STŘEDNÍ ŠKOLA PRŮMYSLOVÁ A UMĚLECKÁ OPAVA

PŘÍSPĚVKOVÁ ORGANIZACE

STUDIJNÍ TEXTY K PŘEDMĚTU

Operační systémy

GNU/LINUX

v. 5.1.2015

Obsah

OBSAH **2**

1 CHAREKTERISTIKA GNU/LINUX **5**

1.1 HISTORIE 5
1.2 DISTRIBUCE 6

2 STRUKTURA GNU/LINUX **6**

2.1 JÁDRO 6
2.2 SOUBOROVÝ SYSTEM 7
2.3 ADRESÁŘOVÁ STRUKTURA LINUX 7
2.4 UŽIVATELÉ A SKUPINY 8

3 OVLÁDÁNÍ LINUXU **9**

3.1 PŘÍKAZOVÝ ŘÁDEK 10
3.2 ROOT SUDO 10
3.3 TYPY PRO TERMINAL 11
3.3.1.1 Kopírování 11
3.3.1.2 Automatické doplňování 11
3.4 PŘIHLÁŠENÍ A ODHLÁŠENÍ 12
3.5 KONZOLY 13
3.6 KONTROLA IDENTITY 13
3.7 PRÁCE S ADRESÁŘI A SOUBORY 14
3.7.1 Roury (pipe |) *18*
3.7.2 Typ souboru *18*
3.7.3 FTP *19*
3.7.4 Práce s archivy *19*
3.7.4.1 TAR 19
3.7.4.2 ZIP 19
3.7.4.3 RAR 20

4 PRÁCE S PROCESY (PROGRAMY) **20**

4.1 INFORMACE O PROCESECH 20
4.2 ZABÍJENÍ PROCESŮ 21
4.3 PRIORITA PROCESU 21

5 PRÁCE S UŽIVATELI **22**

5.1 PŘIDÁVÁNÍ A ODSTRAŇOVÁNÍ UŽIVATELŮ 22

6 VYTVÁŘENÍ SKUPIN **23**

7 NASTAVOVÁNÍ UŽIVATELSKÝCH PRÁV **23**

7.1 NASTAVENÍ PŘÍSTUPOVÝCH PRÁV 25
7.2 ZMĚNA VLASTNÍKA 26
7.3 ACL 26

8 ALIASY **28**

8.1 NASTAVENÍ ALIASU 28

9 INSTALACE LINUXU **29**

9.1 INSTALACE ZE ZDROJOVÝCH KÓDŮ 29
9.1.1 NEVÝHODY 30
9.1.2 VÝHODY 30
9.2 INSTALACE Z BALÍČKU 30
9.3 BALÍČKOVACÍ SYSTÉMY 30
9.3.1 APT 30
9.3.2 RPM 30
9.4 POSTUP INSTALACE 31
10 UŽITEČNÉ PŘÍKAZY 31
11 SKRIPTY 32
11.1 ZPŮSOB ZÁPISU 32
11.2 PROMĚNNÉ 33
11.3 READ 33
11.4 IF 34
11.5 PODMÍNKY 34
11.5.1 Pro řetězce 34
11.5.2 Pro čísla 34
11.5.3 Pro soubory 35
11.6 ŘETĚZENÍ PODMÍNEK 35
11.7 CYKLUS FOR 36
12 WEBMIN 36
12.1 STAŽENÍ A INSTALACE 36
12.2 PŘIHLÁŠENÍ 36
13 SSH 37
13.1 INSTALACE SSH SERVERU 37
13.2 VZDÁLENÝ PŘENOS SOUBORŮ 37
13.2.1 Graficky z Windows 37
13.2.2 Použití příkazové řádky 37
13.3 VZDÁLENÉ PŘIPOJENÍ NA SSH 38
13.4 SSH KLÍČE 38
13.4.1 Přihlášení a kopírování pomocí klíče 38
13.4.2 Omezení vzdáleného přístupu 39
14 CRON 39
14.1 SYNTAXE CRONU 39
14.2 CRON OMEZENÍ 40
14.3 CRON PRO ROOTA 41
15 X11 A SSH 41
16 SAMBA 42
16.1 INSTALACE SAMBY 42
16.1.1 Sdílení adresáře 42
17 SQUID 43
18 BIND 43
19 LIGHTTPD 43

20 PFSENSE 43

21 ZABBIX 43

22 FREENAS 44

SEZNAM OBRÁZKŮ 45

1 CHAREKTERISTIKA GNU/LINUX

Linux je moderní operační systém unixového typu. Jeho jádro vytvořil Linus Torvalds za pomoci vývojářů z celého světa. Linux je vyvíjen pod licencí GNU (GPL) - ***General Public License***, takže jeho zdrojový kód je volně k dispozici všem uživatelům, mohou jej používat, kopírovat, pozměňovat a distribuovat. Přitom žádná jeho část nesmí být zatížena licencí nekompatibilní s GPL. Patří k otevřeným ***Open Source*** systémům.

- ***Linux*** ideově vychází z Unixu, používá svůj vlastní zdrojový kód, ale má stejné aplikační rozhraní i filozofii. Díky tomu ***je kompatibilní s ostatními Unixy*** na úrovní zdrojových kódů, má i stejné aplikace a nástroje.

- ***Linux podporuje souběžnou práci více uživatelů***, z nichž každý může spouštět libovolný počet programů.

- ***Linux byl upraven (portován) k použití na spoustě platforem*** –PC, Apple Macintosh, kapesních počítače a další.

- ***Linux*** se ovládá z příkazové řádky , ale dnes ***má i grafické rozhraní.*** K nejpoužívanějším patří KDE a Gnome, jejichž uživatelská přívětivost i grafická propracovanost patří ke špičce.

- ***Pod Linuxem byl vyvinut velký počet aplikací,*** k dispozici je mimo jiné kvalitní prohlížeč Mozilla Firefox, komplexní kancelářský balík OpenOffice.org, na grafiku je výborný Gimp, multimédia si přehrajete v Mplayeru a řada dalších.

- ***Linux má rozsáhlou podporu*** na internetu je také spousta informací – publikované články, ucelené tutoriály, cenným zdrojem informací je i archiv diskusí. Nicméně nejdříve byste se asi měli naučit, jak a kde hledat linuxové informace. Pro začátek doporučuji české stránky www.abclinuxu.cz nebo www.root.cz .

1.1 Historie

Počátky Linuxu se datují na počátek devadesátých let minulého století, kdy Linus Torvalds používal operační systém Minix. Nebyl spokojen s jeho výkonem a možnostmi, proto začal v roce 1991 psát svoje vlastní jádro. Jeho vývoj oznámil na usenetu v konferenci comp.os.minix. Projekt byl od počátku koncipován jako svobodný, kdokoliv se mohl zapojit a přispět. Nový systém byl pojmenován Linux (Linusův Unix). V té stejné době projekt GNU obsahoval všechny potřebné nástroje pro operační systém (systémové i uživatelské programy) a proto bylo nově vznikající jádro přizpůsobeno pro GNU a vznikl kompletní operační system GNU/Linux. Tento pojem se většinou zkracuje na Linux. První verze byly úzce spjaté s architekturou x86, ale časem došlo k portaci na spoustu jiných architektur.

1.2 Distribuce

Projevíte-li zájem vyzkoušet si Linux, najednou zjistíte, že Linux není jen jeden, že si můžete vybrat mezi Red Hatem, Mandrivou, SuSE či Debianem. U Windows se Vám to nemůže stát, dodává je jediná firma a tou je Microsoft. GNU/Linux může dodávat kdokoliv. A tak vznikly desítky či stovky profesionálních, ale i amatérských distribucí. Množství distribucí vzniklo úpravou jiné stávající distribuce, hovoříme o tzv. ***based distribucích***, např. Debian-based. V podstatě všechny distribuce vycházejí buďto z Slackware, Debianu nebo Red Hatu. Distribuce můžeme členit podle nejrůznějších kritérií:

Podle distributora

- dodávané komerční firmou, která dodává i technickou podporu –*Red Hat, Mandriva, Novell či TurboLinux*
- tvořené komunitou - *Debian, Fedora, OpenSUSE, Slackware, Gentoo*

Podle účelu použití

- pro kancelářskou práci či domácí použití
- pro budování internetových serverů
- zcela univerzální
- jednoúčelové (přehrávání DVD, síťové komponenty).

Podle cenové náročnosti

- některé si můžete stáhnout z internetu zcela zdarma (free software)
- komerční verze pro podniky stojící tisíce (licenční software)

U placené verze, platíme za modul, který příslušná firma dopsala. Jádro je zdarma.

2 STRUKTURA GNU/LINUX

Operační systém se skládá z několika částí. Nejhlouběji je ***jádro,*** které zabezpečuje spolupráci s *hardware*m (paměť, procesor, síťové a zvukové karty, pevné disky, apod.) a poskytuje různé služby *procesům*.

2.1 Jádro

Jádro(kernel) je základem systému. Jádro řídí ***procesy*** (spuštění, ukončení, komunikace, uspání, přístup k zařízením,...), dále spravuje systém souborů, přiděluje a chrání paměť, odkládá dočasně nepoužívanou paměť (***swap***). Ještě má na starosti přidělování CPU

času procesům a určování priorit. V původních verzích Linuxu bylo ***jádro monolitické***, což znamenalo, že pro získání podpory jiného zařízení se jádro muselo překompilovat. Dnešní ***jádro*** je ***modulární,*** potřebné ovladače se ve formě modulů - knihoven - mohou do jádra nahrát, případně z něj zase odstranit. Verze jádra se označují 2.4.x (řada 2.4), 2.6.x, např. 2.6.10 - většina běžných linuxových distribucí používá jádro řady 2.6. Sudé verze jsou stabilní, liché verze (2.5.x) jsou vývojové.

2.2 Souborový system

Z předchozího studia víte, že informace se v počítači ukládají na disk. Je nutné zajistit, abychom věděli, kde jsou uloženy a jak je v případě potřeby vyhledáme. Abychom uložená data vždy mohli vyvolat v nezměněné podobě, o to se stará software zvaný ***souborový system.*** Ve Windows jste používali např. FAT32, NTFS – moderní OS Linux používá Ext2 nebo Ext3. Co přesně máme na mysli? Souborový systém je struktura souborů, jejich atributů a dalších informací potřebných pro práci se soubory (tzv. ***metadata***).

Úkol k textu

V souvislosti se souborovým systémem si zopakujte pojmy - soubor, adresář, program.

2.3 Adresářová struktura LINUX

Hlavním diskem je vždy adresář "/", říká se mu kořenový (root). Důležitým poznatkem je, že lomítka v cestě k souborům či adresářům nejsou zpětná (na to jste možná byli zvyklí), ale přímá.

Adresář "/" obvykle obsahuje následující složky:

/boot - jádro systému, zavaděč systému LILO
/dev - speciální soubory (zařízení, devices) např. `/dev/dsp` je obvyklý pro zvukovou kartu
/etc - konfigurační adresář
/home - kořen domovských adresářů
/lib - základní systémové knihovny
/usr - systémové soubory (velmi zjednodušeně)
/usr/include - hlavičkové soubory knihoven **C**
/usr/man - manuálové stránky
/var - konfigurační a další soubory (ftp, www,…)
/var/log - logovací soubory
/var/spool - pošta, tisk, ...
/mnt/ dočasné připojení diskových oblastí, např. chceme-li přistupovat k souborům na disketě, připojíme ji do **/mnt/floppy ;** obdobně připojení CDROM **do /mnt/cdrom** umožní zpřístupnění dat na CD

K zařízením i adresářům se v linuxu přistupuje jednotně ("vše je soubor").

Každý soubor/adresář (adresář je speciální soubor) má svá přístupová práva, tzn. že nějaký uživatel či skupina uživatelů jej může nebo nemůže číst, zapisovat do něj, příp. jej spouštět. Změnou přístupových práv se budeme zabývat později. Souborový systém se připojuje příkazem *mount* a odpojuje příkazem *umount.* Abychom se při každém spuštění nemuseli zabývat tím, co se kam připojí, je vše popsáno v souboru **/etc/fstab** a většina distribucí si všechno nastaví při instalaci, uživatel se nemusí o nic starat. **Proces** je zjednodušeně řečeno běžící uživatelský nebo systémový program. Jednotlivé procesy jsou charakterizovány unikátním číslem procesu (**PID**). Proces vzniká duplikací rodičovského procesu (hlavní systémový proces **init** má PID 1. Procesy spolu komunikují – otec(zdrojový proces) předává při startu synovi (následnému procesu) data, avšak syn otci data předat nemůže!!! Důležitým nástrojem pro komunikaci procesů je **roura**, pomocí níž lze jednosměrně předat data. Procesy mohou komunikovat ještě dalšími způsoby (např. sdílenou pamětí).

2.4 Uživatelé a skupiny

Uživatel, skupina (/etc/passwd,/etc/group)

Chci-li pracovat v OS Linux, je nutné mít k dispozici uživatelský účet. Existují dva typy účtů:

• **root** – superuživatelský bez jakýchkoli omezení – vytvoří se při instalaci

• **běžný účet** – je vytvořen rootem a jsou mu nastavena pravidla pro činnost v prostředí OS

Root, neboli superuživatel je uživatel s kompletními pravomocemi. Má právo provádět instalaci programů, globální změny nastavení a jiné speciální akce v systému, které většinou nelze provést jako běžný uživatel. Pracovat na počítači jako **běžný uživatel** má své výhody, např. z důvodu omezených pravomocí se snadno brání virům (viry se nemohou vetřít do systému, protože prostě nemají přístupová práva pro zápis do souborů, které běžný uživatel potřebuje jen číst, případně spouštět).

Group je skupina uživatelů. Skupiny jsou výhodné k hromadnému definování práv pro více uživatelů, např. přidáním uživatele do skupiny *cdrom* mu umožníme čtení kompaktních disků. Jeden uživatel může patřit do více skupin. Všichni uživatelé jsou definováni v souboru ***/etc/passwd***. Každý řádek definuje jednoho uživatele. Obvyklý první řádek tohoto souboru je uživatel "root":

```
root:x:0:0::/root:/bin/bash
```

Jednotlivá pole oddělená dvojtečkou popořadě znamenají:

- uživatelské jméno

- `x` znamená, že zakódované heslo je uloženo v jiném souboru (`/etc/shadow`), oddělením hesel do jiného souboru se zvyšuje bezpečnost systému, protože `shadow` není čitelný pro běžné uživatele)
- **UID** (user ID) je identifikační kód uživatele (uživatel root má UID 0)
- **GID** (group ID) je identifikační kód skupiny, taktéž skupina root má GID 0.
- prázdná položka je místo pro poznámku (často se sem píše celé jméno uživatele).
- domovský adresář (u superuživatele obvykle `/root`)
- příkaz programu spuštěného po úspěšném přihlášení, většinou tedynějaký shell - v tomto případě `/bin/bash`

Skupiny najdeme v souboru **`/etc/group`**, na prvním řádku je opět root (tentokráte skupina).

```
root::0:root
```

Pole oddělené dvojtečkami:

- jméno skupiny,
- zakódované heslo (prázdné znamená bez hesla),
- **GID** (identifikace skupiny ; 0 znamená, že je to skupina root)
- seznam uživatelů patřících do skupiny (oddělení čárkou), do skupiny navíc patří uživatelé, kteří mají skupinu uvedenou jako svoji primární v `passwd` souboru.

Úkol k textu.

Zopakujte si základní pojmy

GPL/GNU	Open Source system	Distribuce
Jádro monolitické	Jádro modulární	Program
Proces	Démon	Souborový system
Fstab	Mount	Umount
Root	Uživatel	Skupina

3 OVLÁDÁNÍ LINUXU

Komunikace s operačním systémem Linux může probíhat dvěma způsoby:

- prostřednictvím příkazové řádky – CLl (Command Line Interface)
- v grafickém režimu – GUI (Graphical User Interface)

3.1 Příkazový řádek

Uživatel komunikuje s operačním systémem pomocí programu zvaného shell. Uživatel zadává příkazy z klávesnice a tento program je předává operačnímu systému k vykonání. Existuje celá řada „shellů“. Většina linuxových systémů má jako výchozí nainstalován ***shell bash.*** Prostředí shellu je charakterizováno tzv. shellovým promptem, který je ukončený značkou $:

[uzivatel@linuxbox uzivatel]$

Můžeme si zkusit do této řádky něco napsat a potvrdit stiskem klávesy Enter:

[uzivatel@linuxbox uzivatel]$ ahoj

Pokud všechno šlo dobře, objeví se hlášení:

[uzivatel@linuxbox uzivatel]$ ahoj
bash: ahoj: command not found

což znamená: bash nám zpátky hlásí, že příkaz nebyl nalezen (protože textu *ahoj* nerozumí). Pro ovládání Linuxu z příkazové řádky je nutné znát přesně jednotlivé příkazy a jejich význam. Je to náročné, proto se budeme této problematice podrobně věnovat. Příkazy spouštíme takto:

příkaz –přepínače parametry

Pozor!!! Při zadávání jmen a hesel je důležité rozlišovat velká a malá písmena (Linux je *case sensitive*).

3.2 Root sudo

V Linuxu (a Unixu obecně) existuje superuživatel jménem root. Obdobou rootave Windows je Administrátor (správce). Tento správce počítače může dělat naprosto cokoli, takže provádění běžné denní práce pod účtem správce může být velice nebezpečné. Můžete napsat nesprávně příkaz a zhroutit si tak systém. Proto je důležité používat jen uživatele s takovými oprávněními, která jsou zrovna pro daný úkol potřeba. V některých případech je třeba použít účet roota, ale většinou (pro klasickou práci na počítači) úplně postačuje běžný uživatel.

Výchozím nastavením Ubuntu je, že účet roota je zamčen, deaktivován. To znamená, že se nemůžete přihlásit jako root anebo použít příkaz su (příkaz pro změnu identity). Místo toho instalátor nastaví systém tak, aby správu počítače mohl provádět uživatel vytvořený při instalaci (tzn. prvně vytvořený uživatel), a to pomocí příkazu sudo. Ostatní uživatelé příkaz sudo používat nemohou, ale toto právo se jim může udělit.

To znamená, že v Terminálu můžete používat sudo pro příkazy vyžadující rootovská oprávnění. Všechny programy v menu, které budou vyžadovat heslo (např. aplikace v menu Systém → Správa), použijí grafickou podobu sudo (zešedne obrazovka a objeví se políčko pro zadání hesla). Když sudo žádá o heslo, žádá o VAŠE UŽIVATELSKÉ heslo - to znamená, že žádné rootovské heslo není potřeba.

Heslo je standardně uloženo po dobu 15 minut. Po uplynutí této doby budete muset zadat heslo znovu.

Při psaní hesla se vaše heslo nebude vůbec ukazovat na obrazovce, a to ani jako řada hvězdiček (*****) - důvodem je vaše bezpečnost, aby vaše heslo nemohl žádný člověk odpozorovat. Přesto se vámi zadané heslo skutečně píše!

Pro spuštění grafických konfiguračních nástrojů s využitím sudo (s využitím správcovských oprávnění) stačí úplně obyčejně spustit program z nabídky. Program, pokud bude třeba, si o heslo sám řekne.

Pro povolení dalšímu uživateli spravovat počítač pomocí sudo, otevřete Terminál a napište příkaz:

- **sudo adduser <uživatel> sudo** - kde <uživatel> nahradíte přihlašovacím jménem uživatele.

3.3 Typy pro terminal

3.3.1.1 Kopírování

Pro kopírování textu existují v Ubuntu 2 možnosti:

- První možností je označit daný text v prohlížeči, pravé tlačítko myši, kopírovat, potom se přesunout do Terminálu, pravé tlačítko myši, vložit. Stejně to funguje samozřejmě i z Terminálu do textového editoru nebo kamkoliv jinam. Pokud byste chtěli používat klávesové zkratky CTRL + C a CTRL +V v Terminálu musíte stisknout navíc SHIFT (CTRL+SHIFT +C, CTRL+SHIFT+V).

- Druhá možnost je specifická pro Linux a je velice pohodlná. Stačí označit jakýkoliv text v prohlížeči, přesunout se do Terminálu, a zmáčknout prostřední tlačítko myši (kolečko). Stejně to funguje i opačným směrem, z Terminálu kamkoliv jinam. Tento způsob kopírování je velice pohodlný a rychlý.

3.3.1.2 Automatické doplňování

Při práci s Terminálem je vaší nejdůležitější klávesou klávesa Tab. Tato klávesa dělá v Terminálu úplnou magii. Umí totiž inteligentně doplňovat téměř vše, co chcete napsat. Stačí tudíž napsat jen pár písmen a Tab už vše dopíše za vás. Pokud píšete příkaz, napište jen

prvních pár písmen, a stiskněte Tab. Pokud je příkaz již jednoznačný, Tab vám jej sama dopíše. Pokud není jednoznačný, dopíše co nejvíce písmen a počká. Na dvojité stisknutí Tab vám vypíše všechny možnosti, které máte.

Napište tedy na příkazový řádek **dat** a stiskněte TAB. Samo se vám to doplní na date, protože jiný příkaz na dat neexistuje. Pokud na další řádek napíšete who a stisknete Tab, tak se nic nestane. Je totiž více příkazů, které začínají na who. Pokud stisknete TAB podruhé, vypíše vám to možnosti: who, whoami, whois. Po dopsání písmenka a (tzn. na whoa) a stisknutí Tab, pak už jednoznačně doplní na whoami.

Stejně tak funguje klávesa TAB na doplňování adresářů a souborů. Pokud tedy chcete přejít do adresáře hudba, stačí napsat na příkazový řádek cd hu a stisknout Tab. Pokud nemáte jiný adresář začínající na hu, tak se doplní na cd hudba. Tímto způsobem se dobře píší názvy adresářů obsahující mezery, které byste jinak museli uzavírat do uvozovek nebo apostrofů. Klávesa Tab je jeden z nejmocnějších nástrojů příkazové řádky, který začátečníci často podceňují. Rapidně ovšem urychluje vaši práci a chrání vás před překlepy. Mnoho pokročilých uživatelů ji používají častěji, než kteroukoliv jinou klávesu. Naučte se ji určitě používat, je to v příkazové řádce váš nejlepší přítel.

3.4 Přihlášení a odhlášení

- **Login [user] –** přihlásí uživatele „user“, většinou ještě požaduje druhý údaj (password – heslo). Musíme ho psát "poslepu", neboť se většinou z důvodů bezpečnosti nezobrazují ani hvězdičky.

- **Logout** – odhlásí aktuálního uživatele

- **su [-] [user]** dočasně změní uživatele (bez uvedení user se jedná o změnu na superuživatele), příkazem `exit` ukončíme `su`

Příklad

Jsem uživatel Pepik a chci se dočasně změnit na roota.

příkaz : **su** *enter* objeví se password: požadavek na heslo administrátora
prompt: [root@localhost] $: z běžného uživatele se stal administrátor

příkaz : **exit** *enter* spadnu zpět na běžného uživatele Pepik
prompt: [pepik@localhost]$

Jsem uživatel root a chci se dočasně změnit na běžného uživatele Pepik.

příkaz :	**su pepik** *enter* přehodí se mi práva Pepika, zůstanu ve vlastní složce
prompt:	[pepik@localhost root] $: z uživatele root se stal běžný uživatel Pepik
příkaz :	**su - pepik** *enter* přehodí se mi práva Pepika a přesune mě i do jeho složky
prompt:	[pepik@localhost pepik] $: z uživatele root se stal běžný uživatel Pepik

3.5 Konzoly

Je možné, aby na jednom počítači současně pracovalo víceuživatelů? Ano, **jen u LINUXU!!!** Každému uživateli přidělíme jednu konzolu a mezi nimi se můžeme přepínat. Takže můžeme současně pracovat jako root i jako běžný uživatel. To je zvlášť výhodné při nastavování práv pro uživatele, protože si můžeme okamžitě vyzkoušet správnost nastavení. Jak na to?

V Linuxovém systému je 12 virtuálních terminálů (jako byste měli 12 počítačů). Prvních 6 je textových ostatní jsou používány pro grafické rozhraní. Přepínají se kombinací kláves **CTRL + Alt + F1 až F12** (F7 – F12 se používají pro grafiku).

- **passwd** - nastavení hesla

  Old passwd ..... napíšu staré heslo (nic se neobjevuje)
  New passwd ..... napíšu nové heslo (nic se neobjevuje)
  Retype new passwd ..... napíšu ještě jednou nové heslo

Provede se změna hesla (minimálně 6 znaků, je vhodné kombinovat písmena a čísla, odlišuje velká a malá písmena).

Příklad

Zadání:	**Řešení:**
Jsem uživatel root, chci změnit heslo uživateli Pepik	**passwd PEPIK**
Jsem uživatel Pepik, chci změnit své heslo	**passwd**

Objeví se:

New password: zadám nové heslo (POZOR nic nevidím)
Retype password: znovu zadám nové heslo, pokud se nespletu, heslo je úspěšně změněno

3.6 Kontrola identity

Abychom zjistili, jakou máme identitu (kdybychom to náhodou zapomněli), stačí se zeptat příkazem:

- **Id –** vypíše název uživatele i skupiny, k nimž patří
- **W** nebo **Who** - vypíše seznam všech aktuálně nalogovaných uživatelů
- **Last** - vypíše kdo, kdy a odkud se nám logoval na počítač

Úkol

Procvičte si nastudované příkazy:

1. Přihlaste se jako superuživatel root
2. Přepněte se na 2. konzolu (CTRL+ALT+F2)
3. Přihlaste se jako uživatel student
4. Vypište si aktuální přihlášené uživatele
5. Přepněte se na 1. konzolu
6. Ověřte si, že jste root
7. Přepněte se opět na 2. konzolu a odhlaste uživatele student
8. Přepněte ze zpět na 1. konzolu
9. Dočasně změňte uživatele na student
10. Vypište seznam všech aktuálně nalogovaných uživatelů
11. Ukončete dočasný účet
12. Pokaždé se v předchozích 2 úkolech přesvědčte o své identitě
13. Vypište přehled posledních logovaných účtů

3.7 Práce s adresáři a soubory

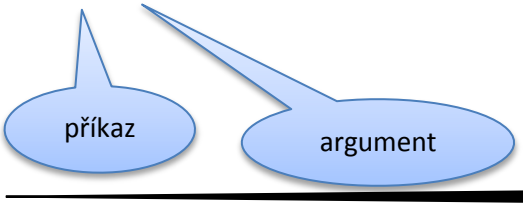
- **ls** nebo **dir -** výpis adresáře

Příkaz je doplněn různými argumenty, přepínači a parametry, jejich přehled a význam můžeme vypsat s využitím nápovědy **man ls**. Parametry se vypisují nejčastěji s jednou čárkou, např **ls –a**

příkaz – určuje, co se má dělat
přepínače – určují, jak se to má dělat
argumenty – určují, s čím se to má dělat

Např.:

- **ls –l /etc** – vypíše adresář /etc v širokém formátu



- **ls –la /etc** -vypisuje všechny soubory z adresáře /etc i skryté soubory

Pokud hledáme určité slovo a nevíme ve které je nápovědě, zadáme příkaz **man –k „klíčové slovo“** vypíše seznam nápověd, v nichž se dané klíčové slovo nachází.

- **cd** – změna adresáře
- **cd /home** -otevře adresář home, který je umístěn v kořenovém adresáři
- **cd knihy** -otevře adresář knihy, který je umístěn v aktuálním adresáři
- **cd ..** -uzavře aktuální adresář a přejde do nadřazeného adresáře
- **cd /** - uzavře všechny otevřené adresáře a přejde do kořenového adresáře /

- **mkdir** - vytvoření adresáře
- **mkdir knihy** - vytvoří adresář knihy, který bude umístěn v aktuálním adresáři
- **mkdir /knihy** - vytvoří adresář knihy, který bude umístěn v kořenovém adresáři

- **rmdir** - zrušení adresáře, adresář musí být prázdný
- **rmdir knihy** - zruší adresář knihy, který je umístěn v aktuálním adresáři
- **rmdir /knihy** - zruší adresář knihy, který je umístěn v kořenovém adresáři
- **rm –r** - zrušení neprázdného adresáře

- **pwd** - vypíše cestu k aktuálně otevřenému adresáři

- **cp** - kopírování souboru (musí mít vždy dva argumenty odkud kam !! Pokud napíšeme tečku, znamená to, kopíruj do aktuálního adresáře)
- **cp knihy/texty.txt /dokumenty** zkopíruje soubor texty.txt z adresáře knihy, umístěného v aktuálním adresáři do adresáře dokumenty, umístěného v kořenovém adresáři

- **mv** - přesun souboru (musí mít vždy dva argumenty odkud kam !! Pokud napíšeme tečku, znamená to, přesouvá se do aktuálního adresáře)
- **mv knihy/texty.txt /dokumenty** - přesune soubor texty.txt z adresáře knihy, umístěného v aktuálním adresáři do adresáře dokumenty, umístěného v kořenovém adresáři

- **rm** - vymazání souboru daného názvu
- **rm knihy/texty.txt** - vymaže soubor texty.txt z adresáře knihy, umístěného v aktuálním adresáři

- **less** - vypsání obsahu souboru daného názvu

- **less knihy/texty.txt** - vypíše obsah souboru texty.txt z adresáře knihy, umístěného v aktuálním adresáři
- **less knihy/texty.txt | more** - vypíše obsah souboru texty.txt z adresáře knihy, umístěného v aktuálním adresáři po stránkách

- **grep „text“** - vyhledávání zadaného textu v souboru, když ho najde, vypíše celou řádku, většinou se používá v kompinaci s jinými příkazy
- **ls –la ¦ grep bys** vypíše všechny soubory v adresáři obsahující slovo *bys*
- **ls –la ¦ grep bys ¦ head – n 2** vypíše všechny soubory v adresáři obsahující slovo *bys*, ale jen první dva

- **du** zobrazí součet velikosti souborů v adresáři

- **cat** - standardní vstup a výstup - vypíše nebo spojí soubory

Standardní vstup (neboli stdin) je místo, ze kterého programy standardně berou data, většinou klávesnice. Standardní výstup (stdout) je místo, kam se data obvykle vypisují, většinou na monitor. Příkaz **cat** bez parametrů nedělá nic jiného, než že čte data ze standardního vstupu a vypisuje je na standardní výstup.

- **cat text.txt** - vypíše obsah soubor text.txt na obrazovku bez možnosti editace
- **cat > text.txt** – otevře soubor pro zápis, na obrazovku se napíše text, po stisknutí kláves CTRL + C se text v souboru nahradí napsaným textem
- **cat >> text.txt** – otevře soubor pro zápis, na obrazovku se napíše text, po stisknutí kláves CTRL + C se napsaný text připojí k dosavadnímu textu v souboru text.txt

Příklad

1. Napište v konzole na klávesnici slovo ***cat*** a stiskněte enter.
2. Napište jakoukoliv větu a po stisku enter se vám zobrazí na obrazovce.
3. Ukončíte stiskem Ctrl+C.

Příklad

1. Vytvořte soubor zdroj.txt pomocí příkazu **cat** do nějž napište tento text:

Kumpel ktory widzial sto teleturniejow
mowi: Z taka glowu wygrasz Milionerow
Pomyslunek masz niezgorszy
w telewizji tyle forsy
wszystko zgarniesz kiedy siedziesz przed kamero
wszystko zgarniesz kiedy siedziesz przed kamero

2. Přesměrujte vstup dat ze souboru zdroj.txt a výstup do souboru cílový.txt

Řešení

$ cat > zdroj.txt příkaz cat vytvoří soubor zdroj.txt a otevře jej pro zápis
Napíšeme text a ukončíme CTRL + C
$ cat zdroj.txt > cilovy.txt příkaz cat vloží načtené informace ze souboru *zdroj.txt* do souboru *cilovy.txt*
$ cat zdroj.txt >> cilovy.txt příkaz cat vloží načtené informace ze souboru *zdroj.txt* do souboru *cilovy.txt* tak, že je přidá na konec stávajícího textu v souboru *cilovy.txt*

Pro kontrolu napíšeme příkaz

- **$ cat cilovy.txt** vypíše obsah souboru cilovy.txt na obrazovku (standardní výstup)

Úkol

Před vlastním přihlášením se vypíše, obvykle obsah souboru /etc/issue, který obsahuje základní informace o systému, např. verzi jádra, název stroje, apod. Pomocí příkazu CAT do tohoto souboru na druhý řádek dopište své jméno a příjmení.

Po úspěšném přihlášení se vypíše, obvykle obsah souboru /etc/motd, jehož obsah doslova znamená "messages of the day" a lze ho použít např. k zobrazování náhodných tipů či hlášek, které do tohoto souboru zapíšeme. Pomocí příkazu CAT do thoto souboru na druý řádek dopište své jméno a příjmení.

- **less** - vypsání obsahu souboru na obrazovku s možností rolování (rolování ukončíme klávesou Q)
- **less /etc/passwd** - zobrazí obsah souboru /etc/passwd
- **touch, cat -** vytvoření nového souboru

1. variant

- **touch název_souboru** - vytvoří soubor zvoleného názvu

2. varianta

- **cat >a.txt, Ctrl+C**

- **date** - vložení data, času do souboru
- **date ¦ cat > název_souboru** - uloží datum a čas do souboru
- **date > název_souboru** - uloží datum a čas do souboru

Úkol

Vytvořte nový soubor identita.txt ve svém domovském adresáři , vložte do něj název naší školy, adresu a na závěr přidejte aktuální datum a čas.

3.7.1 Roury (pipe |)

Přesměrují výstup programu na standardní vstup jiného programu, což umožňuje kombinovat více filtrů do jedné kolony. Klávesová zkratka pro napsání roury je ALTgr + W.

Příklad

Máme seznam jmen v souboru *lidi.txt*, který chceme setřídit podle abecedy, vyřadit duplicity a zobrazit prvních 8 lidí.

Řešení

sort -u lide.txt | head –n 8

Program **sort** setřídí seznam lidí a s parametrem **-u** také data zbaví duplicit. Sort předá data dál a příkaz **head** vytiskne prvních osm řádků.

3.7.2 Typ souboru

Nejjednodušším způsobem, jak můžete určit typ souboru je příkaz **file**. Tento příkaz používá takzvané magic files (jsou v /usr/share/misc/file/), což jsou soubory obsahující charakteristické znaky jednotlivých souborů.

Pomocí příkazu například velmi snadno zjistíte, jakým kodekem je komprimován film.

file *avi

Výstup

- foo1.avi: RIFF (little-endian) data, AVI, 320 x 240, 23.98 fps,
- video: DivX 3 Fast-Motion, audio: MPEG-1 Layer 3 (stereo, 44100 Hz)
- foo2.avi: RIFF (little-endian) data, AVI, 576 x 432, 25.00 fps,
- video: XviD, audio: MPEG-1 Layer 3 (stereo, 48000 Hz)
- foo3.avi: RIFF (little-endian) data, AVI, 320 x 240, 25.00 fps,
- video: DivX 4, audio: MPEG-1 Layer 3 (stereo, 44100 Hz)

Úkol

- Vypište podrobný obsah aktuálního adresáře
- Otevřete adresář /home
- Vytvořte v adresáři /home podadresář soubory
- Do vytvořeného adresáře soubory nakopírujte soubor issue z adresáře /etc
- Zkopírovaný soubor přesuňte z adresáře soubory do domovského adresáře
- Vypište cestu k aktuálně otevřenému adresáři dev
- Vypište na obrazovku soubor /etc/passwd
- Vypište do souboru /student/ucty soubor /etc/passwd
- Vytvořte nový soubor pokus.txt, vložte do něj své jméno a adresu, opatřete aktuálním datem

3.7.3 FTP

- **ftp 192.168.1.1** – připojení k ftp server se zadanou IP, dale jsme vyzvání k zalogování
- **get jmena.txt /home/student/seznam.txt** – stažení souboru jmena.txt z ftp serveru a uložení do adresáře student pod novým názvem seznam.txt
- **put /home/student/jmena.txt seznam.txt** – nahrání souboru jmena.txt z adresáře student na ftp server pod novým názvem seznam.txt

3.7.4 Práce s archivy

3.7.4.1 TAR

Základní program pro archivace v Unixových systémech původně určený pro zálohování na páskové zařízení, samotný tar se dnes nejvíce používá pro nekomprimované uložení více souborů do jednoho a to včetně práv, data a adresářové struktury. Dále TAR podporuje komprimaci takto zarchivovaných dat pomocí další kompresních programů gzip, bzip2 a lzma.

- **tar – cf archiv.tar /home/složka/** - zarchivuje adresář složka do souboru archiv.tar
- **tar – cf archiv.tar film.avi** - zarchivuje soubor film.avi v aktualním umístění do souboru archiv.tar

Přepínač **c** znamená vytvoř, přepínač **f** soubor.

- **tar –xf archiv.tar** – rozbalí soubor archiv.tar do aktualního umístění
- **tar – xf archiv.tar /home/student** – rozbalí archiv.tar do adresáře student

Přepínač **x** znamená extrahuj, přepínač **f** soubor.

3.7.4.2 ZIP

- **zip archiv.zip film.avi** – zkomprimuje soubor film.avi do nového souboru archiv.zip
- **unzip archiv.zip** – extrahuje soubor archiv.zip do aktualního umístění

3.7.4.3 RAR

Program RAR není svobodný software, jedná se o shareware, je nutné jej registrovat. Proto není součástí instalací GNU/Linux. Lze jej nainstalovat příkazem **apt-get install rar**.

- **rar a archiv.rar film.avi** – zkomprimuje soubor film.avi do nového souboru archiv.rar
- **unrar x archiv.rar** – extrahuje soubor archiv.rar do aktualního umístění

4 PRÁCE S PROCESY (PROGRAMY)

4.1 Informace o procesech

Proces vlastně není nic jiného, než program, který je spuštěn a běží. Synonymem pro process může být slovo úloha (task). Linux je od počátku víceúlohový systém, takže v něm může současně běžet více procesů. Stejně jako soubory, tak i procesy jsou vlastněny určitým uživatelem, podléhají přístupovým právům. Na uživatelské úrovni jsou soubory identifikovány svým jménem (které odpovídá jménu spustitelného souboru), ale systém je jednoznačně identifikuje číslem. Jak je vidět na části příkazu **top**.

```
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
13743 root 15 0 98.4m 21m 1660 S 12.3 8.5 1:52.07 X
14057 misak 15 0 61884 45m 19m S 4.3 9.0 0:59.27 konqueror
14047 misak 15 0 80420 36m 22m S 2.3 7.5 1:14.01 amarokapp
```

Prvním procesem, který je spuštěn a má pořadové číslo 1 je init. Všechny další procesy jsou jeho potomky a jeho zabití vede k ukončení celého systému. Další proces má přiřazeno číslo 2, 3, ..., až po jisté číslo n. Jeho hodnota bývá nejčastěji 32768. Skutečná hodnota na vašem systému je uložena v souboru /proc/sys/kernel/pid_max. Pokud systém dosáhne maxima, začne přidělovat volná čísla zase od začátku.

Jeden příkaz byl již uveden a to je příkaz **top**. Je to interaktivní program a slouží k výpisu procesů, který je seřazen podle určitého klíče. Nejčastěji se řadí podle spotřebovaného procesorového času, ale je možné klíč změnit a řadit třeba podle zabrané paměti. Ještě pokročilejší je program **htop** který dokáže procesům posílat signály a podporuje barevné terminály. Klasičtějším způsobem zobrazení procesů je program **ps**.

- **ps** - bez parametrů pouze vypíše procesy spojené s jedním konkrétním terminálem

- **ps x** - volba x zajistí vypsání všech vašich procesů
- **ps ax** - zajistí výpis procesů všech uživatelů na všech terminálech (na něm vidíte, že init má skutečně číslo 1)
- **ps aux** – vypíše navíc vlastníky procesu
- **CTRL +Z** - proces se pozastavi a přepne na pozadí
- **CTRL +C** - zastaví běh procesu
- **fg PID** - pošle úlohu na popředí dle zadaného PID
- **jobs** - vypíše veškeré úlohy a jejich stav
- **bg PID** - pošle úlohu na pozadí dle zadaného PID

4.2 Zabíjení procesů

Dejme tomu, že chceme ukončit program gvim. Pokud si necháme vypsat všechny procesy pomocí **ps aux**, jen těžko budeme hledat náš program. Proto můžeme využít rouru (pipe), skrz kterou pošleme výpis programu **ps** do programu **grep** jež výsledek vypíše na obrazovku.

```
$ ps aux |grep gvim
misak 32128 0.0 3.3 19084 8520 ? Ss 18:34 0:01 gvim
```

A následně můžeme procesu poslat zprávu o ukončení. Jako identifikaci použijeme PID procesu:

```
$ kill 32128
```

Pokud chceme ukončit proces dle jména (vhodné při mnoha procesech jednoho programu) , lze využít program killall:

```
$ killall gvim
```

Pokud náš program nelze ukončit žádným z obyčejných způsobů a dokonce neodpovídá ani na příkaz kill (je zřejmě zacyklený a k vykonání příkazu se nedostane) je možné využít jiné typy signálů vysílaných procesu programem kill. Standardně je zasílán signál 15 neboli TERM, který vyvolává exit. Můžete zkusit využít signálu 9, KILL, který má větší šanci, že nebude blokován.

```
$ kill -9 32128
```

4.3 Priorita procesu

Priorita procesu se nastavuje příkazem nice, který v překladu znamená "ohleduplnost". Hodnota ohleduplnosti je číselné doporučení pro jádro, jak by měl být proces obsloužen ve vztahu k ostatním procesům. Rozsah hodnot pro příkaz nice je -20 až +19. Vysoká hodnota znamená nízkou prioritu (budete ohleduplní vůči ostatním uživatelům) a nízká nebo záporná hodnota znamená vysokou prioritu (nebudete ohleduplní). Nově vzniklý proces dědí hodnotu

ohleduplnosti po rodičích. Uživatel může pouze zvyšovat hodnotu ohleduplnosti svého procesu, ale nesmí jí už poté snižovat. Superuživatel root má neomezenou moc v ovládání ohleduplnosti procesů. Hodnotu ohleduplnosti může uživatel nastavit již při vzniku procesu a při běhu procesů jí může měnit příkazem renice. Příkaz nice přijímá jako argument příkazový řádek a příkaz renice očekává jako argument PID procesu nebo jméno uživatele. Příkazu renice se zadává hodnota ohleduplnosti v absolutním tvaru, u příkazu nice se ohleduplnost jako inkrement, který se odečte nebo přičte od aktuální hodnoty ohleduplnosti.

- **nice -n 8 /bin/velmi_narocna_uloha** - zvýší ohleduplnost o 8
- **renice -3 7623** - nastaví ohleduplnost na 3
- **sudo renice 2 -u jirka** - nastaví ohleduplnost procesů patřící Jirkovi na 2

5 PRÁCE S UŽIVATELI

5.1 Přidávání a odstraňování uživatelů

- **adduser jméno** - vytvoření nového uživatele, musíme být přihlášení jako root

Příklad

Vytvořte uživatele jarda a nastavte mu heslo jarda. Výpisem všech uživatelů zkontrolujte, že uživatel je vytvořen.

Řešení:

adduser jarda <enter>

Objeví se: New password : Napíšu : **jarda**
Retype password : Napíšu : **jarda**

cat /etc/passwd <enter>

- **userdel jméno** - zrušení uživatele jméno, musíme být přihlášení jako root
- **userdell –r jmeno** – zruší uživatele se všemi vazbami

Příklad

Zrušte uživatele jarda. Výpisem všech uživatelů zkontrolujte, že uživatel je zrušen.

Řešení

Userdel –r jarda <enter>
Cat /etc/passwd <enter>

6 VYTVÁŘENÍ SKUPIN

groupadd jméno - vytvoření nové skupiny
gpasswd –a jméno uživatele název skupiny -přidání uživatele do skupiny

Příklad

Vytvořte skupinu ucitele a přidejte do ní uživatele student. Výpisem všech skupin zkontrolujte, že skupina je vytvořena a že student je přiřazen do skupiny studenti.

Řešení:

groupadd ucitele <enter>
cat /etc/group <enter>
gpasswd – a student ucitele <enter>
Cat /etc/group <enter>

- **userdel - r jméno** - zrušení uživatele jméno se všemi vazbami

Úkol

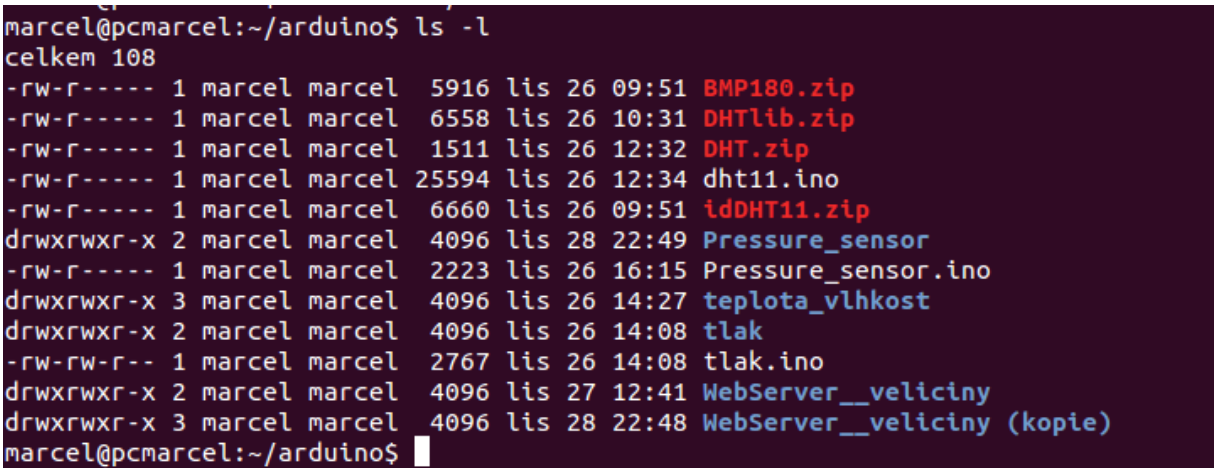
Vytvořte uživatele kaspar s heslem kaspar, melichar s heslem melichar a baltazar s heslem baltazar.

Vytvořte skupinu kralove a zařaďte do ní kaspara, meliachara a baltazara. Úspěšnost ověřte výpisem uživatelů a skupin.

Vytvořte soubor prijmeni_uzivatele do něj přesměrujte výpis souboru passwd a soubor prijmeni_skupiny do kterého přesměrujte výpis souboru groups a zašlete pomoci ftp.

7 NASTAVOVÁNÍ UŽIVATELSKÝCH PRÁV

Linux je víceuživatelský systém a je proto nutné zajistit, aby mi ostatní nemohli bez povolení měnit, číst moje dokumenty. Každý soubor (i adresář) v Linuxu má přidělená práva, která upravují práci se souborem. Příkaz **ls** vypisuje obsah zadaného adresáře. Přidáním přepínače **-l** si zobrazíme podrobnější informace, včetně práv.



obr. 1. Podrobný výpis

Práva jsou uvedená v prvním sloupci, mají 10 pozic.

První znak označuje typ souboru:

- **pomlčka '-'** označuje normální soubor,
- '**d**' adresář,
- '**l**' je symbolický odkaz,
- '**c**' znakové zařízení,
- '**b**' blokové zařízení ,
- '**p**' je pojmenovaná roura

- **Normální soubor** je označen znakem - a je to klasický soubor, na který jsme zvyklí (dokument, aplikace, obrázek, nebo cokoli jiného).
- **Adresář** (directory), označuje znak **d**. V linuxu je adresář speciálním druhem souboru.
- **Symbolický odkaz** (symbolic link, symlink), označujeme znakem **l** a dá se přirovnat k hypertextovému **odkazu** na webu. Ve výpisu vidíme, na který soubor odkaz ukazuje. Odkaz je svázán se jménem souboru, tudíž jeho přejmenování odkaz zneplatní. Můžeme vytvářet odkazy na adresáře. Vytváříme příkazem **ln -s.**
- **Blokové a znakové zařízení** označeno znakem **b** (respektive **c**). Vyskytují se v adresáři /dev a jsou to souborové reprezentace blokových (pevný disk) a znakových (terminál) zařízení, která jsou připojena k počítači. Vytváříme ručně příkazem **mknod**, nebo se o ně stará udev, či starší devfs.
- **Pojmenovaná roura** (named pipe) – pokud jste zapomněli, co to vlastně roura je,vraťte se v tomto textu výše.

Za typem souboru je devět znaků, které symbolizují práva. Vždy jsou vypsány v pořadí právo pro:

- čtení '**r**'- read

- právo pro zápis '**w**'- write
- právo pro spuštění '**x**'- executable

Písmenko znamená přístup povolen, pomlčka znamená přístup zakázán.

Na úrovni adresářů značí:

- **'r',** že je možné vypsat seznam souborů
- **'w'**, že je možné do adresáře soubory přidávat nebo je z něj odebírat
- **'x'**, že adresář smí být součástí cesty

První trojice pro vlastníka souboru (třetí sloupeček -student).
Druhá trojice pro skupinu (čtvrtý - studenti).
Třetí trojice je nastavení platné pro ostatní.

-rwxrw-rw- 2 student student 23214 Oct 3 14:55 film.avi

Vlastník souboru je uživatel, který jej vytvořil (nebo bylo na něj vlastnictví převedeno příkaze **chown**) a obvykle má největší pravomoce. Skupina usnadňuje sdílení souborů mezi uživateli. Můžeme zvýšit práva lidem jen ze svojí skupiny a ne těm, kteří tam nepatří.

7.1 Nastavení přístupových práv

- **chmod práva název souboru**
- **chmod 646 seznam.txt** – nastaví souboru seznam.txt oprávnění –rw-r--rw

Místo písmenek použijeme číselné kódy – viz následující tabulka pro nastavení práv k souboru **seznam.txt**

4	2	1	4	2	1	4	2	1	
r	w	x	r	w	x	r	w	x	
r	w	-	r	-	-	r	w	-	chmod 646 seznam.txt
r	-	x	r	w	-	-	w	x	chmod 563 seznam.txt

Příklad

Vytvořte soubor dokumenty.txt. Nastavte práva rw-r--r-- k souboru dokumenty.txt

Řešení

chmod 644 dokument.txt

Úkol

- Vytvořte v domovském adresáři vytvořte tyto objekty:

seznam	-rw-r--r--	
obrazek	-rwxrw-r-x	
ucitele	dr-xr--rw-	
odkaz	lrwxrwxrwx	tento odkaz bude odkazovat na ucitele

7.2 Změna vlastníka

- **chown uzivatel soubor** – změní vlastníka souboru
- **chown student seznam.txt** – změní vlastníka souboru seznam.txt na studenta
- **chown .skupina soubor** – změní vlastnickou skupinu souboru
- **chown .ucitele seznam.txt** – změní vlastnickou skupinu souboru seznam.txt na skupinu ucitele

Tím, že změníme vztah uživatele k souboru, přiřadíme mu tím příslušná práva, jak byla nastavena.

Ukázka

chown student /etc/dokument.txt

student se stává vlastníkem souboru /etc/dokument.txt

chown .studenti /etc/dokument.txt

všichni členové skupiny studenti se stávají vlastníky souboru /etc/dokument.txt

Úkol

- Vytvořte uživatele petr a užitele adam. Vytvořte skupinu student a skupinu učitele. Uživatele petr přiřaďte do skupiny student a uživatele adam do skupiny učitele. Jako uživatel petr vytvořte soubor ukol.txt. Zkontrolujte pomocí podrobného výpisu vlastníka a vlastnickou skupinu.
- Nyní změňte vlastníka souboru ukol.txt na adama a změňte také vlastnickou skupinu na učitele.
- Podrobný výpis adresáře zapište do souboru prijmeni.txt a zašlete pomocí ftp.

7.3 ACL

Standardní model oprávnění v Linuxu neumožňuje komplexní nastavení jako ve Windows. Od jádra 2.6.x je ACL implementováno standardně, ale ne všechny distribuce jej podporují. ACL se ukládají do rozšířených atributů příslušných souborů. ACL je plně kompatibilní s klasickým modelem oprávnění a rozšiřuje jej. Každý soubor má 1 ACL s

libovolným množstvím záznamů, seznamy ACL mohou být libovolně dlouhé, každá položka ACL představuje jeden rozšířený systémový atribut.

U klasického oprávnění Linux je možno nastavit oprávnění pro vlastníka, skupinu a všechny ostatní uživatele. Nedostatkem tohoto řešení je nemožnost nastavení odlišných oprávnění pro různé uživatele patřící do položky ostatní. Tuto nevýhodu řeší ACL.

Příklad

- Vytvořme jako uživatel adam soubor ukoly.txt.
- Vlastníkem je adam, vlastnickou skupinou je skupina adam (a všichni kdo do ní patří).
- Nastavme vlastnické oprávnění pro soubor ukoly.txt na 774.
- To znamená adam může rwx, členové skupiny adam mohou rwx a všichni ostatní jen r--.
- Mějmě mnoho dalších uživatelů nepatřící do skupiny adam.
- Chceme aby uživatel petr mohl se souborem ukoly.txt rw- a zároveň ostatním uživatelům zůstalo oprávnění r--.
- Pomocí klasických oprávnění nejsme schopni tohoto dosáhnout.

- **getfacl soubor** – výpis ACL k souboru
- **getfacl ukol.txt** – vypíše ACL k souboru ukol.txt
- **setfacl –m u:uzivatel:prava nazev souboru** – nastaví uzivateli práva (ACL) k souboru
- **setfacl –m u:student:6 ukol.txt** – nastaví uživateli student práva rw- k souboru ukol.txt, uživatel student není vlastníkem ani členem skupiny vlastníka

Ukázka

- Vytvořte uživatele marek, jiri a hanka.
- Jako uživatel marek vytvořte soubor seznam s libovolným textem.
- Nastavte oprávnění pro seznam na rw-rw-r--.
- Zkontrolujte správnost nastavení podrobným výpisem.
- Přepněte se na uživatele jiri a zkuste vypsat text ze souboru seznam.
- Zkuste zapsat text do souboru seznam.
- Pepněte se na uživatele hanka a zkuste vypsat text ze souboru seznam.
- Zkuste zapsat text do souboru seznam.
- Přepněte se zpět na uživatele marek.
- Zkontrolujte nastavení ACL.
- Nastavte ACL pro uživatele hanka k souboru seznam, tak aby uživatel hanka měl oprávění rw-.
- Zkontrolujte oprávnění podrobným výpisem.
- Zkontrolujte ACL.

Postup

- **adduser marek**
- **adduser hanka**
- **adduser jiri**
- **su marek**
- **cat>seznam**
- **chmod 664 seznam**
- **ls –l**
- **su jiri**
- **cat seznam**
- **cat>seznam**
- **su hanka**
- **cat seznam**
- **cat>seznam**
- **su marek**
- **getfacl seznam**
- **setfacl –m u:hanka:6 seznam**
- **ls –l**
- **getfacl seznam**

Úkol

- Jako uživatel student vytvořte soubor tabulka.txt s libovolným textem.
- Nastavte souboru tabulka.txt oprávnění rw-rw----.
- Vytvořte uživatele jan a honza, kteří nebudou vlastnící ani členové vlastnické skupiny souboru tabulka.txt.
- Pomocí ACL nastavte pro uživatele jan oprávnění r--.
- Pomocí ACL nastavte pro uživatele honza oprávnění –w-.
- Do souboru prijmeni.txt vypište ACL k souboru tabulka.txt a zašlete na FTP.

8 ALIASY

Alias je aparát umožňující zavádět pro příkazy shellu nová jména (zástupce pro příkazy). Nejčastěji se aliasy používají k zastoupení příkazů pro spouštění programů. Používají se především u složitějších příkazů s řadou parametrů či argumentů, které je problematické si zapamatovat a zdlouhavé zapisovat.

8.1 Nastavení aliasu

- **alias název = "výraz"** – vytvoření aliasu s názvem název, který provede příkaz v poli výraz

- **unalias název** – zrušení aliasu

Pokud napíšeme jméno aliasu na začátek příkazu, shell jej převede na přiřazený výraz v aliasu. Mezi často nastavované aliasy patří například používání barev v programu ls a výpis binárních dat v programu less. Barevný výpis souborů slouží k jednodušší orientaci.

Příklad

Vytvořte alias s názvem uzivatele, který vypíše všechny uživatele z passwd.
Vytvořte alias s názvem pripojení, který se připojí na ftp s adresou 192.168.1.1.
Vytvořené aliasy zrušte.

Řešení

```
alias vypis=”cat /etc/passwd”
alias pripojeni=”ftp 192.168.1.1”

unalias vypis
unalias pripojeni
```

Úkol

- Vytvořte alias s názvem uzivatele, který vypíše prvních 7 uživatelů z passwd seřezených podle abecedy.
- Vytvořte alias s názvem vytvor, který vytvoří 3 soubory s názvem jedna, dva a tri.
- Vytvořte alias s názvem smaz, který smaže soubory s názvem jedna, dva a tri.
- Vytvořte alias s názvem proces, který vypíše procesy, vyfiltruje proces s názvem init a zapíše tento výpis do souboru s názvem proces.txt.

9 INSTALACE LINUXU

Instalaci operačního systému Linux je možno realizovat dvěma způsoby.

- instalace z balíčku (ucelená distribuce)
- instalace ze zdrojových kódů

9.1 Instalace ze zdrojových kódů

Většina programů používaných v Linuxu je k dostání i ve formě zdrojových kódů, které si může uživatel sám upravovat a překládat do výsledné spustitelné podoby. Editovat je lze libovolným textovým editorem.

9.1.1 NEVÝHODY

Pro překlad je třeba určitých zkušeností a znalostí operačního systému. Začínajícím uživatelům se tento způsob instalace ze zdrojových kódů nedoporučuje, protože

- velmi rychle zapomínají co a kam nainstalovali,
- nemají představu, které soubory patří k danému programu,
- nemusí program přeložit vždy úplně správně.

Vznikají problémy s odinstalací programů takto nainstalovaných a velmi často dochází ke kolizím s balíčkovacím systémem. Kompilace také zabere množství času. Při překladu zdrojových kódů do binární podoby dochází k vysokému vytížení systému. U jednodušších aplikací kompilace zabere vteřiny či minuty u složitějších a rozsáhlejších projektů, jako je například KDE, může kompilace zabrat deset a více hodin. Samozřejmě záleží na výkonnosti vašeho hardware.

9.1.2 VÝHODY

Program, který je přeložen přímo pro váš typ hardware by měl běžet (a většinou běží) o něco rychleji. Teoreticky by se měla i zvýšit stabilita.

9.2 Instalace z balíčku

Instalace pomocí balíčků je pohodlnější způsob - jde o již (většinou) kompilovanou aplikaci, zabalenou do jednoho souboru. Některé balíčkovací systémy řeší i závislosti aplikace. To znamená, že stáhnou z internetu (či si jinak vyžádají) ostatní programy a knihovny, které aplikace ke svému běhu potřebuje. Instalace je většinou velmi snadná. Toto je většinou nejpoužívanější způsob instalace nových programů.

9.3 Balíčkovací systémy

V současnosti se nejčastěji používají tyto balíčkovací systémy:

9.3.1 APT

Balíčkovací systém distribuce Debian. Balíčky mají koncovku .deb.

9.3.2 RPM

Velice známý, a hodně používaný systém balíčků. RPM znamená „RPM Package Manager“ (dříve „Red Hat Package Manager“ — Redhatovský manažer balíčků) a kromě samotného Red Hatu se používá především v distribucích jako je Mandriva, Fedora Core nebo Suse. Tudíž je pravděpodobnější, že se s ním jakožto začátečník setkáte.

9.4 Postup instalace

- **apt- get install název balíku** - prohledají se všechny dostupné zdroje a nejnovější verze se nainstaluje
- **apt- get remove název balíku** - odebírání balíčků, odebere vše mimo konfigurační soubory
- **apt- get remove --purge název balíku** - odebere vše včetně konfiguračních souborů
- **dpkg –i balik.deb** – instalace staženého balíku s přípodou .deb

10 UŽITEČNÉ PŘÍKAZY

- **apt-get install balik** - instaluje zadaný balík
- **apt-get remove balik** - odstraní zadaný balík
- **history** – vypíše historii provedených příkazů, jakýkoli příkaz z historie je možné zopakovat takto (n=pořadové číslo příkazu)
- **apt-get update** – aktualizuje seznam aktualizací
- **apt-get upgrade** – nainstaluje aktualizace
- **ln –s umisteni nazev** – vytvoří symbolický odkaz (zástupce) na umístění s názvem nazev

- **locate hledany_soubor** – vyhledání souboru
- **acpi** – stav baterie na notebooku
- **shutdown –h 22:50** – vypnutí počítače ve 22:50
- **sleep 5; ls** – spuštění programu s časovou prodlevou, spustí program ls za 5 s
- **wget www.seznam.cz** – konzolové stahování, do domovské složky se stáhne stránka seznamu

- **nl** – očísluje standardní vstup a na výstup vypíše výsledek
- **ls|nl** – vypíše obsah adresáře a očísluje řádky
- **clear** – smaže obrazovku teminálu
- **wc** – vypíše počet znaků, slov a řádků standardního vstupu na standardní výstup
- **df** – vypíše volné místo na všech discích
- **cal** – zobrazí calendar

- **ifconfig eth0 192.168.1.10 netmask 255.255.255.0** – nastavení ip a masky na rozhraní eth0
- **route add default gw 192.168.1.1** – nastavení výchozí brány

- nastavení DNS provedeme editací souboru **/etc/resolv.conf,** do nějž dopíšeme jeden řádek: **nameserver 192.168.1.111**

11 SKRIPTY

V této části se dozvíte jak používat proměnné a řídící struktury. Také budete schopni napsat své první jednoduché skripty.

11.1 Způsob zápisu

U programování skriptů nepoužívejte diakritiku pro názvy proměnných či funkcí. Na výstupu čeština nevadí, ale u proměnných může způsobovat problémy. Doporučuji používat následující způsob:

- na jednotlivý řádek souboru = jeden řádek kódu
- nad každý řádek kódu = jeden řádek komentáře, v počátcích Vám to hodně pomůže
- pokud chcete na jeden řádek napsat více příkazů, oddělte jej středníkem **;** !

V Bashi se komentáře označují znakem mřížky **#**, všechno za tímto znakem je ignorováno a slouží to především k tomu, abyste ve velkém kusu kódu věděli co dělá která část.

Nyní si řekneme jak si začít psát vlastní skript.

- Pokud máte otevřen terminál, vytvořte si nový soubor **touch skript.sh**. Koncovka **.sh** značí že se jedná o skript, nezáleží na ní, ale jedná se nám o zpřehlednění skriptu od ostatních programů pomocí koncovky.
- Nyní si spusťte libovolný editor pro úpravu textu. Například nano nebo vi.
- Jako první řádek každého bashové skriptu musí být následující řádek:

```
#!/bin/bash
```

- Tento řádek říka interpretu jazyka Bash, že se jedná o skript.
- Nyní musíme skriptu nastavit oprávnění pro spuštění. Chmod 700 skript.sh
- Tento skript se pak v konzoli spustí takto:

```
./skript.sh
```

- Nyní budeme všechny naše kusy kódu a příkazy psát od druhého řadku dolů.

11.2 Proměnné

Pro uchování dat používá každý programovací jazyk proměnné a Bash není výjimkou. Proměnné v Bashi nemají svůj datový typ, proto se do nich dá ukládat cokoliv. Mají pevně definovanou syntax:

```
Promenna=obsahPromenne
```

Deklarace je pouze alokování proměnné a definice je přiřazení hodnoty do této proměnné.

Dbejte na to, abyste používali proměnné, které mají již nějakou hodnotu ! Proměnná se skládá ze svého názvu a obsahu. Název může obsahovat písmena a čísla (nesmí začínat číslem !), které vystihují to, jakých bude proměnná nabývat hodnot Obsah může být cokoliv, textový řetězec, číslo nebo výstup programů. Pomocí následující příkladů Vám demonstruji použití proměnných:

```
        # definice proměnné s deklarací
promenna=”ahoj svete”
        #promennou vypíšeme přes echo
echo „$promenna“
        #ekvivalentní výpis je take takto s obtékáním textu
echo “data $promenna data“
```

11.3 Read

Příkaz **read** slouží k získání dat od uživatele.

```
read promenna
```

Příklady použití příkazu read

```
echo –n „Zadejte svůj vek: “

read vek

echo „Vas vek je $vek“
```

11.4 IF

If je řídící struktura, která slouží k rozhodovací podmínce jestliže něco platí či neplatí dělej. Příklad syntaxe:

```
        # Jednoduchý if

if [podmínka]; then
        příkazy
fi

        #Složený if

if [podmínka]; then
        Příkazy
else
        příkazy
fi
```

Příklad

```
cislo="2"
if [! "$pom" –eq „2"]; then
        echo "Neni to cislo dva"
else
        echo "Je to cislo dva"
fi
```

11.5 Podmínky

Podmínky můžou být následující:

11.5.1 Pro řetězce

- **$promenna** délka řetězce je nenulová
- **-z $promenna** délka řetězce je nulová
- **„$promenna“ = „$promenna2“** řetězce jsou shodné
- **„$promenna“ != „$promenna2“** řetězce jsou různé

11.5.2 Pro čísla

- **„$promenna“ -eq „$promenna2“** čísla jsou shodná
- **„$promenna“ -ne „$promenna2“** čísla jsou rozdílná
- **„$promenna“ -le „$promenna2“** promenna1 ⇐ promenna2
- **„$promenna“ -lt „$promenna2“** promenna1 < promenna2
- **„$promenna“ -ge „$promenna2“** promenna1 >= promenna2
- **„$promenna“ -gt „$promenna2“** promenna1 > promenna2

11.5.3 Pro soubory

- **„$promenna“ -ef „$promenna2“** soubory sdílí stejný i-uzel
- **„$promenna“ -nt „$promenna2“** první soubor je novější než druhý
- **„$promenna“ -no „$promenna2“** první soubor je starší než druhý
- **-e „$promenna“** soubor existuje
- **-d „$promenna“** soubor je adresář
- **-f „$promenna“** soubor je obyčejný soubor
- **-L „$promenna“** soubor je symbolický odkaz
- **-w „$promenna“** soubor je zapisovatelný
- **-x „$promenna“** soubor je spustitelný

Pokud před výraz napíšu vykřičník !, celý výraz se neguje. Takže pokud mám např. -e soubor že testuje jestli soubor existuje, tak ! -e soubor testuje jestli soubor neexistuje.

11.6 Řetězení podmínek

Podmínky se dají spojovat do jedné přes operátor && log. AND a || log. OR. And (&&) znamená a současně. Takže musí platit všechny podmínky, aby byla splněna výsledná podmínka. Místo toho || znamená nebo, takže stačí aby platila jedna z podmínek. Způsob zápisuje je dvojí:

```
if [ -e "$pom" ] && [ -e "$pom2" ] || [ -e "$pom3" ] ; then ...
```

Testuje jestli současně platí test existence souborů v proměnných **pom** a **pom2**. Tyto soubory musí oba existovat, aby platila podmínka. Pokud jeden neexistuje, podmínka nikdy platit nebude. V tomto případě je třetí podmínka zbytečná.

Druhý styl je ve výsledku stejný jako u prvního, ale způsob zápisu mně příjde elegantnější:

- -o znamená nebo, ||, OR
- -a znamená a současně, &&, AND

```
if [ -e "$pom" -a -e "$pom2" -o -e "$pom3" ] ; then ...
```

11.7 Cyklus FOR

For patří k zástupcům cyklů. Použijeme jej tehdy, když chceme zajistit nějaký určitý počet opakování příkazů. Zde je ukázka syntaxe:

```
for (( i=1; i<=5; i++ ))

do

        echo "Promenna i = $i"

done
```

12 WEBMIN

Webmin je webová aplikace napsaná v Perlu, určená pro správu unixového systému, a tedy i Linuxu (ale také např. BSD). Vše, co potřebujete k jeho provozování na straně uživatele, je jen webový prohlížeč s podporou formulářů a tabulek (popř. i Javy, pokud chcete využít modul správce souborů). Webmin jako takový se skládá z jednoduchého webového serveru a ze spousty modulů, které zajišťují dané konkrétní činnosti - např. správu procesů, používání MySQL serveru nebo firewallu.

12.1 Stažení a instalace

Ze stránek výrobce www.webmin.com stáhneme příslušný balíček (pro ubuntu *.deb) a nainstalujeme.

12.2 Přihlášení

Po instalci stačí do webového prohlížeče zadat:

- https://ip_pocitace:10000

13 SSH

SSH poskytuje zabezpečený vzdálený přístup příkazovou řádkou, vzdáleného spouštění grafických aplikací, bezpečného přesunu souborů pomocí protokolů zabezpečeného kopírování (SCP) a zabezpečeného FTP. Dodatečně může také sloužit jako šifrovací tunel pro ostatní aplikace za pomoci přesměrování portů.

SSH nahrazuje starší, nezabezpečené aplikace jako telnet, rlogin a FTP. Tyto aplikace přenášely hesla po internetu bez šifrování, čímž mohla být hesla snadno odposlechnuta a ukradena. Použitím šifrování SSH těmto problémům předchází.

13.1 Instalace SSH serveru

Pokud se chcete bezpečně spojit s vaším počítačem z jiného počítače, musíte si nainstalovat server, který toto spojení dovolí. Poté se můžete připojit i z počítače běžícím na MS Windows (klient), například pomocí programu Putty. Ubuntu používá openssh-server, který můžete nainstalovat jako balík **openssh-server**.

13.2 Vzdálený přenos souborů

13.2.1 Graficky z Windows

Pro přenos souborů mezi počítačem s Windows a počítačem s Ubuntu můžete použít program WinSCP, který můžete (zdarma) stáhnout z http://winscp.net.

13.2.2 Použití příkazové řádky

Pro kopírování souborů z vašeho počítače do jiného se spuštěným SSH serverem, budete potřebovat zabezpečené kopírování (secure copy) - příkaz scp. Použití příkazu by mělo vypadat asi takto:

- **scp soubor uzivatelske_jmeno@ip:cilovy_adresar**

Příklad

- **scp /home/student/film.avi ucitel@192.168.10.2:/home/ucitel/filmy**

Pro kopírování souborů ze vzdáleného počítače na místní disk použijte příkaz:

- **scp ucitel@192.168.10.2:/home/ucitel/film.avi .**

Znak „.“ Znamená, že se soubor bude kopírovat do aktuálního adresáře.

13.3 Vzdálené připojení na SSH

Pro připojení na vzdálený počítač je zapotřebí zadat tento příkaz:

- **ssh uživatel@ip**

Příklad

- **ssh student@192.168.11.9**

13.4 SSH klíče

Kdysi všichni používali k prokazování identity klasické uživatelské jméno a heslo. Nicméně pokud někdo uhodl nebo odposlechl vaše heslo, tak bylo veškeré zabezpečení pryč. Proto SSH nabízí ověření veřejného klíče. To využívá privátní a veřejné klíče namísto jednoduchých hesel.

Je nutno začít vygenerováním dvojice veřejného a soukromého klíče příkazem

- **ssh-keygen**

Poté budete dotázáni, kam mají být klíče uloženy (pouze potvrďte standardní umístění) a k určení vstupní fráze. Vstupní fráze je použita k zašifrování vašeho privátního klíče. Každý, kdo se dostane k vašemu (nezašifrovanému) privátnímu klíči bude mít vaše práva na jiných počítačích.

Pro získání přístupu do vzdálených počítačů musí tyto počítače vašemu veřejnému klíčí důvěřovat. Váš veřejný klíč byl vytvořen společně s novým privátním klíčem. Obvykle bývá umístěn v:

- /home/uživatel/.ssh/

Cílový uživatel musí mít tento veřejný klíč id_rsa.pub (je to řádek ASCII znaků) ve svém souboru autorizovaných klíčů umístěného na cílovém počítači v:

- /home/uživatel/.ssh/authorized_keys

Takže zkopírujte a vložte tento řádek do souboru s autorizovanými klíči.

Pokud při vytváření Vašeho klíče zadáte vstupní frázi, budete při použití klíče na ni dotázáni. Pokud nezadáte, přihlásíte se jen pomocí klíče bez vstupní fráze.

13.4.1 Přihlášení a kopírování pomocí klíče

- **ssh –i klic uživatel@ip**
- **ssh –i /home/petr/id_rsa student@192.168.1.14** – přihlásíme se na vzdálený počítač se zadanou ip, na účet student s použitím klíče uloženém v /home/student
- **scp –i klic soubor uživatel@ip:/adresar**
- **scp –i /home/petr/id_rsa /home/petr/filmy/film.avi student@192.168.1.14:/home/pohadky** -zkopíruje film.avi z lokálního počítače do vzdáleného počítače se zadanou ip na účet student do adresáře pohádky za pomocí klíče

13.4.2 Omezení vzdáleného přístupu

Ověřování pomocí hesla je standardně v Ubuntu zapnuto. Pokud chcete zabránit uživatelům ve vzdáleném přihlašování pomocí hesla, tak v souboru **/etc/ssh/sshd_config** změňte **PasswordAuthentication no**. Nezapomeňte restartovat váš ssh server po změně konfigurace pomocí příkazu(**sudo /etc/init.d/ssh restart**) nebo nověji od verze Ubuntu 8.10 **sudo service ssh restart**).

14 CRON

Cron je softwarový démon, který v operačních systémech automatizovaně spouští v určitý čas nějaký příkaz resp. proces (skript, program apod.). Jedná se vlastně o specializovaný systémový proces, který v operačním systému slouží jakožto plánovač úloh, jenž umožňuje opakované spouštění periodicky se opakujících procesů (např. noční běhy dávkových úloh při hromadném zpracování dat) apod. Každý uživatel si může vytvářet svoje crony sám.

14.1 Syntaxe Cronu

```
* * * * * příkaz/script ke spuštění
- - - - -
| | | | |
| | | | +----- den v týdnu (0 - 6) (Neděle=0)
| | | +------- měsíc (1 - 12)
| | +--------- den v měsíci (1 - 31)
| +----------- hodina (0 - 23)
+------------- minuta (0 - 59)
```

- **crontab -e** - upravit váš crontab soubor nebo vytvořit v případě, že neexistuje

Příklad

Dělej něco každou lichou hodinu každý sudý den
*** 1-23/2 */2 * * root /usr/local/sbin/delej_tady_neco.sh**

Zálohování dat jednu za týden v noci z pátku na sobotu v 0:42
42 0 * * 6 root /usr/local/sbin/zal_data_tyden.sh

14.2 Cron omezení

Nastavení práv pro použití cronu se provádí pomocí souborů **/usr/lib/cron/cron.allow** a **/usr/lib/cron/cron.deny**. Do těchto souborů mapište jednoho uživatelské jméno na řádek. Pokud je k dispozici pouze cron.deny a je prázdný, crontab mohou použít všichni uživatelé.

Příklad

Řádek v souboru crontab níže odstraňuje všechny soubory z /home/someuser/tmp každý den v 18:30 hodin.
30 18 * * * rm /home/someuser/tmp/*

Každou hodinu (v 0 minut):
0 * * * *

Každých 5 minut:
***/5 * * * ***

Každou minutu:
*** * * * ***

00:30 Hodin 1. Ledna, Června a Prosince:
30 0 1 1,6,12 *

20.00 Hod. každý všední den v týdnu, ale jen v Říjnu:
0 20 * 10 1-5

Půlnoc 1. ,10. a 15. v měsíci:
0 0 1,10,15 * *

Ve 12.05 a 12.10 každé Pondělí a 10. každý měsíc:
5,10 0 10 * 1

14.3 Cron pro roota

Uživatel root může také přistupovat do crontab editací souboru **/etc/crontab**. Tento soubor je primárně určen pro systémové spouště (rotace logu a pod.). Má navíc jeden nepovinný parametr, jméno uživatele pod kterým se má script spustit. (Např: **30 18 * * * root /home/petr/muj_script.sh)**

15 X11 A SSH

X11 je protokol pomocí nějž s využitím SSH můžeme vzdáleně spouštět grafické programy. Samozřejmosti je na serveru předpoklad nainstalovaného ssh serveru (například openssh-server).

Nejprve musíme na serveru nainstalovat X client a také budeme potřebovat autorizaci pro X Window. Tyto aplikace nainstalujeme příkazy:

- sudo apt-get install xbase-client
- sudo apt-get install xauth

Zkontrolujeme konfigurační soubor **/etc/ssh/sshd_config**, zda-li máme povolený forwarding. V tomto souboru vyhledáme položku **X11Forwarding**, která musí být nastavená na **yes**.

V domovském adresáři vytvoříme soubor config v cestě

- /home/uživatel/.ssh/config

a zapíšeme do něj následující text:

- ForwardX11 yes

a uložíme. Nyní se můžeme pomocí ssh připojit k serveru za použití přepínače –X.

- **ssh –X uživatel@ip**

Spuštění vzdálené grafické aplikace na serveru provedeme stejně jako při spouštění z terminálu. Všimněme i, že v případě spuštění ze serveru, nikoli z fyzického lokálního počítače, je toto vždy uvedeno v závorce u názvu okna.

16 SAMBA

Samba umožňuje sdílet soubory a tiskárny mezi počítači s operačním systémem Windows a počítači s operačním systémem Unix. Jedná se o implementaci desítky služeb a tuctu protokolů. Samba zřizuje sdílení souborů pro vybrané Unixové adresáře (včetně všech podadresářů). Tyto se zobrazí uživatelům Windows jako normální složky Windows přístupné prostřednictvím sítě. Každý sdílený adresář může mít různá práva přístupu, která překrývají běžná oprávnění v Unixu.

Služby Samby jsou realizovány jako dva démoni:

- **smbd**, který poskytuje sdílení souborů a tiskáren, a
- **nmbd**, který poskytuje překlad NetBIOS na IP adresu. NetBIOS přes TCP/IP vyžaduje určitou metodu pro mapování názvů NetBIOS počítače na IP adresy TCP/IP síti

Konfiguraci Samby je dosaženo editací jednoho souboru (typicky se nachází **/etc/smb.conf** nebo **/etc/samba/smb.conf**).

16.1 Instalace Samby

Nejprve je nutno nainstalovat balík Samba a rozšiřující balíky pro zabezpečení sdílení.

- sudo apt-get install samba
- sudo apt-get install samba-common
- sudo apt-get install python-glade2
- sudo apt-get install systém-config-samba

16.1.1 Sdílení adresáře

Otevřeme konfigurační soubor samby /etc/samba/smb.conf do nějž napíšeme následující text:

```
#globální nastavení
[global]
Workgroup=WORKGROUP
server string = Samba Server %v
netbios name = ubuntu
security = user
map to guest = bad user
dns proxy = no
```

```
#nastavení sdílení
[Filmy]
path = /samba/videa
browsable = yes
writable = yes
guest ok = yes
read only = no
```

Sdílenému adresáři musíme nastavit oprávnění

- **chmod –R 0777 videa**

Následně ve Windows vyhledáme sdílenou složku, která se bude jmenovat Filmy.

17 SQUID

18 BIND

19 LIGHTTPD

20 PFSENSE

21 ZABBIX

22 FREENAS

Seznam obrázků

obr. 1. Podrobný výpis 24