...
```

Tip

Skoro jistě můžete místo `ls -l` používat `ll`. Je to tzv. alias pro `ls -l`.

Další druhy souborů najdete v tabulce.

Druhy souborů a odpovídající symboly ve výpisu `ls -l`.

Symbol v <code>ls -l</code>	Význam
--------------------------------	--------

-	<i>normální soubor</i> s texty, obrázky, hudbou ap.
---	---

Symbol v <code>ls -l</code>	Význam
d	<i>složka (directory)</i> - speciální soubor obsahující názvu jiných souborů
l	<i>symbolický odkaz (link)</i> na jiný soubor
c	<i>znakové (character) zařízení</i> . Viz bloková a znaková zařízení .
s	<i>doménový soket (domain socket)</i> je podobný soketům protokolu TCP/IP. Slouží jako prostředek interprocesové komunikace (IPC).
p	<i>pojmenovaná roura (named pipe)</i> . Podobně jako sokety jsou prostředkem pro IPC, ale bez sémantiky síťových socketů.
b	<i>blokové zařízení</i> . Viz bloková a znaková zařízení .

1.2. Oddíly a souborové systémy

Fyzický pevný disk může být rozdělen na více *oddílů (partitions)*. Oddíly jsou na sobě nezávislé a pád nebo chyba jednoho neovlivní druhý. OS Linux v typické instalaci používá více oddílů.

Důvod je především historický, kdy ještě nebyly používány *žurnálovací souborové systémy (filesystems)* a problém by mohl vést až ke ztrátě dat. Ale i dnes má rozdělení na více oddílů význam. Databáze může např. nekontrolovaně zaplnit veškeré místo do posledního bajtu a zastavit systém. Tím že jsou však soubory na jiném oddílu, než OS, nebude tím chod OS ovlivněný.

Žurnálovací file systémy

Zabezpečují data před neočekávaným výpadkem napájení nebo odpojením zařízení tím, že zapisují změny nejprve do speciálního záznamu - žurnálu - a teprve pak skutečně na disk. Následně je údaj o úspěšné „realizaci“ na disku zapsán do žurnálu a nakonec zrušen. Díky tomu je možné v případě přerušení vrátit stav na před operací nebo ji dokončit.

Žurnálování nechrání však před chybami na disku a logickými chybami. Pro ochranu tohoto druhu slouží např. RAID.

Staričkový FAT žurnálový není. Jinak žurnálují takřka všechny moderní filesystemy: NTFS (Win), HFS+ (Mac), všechny unixové a linuxové FS jako ext3, ext4, ReiserFS, XFS, JFS, ZFS.

Linux obvykle ke správné činnosti využívá nejméně dva druhy oddílů:

- **datový (běžný) oddíl** naformátovaný na některý z řady podporovaných souborových systémů pro ukládání dat OS nebo uživatelských dat. Běžných oddílů lze mít více.
- **odkládací (swap) oddíl**, který nemá žádné naformátování (je tzv. raw). Využívá ho OS při nedostatku RAM. Tento princip virtuální paměti podporují dnes všechny velké OS.

1.3. Startování z disku

1.3.1. Master Boot Record (MBR)

MBR je bootovací sektor (512 bajtů) na úplném začátku disku. MBR obsahuje

- informace o rozdělení disku na logické oddíly obsahující souborové systémy
- proveditelný kód, tzv. boot loader, který většinou jen předá řízení skutečnému spouštěči jako GRUB

1.3.2. GUID Partition Table (GPT)

Koncept MBR pochází z roku 1983 a je postupně nahrazován GPT. GPT tabulku najdeme u všech moderních strojů s UEFI firmwarem (nástupce BIOSu). Z GPT můžou bootovat všechny moderní OS včetně OS X a Windows a samozřejmě Linuxu. Hlavní linuxové nástroje pro práci z disky byly aktualizovány pro podporu GPT.

1.3.3. GRUB

GRUB je standardní boot loader většiny linuxových distribucí. Nahradil starší LILO (Linux LOader). Je to „to menu po startu PC s výběrem OS“. Umožňuje startovat nejen Linuxy, ale i DOS, Windows, BSD a Solaris systémy.

1.4. Souborová zařízení

Složka `/dev/` obsahuje speciální soubory reprezentující zařízení a komponenty připojená k počítači. Kromě skutečného hardwaru jde i o různá pseudo a simulovaná zařízení jako generátor náhodných čísel ap.

„Vše je soubor“ je doslova geniální rozhodnutí. Zařízení můžete nastavovat <<../usrv1/06-souborova-opravneni.adoc,vlastníka, skupinu a oprávnění>>, vytvoření image oddílu se rovná čtení souboru zařízení oddílu, vtištění může být posláno na soubor zařízení tiskárny atd.

1.4.1. Bloková a znaková zařízení

Linux rozlišuje mezi znakovými a (běžnějšími) blokovými zařízeními.

Bloková (block device):

- umí „udržet“ data
- disketa, pevný disk, USB flash paměť, USB disk ap.
- můžete číst/zapisovat jakýkoli blok bajtů
- bufferuje
- nevýhoda buferování je, že nevíte, že data byla už zapsána na zařízení
- náhodný přístup

Znaková (character device):

- slouží k „protékání“ dat
- pásky, sériové linky
- nebufferovaný přímý přístup
- neznamena, že můžete číst/zapisovat jen po jednom znaku (toto rozhodnutí je na zařízení samotném)
- sekvenční přístup

1.4.2. Nejdůležitější zařízení v /dev/

Následující výčet není v žádném případě úplný. Seznámíme se jen s některými nejdůležitějšími skutečnými i pseudo zařízeními. Ne všechna musí být ve vašem Linuxu, resp. v počítači existovat.

Poznámka

Některá zařízení a souborové zařízení jsou již pomalu počítačovým dávnověkem, ale přesto se domníváme, že stojí za to je zmínit.

/dev/fd[0-9]

První disketová jednotka je **fd0**. Druhá **fd1** ap.

/dev/hd[a-d]

Pevné disky připojené přes IDE rozhraní. **hda** je primary master, **hdb** je primary slave, **hdc** secondary master, **hdd** secondary slave.

/dev/hd[a-d][1-9]

Oddíly na daném IDE disku. Oddíly 1-4 jsou primární oddíly. Oddíly 5+ jsou logické oddíly uvnitř rozšířených oddílů. Takže např. **hdb1** je primární partition na primary master.

/dev/lp[0-9]

lp0 je první paralelní tiskárna ap.

/dev/loop[0-9]

Tzv. loopback zařízení jsou pseudozařízení sloužící ke zpřístupnění souboru jako blokového zařízení (např. připojení .iso obrazu jako disku).

/dev/null

„Černá díra“ ve které nenávratně zmizí cokoli tam zapíšete. Užitečnost tohoto zařízení je hlavně pro skripty, kdy do černé díry přesměrujete výstup, která vás nezajímá.

```
# stderr výstup findu nás nezajímá - přesměrován do černé díry  
# stdout bude stále na konzoli  
$ find / -name foo 2> /dev/null
```

/dev/psaux

PS/2 port.

/dev/cdrom a /dev/dvd

CD, resp. DVD mechanika. Jde o linky na konkrétní [sr* zařízení](#).

/dev/random a /dev/urandom

Generátory náhodných čísel. `random` je nedeterministický, což znamená, že následující číslo nelze odhadnout z předchozích čísel. `urandom` je „pouze“ pseudonáhodný, ale taky rychlejší. Nezáleží vám na vysoké bezpečnosti, postačí `urandom`.

Tip

Příklady využití souborů `random` a `urandom`.

Chcete vytvořit umělé zatížení PC? Čtete z `urandom` a posílejte ho do `null`.

```
cat /dev/urandom > /dev/null
```

Chcete vytvořit 10 MB „náhodný“ soubor (náhodného obsahu)?

```
dd if=/dev/urandom of=random.bin bs=1M count=10
```

/dev/sd[a-z]

Původně `sd` zařízení byli SCSI disky, ale toto rozhraní se nikdy výrazněji nerozšířilo a s masivním nástupem SATA disků se vývojáři rozhodli využít tohoto značení pro zařízení s tímto rozhraní. Písmena abecedy jsou přiřazovány, tak jsou zařízení nalezeny na sběrnici - první `sda`, druhé `sdb` ap.

/dev/sd[a-z][0-9]

Určuje oddíl na konkrétním SATA disku. Oddíly jsou číslovány od 1. Např. `sda3` je třetí oddíl na prvním SATA disku.

/dev/sr[0-9]

Souborové zařízení pro CD/DVD-ROM. `sr0` je první, `sr1` druhé atd.

`/dev/ttyS[0-9]`

Sériový port.

`/dev/zero`

Čtení zero zařízení vrátí vždy nulové znaky (0x00). Užitečnost je opět spíše pro skripty, kdy chcete vytvořit velký soubor vyplnit do určité velikosti „ničím“.

Poznámka

Nulový znak (občas `NUL` nebo `\0`) není nula! Jde o kontrolní znak podobně jako `\t` (tab), `\n` (nový řádek) ap. V Unicode i ASCII má hodnotu nula (0x00). Původně význam byl ignorovaný znak, ale dnes v řadě programovacích jazyků indikuje konec řetězce.

1.5. Připojení a odpojení

Před použitím se musí souborový systém připojit. Linux má plochou adresářovou strukturu. Každá složka může být na zcela jiném oddílu.

Tip

Často se to používá např. na serveru samostatný oddíl pro `/var/` a samotný OS na `/`. Nebo na notebooku umístění `/` na menší, ale rychlejší SSD, a `/home/` na pomalejší, ale velký mechanický HDD.

1.5.1. mount

Pro připojení diskového oddílu slouží příkaz `mount`. V základní podobě akceptuje dva parametry - soubor zařízení na kterém leží připojovaný filesystem a složku, kam ho připojit. Této cílové složce se často říká *přípojný bod (mount point)*.

Základní podoba

Pokud není nastaveno jinak, smí mount provádět jen root.

```
# Připojení sdc5 jako /home/sally/ mount
sudo mount /dev/sdc5 /home/sally/
```

Důležité

Cílová složka nemusí být prázdná, ale musí existovat. Případný předchozí obsah po připojení se neztratí, ale je zastíněn a dočasně nedostupný.

Určení souborového typu

Linux podporuje téměř všechny myslitelné souborové systémy a pokusí se jej na zařízení rozpoznat. Přesto bývá dobrým zvykem typ souborového systému určit parametrem `-t`:

```
# Explicitní určení souborového systému
mount -t ntfs /dev/sdc5 /home/sally/win_backup
```

Volby připojení

Dalším často používaným parametrem je `-o` pro upřesnění způsobu připojení. Např.:

```
mount -t ntfs -o ro,user /dev/sdc5 /home/sally/win_backup
```

Některé volby jsou nezávislé na souborovém systému, některé platí jen pro konkrétní souborové systémy. Z obecných a vždy použitelných jsou důležité zejm.:

- `auto` a `noauto` – viz Připojení po startu.
- `rw` a `ro` – připojí zařízení ke čtení i zápisu/pouze ke čtení
- `suid` a `nosuid` – umožní/zakáže spouštění souborů s právem SUID
- `exec` a `noexec` – umožní/zakáže spouštění souborů s právem spustit

- `user` a `users` - viz [Připojování pro běžné uživatele](#).

Všechny další obecné volby najdete popsány v `man mount`. Ty specifické v manuálových stránkách jednotlivých filesystémů (např. `man mount.ntfs` pro NTFS).

1.5.2. `umount`

Poznámka

Je to opravdu `umount`, nikoli `unmount`.

Pro odpojení slouží `umount` a má jediný parametr - buď soubor zařízení nebo přípojný bod (složku):

```
# Odpojení přes zařízení
umount /dev/sdc5

# nebo přes přípojný bod
umount /home/joe/
```

1.5.3. `/etc/fstab`

Hlavním účelem souboru `/etc/fstab` je definice připojení, který se mají provést během startu PC.

Formát souboru

Jde o jednoduchý textový soubor, kde každý řádek definuje, jedno připojení. Sloupce (pole) oddělené mezerami nebo tabulátory jsou

`<zařízení> <přípojný bod> <typ filesystému> <volby> <dump> <pass>`

které mají postupně tento význam:

- 1. pole: **zařízení** – zdrojové zařízení (odkud), které bývá často místo souborového zařízení (např. `/dev/sdc3`) specifikováno UUID (Universally Unique Identifier).

Poznámka

UUID (Universally Unique Identifier) je unikátní identifikace diskového zařízení (např. fc64422e-669c-11e8-bd41-0800272870d0). Má výhodu, že je jednoznačné a vytváří se již při naformátování. Stejně zařízení bude připojeno vždy stejně. Klasické určení souborovým zařízením jako např. `/dev/sda1` je závislé na pořadí nalezení na sběrnici ap.

Ke zjištění UUID slouží `blkid`. Bez parametrů vypíše UUID všech diskových zařízení. UUID konkrétního zařízení např. `/dev/sda` zjistíme zadáním `blkid /dev/sda`.

- 2. pole: *přípojný bod* – cílový přípojný bod (kam)
- 3. pole: *typ filesystemu* – typ filesystemu na zařízení (ext4, xfs, nts ap.)
- 4. pole: *volby (mount options)* – obecné nebo pro file systém specifické volby připojení.
- 5. pole: *dump* – 0 nebo 1, pro zálohovat/nezálohovat programem dump. Defaultně 0.

Poznámka

Dump je staričkový zálohovací program, který se kterým se skoro jistě ne-setkáte a proto toto pole fstabu nemá z dnešního pohledu význam. Může se však přece jen stát, že dump nebo jiný program pro zálohování tento údaj čte.

- 6. pole: *pass* – pořadí při kontrole svazku programem fsck při startu počítače. 0 znamená nekontrolovat.

Podívejme se na příklad `/etc/fstab` serveru s rozdělením `/`, `/var/`, swapem a jednou vzdálenou složkou připojenou přes NFS:

UUID=fc64422e-669c-11e8-bd41-0800272870d0	/	ext4	default
UUID=fc64422f-669c-11e8-bd41-0800272870d0	/var	ext4	default
/swap.img	none	swap	sw
192.168.121.27:/var/logs/patton/	/opt/ezclue/logs/	nfs	bg,hard

Připojení po startu

I když je hlavním úkolem `fstab` připojit záznamy během bootování PC, můžeme mít ve volbách (4. pole) záznamu mít uvedeno `noauto` a řádek se vynechá.

Pokud `noauto` není mezi volbami uvedeno nebo je uvedeno `auto` je zařízení připojeno na zavolání `mount -a` (obvykle ve startovacích skriptech).

Připojování pro běžné uživatele

Jak jsme řekli výše, může připojení a odpojení provádět jen root. Existuje však možnost jak tyto operace povolit i pro běžné uživatele. Jestliže mezi volbami (4. pole) v záznamu ve `fstab` je volba

- `user` – umožní se připojení zařízení i ne-root uživatelům. Odpojit ho může jen root a uživatel, který zařízení připojil.
- `users` – umožní se připojení jakémukoli uživateli. Odpojit ho může také kdokoli, dokonce jiný uživatel, než ho připojil.

1.6. Programy pro práci s disky a oddíly

Upozornění

Většina programů vyžaduje root oprávnění (provádějte pomocí `<<../usrv1/04-uzivatele-skupiny.adoc#sudo,sudo>>`). Vychovanější programy vypíší chybu, jiné bohužel bez `sudo` na obrazovku vůbec nic vytisknout.

1.6.1. dd

Program pro nízkourovňové binární kopírování bajt po bajtu. Vhodné pro image disku, kopii MBR ap.

Základními parametry jsou `if` (input file), `of` (output file), `bs` (block size), a `count` (počet).

```
# Vytvoření náhodného 10 MiB souboru
dd if=/dev/urandom of=random.bin bs=1M count=10
```

1.6.2. lsusb

Vypíše informace o USB sběrnících a zařízeních k nim připojených.

Důležité parametry jsou `-t` pro výpis ve stromu a `-v` pro detailní výpis.

```
$ sudo lsusb -t
/: Bus 04.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
   |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/8p, 480M
/: Bus 03.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
   |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/6p, 480M
       |__ Port 3: Dev 3, If 0, Class=Vendor Specific Class, Driver=rtss_usb,
           |__ Port 4: Dev 4, If 0, Class=Wireless, Driver=btusb, 12M
           |__ Port 4: Dev 4, If 1, Class=Wireless, Driver=btusb, 12M
           |__ Port 5: Dev 5, If 0, Class=Video, Driver=uvcvideo, 480M
           |__ Port 5: Dev 5, If 1, Class=Video, Driver=uvcvideo, 480M
/: Bus 02.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/4p, 5000M
/: Bus 01.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/4p, 480M
```

1.6.3. lshw

Základní program pro výpis informací o hardwaru. Může zjistit údaje o přesné konfiguraci paměti, obsazených bankách, firmwaru, CPU ap.

Bez parametrů vypíše všechny známé údaje. Druh informací omezíte parametrem `-C`, `-class` např. jen o síťovém hardware:

```
sudo lshw -class network
```

Dostupné třídy (kategorie) zjistíte ve výpisu `sudo lshw -short`.

1.6.4. lsof

Vypisuje na STDOUT informace o souborech otevřených procesem.

Bez parametrů vytvoří velmi dlouhý výstup otevřených souborů všech aktivních procesů. Mezi velkým množstvím parametrů zmíníme jen `-i` pro zjištění jaký proces okupuje síťový port.

```
$ lsof -i:8000
COMMAND PID  USER  FD   TYPE DEVICE SIZE/OFF NODE NAME
python  2143 sally  3u   IPv4 394753      0t0  TCP localhost:8000 (LISTEN)
```

1.6.5. fdisk

Program stejného jména z MS-DOSu existuje i pro Linux, ale jeho ovládání je založeno na parametrech příkazové řádky.

Protože existují uživatelský příjemější alternativy jako [cfdisk](#) nebo [parted/gparted](#), použití tohoto program je většinou spíše jen k vypsání informací s `-l` - disk, UUID, oddíl, velikosti, souborové systémy a typ tabulky rozdělení (GPT, MBR, ...):

```
$ sudo fdisk -l
Disk /dev/loop0: 86.6 MiB, 90759168 bytes, 177264 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop1: 86.6 MiB, 90812416 bytes, 177368 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/sda: 30 GiB, 32212254720 bytes, 62914560 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 17104833-25AA-43D6-8092-97CF7324D8BC

Device      Start      End  Sectors  Size Type
/dev/sda1    2048      4095     2048    1M BIOS boot
/dev/sda2    4096 20975615 20971520  10G Linux filesystem
/dev/sda3  20975616 62912511 41936896  20G Linux filesystem
```

1.6.6. cfdisk

Snadnější a modernější alternativou je `cfdisk`, který připomíná ovládáním `fdisk` z MS-DOSu.

```
sudo cfdisk
```

```

Disk: /dev/sda
Size: 30 GiB, 32212254720 bytes, 62914560 sectors
Label: gpt, identifier: 17104833-25AA-43D6-8092-97CF7324D8BC

Device          Start      End        Sectors    Size Type
/dev/sda1       2048       4095        2048       1M BIOS boot
/dev/sda2       4096      20975615   20971520   10G Linux filesystem
>> /dev/sda3     20975616   62912511   41936896   20G Linux filesystem

Partition UUID: 8AE66D0A-3557-47AA-9105-F22657044C1A
Partition type: Linux filesystem (0FC63DAF-8483-4772-8E79-3D69D8477DE4)
Filesystem UUID: fc64422f-669c-11e8-bd41-0800272870d0
Filesystem: ext4
Mountpoint: /var (mounted)

[ Delete ] [ Resize ] [ Quit ] [ Type ] [ Help ] [ Write ] [ Dump ]

Quit program without writing changes

```

Program cfdisk

1.6.7. parted/gparted

Textový parted a grafický GParted jsou zástupci pokročilých programů pro správu disků a oddílů. parted bývá součástí instalace. Klikací GParted připomínající komerční Partition Magic se velmi lehce ovládá.

GParted je k dispozici také jako [GParted Live](#), tj. jako malá bootovatelná distribuce obsahující nejen GParted, ale i mc, fdisk, SSH, telnet ap.


```
sally@tristar:~$ sudo parted
GNU Parted 3.2
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) list
align-check TYPE N          check partition N for TYPE(min|opt) alignment
help [COMMAND]              print general help, or help on COMMAND
mklabel,mktable LABEL-TYPE create a new disklabel (partition table)
mkpart PART-TYPE [FS-TYPE] START END make a partition
name NUMBER NAME            name partition NUMBER as NAME
print [devices|free|list,all|NUMBER] display the partition table, available devices, free
                             space, all found partitions, or a particular partition
quit                          exit program
rescue START END            rescue a lost partition near START and END
resizepart NUMBER END       resize partition NUMBER
rm NUMBER                    delete partition NUMBER
select DEVICE                choose the device to edit
disk_set FLAG STATE         change the FLAG on selected device
disk_toggle [FLAG]          toggle the state of FLAG on selected device
set NUMBER FLAG STATE       change the FLAG on partition NUMBER
toggle [NUMBER [FLAG]]      toggle the state of FLAG on partition NUMBER
unit UNIT                    set the default unit to UNIT
version                       display the version number and copyright information
                             of GNU Parted
(parted) █
```

The screenshot shows the GParted graphical interface for /dev/sda (465.76 GiB). The main display shows two partitions: /dev/sda2 (160.48 GiB) and /dev/sda3 (270.45 GiB). Below this is a detailed table of all partitions on the disk.

Partition	File System	Mount Point	Label	Size	Used	Unused	Flags
/dev/sda1	ntfs		System Reserved	350.00 MiB	283.12 MiB	66.88 MiB	boot
/dev/sda2	ntfs			160.48 GiB	--	--	
unallocated	unallocated			1.29 GiB	--	--	
▼ /dev/sda4	extended			33.20 GiB	--	--	
/dev/sda5	ext4			3.90 GiB	215.39 MiB	3.69 GiB	
/dev/sda8	ext4	/		12.10 GiB	8.06 GiB	4.04 GiB	
unallocated	unallocated			4.00 MiB	--	--	
/dev/sda6	ext4			14.90 GiB	404.89 MiB	14.50 GiB	
/dev/sda7	linux-swap			2.29 GiB	4.00 KiB	2.29 GiB	
/dev/sda3	ntfs			270.45 GiB	223.85 GiB	46.60 GiB	
unallocated	unallocated			1.02 MiB	--	--	

1.7. Logical Volume Management (LVM)

Důležité

Úvod a výhody LVM místo tradičního rozvržení disku najdete v „./usrv1/02-instalace.adoc#lvm,kapitole o instalaci“.

LVM je alternativním a moderním způsobem správy disků v Linuxu. Hlavní výhodou LVM je, že všechny operace jsou online za běhu bez nutnosti zastavení, restartu vč. změny velikosti oddílů, snapshotů ap. Všechny současné distribuce a nástroje LVM podporují.

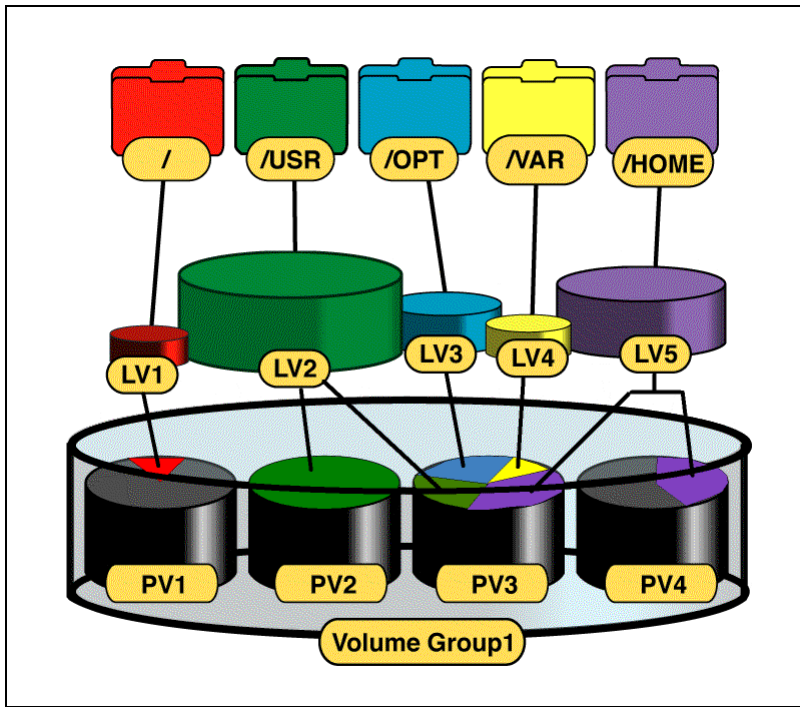
Upozornění

Současná verze o které budeme dále hovořit je LVM 2. Nedoporučujeme pracovat se starší verzí LVM.

1.7.1. Terminologie LVM

LVM používá několik klíčových termínů, které musíme jako první vysvětlit:

- **Volume group (skupina svazků) (VG)** sdružuje LV a PV do jedné administrativní jednotky. Je to nejvyšší úroveň členění v LVM. Můžeme ji pojmenovat např. podle počítače nebo „dpt1“ ap.
 - VG mohou být zvětšena/zmenšena přidáním/odebráním nových PV
- **Physical volume (fyzický svazek) (PV)** je obvykle odpovídá fyzickému zařízení pevného disku, ale může se jednat jen o „pohled“ na disk třeba v případě softwarového RAIDu.
 - každý PV je rozdělen do úseků dat známých jako physical extent (PE) o stejné velikosti jako logical extent (LE) (viz další odrážka).
- **Logical Volume (logický svazek) (LV)** je ekvivalent diskového oddílu v ne-LVM systému. LV je viditelný jako běžné blokové zařízení a obsahuje samotný souborový systém.
 - každý LV je rozdělen do úseků dat známých jako logical extent (LE) ve velikosti stejné pro všechny LV ve VG.
 - LV mohou být zvětšena/zmenšena spojením/rozpojením nových extents
 - LV mohou být přemístěny mezi PV
 - na rozdíl od tradičních oddílů mají jména místo čísel, mohou se rozkládat napříč více PV (disky), které nemusí být ani fyzicky v řadě



Organizace prvků LVM (obrázek převzat z <http://www.markus-gattol.name/ws/lvm.html>)


1.7.2. Nastavení LVM během instalace

Nastavit LVM již během instalace je nejjednodušší a doporučený způsob „jak na LVM“. Ubuntu Server od verze 17.10 používá nový instalátor, který prozatím nemá možnost nastavení LVM již během instalace. Pro LVM a některé další pokročilé volby je nutné stáhnout tzv. *alternativní instalátor*.

1. Na stránce <https://www.ubuntu.com/download/alternative-downloads> najdete „Alternative Ubuntu Server installer“.

Alternative Ubuntu Server installer

If you require advanced networking and storage features such as; LVM, RAID, multipath, vlans, bonds, or re-using existing partitions, you will want to continue to use the alternate installer.

[Download the alternate installer](#) 

Other image mirrors

For other versions of the UI installer please select your mirror; however, we recommend the [standard installer](#) as all packages are available in UI Software Centre.

[See all Ubuntu mirrors](#) 

Pro pokročilé možnosti instalace jako LVM je třeba sáhnout po alternativním instalátoru.

2. Dostanete se na výpis souborů podle procesorové architektury a podle způsobu stažení (iso/bittorrent ap.). Pravděpodobně hledáte soubor `ubuntu-18.04-server-arm64.iso` (klasický instalátor se jmenuje `ubuntu-18.04-live-server-amd64.iso`).
3. Instalátor nabídne nastavit LVM hned při instalaci.

[!!] Partition disks

If you choose guided partitioning for an entire disk, you will next be asked which disk should be used.

Partitioning method:

Guided - use entire disk
Guided - use entire disk and set up LVM
Guided - use entire disk and set up encrypted LVM
Manual

<Go Back>

<Tab> moves; <Space> selects; <Enter> activates buttons

[!!] Partition disks

This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialize its partition table.

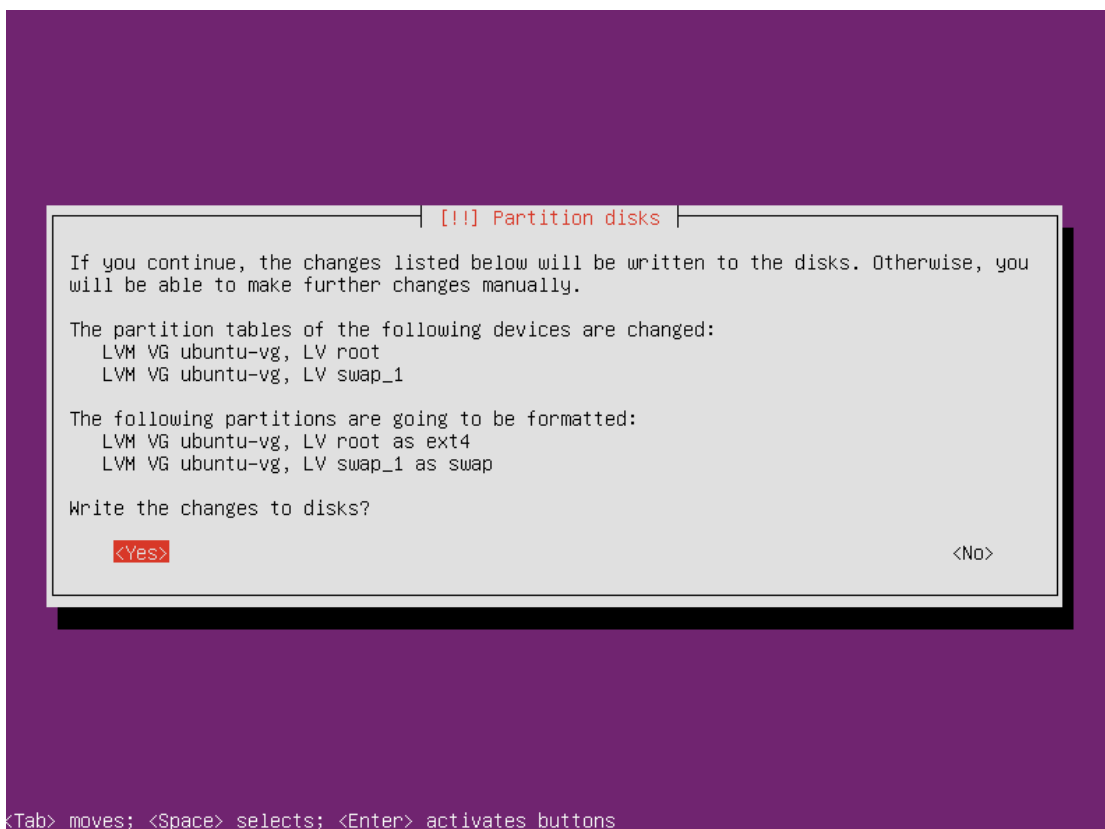
Guided partitioning
Configure software RAID
Configure the Logical Volume Manager
Configure encrypted volumes
Configure iSCSI volumes

LVM VG ubuntu-vg, LV root - 31.2 GB Linux device-mapper (linear)
#1 31.2 GB f ext4 /
LVM VG ubuntu-vg, LV swap_1 - 1.0 GB Linux device-mapper (linear)
#1 1.0 GB f swap swap
SCSI3 (0,0,0) (sda) - 32.2 GB ATA VBOX HARDDISK
#1 primary 32.2 GB K lvm

Undo changes to partitions
Finish partitioning and write changes to disk

<Go Back>

<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons



1.7.3. Praktický příklad použití LVM

Nástroje pro LVM tvoří sada programů s názvy začínající `vg*`, `pv*` a `lv*` (jako volume group, physical volume a logical volume) + sloveso jako `create`, `display`, `remove` ap. Např. `vgdisplay` zobrazí informace o volume group.

LVM nástroje by měli být standardně nainstalovány, ale kdyby ne, získáte je provedením `sudo apt-get install lvm2`.

```
# 2 disky při instalaci jeden jako LVM. Oba třeba 8 GB.

# druhý není nijak naformátovaný

sudo fdisk -l

# Jestli chci využít celý disk tak jak leží běží pro LVM, pak v příkazu pvcreat
sudo cfdisk /dev/sdb
# a vytvořte jeden Linux LVM (8e) oddíl plné velikosti disku

# Zapiše do parittion LVM hlavičku
sudo pvcreate /dev/sdb1

# Zobrazí physical volumes
sudo pvdisplay
# nebo
```

```
sudo pvs

# Vytvoří volume group ze zařízení
sudo vgcreate vg0 /dev/sdb1
# (lze zadat i více zařízeních)

# Zobrazí volume groups
sudo vgdisplay
# pozor, že "vgs" neexistuje

# Vytvoření logical volumes
# -n = jméno, -L = velikost vg0 = ze které volume group
sudo lvcreate -n g0_root -L 2g vg0
sudo lvcreate -n g0_home -L 2g vg0
sudo lvcreate -n g0_var -L 2g vg0

# Zobrazíme a uvidíme /dev/myvg1/logicka1
sudo lvdisplay
# nebo
sudo lvs

# Naformátování logických svazků na ext4
sudo mkfs.ext4 /dev/vg0/g0_root -L root
sudo mkfs.ext4 /dev/vg0/g0_home -L home
sudo mkfs.ext4 /dev/vg0/g0_var -L var

# Rozšíření o 1 GiB
sudo lvresize -L +1G vg0/g0_root
# příkazu můžete doplnit na konec PV ze kterých má být "ukrojeno", jinak se pou

# Přesvědčte se výpisem logical volumes
sudo lvdisplay

# Je třeba rozšířit i filesystem na novou velikost (pro ext4):
sudo resize2fs /dev/vg0/g0_root

# Připojte oddíly
sudo mkdir /mnt/g0_root
sudo mount /dev/vg0/g0_root /mnt/g0_root

# Přesvědčte se
df -h
```

Tip

Výmaz VG - Umí vymazat VG a s ní všechny LV. LV musí být umountovány.

Tip

Zvětšení oddílu i filesystemu v jednom kroku - `lvresize` má od vyšší verze LVM2 volbu `-r`, kdy zvětší i filesystem, takže nemusíme následně volat `resize2fs`.

2. Konfigurace sítí v Linuxu

Už před léty s příchodem [systemd](#) došlo k poměrně hlubokým změnám v konfiguraci sítí v Linuxu: překlad jmen zajišťuje `systemd-resolved` (nebo prostě jen `resolved`), síťová rozhraní na serverech spravuje `systemd-networkd` (nebo prostě jen `networkd`), na desktopech obvykle `Network Manager (NM)` atd.

Tyto změny přináší výhody pro složité systémy s mnoha síťovými adaptéry, systémy běžící v cloudu, minipočítače pro IoT atd. Bohužel taky komplikují nastavení sítě a banální požadavek „chci jen nastavit IP, masku, bránu, DNS“ je doslova věda. Kde jsou doby, kdy stačilo upravit soubor `/etc/network/interfaces...`

Ubuntu 18.04 obsahuje další změny v tradiční správě sítí v podobě správce [Netplan](#), který nahrazuje `ifupdown` (soubor `/etc/network/interfaces`). Motivací k dalšímu řešení je zjednodušení poměrně velké náročnosti konfigurace a různá konfigurace stejných věcí v `networkd` a `Network Manageru`.

V této kapitole se pokusíme vysvětlit všechny způsoby, abyste byli schopni administrovat nové i staré instalace Ubuntu.

2.1. Síťová rozhraní

Síťová rozhraní či síťové adaptéry jsou [souborová zařízení](#) ve složce `/dev/`. Můžete se setkat se dvěma způsoby pojmenovávání síťových rozhraní.

Tradiční pojmenování

Velice dlouhou dobu to bylo jediné pojmenování síťových rozhraní. Toto jednoduché schéma je podobné ostatním [souborovým zařízením](#). Např. pro kabelově vedený ethernet se adaptéry jmenují jako `eth0`, `eth1`, ap. tak jsou nacházena ovladačem.

Název se řídí typem adaptéru:

- `eth0`, `eth1`, ... – klasický kabelově vedený ethernet. První rozhraní je `eth0` atd.
- `lo` – loopback zařízení. Je speciální síťové rozhraní, které je vždy přítomné i na počítači bez skutečné síťové karty. Slouží k provozu síťových aplikací i bez nutnosti „jít“ na skutečnou síť. Má přiřazenou pevnou IP adresu 127.0.0.1.
- `wlan0`, `wlan1`, ... – bezdrátové Wi-Fi karty jsou označeny `wlan` (wireless LAN). První rozhraní je `wlan0` ap.

Tento způsob pojmenování má nevýhodu v tom, že není zcela spolehlivý. Může se např. stát, že při změně hardware dojde k záměně - adaptér je `eth0` je při jednom startu, `eth1` při dalším.

Pojmenování rozhraní v systemd

S příchodem `systemd` se změnilo v linuxových systém mnoho - i schéma přidělování jmen síťových rozhraní zvané [Predictable Network Interface Names](#) (predikovatelná jména síťových rozhraní). Místo tradičním jmen jako `eth0` se setkáte s názvy `enp5s0` ap., která jsou stabilní (predikovatelná), protože se tvoří na základě údajů jako číslo přečtené z firmwaru nebo BIOSu, číslem PCI sběrnice do které je karta vložena, fyzickým umístění ap.

Konkrétní postup tvorby jména je poměrně složitý a najdete zdokumentován ve [zdrojových kódech systemd](#). Např. název rozhraní `enp0s3` se vytvoří nějak takto:

- první dvě písmena (`en`) – typ rozhraní: např. `en` pro ethernet, `wl` pro wireless LAN (Wi-Fi)
- třetí písmeno a číslo (`p0`) – typ připojení a pořadí: např. `p` připojeno nultou PCI sběrnici
- poslední písmeno a číslo (`s3`) – číslo slotu: např. třetí slot na nulté PCI sběrnici.

Speciální IP adresy

Kromě IP adresy přiřazené administrátorem nebo DHCP systémem se používá několik speciální IP adres s pevným významem, které si nemůžeme zvolit pro síťové rozhraní:

- `0.0.0.0` (IPv4)/`::` (IPv6) – všechna síťová rozhraní počítače
- `127.0.0.1` (IPv4)/`::1` (IPv6) – loopback rozhraní (`/dev/lo`) (hostname `localhost`)

2.2. Netplan

Ubuntu 18.04 přináší další změny v tradiční správě sítí v podobě manažeru [Netplan](#), který nahrazuje `ifupdown` (soubor `/etc/network/interfaces`). Poprvé se Netplan objevil v Ubuntu 17.10 a motivací k novému řešení je zjednodušení poměrně velké náročnosti konfigurace a různá konfigurace stejných věcí v `systemd-networkd` (na serveru) a Network Manageru (na desktopu).

2.2.1. Jak Netplan pracuje

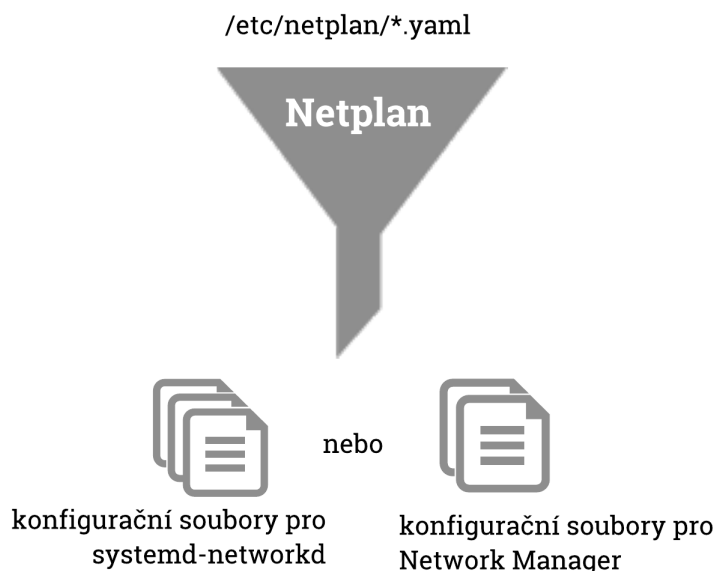
Netplan pracuje tak, že pomocí YAML souborů popíšete libovolně jednoduchou nebo složitou síťovou konfiguraci ze které Netplan vygeneruje konfiguraci pro zvolený *renderer* - buď `networkd` (výchozí) nebo `Network Manager`. Podstatně tím zjednodušuje naši práci, protože zejm. v `networkd` může znamenat jednoduchá změna nastavování až ve třech různých souborech (!).

Poznámka

„Skrýváním“ komplexity je Netplan podobný [UFW](#), jinému nástroji z Ubuntu pro správu firewallu.

Netplan očekává konfiguraci v `/etc/netplan/*.yaml` souborech. Můžete je zde vytvářet správce, ale často balíčky a podobné nástroje. Během startu počítače Netplan vygeneruje příslušné konfigurační soubory v `/run/`. Pokud není v YAML souboru jinak, vytváří konfiguraci pro `systemd-networkd`.

Všechny nalezené YAML soubory se seřadí. Klíče nalezené v souborech později v abecedě můžou přepsat klíče ze souborů dříve v abecedě, nebo vytvořit nové klíče. Právě z tohoto důvodu se setkáte soubory pojmenovanými jako `50-cloud-init.yaml`.



2.2.2. Konfigurace Netplan

Netplan používá jednoduchý textový formát YAML. Soubory s příponou `.yaml` hledá ve složce `/etc/netplan/`, které vypadají např. takto:

```
network:
  version: 2
  ethernets:
    eno1:
      dhcp4: true
```

YAML dokument začíná klíčem `network`, následuje `version: 2` (v tuto chvíli aktuální verze konfiguračního souboru) a konfiguracemi zařízení seskupené podle typu jako např. `ethernets` a `wifis`. Pod blokem typu zařízení je název síťového zařízení a jeho konfigurace.

Nejdůležitější vlastnosti společné pro všechny typy zařízení.

Klíč	Hodnota	Význam
<code>renderer</code>	<code>networkd</code> (výchozí) nebo <code>NetworkManager</code>	Pro jaký síťový systém vygenerovat konfiguraci.
<code>dhcp4</code>	<code>true/false</code> (výchozí)	Podpora DHCP pro IPv4.

Klíč	Hodnota	Význam
dhcp6	true/false (výchozí)	Podpora DHCP pro IPv6.
addresses	seznam IP adres	Pevná IP adresa nebo adresy zařízení vč. síťové masky v CIDR notaci. Např. addresses: [192.168.14.2/24, "2001:1::1/64"].
gateway4 a gateway6	seznam IP adres	IPv4/IPv6 výchozí brána. Vyžadováno, pokud je uvedeno addresses. Např. gateway4: 172.16.0.1 nebo gateway6: "2001:4::1"
nameservers	mapování s klíči addresses a search	Nastavení DNS serverů a domén. Hodnotou je mapování s klíčem addresses pro seznam IPv4 nebo IPv6 adres) a search pro seznam vyhledávaných domén). Např.: <pre>nameservers: search: [vacademy.net, vacademy.cz] addresses: [8.8.8.8, "FEDC::1"]</pre>

Netplan má řadu dalších možností - např. nastavení routovacích tabulek, bondovací a bridgeování zařízení, určení zařízení pomocí wildcard výrazů jako `enp2*` (všechny karty na druhé PCI směrnicí) ap. Kompletní seznam konfigurace najdete v `man netplan`.

Dvě nejčastější možnosti nastavení sítě - manuální a DHCP - si ukážeme na příkladech.

Příklad nastavení přes DHCP

```
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: true
```

Příklad manuálního nastavení

```
network:
  version: 2
  ethernets:
```

```
enp0s3:
  addresses: [192.168.1.115/24]
  gateway4: 192.168.1.1
  nameservers:
    search: [mycompany.local, myorg.local]
    addresses: [8.8.8.8, 192.168.1.2]o
```

2.2.3. Netplan příkazy

Ovládání Netplanu je velmi jednoduché. V podstatě jde jen o dva podpříkazy v podobě `netplan <podpříkaz>`.

netplan apply

Vygeneruje z YAML souborů konfiguraci pro zvolený renderer (defaultně `systemd-networkd`) a aplikuje změny (restartuje renderer). Např. pro výše uvedený příklad manuálního nastavení se vytvoří soubor `/run/systemd/network/10-netplan-enp0s8.network` s tímto obsahem:

```
[Match]
Name=enp0s8

[Network]
Address=192.168.88.115/24
Gateway=192.168.1.1
DNS=192.168.88.1
DNS=8.8.8.8
Domains=mycompany.local myorg.local
```

netplan generate

Jen vygeneruje konfiguraci pro renderer, neaplikuje změny.

2.3. ifupdown

Ubuntu 16.04 nahradilo tradiční konfiguraci sítí souborem `/etc/network/interfaces` novým manažerem [Netplan](#). Balíček `ifupdown` již není součástí instalace a proto ani programy `ifup`, `ifdown` a `ifquery`.

Důležité

V Ubuntu 18.04 se můžete vrátit k dřívější konfiguraci sítě, tak že nainstalujete balíček `ifupdown` a nastavíte `/etc/network/interfaces` manuálně jako dříve. **Pokud nejste na starém systému nebo nemáte jiný závažný důvod, doporučujeme však již nepoužívat ifupdown.**

Další možností je vynutit si ifupdown volbou `netcfg/do_not_use_netplan=true` během instalace, tak že na první obrazovce instalátoru stisknete F6, pak e a tuto volbu přidáte do příkazové řádky.

2.3.1. `/etc/network/interfaces`

Hlavní konfiguračním souborem sítě v Debianu a Ubuntu byl dlouhou dobu `/etc/network/interfaces` s informací pro nástroje `ifup` a `ifdown`.

Důležité

Pokud jste na Netplan systému v souboru najdete upozornění, že *ifupdown has been replaced by netplan(5) on this system* (ifupdown byl na tomto systému nahrazen netplan) a tohoto souboru si nevšímejte.

Konfigurační soubor `interfaces` obsahuje informace, jak se připojit k síti. Na rozdíl od později probíraných programů jako `ifconfig` je nastavení v souboru přečteno při startu počítače a tedy trvalé.

Pokud používáte v síti DHCP není třeba vůbec žádné nastavení a váš soubor `interfaces` obsahuje jen dva řádky pro zapnutí (`auto`) a nastavení `iface` loopback zařízení:

```
auto lo
iface lo inet loopback
```

Povely `auto`, `iface` a další nazývá dokumentace jako *stanzas*. Konfigurace je velmi rozsáhlá a proto pro pokročilejší nastavení odkazujeme na `man 5`

`interfaces`. Ukážeme si proto jen další typický setup se statickou IP adresou bez DHCP:

```
auto eth0
iface eth0 inet static
    address 192.168.1.1
    netmask 255.255.255.0
    gateway 192.168.1.50
    dns-nameservers 192.168.1.50
```

Potřebujete-li nastavit sekundární, terciární, ... DNS, pak do předchozího řádku přidejte mezerou oddělení jejich IP adresy:

```
dns-nameservers 192.168.1.50 192.168.1.51
```

Pro aplikaci nastavení je třeba provést restart sítě nebo restart počítače.

2.3.2. ifconfig

Poznámka

V DOSu a Windows je podobný program nazván `ipconfig`.

Ifconfig neboli interface configuration je nejznámější nástroj pro konfiguraci a diagnostiku sítí. Ipconfig slouží k nastavení síťových zařízení a jako takový neumí nastavit bránu (gateway) nebo DNS.

Upozornění

Veškeré změny provedené ifconfigem „nepřežijí“ restart. Pro trvalé změny musíte rozhraní nastavit konfigurací systémů [Netplan](#) nebo [ifupdown](#).

2.3.2.1. Informace o zařízeních

Bez parametrů vypíše základní informace o aktivních síťových rozhraních. S parametrem `-a` (all) i o těch neaktivních. `-s` (short) slouží ke stručnému výpisu.

```
$ sudo ifconfig -a -s
Iface      MTU      RX-OK RX-ERR RX-DRP RX-OVR      TX-OK TX-ERR TX-DRP TX-OVR Flg
enp0s3    1500    93652   0     0 0         46177   0     0     0 BMRU
enp0s8    1500     0       0     0 0           0       0     0     0 BM
lo        65536    231     0     0 0          231     0     0     0 LRU
```

2.3.2.2. Povolení/zakázání rozhraní

Povolení:

```
sudo ifconfig <rozhraní> up
```

Zakázání:

```
sudo ifconfig <rozhraní> down
```

2.3.2.3. Přiřazení IP adresy a masky podsítě

Přiřazení IP:

```
sudo ifconfig <rozhraní> <ip>
```

Přiřazení IP a masky:

```
sudo ifconfig <rozhraní> <ip> netmask <maska>
```

Pro ověření nového nastavení si můžete zkontrolovat přes `ifconfig <rozhraní>`.

2.3.2.4. Povolení/zakázání promiskuitního režimu

Povolení promiskuitního režimu:

```
sudo ifconfig <rozhraní> promisc
```

Zakázání promiskuitního režimu:

```
sudo ifconfig <rozhraní> -promisc
```

2.3.2.5. Změna MAC adresy

Inconfig dokonce umožňuje nastavit MAC adresu rozhraní, např.:

```
sudo ifconfig enp0s8 hw ether AA:BB:CC:DD:EE:FF
```

Poznámka

Další pokročilejší nastavení jako duplex režim, wake-on-LAN můžete spravovat nástrojem `ethtool` (nemusí být součástí instalace).

2.4. Routovací tabulky

Routovací (nebo též směrovací) tabulky můžete vypsát a modifikovat příkazem `route`.

Důležité

Změny nastavené příkazem `route` nepřežije restart. Pro trvalé změny je třeba provést nastavení routovacích tabulek v [Netplan](#) souborech.

Nastavení výchozí brány

Výchozí brána (default gateway) se nenastavuje na rozhraní, ale v routovacích tabulkách jádra. Používáme proto příkaz `route`.

```
sudo route add default gw 192.168.123.1 enp0s8
```

Výpis routovací tabulky

Pro ověření můžete vypsát tabulku pomocí `route -n`:

```
$ route -n
Kernel IP routing table
Destination      Gateway          Genmask         Flags Metric Ref    Use Iface
0.0.0.0          192.168.88.1    0.0.0.0        UG    0      0      0 enp0s8
0.0.0.0          192.168.88.1    0.0.0.0        UG    100    0      0 enp0s3
192.168.88.0    0.0.0.0         255.255.255.0  U     0      0      0 enp0s8
192.168.88.0    0.0.0.0         255.255.255.0  U     0      0      0 enp0s3
192.168.88.1    0.0.0.0         255.255.255.255 UH    100    0      0 enp0s3
```

Tip

Routovací tabulku dokáže vypsát také `netstat -r` (viz [netstat](#)).

2.5. Hostname (jméno počítače)

Důležité

Změna přes `hostname` je opět platná dokud nerestartujete počítač. Pro trvalou změnu napište jméno do souboru `/etc/hostname`, který je čten startovacími skripty.

Poznámka

Jméno počítače nemá prakticky žádný vliv. Pokud nemáte centrální správu jmen (DNS) ostatní stanice vámi zvolený `hostname` neznají a mohou se na vás odkazovat pouze číselnou IP adresou.

Zjištění aktuálního jména počítače:

```
$ hostname
tristar
```

Nastavení nového jména počítače:

```
$ sudo hostname bomber
$ hostname
bomber
```

2.6. Soubory `/etc/hosts` a `/etc/services`

Soubor `/etc/hosts` je textovým souborem do kterého se Linux podívá jako prvního, jestliže má přeložit (resolve) jmenný název (hostname) na IP adresu (např. `vacademy.cz` na `210.102.2.189`). IP mohou být jak místní, tak platné z internetu.

Protože tento soubor se prohledává ještě před dotazem na nastavený DNS server je to vhodné místo pro „zfalšování“ adresy hostname serveru na kterém má běžet aplikace, kterou ještě není hotová ap.. Všechny odkazy a dotazy na např. `www.mujs-server.cz`, tak můžete přesměrovat na `127.0.0.1` (místní počítač).

IP adresa je od DNS jména nebo jmen oddělena jedním tabelátorem. Na jednom řádku můžete pro stejnou IP vypsát více jmen.

Příklad `/etc/hosts`

```
127.0.0.1      localhost nb-mujnb www.vacademy.cz.local
192.168.0.100  fileserver

# The following lines are desirable for IPv6 capable hosts
::1          ip6-localhost ip6-loopback
fe00::0      ip6-localnet
ff00::0      ip6-mcastprefix
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters
```

`/etc/services` má podobný účel, ale slouží k „překladi“ služeb (protokolů) na čísla portů. Používá ho řada programů, aby zobrazovala např. místo `22` symbolický název „ssh“.

Příklad `/etc/services`

```
tcpmux        1/tcp          # TCP port service multiplexer
echo          7/tcp
echo          7/udp
discard       9/tcp          sink null
```

```
discard      9/udp      sink null
systat       11/tcp     users
daytime      13/tcp
daytime      13/udp
netstat      15/tcp
qotd         17/tcp     quote
msp          18/tcp     # message send protocol
msp          18/udp
chargen      19/tcp     ttytst source
chargen      19/udp     ttytst source
ftp-data     20/tcp
ftp          21/tcp
fsp          21/udp     fspd
ssh          22/tcp     # SSH Remote Login Protocol
ssh          22/udp
```

Více informací o souboru naleznete v `man 5 services`.

3. Síťové nástroje

3.1. Diagnostika sítí

3.1.1. ifconfig

Ifconfig je základní program pro diagnostiku. Bez parametrů nebo se jménem adaptéru vypíše informace o síťových rozhraních.

```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr e0:db:55:a8:d7:0a
          inet addr:192.168.123.102  Bcast:192.168.123.255  Mask:255.255.255.0
          inet6 addr: fe80::e2db:55ff:fea8:d70a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:436 errors:0 dropped:0 overruns:0 frame:0
          TX packets:518 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:363793 (363.7 KB)  TX bytes:82767 (82.7 KB)
          Interrupt:17
```

Upozornění

Dejte si pozor, že se mohou lišit výsledky ifconfigu bez sudo a se sudo.

Další použití naleznete v kapitole [Konfigurace sítí v Linuxu](#).

3.1.2. lshw -class network

Pro nás již známý program lshw s parametrem `-class network` vypíše detailnější údaje o síťových rozhraních jako chipset, sběrnici ap.

3.1.3. ping / ping6

Elementární program ping (resp. ping6 pro IPv6) pro zjišťování síťového stavu cíle. Využívá ICMP výzvy nad protokolem UDP. Není vhodný proto náročnější služby fungující nad TCP jako HTTP, SSH ap.

Na rozdíl od Windows ping provádí ICMP ECHO_REQUEST dokud není zastaven pomocí `Ctrl-C`.

Vyžaduje root oprávnění, ale má nastaven SUID, takže ho mohou spouštět všichni uživatelé.

```
$ ping vacademy.cz
PING vacademy.cz (95.85.61.132) 56(84) bytes of data.
64 bytes from 95.85.61.132: icmp_seq=1 ttl=53 time=24.5 ms
64 bytes from 95.85.61.132: icmp_seq=2 ttl=53 time=23.1 ms
64 bytes from 95.85.61.132: icmp_seq=3 ttl=53 time=23.1 ms
^C
--- vacademy.cz ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 23.101/23.607/24.524/0.672 ms
```

3.1.4. tracepath / tracepath6

Ke zjištění trasy k cíli přes všechny brány slouží dvoje tracepath / tracepath6 (pro IPv6). Využívá ICMP nad UDP a proto se nehodí k diagnostice hostitelů se zákazem UDP.

```
$ tracepath6 3ffe:2400:0:109::2
1?: [LOCALHOST] pmtu 1500
1: dust.inr.ac.com 0.411ms
2: dust.inr.ac.com asymm 1 0.390ms pmtu 1480
2: 3ffe:2400:0:109::2 463.514ms reached
Resume: pmtu 1480 hops 2 back 2
```

3.1.5. traceroute / traceroute6

Pokročilejší varianta sledování trasy, než tracepath / tracepath6, která dovede posílat nejen UDP, ale i TCP SYN.

Některé verze Ubuntu mají pouze traceroute6 pro IPv6. Starší verze Ubuntu neobsahují tento program vůbec. Variantu pro IPv4 si případně doinstalujte pomocí sudo apt-get install traceroute.

Porovnání tracepath a traceroute

	tracepath	traceroute
nainstalován defaultně	ano	většinou ne
vyžaduje root oprávnění	ne	ne pro UDP
využívá protokoly	UDP	ICMP ECHO, UDP nebo TCP SYN

Důležité parametry:

- `-U`, `--udp` pro zjišťování pomocí UDP paketů jako `tracert / tracert6` (nevyžadují root)
- `-I`, `--icmp` pro zjišťování pomocí ICMP ECHO (vyžadují root v závislosti na verzi jádra)
- `-T`, `--tcp` nebo `tcptraceroute` pro zjišťování pomocí TCP SYN paketů (vyžadují root)
- `-p`, `--port` upřesní port pro TCP probes (defaultně 80)

Bez parametrů použije UDP a tedy nevyžaduje root.

```
$ sudo traceroute --tcp virtage.com
traceroute to virtage.com (55.15.62.72), 30 hops max, 60 byte packets
 1  12.68.323.8 (192.168.123.1)  0.403 ms  0.411 ms  0.471 ms
 2  20.18.60.1 (10.5.60.1)  32.804 ms  33.770 ms  33.760 ms
 3  20.18.214.91 (10.5.224.41)  33.751 ms  34.319 ms  34.297 ms
 4  20.18.214.19 (10.5.224.29)  34.667 ms  35.213 ms  37.564 ms
 5  63-188-70-118.foo.com (118.45.28.16)  38.136 ms  38.117 ms  38.121 ms
 6  * * *
 9  55.15.62.72 (55.15.62.72)  56.568 ms  54.614 ms  54.652 ms
```

3.1.6. netstat

Netstat je program pro získání informací o síťových spojeních, routovací tabulky, maškarád a statistik. Nejčastější využití pro výpis spojení zobrazuje TCP, UDP spoje i Unixové doménové sokety.

Poznámka

Pro některé parametry např. `-p` je třeba třeba root oprávnění, proto je vhodnější se naučit používat `netstat` se `sudo`.

Představíme nejdůležitější parametry:

- `-a`, `--all` – naslouchající i nenaslouchající sokety
- `-t`, `--tcp` a `-u`, `--udp` – jen TCP nebo UDP spojení
- `--route`, `-r` – zobraz routovací tabulku

- `-p, --program` – zobraz PID a jméno programu využívající socket
- `-e, --program` – pro proces využívající socket zobraz i jeho uživatele
- `-n, --numeric` – zobrazuj všechny údaje číselně místo jména (rychlejší, IP místo hostname, port místo protokolu, UID místo username ap.)

Parametry můžete samozřejmě kombinovat. Nejčastější parametry jsou `netstat -etupan` (číselné vyjádření) nebo `-etupa` (jména místo čísel).

Příklad výstupu `netstat -etupan` (zkráceno)

```
$ sudo netstat -etupan
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 127.0.0.1:5984         0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:42820       0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:25162         0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:139           0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:34860       0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:63342       0.0.0.0:*               LISTEN
tcp      0      0 127.0.1.1:53          0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:631         0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:37565         0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:8445          0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:445           0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:6942        0.0.0.0:*               LISTEN
tcp      1      0 192.168.123.104:45708 162.213.33.242:80      CLOSE_WAIT
tcp      1      0 192.168.123.104:46320 162.213.33.242:80      CLOSE_WAIT
tcp      1      0 192.168.123.104:42718 162.213.33.241:80      CLOSE_WAIT
tcp      1      0 192.168.123.104:46141 162.213.33.242:80      CLOSE_WAIT
tcp      1      0 192.168.123.104:46335 162.213.33.242:80      CLOSE_WAIT
tcp      1      0 192.168.123.104:45794 162.213.33.242:80      CLOSE_WAIT
tcp      0      0 192.168.123.102:53492 173.194.116.225:443    ESTABLISHED
...
```

Výpis routovací tabulky jádra

```
$ netstat -rn
Kernel IP routing table
Destination      Gateway          Genmask         Flags   MSS Window  irtt Iface
0.0.0.0          192.168.123.1  0.0.0.0         UG      0 0        0 eth0
192.168.123.0   0.0.0.0         255.255.255.0  U       0 0        0 eth0
```

Stejný výpis získáte také pomocí `route -n` (viz [Routovací tabulky](#)).

*Výpis jen aktivních spojení (Aktivní spojení se nachází ve stavu „ESTABLISHED“.
Můžeme si proto grepem:)*

```
netstat -etupan | grep ESTA
```

3.1.7. nslookup a dig

Oba programy slouží k dotazování a diagnostice DNS systému.

Nslookup je základní nástroj pro jednoduché dotazy:

```
$ nslookup virtage.cz
Server:          127.0.1.1
Address:         127.0.1.1#53

Non-authoritative answer:
Name:            virtage.cz
Address: 95.85.61.132
```

Dig je daleko pokročilejší a umožňuje např. dotaz jen na určité DNS záznamy - např. MX:

```
$ dig mx virtage.cz

; <<>> DiG 9.9.5-3ubuntu0.5-Ubuntu <<>> mx virtage.cz
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 54955
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 4, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags;; udp: 1280
;; QUESTION SECTION:
virtage.cz.                IN      MX

;; ANSWER SECTION:
virtage.cz.                3600   IN      MX      10 alt3.aspmx.l.google.com.
virtage.cz.                3600   IN      MX      1 aspmx.l.google.com.
virtage.cz.                3600   IN      MX      10 alt4.aspmx.l.google.com.
virtage.cz.                3600   IN      MX      5 alt2.aspmx.l.google.com.
virtage.cz.                3600   IN      MX      5 alt1.aspmx.l.google.com.

;; AUTHORITY SECTION:
virtage.cz.                3600   IN      NS      ns.wedos.com.
virtage.cz.                3600   IN      NS      ns.wedos.eu.
virtage.cz.                3600   IN      NS      ns.wedos.cz.
virtage.cz.                3600   IN      NS      ns.wedos.net.

;; Query time: 132 msec
;; SERVER: 127.0.1.1#53(127.0.1.1)
;; WHEN: Mon Dec 14 16:40:54 CET 2015
;; MSG SIZE rcvd: 254
```

3.1.7.1. Analýza a ladění

3.1.8. telnet

Telnet je klient pro stejnojmenný protokol sloužící pro komunikaci se vzdáleným hostitelem. Nepoužívá žádné zabezpečení a proto je již dlouho nahrazován protokolem [SSH](#). Nicméně pro jeho jednoduchost je telnet často využíván pro „simulaci“ klienta jiných textově orientovaných protokolů jako HTTP, SMTP ap.

Syntaxe pro připojení je:

```
telnet host port
```

Zkusíme si „předstírat“ činnost webového browseru:

```
$ telnet vacademy.cz 80
Trying 95.85.61.132...
Connected to vacademy.cz.
Escape character is '^]'.
```

V tento moment se zobrazí výzva a můžete začít psát. Např. HTTP povel pro získání hlavní stránky (/) je:

```
GET / HTTP/1.1
Host: vacademy.cz
```

a odešlete 2x stiskem Enter. Uvidíte HTTP hlavičky odpovědi a zdrojový kód HTML stránky (říká se tam, že máte použít místo http adresu <https://vacademy.cz>):

```
HTTP/1.1 301 Moved Permanently
Server: nginx
Date: Wed, 16 May 2018 16:20:29 GMT
Content-Type: text/html
Content-Length: 178
Connection: keep-alive
Location: https://vacademy.cz/
X-Rosti: lb.rosti.cz

<html>
<head><title>301 Moved Permanently</title></head>
<body bgcolor="white">
<center><h1>301 Moved Permanently</h1></center>
<hr><center>nginx</center>
</body>
</html>
```

3.1.9. nmap

Nmap je program pro skenování portů a běžících služeb. Umí odhalit nejen otevřené porty (např. pro testování funkčnosti firewallu), ale v některých případech i druh zařízení a jeho software, či živé síťové zařízení, které má ale zakázaný ping. Nmap je pro správce sítě společně s pingem nepostradatelný nástroj.

Nmap (a případně GUI Zenmap) nainstalujete standardním způsobem z repozitářů Ubuntu:

```
sudo apt-get install nmap
sudo apt-get install zenmap
```

Použití vypadá:

```
nmap [volby] <cilovy-hostitel>
```

Hostitel může být specifikován IP adresou nebo jménem:

```
nmap 192.168.123.47
```

ale můžete určit více cílů pomocí intervalů např. všechny IP 192.168.1.1 až 192.168.123.254:

```
nmap 192.168.1-123.1-254
```

Mezi užitečné volby patří zejm.

- **-PN** – ověření hosta i když blokuje ICMP ping
- **-p 8081**, **-p 8081,2900**, **-p 8081-8090** – jen port 8081, porty 8081 a 2900, porty 8081 až 8090
- **-F** – rychlý sken (méně služeb, než ve výchozím skenu)
- **-A** – pokusí se zjistit OS a jeho verzi
- **-v** – podrobný výpis

Příklad použití nmapu

```
$ nmap google.com
```

```
Starting Nmap 6.40 ( http://nmap.org ) at 2015-12-17 11:25 CET
Nmap scan report for google.com (173.194.122.3)
```

```
Host is up (0.0085s latency).
Other addresses for google.com (not scanned): 173.194.122.14 173.194.122.4 173.
rDNS record for 173.194.122.3: prg02s12-in-f3.1e100.net

Not shown: 998 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https

Nmap done: 1 IP address (1 host up) scanned in 4.72 seconds
```

3.1.10. tcpdump

Důležité

Pro odposlouchávání tcpdumpem nebo později probíraným [Wiresharkem](#) je třeba, aby síťové rozhraní podporovalo a mělo zapnutý tzv. [promiskuitní režim](#).

Tcpdump je základní, ale velmi dobře použitelný *sniffer* (*program pro odposlouchávání*). V Ubuntu je předinstalovaný, dokáže odposlouchávat příchozí i odchozí pakety na síťových rozhraních, filtrovat je jen na určitého hostitele, port ap. Provoz zobrazuje rovnou na obrazovku nebo ukládá do souboru PCAP pro pozdější analýzu.

Tcpdump má velmi mnoho voleb. Představíme si jen několik důležitých parametrů a příkladů. Pro odposlouchávání je třeba spouštět tcpdump s právy superuživatele.

Sleduj jen určité rozhraní

Pomocí volby `-i` bude vypisovat přijaté a odeslané pakety jen na určeném síťovém rozhraní. Adresy se pokusí překládat na jména.

```
sudo tcpdump -i eth0
```

Nepřekládej jména

Překlad na jména je často zbytečný a vždy zdlouhavý. Parametrem `-n` se vypíší jen IP adresy.

```
sudo tcpdump -i eth0 -n
```

Omezení na IP adresu, port, protokol

Také sledovat veškerý provoz je často nežádoucí. Filtrovat můžeme oba směry komunikace na IP adresu pomocí `host <ip>`. Pokud uvedeme `dst` bude omezení platit pro zvolenou IP jako cíl, nebo `src` jako zdroj.

```
sudo tcpdump -i eth0 -n dst host 192.168.123.150
```

Podobně lze omezit provoz i na konkrétní port pomocí `port <port>`.

```
sudo tcpdump -i eth0 -n port 8080
```

Nebo na protokol TCP, UDP nebo ICMP pomocí stejnojmenných parametrů.

```
sudo tcpdump -i eth0 -n tcp
sudo tcpdump -i eth0 -n udp
sudo tcpdump -i eth0 -n icmp
```

Podmínky lze kombinovat s omezením na IP nebo použít samostatně.

```
sudo tcpdump -i eth0 -n src host 192.168.123.150 port 8080
sudo tcpdump -i eth0 -n src host 192.168.123.150 port 8080 tcp
```

Uložení do PCAP souboru

Pokud je i tak zobrazovaný provoz příliš veliký nebo chcete provést analýzu později, je tu možnost uložit soubor do speciální PCAP formátu pomocí `-w <soubor.pcap>`, který umí číst např. dále zmiňovaný Wireshark.

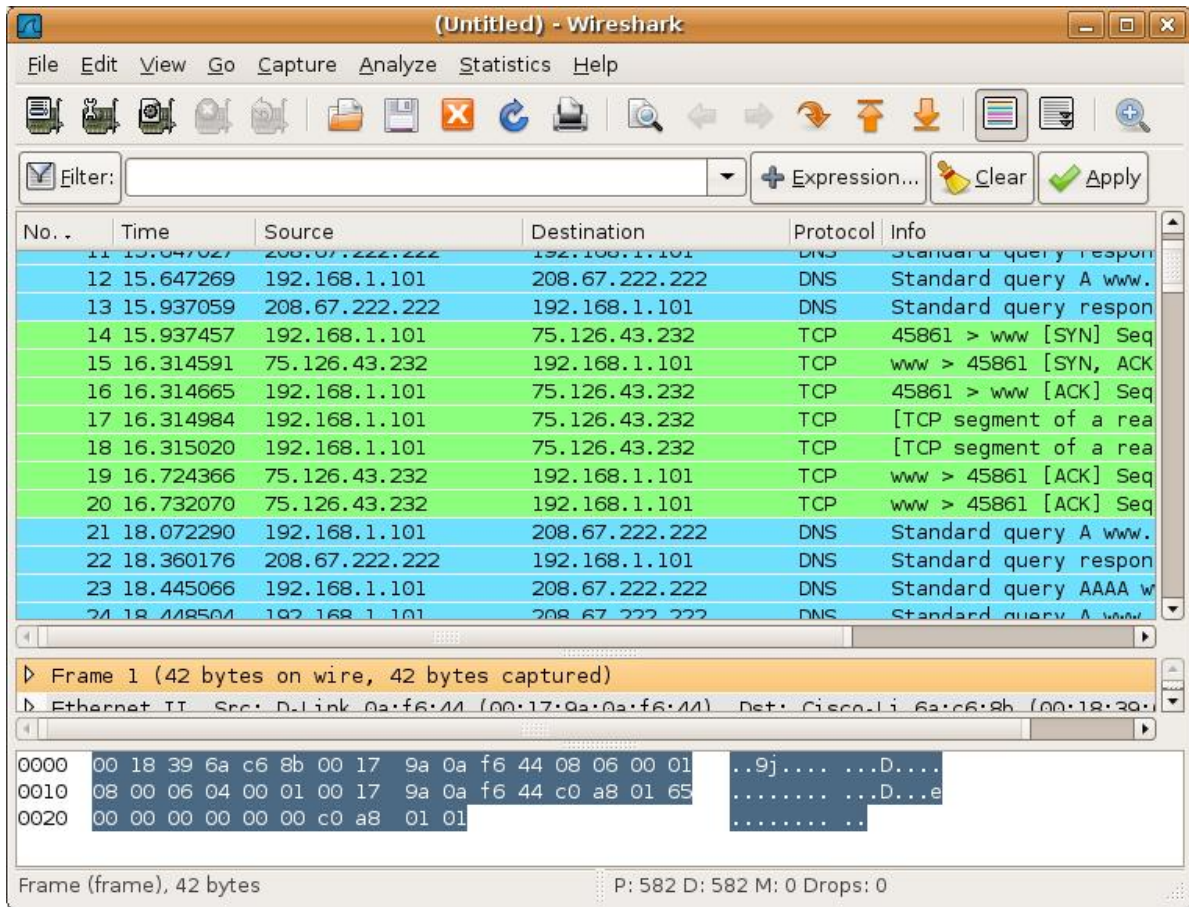
Pozor na to, že standardně se omezuje délka ukládaných paketů na 65 535 bajtů a proto je vhodné ještě použít `-s 0` pro zrušení tohoto limitu.

```
sudo tcpdump -i eth0 -n -w soubor.pcap -s0
```

3.1.11. Wireshark

Wireshark (dříve Ethereal) je grafický sniffer. Umí sám zachytávat, ale často je používán k tzv. post-mortem analýze, kdy zachytíme provoz na problémové negrafické

stanici do .pcap souboru a na stolním počítači s Wiresharkem následně v klidu analyzujeme.



Wireshark vyniká ve snadném ovládní, hledání v zachyceném provozu a je nabízen zdarma pro všechny platformy.

3.1.11.1. Přenosy po síti

Představíme si programy a protokoly používané ve světě Linux k zabezpečenému kopírování souborů po vnitřní síti nebo internetu.

3.1.12. wget

Wget a [curl](http://curl.haxx.se/) patří mezi velmi populární download managery z příkazové řádky na textových stanicích nebo ze skriptů. Dokáží stahovat z HTTP, HTTPS i FTP.

Např. stažení zdrojového kódu této knihy jako stejnojmenný soubor v aktuální složce pomocí wgetu:

```
$ wget https://bitbucket.org/virtage/book-usrv2-cz/get/549b3808a0ec.zip
```

3.1.13. curl

Curl (někdy čteno jako „kár!“) je mocnější, než wget - podporuje řadu dalších protokolů jako IMAP(s), LDAP(s), POP3(s), SCP, SFTP, SMTP a dovede i soubory nahrávat. Pro své bohaté možnosti se curl používá i pro testování RESTful webových služeb.

Bez parametrů vypisuje stahovaný soubor na STDOUT (terminál). S volbou -O uloží soubor pod jménem jako na vzdáleném serveru (tj. to co obvykle chcete, když stahujete):

```
$ curl -O https://bitbucket.org/virtage/book-usrv2-cz/get/549b3808a0ec.zip
```

Stažený soubor bude 549b3808a0ec.zip. Chcete-li ale vybrat pro stažený soubor jiné jméno použijte -o <nazev>:

```
$ curl -o prirucka_usrv2.zip https://bitbucket.org/virtage/book-usrv2-cz/get/54
```

Curl zvládá ohromné množství protokolů. Např. užitečné do skriptu může být poslání emailu:

```
$ curl --mail-from blah@virtage.com --mail-rcpt foo@virtage.com smtp://somemail
```

HTTP POST požadavek na WWW-Basic zabezpečený cíl s tělem jako application/x-www-form-urlencoded (HTML formulář):

```
$ curl -X POST --user "user:pass" -d "instance_id=i-d77cd0ac" -d "name=marioste
```

Možnosti a tomu odpovídající počet parametrů je obrovský, ale úsilí při seznámení se s curl stojí za to.

3.1.14. scp

Program scp (secure copy) používá ke kopírování [SSH](#). Můžeme ho použít pro jednorázový zabezpečený přesun složky nebo souborů z/do vzdáleného počítače. Dokonce umí přenášet soubory i mezi dvěma vzdálenými servery bez mezikroku.

Jeho obecná syntaxe vypadá


```
scp [-12346BCpqrV] [-c cipher] [-F ssh_config] [-i identity_file]
[-l limit] [-o ssh_option] [-P port] [-S program]
[[user@]host1:]file1... [[user@]host2:]file2
```

Např. ke stažení na lokál a upload na vzdálený server jen přehazujeme první a druhé místo:

```
$ scp joe@magnolia:/var/log/cups/error.log ~/tmp/
$ scp ~/tmp/error.log joe@magnolia:/var/log/cups/
```

Mezi užitečné volby patří:

- **-P** určující port, není-li SSH na standardním 22 (pozor nepoužívá **-p** jako ssh klient)
- **-i** použití jiného SSH klíče, než našeho hlavního v `~/ .ssh/`
- **-r** kopírování složek (bez této volby vypíše při pokusu o kopírování složky scp chybu!)

3.1.15. rsync

Další důležitým programem každého správce Linuxu pro kopírování a přesuny souborů bude rsync. Ten se od ostatních liší tím, že dovede přenášet jen rozdíly. Když změníte ve 100 MiB souboru 1 byte budete se cp, scp a další přenášet znovu celých 100 MiB. S rsync jen pár bajtů - změněný bajt a několik bajtů jako režii.

Mezi další úžasné vlastnosti rsync patří, že dovede přenášet lokálně i po síti, pro ještě větší ušetření kapacity dokáže přenosy komprimovat a přenos po síti lze zabezpečit přes [SSH](#).

Základní syntaxe je

```
$ rsync [volby] <zdroj> <cíl>
```

Zdroj nebo cíl může být soubor nebo složka. Mezi důležité volby patří

- **-a** – „archivační režim“ - zachovej u kopírovaných souborů oprávnění, vlastnictví, přenes symbolické odkazy ap.
- **-v**, **-vv** nebo **-vvv** – vypisuj prováděnou činnost od nejdůležitějších (jedno „v“) po velmi detailní (tři „v“)
- **--progress** – zobrazí postup činnosti
- **--exclude** – vynech soubory/složky vyhovující masce. Např. `--exclude=*.bak` nebude přenášet BAK soubory.

- `--delete` – z cílové složky odstraní soubory, které již ve zdrojové složce neexistují. Vhodné pro „synchronizaci“ mezi dvě místy. Při používání této volby buďte opatrní!
- `--dry-run` – běh „na sucho“, tj. jen předstírej, ale neprováděj žádnou činnost. Vhodné pro odzkoušení nebezpečných voleb jako `-delete` před během na ostro.
- `-z`, `--compress` – přenos komprimuj pro ušetření kapacity pásma

Volby můžete kombinovat: např. `-av` je rovnocenné k `-a -v`.

Rsync se chová odlišně, pokud je ve zdrojové cestě ukončující lomítko (trailing slash) nebo není. (Koncová závorka se pro cílovou cestu nerozlišuje.) Jinými slovy jiné chování způsobí bez a s uvedením „/“ u zdroje.

```
$ rsync /home/libor/Documents /mnt/backup # bez ukončujícího lomítka
$ rsync /home/libor/Documents/ /mnt/backup # s ukončujícím lomítkem
```

Při vynechání „/“ rsync nejdříve vytvoří zdrojovou složku v cíli a až do ní nakopíruje obsah.

```
$ rsync /home/libor/Documents /mnt/backup
$ tree /mnt/backup
Documents/
- accounting/
  - home/
  - company1/
  - 2012.ods
  - 2013.ods
- photos/
  - family/
  - pets/
```

Při uvedení „/“ kopíruje podsložky a soubory zdroje rovnou do cíle.

```
$ rsync /home/libor/Documents/ /mnt/backup
$ tree /mnt/backup
- accounting/
- home/
- company1/
  - 2012.ods
  - 2013.ods
- photos/
- family/
- pets/
```

Rsync obsahuje i serverovou část démona rsyncd, který je sice rychlý, ale přenosy nešifruje. Proto se v praxi téměř výlučně používá přenos přes [SSH](#), kdy přenosu na vzdálený stroj vypadá např.:

```
$ rsync -av -e ssh --delete /home/libor/Documents user@remote.host:/mnt/backup
```

Nová je tu volba `-e` s hodnotou `ssh`. Jiná je syntaxe cíle. Samozřejmě můžeme i kopírovat zpět (obnovit) ze vzdáleného serveru na lokální prohozením cíle a zdroje:

```
$ rsync -av -e ssh user@remote.host:/mnt/backup /home/libor/Documents
```

Pokud SSH nepoužívá standardní port 22, ale např. 2810 příkaz musí vypadat:

```
$ rsync -av -e "ssh -p 2810" /home/libor/Documents user@remote.host:/mnt/backup
```

4. Síťová bezpečnost

4.1. Firewall

Potřebuji firewall?

Otázka je to celkem složitá a také by měla být lépe položená, ale jednoduchá odpověď zní - pokud neprovozujete server, tak ne.

Nyní se na tuto problematiku podíváme trochu šířeji. Jak se dočtete níže, váš Linux obsahuje vestavěný firewall iptables, který se automaticky stará o veškerý síťový provoz a podle přednastavených pravidel (respektive podle vámi předepsaných pravidel, pokud se je rozhodnete změnit) jej také spravuje a rozhoduje, co s jednotlivými pakety udělá.

Ve výchozím nastavení je iptables nastaveno velmi volně a síťový provoz prakticky nijak neomezuje. Ač to nemusí být zřejmé, vaše bezpečnost není nijak výrazně ohrožena. Většina domácích počítačů je firewallem chráněna aniž o tom jejich uživatelé vědí (firewall se totiž nachází v různých zařízeních jako jsou switche, routery a další) a navíc je většina „zneužitelných“ služeb a nástrojů (jako SSH, webový server a další) vypnuta nebo není v základní instalaci Ubuntu vůbec zahrnuta.

4.1.1. UFW

UFW, neboli Uncomplicated FireWall, je pro Ubuntu výchozí a velmi jednoduché řešení, které však pokryje takřka 100% požadavků na firewall. Jak název napovídá program se snaží o nekomplikovaný způsob nastavování firewallu a sami zjistíte, že se mu to daří. Ve skutečnosti se jedná o nadstavbu nad poměrně složitým iptables firewallem.

Instalace není nutná - UFW je součástí Ubuntu desktop i server edicí.

4.1.1.1. Stavové informace

Nejprve zjistěte stav firewallu:

```
$ sudo ufw status
```

Ten je po instalaci Ubuntu vypnutý. Můžeme ho povolit i dočasně zakázat jednoduchými příkazy:

```
$ sudo ufw enable
$ sudo ufw disable
```

4.1.1.2. Pravidla

Po zapnutí použije defaultní sadu pravidel, která bude vyhovovat většině domácích uživatelů - povolen ping, zakázána všechna příchozí spojení s výjimkou portů 21 (ftp), 554 (rftp - domácí streaming), 7070 (RealAudio streaming), povolena všechny odchozí spojení.

Teď i později můžete použít tyto příkazy pro více informací:

```
$ sudo ufw status verbose      # detailní stav firewallu
$ sudo ufw status raw         # výpis ve formátu iptables
$ sudo ufw status numbered    # pravidla firewallu
```

Pravidla povolení i zákazu mají stejnou obecnou syntaxi:

```
$ sudo ufw (allow|deny) <port>/[protocol]
```

Pokročilejší podoba pro určení zdrojového hostitele a cílového portu:

```
$ sudo ufw (allow|deny) from <target> to any port <port-number>
```

Příklady povolujících pravidel:

```
# Povolení portu 80 (viz /etc/services)
$ sudo ufw allow http
$ sudo ufw allow 3306

# Povolení UDP provozu na portu 2807
$ sudo ufw allow 2807/udp

# Povol příchozí provoz od dané IP
$ sudo ufw allow from 207.46.232.182

# Můžete použít i netmask sítě
$ sudo ufw allow from 192.168.1.0/24

# Povolení zdrojové IP přístup jen na port 22
$ sudo ufw allow from 192.168.0.4 to any port 22
```

Příklady pravidel zákazu:

```
$ sudo ufw deny ssh
$ sudo ufw deny 3306
$ sudo ufw deny 2807/udp
$ sudo ufw deny from 192.168.0.1
$ sudo ufw deny from 192.168.0.1 to any port 22
```

Pro smazání pravidla musíte nejprve zjistit její číslo v ufw status numbered a potom

```
$ sudo ufw delete <čísloPravidla>
```

Upozornění

Pořadí vyhodnocování pravidel

Při hledání důvodu, proč provoz neprochází firewallem se často zapomíná, že jakmile je nalezeno vyhovující pravidlo, tak se další již nevyhodnocují. Jinými slovy **nejprve vytvořte specifitější pravidlo a teprve potom obecnější**.

4.1.1.3. Další dovednosti

Defaultně UFW loguje svůj provoz do souborů `/var/log/ufw*`. Logování můžete kdykoli vypnout nebo zapnout:

```
$ sudo ufw logging [off|on]
```

Kdyby se chtěli vrátit do výchozího stavu můžete UFW resetovat pomocí

```
$ sudo ufw reset
```

4.1.2. Iptables

Nástroj pro nastavování pravidel firewallu v jádře. Pro složitost se nebudeme zabývat. Nahradil zastaralý ipchains.

UFW je nadstavbou nad iptables, která skrývá složitost iptables úkolů a spolehlivě postačí na většinu požadavků na nastavení firewallu.

4.1.3. GUI nadstavby

- GFW - nadstavba nad UFW

- Fwbuilder
- Shorewall
- Firestarter
- Lokkit

4.2. TCP wrappers

- `/etc/hosts.allow` a `hosts.deny`
- aplikace musí přístup konzultovat s TCP wrapper vrstvou, což už příliš mnoho aplikací nedělá
- iptables/ufw jsou mnohem efektivnějším řešením fungujícím vždy

Je aplikace kompatibilní ap.? <https://ubuntu-tutorials.com/2007/09/02/network-security-with-tcpwrappers-hostsallow-and-hostsdeny/>

4.3. AppArmor

- rozšíření jádra pro řízení přístupu
- umožňuje definovat oprávnění k provedení určité operace na úrovni jednotlivých procesů v závislosti na umístění spustitelného souboru tím, že na kritická místa jádra umísťuje volání svých kontrolních rutin. Tím dochází ke zvýšení režie systému, avšak je možné zabránit programu, aby provedl potenciálně nebezpečnou akci, která může vést k narušení bezpečnosti (včetně elevace oprávnění).
- podobný SELinuxu v Red Hat

5. Vzdálený přístup

5.1. SSH

Sada protokolů SSH, konkrétně v Ubuntu implementace [OpenSSH](#), je nejběžnější a jeden z nejbezpečnějších způsobů přístupu na vzdálený Linuxový server. SSH vytváří mezi vámi a vzdáleným počítačem šifrované spojení vhodné nejen pro vzdálený přístup, ale i přenos souborů.

Na rozdíl od telnetu nebo rcp je celé spojení (včetně zasílání hesla) šifrováno a proto bezpečné i při přístupu přes internet.

OpenSSH je rozděleno na démona `sshd` běžící na serveru a klienta, kterým bývá nejčastěji `ssh` pro vzdálený přístup a `scp` pro zabezpečený přenos souborů.

OpenSSH podporuje ověřování uživatelů (autentizaci) pomocí více mechanismů jako jméno-heslo, veřejný klíč nebo Kerberos. Nejrozšířenější je autentizace pomocí veřejného klíče, kdy se pro uživatele vygeneruje RSA nebo DSA klíč s veřejnou a soukromou částí.

5.1.1. Instalace

Instalace OpenSSH je velmi jednoduchá a spočívá jen v instalaci balíčku `openssh-server`, případně i `openssh-client`. OpenSSH je rovněž možné vybrat rovnou během instalace Ubuntu Server.

```
$ sudo apt-get install openssh-client  
$ sudo apt-get install openssh-server
```

5.1.2. Připojení SSH klientem

Klientský program pro připojení na SSH server se jmenuje prostě `ssh`. Pro přihlášení ke vzdálenému terminálu serveru použijte příkaz:

```
$ ssh [-p 1084] [<user>@]<server>
```

kde můžete port (`-p`) můžete vynechat, pokud SSH běží na standardním portu 22. Rovněž vynecháte-li uživatele, pak se SSH pokusí přihlásit jako aktuální uživatel.

Jiný klíč

Bude-li vzdálený server požadovat ověření veřejným klíčem, bude se ssh klient hledat váš soukromý klíč v souboru `id_dsa` nebo `id_rsa` ve `.ssh/` ve vaší domovské složce.

Chcete-li poslat jiný soukromý klíč použijte volbu `-i`:

```
$ ssh -i /path/to/different/private-key user@server-name-or-ip
```

Když něco nefunguje

Pokud něco nefunguje hodí se volba `-v`, kterou zobrazíte ladící zprávy užitečné při odhalování, co se vlastně děje:

```
$ ssh -v [-p 1084] [<další volby>] [<user>@]<server>
```

5.1.3. Konfigurace SSH serveru

Hlavní konfigurační OpenSSH serveru je v souboru `/etc/ssh/sshd_config`. Úplnou dokumentaci najdete v manuálové stránce zadáním `man sshd_config`.

Upozornění

Když provádíte změny SSH serveru přes SSH můžete přerušit aktuální spojení nebo až při příštím pokusu o připojení zjistit, že se na server už nedostanete. Před jakoukoli změnou SSH serveru je proto dobré se ujistit, že máte k serveru jiný přístup.

5.1.3.1. Změna portu SSH

Standardní port SSH je 22, ale často se tento port mění z důvodu bezpečnosti na jiný. K tomu stačí změnit hodnotu proměnné `Port` v konfiguračním souboru a pak provést restart SSH.

Ať už standardní port 22 nebo jiný nezapomeňte to ho povolit na [firewallu](#).

5.1.3.2. Zákaz přihlašování heslem

Ve standardní konfiguraci je možné se přihlásit zadáním jména-hesla vzdáleného uživatele. To se hodí spíše jen v interní síti. Výhodou je, že nemusíme vytvářet a spravovat certifikáty uživatelů. U serveru viditelného z internetu je bezpečnější autentizaci heslem zakázat a přihlašovat se jen certifikátem.

Nastavení je opět jednoduché - nastavte `PasswordAuthentication` na hodnotu `no` a proveďte restart SSH démona.

Upozornění

Ještě, než vypnete ověření heslem, protože chcete použít jen ověření klíčem, měli byste mít k serveru fyzický přístup nebo již alespoň jednoho uživatele s klíčem přidaného. Po vypnutí hesel totiž již nepůjde použít příkaz `ssh-copy-id` kopírující klíč na server.

5.1.3.3. Zákaz SSH přihlášení roota

Další časté nastavení je zákaz přihlášení uživatele root přes SSH. Root tak bude vždy použit fyzický přístup nebo KVM.

Nastavte klíč `PermitRootLogin` na `no` a restartujte SSH.

5.1.3.4. Restart SSH serveru

Aby se projevil změny je třeba restartovat SSH démona:

```
$ sudo systemctl restart ssh
```

5.1.4. Založení SSH uživatele

SSH nemá vlastní uživatele - SSH pouze ověřuje, že připojující se uživatel odpovídá již existujícímu účtu na serveru. Vytvoření SSH uživatele znamená tedy předně vytvoření účtu v OS.

Pokud je povoleno přihlašování veřejným klíčem, je třeba pro každého uživatele vygenerovat klíč a přidat ho do jeho domovské složky do souboru `~/.ssh/authorized_keys`.

5.1.5. Veřejné klíče

Ověřování založené na veřejných klíčích představuje bezpečnější alternativu k přihlašování heslem. Výhodou je, že uživatel nemusí zadávat svoje heslo a proto se často používá pro přístup ze skriptu a všude jinde, kde není možné zadat heslo.

5.1.5.1. Povolení přihlášení veřejným klíčem

Ze všeho nejdříve je třeba povolit přihlašování veřejným klíčem nastavením `PubkeyAuthentication` `yes`.

5.1.5.2. Vygenerování klíčů

Uživatel na místo hesla používá námi vygenerovanou dvojici veřejného a soukromého klíče. Klíče jsou běžné textové soubory. SSH klíče mohou používat DSA nebo RSA algoritmus, který je o trochu rozšířenější.

Pro samotné vygenerování použijte příkaz

```
$ ssh-keygen -t [rsa|dsa] [-b <početBitů>]
```

např.:

```
$ ssh-keygen -t rsa -b 4096
```

Parametr `-t` určuje algoritmus, `-b` počet bitů (standardně se generují 2048 bitové klíče). Součástí RSA klíčů je i komentář, kam lze vysvětlit účel klíče ap. Defaultně má hodnotu `<aktuální uživatel>@<počítač>`. Komentář ke klíči můžete nastavit na vlastní volbou `-C <komentář>` (pozor velké C!).

Budete postupně dotázáni na jméno a další údaje budoucího uživatele. Nejdůležitější otázka je vybrání *passphrase*, který bude nutné zadat kdykoli při použití klíče. Passphrase může být náhodná skupina např. 8 znaků nebo věta ap. Pokud má být klíč používán ze skriptu samozřejmě nebudete volit žádný passphrase. Naopak pro lidské uživatele si vždy dostatečně bezpečný passphrase vymyslete.

Příkaz vytvoří standardně v `~/ .ssh/`:

- soubor veřejného klíče `id_rsa.pub` (resp. `id_dsa.pub` pro DSA) a
- soubor soukromého klíče `id_rsa` (resp. `id_dsa` pro DSA)

Upozornění

Dejte si velký pozor a nespleťte si veřejný (přípona .pub) a soukromý klíč (bez přípony). Soukromý klíč nedávejte z ruky.

5.1.5.3. Otisk (fingerprint) serveru

Rovněž SSH server má svůj soukromý a veřejný klíč. Dokonce několik dvojic klíčů pro algoritmy jako DSA, ECDSA, RSA a případně další. Soubory jsou pojmenované jako `ssh_host_<algoritmus>_key` (soukromý klíč) a `ssh_host_<algoritmus>_key.pub` (veřejný klíč) najdete uložené ve složce `/etc/ssh/`.

Z veřejného klíče můžeme spočítat tzv. *otisk (fingerprint)*, což je jednoznačná identifikace klíče reprezentovaná jako krátký hexa kód. Ve skutečnosti to je vlastně jen hash (digest) klíče, který vypadá např. jako `13:05:84:80:69:d9:81:c5:2d:17:2e:ce:a6:c0:aa:30`.

K čemu je to dobré? Díky tomu, že žádné dva různé klíče na světě nemají stejný otisk, se můžete takto ubezpečit, že identita serveru nebyla pozměněna.

Otisk libovolného veřejného klíče zjistíte programem `ssh-keygen`. Ten vypíše nejprve počet bitů klíče, samotný otisk a algoritmus. Např. pro ECDSA klíč serveru:

```
$ ssh-keygen -lf /etc/ssh/ssh_host_ecdsa_key.pub
256 13:05:84:80:69:d9:81:c5:2d:17:2e:ce:a6:c0:aa:30 root@stkfactory (ECDSA)
```

Tip

ECDSA otisky vašich SSH serverů byste si měli poznamenat. SSH klient (program `ssh`) se zeptá, jestli je ECDSA otisk správný, když se připojujete poprvé nebo když se změní klíče na serveru (z důvodu přeinstalace, ale i eventuálního napadení).

5.1.5.4. Přidání SSH klíče na server

Veřejný klíč `id_rsa.pub` (resp. `id_dsa.pub` ap.) se pak musí nakopírovat na server, aby mohl SSH démon ověřit identitu porovnáním veřejného klíče a uživatelova soukromého klíče přítomném pouze na klientově počítači.

SSH server dívá do `~/.ssh/authorized_keys` domovské složky uživatele pod kterým se pokoušíme připojit. Když se připojujeme jako `ssh tomas@server`, hledá se v souboru `/home/tomas/.ssh/authorized_keys`.

SSH může pro jednoho uživatele serveru evidovat více veřejných klíčů, ale pokud doržujeme, že uživatelé by neměli sdílet společný účet, pak je většinou v souboru `~/.ssh/authorized_keys` uložen jen jeden klíč.

Běží-li SSH na standardním portu 22 použijte:

```
$ ssh-copy-id <username>@<server>
```

V případě, že SSH naslouchá na jiném portu je syntaxe trochu složitější. Případně můžete zadat jinou cestu k veřejnému klíči. Dohromady např.:

```
$ ssh-copy-id -i <cesta/k/id_rsa.pub> , -p <port> <username>@<server>
```

Při prvním přístupu nás SSH vyzve k potvrzení identity vzdáleného serveru (jeho fingerprintu). Musíme napsat **yes** a stisknout **Enter**. Pokud není autentikace heslem vypnutá, může být třeba zadat heslo vzdáleného uživatele.

Poznámka

Program `ssh-copy-id` ve skutečnosti neudělá nic jiného, než přidání obsahu veřejného klíče do `~/.ssh/authorized_keys`. Proto může být někdy jednodušší dostat klíč na server a na serveru:

```
$ cat mujklic.pub >> ~/.ssh/authorized_keys
```

Nebo otevřít `authorized_keys` v editoru na serveru a řádek přidat přes `copy-paste` schránkou.

5.1.5.5. Odebrání klíče

Upozornění

Pouhé zakázání účtu uživatele v OS (`passwd -l`) nestačí! Pokud je uživatelem veřejný klíč stále v `~/.ssh/authorized_keys`, může se i nadále přihlásit. Podobně zakaz účtu nepřeruší SSH spojení, je-li uživatel právě připojen.

Pro zrušení SSH přístupu přístupu stačí jen přejmenovat složku `~/.ssh/` na název, který SSH server neočekává.

Když jen skončila např. platnost klíče musíme otevřít `~/.ssh/authorized_keys` ručně zde odebrat starý veřejný klíč.

5.1.6. SSH tunneling

SSH je rovněž použit pro vytvoření zabezpečeného tunelu z našeho počítače na vzdálený. Tunel funguje tak, že ssh na lokálním PC vytvoří port, který bude namapován na specifikovaný port vzdáleného serveru. Lokální aplikace se připojují lokálně, ale ve skutečnosti je provoz přesměrován na port vzdáleného serveru. Podobně vzdálené aplikace netuší, že připojení k nim je iniciováno ze jiného, než místního PC.

Tímto lze obejít řadu omezení jako např. u MySQL/MariaDB serveru, který defaultně neumožňuje jiná připojení, než localhostu.

Vytvoření tunelu v SSH má syntaxi:

```
ssh -Ng -L <lokalniPort>:localhost:<vzdalenyPort> [-p <portSSH>] [<user>@]<serv
```

kterou lze číst jako „vytvoř SSH spojení na <server>, a současně naslouchej na portu <lokalniPort> mého počítače, a přesměruj všechna spojení na tento port na <vzdalenyPort> serveru“.

5.2. OpenVPN

[OpenVPN](#) je linuxovým de facto standardem pro vytváření VPN řešení. Je to roky ověřený open source, aktivně vyvíjený, dostupný přímo z Ubuntu repozitářů, flexibil-

ní, spolehlivý a bezpečný VPN software. Používá SSL/TLS typ VPN (jinou možností jsou IPsec VPNky).

5.2.1. OpenVPN Access Server

Je snadnějším VPN řešením postaveným nad OpenVPN, které tvoří OpenVPN server, webové administrační rozhraní a klient. Vytvoření VPN s OpenVPN Access Serverem je mnohem méně složité, než s tradičním OpenVPN. Kromě toho Access Server nabízí i klienty pro Android a iOS.

OpenVPN Access Server je pro první dva uživatele zdarma. Poté vyžaduje zakoupení licence, která však stojí opravdu minimálně a bezstarostnost Access Serveru oproti tradiční OpenVPN určitě vyváží.

The screenshot displays the OpenVPN Access Server web interface. The main content area is titled 'Status Overview' and is divided into two sections: 'Server Status' and 'Active Configuration'. The 'Server Status' section indicates that the server is currently ON and provides a 'Stop the Server' button. The 'Active Configuration' section lists various server parameters: Access Server version (2.0.3), Server Name (10.0.2.15), authentication method (pam), listening IP address (eth0: 172.16.1.11), listening ports (tcp/443, udp/1194), OSI Layer (3 (routing/NAT)), NAT settings, and the node name (s-lljatest). On the right side, the 'At a glance' sidebar shows the server status as 'on', the license as '2 users', and the current number of users as '0'. The left sidebar contains navigation menus for 'Status', 'Configuration', 'User Management', and 'Authentication'.

5.2.1.1. Instalace

1. Z webu OpenVPN si stáhněte a nainstalujte OpenVPN Access Server balíček pro Ubuntu (.deb soubor):

```
$ sudo dpkg -i openvpn-as-<verze>.deb
```

2. Access Server se nainstaloval do do `/usr/local/openvpn_as`. Konfigurační log najdete v

/usr/local/openvpn_as/init.log.

1. Byl založen uživatel `openvpn`. Musíme mu nastavit heslo pomocí `sudo passwd openvpn`.
2. Webová administrace běží na `https:<server>:943/admin`, uživatelská část na `https:<server>:943/`.
3. Pro pozdější manuální rekonfiguraci můžeme použít nástroj `/usr/local/openvpn_as/ovpn-init`.

5.2.1.2. Konfigurace

Přihlaste se do OpenVPN Access Server administrace. Minimálně budeme chtít vytvořit nějaké VPN uživatele v sekci User Permissions. V defaultní konfiguraci musí uživatelé odpovídat uživatelům v OS (PAM autentizace).

The screenshot shows the OpenVPN Access Server web interface. The main heading is "User Permissions". On the left, there is a navigation menu with sections: "Status" (Status Overview, Current Users, Log Reports), "Configuration" (License, Server Network Settings, VPN Mode, VPN Settings, Advanced VPN, Web Server, Client Settings, Failover), and "User Management" (User Permissions, Group Permissions, Revoke Certificates). The main content area includes a search bar for "User Name or users in group by Group Name". Below the search bar is a table with columns: Username, Group, More Settings, Admin, Allow Auto-login, Deny Access, and Delete. Two users are listed: "openvpn" and "libor". Below the table, there are configuration options for "Select IP Addressing" (Use Dynamic, Use Static), "Select addressing method" (Use NAT, Use routing), and "Allow Access To these Networks". There are also checkboxes for "Allow Access From" (all server-side private subnets, all other VPN clients). On the right, there is a "At a glance" summary showing "Server Status: on", "License: 2 users", and "Current Users: 0".

5.2.1.3. OpenVPN Access Server klienti

Všechny uživatele nasměrujte na `https://<server>:943/`, kde se musí přihlásit svým jménem a heslem (heslem pro OS v případě defaultní konfigurace) a poté si vyberou stažení OpenVPN klienta pro jejich operační systém.



Logout

To download OpenVPN Connect for installation on another computer, please choose a platform below:

- [OpenVPN Connect for Windows](#)
- [OpenVPN Connect for Mac OS X](#)
- [OpenVPN Connect for Android](#)
- [OpenVPN Connect for iOS](#)
- [OpenVPN for Linux](#)

If required, connection settings (profiles) can be downloaded for:

- [Anyone at this server \(server-locked profile\)](#)
- [Yourself \(user-locked profile\)](#)

V případě Ubuntu stačí provést `sudo apt-get install openvpn`.

Rovněž si nainstalujte OpenVPN plugin pro GUI správce sítě pro snadné vytvoření OpenVPN spojení:

```
$ sudo apt-get install network-manager-openvpn network-manager-openvpn-gnome
$ sudo service network-manager restart
```

6. Řízení a plánování procesů

6.1. Vznik a zánik procesů

Proces může vyvolat nový proces a ten další proces. Nový proces je kopií svého rodiče - liší se jen PID číslem, ale jinak má stejné prostředí, vstupy a výstupy, proměnné prostředí a prioritu

Tyto stromové vazby zobrazíte programem `pstree`.

Výpis programu `pstree` bez parametrů. Výstup na Ubuntu 16.04. Ve starším Ubuntu 14.04 je kořenovým procesem „init“.

```
$ pstree
systemd├─ModemManager──2*[{ModemManager}]
│├─NetworkManager├─dhclient
││├─dnsmasq
││└─3*[{NetworkManager}]
├─accounts-daemon──2*[{accounts-daemon}]
├─gnome-keyring-d──5*[{gnome-keyring-d}]
├─irqbalance
├─kerneloops
├─lightdm├─Xorg──2*[{Xorg}]
││├─lightdm├─init├─/usr/bin/termin├─bash──pstree
│││├─gnome-pty-helpe
│││└─4*[{/usr/bin/termin}]
││├─AsciidocFX──java──81*[{java}]
││├─GoogleTalkPlugi──7*[{GoogleTalkPlugi}]
││├─at-spi-bus-laun├─dbus-daemon
│││└─3*[{at-spi-bus-laun}]
│└─...
└─...
```

Když proces skončí očekávaným způsobem (nedojde k jeho zabití nebo havárii), může rodičovskému procesu vrátit návratový kód. Zda a jaký význam jednotlivým číselným kódům bude rodič přiřkládat se různí. Podle konvence vše kromě nuly obvykle indikuje chybu.

6.2. Init systémy

System V (nebo SysV)

Na výpisech `pstree` jistě povšimnete, že všechny „skutečné“ procesy jsou potomky procesu `systemd`. Tak se jmenuje od Ubuntu 15.04 výchozí init manager. Tento úplně první proces spuštěný jádrem proto dostane PID vždy 1 a je přímým či nepřímým

rodičem všech ostatních procesů. Prvotní proces `init` má historicky několik implementací. Tradičně všechny vycházejí z tzv. System V `init` (SysV).

Upstart

Ubuntu dlouho používalo svůj vlastní `init` manager `Upstart`, který vznikl přímo na míru Ubuntu. Používala ho však jistou dobu i konkurenční Fedora a stále používá např. Chrome OS.

systemd

V Ubuntu 15.04 došlo k důležité, i když na první pohled těžko postřehnutelné, změně - `Upstart` byl nahrazen manažerem `systemd`.

Poznámka

Autoři `systemd` si přejí, abychom název důsledně psali jako `systemd`, nikoli `SystemD`, `System D` ap. Pokud bude název na začátku věty, budeme však používat `Systemd`.

`Systemd` je taktéž výchozím `init` systémem Debianu 8 (Jessie) a vyšší.

Tato změna na `systemd` byla poměrně silně kontroverzní a řada uživatelů Ubuntu i Debian rozhodnutí považuje za špatné. Důvod k této nepopularitě je především podle názoru odpůrců zbytečná složitost, a příliš široký záběr, který porušuje unixová pravidla menších jednodušších nástrojů. `systemd` je však bezesporu nejpokročilejší `init` systém nabízející agresivní paralelní provádění, bohatou konfiguraci, logovací služby, správu `mount` a `automount` pointů a další.

Přijmání `systemd` usnadňuje, že je koncipován jako náhrada za `sysvinit` a naštěstí tedy podporuje tradiční SysV `init` skripty.

V důsledku tohoto historického vývoje najdete v Ubuntu vše - znát bychom měli tradiční SysV skripty, jejich `Upstart` ekvivalent tzv. `jobs` i nově se rozšiřující `targets` jak skriptům říká `systemd`.

Porovnání SysV, Upstart a systemd

	SysV	Upstart	systemd
konfiguraci služby se říká	skript	job	unit
hlavní složka s konfiguračními službami	/etc/init.d/	/etc/init/	/etc/systemd/
výchozí init systém od			v Ubuntu od 15.04 (Vivid), v Debianu od 8 (Jessie)

6.2.1. Upstart

Upozornění

Vzhledem k tomu, že se Upstart již od Ubuntu 15.04 neupoužívá a v Debianu se nepoužíval nikdy, zmíníme Upstart jen okrajově. Případné zájemce odkazujeme na <http://upstart.ubuntu.com/>.

Upstart konfiguraci načítal ze složky `/etc/init/`, kde hledá textové soubory s koncovkou `.conf`. Jeden soubor pro jednu službu (zvanou v Upstartu job). Obvykle obsahuje něco málo dokumentace, a zejm. jak proces nastartovat a ukončit, ve kterých runlevech ap. Více se o podobě souborů dozvíte v `man 5 init`.

Tradiční System V init používal ke konfiguraci soubor `/etc/inittab`. V Ubuntu s Upstartem nepoužívá a ani neexistuje.

6.2.2. System V init

System V je natolik tradiční, že Upstart i systemd mají podporu pro tradiční System V init skripty umístované do složky `/etc/init.d/`. Tento typ init skriptů je široce rozšířený a díky podpoře pozdějších init systémů můžeme nadále zůstat u těchto skriptů. Rovněž množství DEB balíčků je dodáváno s SysV init skripty.

Poznámka

Bohužel se liší (i když ne zásadně) styl psaní Debian/Ubuntu a Red Hat init.d skriptů. My se zaměříme na Debianí styl.

Vytvoření démonu znamená vytvoření (Bash) skriptu v /etc/init.d/. Ve složce existuje šablona skeleton, kterou použijte pro vytvoření nového démonu:

```
cp /etc/init.d/skeleton /etc/init.d/helloworld
sudo chmod 775 /etc/init.d/helloworld
```

Skripty by měli začínat tzv. *shebang* řádkem, pak musí obsahovat hlavičku s povinnými a nepovinnými údaji, a zbytek souboru jsou běžné příkazy pro spuštění služby.

Ukázka init.d skriptu

```
#!/bin/sh

### BEGIN INIT INFO
# Provides:          samba
# Required-Start:    smbd nmbd
# Required-Stop:     smbd nmbd
# Default-Start:
# Default-Stop:
# Short-Description: ensure Samba daemons are started (nmbd and smbd)
### END INIT INFO

...příkazy pro spuštění služby...
```

Zbytek skriptu je obsluha parametrů „startovače“ service: start, stop, restart, reload, force-reload a status odpovídající bash funkcím `do_*()` ve skriptu. Např.:

```
....

do_start()
{
    # Return
    # 0 if daemon has been started
    # 1 if daemon was already running
    # 2 if daemon could not be started
    start-stop-daemon --start --quiet --pidfile $PIDFILE --exec $DAEMON --t
        || return 1
    start-stop-daemon --start --quiet --pidfile $PIDFILE --exec $DAEMON --
        $DAEMON_ARGS \
        || return 2
    # Add code here, if necessary, that waits for the process to be ready
```

```
# to handle requests from services started subsequently which depend
# on this one. As a last resort, sleep for some time.
}
...

```

6.2.2.1. Instalace init.d skriptů pomocí `update-rc.d`

Přestože vytvoření a mazání init.d skriptů se nezdá těžké, použijte nástroj `update-rc.d`.

Poznámka

Důvod souvisí s runlevely (úrovně běhu), protože `update-rc.d` umí především aktualizovat symlinky `NNjméno` v složkách `/etc/rc?.d/`. Nástroj jen parametrem určíte ve kterých runlevelech se má služba spouštět. Manipulace v těchto složkách ručně by byla náročná a náchylná na chyby (překlep v názvu, opomenutí symlinku, ...).

Ke všem příkazům můžete přidat `-n`, aby se povel neprovedl, ale jen se vypsalo, co by se provedlo („běh na sucho“).

Nejčastější použití je:

```
$ update-rc.d <služba> defaults
```

např.:

```
$ update-rc.d helloworld defaults
```

pro vytvoření služby `helloworld`. Jméno služby musí odpovídat jménu souboru v `/etc/init.d/`, tj. ze skriptu `/etc/init.d/helloworld`. Vytvoření znamená ve skutečnosti vytvoření linků v `/etc/rc?.d/` složkách. Volba `defaults` znamená start pro runlevely 2, 3, 4 a 5; a stop pro runlevely 0, 1 a 6.

Smazání smaže všechny linky z `/etc/rc?.d/` složek. Skript v `/etc/init.d/` musí být odstraněn ručně, jinak se vypíše chyba:

```
$ update-rc.d helloworld remove
```

Rada

Viz `man update-rc.d`.

Ovládání služby (spuštění, zastavení, ...) provádíme přímo spuštěním skriptu nebo (lépe) spouštěčem služeb `service`.

6.2.3. Runlevel (úroveň běhu)

Tradiční System V podporuje koncept zvaný runlevel, kdy určité služby běží jen v určité úrovni běhu, protože v jiné nejsou třeba. Např. při opravě poškozeného serveru, jinak obsluhujícího mnoho uživatelů, se při spuštění do runlevel 1 nepřihlásí nikdo kromě vás.

Runlevely jsou dnes se `systemd` zastaralým způsobem, jak spouštět a zastavovat skupiny SysV služeb, ale kvůli zpětné kompatibilitě stále fungují. Pro seskupování služeb v `systemd` slouží tzv. targets (cíle). `Systemd` dokáže provozovat více cílů, může být v SystemV aktivní jen jeden runlevel v jeden okamžik. Srovnání runlevelu a targetu v následující tabulce je tedy spíše orientační.

Runlevely v Debian/Ubuntu a jejich odpovídající systemd target

SysV runlevel	systemd target	význam
0	poweroff.target	zastavení
1	rescue.target	jednouživatelský mód
2, 3, 4	multi-user.target	víceuživatelský režim
5	graphical.target	grafický víceuživatelský režim
6	reboot.target	reboot

Předchozí a aktuální runlevel oddělený mezerou vám řekne příkaz `runlevel`. Pokud není předchozí runlevel znám, vypíše se „N“:

```
$ runlevel
N 2
```

6.2.3.1. /etc/rc?.d/ složky

Varování

S obsahem těchto složek raději nemanipujte ručně, ale s pomocí nástroje [Instalace init.d skriptů pomocí update-rc.d.](#)

V /etc/ se nachází několik složek pojmenovaných rc?.d, kde ? je číslo odpovídající runlevelu.

rc?.d/ složky

```
$ ls -d /etc/rc?.d/
/etc/rc0.d /etc/rc2.d /etc/rc4.d /etc/rc6.d
/etc/rc1.d /etc/rc3.d /etc/rc5.d /etc/rcS.d
```

Složka /etc/rcS.d/ je pro skripty spouštěné výhradně při bootování systému. Všechny ostatní jsou číslovány podle runlevelu ve kterém jsou skripty vykonány.

Tyto složky neobsahují skutečné soubory, ale jen symlinky na skripty v /etc/init.d/. To proto, aby bylo možné přidávat a mazat skripty odpovídající služeb do více úrovní současně a bez ovlivnění init.d skriptů samotných.

Odkazy mají prefix K pro skript při ukončení a S při vstupu do runlevelu. Po K/S následuje číslo a jméno. Číslo určuje pořadí provedení. Je-li stejného čísla více souborů, seřadí se abecedně a proto jméno nemá jen popisný charakter.

Ukázka obsahu složky /etc/rc4.d/

```
$ ls -l /etc/rc6.d/
total 4
lrwxrwxrwx 1 root root 17 bře 22 2015 K09apache2 -> ../init.d/apache2
lrwxrwxrwx 1 root root 29 úno 23 2015 K10unattended-upgrades -> ../init.d/una
lrwxrwxrwx 1 root root 20 úno 23 2015 K20kerneloops -> ../init.d/kerneloops
lrwxrwxrwx 1 root root 15 úno 23 2015 K20rsync -> ../init.d/rsync
lrwxrwxrwx 1 root root 27 úno 23 2015 K20speech-dispatcher -> ../init.d/speech
lrwxrwxrwx 1 root root 15 úno 23 2015 K21mysql -> ../init.d/mysql
lrwxrwxrwx 1 root root 31 úno 24 2015 K65vboxautostart-service -> ../init.d/v
lrwxrwxrwx 1 root root 33 úno 24 2015 K65vboxballoonctrl-service -> ../init.d
```



```

lrwxrwxrwx 1 root root 25 úno 24 2015 K65vboxweb-service -> ../init.d/vboxweb
lrwxrwxrwx 1 root root 17 úno 24 2015 K80vboxdrv -> ../init.d/vboxdrv
-rw-r--r-- 1 root root 351 bře 13 2014 README
lrwxrwxrwx 1 root root 18 úno 23 2015 S20sendsigs -> ../init.d/sendsigs
lrwxrwxrwx 1 root root 17 úno 23 2015 S30urandom -> ../init.d/urandom
lrwxrwxrwx 1 root root 22 úno 23 2015 S31umountnfs.sh -> ../init.d/umountnfs.
lrwxrwxrwx 1 root root 18 úno 23 2015 S40umountfs -> ../init.d/umountfs
lrwxrwxrwx 1 root root 20 úno 23 2015 S60umountroot -> ../init.d/umountroot
lrwxrwxrwx 1 root root 16 úno 23 2015 S90reboot -> ../init.d/reboot

```

6.2.4. Systemd

Poznámka

Systemd je rozsáhlý software, který řeší daleko více úkolů, než pouze provoz služeb. Zahrnuje např. správu mount pointů, uživatelských sessions (sezení), logování a několik dalších. Popis všeskeré jeho funkcionality by vydal na samostatnou příručku a školení. V této knize se zaměříme na to nejpotřebnější z pohledu správce - správu služeb.

Poznámka

Systemd má vlastní bohatou terminologii jako target, unit ap. Možný český překlad uvedeme v závorce při prvním výskytu. Nadále však budeme používat počestěné anglické názvy jako „několik targetů“ místo kompletně českého překladu „několik cílů“, který by byl podle našeho názoru více zmatením, než přínosem.

Hlavním programem pro komunikaci se systemd se jmenuje `systemctl`. Systemd dokáže ovládat více druhů tzv. *units (jednotek)* podle typu spravovaného zdroje. Konfigurace unitů je v textových souborech, které mají syntaxi velmi podobou INI souborům. Přípona souboru určuje typ unitu. Např. `cups.service` je konfigurací služby CUPS nebo `graphical.target` targetu „graphical“. Protože nejčastější jsou servisy, tak pokud příponu nevedete systemd doplní `.service`.

Šetřete prsty - alias pro sudo systemctl

Program `systemctl` skoro ve všech případech ho musíme zavolat jako root (se `sudo`). Vypisování tak často používaného programu s celkem dlouhým názvem je skvělým kandidátem na alias.

Do vašeho `~/.bash_aliases` (soubor případně vytvořte) přidejte alias např. pojmenovaný `sc`:

```
$ echo alias sc='sudo systemctl' >> ~/.bash_aliases
```

ukončíte současný terminál, spusťte nový a místo např. `sudo systemctl list-units` můžete psát jen `sc list-units`.

Nevýhoda je, že tím přicházíte o poměrně propracovaný `Tab` auto-completion.

6.2.4.1. Units (jednotky)

Units jsou pro `systemd` obecné jednotky mezi kterými je možné určovat vzájemné závislosti. Existuje 12 druhů unitů podle spravovaného zdroje. Mezi nejdůležitější však patří:

- *service units* - startují a kontolují démony - pro nás nejdůležitější druh unitu
- *socket units* - zapouzdřují lokální IPC nebo síťové sokety
- *target units* - seskupení více units
- *device units* - vystavují zařízení jádra
- *mount units* - kontrolují přípojně body

Unit mohou mohou nacházet v těchto 5 základních stavech:

- *activating* (v průběhu aktivace) a *active* (již aktivované)
- *deactivating* (v průběhu inaktivace) a *inactive* (již inaktivované)
- *failed* (nepodařilo se aktivovat)

Unit určitého typu mohou mít vlastní „podstavy“.

6.2.4.2. Spouštění, zastavení a restart služeb

K ovládání všech typů unit slouží program `systemctl`, který jako povinný parametr potřebuje prováděnou operaci, jednotku. Pokud není uvedena přípona předpokládá se `.service`. Obecná syntaxe je tedy:

```
sudo systemctl <operace> <unit>[.service]
```

Pro základní ovládání služeb používáme operace

- `start` pro spouštění unit
- `stop` pro zastavení unit
- `restart` pro restart unit

Např. zastavení služby CUPS zařídíme pomocí:

```
sudo systemctl stop cups
```

Některé unit mohou podporovat tzv. `reload` neboli načtení konfigurace bez výpadku služby. Užitečná operace je `reload-or-restart`, která zkusí `reload` a pokud není unitou podporován udělá `restart`.

```
sudo systemctl reload-or-restart cups
```

6.2.4.3. Povolení a zakázání služeb

Operace `start`, `stop` a další výše popsané znamenají „proved' teď“. Aby se unit spustil automaticky při startu ho musíme povolit pomocí operace `enable`:

```
sudo systemctl enable <nazev_sluzby>
```

Povolení znamená vytvoření symlinku výchozí konfigurace služby (obvykle ve složce `/lib/systemd/system/`) do místa, kde `systemd` očekává units aktivované při bootování (obvykle ve složce `/etc/systemd/system/` `<nejaky_target>.target.wants`). Více o [Targets \(cíle\)](#) za okamžik.

Opačná operace `disable`:

```
sudo systemctl disable <nazev_sluzby>
```

odstraní symlinky vytvořené při povolení služby.

Důležité

Povolení/zakázání neznamena spouštění/zastavení služby v aktuální session. Chcete-li službu povolit i spustit musíte použít dva příkazy - enable následované start.

6.2.4.4. Konfigurace služeb

Systemd hledá jednotky na dvou hlavních místech. Konfigurace tak, jak vypadají ve výchozím stavu, byste našli ve složce `/lib/systemd/system/`. Např.

`cups.service` soubor služby CUPS zajišťující tisk má systémovou definici zde:

```
$ ll $(find /lib/systemd/system -name cups.service)
-rw-r--r-- 1 root root 175 úno 16 21:46 /lib/systemd/system/cups.service
```

Definice CUPS služby určená k přizpůsobení správcem je ve složce `/etc/systemd/system/`. Tam najdete kopii `cups.service` a symlink na originál v `/lib/systemd/system/` ze složky cíle `printer.target.wants/`:

```
$ ll $(find /etc/systemd/system -name cups.service)
-rw-r--r-- 1 root root 175 úno 16 21:46 /etc/systemd/system/cups.service
lrwxrwxrwx 1 root root 32 bře 5 18:13 /etc/systemd/system/printer.target.want
```

6.2.4.5. Stav služby

Detailní informaci o stavu služby poskytne operace `status`. Z výstupu zjistíme kromě stavu cgroups hierarchie a užitečné je i několik prvních řádků z logu:

```
$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: ena
Active: active (running) since Čt 2017-03-02 13:58:48 CET; 3 days ago
   Docs: https://docs.docker.com
Main PID: 11667 (dockerd)
   Tasks: 56
Memory: 889.2M
   CPU: 10min 14.811s
CGroup: /system.slice/docker.service
├─11667 /usr/bin/dockerd -H fd:// -H tcp://0.0.0.0:4243
├─11674 docker-containerd -l unix:///var/run/docker/libcontainerd/docke
├─14172 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 32
└─14177 docker-containerd-shim e71dee1a12ebf7579765d53b1d15376c4a5f186c
Mar 04 08:32:59 jell-nb dockerd[11667]: time="2017-03-04T08:32:59.872897132+01:
```

```
Mar 04 18:00:44 jell-nb systemd[1]: Started Docker Application Container Engine
Mar 04 18:03:40 jell-nb systemd[1]: Reloading Docker Application Container Engi
Mar 04 18:03:40 jell-nb dockerd[11667]: time="2017-03-05T18:03:40.911321154+01:
Mar 04 18:03:40 jell-nb dockerd[11667]: time="2017-03-05T18:03:40.911382634+01:
Mar 04 18:03:40 jell-nb systemd[1]: Reloaded Docker Application Container Engin
Mar 04 18:04:03 jell-nb systemd[1]: Reloading Docker Application Container Engi
Mar 04 18:04:03 jell-nb dockerd[11667]: time="2017-03-05T18:04:03.078121402+01:
Mar 04 18:04:03 jell-nb dockerd[11667]: time="2017-03-05T18:04:03.078275229+01:
Mar 04 18:04:03 jell-nb systemd[1]: Reloaded Docker Application Container Engin
```

Postačí-li odpověď, zda je služba aktivní použijte:

```
$ sudo systemctl is-active docker
active
```

nebo, zda je povolena:

```
$ sudo systemctl is-enabled docker
enabled
```

6.2.4.6. Výpis aktivních nebo všech units

Ve výpisu všech aktivních jednotek v systemd operací `list-units` si všimnete různých typů unit, které poznáte podle přípony (zkráceno):

```
sudo systemctl list-unit
```

Výpis aktivních jednotek je výchozí operací, takže stejný výstup nám dá i jen `systemctl` (zkráceno):

```
$ sudo systemctl
UNIT                                LOAD    ACTIVE SUB    DES
proc-sys-fs-binfmt_misc.automount  loaded active waiting Arb
sys-devices-pci0000:00-0000:00:02.0-drm-card0-card0\x2deDP\x2d1-intel_backlight
sys-devices-pci0000:00-0000:00:14.0-usb1-1\x2d6-1\x2d6:1.0-bluetooth-hci0.device
....
snap-core-1264.mount                loaded active mounted  Mou
snap-core-1287.mount                loaded active mounted  Mou
snap-core-1337.mount                loaded active mounted  Mou
snap-keepassx\x2delopio-1.mount     loaded active mounted  Mou
...
init.scope                          loaded active running  Sys
session-c2.scope                    loaded active running  Ses
accounts-daemon.service             loaded active running  Acc
acpid.service                       loaded active running  ACP
alsa-restore.service                loaded active exited  Sav
```

```

apparmor.service          loaded active exited    LSB
apport.service           loaded active exited    LSB
avahi-daemon.service     loaded active running    Avahi
bluetooth.service        loaded active running    Bluetooth
...
LOAD    = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB     = The low-level unit activation state, values depend on unit type.

231 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.

```

Jednotlivé sloupce mají tento význam:

- **UNIT** - název systemd unity
- **LOAD** - stav načtení unity
- **ACTIVE** - je unit aktivní?
- **SUB** - podstav (substate) unitu. Každá unita může mít své vlastní podstavy
- **DESCRIPTION** - popis účelu unity

Vypisované unity můžeme omezit na určitý typ pomocí `--type`, např.:

```
sudo systemctl list-units --type=service
```

Výše použité `list-units` vypisuje jen na units, které se systemd pokusil načíst. Pro zobrazení všech unitů, které systemd zná slouží `list-unit-files`:

```

$ sudo systemctl list-unit-files
UNIT FILE                                STATE
proc-sys-fs-binfmt_misc.automount       static
proc-sys-fs-binfmt_misc.mount           static
snap-core-1264.mount                     enabled
snap-core-1287.mount                     enabled
snap-tpad-17.mount                       enabled
sys-fs-fuse-connections.mount            static
alsa-utils.service                       masked
anacron-resume.service                   enabled
anacron.service                           enabled
apport-forward@.service                  static
apt-daily.service                         static
autovt@.service                           enabled
avahi-daemon.service                     enabled
...

```

Sloupec *STATE* může nabývat těchto hodnot:

- *enabled* - unit je možné používat (neznamená, že je aktivní)
- *disabled* - unit je zakázaná
- *static* - unit nemůže být samostatně použita, ale jen jako např. závislost jiné unit
- *masked* - unit je zamaskovaná a neschopná startu

6.2.4.7. Maskování služeb

Ve výstupu `list-unit-files` jste si mohli povšimnout stavu *masked*. Službu (nebo obecně unit), který označíte jako maskovaný je nemožné automaticky i manuálně spustit. Systemd toho dosáhne opět pomocí symlinků, kdy unit nacílí na `/dev/null`.

Řekne nám to `systemctl`, když službu zamaskujeme operací `mask`:

```
$ sudo systemctl mask cups
Created symlink from /etc/systemd/system/cups.service to /dev/null.
```

Pokus o start se nám pak nepovede:

```
$ sudo systemctl start cups
Failed to start cups.service: Unit cups.service is masked.
```

Dokud ji neodmaskujeme s `unmask`:

```
$ sudo systemctl unmask cups
Removed symlink /etc/systemd/system/cups.service.
```

6.2.4.8. Vytváření služeb

V následujícím cvičení vytvoříme minimální, ale plně funkční službu, která bude do souboru zapisovat aktuální datum.

Začneme napsáním skriptu, který zjistí aktuální datum a zapíše ho do souboru. Skript můžete napsat v jakémkoli oblíbeném jazyce - např. v Bashi nebo Pythonu. My zvolíme Bash. Někde na disku, např. ve vaší domovské složce, vytvořte spustitelný soubor `today.sh`:

```
$ touch /home/joe/today.sh
$ chmod +x /home/joe/today.sh
```

a pomocí editoru (např. nano) vložte do souboru tento obsah:

```
#!/bin/bash
while true; do sleep 15; date -I > /tmp/today.txt; done
```

Vytvořte minimální definici služby ve složce `/etc/systemd/system/`:

```
sudo nano /etc/systemd/system/today.service
```

s tímto obsahem:

```
[Unit]
Description=Today date service

[Service]
ExecStart=/home/joe/today.sh

[Install]
WantedBy=multi-user.target
```

Příkaz `ExecStart` je příkaz, který se spustí při aktivaci služby.

Poznámka

Kromě odkazu na náš soubor skriptu by se takto jednoduchý skript nechal zapsat přímo v `.service` souboru:

```
...
[Service]
ExecStart=/bin/bash -c "while true; do sleep 15; date -I > today.txt; done
...
```

Druhá klíčová věc v definici služby je `WantedBy` určující v jakém targetu očekáváme spuštění této služby.

Každopádně zbývá již jen povolení služby neboli vytvoření odpovídajících symlinků:


```
$ sudo systemctl enable today.service
Created symlink from /etc/systemd/system/multi-user.target.wants/today.service
```

A nyní už jen restart do určeného targetu nebo spuštění můžeme provést okamžitě:

```
sudo systemctl start today
```

Případně si ověřit, že jsme neprovedli někde chybu:

```
$ sudo systemctl status today
● today.service - Today date
   Loaded: loaded (/etc/systemd/system/today.service; enabled; vendor preset: e
   Active: active (running) since Sun 2017-03-05 22:34:41 CET; 31s ago
 Main PID: 8409 (today.sh)
    Tasks: 2
   Memory: 616.0K
      CPU: 2ms
   CGroup: /system.slice/today.service
           └─829 python3 /bin/bash /home/joe/today.sh
           └─837 sleep 15

Mar 05 22:34:41 vbox-joe systemd[1]: Started Today date service.

$ cat /tmp/today.txt
2017-03-05
```

6.2.4.9. Targets (cíle)

Targety jsou obdobou [runlevelů](#) v System V. Slouží hlavně pro seskupení ostatních druhů unitů. Konfigurační soubory mají příponu `.target`. Seskupením jiných unitů do targetu nastavujeme stav celého systému. Aktivace určitého targetu znamená aktivaci všech unitů v targetu.

Definice targetů

Např. target `multi-user.target` odpovídající zhruba runlevelům 2, 3 a 4 je definován takto:

```
$ sudo systemctl cat multi-user.target
# /lib/systemd/system/multi-user.target
# This file is part of systemd.
#
# systemd is free software; you can redistribute it and/or modify it
# under the terms of the GNU Lesser General Public License as published by
# the Free Software Foundation; either version 2.1 of the License, or
```

```
# (at your option) any later version.

[Unit]
Description=Multi-User System
Documentation=man:systemd.special(7)
Requires=basic.target
Conflicts=rescue.service rescue.target
After=basic.target rescue.service rescue.target
AllowIsolate=yes
```

Většinou nám více poví operace `list-dependencies`, která sestaví strom závislostí jakékoli jednotky. Použito pro náš `multi-user.target` zjistíme (zkráceno):

```
$ sudo systemctl list-dependencies multi-user.target
multi-user.target
● └─anacron.service
● └─apport.service
● └─avahi-daemon.service
● └─cgroupfs-mount.service
● └─cron.service
● └─cups-browsed.service
● └─cups.path
● └─dbus.service
● └─dns-clean.service
● └─docker.service
● └─flexibee.service
● └─glances.service
● └─grub-common.service
● └─hddtemp.service
...
```

Výchozí target

Výchozí target můžeme zjistit pomocí `systemctl get-default`. Na desktopové stanici to bude nejspíš `graphical.target`:

```
$ sudo systemctl get-default
graphical.target
```

Změnit výchozí target můžete pomocí `systemctl set-default`:

```
sudo systemctl set-default multi-user.target
```

Dostupné a aktivní targety

Všechny dostupné targety zjistíte známým příkazem `systemctl list-unit-files --type=target`. Na rozdíl od runlevelů může být targetů aktivních v jeden moment více - důkaz nám poskytnete:

```
$ sudo systemctl list-units --type=target
UNIT                                LOAD    ACTIVE SUB    DESCRIPTION
basic.target                        loaded active active Basic System
bluetooth.target                    loaded active active Bluetooth
cryptsetup.target                    loaded active active Encrypted Volumes
getty.target                         loaded active active Login Prompts
graphical.target                     loaded active active Graphical Interface
local-fs-pre.target                  loaded active active Local File Systems (Pre)
local-fs.target                      loaded active active Local File Systems
multi-user.target                    loaded active active Multi-User System
network-online.target                loaded active active Network is Online
network-pre.target                   loaded active active Network (Pre)
network.target                       loaded active active Network
nss-user-lookup.target               loaded active active User and Group Name Lookups
paths.target                         loaded active active Paths
remote-fs-pre.target                 loaded active active Remote File Systems (Pre)
remote-fs.target                     loaded active active Remote File Systems
slices.target                        loaded active active Slices
sockets.target                       loaded active active Sockets
sound.target                         loaded active active Sound Card
swap.target                          loaded active active Swap
sysinit.target                       loaded active active System Initialization
time-sync.target                     loaded active active System Time Synchronized
timers.target                        loaded active active Timers

LOAD    = Reflects whether the unit definition was properly loaded.
ACTIVE  = The high-level unit activation state, i.e. generalization of SUB.
SUB     = The low-level unit activation state, values depend on unit type.

22 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
```

Izolování targetů

Občas se může hodit spustit jen unity v targetu a zastavit všechny ostatní z dalších targetů. K tomu slouží operace `isolate`. Je to dost podobné změně runlevelů v System V.

Např. přepnutí na `rescue.target` a vypnutí všech units, které nepatří do tohoto targetu napíšete jako:

```
sudo systemctl isolate rescue.target
```

Systemd má pro izolaci několik čas urychlujících zkratek. Např.

- místo `systemctl isolate rescue.target` můžete psát jen `systemctl rescue`
- místo `systemctl isolate default` můžete psát jen `systemctl default`

Další zkratky jsou pro zastavení systému:

```
sudo systemctl halt
```

Vypnutí systému:

```
sudo systemctl poweroff
```

Restart systému:

```
sudo systemctl reboot
```

Tyto zkratky jsou vhodnější, než pouhá izolace na určený target a dovedou informovat přihlášené uživatele, že se bude stroj např. restartovat.

Pro ještě větší komfort původně samostatné programy pro ukončení a restart PC `halt`, `poweroff` a `reboot` volají ve skutečnosti tyto příkazy `systemd`.

6.3. Plánování úloh

Chcete-li pozdržet spuštění procesu, použijte `sleep`. Chcete-li ho odložit, pak `at`. Chcete-li ho naplánovat na za týden, 2x měsíčně atd., využijte `cron`.

6.3.1. sleep

`Sleep` je velmi jednoduchý - prostě nic nedělá zadaný počet sekund. Hodí se to hlavně do skriptů, kdy `sleep` „pozastaví“ jeho provádění. Nebo si chcete nechat něco připomenout za 10 minut:

```
(sleep 10m; echo 'Miluju Linux!')&
```

Číslo bez označení nebo číslo s příponou „s“ znamená počet sekund (např. 30s). Dále je možné použít „m“ pro minuty, „h“ pro hodiny, „d“ pro dny.

6.3.2. at

S `at` naplánujete spuštění v určitý čas nebo za určitý interval. Definice „kdy“ je velmi lidská, např.:

```
$ at now + 2 hours
$ at tomorrow + 4 days
$ at 10:39
```

Na to se objeví prostředí podobné promptu, kde můžete zadat jakékoli příkazy ke spuštění v daný čas. Sekvenci příkazů ukončíte znakem EOF neboli stiskem `Ctrl+D`.

Výpis čekajících úloh ve frontě zjistíte pomocí `atq`.

Z výpisu uvidíte také ID úlohy, které můžete použít k odstranění pomocí `atrm <ID>`.

6.3.3. Cron

Pro pravidelné provádění úloh poslouží Cron. Z konfiguračních souborů zvaných `crontab` zjistí kdy a jaké příkazy spustit.

Tip

Uživatelé GUI mohou zkusit grafický nástroj pro správu cronů <http://gnome-schedule.sourceforge.net/> (sudo apt-get install gnome-schedule).

6.3.3.1. crontab

`Crontab` je jednoduchý textový soubor se seznamem příkazů a výrazem určující „kdy“ příkaz provést.

Editace

Soubor `crontab` byste nikdy neměli editovat na přímo, ale prostřednictvím `crontab -e`, který otevře váš textový editor a po uložení a opuštění editoru zkontroluje případné chyby.

Vyvolání `crontab -e` otevře váš uživatelský crontab. Pro editaci roota použijte `sudo crontab -e`.

Tip

Náš „oblíbený“ editor určíte do proměnné prostředí EDITOR. Např. má-li to být nano, pak `export EDITOR=nano`.

Zobrazení a vymazání

Podobně pro zobrazení můžete použít `crontab -l`, resp. pro odstranění `crontab -r`.

Povolení/omezení uživatelů cronu

V Ubuntu a Debianu standardně můžou všichni uživatelé použít cron. Můžete ale určit uživatele, kteří smí/nesmí používat crontab editací souborů `/etc/cron.allow` a `cron.deny`. Tyto soubory dokonce v Ubuntu ani standardně neexistují. Prostudujte dokumentaci, protože toto řešení má řadu „ale“.

Viz také `man 1 crontab`.

6.3.3.2. cron výrazy

Poznámka

Cron výrazy je důležité alespoň zběžně ovládat. Používají se i v řadě knihoven programovacích jazyků. V podstatě je cron výraz de facto standard pro zápis časového plánu.

Na první pohled je syntaxe crontabů zcela nepochopitelná:

```
5 * * * * curl http://virtage.com
```

Každý řádek je rozdělen na plán a příkaz(y) k provedení Bashem. Výstup příkazů se standardně posílá na mail uživatele daného crontabu, ale často bývá přesměrován do log souboru.

Plán obsahuje povinných 5 elementů, kdy každá hodnota postupně odpovídá

- minutě (0-59)
- hodině (0-23)
- dne měsíce (1-31)
- měsíci (1-12)
- dni týdne jako číslo (0-6, kde 0 je neděle) nebo zkratka anglického názvu (MON, TUE, WED, THU, FRI, SAT, SUN)

Poznámka

Pro jistotu uvádíme, že první minuta nového dne je 0:00 a poslední je 23:59. Podobně první minuta nové hodiny je 0 a poslední 59.

Na místě elementu můžete taky použít

- hvězdičku (*) pokud chcete vyjádřit každou minutu, hodinu, atd.
- čárku (,) pro seznam hodnot
- operátor dělení (/) pro každou n. hodnotu
- operátor rozsah (-) pro rozpětí hodnot

Příklady cron výrazů

Výraz	Popis
* * * * *	každou minutu
15 * * * *	každých 15. minut každé hodiny
0 9 * * 1	každé pondělí v devět hodin
0 8,12,17,22 * * 6,0	každou sobotu a neděli v 8, 12, 17 a 22 hodin
*/15 * * * *	každých 15 minut

Výraz	Popis
* / 5 * * * 1-5	každých 5 minut od pondělí do pátku (1-5)

Kromě těchto výrazů můžete použít i tzv. „@zkratky“ pro časté plány.

@Zkratky cron výrazů

@Zkratka	Popis
@reboot	po startu PC
@yearly nebo @annually	ročně (jako 0 0 1 1 *)
@monthly	měsíčně (jako 0 0 1 * *)
@weekly	týdně (jako 0 0 * * 0)
@daily nebo @midnight	denně (jako 0 0 * * *)
@hourly	každou hodinu (jako 0 * * * *)

Tip

Na internetu najdete spoustu testerů a rozepisovačů cron výrazů. Např. <http://cron.schlitt.info> nebo <http://www.cronchecker.net>.

Tip

Viz také man 5 crontab.

6.3.3.3. System-wide a uživatelské crontaby

Každý uživatel má vlastní crontab. Cron démon provede crontab uživatele bez ohledu na to, jestli je zrovna přihlášen. System-wide crontab(y) jsou editovatelné jen rotem obsahující úkoly jako rotování logů, zálohování ap.).

Crontaby uživatelů leží ve `/var/spool/cron/crontabs/`. Systémové crontaby jsou na několika místech.

Upozornění

Hlavní soubor `/etc/crontab` byste nikdy neměli editovat, ale umístit crontaby do centrální system-wide složky `/etc/cron.d/`.

Syntaxe systémových crontabů je téměř stejná jako u uživatelských. Uživatelské crontaby se provádí pod daným uživatelem. Pro systémové musíte uživatele určit (nejčastěji to bývá root).

Podoba záznamu systémového crontabu navíc proto obsahuje username uživatele:

```
<výraz> <uživatel> <příkaz>
```

např.:

```
@daily root apt-get autoremove
```

6.3.3.4. Anacron a Debian/Ubuntu vylepšení

Anacron je modernější implementací tradičního Cronu zvaného Vixie podle svého autora. Jedno vylepšení je hlavně pro uživatele, kteří nemají počítač stále zapnutý - anacron umí spustit úlohy, které měli provést, když byl počítač vypnutý.

Debian/Ubuntu nabízí také rozšíření Vixie Cronu pro system-wide úkoly prováděné každou hodinu, den, týden a měsíc. Stačí umístit skript (nebo odkaz) do příslušné složky v daný čas provedou:

- `/etc/cron.hourly/`
- `/etc/cron.daily/`
- `/etc/cron.weekly/`
- `/etc/cron.monthly/`

7. Výkon, dohled, logy

V této závěrečné stručné kapitole se seznámíme s příkazy pro sledování výkonu, programy na monitoring systému a logování.

7.1. Sledování výkonu

7.1.1. time a time

Bash nabízí vestavěný příkaz `time` pro přesné měření délky trvání vykonávaných programů.

```
$ time find ~ -name java
...

real      0m2.686s
user      0m0.566s
sys       0m0.634s
```

Existuje rovněž program `time`. Protože koliduje s zabudovaným příkazem Bashe stejného jména, musíme uvést plnou cestu:

```
$ /usr/bin/time find ~ -name java
...

Command exited with non-zero status 1
0.51user 0.55system 0:01.08elapsed 99%CPU (0avgtext+0avgdata 5852maxresident)k
0inputs+0outputs (0major+5300minor)pagefaults 0swaps
```

7.1.2. uptime

Jednoduchý prográmeček pro zjištění jak dlouho stroj běží a kromě toho zobrazí i aktuální čas, počet přihlášených uživatelů a průměrnou zátěž za poslední 1, 5 a 15 minut.

Tip

Jako „běží“ se počítá i suspendace a hibernace.

::

```
$ uptime 14:35:46 up 5 days, 17:30, 2 users, load average: 1,00, 0,74, 0,54
```

S parametrem -p vypíše čas podrobněji:

```
$ uptime -p
up 5 days, 17 hours, 30 minutes
```

7.1.3. ps, top

S oběma nástroji jsme se již seznámili v prvním díle.

7.1.4. du, ncd

S oběma nástroji jsme se již seznámili v prvním díle.

7.1.5. df

S nástrojem jsme se již seznámili v prvním díle.

7.1.6. free

Zobrazí velikost volné a použité paměti RAM. Obvyklá volba -h přepne na zobrazení v KiB/MiB/... místo bajtů.

```
$ free -h
              total        used         free       shared    buffers     cached
Mem:           7,7G         6,9G         759M         871M         302M         3,0G
-/+ buffers/cache:
Swap:          15G          141M          15G
```

7.2. Programy pro dohled

Jistě nechcete, aby vám např. spadlý docházkový web oznamoval váš šéf. Proto existuje spousta aplikací, které vás dovedou informovat na zadaný email/chat o splnění určitých podmínek jako > 90% zatížení CPU, zaplnění disku, spadlý démon ap.

Tyto aplikace také údaje pomocí agentů nebo veřejných protokolů v pravidelných intervalech sbírají a nabízejí z nich různé tabulky a grafy.

Bohužel není v možnostech této učebnice projít jednotlivé produkty, proto jen stručně uvedeme několik nejznámějších systémů s naším subjektivním hodnocením.

- Nagios – nejstarší, nejznámější a nejotřesnější. Nabízí zastaralý vzhled webové UI a složitou konfiguraci pomocí souborů
- Icinga – fork Nagiosu, moderní velmi dobře navržené web UI
- Cacti – vyniká ve vytváření grafů z nasbíraných dat. Ovládáním a konfigurací patří ke snadnějším.
- Zabbix – mocný, ale i velmi složitý software. Nulová podpora pro Debian/Ubuntu.
- Munin – jednoduchý monitorovací a sběrací nástroj s grafy. Rychlá a snadná instalace i nastavení. Rychlý výsledek.
- Monit a M/Monit – dvojice open-source a komerčního nástroje. Monit je sběrací agent, který může volitelně posílat data do centrálního M/Monit serveru. Jednoduchá konfigurace. Doporučujeme, pokud jste ochotni zaplatit mírný licenční poplatek.

7.3. Logování

Logování je nesmírně důležitou částí každé aplikace a systému, jakmile dojde na problémy. Ale i bez nich bychom měli pravidelně prohlédnutí logů věnovat čas. Pohled do logů může odhalit vážné potíže.

Aplikace na linuxovém OS mohou logovat dvěma způsoby

1. aplikace sami vytváří a případně i archivují a rotují textové log soubory
2. aplikace logovací zprávy posílají do centrálního Syslog, který všechny přijaté logy spravuje a obvykle je ukládá také jako textové soubory

Aplikace často dovedou logovat oběma „cestami“. Obvyklá složka pro oba zdroje logů je `/var/log/`. Pro informace o logování aplikací se obraťte na jejich dokumentaci.

7.3.1. Syslog

Asi mnohými správci preferovaný způsob je sjednocené logování Syslog standardem (RFC 1364). (V Ubuntu jeho moderní implementace [RSYSLOG](#)). Syslog API nabízí programátorům aplikací knihovnu pro logování, takže ji nemusí ve své aplikaci řešit sami.

Formát syslog souboru je jednoduchý. Zpráva se skládá z těchto částí:

- druh zprávy (facility) – „od koho“
- důležitost (severity) – od 0 (emergency nebo emerg), 1 (alert), 2 (crit), 3 (err), 4 (warning), 5 (notice), 6 (info), po nejméně důležitá 7 (debug)
- čas (timestamp) – čas přijetí
- jméno/IP odesílatele
- zpráva samotná

Ukázka syslog souboru

```
Jan 4 07:16:09 jell-nb kernel: [195311.761217] [UFW BLOCK] IN=wlan0 OUT= MAC=0
Jan 4 07:17:01 jell-nb CRON[13901]: (root) CMD ( cd / && run-parts --report
Jan 4 07:18:14 jell-nb kernel: [195436.406373] [UFW BLOCK] IN=wlan0 OUT= MAC=0
Jan 4 07:19:10 jell-nb dhclient: DHCPREQUEST of 192.168.123.104 on wlan0 to 19
Jan 4 07:19:10 jell-nb dhclient: DHCPACK of 192.168.123.104 from 192.168.123.1
Jan 4 07:19:10 jell-nb dhclient: bound to 192.168.123.104 -- renewal in 1320 s
Jan 4 07:19:10 jell-nb NetworkManager[1004]: <info> (wlan0): DHCPv4 state chan
Jan 4 07:19:10 jell-nb NetworkManager[1004]: <info> address 192.168.123.104
Jan 4 07:19:10 jell-nb NetworkManager[1004]: <info> prefix 24 (255.255.255.0
Jan 4 07:19:10 jell-nb NetworkManager[1004]: <info> gateway 192.168.123.1
```

Pokud byste narazili ještě na klasický Syslog, tak má konfiguraci v `/etc/syslog.conf`. RSyslog v Ubuntu používá

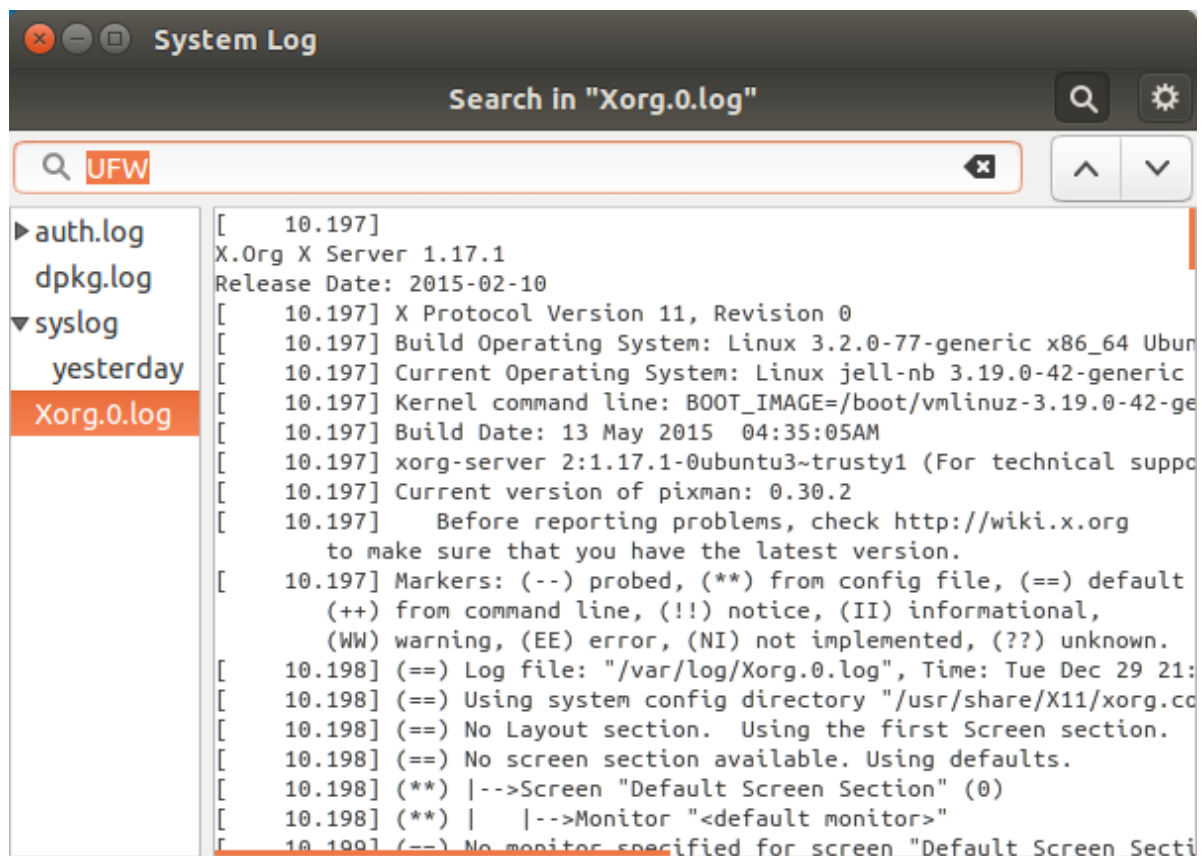
- `/etc/rsyslog.conf` a
- `/etc/rsyslog.d/`

Pokud není zkonfigurováno jinak, pak zprávy spadnou do souboru `/var/log/syslog`.

7.3.2. Prohlížeče logů

7.3.2.1. System Log Viewer (GUI)

Nainstalovaný v grafickém Ubuntu. Splní základní potřeby.



7.3.2.2. Glogg (GUI)

Nutný dodatečně nainstalovat, ale poradí si i obrovskými soubory (několik GB).

The screenshot shows a log viewer window titled "glogg_latest.log - glogg". The main pane displays a log file with various entries, including timestamps, log levels (DEBUG, INFO), and file names. A search bar at the bottom contains the text "(DEBUG.*logdata.cpp.*indexingFinished|New data)". Below the search bar, it indicates "6750 matches found. Search is auto-refreshing...". The search results are displayed in a list below, with several entries highlighted in blue, including "New data on disk" and "Entering LogData::indexingFinished" messages.

7.3.2.3. Komerční správci logů

V posledních letech vznikla řada velmi povedených komerčních správců logů s pokročilým vyhledáváním, nádherným webovým rozhraním, spoustou funkcí pro lokální provoz i cloudově založených. Namátkou např.:

- LogMX – desktopová aplikace pro vizualizaci, vyhledávání a sledování logů. Rychlé, snadné ovládání, schopnost pracovat s obrovskými soubory
- Elasticsearch/Logstash/Kibana
- Loggly
- Papertrail