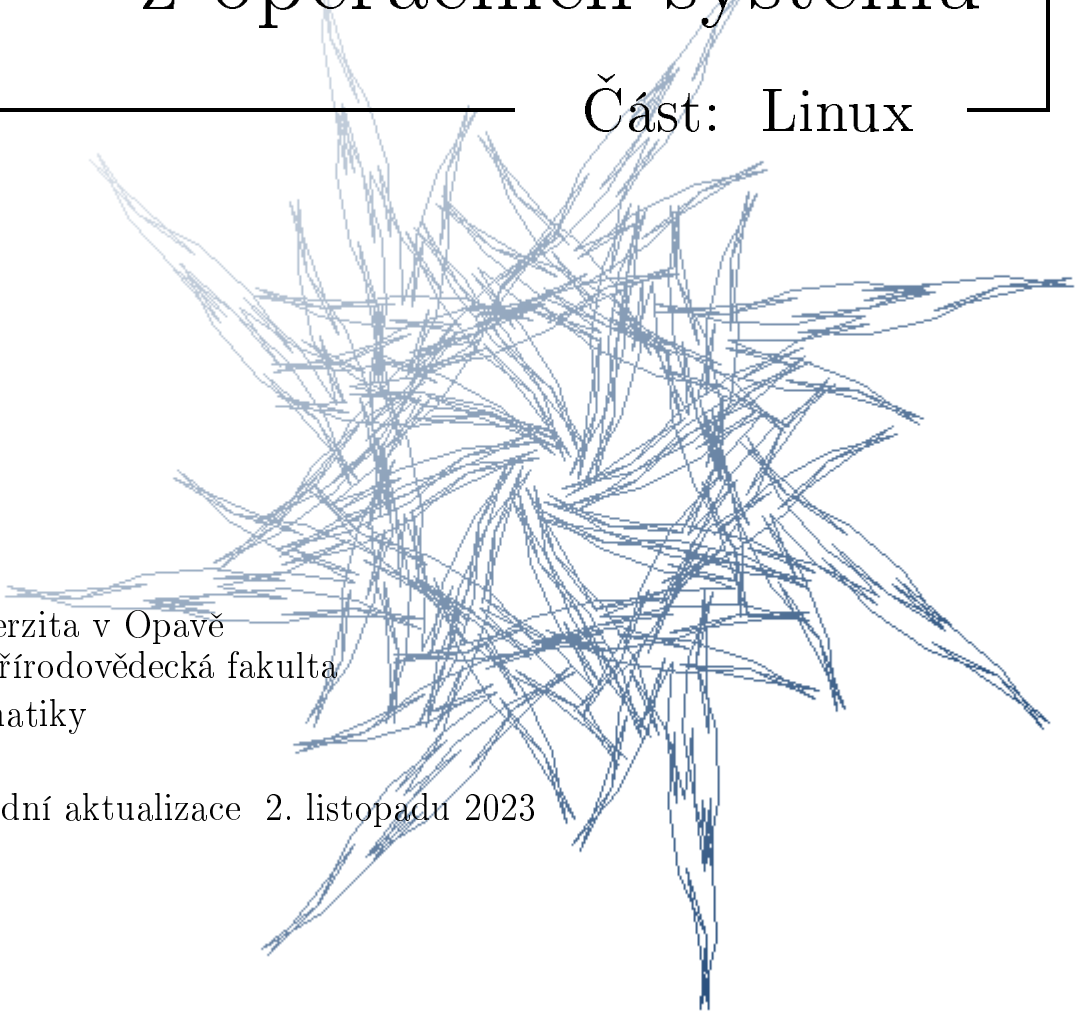
evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
Slezská univerzita v Opavě
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Šárka Vavrečková

Operační systémy I

Praktikum z operačních systémů

Část: Linux



Slezská univerzita v Opavě
Filozoficko-přírodovědecká fakulta
Ústav informatiky

Opava, poslední aktualizace 2. listopadu 2023

Anotace: Tento dokument je určen pro studenty druhého ročníku IVT na Ústavu informatiky Slezské univerzity v Opavě. Používá se při výuce předmětu Praktikum z operačních systémů (nový název: Operační systémy I), přičemž v tomto předmětu se probírají tři základní okruhy:

- úvod do správy Windows,
- bezpečnost a licence,
- úvod do správy Linuxu.

Tento dokument obsahuje studijní látku pro třetí z těchto okruhů – úvod do správy Linuxu.

Praktikum z operačních systémů, Operační systémy I

Část II: Linux

RNDr. Šárka Vavrečková, Ph.D.

Dostupné na: http://vavreckova.zam.slu.cz/pos.html

Ústav informatiky
Filozoficko-přírodovědecká fakulta v Opavě
Slezská univerzita v Opavě
Bezručovo nám. 13, Opava

Sázeno v systému LaTeX

Tato inovace předmětu *Praktikum z operačních systémů* je spolufinancována Evropským sociálním fondem a Státním rozpočtem ČR, projekt č. CZ.1.07/2.2.00/28.0014, „Interdisciplinární vzdělávání v ICT s jazykovou kompetencí“.

Předmluva

Co najdeme v těchto skriptech

 *Rychlý náhled:* Cílem výuky v tomto předmětu je nacvičit si základy ovládání operačních systémů Windows a Linux, v těchto skriptech se zabýváme operačním systémem Linux.

V tomto předmětu jsou studijní materiály rozděleny do tří částí: Windows, Bezpečnost a pořádek, Linux. V tomto dokumentu (třetím v pořadí) se seznámíme se systémy unixového typu (UNIX-like systems), pak se zaměříme na Linux – nejdřív na práci v grafickém režimu, pak si osvětlíme témata ze správy unixových systémů a nakonec se naučíme pracovat v textovém režimu unixových systémů.

Některé oblasti jsou také „navíc“ (jsou označeny ikonami fialové barvy), ty nejsou probírány a ani se neobjeví na testech – jejich úkolem je motivovat k dalšímu samostatnému studiu nebo pomáhat v budoucnu při získávání dalších informací dle potřeby v zaměstnání.

Značení

Ve skriptech se používají následující barevné ikony:

-  *Rychlý náhled* (skript, kapitoly), ve kterém se dozvíme, o čem to bude.
-  *Klíčová slova* kapitoly.
-  *Cíle studia* pro kapitolu nám řeknou, co nového se v dané kapitole naučíme.
-  Nové *pojmy*, značení apod. jsou označeny modrým symbolem, který vidíme zde vlevo. Tuto ikonu (stejně jako následující) najdeme na začátku odstavce, ve kterém je nový pojem zaváděn.
-  Konkrétní *postupy a nástroje* (příkazy, programy, soubory, skripty), způsoby řešení různých situací, do kterých se může administrátor dostat, syntaxe příkazů atd. jsou značeny také modrou ikonou.
-  Pokud si student má *vybrat* jednu z několika možností (například vybrat si jeden UNIXový systém a ten podrobněji popsat), najdeme u jednotlivých alternativ tuto oranžovou ikonu.

-  Některé části textu jsou označeny fialovou ikonou, což znamená, že jde o *nepovinné úseky*, které nejsou probírány (většinou; studenti si je mohou podle zájmu vyžádat nebo sami prostudovat). Jejich účelem je dobrovolné rozšíření znalostí studentů o pokročilá témata, na která obvykle při výuce nezbývá moc času.
-  Žlutou ikonou jsou označeny odkazy, na kterých lze získat *další informace* o tématu. Nejčastěji u této ikony najdeme webové odkazy na stránky, kde se dané tématice jejich autoři věnují podrobněji.
-  Červená je ikona pro *upozornění* a poznámky.

Pokud je množství textu patřícího k určité ikoně větší, je celý blok ohraničen prostředím s ikonami na začátku i konci, například pro definování nového pojmu:

 **Definice**

V takovém prostředí definujeme pojem či vysvětlujeme sice relativně známý, ale komplexní pojem s více významy či vlastnostmi.

Podobně může vypadat prostředí pro delší postup nebo delší poznámku či více odkazů na další informace. Mohou být použita také jiná prostředí:

 **Příklad**

Takto vypadá prostředí s příkladem, obvykle nějakého postupu. Příklady jsou obvykle komentovány, aby byl jasný postup jejich řešení.

 **Úkol**

Otázky a úkoly, náměty na vyzkoušení, které se doporučuje při procvičování učiva provádět, jsou uzavřeny v tomto prostředí. Pokud je v prostředí více úkolů, jsou číslovány.

Jak probíhají testy

Na zápočtových testech z předmětů týkajících se operačních systémů lze používat počítač, a to nápovědu a nástroje běžně dostupné ve standardní instalaci daného operačního systému, ale bez přístupu na Internet. Nejsou dovoleny dokumenty vlastní ani cizí výroby, které nejsou součástí standardní instalace, nelze používat internetový prohlížeč ani jiný způsob přístupu na externí zdroje informací.

Na stránkách předmětu je k dispozici orientační seznam otázek a úkolů, které se mohou objevit na testu, ovšem v testu se mohou objevit mírné odlišnosti (například v názvech zpracovávaných souborů či adresářů, jiné přepínače příkazů, apod.).

Tato skripta zároveň se skripty „Mezičást: bezpečnost a pořádek“ plně pokrývají odpovědi na otázky, které se mohou objevit v zápočtovém testu č. 2 (Linux).

Obsah

Předmluva **iii**

1 Od UNIXu k Linuxu **1**
1.1 UNIX . . . 1
1.2 Co je a co není UNIX . . . 2
1.2.1 Větve UNIXového světa . . . 2
1.2.2 UNIXové standardy . . . 3
1.2.3 Linux . . . 4
1.2.4 Apple MacOS . . . 5
1.2.5 Některé další UNIXové systémy . . . 6

2 Linux **8**
2.1 Co je to Linux . . . 8
2.2 Grafické uživatelské rozhraní v Linuxu . . . 10
2.2.1 Jak to funguje . . . 10
2.2.2 Přehled nejznámějších správců oken . . . 11
2.2.3 Přehled nejznámějších desktopových prostředí . . . 13
2.2.4 3D prostředí . . . 15
2.3 Instalace softwaru v Linuxu . . . 15
2.4 Verze distribucí . . . 18
2.5 Některé nejznámější distribuce . . . 18
2.6 Jak si vyzkoušet Linux . . . 24

3 Práce v grafickém uživatelském rozhraní **27**
3.1 Základní průzkum systému . . . 27
3.1.1 Okna, plochy, virtuální plochy . . . 27
3.1.2 Pracovní prostředí . . . 28
3.1.3 Systémové menu . . . 30
3.1.4 Ovládací centrum . . . 30
3.2 Přizpůsobení prostředí . . . 31
3.3 Správa zařízení . . . 33
3.3.1 Paměťová média . . . 33
3.3.2 Periferní zařízení . . . 34
3.3.3 Správa napájení . . . 37
3.4 Konfigurace sítě . . . 38
3.5 Správa softwaru . . . 40
3.6 Správa procesů, zdrojů a souborů . . . 41

3.7 Nastavení související s uživateli . . . 43
3.7.1 Osobní nastavení . . . 43
3.7.2 Správa uživatelů a skupin . . . 44
3.7.3 Přizpůsobení pro uživatele . . . 46
3.8 Nastavení data a času . . . 47

4 Vlastnosti UNIXových systémů 48
4.1 Adresáře a soubory . . . 48
4.1.1 Souborový systém . . . 48
4.1.2 Začínáme zkoumat souborovou strukturu . . . 49
4.1.3 Odkazy na soubor . . . 51
4.2 Spustitelné soubory, knihovny a související . . . 53
4.3 Bootování systému . . . 54
4.4 Zařízení . . . 54
4.4.1 Zajištění přístupu k zařízením . . . 54
4.4.2 Speciální soubory . . . 55
4.4.3 Soubory s evidencí paměťových médií . . . 59
4.5 Správa uživatelů . . . 61
4.5.1 Uživatelé a skupiny . . . 61
4.5.2 Adresáře a soubory související s uživateli . . . 62
4.6 Procesy . . . 64
4.7 Virtuální souborové systémy . . . 65
4.8 Přístupová oprávnění a vlastnosti souborů . . . 67
4.8.1 Úvod do přístupových oprávnění . . . 67
4.8.2 Speciální příznaky . . . 69
4.8.3 Atributy souborů . . . 72

5 Textový režim v Linuxu 78
5.1 Textové shelly a příkazy . . . 78
5.1.1 Kdy, kde a jak používat textový shell . . . 78
5.1.2 Konzoly . . . 79
5.1.3 Struktura příkazů . . . 80
5.2 Zástupné znaky . . . 81
5.3 Nápověda . . . 81
5.3.1 Jak volat o pomoc . . . 81
5.3.2 Manuálové stránky . . . 82
5.3.3 Znám příkaz, chci vědět, k čemu slouží . . . 84
5.3.4 Neznám příkaz . . . 86
5.4 Práce s adresáři a soubory . . . 87
5.4.1 Adresáře . . . 87
5.4.2 Soubory . . . 89
5.4.3 Pevné a symbolické odkazy . . . 91
5.5 Přesměrování a filtry . . . 92
5.5.1 Směrování a deskriptory . . . 92
5.5.2 Filtry . . . 93
5.6 Základy práce s proměnnými . . . 95
5.7 Aliasy . . . 98
5.8 Další příkazy . . . 99

Kapitola 1

Od UNIXu k Linuxu

Rychlý náhled: V této kapitole si ujasníme, co se rozumí pod pojmem UNIX, podíváme se na standardy, které takový systém musí splňovat. Také je zde uvedeno několik nejznámějších UNIXů a UNIX-like systémů.

Klíčová slova: UNIX, POSIX, SUS, BSD, System-V, Linux, MacOS, Solaris, OpenBSD, FreeBSD

Cíle studia: Po prostudování této kapitoly si ujasníte pojem UNIX a získáte základní přehled o systémech splňujících standardy definující UNIX.

1.1 UNIX

UNIX je víceuživatelský multitaskový (procesy se střídají na procesoru takovou rychlostí, až to vytváří dojem paralelismu) síťový operační systém běžící prakticky na čemkoliv od i386, včetně nových typů počítačů (64bitové varianty existovaly už dlouho před Windows 64bit). Je to víceprocesorový systém (dokáže rozdělovat úlohy mezi více procesorů).

UNIX vznikl roku 1969 v Bell Laboratories firmy AT&T (první provozuschopné verze byly asi o 2 roky později), pracovali na něm především programátoři Ken Thompson a Denis Ritchie. Jeho předchůdcem je MULTICS, operační systém sice velmi výkonný, stabilní a bezpečný, ale velmi složitý. Byl navržen především pro velké střediskové počítače, ale protože jel téměř na čemkoliv, začal se rozšiřovat i na menší počítače. Dnes je chápán především jako síťový operační systém použitelný také na malých zařízeních.

Záměrem tvůrců UNIXu bylo vytvořit operační systém, který by měl kladné vlastnosti MULTICSu a zároveň aby měl jednoduchou snadno rozšiřitelnou strukturu. Podle toho byl také původně pojmenován – UNICS (Uniplex Information and Computing System, později se CS změnilo na „moderní“ X).

UNIX byl původně psán v assembleru, později přepsán do jazyka BCPL (nazvaného údajně podle Bucephala, koně Alexandra Velikého), v roce 1972 byl zdrojový text přepsán do jazyka C (tento jazyk byl ostatně vytvořen právě k tomuto účelu). Důsledkem bylo nejen zpřehlednění, ale také zvýšení přenositelnosti. Pokud programátoři chtěli použít UNIX na jiné hardwarové platformě (jiný typ procesoru apod.), nebylo již třeba překopávat celý zdrojový kód, změnily se pouze hardwarově závislé části.

Celý operační systém byl původně koncipován tak, aby jednotlivé úkony bylo možné provádět pomocí řady malých nenáročných programů či příkazů, a proto v jednom kterémkoliv okamžiku vlastně není tolik prostředků zapotřebí, tedy nemá velké systémové požadavky. UNIX disponuje propracovaným systémem komunikace mezi procesy, tedy systémem pro snadné propojování procesů.

Poznámka:

Takže jednou z důležitých myšlenek UNIXu je vytvářet malé jednoduché programy a mít k dispozici velice dobrý systém komunikace mezi nimi. Složitější úlohy se prostě poskládají z více jednoduchých programů. Důsledkem je snadnější ladění kódu, u malých programů se „rychle přijde na chybu“.

Dalším typickým rysem je používání textových konfiguračních souborů (binární soubory vyžadují speciální programy pro jejich zpracování) a princip „všechno je soubor“, jehož důsledkem je možnost prakticky k čemukoliv přistupovat jako k souboru (včetně zařízení a některých objektů v systému); to znamená, že programátor nemusí programovat každý vstup/výstup zvlášť, pro přechod mezi různými vstupy/výstupy stačí drobná změna v kódu nebo není nutná žádná změna.

UNIX se vyznačuje tím, že na rozdíl od Windows „pokorně“ přijme to, co mu na hardwaru dáme, když je toho málo, zbytečně kvůli tomu „nepadá“ a nevypisuje jedovatá hlášení, a když je toho moc, dokáže prostředky co nejlépe využít. Udává se, že UNIX vyžaduje minimálně procesor kompatibilní s i386, pokud možno také matematický koprocesor, operační paměti minimálně 8 MB, a pokud nechceme grafické rozhraní, tak ještě méně (údajně stačí 2 MB RAM). Minimální hodnoty pro požadavky na systém samozřejmě platí v případě, že nevyužíváme grafické rozhraní. U grafického rozhraní jsou požadavky relativní, existuje totiž mnoho různých grafických rozhraní a každé z nich je jinak „hladové“.

1.2 Co je a co není UNIX

1.2.1 Větve UNIXového světa

Dále budeme hovořit o UNIXových systémech (lépe systémech UNIXového typu – UNIX-like systémech), tedy operačních systémech se strukturou a vlastnostmi odvozenými od původního UNIXu.

Původně existoval jediný UNIX (vyvíjel se a vylepšoval, ale jen v jedné vývojové větvi), ale kolem verze V6 se jeho vývoj rozdělil do dvou větví, které lze vysledovat i dnes, i když UNIXů je dnes celá řada:

BSD (Berkeley Software Distribution): větev vyvíjená původně na univerzitě v Berkeley, sem můžeme zařadit například NetBSD, OpenBSD, FreeBSD, SunOS (od společnosti Sun), Linux (i když ne zcela), Apple MacOS a jeho předchůdce NeXTStep, atd.

System V: větev vyvíjená původně firmou UNIX System Laboratories (USL), později AT&T, dnes Novell, zde patří například HP UX (od společnosti HP), Solaris (společnost Sun, dnes je to nejúspěšnější komerční UNIX), AIX (IBM), SCO UNIX (původně to byl XENIX od Microsoftu), atd.

Rozdíly mezi těmito větvemi se postupně stírají, dnešní UNIXy obvykle mají některé vlastnosti z každé větve (to platí také o Linuxu, který je založen především na myšlenkách BSD, ale najde se v něm také

něco z System V). Komerční UNIXy většinou patří do větve System V, ale přesto obsahují některé vlastnosti z větve BSD.

1.2.2 UNIXové standardy

Kromě toho, že UNIXové systémy se víceméně řadí do některé z těchto větví, také musí splňovat určité *standardy*, aby vůbec mohly být za *UNIX-like systémy* (tj. UNIXové) považovány. Standardů bylo v historii více, v současné době jsou zachovávány dva – POSIX a Single UNIX Specification.

Poznámka:

UNIX není jen specifikace (daná standardem SUS), je to i ochranná známka (tedy název UNIX je takto chráněn). Vlastníkem ochranné známky je konsorcium Open Group, a pouze ty systémy, které byly tímto konsorciem certifikovány jako UNIX, se mohou UNIXem nazývat. Operační systémy, které splňují SUS, ale neprošly certifikací, se označují jako UNIX-like (systémy UNIXového typu).

Certifikačním procesem prošel například Solaris, kdežto Linux je pouze UNIX-like systém.

Podíváme se podrobněji na standardy POSIX a SUS.

POSIX (Portable Operating System Interface) je standard publikovaný standardizační institucí IEEE, ve verzích POSIX-1 a POSIX-2. Standardizuje formu systémových volání (tj. způsobů, jak proces komunikuje s jádrem), funkcí v systémových knihovnách a chování procesů včetně jejich vzájemné komunikace. Účelem je co nejvíc zjednodušit přenos programů mezi různými operačními systémy (tzv. portování). POSIX je podporován nejen UNIXovými systémy, ale také některými verzemi Windows.

Standard POSIX tedy neurčuje, jak mají procesy vypadat a fungovat „uvnitř“, pouze určuje, jak má vypadat komunikace mezi procesem a jádrem operačního systému, případně mezi procesy navzájem. Stanovuje tedy komunikační rozhraní.

Single UNIX Specification (SUS) – splnění tohoto standardu je předpokladem pro to, aby mohl být systém nazýván UNIXovým. Vychází z POSIXu a zahrnuje určité specifikace související s programovým rozhraním programů a knihoven (předpokládá se jazyk C nebo podobný), seznam nástrojů (míněno většinou programů–příkazů pro uživatele a administrátory), požadavky na shell (program, kterým uživatel komunikuje se systémem), systémová volání, služby včetně vstupně/výstupních. Součástí nejsou požadavky na konkrétní kód, proto SUS mohou splňovat i takové systémy, které neobsahují ani řádek kódu z původního UNIXu.

SUS je v několika verzích. Nejnovější (verze 4) z roku 2008 se také označuje UNIX V7, z komerčních UNIXů byl pod ní certifikován pouze Solaris. U verze 3 (UNIX 03) prošlo certifikací více systémů – kromě Solarisu také například HP-UX, IBM AIX a všechny novější verze Apple MacOS.

Poznámka:

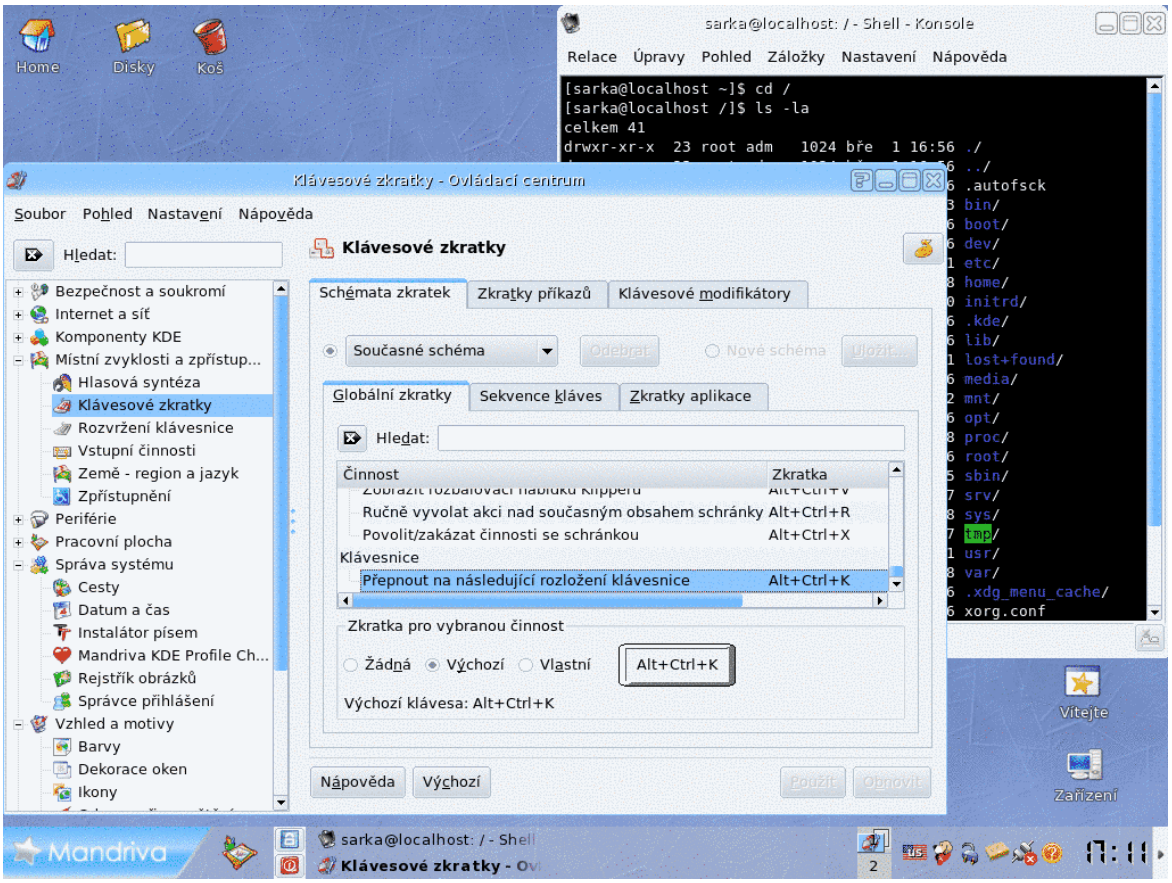
Jaký je rozdíl mezi POSIXem a SUS? Na rozdíl od POSIXu je SUS kompletnější, má mnohem více požadavků a zahrnuje mnohem víc než jen popis komunikace mezi procesy a jádrem. Přesněji – SUS přímo říká, co je a co není UNIX. Navíc SUS může vyústit až do fáze certifikace.

Další informace:

- Na http://www.faqs.org/docs/artu/ch17s02.html najdeme informace o různých standardech souvisejících s UNIXem, FAQs v doméně znamená „Frequently Asked Questions“.
- Text standardu SUS je na adrese http://www.opengroup.org/onlinepubs/009695399/ nebo také na http://www.UNIX.org/version3/, dále na http://www.UNIX.org/what_is_UNIX/single_UNIX_specification.html.
- Stránky o certifikaci UNIXových systémů od Open Group: http://www.opengroup.org/certification/idx/unix.html

1.2.3 Linux

Linux je operační systém UNIXového typu šířený pod licencí GNU GPL (resp. jeho jádro). Je to plně 32bitový nebo 64bitový (podle platformy) systém. První linuxové jádro vytvořil finský student *Linus Torvalds* (bylo mu tenkrát 21 let), dnes se na tomto operačním systému podílejí tisíce dalších programátorů po celém světě. Linux převzal z UNIXu vnitřní strukturu, stabilitu a bezpečnost, i když zdrojové kódy má znovu napsané.



Obrázek 1.1: Linux Mandriva Spring 2007

Již první verze zaujaly programátory projektu GNU a Linux se postupně stal hlavním jádrem operačního systému projektu GNU (původní jádro, GNU/Hurd, dlouho neexistovalo). Správný název

toho systému, který budeme používat, je ve skutečnosti GNU/Linux, většinou se však zkracuje na Linux.

Poznámka:

Abychom si to ujasnili: Linux je název jádra, kdežto celý systém (jádro plus to ostatní) je GNU/Linux.

V devadesátých letech byl chápán jako systém vhodný spíše pro odborníky na počítače, ale po rychlém rozvoji grafických rozhraní provozovatelných na tomto systému je již natolik uživatelsky přívětivý, že je instalován i na počítače úředníků státní správy po celém světě. Jeho podíl na trhu stoupá mimo jiné proto, že ho lze pořídit zdarma a platí se obvykle pouze za servis, podporu a manuály (a případně papírovou krabici).

Nejznámější linuxové distribuce jsou velmi rozsáhlé (obsahují totiž kromě jádra a grafického rozhraní mnoho aplikací a konfiguračních nástrojů), a uživatelsky přívětivé. Nejrozšířeněnší jsou zřejmě Mandriva, SUSE, Ubuntu, Debian a další (ale pořadí se každoročně trochu mění). Na obrázku 1.1 je obrazovka linuxového systému Mandriva na starším počítači (menší rozlišení obrazovky, vypnuté 3D efekty). Ovšem linuxové systémy mají velmi variabilní grafické prostředí, mohou vypadat značně odlišně.

Hlavní adresy: http://www.linux.org, http://www.linux.cz

1.2.4 Apple MacOS

Tento operační systém vytvořila společnost Apple pro své počítače s vlastní hardwarovou platformou, která nebyla kompatibilní s tehdy se pomalu rozvíjející platformou *Intel*. Jako jeden z prvních (dlouho před MS Windows) měl velmi propracované uživatelské rozhraní, operační systémy společnosti Microsoft se zjedně tímto systémem hodně inspirovaly. Jako jeden z prvních podporoval výkonné procesory.

Původně byl tento systém vyvíjen celý samostatně, dnes se již jedná o systém UNIXového typu podobně jako linuxové distribuce. Ke změně došlo díky jednomu ze zakladatelů společnosti Apple, Stevu Jobsovi. *Steve Jobs* odešel z této společnosti a založil svou vlastní – *NeXT*, která začala vyvíjet vlastní operační systém NeXTStep založený na struktuře UNIXu, především varianty FreeBSD. Pro tento systém byla vyvinuta i vlastní platforma založená na procesorech PowerPC (jiné procesory té doby měly příliš nízký výkon). NeXTStep se proslavil hlavně používáním objektových technologií a mimořádně propracovaným uživatelským a vývojovým rozhraním.

Společnost Apple nakonec koupila společnost NeXT a operační systém NeXTStep se stal základem pro novou verzi MacOS, pojmenovanou MacOS X.[1]

Jádro MacOS X je *Mach* (je přejaté z NeXTStep, má vnitřní strukturu vycházející ze standardů pro UNIXové systémy). Je považováno za jedno z nejefektivnějších a nejstabilnějších řešení, dokonce i na UNIXový systém. Toto jádro má i některé rysy real-time systémů. Nižší vrstvy tohoto systému jsou přístupné v rámci projektu *Darwin*.

Přechodem na verzi X MacOS změnil hardwarovou platformu na *PowerPC* (také je nekompatibilní s Intelem), na které běžely počítače s NeXTStepem (tyto procesory vyrábí společnost IBM). Na konci

[1] Písmeno X ve skutečnosti znamená spíše římské číslo 10, jde o verzi systému, i když zároveň může být chápáno jako vazba na UNIX.

roku 2005 byla překvapivě ohlášena migrace MacOS X na platformu Intel, což znamená, že tento operační systém je nyní možné používat na stejném počítači jako ostatní rozšířené operační systémy bez hardwarové emulace, ale legálně a v plné akceleraci pouze na strojích od společnosti Apple.

Později nastala další změna – v době, kdy Microsoft vyrukoval s Windows 10, se Apple rozhodl, že písmeno „X“ (tedy římskou 10) z názvu systému odstraní. Takže teď už zase jde o Apple MacOS.

Poznámka:

MacOS jako software je licencí úzce svázán s daným hardwarem. Pokud někdo nainstaluje MacOS na jiný počítač než ten od Applu, porušuje licenční podmínky.

Vzhledem k tomu, že se jedná o systém UNIXového typu, je možné zde používat i některé aplikace určené pro UNIX/Linux. S přihlédnutím ke značným změnám ve struktuře systému a dosavadní hardwarové nekompatibilitě s intelovskou architekturou se však jedná většinou o kompatibilitu na úrovni zdrojových kódů, tedy je nutné přenést zdrojové kódy a ty potom přeložit pro tuto architekturu. Programy, které přistupují přímo ke zdrojům výpočetního systému nebo k nižším vrstvám operačního systému, přenositelné nemusejí být (bylo by potřeba provést i změny ve zdrojových kódech).

Mac OS u nás původně nebyl moc rozšířen (používal se hlavně ve firmách pracujících s grafikou, jako jsou reklamní studia, je však také na některých školách), byl to systém typický spíše pro USA. Situace se však postupně mění a zvláště notebooky a malá přenosná zařízení jsou čím dál oblíbenější.

Adresy: http://www.apple.com/macosx/, http://www.apple.cz/, http://www.root.cz/serialy/mac-os-x-je-taky-UNIX/

1.2.5 Některé další UNIXové systémy

UNIXové systémy jsou dodnes hodně používány na serverech (například Solaris), ale existují také varianty pro desktop, dokonce i volně šiřitelné (například Linux a FreeBSD).

OpenBSD je volně šiřitelný operační systém UNIXového typu vycházející z větve BSD. Je distribuován pod licencí BSD. Hlavní důraz je zde kladen na bezpečnost, OpenBSD je považován za jeden z nejbezpečnějších UNIXových systémů. Celkově spolupráce na tomto projektu funguje hodně podobně jako na Linuxu, jen je trochu méně populární a rozšířený.

V OpenBSD najdeme i některé programy a technologie přejaté z Linuxu, licence BSD je kompatibilní s GPL, a navíc jako každý jiný UNIXový systém je i OpenBSD kompatibilní s Linuxem na úrovni zdrojových kódů. Můžeme zde používat kompilátor jazyka C gcc, Perl, Apache, Sendmail, atd. Grafické prostředí je realizováno na stejném principu jako na Linuxu. OpenBSD se vyznačuje dostupností pro velké množství hardwarových platforem.

Adresy: http://www.openbsd.org, http://www.openbsd.cz

FreeBSD je podobně jako OpenBSD šířen pod licencí BSD. Je to velmi dobře vybavený systém, jeho grafické prostředí je realizováno taktéž na stejném základě jako v Linuxu, instalace je velmi snadná. Je postaven na podobné filozofii jako Linux a vývojáři těchto systémů se navzájem inspirují. Je určen především pro hardwarovou platformu x86 a příbuzné.

Existuje také live varianta tohoto systému (tj. nemusí se instalovat, spouští se přímo z CD/DVD, stačí z něho nabootovat) – FreeSBIE. Je trochu osekaná (úspornější grafické rozhraní, méně aplikací),

přesto však slušně vybavená, lze ji také nainstalovat na disk.

Adresy: http://www.freebsd.org, http://www.freebsd.cz, http://www.freesbie.org

NetBSD je dalším BSD klonem. Projekty NetBSD a FreeBSD velmi úzce spolupracují, dokonce do té míry, že oba systémy jsou kompatibilní i binárně (nejen na úrovni zdrojových kódů, tj. programy přeložené pro jeden systém můžeme přímo spustit i na druhém).

Na rozdíl od FreeBSD je dostupný pro velké množství hardwarových platforem. Podpora výrobců hardwaru je bohužel nižší, seznam podporovaného hardwaru je proto kratší než třeba u Linuxu. Existuje také live varianta Jibbed Live CD. NetBSD se často používá v menších jednoúčelových zařízeních, dokonce včetně síťových (například v hardwarových firewallech) – velký důraz je kladen na bezpečnost (také vlivem spolupráce s OpenBSD).

Adresy: http://www.netbsd.org/, http://www.netbsd.org/gallery/products.html, http://www.jibbed.org/

OpenSolaris samotný Solaris je komerční UNIX společnosti Sun Microsystems (vlastněné společností Oracle Corporation), je považován za jeden z nejlepších komerčních UNIXů. Patří do větve System V. Společnost Sun se rozhodla zpřístupnit odvozený systém, *OpenSolaris*, pod vlastní open-source licencí CDDL.

Existuje několik odnoží (portů) projektu OpenSolaris, například *Illumos* (plně open-source systém, protože samotný OpenSolaris vlastně obsahuje některé části, které nejsou distribuovány se zdrojovým kódem), dále *OpenIndiana* (pro servery, dbá se na zajištění zabezpečení, a to volně šiřitelné), atd.

Adresy: http://www.sun.com, http://www.opensolaris.org, http://www.illumos.org/, http://openindiana.org/

QNX je realtimový UNIXový systém (tj. pracující v „reálném čase“). To neznamená, že by dokázal vždy reagovat okamžitě na jakýkoliv požadavek (to je technicky nemožné), ale pro každou časově rizikovou operaci existuje pevně daný časový limit, do kterého musí být tato operace vyřízena. Realtimové systémy se používají například při řízení rizikových provozů (elektrárny včetně atomových, chemické laboratoře), letadel apod. QNX byl původně komerční systém kanadské firmy QNX Software Systems, delší dobu však existovala volně šiřitelná varianta (zdarma pro nekomerční použití). Bohužel tento zdroj se uzavřel, jistou náhradou je projekt *OpenQNX*, dále na stránkách firmy QNX najdeme demonstrační verzi.

QNX je určen k použití v embedded systémech (tj. v zařízeních, která vlastně ani nejsou počítače, třeba autech), dále v síťových zařízeních (jako jsou například routery), set-top boxech, apod. Jeho jádro se jmenuje *Neutrino*, používá jednoduché grafické rozhraní *Photon*.

Adresy: http://www.qnx.com/, http://www.openqnx.com/, http://www.embedded-computing.com/pdfs/QNX.Jan05.pdf

Úkol

Vyberte si některý z UNIXových systémů (nemusí být uveden v této kapitole) a zjistěte si o něm podrobnější informace.

Kapitola 2

Linux

Rychlý náhled: V této kapitole se seznámíme s operačním systémem GNU/Linux a podíváme se na jeho typické vlastnosti. Celá kapitola směřuje k tomu, abychom věděli, jak si Linux nainstalovat a zprovoznit.

Klíčová slova: Linux, distribuce, verze, jádro, grafické rozhraní, správce oken, desktopové prostředí, 3D prostředí, widget knihovna, instalace, správce balíčků

Cíle studia: Po prostudování této kapitoly získáte přehled o fungování grafického uživatelského rozhraní v Linuxu, získáte přehled o nejběžnějších řešeních GUI, budete se orientovat v nejznámějších linuxových distribucích, budete umět instalovat software v Linuxu a budete umět si vyzkoušet jakoukoliv distribuci ve virtualizovaném prostředí.

2.1 Co je to Linux

Vlastním tvůrcem jádra Linuxu je *Linus Torvalds*, který, ještě jako student informatiky na univerzitě v Helsinkách, roku 1991 poslal na internet první verzi Linuxu (0.1). Tato verze zůstala navždy pouze v podobě zdrojového kódu (většinou v jazyce C), první přeložená verze byla 0.2. První více provozovaná verze byla 1.0, objevila se v roce 1994. Na vývoji Linuxu se od té doby podílejí tisíce dalších vývojářů.

Verze jádra Linuxu se skládá z několika čísel oddělených tečkou, první dvě čísla jsou samozřejmě nejdůležitější. V posledních letech se velmi zrychlil vývoj (zatímco verze 2.6.x vydržela přibližně 10 let, verze 3.x a novější mění druhé číslo přibližně jednou ročně). Navíc jsou některé verze označovány jako „longterm“, tedy určené pro dlouhodobou podporu. Momentálně je hlavní číslo verze 6.

Další informace:

Aktuální verzi jádra Linuxu zjistíme nejlépe na oficiálním webu projektu: https://www.kernel.org/. Hned na titulní straně jsou podporované verze s odkazy na další informace. Tabulka *Other resources* obsahuje několik užitečných odkazů, včetně odkazu na wiki stránky.

 To, co si pořizují lidé, není jen jádro, ale *linuxová distribuce*. Distribuce Linuxu je tedy souhrn těchto položek:

- *jádro* operačního systému (kernel),
- *instalační rutina*, která uživatele provede instalací systému přes obvykle dobře propracované grafické rozhraní bez nutnosti instalovat z příkazového řádku,
- *detektor hardwaru* je používán především při instalaci, kde zjišťuje připojená zařízení a celkovou konfiguraci počítače a opět může instalaci hodně usnadnit (pokud ovšem nemáme nějaký speciální hardware), detekce hardwaru se používá ve skutečnosti také při každém spouštění Linuxu,
- *konfigurační nástroje* a *systémové knihovny*,
- *grafické prostředí*,
- spousta *aplikací* pro práci s textem, grafikou, prezentacemi, zvukem, videem, *dokumentace*, atd.

Existuje mnoho různých distribucí, mezi nimi bývají velké rozdíly především ve vybavenosti různými nástroji (přesněji rozdíly jsou ve všech zde uvedených bodech kromě jádra).

 Rozlišujeme distribuce *komerční* a *komunitní* (obvykle volně ke stažení). Komunitní distribuce je vyvíjena komunitou, což je jakési volné sdružení lidí, kteří se rozhodli vyvíjet danou distribuci, naopak za komerční distribucí vždy stojí některá firma.

Komerční distribuce se prodávají za peníze, ale finanční náklady rozhodně nejsou velké, a samozřejmě produkt kupujeme vždy jen v jednom kusu (pořád platí, že z téhož média lze legálně instalovat na jakýkoliv počet strojů). Ve skutečnosti se platí spíše za „práci“, kterou provádí firma samotná, tedy úpravy (včetně zakázkových na míru), zkompletování (včetně přeložení), manuály, školení, vzdálená pomoc, technická podpora apod. V komerčních službách mohou být zahrnuty také aktualizace (ale často bývají zdarma), u komunitních distribucí jsou aktualizace vždy zdarma.

Mezi komunitními a komerčními distribucemi často bývá velmi blízký vztah – firma kromě své komerční distribuce sponzoruje komunitní distribuci a oběma směry putují změny ve zdrojových kódech, tedy jde o spolupráci. Pro firmu je výhodou především to, že příslušná komunitní distribuce je plně kompatibilní a také po grafické stránce velmi podobná, tedy pokud zákazníci chtějí přejít od komunitní distribuce ke komerční se všemi jejími výhodami, je zřejmé, kterou z komerčních distribucí zvolí. To je případ dvojice Fedora–RedHat, CentOS–RedHat Enterprise Server, nebo OpenSUSE–SUSE.

 **Další informace:**

Přehled distribucí najdeme především na webu https://distrowatch.com/. Kromě novinek je v jednom z bočních panelů seznam 100 nejúspěšnějších distribucí – můžeme si vybrat, za jakou dobu to bude posuzováno, výchozí je půl roku. Po výběru distribuce se dostaneme k základním informacím o dané distribuci, včetně recenzí v různých jazycích.

Velmi komplexním zdrojem je stránka https://en.wikipedia.org/wiki/List_of_Linux_distributions, kde najdeme dokonce „vývojové stromy“ ukazující rodinné vztahy mezi distribucemi.

 Mnohé distribuce můžeme dostat také ve variantě *live* (živé). Je to odlehčená verze, která se vejde na jedno CD nebo DVD, případně USB flash disk, a je plně bootovatelná bez nutnosti instalace. Stačí vsunout CD/DVD/USB flash disk, restartovat počítač (v BIOSu musí být povoleno bootování z daného typu média a mechanika nastavena jako *first boot device*), provede se rychlá detekce hardwaru, je načteno jádro, spuštěno grafické prostředí, a můžeme pracovat.

Výhodou je také „nezničitelnost“ live systému, v případě vážné chyby v konfiguraci prostě znovu nabootujeme.

Obvykle v takovém případě nemáme přístup k pevnému disku, ale některé live distribuce jsou výjimkou a disk zpřístupňují. V některých případech máme možnost uložit na nějaké paměťové médium (speciální soubor na pevném disku, USB flash disk nebo paměťový prostor na internetu) některé informace (např. zjištěnou konfiguraci hardwaru, provedené změny v rozhraní nebo vlastní dokumenty).

Live distribuce slouží k různým účelům, například

- k propagaci a demonstraci distribuce (uživatel se seznámí s prostředím a možnostmi distribuce),
- instalace systému na disk se často provádí tak, že uživatel nabootuje do live systému a z něj spustí instalaci poklepáním myši, během instalace můžeme libovolně pracovat v živém systému (třeba používat webový prohlížeč),
- k vyzkoušení funkčnosti (živá distribuce má vpodstatě stejný proces rozpoznávání hardwaru),
- pro opravu nainstalovaného operačního systému, záchranu dat nebo provedení zálohy nainstalovaného systému (třeba vytvoření bitové kopie),
- k samostatné specifické práci (potřebujeme některý z programů na live CD), nebo chceme na cizím počítači pracovat ve vlastním prostředí s vlastními daty (použijeme zároveň s flash diskem s uloženou konfigurací a daty), to je možné také ve virtuálním počítači.

Distribuce se dají sehnat buď přímo stáhnout na internetu (to platí hlavně pro komunitní distribuce, ale i pro některé komerční, případně jejich speciální edice určené k vyzkoušení), nebo na Internetu koupit (obvykle přes web dané distribuce). Obrazy instalačních médií bývají někdy přiloženy k počítačovým časopisům a sem tam se v knihkupectvích objeví i příručka pro danou distribuci s přiloženým instalačním médiem.

Z internetu se distribuce stahují přes FTP nebo přes výměnné sítě (BitTorrent), případně přes NFS, pokud jsou instalační soubory na lokální síti.

Některé distribuce jsou během instalace *překládány*. Pak opět potřebujeme startovací CD se základním systémem a překladačem, vše ostatní již může být stahováno přímo z internetu (třeba přes FTP). Ovšem překlad celé distribuce je časově velmi náročný, je třeba počítat i s několika dny.

2.2 Grafické uživatelské rozhraní v Linuxu

2.2.1 Jak to funguje

Počítač s Linuxem by mohl běžet i bez grafického rozhraní, protože v textovém režimu je přístup prakticky k čemukoliv a například na serverech je to lepší (bezpečnější a procesor není zbytečně zatěžován grafikou). Nicméně – Linux se instaluje i na běžné počítače, tedy máme také grafické rozhraní.

Základem GUI v Linuxu je *X Window System* (žádné Windows!!!), zkráceně *X Window* nebo prostě *X*. Tvůrcem je MIT (Massachusetts Institute of Technology), ale na projektu už téměř od začátku spolupracují zejména univerzity a velké firmy (IBM, Sun, HP a další). Hlavní verze je už léta 11, a zřejmě se ještě dlouho nezmění, a odtud také označení X11.

X Window je koncept (definuje standardy), postupně se objevilo několik jeho konkrétních implementací, z nichž je momentálně nejpoužívanější open-source implementace *X.Org*, dříve to byla XFree86.

Tato implementace se používá v mnoha UNIXových systémech, včetně naprosté většiny linuxových distribucí. V systému Apple MacOS se používá implementace *XQuartz*. Pro Android existuje *Android X Server*.

Princip grafického uživatelského rozhraní podle X Window je následující:

Vrstva X Window zajišťuje *funkcionalitu celého systému* – vede strukturu oken, v oknech strukturu prvků grafického rozhraní, umožňuje na prvky rozhraní napojit určité akce (například určit, která funkce se má spustit, když uživatel klepne na konkrétní tlačítko), zprostředkovává unifikovaný přístup ke grafickému hardwaru (takže proces se nemusí starat o to, jak něco vykreslit na obrazovku, stačí, když má k dispozici „plátno“ svého okna nezávislé na konkrétním zařízení). Z celého komplexu pouze X Window přistupuje k hardwaru (a to ještě ne přímo – pouze přes jádro operačního systému).

Widget knihovna (Widget Toolkit) je *zásobárna widgetů* (widget je prvek grafického uživatelského rozhraní, například tlačítko, titulek okna, menu, zaškrtávací pole, atd.), která určuje, jak konkrétně mají jednotlivé prvky vypadat (to X Window nedělá).

Správce oken (Window Manager) je program, který zajišťuje funkčnost GUI na vyšší úrovni. Využívá služeb X Window a zároveň používá konkrétní Widget knihovnu, takže propojuje funkcionalitu s konkrétním vzhledem. Komunikuje jak se systémem, tak i s procesy a uživatelem.

Desktopové prostředí (Desktop Environment) je komplet X Window, konkrétního správce oken, widget knihovny, konfiguračních nástrojů, aplikací, souborů nápovědy, atd.

aplikace	desktopové
správce oken + widget knihovna	prostředí
X Window System	je celý balík

Celá struktura je modulární, aby byla dostatečně pružná. Můžeme změnit widget knihovnu, můžeme použít jiného správce oken.

Další informace:

- Na webu https://www.gilesorr.com/wm/table.html je dlouhý seznam různých správců oken se stručnou informací a odkazy na weby těchto projektů.
- Na https://sawfish.fandom.com/wiki/Comparison_of_extensible_window_managers je správců oken sice jen několik, ale jedná se o „extensible správce“, jejichž interaktivní chování je možné maximálně ovládat pomocí různých skriptů
- Web https://www.slant.co/topics/343/ best-linux-desktop-environments zobrazuje tabulku desktopových prostředí (je zobrazeno jen pár položek, klepnutím na „See full list“ se tabulka rozbalí). Některé projekty jsou spíše na hranici mezi desktopovým prostředím a jednoduchým správcem oken.
- Srovnání správců oken a desktopových prostředí najdeme na stránkách
  - https://en.wikipedia.org/wiki/Comparison_of_X_window_managers,
  - https://en.wikipedia.org/wiki/Comparison_of_X_Window_System_desktop_environments.

2.2.2 Přehled nejznámějších správců oken

Nejdřív se podíváme na jednoduché správce oken, kteří ještě nestihli „přerůst“ do kategorie desktopového prostředí (tj. ještě na ně není nabaleno dostatek aplikací). Jejich typickou vlastností jsou nízké

nároky na výkon hardwaru, přestože jsou pro běžné použití naprosto dostačující. Z nejznámějších:

- *i3* – jde o velmi flexibilního správce oken pro Linux a BSD systémy určeného spíše pro pokročilejší uživatele a programátory. Vyznačuje se tím, že v prostředí lze ručně nastavit prakticky cokoliv.
  - *Adresa*: https://i3wm.org/
- *Sway* – tento správce oken je odvozen od i3, přidává mu některé další funkce. Spouštěná okna se defaultně sama zarovnávají do mřížky takovým způsobem, aby se maximalizovalo využití plochy monitoru.
  - *Adresa*: https://swaywm.org/
- *awesome* – správce oken určený zejména pro znalejší uživatele a programátory, je rozšiřitelný o moduly programované v jazyce Lua a také existuje mnoho předpřipravených modulů naprogramovaných v Lua. Vyznačuje se velmi dobrým okomentováním zdrojového kódu. V prostředí můžeme používat myš, nebo se můžeme kompletně obejít bez myši (dá se plně ovládat klávesnicí).
  - *Adresa*: https://awesomewm.org/index.html
- *XMonad* – svižný správce oken distribuovaný pod licencí BSD, u kterého je velmi dobře vyřešeno automatické zarovnávání oken na ploše. Dá se rozsáhle konfigurovat, pro což se používá programovací jazyk Haskell. V tomto prostředí lze bez problémů používat i komponenty prostředí Gnome a KDE (viz následující sekci).
  - *Adresa*: https://xmonad.org/
- *Qtile* – tento správce oken se od odstatních odlišuje například tím, že je psán v Pythonu. Podobně jako jiní správci oken, také Qtile je velmi úsporný co se týče využití místa na disku, operační paměti i procesoru.
  - *Adresa*: https://qtile.org/
- *AfterStep* – správce oken založený na rozhraní *NeXTStep*. Má velmi malé hardwarové požadavky a nabízí rozsáhlé možnosti konfigurace.
  - *Adresa*: http://www.afterstep.org/
- *FluxBox* – malý a rychlý správce distribuovaný pod licencí MIT. Plocha má kontextové menu, ve kterém najdeme programy ke spuštění a také konfigurační nástroje.
  - *Adresa*: http://www.fluxbox.org/
- *OpenBox* – opět se jedná o poměrně úsporného a rychlého správce, jehož hlavní nabídka je přístupná přes pravé tlačítko myši na ploše. Je zajímavý především zvláštní implementací menu, která umožňuje položky menu dynamicky měnit a přidávat například pomocí skriptů.
  - *Adresa*: http://openbox.org/wiki/Main_Page
- *fvwm* – zkratka znamená F Virtual Window Manager, kdysi to byl nejpopulárnější správce pod UNIXem. Vyznačuje se mnohem větší flexibilností než než jeho předchůdce twm, prostředí může vypadat například jako prostředí Windows 95.
  - *Adresa*: https://www.fvwm.org/

SawFish – správce oken konfigurovatelný pomocí jazyka Lisp.

Adresa: https://sawfish.fandom.com/wiki/Main_Page

Úkol

Vyberte si některého správce oken, najděte si k němu podrobnější informace (použijte uvedené adresy) a připravte si jeho krátkou charakteristiku. Může se vyskytnout na zápočtovém testu.

2.2.3 Přehled nejznámějších desktopových prostředí

Jak bylo výše uvedeno, desktopové prostředí je souhrn X, některého správce oken, widget knihovny, aplikací všeho druhu (včetně kancelářských, webového prohlížeče apod.), konfiguračních nástrojů a dalších součástí. Typicky se dají rozsáhle konfigurovat a doplňovat pomocí aplikací nebo appletů (applet je jednoduchá aplikace, jejíž aktivní ikona se často „připíná“ na hlavní panel). Podíváme se na několik nejznámějších.

CDE – Common Desktop Environment, prostředí postavené nad widget knihovnou *Motif*, komerční. V Linuxu ho najdeme prakticky jen v komerčních verzích, není moc populární.

Adresa: http://www.opengroup.org/tech/desktop/cde/

KDE – K Desktop Environment. Vybavenost nástroji i aplikacemi je velmi dobrá, najdeme tu prakticky vše, co potřebujeme. KDE má objektovou strukturu a je velmi objemné (běží mnoho procesů zároveň), proto není příliš vhodné na méně výkonné stroje. Typickou vlastností je maximální konfigurovatelnost.

Je postaveno na widget knihovně *Qt* (zkratka z Quasar Toolkit, čte se však [*kjůt*]), správce oken je *KWin*.

Adresa: https://kde.org

GNOME – vedle KDE je to jedno z nejpopulárnějších prostředí. Také obsahuje velké množství aplikací a nástrojů. Je trochu méně, i když dostatečně, vybavené než KDE, ale díky své struktuře je rychlejší, přehlednější, jeho běh je optimálnější, proto je vhodné i na pomalejší počítače.

GNOME používá widget knihovnu *GTK+*, správcem oken původně bylo *Metacity*, ale v novějších verzích najdeme pravděpodobněji jeho nástupce *Mutter*.

Některé distribuce používají prostředí GNOME s některou nástavbou. Nejznámější nástavby jsou *Unity* a *GNOME Shell*, v obou případech jde hlavně o to zcela pozměnit vizuální podobu a uspořádání ovládacích prvků rozhraní. Uživatelské prostředí při použití těchto dvou konkrétních nástaveb připomíná spíše tablet než notebook či desktop. GNOME ve verzi 3 má GNOME Shell jako svou součást.

Adresa: https://www.gnome.org/

Cinnamon – toto desktopové prostředí původně vzniklo jako odnož (fork) prostředí GNOME (přesněji GNOME Shellu), ale od verze 2 má již samostatný vývoj a už to není pouhá nástavba GNOME. Bylo vyvinuto pro distribuci Linux Mint a v té se také používá jako jedna z možností, ale je dostupná i pro další distribuce.

Používá widget knihovnu *GTK+* a správce oken *Mutter*, stejně jako GNOME.

☎ *Adresy*: https://github.com/linuxmint/cinnamon, https://www.zdnet.com/article/how-to-customise-your-linux-desktop-cinnamon/

- *Mate* – je to fork GNOME 2, tedy nemá nic společného s GNOME Shellem. Je pojmenováno podle rostliny „yerba maté“ z jižní Ameriky, ze které se vyrábí obdoba čaje. Účelem bylo vytvořit svižné a přitom dobře konfigurovatelné uživatelské prostředí.

  Používá widget knihovnu GTK+, ale má vlastního správce oken – *Marco*.

  ☎ *Adresa*: http://mate-desktop.com/

- *Pantheon* – poněkud vybočující prostředí, protože oproti ostatním nemá až tak moc možností úpravy k obrazu svému. Za tímto prostředím stojí vývojový tým linuxové distribuce Elementary OS, a také především (ale nejen) v této distribuci se s ním lze setkat. Vzhledově připomíná prostředí MacOS.

  ☎ *Adresy*: https://elementary.io/, https://wiki.archlinux.org/title/Pantheon

- *LXDE* (Lightweight X11 Desktop Environment) – odlehčené svižné prostředí, které je využíváno především v distribucích pro výpočetně slabší zařízení nebo v distribucích se speciálním účelem (například Raspbian pro Raspberry Pi).

  Používá widget knihovnu *GTK+* a výchozím správcem oken je *OpenBox*, ale správce oken lze změnit. Existuje také port *LXQt* postavený na knihovně *Qt*.

  ☎ *Adresa*: http://lxde.org/

- *Enlightenment* – vizuálně i možnostmi nastavení velmi zdařilý projekt někde mezi správcem oken a desktopovým prostředím. Má trochu zvláštní ovládání (ve standardním nastavení) a velmi rozsáhlé možnosti konfigurace, může vypadat prakticky jako kterýkoliv jiný správce oken. Prostředí je doplněno aplikacemi z GNOME. Cílem je, aby práce v grafickém prostředí byla co nejvíce intuitivní, aby zahrnovala prakticky vše, co se dá dělat včetně snadnější manipulace se soubory.

  Widget knihovny jsou dvě – ETK a EWL, první z nich se zdá používanější. Prostředí je samo sobě správcem. Na rozdíl od většiny ostatních je šířen pod licencí BSD.

  ☎ *Adresa*: http://www.enlightenment.org/

- *Xfce* – je to úsporné (malé nároky na hardware) modulární desktopové prostředí, které nabízí rozsáhlé možnosti konfigurace. Máme k dispozici více pracovních ploch, panel ke spouštění aplikací, atd.

  Konfigurace se provádí pro daný objekt přes kontextové menu tohoto objektu (*Vlastnosti*), nebo ve *Správci nastavení* (toho spustíme z hlavního panelu nebo z kontextového menu plochy), to, co nelze nastavit v tomto nástroji, se nastavuje v konfiguračních souborech (některé z nich jsou dokonce ve formátu XML, například `~/.xfce4/menu.xml` pro konfiguraci menu). Obsahuje vlastního správce souborů – `xffm`.

  Správcem oken v Xfce je Xfwm, používá widget knihovnu GTK+ z projektu GNOME.

  ☎ *Adresy*: https://www.xfce.org/, live CD pro prezentaci XFce http://www.xfld.org, další nástroje a programy pro XFce jsou na https://www.berlios.de/software/xfce-goodies/

Úkol

Vyberte si některé desktopové prostředí, najděte si k němu podrobnější informace (použijte uvedené adresy) a připravte si jeho krátkou charakteristiku. Může se vyskytnout na zápočtovém testu.

Poznámka:

Každé desktopové prostředí má svého výchozího správce oken, ale tento správce oken může být vyměněn za jiného. Toho využíváme například při implementaci 3D rozhraní – v „obyčejném“ desktopovém prostředí, které máme nainstalováno, je jako správce oken použit jiný, který zvládá 3D.

2.2.4 3D prostředí

V Linuxu (a ostatně obecně v UNIXových sytémech) je 3D prostředí využíváno už mnohem delší dobu než ve Windows. Existuje více těchto projektů, my se podíváme na nejznámější. Většinou využívají OpenGL a hardwarovou akceleraci grafiky.

Compiz Fusion je 3D správce oken využívající technologii OpenGL, který může být integrován do různých desktopových prostředí včetně KDE a GNOME. Compiz Fusion vznikl sloučením dvou starších projektů – Compiz a Beryl, přičemž u zrodu Compizu byla například společnost Novell.

Při svém vzniku to byl první *kompozitní* správce oken (kompozitní správce dokáže plně využít možností akcelerace zejména 3D grafiky nabízených technologií OpenGL).

Adresy: https://www.youtube.com/watch?v=4QokOwvPxrE (Ubuntu 3D Desktop Cube – KDE & Compiz Fusion & Cairo Dock),
https://compiz-fusion.org/

Croquet je desktopové prostředí distribuované pod vlastní licencí, které však překračuje hranice této definice (má například vlastní správu procesů). Je postaveno na objektovém programovacím jazyce SmallTalk (vlastně jeho svobodné variantě *Squeak*), je například v projektu One Laptop Per Child.

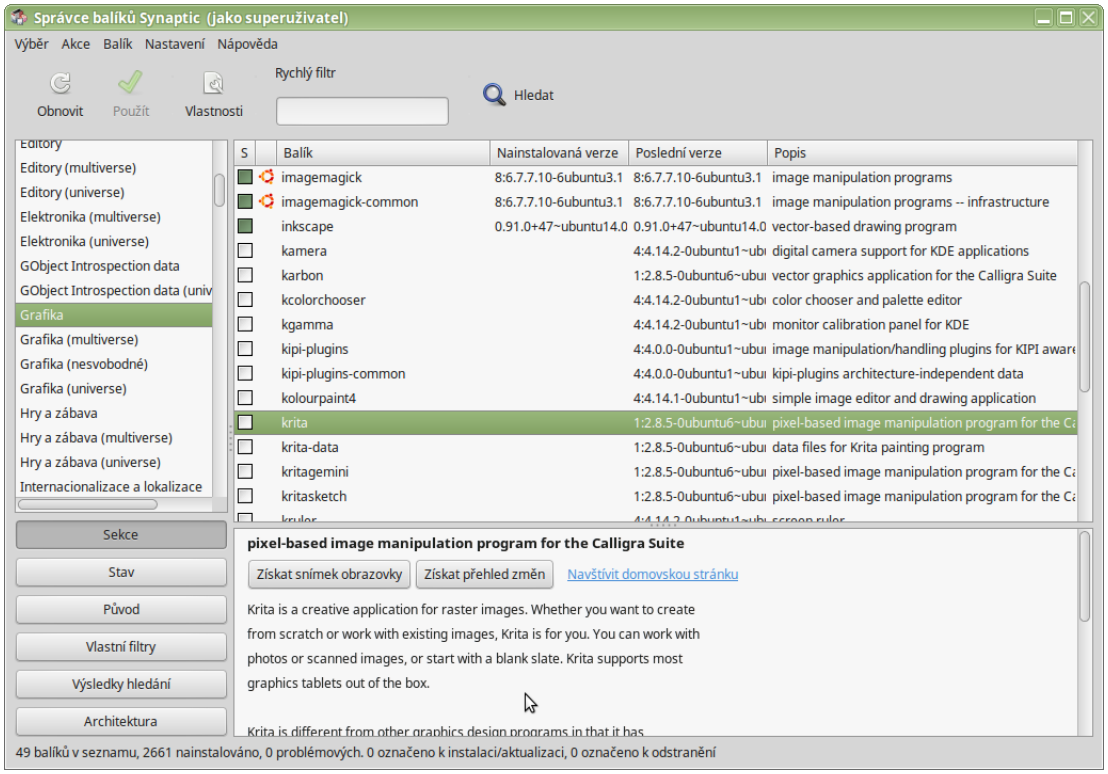
Adresy: http://www.opencroquet.org/, https://www.linuxexpres.cz/modules/marwel/index.php?article=1856, https://www.root.cz/serialy/squeak-navrat-do-budoucnosti/, https://squeak.org/

2.3 Instalace softwaru v Linuxu

Ve světě Linuxu se software distribuuje v *balíčcích*. Aplikaci tedy získáme v jednom nebo několika balíčcích (mohou být zvlášť „zabaleny“ různé přídavné moduly, zvlášť bývají jazykové balíčky, apod.). Balíček si můžeme představit jako obdobu ZIP archivu zahrnující spustitelný soubor, potřebné knihovny, konfigurační soubory, instalační skripty, popis postupu instalace.

Existují dva základní typy (formáty) balíčků – RPM a DEB. Balíčky jsou pak soubory s příponou .RPM nebo .DEB. Distribuce obvykle používají jeden z těchto formátů.

Můžeme se setkat také se *zdrojovými balíčky*, které místo přeloženého spustitelného souboru a knihoven obsahují zdrojový kód, který je během instalace balíčku přeložen „na míru“ danému systému. Další



Obrázek 2.1: Správce balíčků Synaptic v Linux Mint

možnost je čistý *zdrojový kód* (tj. není to balíček), kde na rozdíl od zdrojového balíčku nemáme předpřipravený automatický instalační proces.

✎ Každá větší distribuce má pro své uživatele *repozitář balíčků*. Je to databáze balíčků, ke které se obvykle dá velice jednoduše přistupovat přes příslušný program, který je také součástí distribuce (tento program je vlastně rozhraním k repozitáři). Tento program je obvykle napojen na hlavní repozitář dané distribuce, ale není problém ho napojit na další repozitáře – pak se nám v programu (aplikaci) zobrazují balíčky z více repozitářů zároveň.

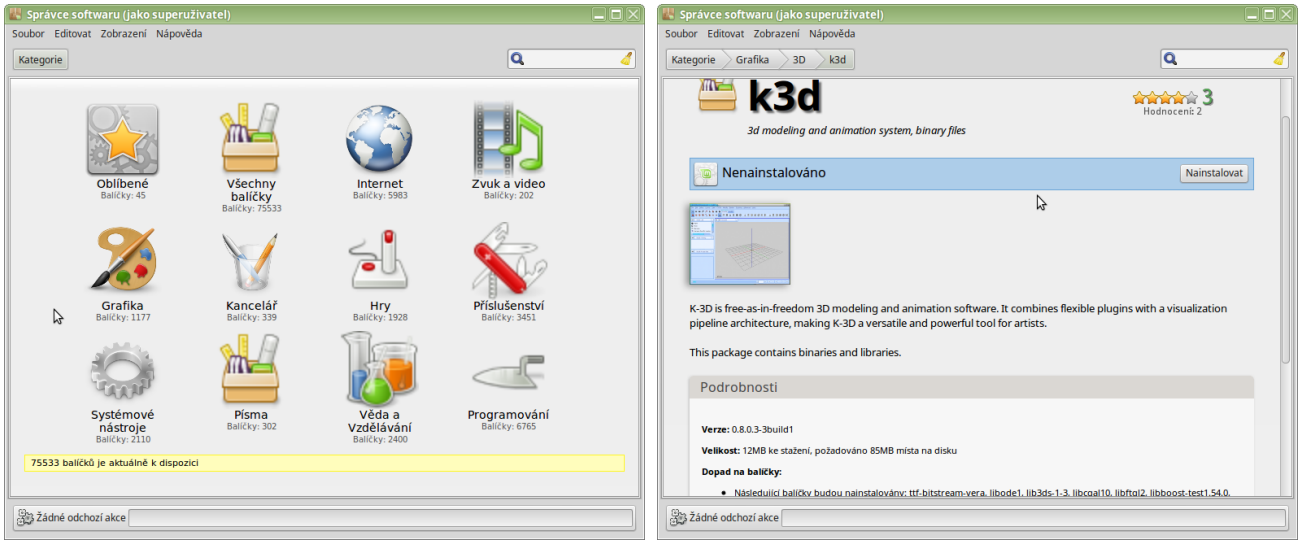
Repozitář však může být i něco jiného než databáze příslušející k určité distribuci. Jako repozitář (tedy zásobárna balíčků) může sloužit třeba instalační DVD nebo stránka soukromého vývojáře, který nabízí jeden svůj vlastní program.

Repozitáře (alespoň ty větší) jsou členěny do kategorií. Na obrázku 2.1 vidíme správce balíčků *Synaptic* v distribuci Linux Mint, kde vlevo je seznam kategorií, vpravo seznam balíčků z vybrané kategorie, dole pak podrobnější informace k vybranému balíčku.

✎ Ve větších distribucích máme kromě správců balíčků také *správce softwaru*. Sice pracují také s balíčky, ale již na vyšší úrovni, uživateli zobrazí informace o aplikaci a „neobtěžují“ ho s informacemi, ve kterých balíčcích se nachází nejnovější verze. Na obrázku 2.2 vlevo je *Správce softwaru* v Linux Mint.

Po spuštění najdeme seznam kategorií, po klepnutí na konkrétní kategorii se dostaneme k podkategoriím a jednotlivým aplikacím. U každé (pod)kategorie je zároveň informace o počtu balíčků, ale to je třeba brát s rezervou – pokud jsme napojeni na víc než jeden repozitář (a to pravděpodobně jsme), najdeme tam mnohé balíčky ve více vydáních (případně novější i starší verzi).

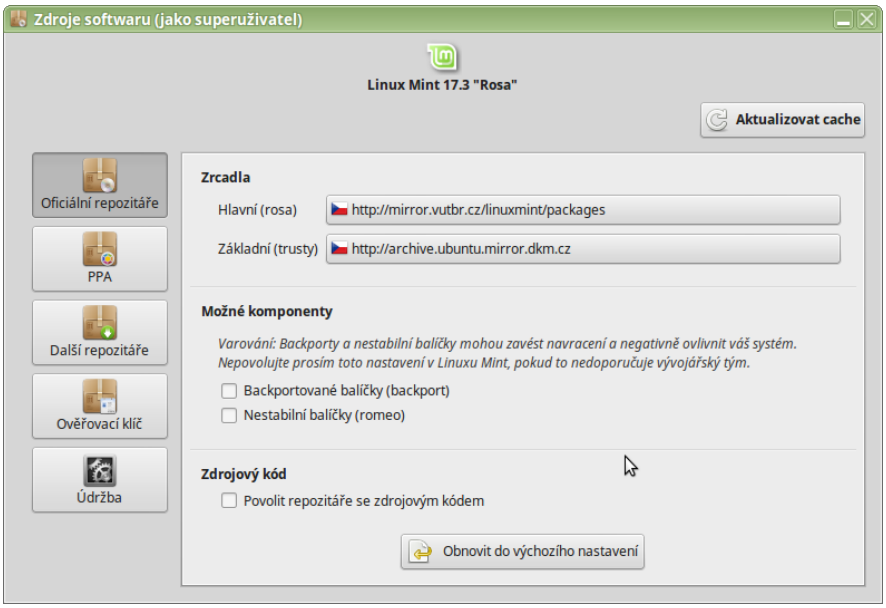
Po vybrání konkrétní aplikace dostaneme informace o této aplikaci. Například na obrázku 2.2 vpravo jsou informace o aplikaci K3d pro 3D modelování z kategorie Grafika, podkategorie 3D. Zjistíme, že



Obrázek 2.2: Správce softwaru v Linux Mint

ještě není nainstalovaná, která je to verze, kolik místa na disku po instalaci zabere, a případně zda závisí na něčem jiném (co tedy bude třeba zároveň doinstalovat, což se děje automaticky). Obvykle bývá i několik screenshotů, tady máme jen jeden.

Pokud známe název, můžeme použít vyhledávací řádek (dá se vyhledávat buď v celém stromě repozitářů nebo jen v určité kategorii) – vyhledávací pole je v okně vpravo nahoře.



Obrázek 2.3: Nastavení repozitářů

Pokud chceme v těchto nástrojích vidět i obsah jiných repozitářů, opět pro to máme příslušný nástroj. Na obrázku 2.3 je nástroj Zdroje softwaru v Linux Mint, v němž můžeme určit, které repozitáře budou „viditelné“.

Tentýž repozitář může být zpřístupněn paralelně přes několik adres (přesněji – je *zrcadlen* na více serverech). Například na obrázku 2.3 vidíme, že hlavní repozitář je hledán na adrese patřící VUT v Brně.

Tlačítkem *Další repozitáře* bychom se dostali k části nástroje, kde se dají vložit alternativní zdroje (typicky tam bývá hlavně instalační médium, ze kterého byla nainstalována celá distribuce). Také si všimněte tlačítka *Aktualizovat cache* (vpravo nahoře) – tímto způsobem aktualizujeme strukturu a seznam balíčků, synchronizujeme se seznamem v repozitáři.

Aktualizace jsou distribuovány naprosto stejným způsobem, přičemž se ve většině velkých distribucí automaticky hlídá repozitář, zda se v něm neobjeví aktualizace některého balíčku (tj. funguje přes všechny balíčky, nejen přes systém).

2.4 Verze distribucí

Zatím víme o verzích jádra Linuxu, ale jednotlivé distribuce mají své vlastní verzování.

Když se objeví nová verze (release) distribuce, vytvoří se pro ni samostatný repozitář (máme instalovánu určitou verzi ⇒ jsme napojeni na repozitář této verze). Obvyklá doba mezi uvedením dvou po sobě následujících verzí se nazývá *release cycle* (vývojový cyklus distribuce). Může to být jeden rok, ale u každé distribuce to je trochu jiné.

Jednotlivé distribuce se o své repozitáře a tedy i aktualizace starají vždy po určitou dobu – této době se říká *maintenance cycle* (life cycle, životní cyklus distribuce). Po uplynutí této doby již nejsou vydávány aktualizace. Tato doba je obvykle několikanásobně delší než vývojový cyklus, tedy v jednom okamžiku je pro distribuci v provozu několik aktivních udržovaných verzí a jejich repozitářů.

Nová verze, která ještě není plně vyzkoušená na různém hardwaru, je označována jako *vývojová*, po určité době (kdy už je vyzkoušená na dostatečném množství zařízení) se stává *stabilní*.

Poznámka:

Velké distribuce vydávají jednou za několik let verzi označenou *LTS (Long Term Support)* – verzi, u které se počítá s hodně dlouhou dobou provozu (aktualizace budou k dispozici delší dobu než u okolních verzí). Především tyto verze se používají na serverech, kde by přechod na novou verzi nemusel být úplně bez problémů.

2.5 Některé nejznámější distribuce

Existuje velké množství linuxových distribucí, ale naprostá většina těch nejběžnějších vychází z několika základních. Je zcela běžné, že některá firma či komunita si vezme za základ některou existující distribuci, přeprogramuje ji, přizpůsobí konkrétnímu účelu, pozmění, protřídí nebo naopak doplní, a vytvoří tak novou vlastní distribuci. Pak hovoříme o distribuci *odvozené (vycházející)* z určité základní distribuce.

RedHat („červený klobouk“) – jedná se o jednu z nejznámějších komerčních distribucí, za ní stojí společnost RedHat (jednu ze svých největších poboček má v Brně). Známá je především komerční distribuce určená především pro servery *RedHat Enterprise Linux* – RHEL.

Používá vlastní formát balíčků aplikací typu RPM (v této distribuci byly jako v první linuxové distribuci použity distribuční balíčky se softwarem, předtím se instalovalo ze zdrojových kódů).

Fedora Linux je komunitní distribuce vycházející z RedHatu určená především pro desktopy, kterou společnost RedHat doporučuje jako variantu k vlastní komerční distribuci (mezi komunitou a společností RedHat vládne čilá spolupráce, která je ku prospěchu oběma stranám). Fedora je známá tím, že velmi rychle implementuje různé inovace, například co se týče podpory nových verzí protokolů či nových typů rozhraní. Hlavním desktopovým prostředím je Gnome, ale existuje možnost zvolit jiné včetně odlehčených správců oken (jako je Awesome nebo LXDE), také Xfce, Cinamon a další.

Lze sehnat také odvozené produkty, kterým se říká Fedora spins, ty se liší zejména obsaženým softwarem, tedy svým zaměřením (různá desktopová prostředí, orientace na bezpečnost, hry, robotiku apod.).

Existují i další distribuce, které velmi úzce koexistují s RedHatem, nejen proto, že z něho vycházejí. Většinou jsou binárně kompatibilní. Kromě Fedory jde především o *CentOS* (Community Enterprise

OS). Až do roku 2020 (do verze CentOS 8) vždy určitá verze CentOS vznikla tak, že se vzaly zdrojové kódy RedHat Linuxu, upravily se a přeložily. To je v případě Linuxu zcela legální a RedHatu to nevadilo (naopak: CentOS pomáhal odstraňovat chyby, vylepšovat vlastnosti apod.). Roku 2020 však tým CentOS tento koncept opustil. Původní distribuce CentOS tedy přestala existovat a byla nahrazena distribucí *CentOS Stream*, která je situována někde mezi vlastním RedHatem a Fedorou.

Společnosti RedHat rozhodně nevadí existence Fedory, CentOS Stream ani dalších odvozených komunitních distribucí, je to reklama na její vlastní služby a jejich uživatelé často při přechodu na komerční řešení přecházejí právě na RHEL od RedHatu. Důvodem přechodu je většinou zájem o související komerční služby, zejména při rozšiřování společnosti.

Zatímco Fedora je určena spíše pro desktopy a notebooky, CentOS Stream cílí spíše na servery. Není k dispozici komerční podpora, tedy je volen spíše tam, kde už někdo příslušné dovednosti má a není problém s tím, že nasazení systému bude trvat o něco déle – často v akademické sféře.

Podobným způsobem vzniklo několik dalších distribucí. Z dalších můžeme jmenovat například *Scientific Linux*, který je vytvářen a distribuován laboratoří Fermi, střediskem CERN a dalšími vědeckými organizacemi a univerzitami na celém světě.

Adresy: https://www.redhat.com/, https://fedoraproject.org/, https://www.centos.org/centos-stream/, https://www.scientificlinux.org/, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/, https://fedoranews.org/

Debian GNU Linux je jediná velká distribuce, která je poskytována pouze zdarma (plně komunitní). Nepoužívá RPM balíčky, ale vlastní DEB balíčky. Tento balíčkovací systém je považován za pružnější a bezpečnější než RPM.

Debian je považován za nejstabilnější a nejbezpečnější linuxový systém, a také nejrozsáhlejší. Jeho distributorem není žádná určitá firma, ale skupina programátorů, proto také uživatelé nemohou počítat s nějakou propracovanou on-line podporou (ale v diskusních skupinách každý rád poradí a na webu lze najít spoustu dokumentů s řešením nejrůznějších problémů).

Pro tuto distribuci je typické, že se vyhýbá jakémukoliv softwaru, který není svobodný. Pokud tedy chceme například používat proprietární ovladač grafické karty, můžeme, ale je třeba si ho stáhnout odjinud než z hlavního repozitáře.

Má jeden z nejdelších intervalů pro vydávání stabilních distribucí (tj. dlouhý release cycle), na stabilitu systému a programů je velmi dbáno, ale jsou dostupné také vývojové verze. Rovněž další zajímavou vlastnosti (a také jedním z důvodů dlouhých intervalů mezi verzemi) je hodně variant pro různé hardwarové platformy. Stabilita a dostupnost pro různé platformy jsou dva z mnoha důvodů, proč je Debian nasazován jak na desktopy, tak často i na servery.

Výchozím jádrem pro Debian je Linux, ale existují také porty s jinými jádry – z větve BSD Linuxů (především NetBSD), které jsou taktéž podporovány komunitou Debianu.

V současnosti je Debian hodně často nasazován na serverech, ale vyskytuje se i na desktopech. Jeho podíl na desktopech snížil především nástup distribucí z Debianu odvozených (především Ubuntu).

Adresy: https://www.debian.org/, http://www.debian.cz/, https://www.debian.org/ports/netbsd/

Mageia – francouzská distribuce původně vycházející z distribuce Mandriva, která má původ v RedHatu. Zatímco původní Mandriva byla komerční, Mageia je komunitní distribuce. Používá formát

balíčků RPM, jinými slovy, co jde nainstalovat na RedHatu, jde bez velkých úprav nainstalovat i zde. Vyznačuje se velkou uživatelskou přívětivostí včetně instalace.

Jednou z dalších odnoží distribuce Mandriva je *PCLinuxOS*. Jeho tvůrce, Bill Reynolds zvaný Texstar, byl původně zaměstnancem Mandrakesoftu (to je původní název tvůrce Mandrivy), ale po neshodách s týmem se rozhodl dále pracovat na své vlastní distribuci. Typickým znakem PCLinuxOS je, že nevytváří nové verze, ale neustále aktualizuje jedinou. Takže uživatel se zapnutými aktualizacemi má vlastně neustále nejnovější verzi. Uživatel má k dispozici také skript *mylivecd*, který dokáže z nainstalovaného systému vytvořit bootovací CD/DVD/USB flash, a to včetně všech nastavení, programů a dokumentů.

Adresy: https://www.mageia.org/, http://www.pclinuxos.cz/

SUSE – německá společnost SUSE je dnes ve vlastnictví společnosti Novell (zároveň s IBM). Uživatelsky příjemná distribuce včetně instalace a konfigurace (obojí má na starosti nástroj Yast2), taktéž vhodná i pro začátečníky. Používá balíčky RPM, vznikla tedy odvozením z RedHatu. V současné době se vyskytuje v serverové podobě jako SLES (SUSE Linux Enterprise Server).

Na poli komerčních Linuxů tedy kralují především dva: RHEL a SLES.

OpenSUSE je komunitní varianta SUSE. Dá se volně stáhnout a hlavní myšlenkou je co největší uživatelská přívětivost systému. Je k sehnání v komerční i komunitní verzi (OpenSUSE).

Adresy: https://www.suse.com/, https://www.opensuse.org/

Ubuntu je distribuce vycházející z Debianu. Používá balíčky aplikací deb, většina softwaru je kompatibilní (Debian/Ubuntu/různé další odvozené distribuce). Jde o komerční distribuci, za ní stojí společnost Cannonical.

Ubuntu používá motto „Linux pro lidi“. Oproti jiným distribucím je poněkud „osekaná“, především z důvodu častých stížností uživatelů přecházejících na Linux z Windows, že „se v tom množství aplikací a nástrojů nevyznají“. To může být výhodou pro uživatele, který do systému prakticky nezasahuje (ostatně není problém si další aplikace doinstalovat zcela podle svého vkusu). Mechanismus ověřování přístupových oprávnění je ve výchozím nastavení určen spíše pro desktop bez nutnosti vzájemného zabezpečení různých uživatelů.

Výchozím grafickým prostředím v Ubuntu je už od začátku GNOME, s rozhraním Gnome Shell/Unity. Toto rozhraní bylo původně určeno pro malé displeje (například netbooků nebo ještě menších zařízení), a bohužel se od toho odvíjejí poněkud chudé možnosti konfigurace prostředí a horší dostupnost méně využívaných aplikací. Naštěstí jsou dostupné varianty s jiným grafickým prostředím:

- Ubuntu Mate Desktop – desktopové prostředí Mate,
- Kubuntu – desktopové prostředí KDE Plasma,
- Xubuntu – desktopové prostředí Xfce,
- Lubuntu – desktopové prostředí LXDE,
- další – je možné použít Cinnamon, Deepin DDE, Budgie,...

Další varianty, kde bylo cílem měnit i jiné vlastnosti než prostředí, jsou například *EdUbuntu* (obsahuje mnoho výukových programů, je určeno zejména pro školy nebo do domácnosti pro děti), *Mythbuntu* (pro domácí multimediální centrum) nebo *Ubuntu Studio* (pro tvůrce multimediálního obsahu).

https://ubuntu.com/, https://ubuntu-mate.org/, https://kubuntu.org/, https://xubuntu.org/, https://lubuntu.me/, https://www.edubuntu.org/, https://ubuntustudio.org/

Linux Mint je distribuce založená na Ubuntu, ale dost přepracovaná. Klade důraz na jednoduchost a práci s multimédii, Na stránkách projektu jsou dostupné příručky v různých jazycích včetně češtiny.

Pro distribuci Linux Mint bylo vyvinuto desktopové prostředí Cinnamon, dále je možné použít prostředí Mate, KDE nebo Xfce. Mate a Cinnamon jsou určeny pro běžná zařízení, KDE má o něco vyšší nároky na procesor a paměť, naopak Xfce je určeno pro zařízení s nízkým výkonem.

Adresy: https://linuxmint.com/, https://www.linuxmint.com/documentation.php

Arch Linux je distribuce původem z Kanady, nezávislá na původních dvou „předcích“ (RedHatu a Debianu). Používá svůj vlastní balíčkovací systém (pacman), ve kterém klade důraz na hlídání závislostí mezi balíčky softwaru a možnosti sdílení pozměněných balíčků mezi uživateli přes repozitář. Lze ji nainstalovat s různými desktopovými prostředími od těch odlehčených (Xfce) až k velmi vybaveným (Gnome, KDE, Mate, Cinamon).

Arch Linux se stal také základem zajímavých distribucí včetně bezpečnostní distribuce *Security Onion* určené například pro dohledová centra nebo bezpečnostní týmy.

Další zajímavou distribucí odvozenou z Arch Linuxu je *Manjaro*. Uživatelé si mohou nainstalovat více různých jader Linuxu (to ostatně jde v jakékoliv linuxové distribuci, zde je pouze větší podpora výběru). Vývojáři zde kladou velký důraz na automatickou detekci a správu hardwaru, jsou přidány speciální skripty pro správu grafických karet. Je na výběr mezi různými grafickými prostředími nebo správci oken, včetně Awesome, Xfce, Mate, KDE a dalších. Existuje také varianta pro Raspberry Pi.

Adresy: https://archlinux.org/, https://securityonionsolutions.com/software, https://manjaro.org/

Elementary OS je distribuce odvozená z Ubuntu, která cílí hlavně na začátečníky. Vzhled připomíná MacOS, ale je odlehčená, účelem je, aby systém nebyl moc přeplácaný. Používá obvykle desktopové prostředí Pantheon, ovšem aplikace hodně přejímá z Ubuntu. Preferuje stručnost, bohužel i v popisu systému na webu.

Adresa: https://elementary.io/

Knoppix – přestože se jedná o poměrně rozsáhlý systém, nevyžaduje instalaci (live). Najdeme zde především to nejnovější, co GNU nabízí. Není lokalizován do češtiny. Původ Knoppixu je v distribuci Debian.

Adresa: https://www.knopper.net/knoppix/index-en.html

Slackware je distribuce určená pro pokročilejší uživatele. Vyznačuje se celkem svižným chodem a stabilitou včetně souborového systému (na internetu se někteří uživatelé vyjadřují, že tam, kde jiné Linuxy selhaly nebo běží pomaleji, Slackware nemá problémy).

Pokud ji chceme nainstalovat a používat, doporučuje se předem důkladně pročíst dokumentaci dostupnou například na internetu (je i na prvním instalačním CD). Instalace probíhá v textovém režimu, ale celkem uživatele vede.

Konfigurace se většinou provádí pomocí konfiguračních souborů, existuje jen několik konfiguračních nástrojů (`mouseconfig`, `netconfig`, ...). Všechny hlavní konfigurační soubory se nacházejí v jednom

adresáři (`/etc/rc.d`), na rozdíl od většiny ostatních rozšířených distribucí (což odpovídá standardu BSD, jiné distribuce obvykle pro umístění konfiguračních souborů používají standard System V), soubory jsou poměrně dobře okomentované (v angličtině samozřejmě). Po instalaci je obvykle nutné nakonfigurovat grafické rozhraní, pokud je chceme použít.

Slackware používá systém balíčků *tgz*, ve kterém se nezachycují závislosti mezi balíčky. Pokud zjistíme, že přesto k běhu některého programu, který jsme nainstalovali, zřejmě potřebujeme nainstalovat ještě něco dalšího (tj. nefunguje), máme možnosti, jak zjistit, co programu chybí (např. příkaz `ldd`), a doinstalovat. Pro instalaci máme k dispozici kromě možnosti instalace ze zdrojů (v `.tar.gz`) pomocí `make install` také nástroj `Checkinstall`, který ze zdrojových souborů vyrobí Slackware balíček a provede instalaci. Pro práci s balíčky slouží nástroje, které mají v názvu zkratku *pkg* (např. `installpkg`).

Live distribuce se jmenuje *Slax*, ISO obraz se vejde na „kapesní“ variantu CD, a přesto se jedná o poměrně dobře vybavený systém. Slax vůbec nepotřebuje pevný disk, protože po startu systému si vytvoří v operační paměti RAM disk a umožňuje využívat i USB flash disk a samozřejmě CD (když je spuštěn, můžeme CD se Slaxem vyndat, vše je na RAM disku). V tom případě se však doporučuje mít především dostatek operační paměti, odkládací prostor totiž neexistuje. Slax také umožňuje vytvořit si vlastní live distribuci.

Adresy: http://www.slackware.com/, https://www.slax.org/

Gentoo je distribuce typická tím, že se při instalaci téměř celá (kromě instalátoru, ten musí být logicky spustitelný) kompiluje na míru počítači, kde instalace probíhá (tzv. překládaná distribuce, z toho vyplývá, že bychom měli mít dostatečně rychlé připojení jako je kabel nebo xDSL). Obvyklý způsob instalace je přes síť z archivů Gentoo, proto je vhodné mít spíše rychlejší připojení k internetu, někdy se objevuje na CD (DVD) přiložených k počítačovým časopisům. Samotná instalace samozřejmě dá větší práci než v případě jiných distribucí, a mnohem déle trvá.

Konfigurace se provádí často spíše v konfiguračních souborech, vybavenost nástroji s GUI je poněkud nižší. S tím však souvisí jedna velká výhoda – konfigurační soubory jsou dobře navržené, velmi přehledné a dobře okomentované (oproti Mandrivě a spol. přímo rajská zahrada).

Gentoo je výjmečná distribuce už kvůli zvláštnímu řešení aktualizací převzatému od FreeBSD. Balíčky jsou uspořádány do stromu *Portage*, kde lze snadno pomocí některých programů kontrolovat verze a závislosti programů. Přímo pro práci s Portage slouží program `emerge`, ale existují nástroje, které poskytují třeba i grafické rozhraní (například Genu, tento program je napsán pomocí Mono, to je open-source rozhraní pro .NET). Pro prohledávání stromu Portage se mohou používat například Esearch nebo Eix (Portagedb).

Projekt Gentoo je známý svou oboustranně výhodnou spoluprací s jinými open-source projekty, například Free/Open/Net BSD (vzájemné využívání výsledků projektů), Mac OS X (přenos Portage na Mac OS X), atd., existují projekty pokoušející se v Gentoo použít jiné jádro než Linux (GNU/Hurd, OpenSolaris). Na Gentoo jsou dnes založeny další projekty, například ChromeOS.

Adresa: https://www.gentoo.org/

System Rescue je live distribuce určená pro přístup k poškozenému systému, případné opravy a také zálohování. Protože je to live distribuce, není problém s přístupem pro zálohování k oblastem disku, které při provozu „normální distribuce“ či dokonce Windows musí být připojeny.

Najdeme zde mnoho užitečných nástrojů, např. Offline NT Password and Registry editor (dokáže změnit heslo administrátora ve Windows a pracovat s Windows registry), Dban (neobnovitelné mazání souborů), sfdisk (zálohování tabulky rozdělení disku) a samozřejmě nástroje na samotné zálohování souborů (můžeme zálohovat také přes FTP na jiný počítač).

Adresa: https://www.system-rescue.org/

Forenzní a bezpečnostní distribuce: tyto distribuce jsou určeny pro speciální účely, například prověření daného systému bez nebezpečí jeho narušení změnami, získání hůře přístupných dat (cookies, historie webového prohlížeče, hesla, konkrétní typy souborů apod.) ze systému bez jeho spuštění, prověření zabezpečení sítě, apod. Nejznámějšími forenzními a bezpečnostními distribucemi jsou Caine Linux, Kali Linux, Tails, Black Arch, BackBox, Parrot OS.

Kali Linux je penetrační distribuce, to znamená, že jejím hlavním účelem je testování počítačové sítě na zranitelnosti. Používají ji nejen hackeři, ale také administrátoři, kteří potřebují vědět, v jakém stavu je jejich síť. Kali se dá stáhnout v mnoha různých variantách – pro nainstalování, bootovací, do cloudu, předpřipravená do virtuálního počítače, kontejnery (Docker a LXC/LXD), a to pro různé platformy (64bitový, 32bitový, Apple ARM, ARM pro mobilní zařízení, Raspberry Pi).

Caine Linux je pojmenován po jednom z biblických bratrů, Kainovi. Je to forenzní distribuce, obsahuje spoustu různých nástrojů pro bezpečné získávání dat z disku, celého systému, pro práci s hesly, registrem Windows, vytváření reportů s časovými razítky, atd. Některé verze jsou určeny pouze pro live použití, jiné se dají i nainstalovat.

Tails je distribuce založená na Debianu. Její hlavní myšlenkou je co největší anonymita uživatele na internetu, což se zohledňuje v konfiguraci všech komunikačních aplikací od webového prohlížeče přes e-mailového klienta až k aplikacím pro instant messaging. Systém není příliš vhodný pro instalaci, ideální je jeho běh z bootovacího média.

Adresy: https://www.caine-live.net/, https://www.kali.org/, https://www.kali.org/tools/, https://tails.net/

Distribuce určené pro školství jsou především nadstandardně vybaveny výukovými programy a didaktickými hrami. Tyto programy si samozřejmě můžeme nainstalovat do kterékoliv jiné linuxové distribuce. Většinou najdeme alespoň sady výukových programů GCompris, KDE-Edu, ChildsPlay a Tux4Kids a samozřejmě spoustu dalších programů specializovaných na konkrétní oblast výuky. Jednou z nejznámějších je DebianEdu (SkoleLinux), Fedora Education.

Přehled je např. na https://www.linuxandubuntu.com/home/6-best-linux-distributions-for-educational-use, https://wiki.debian.org/DebianEdu/, https://fedoraproject.org/wiki/SIGs/Education

Android má také linuxové jádro. Jde o mobilní platformu, tedy primárně pro chytré mobilní telefony, PDA, tablety apod. Firma Android, která stojí za vytvořením Androidu, byla roku 2005 koupena společností Google. Následně roku 2007 se vývoje ujala organizace Open Handset Alliance (jedná se o konsorcium mnoha výrobců mobilních zařízení nebo jejich komponent, kromě samotného Googlu zde patří například Intel, HTC, Motorola, LG, NVidia, Samsung a další).

Nad jádrem je vrstva knihoven obsahujících kód a objekty (aplikace však k nim nemají přímý přístup). Další vrstvou je virtualizační prostředí Dalvik, které slouží k oddělení spuštěných aplikací od jádra a od sebe navzájem. Další vrstvou je běhové prostředí pro aplikace a pak samotné aplikace.

Aplikace jsou psány v Javě a běhové prostředí (application framework) je vlastně běhovým prostředím Javy, ale neznamená to automatickou kompatibilitu s běhovými prostředími pro jiné platformy, protože v Androidu nejsou implementovány některé jinde hodně využívané knihovny – především AWT a Swing (grafické prostředí je postaveno na vlastních knihovnách). Aplikace lze získat v obchodě Google Play nebo jinde, jsou distribuovány jako APK balíčky (tj. Android má vlastní balíčkovací systém).

Android je šířen pod open-source licencí, výrobci si ho mohou jakkoliv přizpůsobit.

Adresa: https://www.android.com/

Realtimové systémy jsou systémy, kde lze pro určité (tzv. realtimové) procesy zaručit provedení požadavku v předem stanoveném časovém rámci (do určitého počtu milisekund). Běžné operační systémy nejsou realtimové, třebaže v nich existují procesy označené jako „realtime“ (ale ve skutečnosti mají prostě vyšší prioritu než ostatní procesy).

Nejznámější čistě realtimové systémy postavené na Linuxu jsou RT Linux a RTAI, kde ve skutečnosti není jádrem Linux, ale speciální mikrojádro řídící hardware a procesy, a samotné linuxové jádro je jedním ze spuštěných procesů. Hodně rysů realtimových systémů mají Linuxy používané jako *Embedded*, tedy vestavěné v různých zařízeních.

To je jen velmi malá část existujících distribucí. Existují distribuce velmi rozsáhlé, ale třeba jen takové, které se vejdou na jedinou disketu, CD, distribuce pro nestandardní architektury.

Další informace:

Na internetu existuje hodně informací o různých distribucích včetně jejich hodnocení, zajímavé stránky jsou například https://distrowatch.com – na hlavní stránce webu máte vpravo žebříček nejoblíbenějších distribucí. Seznam ISO obrazů mnoha live distribucí najdeme také na https://livecdlist.com/.

Úkol

Vyberte si některou z linuxových distribucí a zjistěte o ní podrobnější informace – aktuální verze, hardwarové platformy, používaná grafická prostředí, zda existuje live verze, komerční/komunitní, jaké instalační balíčky používá (většinou DEB nebo RPM), případně z jaké distribuce je odvozena či jaké distribuce jsou odvozeny z ní, typické konfigurační nástroje a jiné specifické vlastnosti, dále kde najít ISO obrazy instalačních (nebo live) médií, kde je dokumentace, HOWTO („Jak na to“ dokumenty) nebo diskusní fóra. Podívejte se také po screenshotech distribuce (obrazovkách).

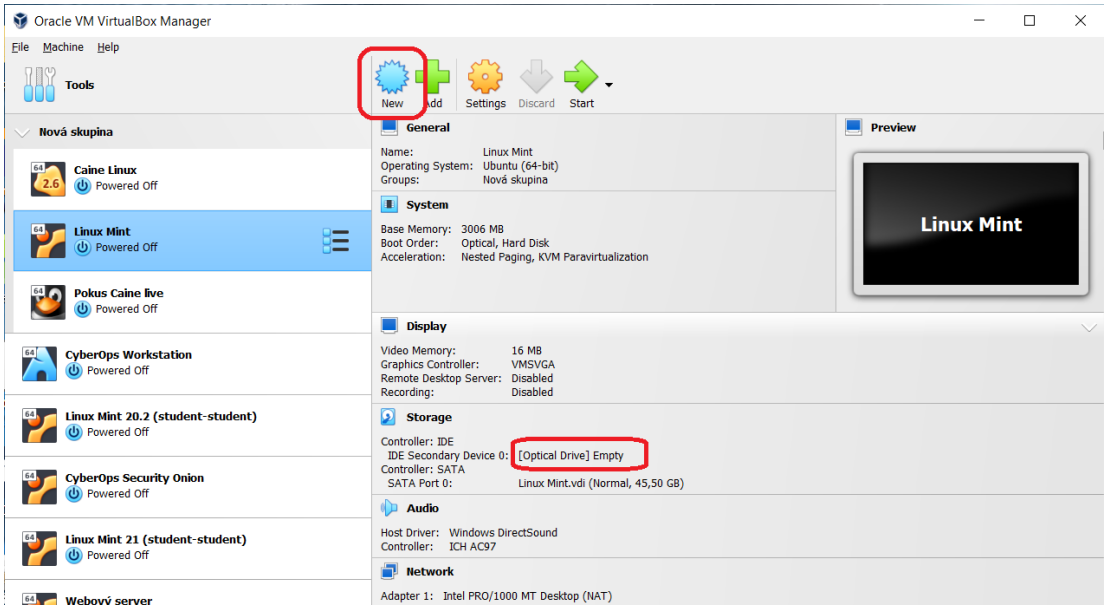
2.6 Jak si vyzkoušet Linux

Linux samozřejmě lze nainstalovat jako hlavní operační systém na počítač nebo jeden z několika (každý operační systém by měl vlastní oddíl na disku), ale existují jednodušší možnosti, jak ho vyzkoušet.

Většinu linuxových distribucí seženeme v live edici, mnohé distribuce se dokonce instalují z live média (stáhneme si ISO soubor, vypálíme na CD nebo DVD (podle velikosti) nebo si vytvoříme bootovací USB flash disk, pak nabootujeme z tohoto média).

Ještě jednodušší je však využití některého *virtualizačního nástroje*. Dá se použít některý z těchto:

- *Oracle VM VirtualBox* – volně šiřitelný svižný nástroj pro různé platformy, rozhraní může být v češtině, pro většinu studentů asi nejlepší řešení, dostupné na https://www.virtualbox.org/,
- VMWare Workstation – o něco komplexnější nástroj (zejména v edici Pro), ale komerční, lze si stáhnout trial verzi na https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html.



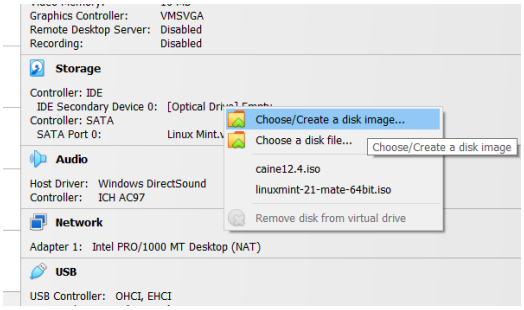
Obrázek 2.4: Prostředí aplikace Oracle VM VirtualBox

Do kteréhokoliv z výše uvedených nástrojů si můžeme nainstalovat jakékoliv množství linuxových distribucí, což je pro vyzkoušení ideální. Stačí si stáhnout .iso soubor (například z odkazů uvedených na předchozích stranách), ve virtualizačním nástroji vytvořit nový virtuální počítač, do jeho (virtuální) optické mechaniky připojit .iso soubor a (virtuálně) ten počítač spustit.

Na obrázku 2.4 nahoře je červeně vyznačeno tlačítko pro vytvoření nového virtuálního počítače. Po klepnutí na toto tlačítko se zobrazí okno, které vidíme na obrázku 2.6 vlevo nahoře. Tam vyplníme název, můžeme určit umístění virtuálního počítače na našem (skutečném) počítači, a také bychom měli zvolit správný typ systému, který chceme mít uvnitř. ISO obraz zatím nebudeme načítat.

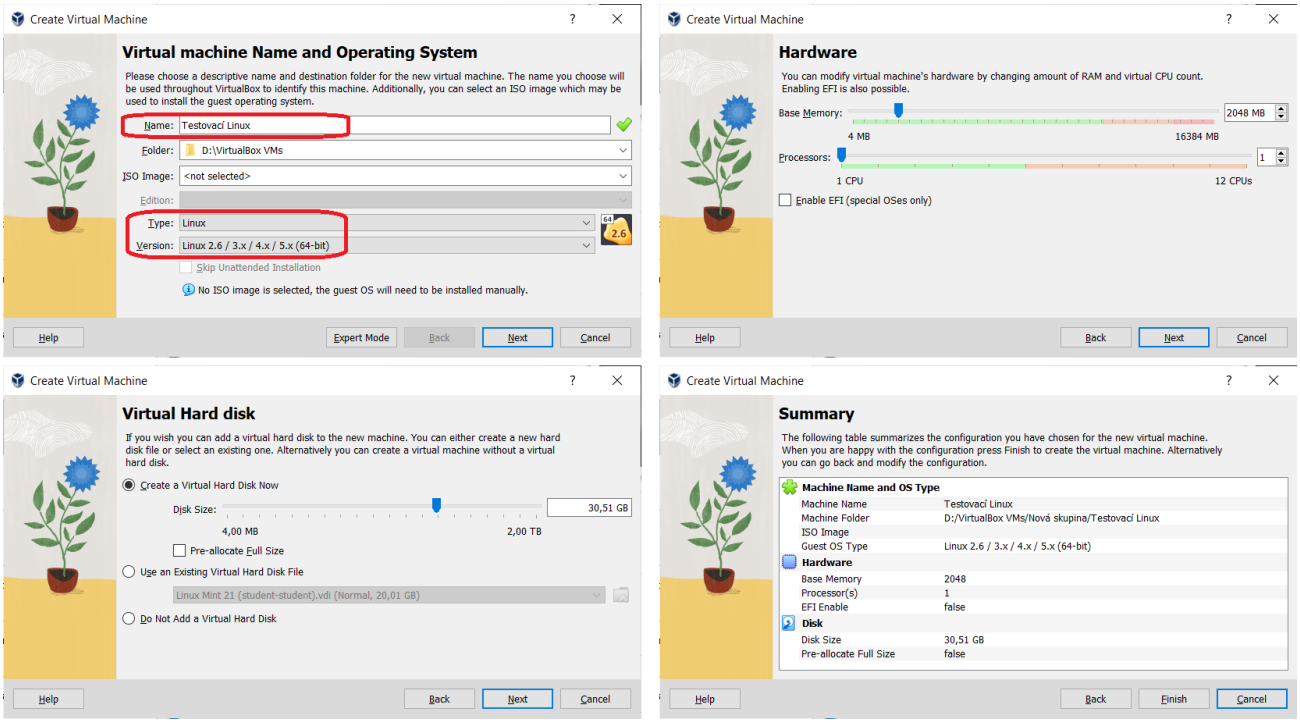
Dále budeme dotázáni na velikost operační paměti (propůjčí se z našeho reálného počítače virtuálnímu počítači, když zrovna bude spuštěný), pro Linux doporučuji alespoň 2 GB (tj. 2048 MB, jak vidíte na obrázku 2.6 vpravo nahoře).

Na dalším snímku budete dotázáni na velikost vyhrazeného místa na disku – záleží na konkrétní distribuci, kolem 30 GB by mělo bohatě stačit (viz obrázek 2.6 vlevo dole). Mnohé distribuce mohou mít méně. Tento prostor ve skutečnosti není na reálném disku vyhrazen hned celý, ale vyhradí se pouze skutečně využité místo, stanovujeme maximální hranici.



Obrázek 2.5: Vložení ISO obrazu do virtuálního počítače

Po dokončení celé procedury by se nám měla v hlavním okně VirtualBoxu objevit nová položka pojmenovaná tak, jak jsme při vytváření určili. Pak stačí napojit stažený .iso



Obrázek 2.6: Vytvoření nového virtuálního počítače

soubor do virtuální mechaniky – na obrázku 2.4 je taktéž naznačeno příslušné místo. Obrázek 2.5 naznačuje, co máme po klepnutí na virtuální optickou mechaniku vybrat. Pak jen najdeme příslušný soubor.

Pak už stačí jen spustit virtuální počítač. To provedeme jednoduše tak, že poklepeme na virtuální počítač, případně nahoře klepneme na tlačítko Start. Virtuální počítač začne bootovat z výměnného média (našeho ISO obrazu).

Výhodou je, že pokud si chceme vyzkoušet několik různých distribucí, stáhneme si jejich ISO obrazy a postupně je připojíme do virtuální mechaniky, spustíme, atd., ukončíme, pak do virtuální mechaniky připojíme jiný obraz,...

Kapitola 3

Práce v grafickém uživatelském rozhraní

Rychlý náhled: Linux je dnes většinou instalován s grafickým uživatelským rozhraním. Už v předchozí kapitole jsme se seznámili s principem grafického rozhraní v Linuxu, a tedy víme, že na každém systému může grafické prostředí vypadat úplně jinak (různí správci oken, desktopová prostředí, to vše může být různě nakonfigurováno), tedy obsah této kapitoly berte spíše orientačně.

V této kapitole se zaměříme na správu systému Linux s využitím nástrojů grafického rozhraní. Podobně jako v případě Windows, i zde máme kapitolu rozdělenou podle různých typů úloh. Podíváme se na možnosti přizpůsobení prostředí, správu zařízení (zaměříme se zejména na paměťová média), konfiguraci sítě, správu softwaru, procesů, souborů, a správu uživatelů.

Klíčová slova: Okno, virtuální plocha, paměťové médium, periferní zařízení, napájení, počítačová síť, správa softwaru, aktualizace, uživatel, skupina

Cíle studia: Po prostudování této kapitoly zvládnete systém Linux v roli uživatele. Naučíte se přizpůsobit si pracovní prostředí s využitím nejznámějších grafických prostředí a najít běžné nástroje pro konfiguraci uživatelů, zařízení a sítě.

Poznámka:

Následuje popis různých úkolů v grafickém rozhraní distribuce Linux Mint verze 17.3 Rosa, desktopové prostředí Mate verze 1.12.0. Ovšem není to výchozí podoba prostředí po instalaci, vše již prošlo úpravami. Pokud máte jinou linuxovou distribuci nebo desktopové prostředí (správce oken) či v jiné verzi, setkáte se zřejmě s odlišnosti ve vybavenosti aplikacemi, vzhledu a ve způsobu práce v systému.

3.1 Základní průzkum systému

3.1.1 Okna, plochy, virtuální plochy

Ve většině desktopových prostředí (či správců oken) můžeme používat pracovní plochu, panely, ikony.

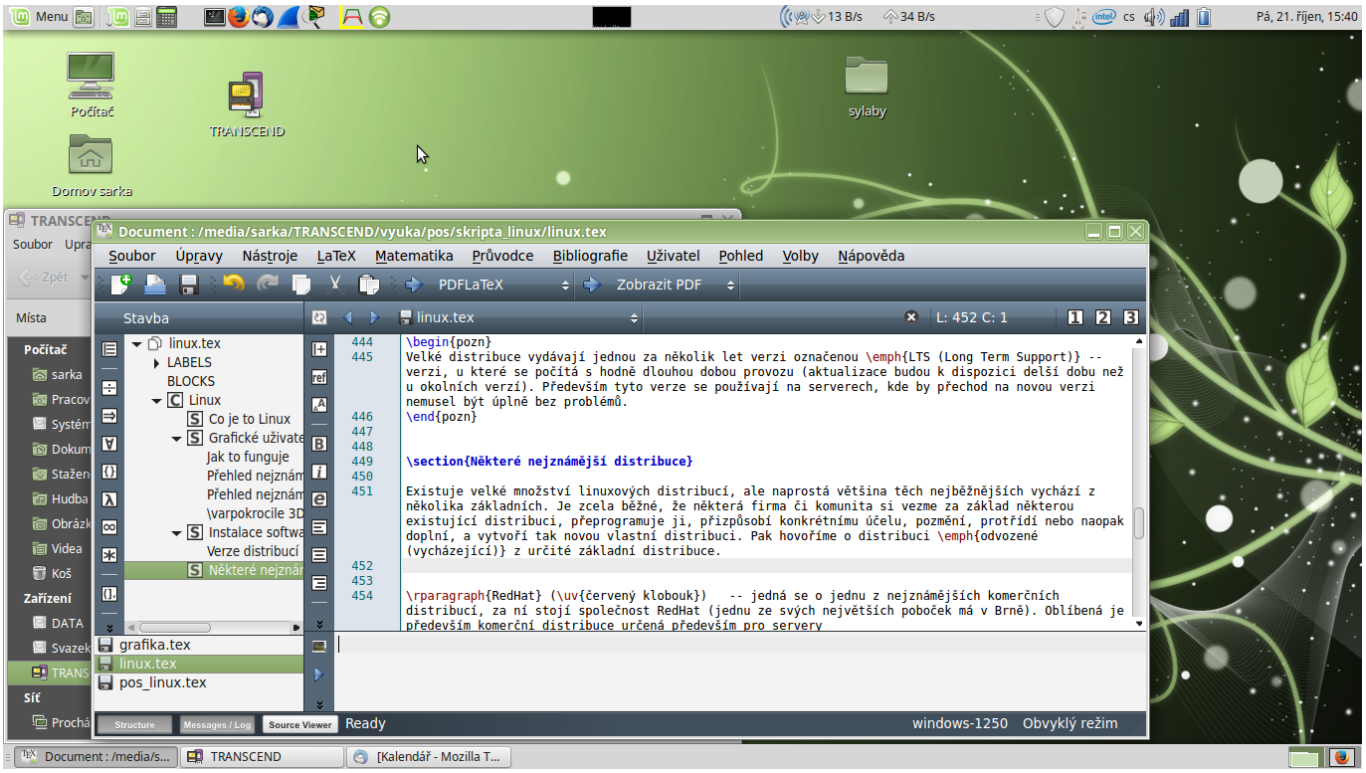
Obvykle máme k dispozici několik *virtuálních ploch*. Jedna z nich je zobrazena na „fyzické“ ploše, tedy aktivní, ostatní jsou pouze uloženy pro pozdější zobrazení.

Aplikace běží vždy na některé z virtuálních ploch (tj. pokud je dotyčná virtuální plocha aktivní, vidíme i okno aplikace), mezi virtuálními plochami je možné aplikace přesouvat obvykle pomocí položky v kontextovém menu či podobně. Obvykle máme k dispozici přepínač virtuálních ploch, kde myší určujeme, která virtuální plocha má být aktivní (zobrazena), případně lze použít klávesovou zkratku.

V X Window jsou objekty uživatelského rozhraní (okna, plochy, tlačítka, menu, atd.) uspořádány do stromu. V kořeni stromu je hlavní okno celého prostředí, jeho potomci jsou běžná okna (obvykle patřící aplikacím), dále ve struktuře máme podokna a další vnořené prvky (například pokud máme v okně aplikace několik tlačítek, menu a boxy s popisky, to vše budou ve stromě potomci okna aplikace).

3.1.2 Pracovní prostředí

Podle obrázku 3.1 si vysvětlíme jednotlivé součásti pracovního prostředí.



Obrázek 3.1: Prostředí Mate v distribuci Linux Mint

Přímo na obrázku vidíme tyto prvky:

- pod okny prosvítá *pracovní plocha* s tapetou (obrázkem na pozadí); pokud například chceme změnit obrázek na pozadí, použijeme pravé tlačítko myši,
- na pracovní ploše máme *několik ikon*, z nichž jedna (Transcend) patří USB flash disku (objevila se po jeho připojení, flash disk odpojíme pomocí položky pro bezpečné odpojení v kontextovém menu této ikony),
- na spodním okraji (v tomto případě) vidíme *panel úloh*, kde se dostaneme k tlačítkům spuštěných aplikací,
- vpravo dole na panelu úloh je *přepínač virtuálních ploch* – momentálně jsou nastaveny dvě virtuální plochy (konfigurace pravým tlačítkem myši na přepínači) a na každé je nejméně jedna aplikace, aktivní je právě první plocha,

- na horním okraji plochy (opět v tomto případě, může tomu být jinak) najdeme *hlavní panel*, který opět konfigurujeme pomocí voleb v kontextovém menu (pravé tlačítko myši),
- na hlavním panelu vlevo je přístup do *systémového menu*, kde najdeme různé aplikace a konfigurační nástroje,
- další ikony na hlavním panelu jsou *spouštěče aplikací* a nejrůznější *applety* (přidávají se a konfigurují přes kontextové menu),
- zcela vpravo na hlavním panelu je *oznamovací oblast* (tu máme i ve Windows), přičemž její konfigurace se také provádí pomocí kontextového menu (pravé tlačítko myši na jezdec na začátku oznamovací oblasti).

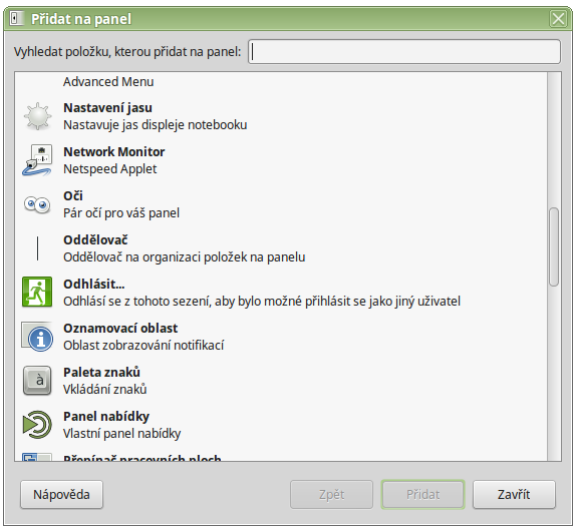
Zaměříme se na hlavní panel, který na obrázku máme u horního okraje obrazovky:

- za ikonou systémového menu je ikona „zobrazit plochu“ (minimalizuje aplikace na ploše),
- další ikona systémového menu (alternativní řešení, tady byla přidána z pokusných důvodů),
- odkaz na správce souborů,
- spouštěče aplikací – zde konkrétně kalkulačka, terminál (obdobu Příkazového řádku z Windows), Firefox, Thunderbird, Wireshark, Cisco Packet Tracer, za nimi dvě navzájem si „konkurující“ aplikace pro analýzu signálu Wi-fi sítí,
- „černé okénko“ je applet pro sledování systému (procesor je momentálně minimálně vytěžován, jinak by se zde zobrazoval graf vytížení procesoru), po klepnutí na applet se spustí nástroj *Sledování systému*, který můžeme brát jako obdobu Správce úloh z Windows,
- applet pro sledování činnosti momentálně aktivního síťového rozhraní (právě je aktivní Wi-fi),
- vpravo jako první ikona oznamovací oblasti je *ikona pro aktualizace*, která nejen indikuje, jestli jsou dostupné aktualizace pro některý balíček (jsou: , nejsou:), ale také poklepáním spustíme aktualizační nástroj, ve kterém můžeme povolit proces aktualizace,
- zbývá několik dalších appletů (přístup k senzorům, přepínač grafických karet, přepínač české/anglické klávesnice, zvuk, síť, baterie) a ukazatel data a času.

Obecně platí, že pokud chceme konfigurovat něco, pro co zde máme applet či spouštěč, můžeme použít kontextové menu (taky panely se konfigurují tímto způsobem – přidání nového panelu, konfigurace polohy, vzhledu, velikosti a obsahu panelu).

Pokud chceme na (jakýkoliv) panel přidat nový applet či spouštěč, zobrazíme si kontextové menu panelu a vybereme volbu *Přidat na panel*. objeví se okno podobné tomu, které vidíme na obrázku 3.2, v něm už si vybereme konkrétní položku.

Nový prvek se přidá na to místo panelu, na které jsme předtím klepli. Pokud ho chceme přesunout, zobrazíme kontextové menu prvku (ne panelu) a zvolíme *Přesunout*. Kurzor se změní na ručičku, kterou přesuneme prvek na konkrétní místo. Když to nejde (volba pro přesun není aktivní), je zřejmě poloha prvku uzamčena – odemčení provedeme opět přes kontextové menu.



Obrázek 3.2: Přidání appletu či spouštěče

3.1.3 Systémové menu

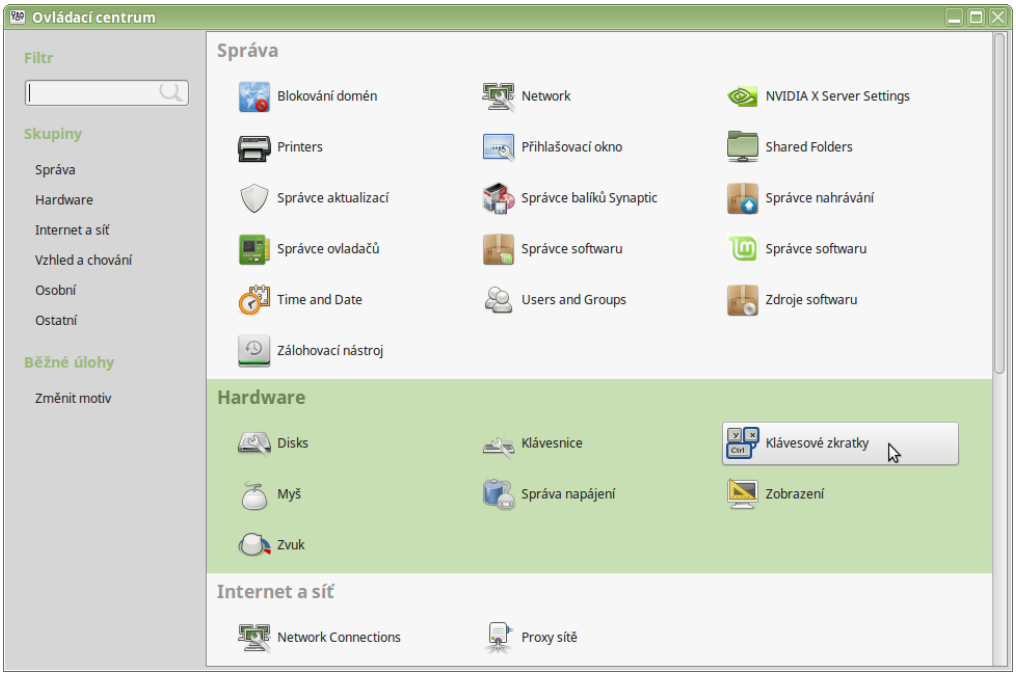
Tlačítko pro systémové menu najdeme prakticky v každé běžné distribuci, obvykle vlevo nahoře nebo dole. Každé desktopové prostředí a taky každá distribuce si své menu uspořádává „podle svého“, většinou tam míváme

- části pro aplikace (obvykle podle kategorií) – kancelářské aplikace, hry, grafika, zvuk a video, příslušenství s drobnými aplikacemi typu kalkulačka, jednoduchý textový editor apod.,
- část pro systémové a konfigurační nástroje, některé z nich bývají zamíchány do části pro aplikace,
- důležitá místa (domovský adresář, místní síť, připojená výměnná média, koš apod.).

V distribucích používajících prostředí GNOME nebo některou jeho upravenou variantu (s knihovnou GTK+) obvykle někde v systémovém menu najdete položky *Volby* a *Správa*. V první z nich jsou většinou nástroje pro konfiguraci toho, co nějak souvisí s konkrétním uživatelem a nebývá nutné mít vyšší přístupová oprávnění (ale neplatí to na 100 %). V druhé z těchto položek jsou již volby, pro které většinou potřebujete vyšší přístupová oprávnění (například správa ovladačů, konfigurace přihlašovacího okna, instalace softwaru a jeho zdrojů, konfigurace uživatelů). V každém případě naprostou většinu konfiguračních nástrojů najdete právě v některé z těchto položek. Další běžná umístění konfiguračních nástrojů jsou v položkách *Systémové nástroje* a *Příslušenství*.

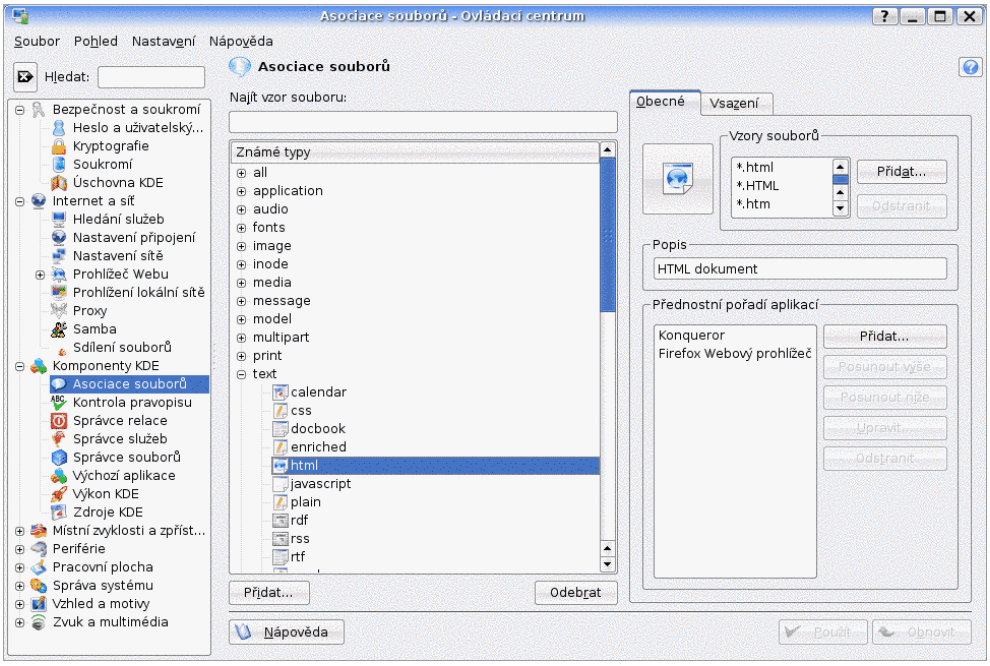
3.1.4 Ovládací centrum

Pro jednotlivé konfigurační úlohy obvykle míváme různé (menší, specializované) nástroje, ale ve větších desktopových prostředích často najdeme i komplexní nástroj, který slouží jako společné rozhraní k více různým „menším“ nástrojům. Většinou se nazývá *Ovládací centrum* (v anglické verzi Control Center).



Obrázek 3.3: Ovládací centrum, prostředí Mate (odvozeno z GNOME)

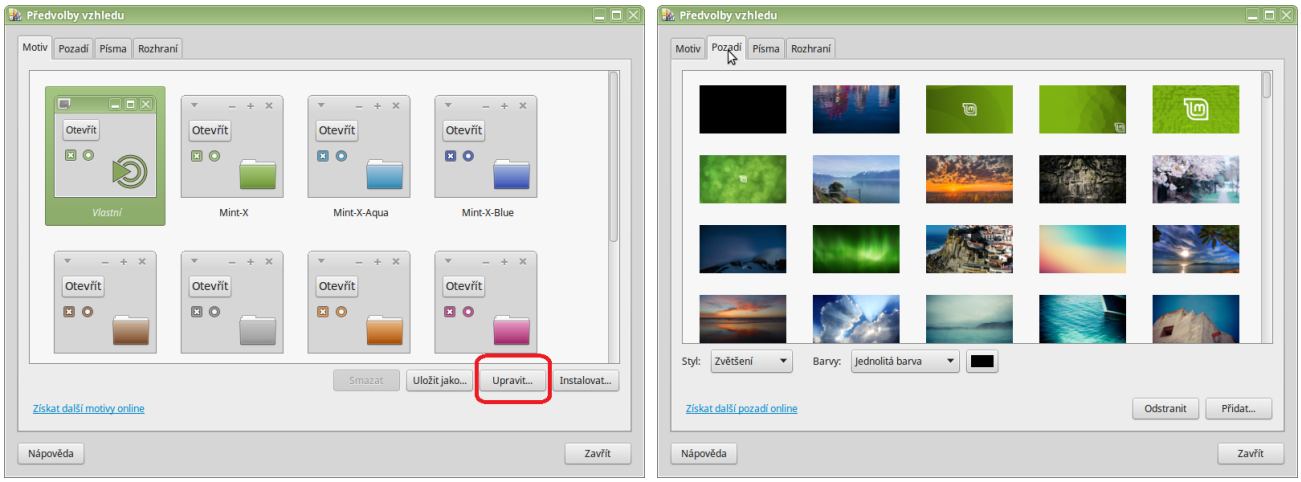
Tento nástroj může vypadat opravdu různě. Na obrázku 3.3 je *Ovládací centrum* z Linux Mint, na obrázku 3.4 je *Ovládací centrum* v KDE (jedna ze starších verzí).



Obrázek 3.4: Ovládací centrum v KDE (starší verze)

3.2 Přizpůsobení prostředí

Tuto a následující sekce berte jen jako inspirační – záleží, kterou distribuci Linuxu si zvolíte, případně také které desktopové prostředí. Odvozená desktopová prostředí a v některých případech i konkrétní distribuce obvykle obsahují prakticky tytéž konfigurační nástroje, jen se třeba mohou trochu jinak jmenovat nebo k nim může být trochu jiný přístup.



Obrázek 3.5: Konfigurace vzhledu oken a plochy

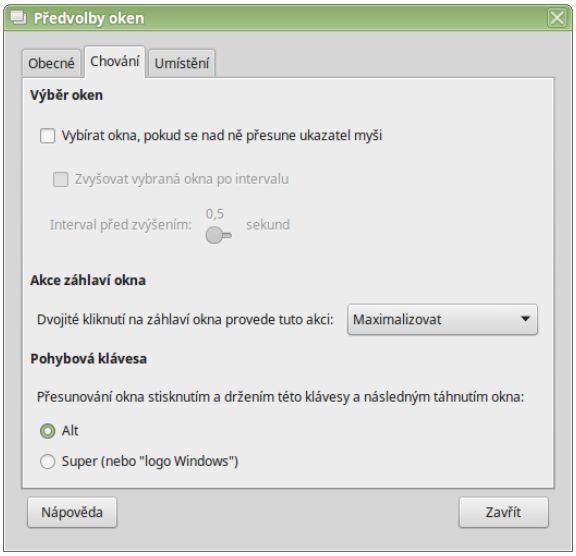
Úkol

Pokud jste si ještě nestihli vybrat některou distribuci Linuxu, teď to už rozhodně udělejte. Projděte si prostředí své vybrané distribuce a u všech níže uvedených postupů najděte, jak se dotyčné nastavení provádí konkrétně v této distribuci.

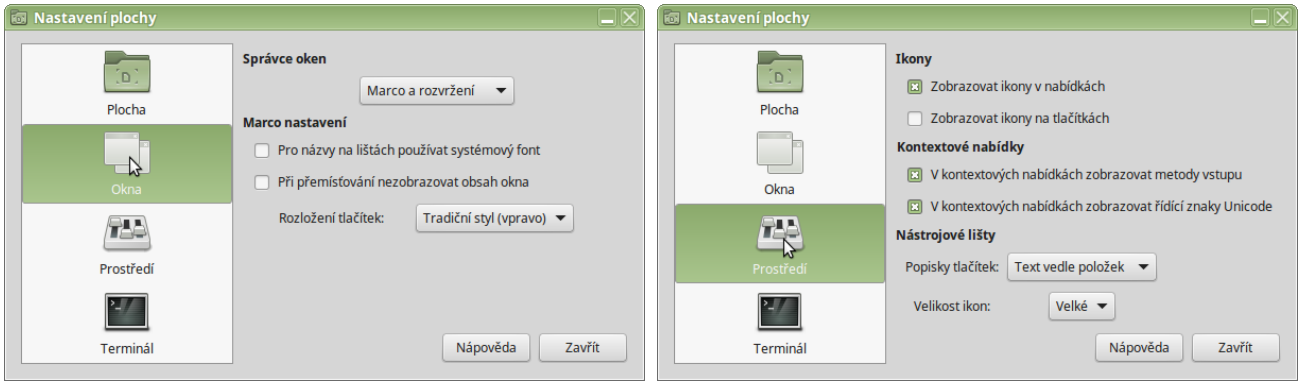
Pro konfiguraci vzhledu oken se v Linux Mint s prostředím Mate používá nástroj *Vzhled* dostupný přes systémové menu v části *Volby* – viz obrázek 3.5.

Vlevo je záložka pro výběr motivu (tj. jak má vypadat titulek okna, tlačítka, zatrhávací pole apod.), přes tlačítko „Upravit“ můžeme pozměnit jakýkoliv parametr motivu (barvy, okraje oken, vzhled ovládacích prvků, vzhled ikon a kurzoru). Na obrázku vpravo je záložka s obrázky na pozadí.

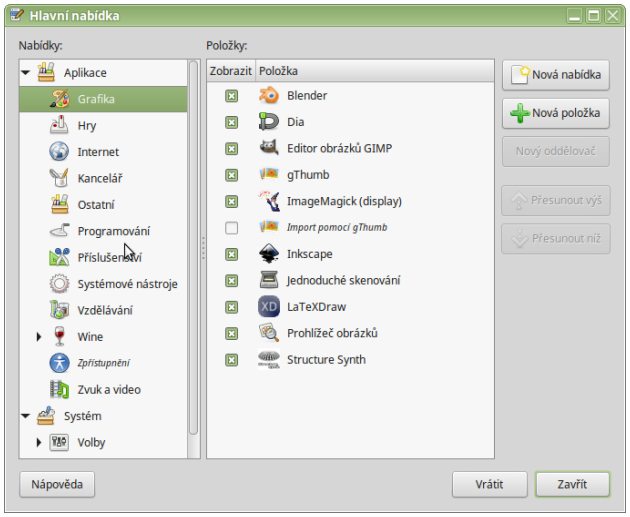
Chování oken se konfiguruje v nástroji *Okna* (v titulku okna je *Předvolby oken*) – kdy má okno získat zaměření (výchozí je po zaktivnění okna například myší nebo klávesovou zkratkou, ale můžeme zvolit, že se má okno zaktivnit po najetí myší), co se má stát při poklepání na záhlaví okna, apod.



Obrázek 3.6: Nastavení chování oken



Obrázek 3.7: Konfigurace plochy a oken



Obrázek 3.8: Konfigurace systémového menu

Nastavení související s okny máme také v nástroji *Nastavení plochy*, kde konfigurujeme svého správce oken. Druhá a třetí část nastavení jsou vidět na obrázku 3.7. V první části se určuje, které ikony mají být zobrazeny na ploše, v druhé se například dá stanovit, zda na titulkovém pruhu okna mají být systémová tlačítka pro uzavření, minimalizaci a maximalizaci zobrazena vpravo nebo vlevo (styl Mac), na třetí určujeme, jestli se mají zobrazovat ikony u položek menu a na předdefinovaných tlačítkách (Otevřít, Nápověda apod.), jak mají vypadat kontextová menu a popisky tlačítek na toolboxech.

Obsah systémového menu (dostupného z hlavního panelu – viz str. 30) určujeme v nástroji *Hlavní nabídka* (obrázek 3.8). Můžeme zde určovat zanoření (strukturu), pořadí, přidávat nové položky, znezviditelňovat nebo naopak zviditelňovat neaktivní jednoduše zaškrtnutím pole před položkou.

3.3 Správa zařízení

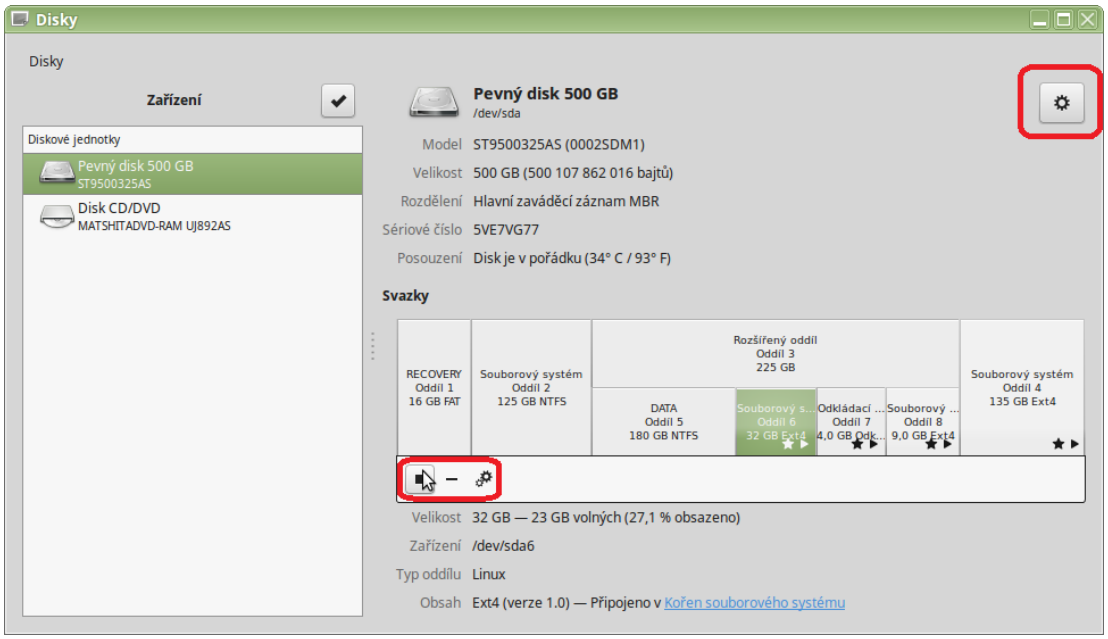
Také v Linuxu máme ovladače, jen se o ně obvykle vůbec nemusíme starat. Pokud však chceme jiný ovladač než výchozí a seženeme ho, můžeme ho samozřejmě také nainstalovat. Tím je obvykle míněno, že se dají nainstalovat například proprietární ovladače pro grafické karty pocházející od výrobce příslušné grafické karty. U výkonnějších grafických karet to stojí za vyzkoušení.

Obvykle existuje některý nástroj pro práci s ovladači (například *Správce ovladačů*), který automaticky zjistí, zda jsou nějaké proprietární ovladače k dispozici, a nabídne nám instalaci.

3.3.1 Paměťová média

Také pro správu disků a jiných paměťových médií najdeme nástroj v každé distribuci, přičemž velmi kvalitní je zejména ten z desktopových prostředí odvozených od GNOME.

Pro správu paměťových médií se využívá nástroj *Disky*, jehož rozhraní vidíme na obrázku 3.9. Vlevo vybíráme konkrétní disk, optickou mechaniku či USB flash disk (podle toho, co je momentálně dostupné), ve větším podokně vpravo vidíme rozdělení vybraného disku na jednotlivé oddíly (svazky). Po klepnutí na konkrétní oddíl se dole vypíšou jeho vlastnosti.



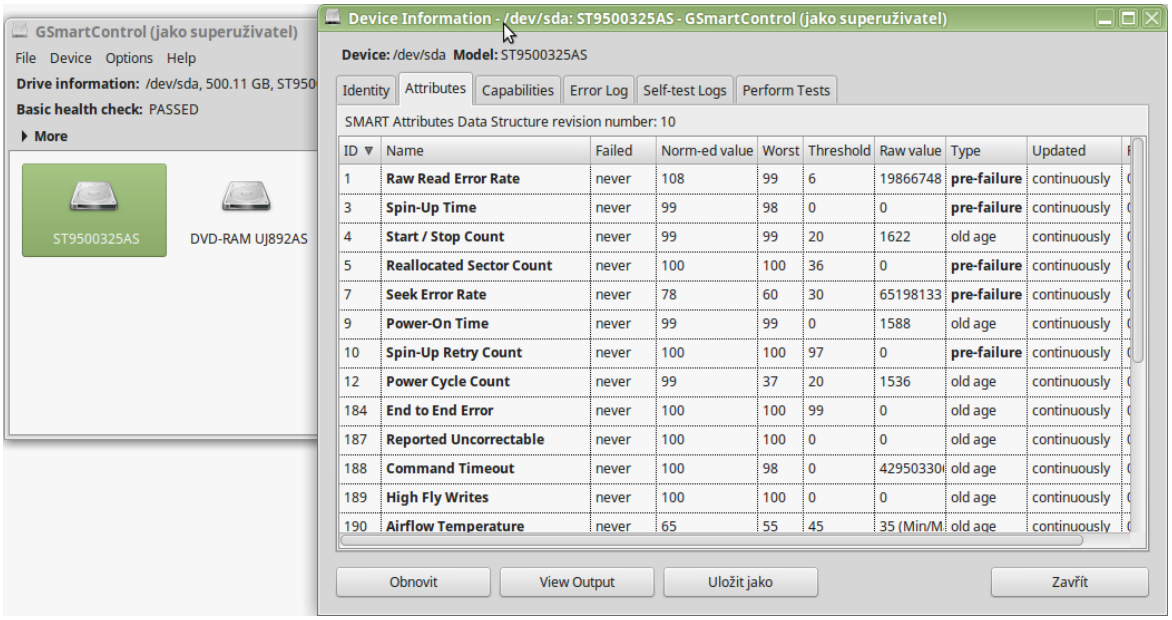
Obrázek 3.9: Nástroj pro správu paměťových médií

Mezi obrázkem s rozdělením disku a popisem vlastností je oblast se třemi ikonami:

- první z nich umožňuje oddíl připojit nebo odpojit (ikona je ve tvaru čtverce nebo „šipky“, podle toho, jestli je oddíl momentálně připojený nebo odpojený),
- druhá slouží k odstranění oddílu (pozor),
- třetí je pro konfiguraci oddílu – můžeme ho zformátovat (vytvořit souborový systém), upravit velikost oddílu, změnit volby pro připojení (jestli se má připojovat hned po spuštění systému, zda má být zobrazován běžnému uživateli, pro jaké účely má být použit, jak se má označit, atd.), také můžeme například vytvořit obraz disku.

Vpravo nahoře pak máme přístup ke konfiguraci disku jako celku včetně voleb souvisejících s úsporou energie, například tam můžeme určit, po jaké době nečinnosti disku bude systém uspán.

U disků se někdy hodí možnost sledovat životnost disku – hodnoty určitých parametrů, které mohou napovědět, že se blíží možné selhání (hodnoty S.M.A.R.T. – Self Monitoring and Reporting Technology). V desktopových prostředích vycházejících z GNOME je k tomu účelu nástroj *GSmartControl*, viz obrázek 3.10.



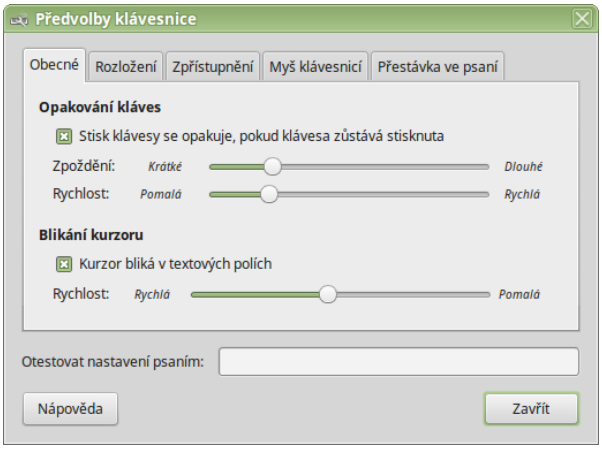
Obrázek 3.10: Sledování životnosti disku

Po vybrání konkrétního disku dostaneme okno, kde na první záložce jsou vlastnosti disku a na druhé jsou právě hodnoty S.M.A.R.T. Důležité jsou sloupečky *Normed Value*, *Worst* a *Threshold*: první z nich je hodnota pro daný parametr normovaná do intervalu 0–100 (0 je nejhorší, tedy selhání, 100 je perfektní), přičemž přepočet nebývá moc přesný – někdy vyjde číslo větší než 100. V dalším sloupečku najdeme nejhorší zatím dosaženou hodnotu (takže pohled do historie), z toho vyplývá, že zde bude číslo stejné nebo menší než v předchozím sloupci. Threshold znamená „bezpečnostní hranici“ – pokud Normed Value nebo Worst klesne pod tuto hranici, měli bychom začít jednat (zálohování, případně nákup nového disku).

3.3.2 Periferní zařízení

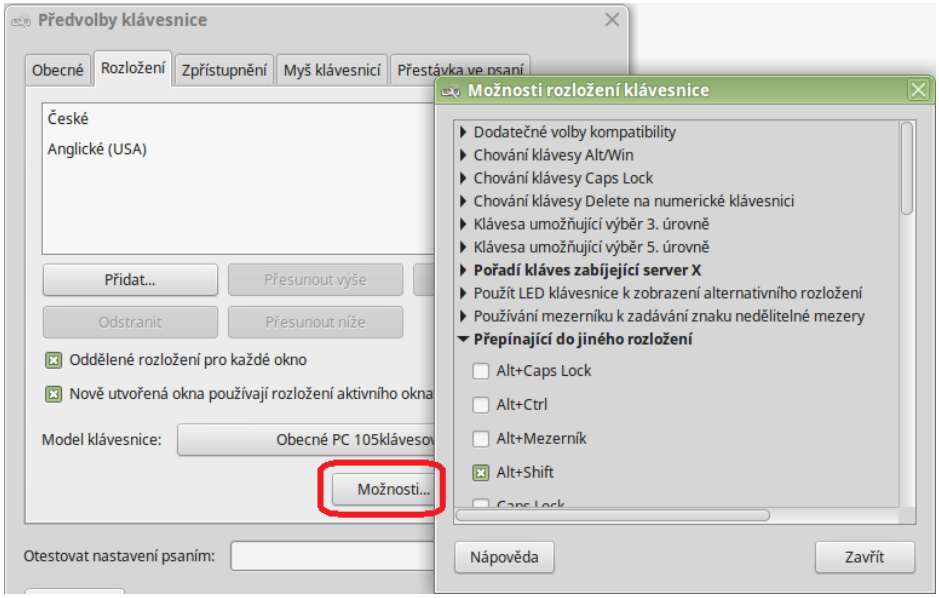
Pro nastavení klávesnice máme nástroj *Klávesnice* (v záhlaví okna je nápis *Předvolby klávesnice*). Je to okno s několika záložkami, přičemž na první záložce (obrázek 3.11) nastavujeme běžné věci typu jak rychle se generují znaky, když držíme klávesu stisknutou, nebo jak rychle má blikat kurzor.

Na druhé záložce (viz obrázek 3.12) určujeme rozložení klávesnice (informatikovi se hodí české a anglické, po instalaci můžeme podle potřeby přidat chy-



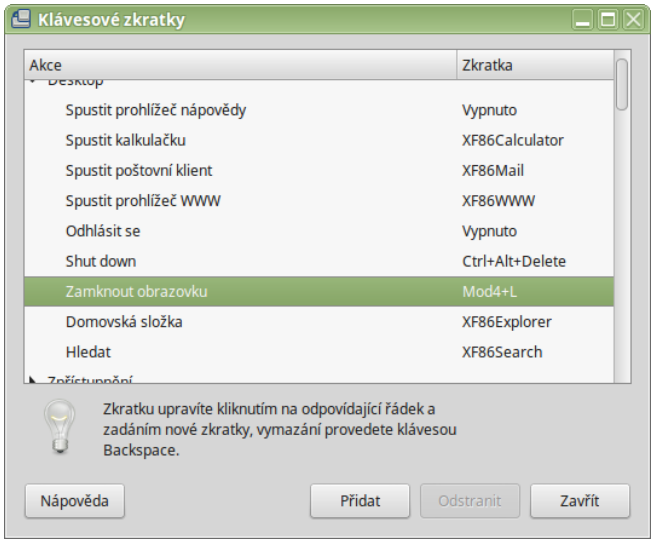
Obrázek 3.11: Základní volby pro klávesnici

bějící), také se tu volí klávesová zkratka pro přepínání mezi jednotlivými rozloženími. Okno s touto volbou (a dalšími nastaveními) dostaneme klepnutím na tlačítko *Možnosti*. Ti, kdo na Linux přecházejí z Windows, si zde mohou navolit stejnou klávesovou zkratku, jakou mají vžitou.



Obrázek 3.12: Nastavení klávesové zkratky pro přepínání mezi rozloženími klávesnice

Ke klávesnici patří klávesové zkratky, pro které existuje samostatný nástroj *Klávesové zkratky*. Jednotlivé činnosti, pro které může být zvolena klávesová zkratka, jsou uspořádány do kategorií ve stromové struktuře. Když chceme změnit či nastavit klávesovou zkratku ke konkrétní činnosti, stačí ji zvolit (třeba myší) a pak přímo stisknout zvolenou klávesovou zkratku. Na obrázku 3.13 je uzamknutí obrazovky zvolena klávesová zkratka ⊞+L („okenní“ klávesa je tady označena zkratkou Mod4).

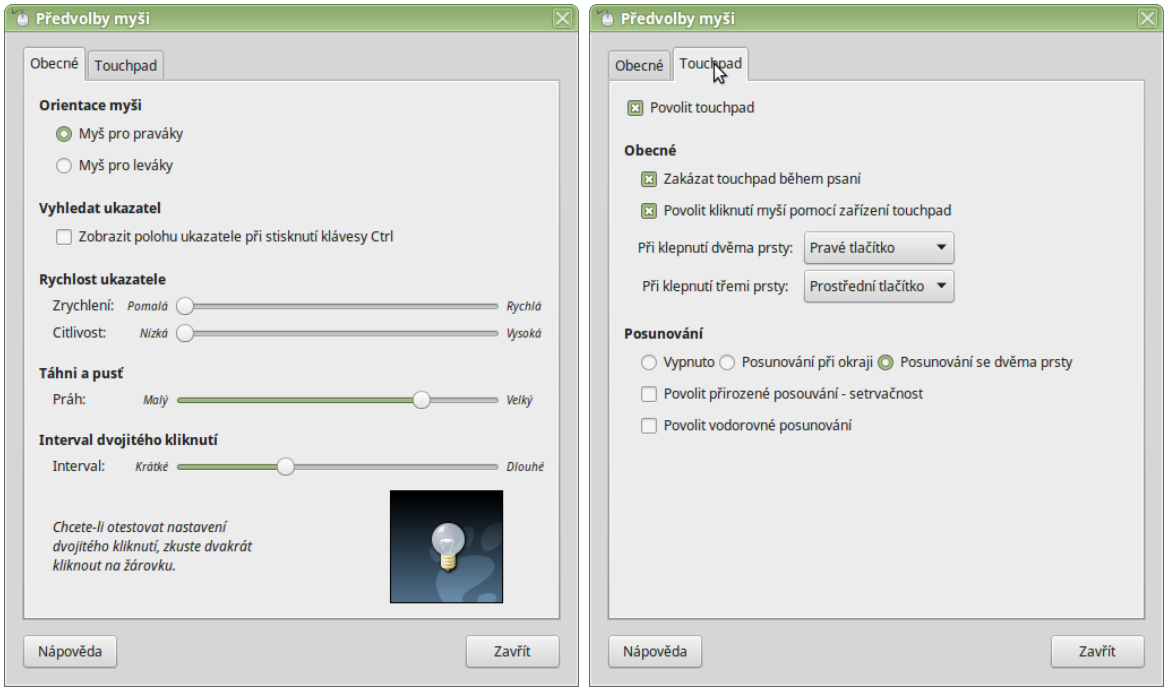


Obrázek 3.13: Klávesové zkratky

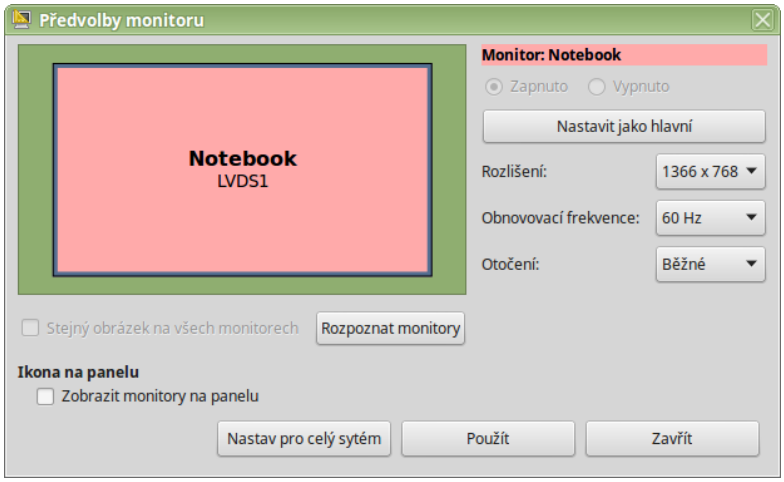
Pozor, v prostředích vycházejících z GNOME v tomto nástroji nenajdeme klávesovou zkratku pro přepínání mezi různými rozloženími klávesnice, ta se určuje v nástroji pro konfiguraci klávesnice na záložce pro rozložení, jak jsme si ukázali výše.

Další periferní zařízení, která může být užitečné konfigurovat, jsou myš a touchpad. Obojí nastavujeme v nástroji *Myš* (obrázek 3.14), přičemž první záložka odpovídá názvu nástroje (konfigurujeme tam myš) a druhá slouží ke konfiguraci touchpadu (pokud jsme jím vybaveni).

U myši nastavujeme orientaci pro leváky nebo praváky, a pak některé „rychlostní ukazatele“. Na druhé záložce můžeme touchpad úplně zakázat (třeba pokud výhradně používáme myš) nebo zakázat během psaní na klávesnici, můžeme nastavit akce při klepnutí na touchpad dvěma nebo třemi prsty nebo určit, jak má fungovat posouvání.



Obrázek 3.14: Konfigurace myši a touchpadu



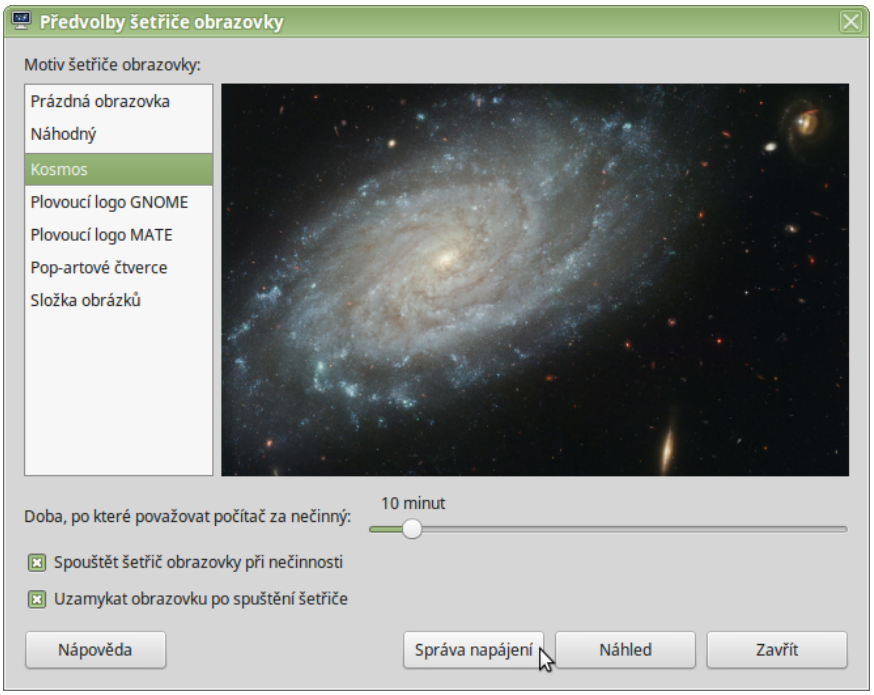
Obrázek 3.15: Nastavení zobrazení na monitoru nebo projektoru

Pro konfiguraci zobrazovacích zařízení (monitoru, projektoru apod.) máme nástroj *Zobrazení* (v titulku okna se vypisuje *Předvolby monitoru*, viz obrázek 3.15). Pro vybrané zobrazovací zařízení určujeme rozlišení, frekvenci a otočení (pokud máme připojeno víc zobrazovacích zařízení, předem vybereme to, které chceme nastavovat). Při více zařízeních určujeme, zda se na všech má objevit stejný obraz nebo jestli mají mít různý obsah (nastavujeme v době, kdy jsou tato zařízení reálně připojena a rozpoznána) – volba *Stejný obrázek na všech monitorech* vlevo uprostřed (na obrázku 3.15 je neaktivní, protože je připojeno jen jedno zobrazovací zařízení).

Pokud často připojujeme různá zobrazovací zařízení a potřebujeme průběžně měnit jejich nastavení, může být praktické mít zatrhnutou volbu *Zobrazit monitory na panelu* (v okně vlevo dole). Pak se nám do oznamovací oblasti přidá ikona tohoto nástroje.

Se zobrazením souvisí nastavení spořiče obrazovky. I pro ten máme svůj nástroj, který vidíme na obrázku 3.16. Nastavujeme motiv šetřiče, po jaké době nečinnosti se má spouštět, a taky zda má

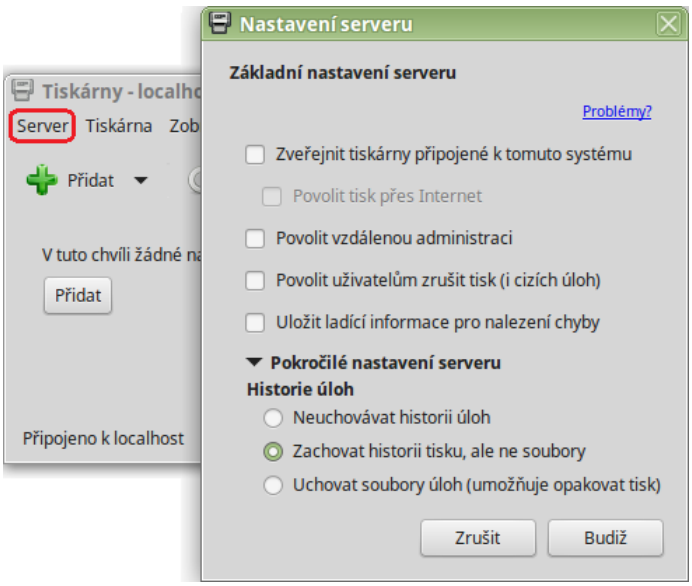
být obrazovka uzamknuta při jeho spuštění (to je rozhodně rozumné nastavení). Motivy pro šetřiče obrazovky se instalují stejně jako jiný software, ne přes tento nástroj.



Obrázek 3.16: Nastavení spořiče obrazovky

Zbývá nástroj *Tiskárny*. Zde nastavujeme vlastnosti tiskáren, procházíme tiskové fronty, určujeme, která tiskárna má být výchozí (pokud jsou nějaké tiskárny instalovány, jsou zde jejich ikony, použijeme tedy kontextové menu), můžeme přidat novou tiskárnu a taky nastavujeme obecné vlastnosti tiskového serveru (tj. vlastnosti zařízení, ke kterému máme připojenou tiskárnu).

Tato obecná nastavení se provádějí přes menu *Server* ⇨ *Nastavení*, jak vidíme na obrázku 3.17. Můžeme například tiskárnu zveřejnit do sítě nebo nechat ukládat informace o historii tisku (tj. kdo co tiskl, taky je možné zachovávat přímo i tisknuté dokumenty).

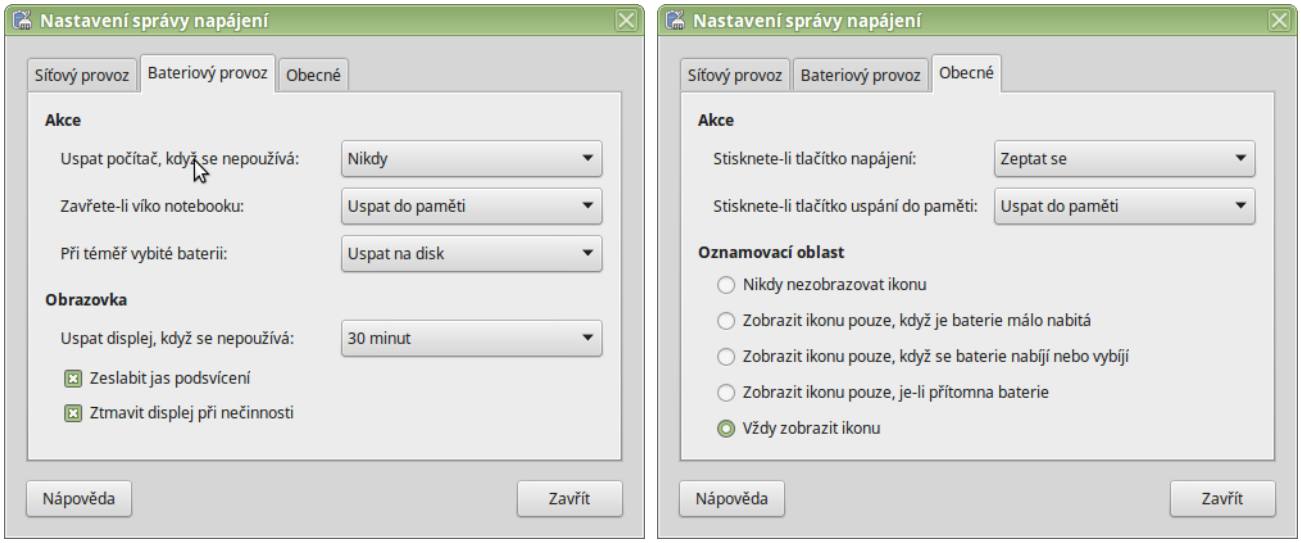


Obrázek 3.17: Nastavení tiskáren

3.3.3 Správa napájení

Také v Linuxu máme možnost řídit energetickou spotřebu zařízení, uspávat do paměti (režim spánku) nebo na disk (režim hibernace) a zjišťovat stav baterie.

V nástroji *Správa napájení* (obrázek 3.18) máme několik záložek. První dvě slouží k nastavení napájení v případech, že je zařízení napájeno buď ze sítě nebo z baterie, na obrázku 3.18 vlevo vidíme druhou z nich. Záložky mohou vypadat trochu jinak v závislosti na tom, na jakém zařízení pracujeme.

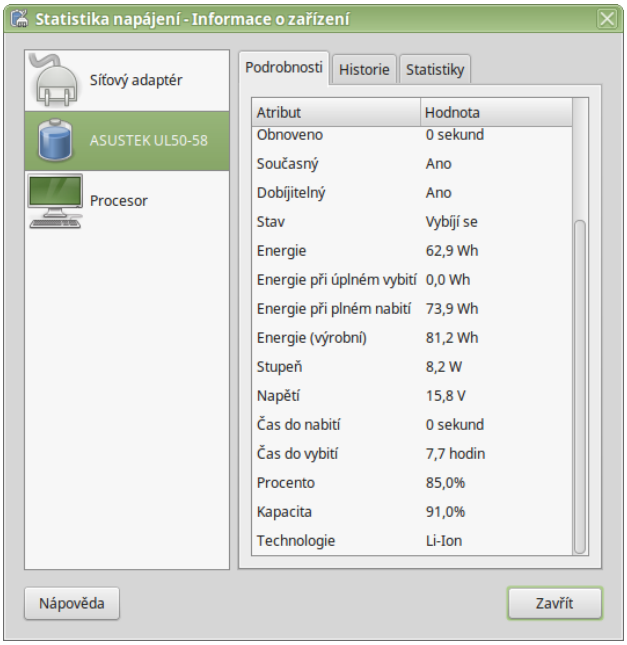


Obrázek 3.18: Nástroj pro správu napájení

Na prvních dvou záložkách nastavujeme, kdy má být počítač uspán, jakým způsobem, jaký má být jas obrazovky (má velký vliv na spotřebu), u notebooku taky akce při uzavření víka, na záložce pro baterii také akci při téměř vybité baterii.

Na třetí záložce můžeme určit, co se stane, když stiskneme vypínací tlačítko (hardwarové – vypnutí zařízení), a zda má být ikona napájení v oznamovací oblasti na hlavním panelu.

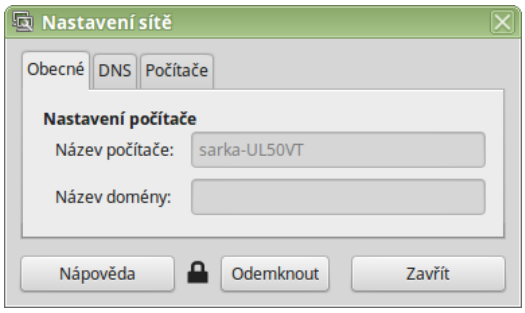
Po klepnutí na ikonu napájení v oznamovací oblasti se objeví seznam připojených baterií (u notebooku zřejmě jedna). Po klepnutí na položku pro konkrétní baterii se spustí nástroj *Statistika napájení* (viz obrázek 3.19), ve kterém kromě jiného najdeme podrobné informace o baterii (nástroj je i v Systémových nástrojích). Je tady také údaj o kapacitě baterie (zde 91 %, jedná se o asi 6 let starý notebook).



Obrázek 3.19: Informace o baterii

3.4 Konfigurace sítě

Nástrojů pro síť bývá v prostředí víc, jsou rozstrkány na několika různých místech.

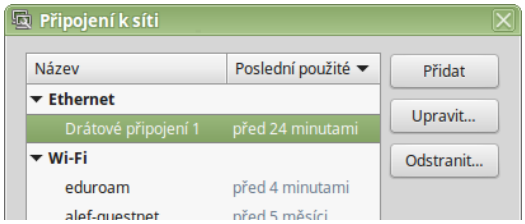


Obrázek 3.20: Nástroj Síť

V oznamovací oblasti je ikona , přes kterou se dostaneme k seznamu sítí pro připojování a odpojování, je tady i volba pro připojení ke skryté síti (musíme znát název sítě) a k VPN.

Velmi jednoduchý nástroj *Síť* (v záhlaví okna máme *Nastavení sítě*, viz obrázek 3.20) nacházející se v menu *Správa*. Určujeme v něm název počítače, případné zařazení do do-

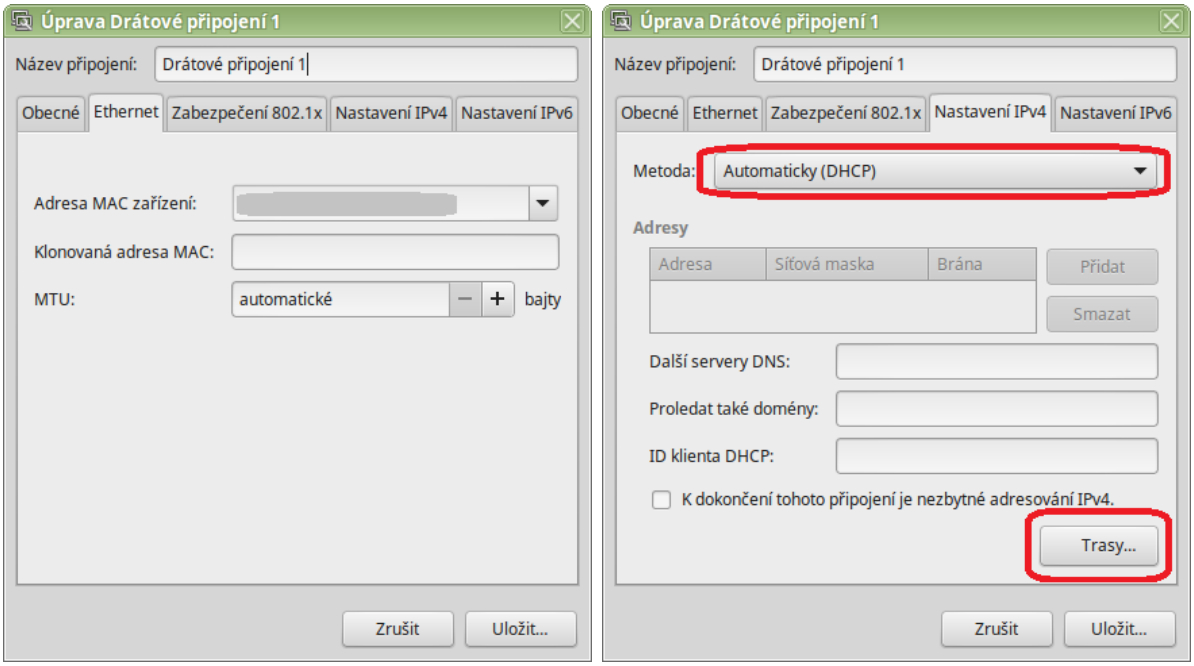
mény (ano, i počítač s Linuxem může být v doméně Active Directory), můžeme v něm zadat adresu DNS serveru, který bude překládat jmenné názvy (například `www.google.cz` na číselné IP adresy (například `216.58.209.195`).



Obrázek 3.21: Připojení k síti

V nástroji *Připojení k síti* (najdeme ho ve *Volbách*) předně dostaneme seznam známých sítí – viz obrázek 3.21. Jsou to opravdu známé sítě (zapamatované), nikoliv momentálně dostupné, takže pokud jsme například někdy v minulosti použili určitou Wi-fi síť, bude v tomto seznamu. Sítě, o kterých si myslíme, že už nebudeme potřebovat, odstraníme klepnutím na tlačítko *Odstranit*.

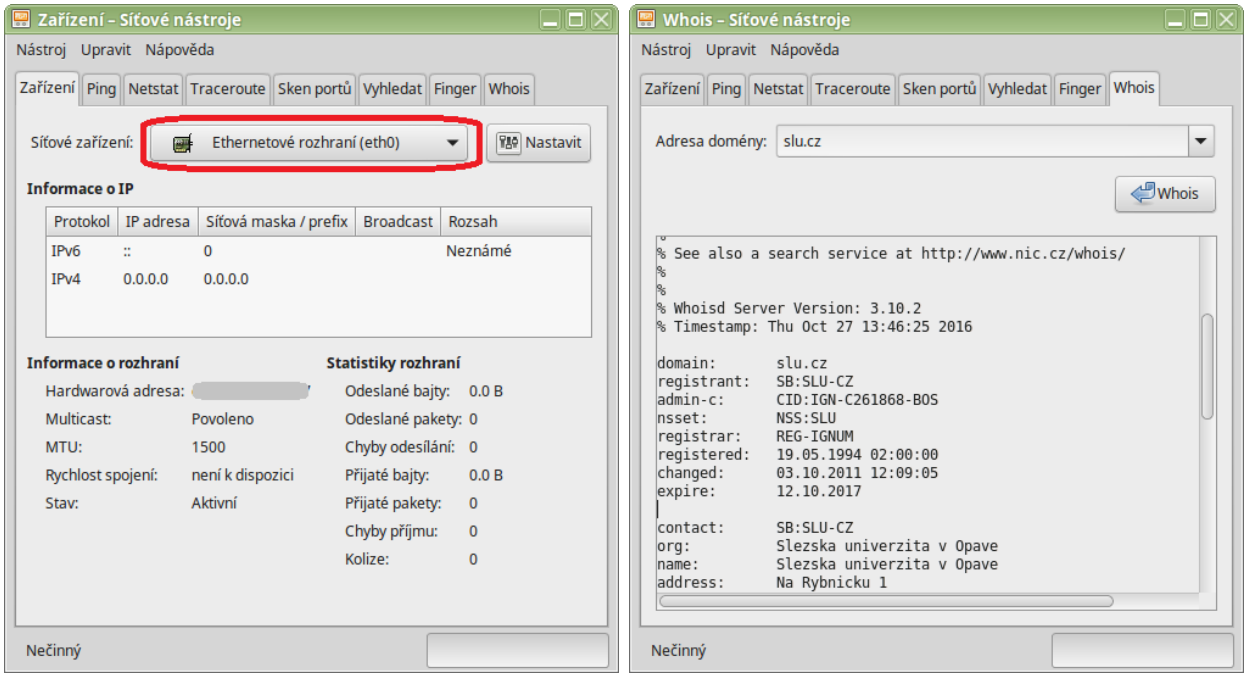
Po poklepání na konkrétní síť se dostaneme do okna s konfigurací této sítě, například u Wi-fi sítí tady je (a může se měnit) název sítě, typ zabezpečení, certifikáty, přístupové údaje a nastavení pro protokoly IPv4 a IPv6. Na obrázku 3.22 vidíme okno s vlastnostmi jedné ze sítí (první – Ethernet). U sítě přes kabel tu najdeme záložku *Ethernet*, na které je zobrazena MAC adresa (fyzická adresa) zařízení, u Wi-fi sítě bychom měli záložku *Wi-fi* a následovala by záložka *Zabezpečení Wi-fi*.



Obrázek 3.22: Konfigurace vlastností vybrané sítě (zde Ethernet)

Na záložce *Nastavení IPv4* se určuje, zda má být IP adresa získána dynamicky od DHCP serveru nebo má být zadána ručně; v tom případě bychom měli od správce přidělenu vlastní IP adresu, kterou bychom zadali sem, místo „Automaticky (DHCP)“ bychom ve výběrovém poli zvolili „Ručně“ a vložili bychom příslušné údaje. Stisknutím tlačítka *Trasy* se dostaneme do okna, kde se zadává adresa brány a maska sítě. Záložka pro nastavení IPv6 je podobná.

Užitečnou aplikací jsou *Síťové nástroje* (pokud je nenajdeme, dá se doinstalovat balíček). Na obrázku 3.23 vidíme okno aplikace s několika záložkami, kde na první (na obrázku vlevo) najdeme souhrnné informace o zařízení síťového rozhraní (zde konkrétně je zvoleno rozhraní Ethernet, ale momentálně není připojen kabel, proto nevidíme žádný provoz ani adresy).

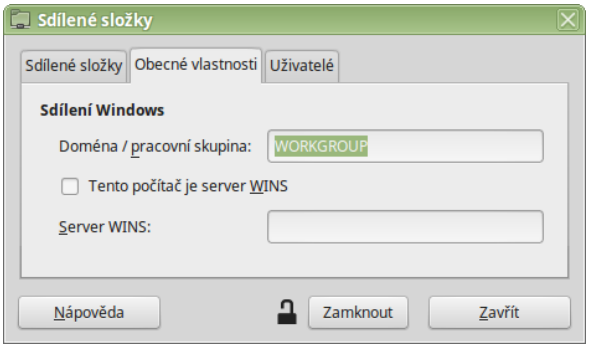


Obrázek 3.23: Síťové nástroje

Na dalších záložkách máme v grafickém prostředí přístup k tomu, co bychom jinak prováděli v textovém rozhraní (například to, co umí příkaz `ping`, je v „myšové podobě“ na druhé záložce). Na obrázku 3.23 vpravo je obsah záložky *Whois* (pro tu taky existuje příkaz), kde můžeme zadat název domény a získat o ní informace z Whois databáze. Zde konkrétně „proklepáváme“ doménu `slu.cz`.

Už dřív bylo naznačeno, že zařízení s Linuxem může bez problémů existovat v síti se zařízeními s Windows, a dokonce se zapojovat do sdílení. V nástroji *Sdílené složky* určujeme název pracovní skupiny nebo domény v síti ve stejném významu jak totéž provádíme ve Windows.

Na první záložce určujeme složky (adresáře) na svém zařízení, které mají být viditelné v místní síti.



Obrázek 3.24: Nastavení sdílení složek v síti

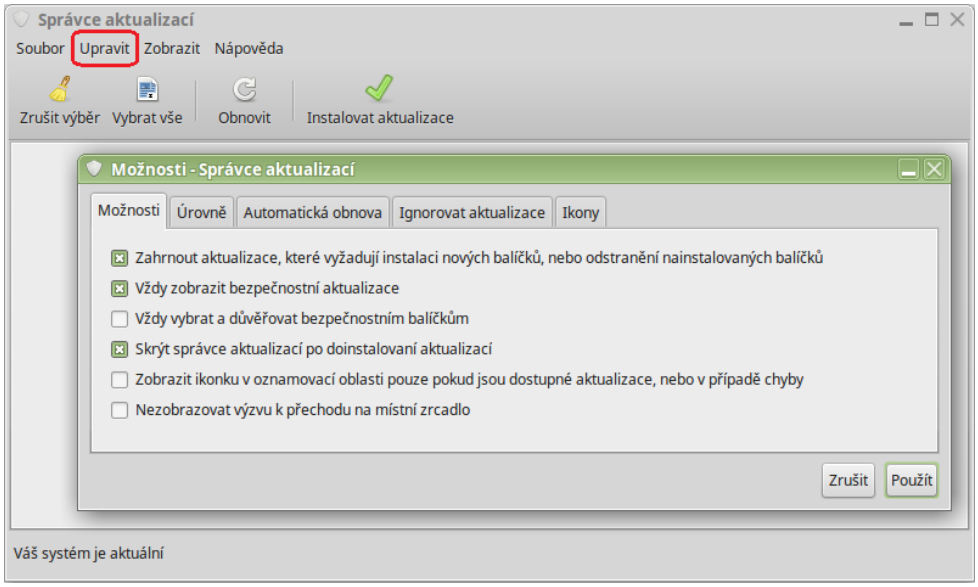
3.5 Správa softwaru

Už v předchozí kapitole (od strany 15) jsme se dozvěděli, jakým způsobem funguje správa softwaru, tedy jen stručně zopakujeme – software je distribuován v balíčcích přes repozitáře, máme k dispozici minimálně jednu aplikaci, přes kterou můžeme do repozitářů nahlížet a instalovat si odtamtud balíčky či aplikace (podle toho, na které úrovni pracujeme). Taky vždy existuje nástroj, ve kterém určujeme repozitáře, jejichž obsah chceme míst ve zmíněné aplikaci dostupný.

Aktualizace jsou řešeny podobným způsobem, taky se zjišťují přes repozitáře, a obvykle máme k dispozici nástroj, ve kterém lze aktualizace nechat automaticky zjistit (či se nechat upozornit), instalovat klepnutím na tlačítko a také konfigurovat postup aktualizací.

Na na obrázku 3.25 je rozhraní *Správce aktualizací* v Linux Mint, přičemž je systém zrovna aktuální

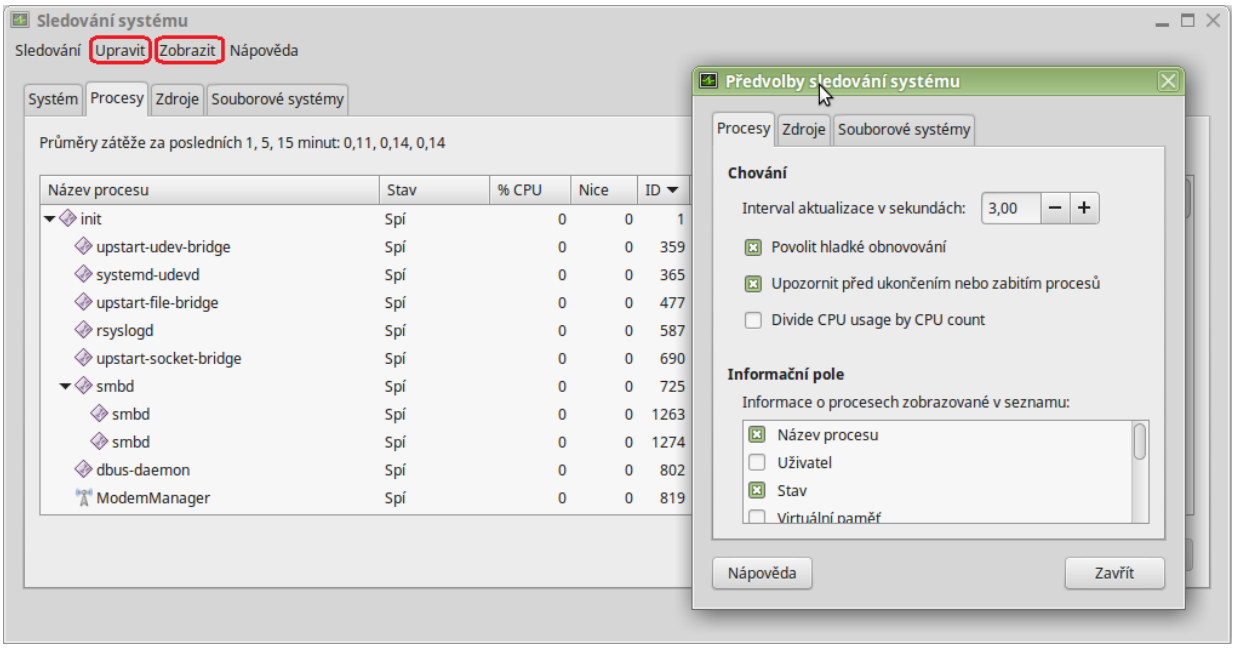
(nejsou zobrazeny žádné aktualizace k instalaci). Přes položku *Upravit* v hlavním menu se dostaneme k nástroji pro určení repozitářů (Zdroje softwaru) a přes vnořenou položku *Možnosti* k nastavení aktualizačního mechanismu.



Obrázek 3.25: Konfigurace aktualizací

3.6 Správa procesů, zdrojů a souborů

V každé linuxové distribuci najdeme nástroj pro práci s procesy, v Mint to je *Sledování systému*. V tomto nástroji najdeme několik záložek, z nichž na první je informace o systému (o kterou jde distribuci včetně verze – je to jedno z míst, kde můžeme zjistit tuto informaci, verze jádra, dále použité desktopové prostředí s jeho verzí, množství operační paměti, procesor).



Obrázek 3.26: Nástroj *Sledování systému*, karta s procesy

Na druhé záložce je seznam procesů (viz obrázek 3.26). Pokud chceme určit, co vše se má o procesech zobrazit, v menu najdeme *Upravit* ⇨ *Předvolby*, tam na kartě *Procesy* můžeme například určit, které sloupce budou zobrazeny. K tomuto místu se vrátíme později, až budeme brát správu procesů v UNIXových systémech.

Obrázek 3.27: Nástroj *Sledování systému*, karta se zdroji

Na třetí záložce (obrázek 3.27) je několik grafů, ze kterých můžeme usoudit na momentální vytížení procesoru, operační paměti a sítě.

Na čtvrté záložce najdeme informace o připojených diskových oddílech nebo výměnných médiích, především o volném místě na nich, a poklepáním na konkrétní záznam se otevře správce souborů a v něm právě tento oddíl či paměťové médium. V menu *Upravit* ⇨ *Předvolby* se dá konfigurovat také obsah této záložky, můžeme například určit, že se zde mají zobrazovat úplně všechny souborové systémy včetně virtuálních (v UNIXových systémech se souborové systémy používají k více účelům než jen pro přístup k paměťovým médiím, obecně jako rozhraní k jakémukoliv „skladu“ dat či datové filtry).

Poznámka:

Pokud máme na hlavním panelu applet Sledování systému (zobrazuje většinou v ikoně vytížení procesoru), pak výše popsaný nástroj získáme poklepáním na ikonu tohoto appletu.

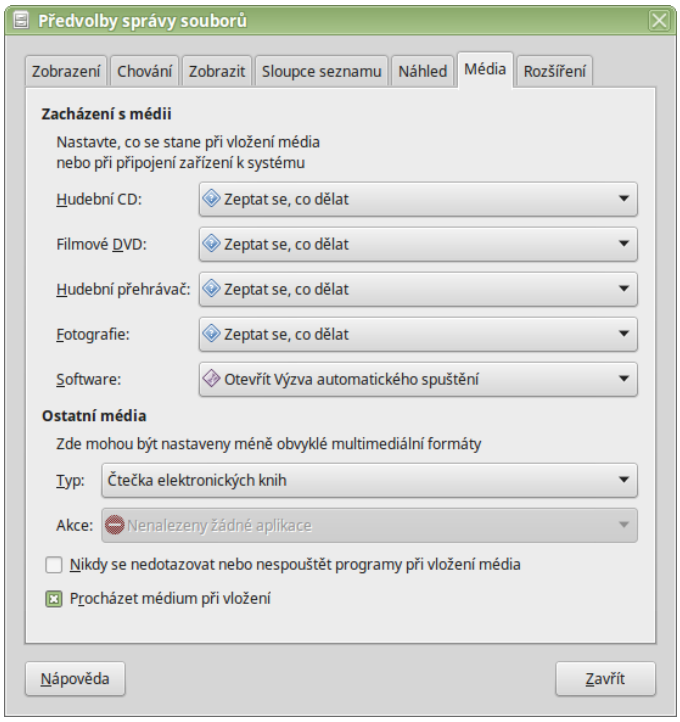
Správou souborů se zatím nemusíme zabývat do hloubky, taktéž se s touto problematikou setkáme v dalších kapitolách. Zde jen upozorníme na aplikaci *Správce souborů*, kterou spustíme například pomocí appletu („kaslík“) na hlavním panelu, nebo poklepáním na některou z ikon na ploše, nebo třeba v systémovém menu přes některou položku v *Místech*.

Vpravo nahoře je rozbalovací pole, ve kterém volíme způsob zobrazení souborů (zde je „Zobrazení v seznamu“, které umožňuje složky rozbalovat a tedy vidíme „rozšířený strom“ včetně údajů o souborech), přes položku menu *Zobrazit* můžeme například zapnout zobrazování skrytých souborů nebo

určovat sloupce k zobrazení (položka Viditelné sloupce, okno s výběrem sloupců je vidět na obrázku 3.28.

Vzhled Správce souborů můžeme konfigurovat jako u kterékoliv jiné aplikace – v menu *Upravit* je položka *Předvolby* (určitě jste postřehli, že tak je to v mnoha nástrojích prostředí odvozených z GNOME, je to konvence pro většinu konfigurovatelných aplikací). Určujeme zde například, jaký typ zobrazení má být výchozí, jestli mají být soubory a složky aktivovány (třeba otevřeny) klepnutím nebo poklepáním, zda se mají ukazovat náhledy souborů určitých typů a velikostí, a taky které sloupce se při zobrazení v seznamu objeví. Na záložce *Média* (viz obrázek 3.29) určujeme akce, které se mají provést při připojení výměnných médií.

Nástroj Správce souborů může pracovat i v dvoupanelovém režimu (jako například Total Commander ve Windows), stačí v menu *Zobrazit* zvolit položku *Další panel*.

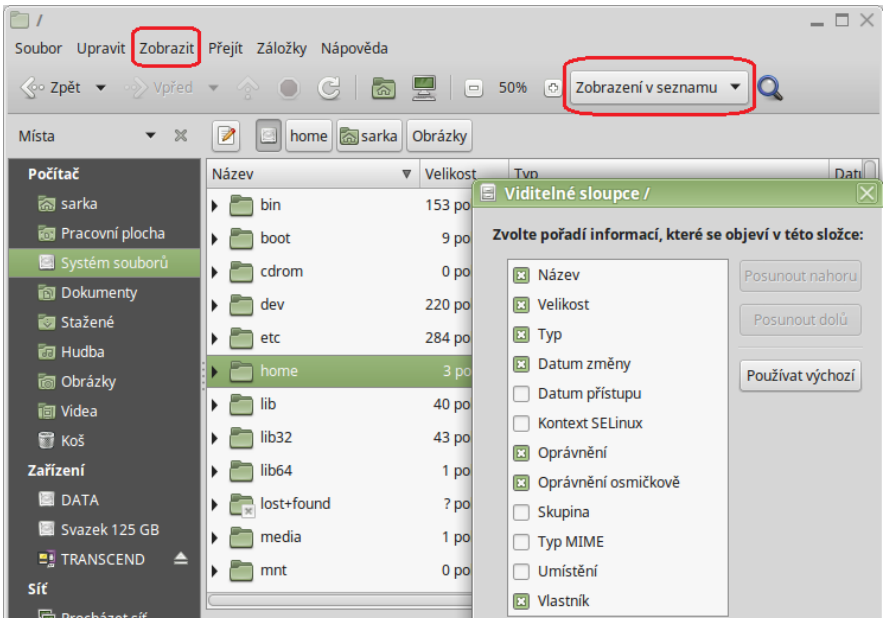


Obrázek 3.29: Předvolby pro *Správce souborů*

3.7 Nastavení související s uživateli

3.7.1 Osobní nastavení

Každý uživatel by si především měl umět změnit heslo. K tomu v Linux Mint najdeme dva nástroje (a navíc to lze provést v textovém režimu). Jednodušší nástroj, ke kterému má každý plný přístup (co

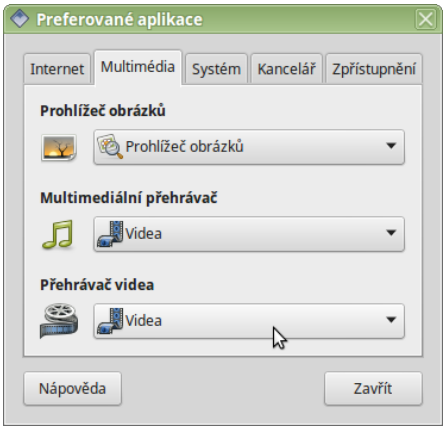


Obrázek 3.28: Nástroj *Správce souborů*, nastavení viditelných sloupců

se týče vlastních uživatelských informací), je *O mně* – můžeme si zvolit reprezentační obrázek nebo změnit heslo, nic moc dalšího neumí. Jiná možnost je použít nástroj pro správu uživatelů, tam si najít sebe jako uživatele a na daném místě taky najdeme volbu na změnu hesla. O tomto nástroji později.

Uživatel čas od času otevírá soubory a taky chce ovlivnit, v čem jsou tyto soubory otevírány. Také v UNIXových systémech (v jejich prostředích) se pracuje s asociacemi souborů. Asi nejjednodušší cesta je zobrazit si kontextové menu souboru s danou příponou a zvolit *Vlastnosti*, na kartě *Otevřít s* vybereme aplikaci nebo do seznamu aplikací přidáme novou přes tlačítko *Přidat*.

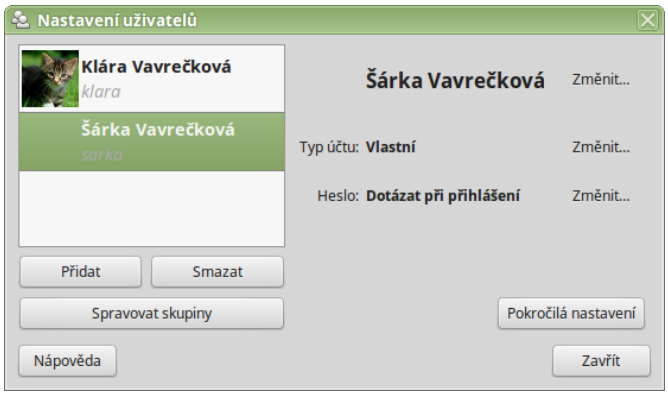
Nástroj k souhrnnější práci s asociacemi je *Preferované aplikace*. Zde jednoduše pro danou kategorii souborů zvolíme aplikaci výběrem ze seznamu. Na obrázku 3.30 je zobrazena karta pro multimediální soubory.



Obrázek 3.30: Preferované aplikace

3.7.2 Správa uživatelů a skupin

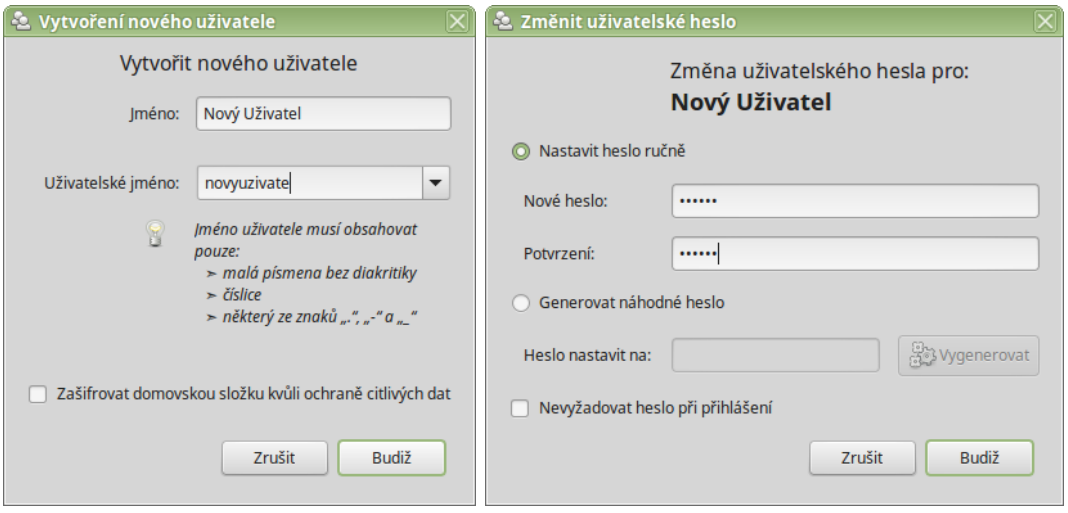
Pro práci s uživateli a skupinami je nástroj *Uživatelé a skupiny*, viz obrázek 3.31.



Obrázek 3.31: Nástroj pro správu uživatelů a skupin

Myší vybereme konkrétního uživatele a pak vpravo nastavujeme základní vlastnosti: jméno, typ účtu (správcovský, běžný nebo vlastní nastavení v pokročilých možnostech), a taky se zde dá nastavit či změnit heslo.

Nového uživatele vytvoříme pomocí průvodce z tlačítka *Přidat*, na obrázku 3.32 vidíme první dva kroky průvodce – zadáváme „občanské“ jméno, uživatelské jméno a případně heslo. Nové heslo můžeme vygenerovat, pak musíme najít bezpečnou cestu pro sdělení hesla novému uživateli.

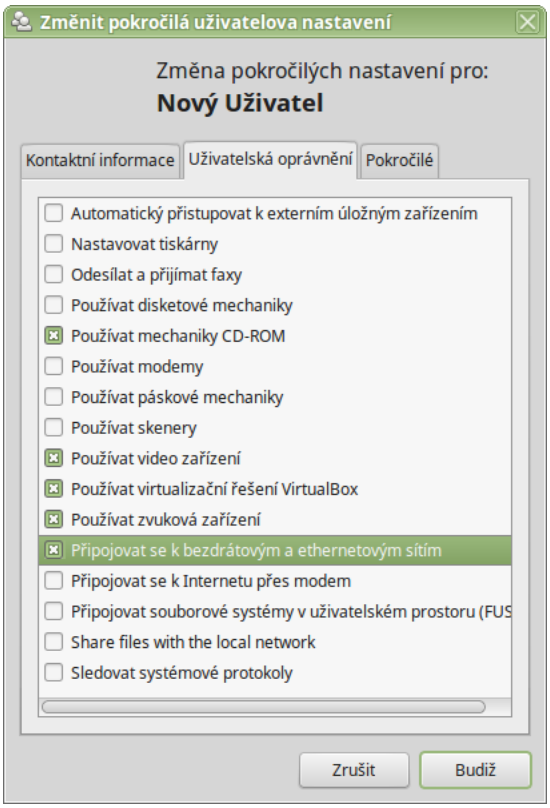


Obrázek 3.32: První dva kroky vytvoření nového uživatele

Pokud pro účet není zvolen typ správcovský ani běžný (je to tedy typ „vlastní“), můžeme oprávnění nastavovat přes tlačítko *Pokročilá nastavení* vpravo dole v okně pro správu uživatelů. Otevře se okno se třemi záložkami, z nichž právě na té druhé máme nastavení oprávnění (viz obrázek 3.33).

Jak vidíme, je tu toho celkem hodně, například můžeme povolit nebo zakázat používání optických mechanik, používat skener, zobrazovací a zvuková zařízení, ale taky připojení do sítě, sdílet soubory v lokální síti či nahlížet do systémových logů.

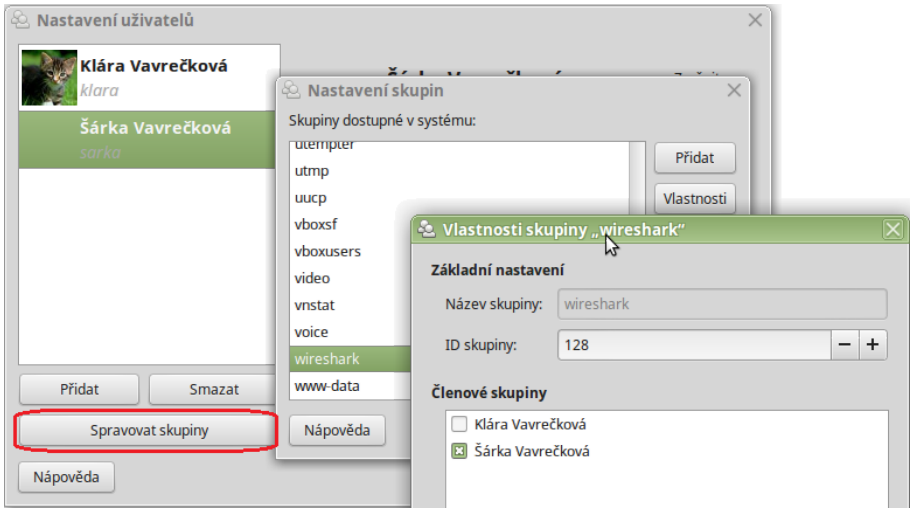
Pokud jsme při vytváření nového uživatele zvolili typ účtu vlastní, je součástí průvodce pro vytvoření také toto okno, tedy můžeme rovnou určit konkrétní oprávnění.



Obrázek 3.33: Pokročilé nastavení oprávnění uživatele

Skupiny jsou v UNIXových systémech používány ve větší míře než ve Windows. Existuje hodně skupin, obvykle se právě přes skupiny řeší přístup k určitému objektu či nástroji, zjednodušuje se tím přiřazování přístupových oprávnění (například existuje skupina „audio“, jejíž členové mohou používat zvuková zařízení, skupina „scanner“ obsahující uživatele, kterým je povoleno skenovat, nebo skupina „vboxusers“, kterou si vytvořila virtualizační aplikace VirtualBox).

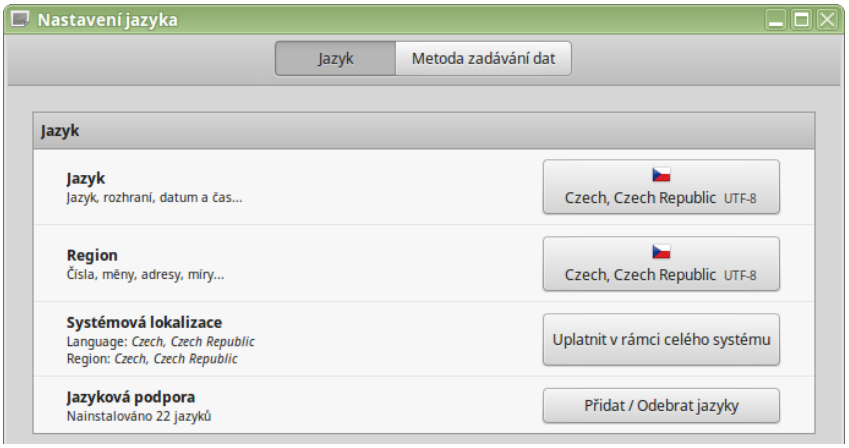
V nástroji Uživatelé a skupiny je tlačítko *Spravovat skupiny* (vlevo dole). Po klepnutí na něj se zobrazí seznam existujících skupin, po poklepání na konkrétní skupinu se otevře okno s vlastnostmi skupiny – momentálně nás zajímá pouze vlastnost členství určitých uživatelů, takže jak vidíme na obrázku 3.34, můžeme klepnutím myši přidávat či odebírat členy.



Obrázek 3.34: Nástroj pro základní správu skupin

3.7.3 Přizpůsobení pro uživatele

Uživatel by rád komunikoval se systémem svým vlastním jazykem – ať už se jedná o texty v rozhraní v daném jazyce, nebo o takové „drobnosti“ jako je zápis data a času či měny v konkrétním jazyce. To vše nastavujeme v nástroji *Jazyky*, viz obrázek 3.35. Výsledek sice není úplně dokonalý (sem tam bývá „zapomenuté“ slovo či odstavec v angličtině), ale dostačuje.

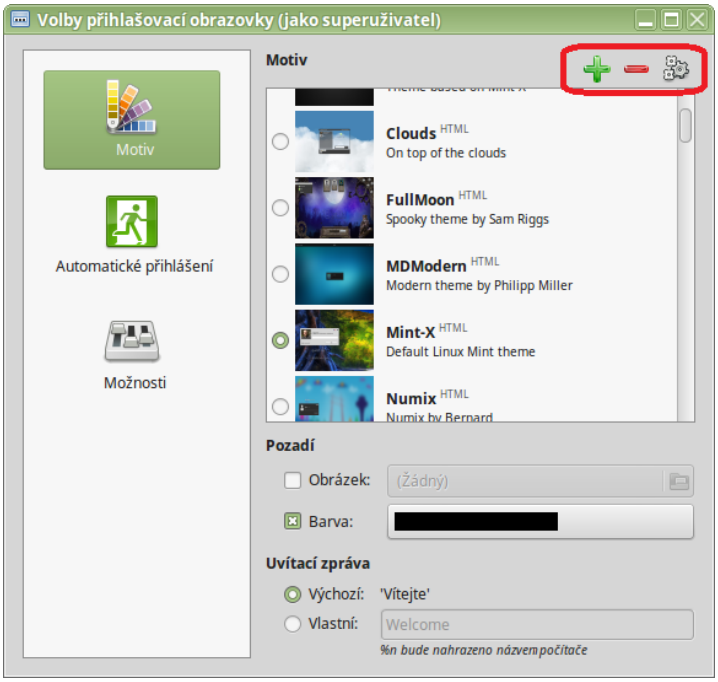


Obrázek 3.35: Nastavení jazyka pro rozhraní

Pokud se jedná o přepínání mezi jazyky rozložení klávesnice, pro to máme ikonu v oznamovací oblasti hlavního panelu (pokud ovšem máme víc než jedno rozložení klávesnice, o tom už bylo psáno v předchozím textu, viz str. 34).

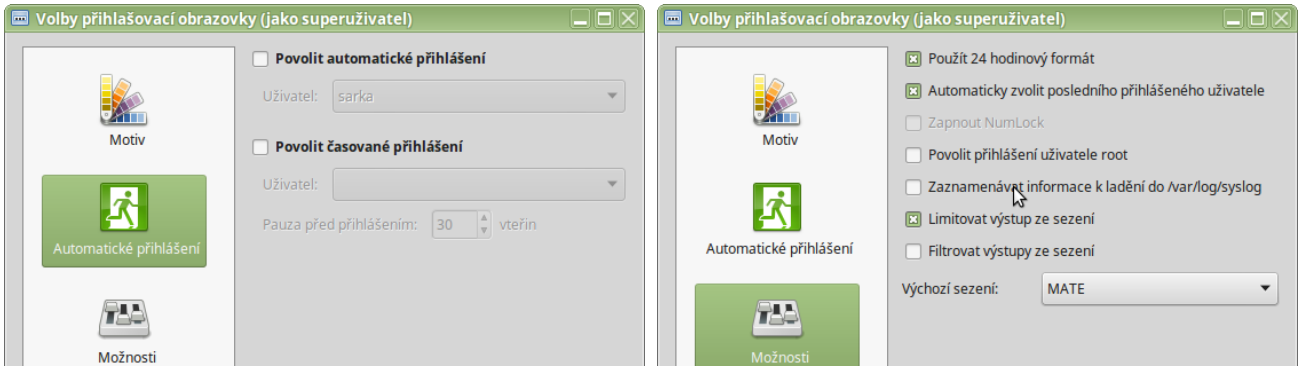
Uživatel svou cestu systémem začíná přihlášením. Nastavení vzhledu přihlašovací obrazovky a další parametry související s přihlášením provádíme v nástroji *Přihlašovací okno* (na titulkovém pruhu nástroje je *Volby přihlašovací obrazovky*).

Okno nástroje má několik „záložek“, z nichž na první nastavujeme motiv přihlašovací obrazovky (jak bude vypadat) – obrázek 3.36. Je z čeho volit, případně se dá přidat další motiv, také můžeme nastavit uvítací text (použije se, pokud motiv něco takového využívá), případně přidávat nebo ubírat motivy (dají se sehnat na internetu) či si vybraný motiv vyzkoušet (pro tyto tři činnosti máme tlačítka vpravo nahoře, na obrázku 3.36 jsou zvýrazněna).



Obrázek 3.36: Konfigurace přihlašovací obrazovky

Na obrázku 3.37 jsou další dvě „záložky“. Můžeme povolit automatické přihlašování (pro konkrétního uživatele), což by bylo vhodné snad jen v případě, když má k zařízení (fyzický) přístup jen jeden uživatel. V poslední části (*Možnosti*) například můžeme povolit přihlašování uživatele root (hlavního administrátora), což ale z bezpečnostního hlediska není dobrý nápad (měli bychom pracovat

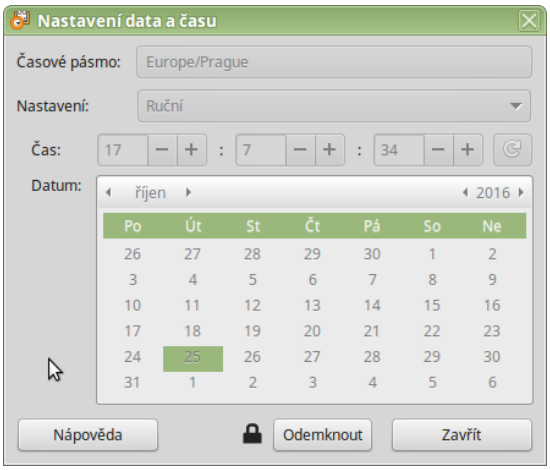


Obrázek 3.37: Další volby pro přihlašovací obrazovku

jako běžný uživatel, systém si v případě potřeby vyžádá administrátorské heslo, což jsme u mnohých nástrojů zjistili), nebo třeba zvolit desktopové výchozí prostředí (může jich být nainstalováno víc, při přihlášení se načítá jedno z nich).

3.8 Nastavení data a času

Dále je potřeba mít nastavený správný čas. K času se dostaneme pomocí appletu v oznamovací oblasti, po klepnutí na applet můžeme „listovat“ v kalendáři, ale možnost ovlivnění data a času tady přímo nenajdeme (ale dá se tam doklepat přes kontextové menu, *Předvolby*, tlačítko *Nastavení času*).



Obrázek 3.38: Konfigurace data a času

Nástroj na ovlivňování těchto nastavení najdeme v systémovém menu *Správa*, položka *Datum a čas*, taky zde určujeme časové pásmo (u nás je to Europe/Prague), obrázek 3.38.

Všimněte si, že na obrázku 3.38 jsou všechny prvky neaktivní, nepřístupné – ve skutečnosti jsme se s tím už setkali i v jiných nástrojích. To znamená, že pokud chceme změnit datum nebo čas, musíme nejdřív klepnout na tlačítko *Odemknout* a zadat heslo, pak teprve můžeme nastavovat. Správný čas je totiž důležitý i pro systém (například pro logování události nebo plánování spouštění procesů), tedy je potřeba jeho změny chránit.

Kapitola 4

Vlastnosti UNIXových systémů

Rychlý náhled: V této kapitole projdeme některé základní charakteristiky systémů UNIXového typu. Nebudeme se zabývat vnitřní strukturou UNIXu, ale spíše vlastnostmi týkajícími se uživatelů a správců systému. Snahou je nezávislost textu kapitoly na grafickém prostředí, ve kterém pracujete, proto jsou některé úkoly méně konkrétní.

Klíčová slova: Adresář, soubor, pevný odkaz, symbolický odkaz, spustitelný soubor, knihovna, bootování, zařízení, speciální soubor, přípojný bod, uživatel, skupina, UID, GID, proces, přístupové oprávnění, speciální příznak, SUID, SGID, Sticky, atribut, mód

Cíle studia: Po prostudování této kapitoly získáte přehled o nejdůležitějších adresářích a souborech, zejména konfiguračních souborech, v systému Linux. Dokážete rozlišit zařízení podle jejich speciálních souborů, a naučíte se principu evidence uživatelů, skupin a běžících procesů, seznámíte se s konceptem virtuálních souborových systémů. Naučíte se principu přístupových oprávnění používaných v UNIXových systémech, včetně speciálních příznaků.

4.1 Adresáře a soubory

4.1.1 Souborový systém

Pod pojmem *adresář* budeme rozumět kontejner na soubory či (pod)adresáře. V poslední době se pro tentýž účel používá pojem *složka*, který doputoval od uživatelů přecházejících z Windows.

Poznámka:

Nejdůležitějším faktem je: VŠECHNO JE SOUBOR (včetně adresářů, zařízení a čehokoliv dalšího) a ke všemu se jako k souboru můžeme chovat včetně uplatňování přístupových práv. Soubory jsou ukládány v stromové struktuře adresářů, způsob reprezentace souboru na paměťovém médiu je dán především použitým souborovým systémem.

Podobně jako jsme se ve Windows setkali se souborovými systémy FAT a NTFS, UNIXové systémy používají mnoho různých souborových systémů, například UFS (UNIX File System, původní, stále

vylepšovaný a používaný), VXFS, PCFS, HSFS, v Linuxu jsou obvyklé ext2, ext3, ext4, ReiserFS, JFS, XFS, ..., používají se také síťové souborové systémy (NFS, SMB, ...). Je jich tolik nejen proto, že od vzniku UNIXu už uběhlo hodně vody (a bylo dost času na vylepšování a „odbočky“ se specifickými vlastnostmi), ale také proto, že souborový systém volíme především podle jeho vlastností, a každý z nich má tyto vlastnosti trochu jiné (dnes se setkáme nejčastěji s ext4).

Soubory mohou mít *příponu*, ale není operačním systémem vyžadována (může být vyžadována nebo podporována některými aplikacemi). Na rozdíl od produktů Microsoftu není přípona speciální částí oddělenou od názvu souboru, ale je součástí názvu včetně oddělující tečky, což znamená, že část za tečkou může být jakkoliv dlouhá. Uživatelé příponu obvykle používají, protože jim umožňuje rozpoznat typ souboru a hodí se také pro asociace souborů s aplikacemi (to je obvyklé v grafických rozhraních). Spustitelné soubory a také některé skripty příponu nemají.

V UNIXových souborových systémech obvykle nejsou taková omezení pro názvy souborů jako například u Windows. Jediným omezením bývá zákaz použití znaku / v názvu souboru, protože tento znak odděluje adresáře v cestě k souboru. Přesto se dodržují určité konvence, jako například nepoužívání nezobrazitelných znaků (prvních 30 znaků v ASCII tabulkách) a dále některých znaků používaných shellem (*, ?, #, apod.).

Maximální povolená délka názvu souboru závisí na použitém souborovém systému, dnes souborové systémy v tomto směru také nekladou vážnější překážky, je to obvykle 255 znaků. Jak už bylo výše napsáno, do tohoto názvu patří i tečka a případná přípona souboru, tedy název souboru `abc.txt` má 7 znaků.

Některé soubory *začínají tečkou* (a obvykle je to jediná tečka v názvu) – tak se značí *skryté soubory*. Jsou to většinou konfigurační soubory systému, shellu nebo některé aplikace, obvykle jde o textové soubory nebo celé skryté adresáře, a při běžném zobrazování souborů je ve výpisech neuvidíme (musíme zapnout zobrazování skrytých souborů).

Ve výpisech můžeme také vidět názvy ve tvaru `nějaký_název.d` (tedy končí tečkou a písmenem d). Nejde o běžné soubory, ale o adresáře (directory).

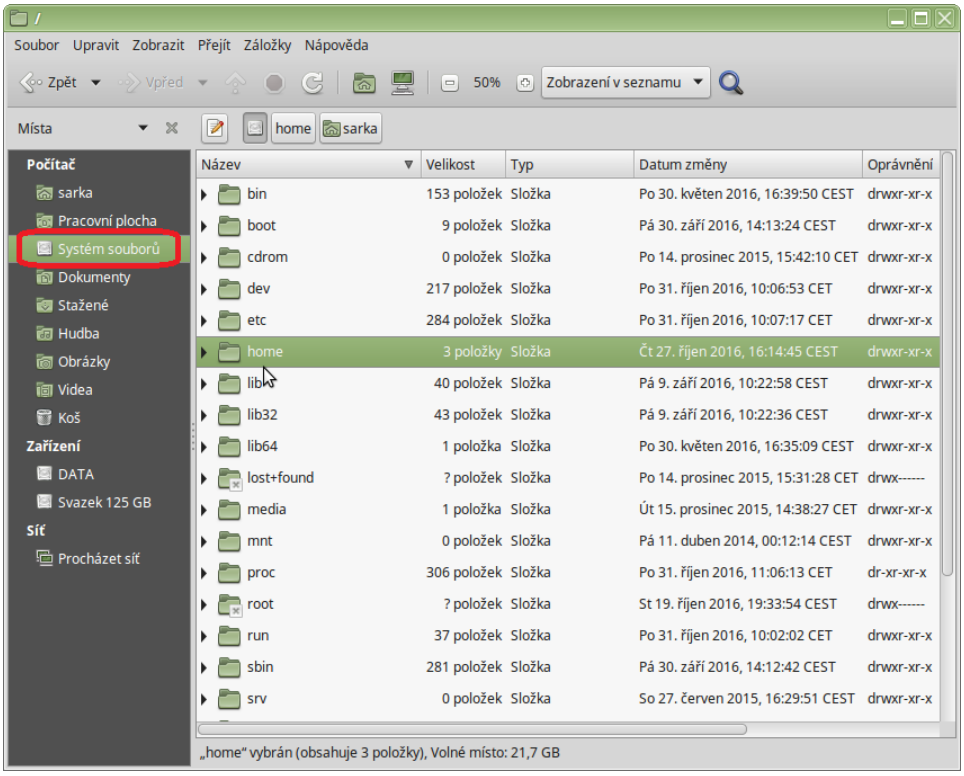
4.1.2 Začínáme zkoumat souborovou strukturu

Při procházení této kapitoly budeme paralelně procházet strukturu souborů a prohlížet soubory, které s konkrétními tématy souvisejí – prakticky veškerá konfigurace je totiž dostupná v souborech. Tedy bychom měli mít otevřeného některého správce souborů tak, abychom se dostali k celé adresářové struktuře.

Na obrázku 4.1 vidíme správce souborů v Linux Mint. V levé části okna jsou záložky, v nich je pro nás momentálně důležitá volba *Systém souborů*, přes kterou se dostaneme k celému stromu adresářů a souborů. V menu *Zobrazit* si můžeme zapnout zobrazování skrytých souborů. V titulkovém pruhu okna se vždy zobrazuje pracovní adresář (to je momentálně kořenový adresář, je tam jen lomítko).

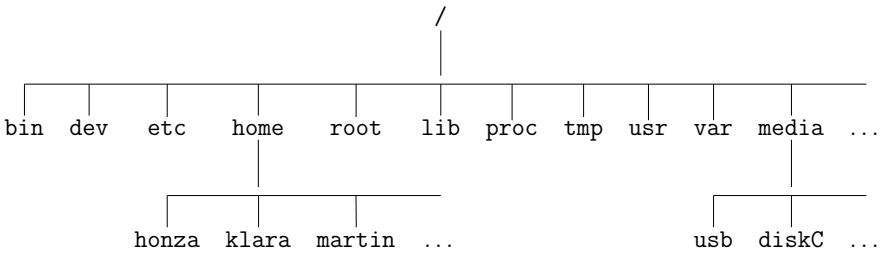
Úkol

Spusťte některého správce souborů. Projděte si možnosti jeho nastavení a zapněte zobrazování skrytých souborů.



Obrázek 4.1: Správce souborů, položka *Systém souborů*

Na rozdíl od struktury obvyklé pro Windows, v UNIXových systémech je adresářová struktura více virtualizována. Existuje jediný adresářový strom (s jedním kořenem), do kterého jsou také připojována paměťová média nebo různé oddíly disku (včetně těch zabraných Windows) a přistupuje se k nim přes předem určené adresáře stromu. Výhodou je možnost univerzálního přístupu k různým typům dat včetně dynamicky vytvářených, tento přístup znamená velké zjednodušení hlavně pro programátory.



Obrázek 4.2: Obvyklá struktura v UNIXových systémech

Některé adresáře mají v určitém okamžiku zvláštní význam, jsou to

- *kořenový adresář*, označuje se znakem / (lomítko, pozor – ne opačné lomítko),
- *aktuální adresář*, označuje se znakem . (použijeme v složitějších příkazech, kdy se chceme odkazovat na aktuální adresář),
- *nadřízený adresář*, označuje se .. (použijeme například pro pohyb v adresářové struktuře),
- *domovský adresář*, označuje se znakem ~ (platí vždy pro určitého uživatele).

Struktura adresářů se může na různých UNIXech trochu lišit. Obvykle přibližně odpovídá struktuře, kterou vidíme na obrázku 4.2.

4.1.3 Odkazy na soubor

V UNIXových systémech existují odkazy dvou typů: pevné a symbolické.

Pevný odkaz (hard link) je alternativní cesta k souboru, je rovnoprávný s „prvním odkazem“ vzniklým při vytvoření souboru, tedy název pevného odkazu na soubor je také název souboru. Pokud existuje na tentýž soubor více pevných odkazů, znamená to, že v souborovém systému existuje více cest k tomuto souboru (naprosto rovnoprávných). V souborových systémech pod Windows se pevné odkazy moc nepoužívají, ke každému souboru existuje jen jedna cesta (i když v souborovém systému NTFS ve vyšších verzích pevné odkazy lze vytvořit).

Pevný odkaz je výhodný zvláště tehdy, když k souboru, na který odkazujeme, přistupujeme z více míst, z nichž „původní“ má třeba hodně složitou adresu či název, případně chceme mít některý systémový soubor dostupný ze svého domovského adresáře.

Příklad

Předpokládejme, že na disku máme soubor s nejnovější fakturou na pojištění auta `/home/novak/dokumenty/faktury/zaplacene/pojistovna-auto/faktura30945876.pdf`, a v témže umístění evidujeme faktury pro tentýž účel (pojistka za auto) i z předchozích let. Zároveň chceme mít v přehlednějším místě aktuální faktury pro daný rok (třeba proto, že chceme vědět, co jsme v které části roku platili a kolik), tedy si vytvoříme pevný odkaz `/home/novak/faktury/faktura-auto.pdf`.

Mohli bychom prostě soubor zkopírovat, ale proč mít tentýž soubor na disku vícekrát, navíc bychom si museli pokaždé hrát s názvem. Když použijeme pevný odkaz, pak `.../faktura-auto.pdf` bude jen alternativním názvem (a alternativní cestou) k „momentálně platné“ faktuře, tedy jakýmsi ukazatelem (pointerem), a když budeme evidovat fakturu na další rok, jednoduše ji uložíme do „delšího umístění“ a v „přehledovém umístění“ jen přesměrujeme pevný odkaz.

Pevný odkaz může být pojmenován jinak než původní soubor, ale může mít stejný název. Samozřejmě pokud vytváříme pevný odkaz nacházející se ve stejném adresáři jako původní soubor, musíme zvolit jiný název.

Výhodou pevných odkazů je, že pokud smažeme soubor na původním místě (tedy první vytvořenou cestu), pak pevné odkazy (tedy alternativní cesty) budou dál fungovat. Soubor se reálně odstraní až tehdy, když na něj nebude vést žádný pevný odkaz. V souborovém systému se u každého souboru (včetně adresářů) eviduje v čítači počet pevných odkazů vedoucích na tento soubor, a v okamžiku, kdy tento čítač klesne na nulu, je soubor odstraněn.

Nevýhodou je, že pevný odkaz nemůže vést na cokoliv:

- pevné odkazy se vytvářejí pouze v rámci téhož souborového systému (takže pevný odkaz nemůže vést na jiný oddíl disku ani na jiný disk), protože evidence čítačů pevných odkazů platí jen pro (uzavřený) souborový systém,
- pevné odkazy můžeme vytvářet jen na běžné soubory, ne na adresáře.

Poznámka:

Z čeho plyne druhá nevýhoda? Uvědomte si, že když se prohledává adresářová struktura (například když spustíme hledání souboru s určitým názvem), prochází se graf adresářů po jednotlivých cestách ve

struktuře. Kdybychom povolili vytváření pevných odkazů (a tedy alternativních rovnoprávných cest) na adresáře, mohl by ve struktuře vzniknout cyklus, a tedy by se nám prohledávací algoritmus mohl zacyklit a tento cyklus by nedokázal zjistit.

Výjimkou z tohoto pravidla jsou odkazy `.` a `..` na sebe sama a nadřízený adresář ve struktuře, které jsou implementovány právě jako pevné odkazy. K zacyklení zde nedochází, protože mají „předvídatelné“ názvy – při prohledávání se adresáře takto pojmenované ignorují.

Poznámka:

Důsledkem je, že čítač pevných odkazů má u adresářů vždy hodnotu nejméně 2 (protože kromě první vytvořené cesty k adresáři se započítává pevný odkaz pojmenovaný jednou tečkou, což je odkaz na sebe sama). Dále se čítač zvýší vždy, když v daném adresáři vytvoříme nový podadresář, protože z toho podadresáře povede do daného adresáře pevný odkaz `..` (odkaz na nadřízený adresář).

Pokud například je čítač u adresáře na hodnotě 9, znamená to, že má pravděpodobně 7 podadresářů (jedna „hlavní“ cesta, jeden odkaz na sebe sama a 7 odkazů `..` vedoucích z podadresářů).

Symbolický odkaz (soft link) ukazuje pouze na jméno souboru, je to obdoba zástupců ve Windows. Obsahuje tedy v sobě informaci o názvu souboru s absolutní nebo relativní cestou.

Výhodou je, že na rozdíl od pevných odkazů symbolický odkaz může odkazovat na cokoliv včetně adresáře, cíl může být i na jiném oddílu disku či jiném paměťovém médiu. *Nevýhoda* je, že když je původní soubor odstraněn, symbolický odkaz ukazuje na neexistující soubor.

Jak je poznat: ve správci souborů, kterého používáme, zajistíme zobrazení sloupců s přístupovými oprávněními a počtem odkazů na soubor (při typu zobrazení, kdy máme nejen názvy souborů, ale i jejich vlastnosti). Bohužel v jednodušších správcích není možnost zobrazit čítač pevných odkazů, pak nezbývá než přejít do textového režimu.

Symbolické odkazy poznáme podle pohledu na řetězec přístupových oprávnění – je to celkem 10 písmen a může vypadat například takto: `drwxr-xr-x`. Pokud se jedná o symbolický odkaz, pak tento řetězec začíná písmenem `l` (link), například `lrw-r--r--`. Navíc se na řádku pro daný soubor objevuje i „cíl“ symbolického odkazu.

Pevné odkazy se nám „zhmotňují“ pouze v čísle znamenajícím počet pevných odkazů na soubor. Pokud nám správce souborů odmítá zobrazit sloupec s touto informací, můžeme si ji zjistit v textovém režimu (procvičíme si později, je to příkaz `ls -la`).

Úkoly

1. Kolik pevných odkazů (cest) vede k podadresářům `.` a `..` ve vašem domovském adresáři? Kolik je jich u stejně pojmenovaných podadresářů v adresáři `/home`?
2. Zjistěte, zda ve vašem domovském adresáři je nějaký symbolický odkaz. Zkuste tento soubor otevřít v textovém editoru (pomocí kontextového menu).

4.2 Spustitelné soubory, knihovny a související

Nejdřív si řekněme, jak je to s příponami:

- binární spustitelné soubory obvykle nemají žádnou příponu,
- textové spustitelné soubory (tedy skripty) mají většinou příponu .sh (zkratka z „shell“) nebo podle toho, v jakém programovacím jazyce jsou psány (jinou příponu budou mít skripty psané v Pythonu, Perlu a dalších skriptovacích jazycích),
- knihovny (obdoba DLL knihoven z Windows) mají většinou příponu .so, přičemž před nebo za příponou může být číslo verze knihovny, například `libcrypt-2.19.so` nebo `libcrypt.so.1`.

Programy a aplikace, které jsou nainstalovány, najdeme na těchto místech:

`/usr` – statické (málo se měnící) části instalace aplikací, například spustitelné soubory, knihovny, konfigurační soubory, případně zdrojové soubory, dokumentace, nápověda,

`/var` – často se měnící části instalace aplikací (variable), například logy, dočasné soubory, tiskové fronty apod.,

`/opt` – původně se do `/usr` a `/var` dávaly aplikace instalované zároveň s instalací celého systému a `/opt` (optional) byl určen pro aplikace instalované později některým uživatelem (tj. zvolené k instalaci některým uživatelem), ale v současné době se ve většině distribucí nepoužívá.

`/etc` – konfigurační soubory pro systém i mnohé aplikace (aplikace si zde pro vlastní účely vytvářejí podadresáře).

Takže většina instalace aplikací (včetně systémových) je v adresářích `/usr` a `/var`, a podle potřeby konfigurační soubory v `/etc`. Některé soubory z adresáře `/etc` si projdeme dále v této kapitole.

Spustitelné soubory a knihovny, a taky s nimi související soubory, najdeme na těchto místech:

`/bin` – *základní* příkazy pro textový režim (jejich spustitelné soubory) pro různé účely (název `bin` je od „binární“), mohou být používány i krátce po startu systému, tyto příkazy může obvykle spouštět i běžný uživatel, je tu například `cp` pro kopírování (ve Windows máme `copy`) nebo `ls` pro výpis obsahu adresáře (ve Windows máme `dir`),

`/sbin` – taky *základní* příkazy pro textový režim, ale obvykle vyžadují určitá přístupová oprávnění (písmeno „s“ znamená „superuser“, tedy hlavní uživatel – root),

`/lib` – sdílené knihovny využívané při startu systému nebo pro programy z `/bin` a `/sbin`, v podadresáři `modules` najdeme moduly jádra,

`/usr` – zde jsou nainstalovány programy a dokumentace využívané až později po startu systému, některé podadresáře:

- `/usr/bin` – programy v podobném významu jako u `/bin`, ale obvykle mají souvislost s nastaveními pro konkrétního uživatele (například `/usr/bin/passwd` pro změnu hesla nebo `/usr/bin/whoami` pro zjištění, pod kterým účtem momentálně pracujeme), také interpretační programy pro mnohé skriptovací jazyky (Perl, Python, Tcl, atd.),
- `/usr/lib` – systémové knihovny,
- `/usr/local` – místo pro instalaci aplikací,
- `/usr/src` – zdrojové soubory.

4.3 Bootování systému

Než se při startu počítače začne načítat operační systém, máme obvykle možnost si zvolit, zda se tedy bude dotyčný systém načítat (a který, pokud jich máme nainstalováno více), nebo jestli se provede něco jiného (před bootováním systému můžeme spustit některý diagnostický program, často bývá k dispozici například MemTest86+ pro otestování operační paměti). Tuto nabídku nám zprostředkovává program, kterému se říká *Boot Manažer*. Program, který zajistí načtení operačního systému do paměti a jeho plné spuštění, se nazývá *Boot Loader*.

V linuxových distribucích se již téměř výhradně používá pro obě tyto role (boot manažer a boot loader) program *Grub*.

V adresáři `/boot` najdeme vše, co souvisí s bootováním systému. Je tam adresář se soubory, které jsou používány boot manažerem/boot loaderem (v našem případě adresář `/boot/grub`) a také spouštěcí soubory pro jednotlivé nástroje, které lze z bootovacího menu spustit (například spustitelný soubor MemTestu).

Dále je v adresáři `/boot` ještě (minimálně) jedna důležitá položka – soubor, z něhož se načítá jádro Linuxu, většinou se ten soubor nazývá `/boot/vmlinuz-čísloverze` (tj. řetězec `vmlinuz` s připojeným číslem verze jádra). Na tomto místě můžeme mít i víc než jen jedno jádro, pak se při spouštění Linuxu použije jeden z těchto souborů.

4.4 Zařízení

4.4.1 Zajištění přístupu k zařízením

Ovladače. Také v Linuxu jsou pro každé zařízení nutné ovladače. Pro běžně prodávaná zařízení se dají většinou sehnat ovladače pro UNIX (Linux) nebo jsou přímo v jádře (tam jsou většinou standardní ovladače, které umožní zařízení používat).

Ovladače pro Linux, pokud vestavěné standardní nevyhovují, se často dají sehnat na internetu na stránkách výrobce zařízení nebo na specializovaných stránkách. Problémy nastávají jen u těch zařízení, jejichž výrobci sami nemíní vydat linuxové ovladače a také nechtějí zveřejnit informace nutné pro jejich naprogramování. Naštěstí se to obvykle dá vyřešit použitím obecného ovladače pro daný typ zařízení, případně naprogramováním ovladače „ad-hoc“ s použitím informací získaných pozorováním chování ovladače pro jiný operační systém.

Aby bylo možné plně využívat 3D grafické rozhraní v UNIXových systémech, je třeba mít grafickou kartu s podporou OpenGL a případně dalších podobných technologií a také k ní vhodné ovladače. Základní akcelerace obvykle funguje i s obecnými ovladači, ale můžeme si doinstalovat proprietární ovladače od výrobce grafického čipu na kartě (u těch kvalitnějších obvykle AMD nebo nVidia), což některé distribuce dělají automaticky při instalaci systému.

Speciální soubory jsou jakési komunikační body, přes které procesy získávají data ze zařízení nebo je tam naopak posílají, jsou to tedy kanály pro nízkoúrovňovou komunikaci. Najdeme je v adresáři `/dev`. Základní operace jsou realizovány jako běžná práce se souborem (podle filozofie „vše je soubor – i zařízení“):

- inicializace zařízení (pokud je nutná) = otevření souboru,
- zaslání dat zařízení (disku, tiskárně apod.) = zápis dat do souboru,
- načtení dat ze zařízení (disku, ...) = čtení dat ze souboru, atd.

Pozor, speciální soubor zařízení není totéž co ovladač (ovladač je modul jádra, speciální soubor zařízení je datová struktura sloužící ke komunikaci uživatele nebo jiného procesu s tímto zařízením).

Připojování paměťových zařízení. Pro zařízení typu paměťové médium nestačí mít pouze speciální soubor. Komunikace s takovým zařízením je mnohem složitější, například když ukládáme soubor, nestačí pouze poslat prakticky nezformátovaná data speciálnímu souboru, ale musíme přidat další informace – název souboru, jeho umístění v adresářové struktuře, přístupová oprávnění, apod. Proto pro paměťová média máme kromě speciálního souboru také jiné kontaktní místo, *bod připojení* (přípojný bod).

Každé paměťové zařízení je nutné před jeho používáním připojit, a po ukončení práce s ním synchronizovat (pokud se používá mezipaměť – cache) a odpojit. Tyto operace bývají obvykle automatizovány, jen odpojení USB flash disku, které je poněkud „náhlé“ a pro operační systém nepředvídatelné, je vhodné provést ručně (to znamená například z příslušného kontextového menu vybrat volbu pro odpojení média). Tyto postupy ostatně známe i z Windows.

Zařízení, která je nutné připojit (a také oddíly zabrané jiným operačním systémem), se připojují do adresáře `/mnt` nebo `/media` nebo jinam (záleží na distribuci), v tomto adresáři pak k zařízením můžeme přistupovat (např. pokud chceme na disk uložit určitý soubor, zkopírujeme ho do podadresáře v tomto adresáři představujícího dotyčný disk).

Obrázek 4.3: Přípojný bod pro USB flash disk

Na obrázku 4.3 je obsah adresáře `/media` ve chvíli, kdy je k počítači připojen USB flash disk. V této distribuci se v adresáři `/media` nacházejí podadresáře pro každého uživatele zvlášť, přípojný bod je v podadresáři toho uživatele, který byl při připojování média přihlášen.

Poznámka:

Takže pro zařízení typu klávesnice, monitor, myš apod. máme

- ovladač (aby fungovalo),
- speciální soubor (pro nízkoúrovňový přístup).

Kdežto pro zařízení, která jsou paměťovými médii (oddíl na disku USB flash disk, optické médium), máme

- ovladač (aby fungovalo),
- speciální soubor (pro nízkoúrovňový přístup).
- přípojný bod (pro přístup využívaný běžnými uživateli).

4.4.2 Speciální soubory

Mnohá přístupová rozhraní, včetně rozhraní pro speciální soubory, jsou v Linuxu koncipována jako virtuální souborové systémy. Tím si zatím nemusíme lámat hlavu, stačí nám jen vědět, který souborový systém se pro tyto účely používá.

Ve starších verzích jsme se mohli setkat se souborovým systémem *devfs*, v novějších verzích se setkáme už téměř výhradně se systémem *udev*, který je především na rozdíl od devfs dynamicky koncipovaný. Z hlediska uživatele je rozdíl především v názvech speciálních souborů pro pevné disky.

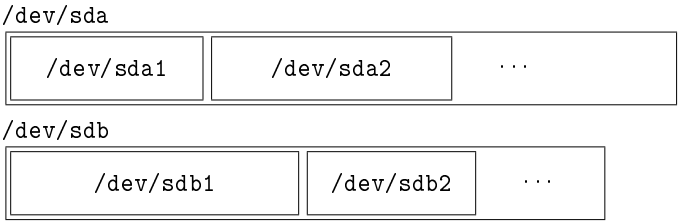
Pevný disk: speciální soubor je `/dev/sda` (první pevný disk), `/dev/sdb` (druhý pevný disk), ..., a protože na discích bývají vytvořeny oddíly, každý oddíl má také vlastní speciální soubor a jsou odlišeny čísly – `/dev/sda1`, `/dev/sda2`, ...

Bod připojení bývá například `/media/disk1` (název může být jakýkoliv, důležité je umístění v adresáři `/mnt` nebo `/media` a také to, abychom jednotlivé disky dokázali rozlišit podle názvu). Hlavní linuxový oddíl se připojuje jako kořenový adresář. Abychom mohli paměťové médium připojit, musí jeho přípojný bod (ten adresář) předem vytvořen.

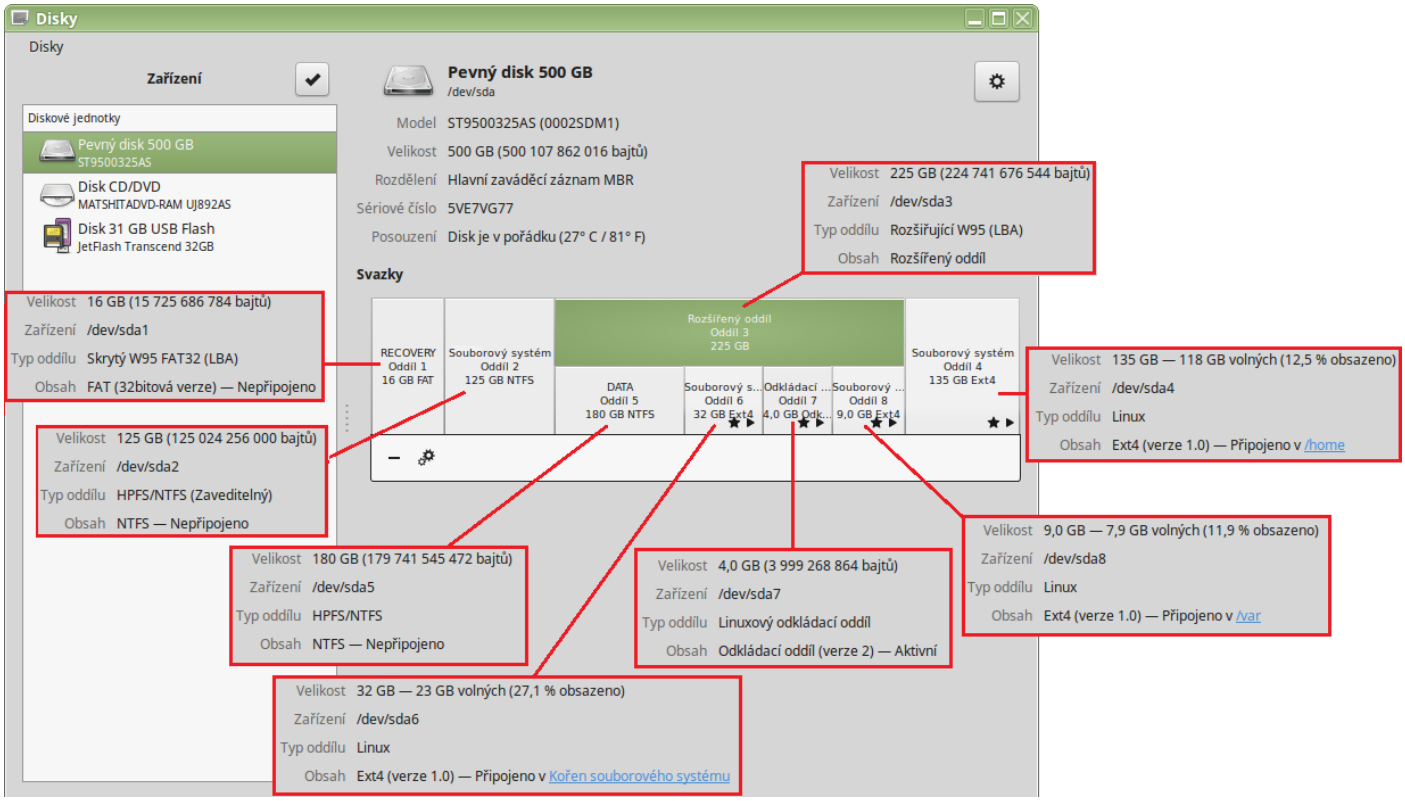
Na obrázku 4.5 jsou poskládány údaje o disku a jeho jednotlivých oddílech. V počítači je jeden hlavní disk, pak optická mechanika (momentálně prázdná) a je připojen jeden USB flash disk. Na pevném disku vidíme víc různých oddílů.

- `/dev/sda1`: je zde informace, že tento oddíl je skrytý a je na něm souborový systém FAT32. Tento oddíl vytvářejí Windows od verze Vista, ale přímo z Windows není dostupný. Není připojen a ani není důvod ho připojovat.
- `/dev/sda2`: na tomto oddílu sou nainstalovány Windows, je na něm souborový systém NTFS a je bootovací (protože z něj lze zavést operační systém). Momentálně není připojen, ale není problém ho připojit.
- `/dev/sda3` (na obrázku je zeleně podbarven): tento oddíl je „rozšířený“, to znamená, že v něm nejsou uložena data, ale je to kontejner na jiné oddíly. Důvodem je, že většina disků používá rozvržení MBR, kde na nejvyšší úrovni můžeme mít maximálně čtyři oddíly (tedy jeden z nich se prohlásí za rozšířený a virtuálně v něm vytvoříme tolik dalších oddílů, kolik budeme potřebovat). Windows nemohou být instalovány dovnitř rozšířeného oddílu, ostatní operační systémy ano.
- `/dev/sda4`: tento oddíl se fyzicky nachází až za rozšířeným oddílem, na konci disku. Je na něm souborový systém ext4, používáme ho jako oddíl pro domovské adresáře (je připojen jako `/home`).
- `/dev/sda5`: opět souborový systém NTFS, tento oddíl je používán ve Windows pro data. Momentálně není připojen, ale klepnutím myši bychom ho mohli připojit a volně přistupovat k datům v něm uloženým.
- `/dev/sda6`: souborový systém ext4, jedná se o hlavní oddíl, ve kterém máme nainstalován Linux, připojuje se jako kořenový adresář.
- `/dev/sda7`: odkládací oddíl, v Linuxu plní podobnou úlohu jako soubor `pagefile.sys` ve Windows.
- `/dev/sda8`: souborový systém je opět ext4, je připojen jako `/var` (to znamená, že „proměnlivé části“ instalace aplikací a systému máme v samostatném oddílu). Důvodem je, že logy a některé další soubory mohou při určitém nastavení logování růst takovou rychlostí, že by ve sdíleném oddílu mohly zbytečně zabírat mnoho místa a utlačovat ostatní typy dat.

Na obrázku 4.6 je disk z jiného počítače, je zde nainstalována jiná distribuce. První oddíl (`/dev/sda1`) byl na notebooku vytvořen prodejcem.



Obrázek 4.4: Pojmenování speciálních souborů (dva pevné disky) při použití *udev*



Obrázek 4.5: Informace o jednotlivých oddílech disku

Na následujícím (`/dev/sda2`) je nainstalován systém Windows Vista, bod připojení je `/media/diskC` a má souborový systém NTFS. Kdybychom například chtěli přistupovat k profilu některého uživatele, pracujeme s adresářem `/media/diskC/Users/`⟨`uživatel`⟩. Na oddílu `/dev/sda3` je souborový systém NTFS, dostaneme se k němu přes bod připojení `/media/diskD`.

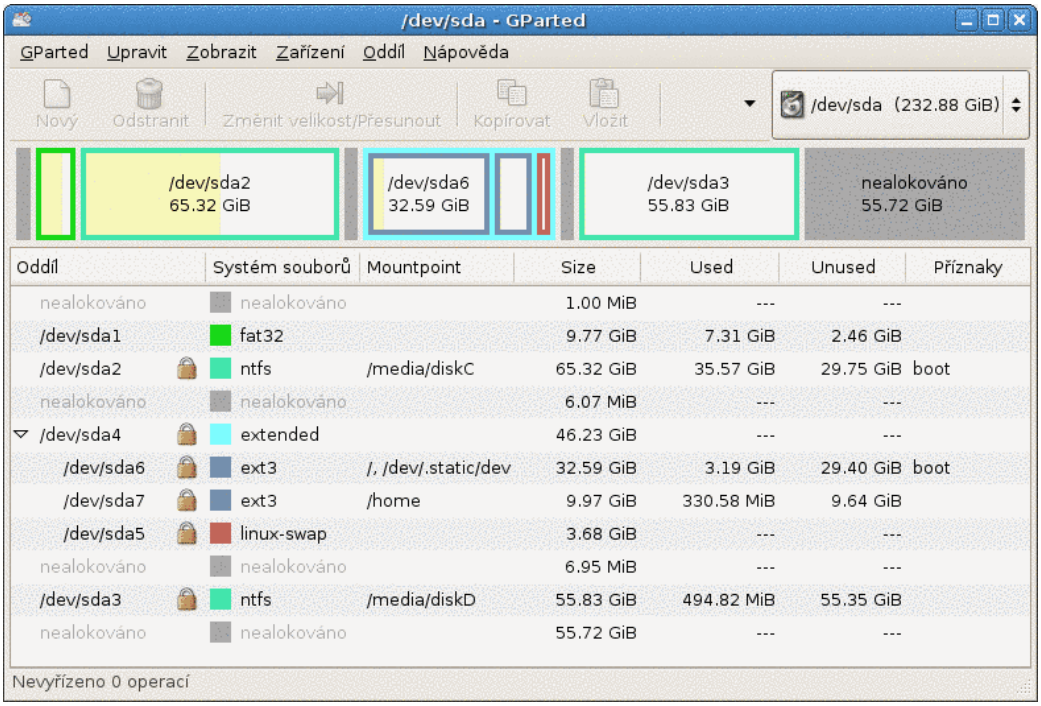
Oddíl `/dev/sda4` je rozšířený. V něm najdeme oddíly `/dev/sda5` (odkládací oddíl), `/dev/sda6` (hlavní oddíl, z něj se načítá Linux) a `/dev/sda7` (podle bodu připojení poznáme, že v něm jsou domovské adresáře uživatelů). Všimněte si, že u `/dev/sda6` jsou uvedeny dva přípojné body oddělené čárkou. První je kořenový adresář systému, ten druhý vede do adresáře `/dev`, do jeho skrytého podadresáře `.static`.

V oddílu `/dev/sda6` se nacházejí všechny adresáře z dříve popsané struktury, které nejsou ve vlastním oddílu. Například

- `/boot` ..... adresář v oddílu 6
- `/home/`⟨`uživatel`⟩ ..... adresář v oddílu 7
- `/etc/fstab` ..... soubor v oddílu 6
- `/dev/sda1` ..... soubor v oddílu 6
- `/usr/lib` ..... adresář s podadresářem v oddílu 6

Při instalaci Linuxu lze vytvořit jakékoliv množství oddílů a uložit v nich kterékoliv adresáře (jako body připojení). Jeden oddíl je vždy hlavní, jeho přípojný bod je `/`, a vše, co nemá vlastní oddíl, se přidá sem. Můžeme se například setkat s tím, že existuje samostatný oddíl pro `/usr` (s nainstalovanými aplikacemi) nebo `/boot`.

Na obrázku 4.7 je nástroj *Informační centrum* z desktopového prostředí KDE z roku cca 2008 (tj. už hodně stará instalace, letitý obrázek). Když srovnáte údaje o paměťových zařízeních s předchozími



Obrázek 4.6: Správa oddílů disku v GNOME, program *GPartEd*

obrázky, určitě si všimnete, že speciální soubory disků a oddílů jsou jinak nazvány (například místo `/dev/sda` je tu `/dev/hda`). Je to proto, že ve starších distribucích se o speciální soubory staral souborový systém devfs místo současného udev, a devfs používal jiné názvy souborů. Mimochodem, pro SCSI disky a USB flash disky se používala syntaxe `/dev/sdxxx`.

Optická mechanika má speciální soubor `/dev/cdrom` nebo podobný, ale také může být použita syntaxe podle pevných disků, zejména pro média v optické mechanice (podobně jako máme disk vs. oddíly). Pro názvy bodů připojení platí totéž co pro běžné disky.

USB flash disky jsou obvykle připojovány jako jako pevné disky, tedy například flash disk může být `/dev/sdb1` (ale může to být `/dev/usb0`, `/dev/usb1`, ...). Opět potřebujeme bod připojení, ve kterém bude přístupná adresářová struktura uvnitř disku.

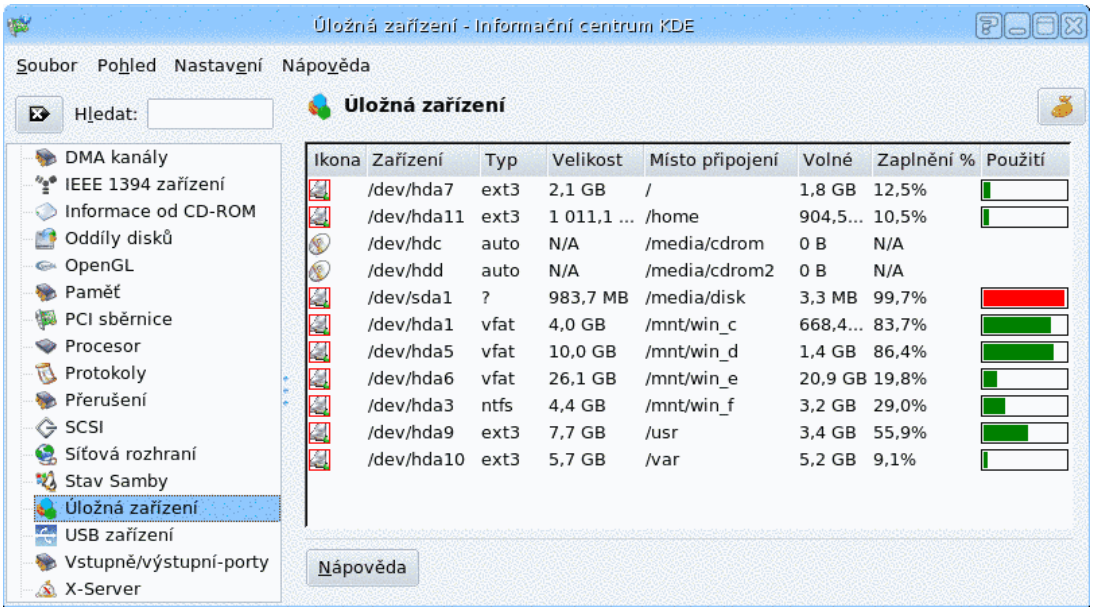
Tiskárna `/dev/lp0`, zařízení připojená k sériovým portům jsou `/dev/tty0`, `/dev/tty1`, ..., k paralelním portům `/dev/lp0`, `/dev/lp1`, ... (také LPT tiskárna).

Virtuální zařízení jsou taková zařízení, pro která nemusí existovat fyzický „protějšek“.

Zařízení `/dev/null` má stejný význam jako zařízení `NUL` ve Windows, přesměrovává se zde vše, u čeho nás nezajímá výstup, například různá upřesňující hlášení některých programů (je to tedy výstupní zařízení) – odpadkový koš.

Zařízení `/dev/random` a `/dev/urandom` generují náhodná čísla a jsou tedy vstupními zařízeními, první z nich používáme, pokud nám více záleží na bezpečnosti (generování kryptografických klíčů), druhé může být zase rychlejší. Zařízení `/dev/zero` zase použijeme jako vstupní, pokud chceme mít na vstupu číslo nula (například chceme vytvořit soubor určité délky bez smysluplného obsahu).

Blokové a znakové speciální soubory. Rozlišujeme speciální soubory *blokové* a *znakové*. Znaková zařízení jsou například `/dev/random`, `/dev/zero`, tiskárny, `/dev/null` – posíláme data o délce 1 B (resp. sekvence takovýchto dat), bloková zařízení jsou obvykle vnější paměti, například pevný disk



Obrázek 4.7: Správa oddílů disku v KDE, starší verze jádra používající devfs

nebo USB flash disk – komunikují se po blocích o stanovené délce, která obvykle bývá značně větší než 1 B s tím, že hranice mezi jednotlivými osmicemi bitů nejsou důležité, ale hlavně: přenášejí se nejen data, ale i metadata (data o datech – například kromě obsahu souboru je třeba dát na vědomí jeho název, pozici v adresářové struktuře, přístupová oprávnění, atd.).

Úkoly

1. Připojte USB flash disk, pokud máte nějaký k dispozici, a sledujte, jaké údaje se objeví v nástroji pro správu disků, který ve své distribuci máte.
2. Projděte si adresář `/dev`. Najděte speciální soubory pro pevné disky včetně těch odkazujících na oddíly na nich.
3. Zjistěte, které diskové oddíly jsou použity pro jednotlivé adresáře.

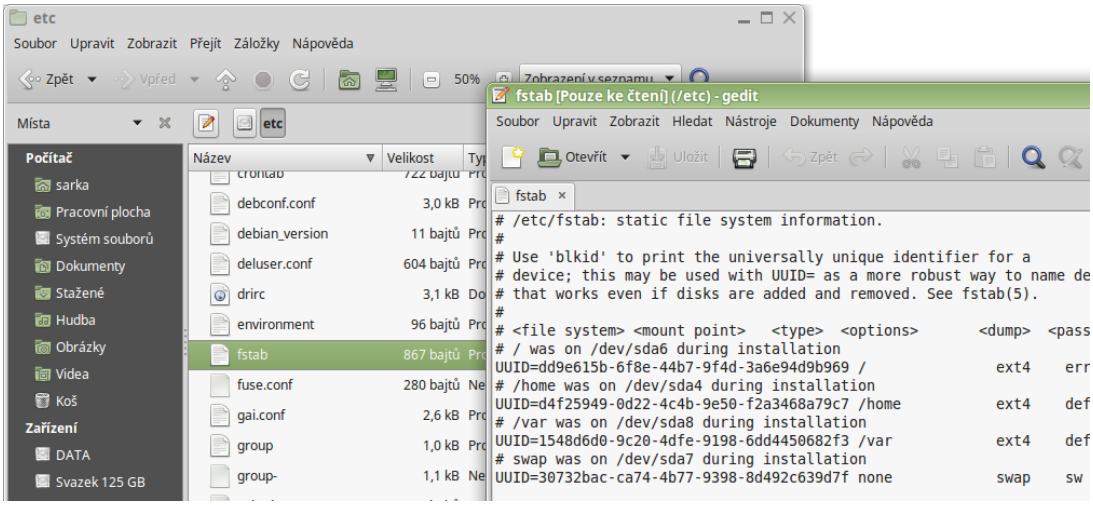
4.4.3 Soubory s evidencí paměťových médií

Zatím jsme si řekli o několika adresářích souvisejících s paměťovými médii:

- v adresáři `/dev` máme speciální soubory zařízení (včetně paměťových médií),
- v adresáři `/media` nebo `/mnt` jsou přípojné body pro paměťová média.

Ale to není všechno.

V souboru `/etc/fstab` najdeme *seznam připojitelných souborových systémů*, jinými slovy – jsou tu záznamy o oddílech disku, výměnných paměťových médiích a virtuálních souborových systémech, které se obvykle připojují buď při startu systému, nebo dynamicky při jejich fyzickém připojení. Účelem těchto záznamů je zjednodušit jejich připojování, tedy dodávají informace potřebné pro připojení, abychom je nemuseli pokaždé zadávat sami.



Obrázek 4.8: Soubor `/etc/fstab`

Struktura souboru se může zdát trochu chaotická. Taky je to textový soubor a vpodstatě tabulka, přičemž oddělovačem údajů na řádku není dvojtečka, ale tabulátor. V záhlaví souboru máme v komentářích vysvětleno, co se v souboru nachází (což je v konfiguračních souborech dobrým zvykem), poslední řádky záhlaví popisují obsah jednotlivých sloupců. Například první sloupec je označen jako „file system“ (tedy souborový systém), což znamená stanovení konkrétního oddílu na disku. Bývá určen buď speciálním souborem nebo dlouhým řetězcem, který je unikátním identifikátorem oddílu, případně v připojeném komentáři k záznamu (nad záznamem) máme i ten speciální soubor. Druhý sloupec je „mount point“ (bod připojení), kterému už „uživatelsky“ rozumíme víc. Následuje souborový systém (většinou ext4fs), pak parametry připojení.

Původně se v souboru `/etc/fstab` objevovala i výměnná paměťová média (například USB flash disk), ale ve většině distribucí se již od toho upouští a pro automatizaci jejich připojování se používá jiný mechanismus.

Podobně se jmenuje soubor `/etc/mtab`, který však na rozdíl od předchozího obsahuje *právě připojené souborové systémy*. Má podobný formát jako `/etc/fstab`, ale jako oddělovač údajů na řádku je použita mezera místo tabulátoru, má víc řádků (protože tam najdeme i virtuální souborové systémy a právě připojená paměťová média. Jako první parametr je v `/etc/mtab` zásadně používán název speciálního souboru, žádný dlouhý nečitelný identifikátor, což taky zvyšuje přehlednost.

Dalším typickým rysem souboru `/etc/mtab` je to, že u jednotlivých souborových systémů (oddílů apod.) jsou skutečné parametry, které byly použity při připojení. Například u diskového oddílu může být v souboru `/etc/fstab` uvedeno „defaults“, což znamená, že když bude tento oddíl připojován, mají být použity typické (výchozí) hodnoty pro daný souborový systém, ale v souboru `/etc/mtab` vidíme, že se za „defaults“ dosadil parametr „rw“ (tedy připojeno pro čtení i pro zápis) nebo u souborového systému FAT32 na USB flash disku volby pro kódování znaků v názvech souborů (dodnes existují podivní lidé, kteří v názvech souborů používají českou diakritiku) a další.

Úkoly

1. Najděte ve své distribuci aplikaci pro práci s disky. Projděte si její rozhraní a prohlédněte si parametry připojených paměťových médií.
2. Projděte si adresář `/dev` a zjistěte, kolik v něm je speciálních souborů pro oddíly na disku. Najděte

také speciální soubory výše jmenovaných virtuálních zařízení.

3. Otevřete soubory `/etc/fstab` a `/etc/mtab`, srovnejte jejich obsah. Najděte v obou hlavní oddíl, na kterém běží Linux, a zjistěte, jestli jsou další části systému v jiných oddílech (pokud jste při instalaci použili výchozí nastavení nebo máte live distribuci, bude asi vše na jediném oddílu). Všimněte si, které záznamy jsou v obou souborech a které jen v jednom z nich.
4. Pokud máte po ruce USB flash disk nebo optický disk, připojte ho. Sledujte, jak se změní soubor `/etc/mtab` (asi bude třeba ho zavřít a znovu otevřít). Podívejte se, kde v adresářové struktuře je přípojný bod pro toto zařízení (zřejmě bude někde v adresáři `/media`). Podívejte se na parametry paměťového média v aplikaci pro práci s disky.

4.5 Správa uživatelů

4.5.1 Uživatelé a skupiny

O uživatelích včetně příslušných nástrojů v grafickém rozhraní jsme si už něco řekli v předchozí kapitole, zde navážeme a prohloubíme své znalosti.

 Nejdůležitějším uživatelem je *root* (v jiných operačních systémech se nazývá supervizor, správce, administrátor). Má nejrozsáhlejší práva, může provádět konfiguraci systému, pracovat s uživatelskými účty včetně definování nových uživatelů, instalovat jakékoliv aplikace nebo přidávat či ubírat moduly jádra.

Poznámka:

Pozor, není pravda, že „root může vše“. Ve většině linuxových distribucí jsou z bezpečnostních v současné době používána různá řešení, která omezují pravomoci roota (například modul SELinux – Security-Enhanced Linux), protože účet roota je zneužitelný v mnohem větší míře než běžné uživatelské účty. Ze stejného důvodu je běžně zakázáno přihlašování pod účtem root.

 Uživatel může patřit i do více uživatelských *skupin*. Skupiny se obvykle používají pro určení uživatelů úžeji spolupracujících například na nějakém projektu, mohou být samozřejmě dočasné.

Jedna skupina je pro konkrétního uživatele vždy hlavní, *primární*, ale může patřit (a taky obvykle patří) do více skupin. Skupiny slouží také jako přístupové filtry (k určitým objektům mohou přistupovat jen takoví uživatelé, kteří jsou členy konkrétní skupiny), tedy když uživatel chce k takovému objektu přistupovat a jeho primární skupina není „ta pravá“, přepne se do jiné skupiny (k tomu je třeba přistupit jen velmi vzácně).

Ve většině distribucí je po přidání nového uživatele vytvořena stejnojmenná skupina (takže když vytvoříme uživatele `novak`, automaticky se vytvoří skupina `novak` s tímto jedním členem).

 Všichni uživatelé mají přiřazeno identifikační číslo UID (User ID), uživatel root má vždy UID = 0. Stejně tak i skupiny mají přiřazeno identifikační číslo GID (Group ID). Uživatel má přiřazeno

- *UID* (s tímto číslem pracují bezpečnostní mechanismy systému),
- *login* – přihlašovací jméno, zadává ho při přihlašování do systému,
- *password* – heslo,

- vlastní *domovský adresář* (Home Directory), je přístupný jen jemu a rootovi, jeho název je obvykle stejný jako login,
- *skupiny*, do kterých patří, jedna ze skupin je pro něj primární (nejdůležitější),
- výchozí *shell* (něco na způsob Příkazového řádku ve Windows).

Čísla UID a GID může root určovat jakkoliv (při vytvoření uživatele a skupiny, UID běžných uživatelů bývá obvykle nejméně trojciferné). Jediné omezení může nastat při propojení více UNIXových počítačů do sítě tak, že uživatel (nebo skupina) může pracovat na různých počítačích bez nutnosti vytváření účtů na všech systémech.

Poznámka:

Existuje hodně systémových uživatelů (mnohé služby mají vlastní účty). Jejich UID je obvykle malé číslo, kdežto UID běžných uživatelů je velké číslo. Hranice mezi těmito dvěma „kategoriemi“ je 500 nebo 1000 (podle nastavení v konkrétní linuxové distribuci).

4.5.2 Adresáře a soubory související s uživateli

Každý uživatel (kromě systémových uživatelů) má svůj *domovský adresář*. Domovské adresáře většiny uživatelů najdeme v adresáři `/home`, jsou obvykle pojmenovány podle přihlašovacího jména uživatele. Tedy například domovským adresářem uživatele *novak* bude adresář `/home/novak`. Výjimkou je uživatel root, jehož domovský adresář je v kořenovém adresáři: `/root`.

V domovském adresáři uživatele najdeme jak dokumenty daného uživatele, tak i veškerá nastavení včetně různých konfiguračních souborů. Většina z nich je skrytá (jejich název začíná tečkou), takže běžný uživatel je nevidí a nemá snahu do nich jakkoliv zasahovat.

Seznam uživatelů najdeme v souboru `/etc/passwd`. Je to textový soubor, takže stačí poklepat a otevře se v textovém editoru (třebaže nemá příponu – v UNIXových systémech se primárně rozpoznává typ souboru podle obsahu). V tomto souboru je pro každého uživatele samostatný řádek, na tomto řádku jsou jednotlivé informace o uživateli odděleny dvojtečkou. Můžeme si to představit tak, že se jedná o tabulku, kde řádky představují jednotlivé uživatele, sloupce pak vlastnosti konkrétního uživatele, přičemž sloupce na řádku jsou odděleny právě dvojtečkami.

Příklad

Řádky souboru `/etc/passwd` mohou vypadat například takto:

```
root:x:0:0:root:/root:/bin/bash
syslog:x:101:104::/home/syslog:/bin/false
pulse:x:110:119:PulseAudio daemon,,,:/var/run/pulse:/bin/false
novak:x:1000:1000:Jan Novák,,,:/home/novak:/bin/bash
deti:x:1001:1001:Účet pro děti,,,:/home/deti:/bin/bash
```

Jednotlivé „sloupce“ (pole na řádku) mají tento význam:

- přihlašovací jméno uživatele, například `novak`,
- pole pro heslo, ale místo hesla tu najdeme jen písmeno „x“, hesla jsou ve skutečnosti jinde,

- následují dvě čísla, z nichž první je UID uživatele a druhé je GID jeho primární skupiny (systémové účty mají tato dvě čísla nízká, běžní uživatelé naopak vyšší než stanovená hranice),
- další pole je dnes většinou „občanské jméno“ uživatele či služby, taky se označuje jako GECOS (General Electric Comprehensive Operating System), což je standardizovaný formát pro identifikaci osoby (mezi těmi „čárkami navíc“ by případně mohla být adresa, telefonní číslo apod.),
- poslední pole je shell (interpretační program pro textový režim) – u lidských uživatelů tam obvykle bývá `/bin/bash`, u systémových „falešný“ `/bin/false`, protože tyto služby by rozhodně neměly provádět příkazy shellu.

 Jak bylo výše uvedeno, hesla (přesněji hashe hesel) jsou jinde – konkrétně v souboru `/etc/shadow`. Tento soubor má přísněji nastavená přístupová oprávnění, ne každý si může zobrazit jeho obsah. Struktura je podobná jako u předchozího souboru (co uživatel, to řádek, pole na řádku jsou oddělena dvojtečkami). V prvním poli na řádku je opět přihlašovací jméno uživatele, v druhém poli pak buď hvězdička (to znamená, že heslo není definováno – typické pro systémové procesy) nebo hash hesla (takže ne přímo heslo, ale pouze jeho kontrolní součet). Další pole pro nás zatím nejsou důležitá, je tam například kdy naposledy bylo heslo změněno, maximální doba platnosti hesla, atd.

 Také seznam skupin je dosažitelný v souboru, konkrétně `/etc/group`. Struktura souboru je opět podobná – na každém řádku skupina, pole na řádku jsou oddělena dvojtečkou.

 **Příklad**

Řádky souboru `/etc/group` mohou vypadat například takto:

```
cdrom:x:24:novak,deti
audio:x:29:pulse,novak,deti
novak:x:1000:
deti:x:1001:
```

 Význam jednotlivých sloupců je zřejmý – první prvek je název skupiny (například skupina `cdrom` je skupinou uživatelů, kteří mohou přistupovat k optickým mechanikám), skupiny `novak` a `deti` jsou pro běžné uživatele. Následuje pole pro heslo (opět jen písmeno „x“, také skupiny mohou mít heslo – v souboru `/etc/gshadow`), pak máme GID (identifikační číslo skupiny) a seznam uživatelů do skupiny zařazených.

 **Poznámka:**

Určitě jste si všimli (a bylo na to upozorňováno), že pro běžné uživatele obvykle existuje primární skupina pojmenovaná stejně jako přihlašovací jméno uživatele. Taková skupina se označuje zkratkou *UPG* (User Private Group – Soukromá/Vlastní uživatelská skupina). Většinou i GID této skupiny se shoduje s UID příslušného uživatele.

V souboru `/etc/group` pro takové skupiny nemáme uvedeny žádné členy, tedy ve skutečnosti je u skupiny vždy členem uživatel se stejným jménem jako je název skupiny, pokud existuje. Kdyby například existoval uživatel s názvem `cdrom`, taky by byl členem skupiny `cdrom`, třebaže není v souboru uveden.

Úkoly

1. Zjistěte informace o sobě jako uživateli – své UID, skupiny, do kterých patříte, GID své primární skupiny. Prohlédněte si výše uvedené soubory, pokud k nim máte přístup. Projděte si také svůj vlastní domovský adresář (nezapomeňte si zapnout zobrazování skrytých souborů).
2. Projděte si svůj domovský adresář (se zapnutým zobrazením skrytých souborů). Zjistěte, kam se ukládají konfigurační skripty, dokumenty, apod.

4.6 Procesy

Procesy v UNIXových systémech jsou seřazeny v *hierarchické* (stromové) *struktuře* založené na vztahu předek–potomci. Potomci jsou spouštěni svým předkem a dědí některé jeho vlastnosti.

Procesy mají také identifikační čísla, a to PID (Process ID), každý proces také musí vědět, kdo je jeho předek (číslo PPID – Parent Process ID), a obvykle platí, že po ukončení činnosti předka končí i jeho potomci (jsou výjimky). Existují i další identifikační čísla, která má proces přidělena (PGID, SID), těmi se budeme zabývat na přednáškách v příštím semestru.

Každý proces má svého *vlastníka*. Obvykle (ne vždy) to bývá uživatel, který tento proces spustil (třeba zprostředkovaně přes jiný proces). Vlastníkem systémových procesů je většinou uživatel *root*. Od vlastníka se také odvíjejí přístupová oprávnění tohoto procesu. Každý proces má také přidělenu skupinu – většinou to bývá primární skupina uživatele, který tento proces spustil. Opět to má význam pro přístupová oprávnění daného procesu.

Z toho vyplývá, že každý proces má přiděleno UID (vlastníka) a GID (skupinu), podobně jako uživatelé.

Po startu systému běží pouze jediný proces, *init*. Jeho vlastníkem je *root*, pro init je PID = 1 (proces s hodnotou PID = 0 vlastně neexistuje, jednalo se o zaváděcí program operačního systému, tedy například Grub). Proces init pak spouští další procesy, tedy všechny další procesy v systému běžící jsou jeho potomci, přímí nebo nepřímí. Spouští především systémové procesy většinou běžící na pozadí, kterým se v UNIXu říká *démony* (daemons). Speciálním procesem je také *jádro* (kernel).

Poznámka:

Podle původní koncepce UNIXu *jsou programy malé a jednoduché*, a tedy i procesy, a hlavní síla UNIXu je v jejich *efektivní spolupráci*, která znamená především předávání výstupů jednoho procesu na vstupy jiného. Je to možné několika způsoby, jeden z nich je používání tzv. rour. *Roura* (Pipe nebo Pipeline) je vlastně dočasný soubor, který slouží k takovémuto předávání dat. V příkazech shellu nebo ve skriptu se obvykle realizuje zřetězením příkazů symbolem roury, |.

Může se stát, že se některý proces odmítá ukončit. Pak nezbývá než ho „zabít“. Jde to jak v grafickém, tak i v textovém prostředí. V předchozí kapitole jsme si ukazovali nástroj pro sledování procesů, tam stačí pro daný proces použít příslušnou volbu v kontextovém menu.

Úkoly

1. Spusťte některý nástroj, ve kterém máte přístup ke struktuře běžících procesů (podle toho, ve kterém prostředí pracujete). Pokud není vyznačena hierarchická struktura, zapněte zobrazení stromu procesů. Najděte proces init a zjistěte, jaké má „přímé potomky“.
2. Ověřte PID procesu init. Dále seřaďte procesy podle PID a u procesů s nejnižšími hodnotami se pokuste odhadnout pořadí, ve kterém byly spuštěny. Zobrazte sloupec s údajem o času spuštění procesů (pokud je podporován používaným nástrojem) a porovnejte svůj odhad.
3. Opět zapněte zobrazení stromové struktury. Spusťte některý program a sledujte, kde se objeví ve struktuře procesů a jaké PID mu bude přiděleno. Pak tento program ukončete prostřednictvím nástroje se seznamem procesů.

4.7 Virtuální souborové systémy

Jak jsme viděli v souboru `/etc/mtab`, existuje poměrně hodně souborových systémů, které nesouvisí s žádným konkrétním paměťovým zařízením. Těmto souborovým systémům říkáme *virtuální*, a obvykle jde o speciální přístupová rozhraní k určitým druhům informací. Fyzicky se nenacházejí na pevném disku ani výměnných paměťových médiích, jsou pouze v operační paměti. S jedním jsme se už setkali – *udev*. Tento souborový systém zajišťuje přístup k zařízením, především jejich speciální soubory.

Souborový systém *VFS* (Virtual File System) je nejdůležitější – udržuje a řídí celou adresářovou strukturu s kořenem `/` a zprostředkovává přístup ke všemu, co je připojeno.

Velice důležitým virtuálním souborovým systémem je *procfs* připojovaný do adresáře `/proc`. Najdeme zde běhové informace o systému a procesech, tj. informace, které se dynamicky mění za běhu systému (proto běhové). Najdeme v něm spoustu podadresářů pojmenovaných čísly, pak několik adresářů pojmenovaných řetězci a několik souborů. Adresáře pojmenované čísly náleží procesům – víme, že každý proces je jednoznačně identifikován svým PID, a právě toto identifikační číslo je použito jako název příslušného podadresáře. Tedy například v podadresáři nazvaném `3548` budou běhové informace o procesu s PID 3548. Vše ostatní jsou běhové informace o systému (jádře).

Ať už se jedná o soubory (nebo „slovní“ podadresáře) přímo v adresáři `/proc` nebo to, co se nachází v adresářích jednotlivých procesů, význam je dost podobný. Například pro procesy může být:

- `exe` je link na spustitelný soubor, ze kterého proces vznikl,
- `cmdline` obsahuje příkazový řádek procesu – zde se dozvíme, z jakého souboru a s jakými parametry byl proces spuštěn,
- `status` je soubor s podrobnými informacemi o procesu (stav, využití paměti, počet vláken procesu, procesory povolené pro tento proces, atd.),
- `net` je adresář (ne soubor) a obsahuje vše, co souvisí s komunikací procesu na síti, atd.

V případě celého systému (tj. rovnou v adresáři `/proc`) taky najdeme soubor `cmdline`, ve kterém je příkaz, jímž byl spuštěn systém (včetně parametrů) a taky najdeme podadresář `net` s informacemi týkajícími se sítě. V systémové části souborového systému jsou i další zajímavé položky, například:

- `cpuinfo` je soubor s informacemi o procesoru,
- `meminfo` obsahuje informace o využívání paměti,

- `modules` je soubor se seznamem modulů právě načtených v jádře (jsou tam ovladače, antivirus, podpora síťových protokolů, kodeků apod.),
- `version` informuje o verzi jádra Linuxu a linuxové distribuce,
- `ioports, iomem, dma, interrupts` obsahují informace o adresách pro nízkoúrovňovou komunikaci s hardwarem, adresy firmwaru různých zařízení, DMA kanály, přerušení, atd.

Poznámka:

Nepokoušejte se otevírat soubory z adresáře `/proc` v textovém editoru. Sice to vpodstatě jsou textové soubory (ne všechny, například `exe` je soft link), ale berte v úvahu, že se nejedná o fyzické soubory – jsou pouze v operační paměti. Textové editory mají ve zvyku otevíraný soubor uzamknout, ale to právě v `/proc` nelze, systém to nedovolí. Takže byste se dočkali pouze chybového hlášení. Nicméně – v textovém režimu to jde, například není problém provést příkaz `cat /proc/cpuinfo` (zobrazíme obsah souboru `/proc/cpuinfo`).

Virtuální souborový systém *sysfs* se připojuje do adresáře `/sys` a obsahuje běhové informace o zařízeních. V jeho podadresářích jsou zařízení členěna podle určitých kritérií a tam jsou uloženy určité typy údajů o daném zařízení, například

- `/sys/devices` – fyzické vztahy mezi zařízeními (co je k čemu připojeno),
- `/sys/bus` – zařízení tříděná podle sběrnic (PCI, USB, atd.),
- `/sys/class` – zařízení tříděná podle tříd (typů zařízení),
- `/sys/block` – přístup k blokovým zařízením včetně disků,
- `/sys/power` – nízkoúrovňový přístup ke správě napájení.

Dalším virtuálním souborovým systémem je *tmpfs* připojovaný do adresáře `/run`. Používá se při startu systému, kdy ještě nejsou připojeny žádné oddíly (ani hlavní oddíl), systém a případně procesy sem zapisují informace, které by jinak zapisovaly do některého adresáře v oddílech.

Další informace:

Informace o standardu pro adresářovou strukturu UNIXů (File Hierarchy Standard) jsou také na http://www.pathname.com/fhs/.

Úkoly

1. V adresáři `/proc` si prohlédněte informace o procesoru, využité operační paměti, atd. – běhové informace o systému.
2. V adresáři `/proc` zvolte některý podadresář pojmenovaný číslem (spíše víceciferným) a prohlédněte si ho – zjistěte, ve kterých souborech se nacházejí informace jako například příkaz, kterým byl proces spuštěn, využití paměti apod. Podobně prozkoumejte také podadresář příslušející procesu init.

4.8 Přístupová oprávnění a vlastnosti souborů

4.8.1 Úvod do přístupových oprávnění

Každý soubor (včetně adresářů) má přiřazeny tři důležité údaje:

- svého *vlastníka* (obvykle ten uživatel, který soubor vytvořil, u systémových souborů je to *root*),
- přidruženou *skupinu*, jejíž členové obvykle mívají k souboru o něco vyšší přístupová práva než ostatní,
- určení přístupových oprávnění pro vlastníka/členy skupiny/ostatní.

Každý soubor má přiřazeny tři sady přístupových práv – pro svého *vlastníka*, *skupinu* (group) a *ostatní* (others – ty, kteří nejsou ani vlastníkem, ani členem přiřazené skupiny). Přístupová práva může ve většině případů měnit jen vlastník souboru nebo *root* (přesněji, kdykoliv *root* něco dělá, jeho práva nejsou až na výjimky prověřována; kdykoliv něco provádí jiný uživatel, jsou při každém přístupu k souborům včetně příkazů jeho práva prověřována).

V každé ze tří sad (vlastník, skupina, ostatní) najdeme nastavení tří oprávnění (celkem tedy devět). Řetězec s přístupovými oprávněními tedy má celkem devět znaků. Tyto znaky mohou být `r` (read, právo čtení), `w` (write, právo zápisu), `x` (execute, právo spouštění), nebo pokud toto právo není přiděleno, pak je zobrazen znak „`-`“ jako symbol odepření práva, to vše pro vlastníka, skupinu a ostatní (tedy tři skupiny po třech znacích). Význam jednotlivých znaků najdeme v tabulce 4.1.

	`r`	`w`	`x`
dokument	číst	zapisovat, měnit	–
spustitelný soubor	číst (zobrazit)	zapisovat, měnit	spustit
adresář	vypsat (zobrazit) obsažené položky	přidávat a mazat položky	zvolit jako pracovní (aktivní) adresář

Tabulka 4.1: Význam přístupových práv

Pro běžné soubory typu dokument, obrázek, video, konfigurační soubor apod. nemá oprávnění `x` smysl, nebývá nastaveno.

Příklad

Řetězec souboru, ke kterému mají všichni povolen jakýkoliv typ přístupu, by byl `rwxrwxrwx`. Ukážeme si řetězce, ve kterých je vždy alespoň jeden typ přístupu odepřen.

`rwxrwxr--` pro vlastníka a stanovenou skupinu jsou přidělena všechna práva, ostatní mohou jen číst (zápis a spouštění jsou pomlčkou zakázány),

`rwx------` vlastník souboru může všechno, ostatní nemohou nic,

`rwxrwx---` vlastník souboru a členové přidružené skupiny mohou všechno, ostatní nemohou nic,

`rwxr--r--` znamená všechna práva pro vlastníka, pro skupinu a ostatní jen čtení,

`rwxrw----` znamená všechna práva pro vlastníka, pro skupinu čtení a zápis, pro ostatní nic,

`rw-r-----` vlastník může číst a zapisovat (spouštění zřejmě nemá smysl, půjde o nějaký dokument), členové stanovené skupiny mohou číst, ostatní nemohou nic,

`r--------` vlastník může číst (je to zjevně soubor pouze pro čtení), ostatní nemají žádný přístup.

Pro *soubor* je význam práv `r`, `w` a `x` jasný (číst soubor, zapisovat, spouštět *spustitelný* soubor). Pokud jde o *skript* (tedy textový soubor s příkazy), můžeme ho spustit buď s použitím příslušného interpretačního programu (tedy jako parametr programu interpretujícího perl, bash, php apod.), anebo přímo – to lze, když jsou splněny tyto podmínky:

1) soubor skriptu má nastaveno právo x (spouštění),
2) na začátku souboru je stanoveno, kterým programem má být skript interpretován (budeme probírat později).

Pro *adresář* znamená čtení vypsat si jeho obsah, zápis změnit obsah adresáře (přidat soubory, změnit, vymazat), spouštění zvolit si ho jako pracovní (možnosti jsou v tabulce 4.1).

Jenže to není všechno. Abychom se mohli dostat k určitému souboru, musí nám být přístupná i cesta k němu, což znamená, že potřebujeme mít právo `x` pro všechny adresáře na cestě k souboru, včetně kořenového.

Příklad

Předpokládejme, že u souboru `/seznamy/xdata.txt` a adresářů na cestě k němu jsou tato oprávnění:

Soubor	Řetězec oprávnění	Vlastník	Skupina
/ (kořenový adresář)	`rwxr-xr-x`	`root`	`root`
`seznamy`	`rwxr-xr--`	`jan`	`marketing`
`xdata.txt`	`rw-r-----`	`jan`	`marketing`

Root má všechna oprávnění ke všem těmto objektům.

Uživatel `jan` se dostane do všech adresářů na cestě (jen nemůže ovlivňovat obsah kořenového adresáře, u nějž pro Jana platí poslední třetina řetězce oprávnění) – u obou adresářů má právo `x`, také může jakkoliv manipulovat se souborem `xdata.txt`, protože pro něj jako vlastníka platí první třetina řetězce oprávnění.

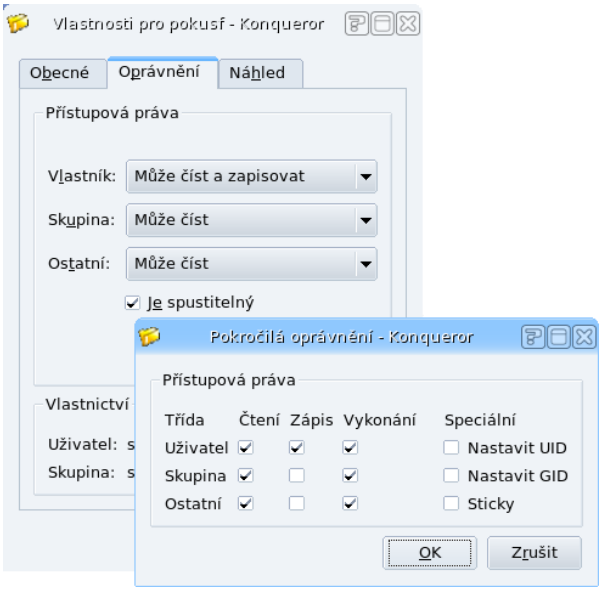
Vezměme si uživatele `pavel`, který je členem skupiny `marketing`. Pro něj platí

- `/` – třetí třetina: `r-x` dostane se dovnitř
- `seznamy` – druhá třetina: `r-x` dostane se dovnitř
- `xdata.txt` – třetí třetina: `r--` může pouze číst

Naproti tomu pro uživatele `lukas`, který není ani vlastníkem, ani členem přidružené skupiny `marketing`, to bude jiné:

- `/` – třetí třetina: `r-x` dostane se dovnitř
- `seznamy` – třetí třetina: `r--` tady se zasekne, nemůže mít tento adresář jako pracovní, nemůže dovnitř
- `xdata.txt` – třetí třetina: `---` nemá žádná oprávnění; i kdyby je měl, je mu to na nic, protože se k souboru tak jako tak ani nedostane (potřeboval by přidat právo `x` pro `seznamy`).

4.8.2 Speciální příznaky



Obrázek 4.9: Přístupová oprávnění v Linuxu, včetně speciálních příznaků

Přístupová práva pro uživatele jsou celkem jasná – stanovují, co je dovoleno uživateli, pokud přistupuje k určitému objektu. Když vzniká něco nového (proces, nový soubor apod.), musí se také stanovit jeho přístupová práva (u procesu) nebo přístupová práva k němu (nový soubor).

Za obvyklých okolností nově vytvořený objekt dědí přístupové vlastnosti od svého tvůrce, tedy při vytváření nového souboru (včetně adresáře) se stává vlastníkem ten, kdo tento soubor vytvořil, a když vznikne proces, jeho přístupová oprávnění se dědí také od uživatele, který ho spustil. Totéž platí o nově vytvořeném adresáři. Dědění těchto vlastností však lze ovlivnit, a to speciálními příznaky přístupových práv – *SUID* (SetUID), *SGID* (SetGID) a *Sticky*.

Jakýkoliv soubor nebo adresář může mít tyto příznaky nastaveny, i když u většiny tomu tak nebývá. Mají význam pouze pro adresáře a spustitelné soubory.

V tabulkách 4.2 a 4.3 je přehledně vysvětlen význam nastavení bitů SUID, SGID a Sticky (SUID se používá jen u procesů, Sticky zase jen u adresářů; SGID u procesů a i adresářů).

Nejdřív se podíváme na procesy. Když spustíme spustitelný soubor, vznikne proces. Proces přistupuje k různým objektům (souborům, zařízením apod.), a k nim potřebuje určitá přístupová oprávnění, tedy na něj nahlížíme podobně jako na uživatele.

U starších verzí UNIXů byl Sticky bit používán také u spustitelných souborů – když byl nastaven, pak po ukončení zůstal kód v operační paměti, aby v případě opětovného spuštění téhož programu nebylo nutné kód znovu do paměti načítat. V současné době se již nikde nepoužívá.

Efektivním uživatelem procesu je uživatel, jehož oprávnění proces uplatňuje, *efektivní skupinou* je skupina, jejíž oprávnění proces uplatňuje. Za normálních okolností je efektivním uživatelem ten uživatel, který proces spustil, efektivní skupinou jeho primární skupina (nebo jiná skupina, do které uživatel patří, pokud přepnul mezi skupinami). SUID bit nastavený u spustitelného souboru může ovlivnit nastavení efektivního uživatele, SGID bit může ovlivnit nastavení efektivní skupiny.

Nastavení SUID na 1 se používá pro programy, které obvykle spouští běžný uživatel, ale tento program při své činnosti potřebuje vyšší přístupová oprávnění.

Příklad

Při změně hesla musí běžný uživatel spustit proces, který přistupuje k souboru `/etc/shadow`, jenže k tomuto souboru jsou oprávnění následující: `rw-r-----`, přičemž vlastníkem je uživatel root. Z toho vyplývá, že nikdo kromě roota nemá k tomuto souboru právo zápisu, a proces spuštěný běžným uživatelem taky ne.

Jenže spustitelný soubor, který se používá pro změnu hesla (`/usr/bin/passwd` má nastaven SUID bit a jeho vlastníkem je root, proto při spuštění běžným uživatelem vznikne soubor, jehož efektivním uživatelem je root ⇒ tento proces má oprávnění zapisovat do souboru `/etc/shadow` a běžný uživatel si takto může změnit heslo.

Spustitelný soubor:	**= 0**	**= 1**
SUID =	Když spouštíme spustitelný soubor, efektivním uživatelem vzniklého procesu se stáváme my; proces přebírá přístupová práva od toho, kdo ho spustil.	Když spouštíme soubor, efektivním uživatelem vzniklého procesu se stává vlastník souboru, proces zdědí přístupová práva od vlastníka souboru.
SGID =	Efektivní skupinou procesu se stává skupina uživatele, který tento proces spustil. Přístupová práva pro skupinu se určují podle skupiny, do které patří spouštějící uživatel.	Efektivní skupinou procesu je skupina nastavená pro daný spustitelný soubor. Přístupová práva pro skupinu se určují podle skupiny nastavené u souboru.

Tabulka 4.2: Speciální práva pro spustitelné soubory

Aplikace v grafickém rozhraní je pouze frontend (přístupové rozhraní) k spustitelnému souboru `/usr/bin/passwd`, tedy pro ni platí totéž.

SUID bit mají kromě `/usr/bin/passwd` nastaven i jiné spustitelné soubory, například `/bin/ping` (ověřování dostupnosti zařízení na síti), `/bin/mount` (připojování paměťových médií), `/usr/bin/sudo` (spouštění programů s vyššími oprávněními).

SGID bit má nastaven například příkaz `/usr/bin/chage` sloužící k nastavení parametrů uživatelského účtu souvisejících s platností a heslem (například se dá účet deaktivovat, nastavit maximální doba platnosti hesla apod.), `/usr/bin/wall` pro komunikaci mezi uživateli (píšou si „na zeď“), `/usr/bin/ssh-agent` (autentizační agent) nebo některé hry.

Poznámka:

SUID bit u spustitelného souboru může být nebezpečný (vlastně taky SGID) – spustitelných souborů, jejichž vlastníkem je root a mají nastavený SUID bit, by mělo být co nejméně, jen v nejnutnějších případech. Většina případů SUID bitu se ve skutečnosti dá řešit jinak, například příkazem pro navýšení práv (taktéž používaným s rozmyslem).

Nyní se podíváme na speciální příznaky u adresářů. Když v adresáři vzniká nový soubor (včetně podadresáře), je třeba stanovit jeho vlastníka, přiřazenou skupinu (tj. skupinu, jejíž členové budou mít k adresáři vyšší oprávnění než „zbytek světa“) a řetězec přístupových oprávnění. Vlastníkem se stává ten, kdo soubor vytváří, jako přiřazená skupina se většinou použije skupina vytvářejícího uživatele. SGID bit adresáře ovlivňuje přiřazení skupiny pro soubory vytvářené uvnitř něj.

Sticky bit má vliv trochu jiného typu. Za normálních okolností potřebujeme ke smazání souboru právo zápisu pro nadřízený adresář (ano, můžeme klidně smazat i takový soubor, ke kterému nemáme žádná oprávnění, stačí mít právo `w` k nadřízenému adresáři), protože vlastně jde o operaci změny obsahu nadřízeného adresáře. Za určitých okolností to může dost vadit – například pokud skupina lidí pracuje na tomtéž projektu a mají soubory projektu v jednom sdíleném adresáři (třeba na serveru), pak samozřejmě všichni budou potřebovat právo zápisu do tohoto adresáře (aby tam případně mohli

přidávat nové soubory). Potom by však někdo ze skupiny mohl (ať už úmyslně nebo neúmyslně) smazat soubor vytvořený někým jiným. To je špatně. Pokud ale má nadřízený adresář nastaven Sticky bit, jsou omezeny kompetence pro odstraňování souborů: mazat soubory může pouze ten, kdo je vlastníkem mazaného souboru (nebo root). Přidávat nové soubory není problém, ale mazání je omezeno, třebaže pro obojí by jinak stačilo právo `w` pro nadřízený adresář.

Nastavení speciálních oprávnění se také projeví v řetězci s přístupovými právy, a to na místech, kde obvykle bývá „x“ pro spuštění.

- SUID bit se projevuje písmenem „`s`“ nebo „`S`“ místo „`x`“ v první třetině řetězce,
- SGID bit se projevuje písmenem „`s`“ nebo „`S`“ místo „`x`“ v druhé třetině řetězce,
- Sticky bit se projevuje písmenem „`t`“ nebo „`T`“ místo „`x`“ v třetí třetině řetězce.

Ve všech třech případech malé písmeno znamená, že zároveň je přiřazeno i právo „`x`“, velké písmeno znamená, že právo „`x`“ není přiřazeno. V případě SUID a SGID bitu je velké písmeno chybou, protože bez oprávnění spouštět nemají tyto dva bity smysl (takže pozor – když vidíte velké „`S`“ v první nebo druhé třetině řetězce oprávnění, znamená to problém).

Příklad

Podíváme se na několik řetězců oprávnění s nastavenými speciálními příznaky.

`rwsrwx---` je nastaven SUID bit (písmeno „`s`“ v první třetině), vlastník a členové přiřazené skupiny mají právo čtení, zápisu a spouštění ⇒ proces vzniklý z tohoto spustitelného souboru si vezme přístupová oprávnění od vlastníka souboru, nikoliv od toho, kým byl spuštěn,

`rwSrw----` podobně, ale právo spouštění není nastaveno ani u vlastníka, ani u členů skupiny, tedy SUID bit se v praxi neuplatní (náprava chyby by spočívala buď v odstranění SUID bitu nebo v přiřazení práva spouštění vlastníkovi nebo případně také skupině),

`rwxrwsr--` je nastaven SGID bit (písmeno „`s`“ v druhé třetině), vlastník a členové skupiny mají právo čtení, zápisu a spuštění ⇒ pokud se jedná o spustitelný soubor, vzniklý proces bude mít jako efektivní skupinu tu, která je přiřazena souboru, nikoliv skupinu spouštějícího uživatele (zdědí skupinu od spustitelného souboru); pokud se jedná o adresář, pak položky v něm vytvořené zdědí skupinu od nadřízeného adreáře, nezíská ji od vytvářejícího uživatele,

`rwxrwSr--` opět chyba, je třeba buď zrušit SGID bit nebo nastavit oprávnění ke spuštění (pokud by se například jednalo o spustitelný soubor, nebylo by možné ho ani spustit pod danou skupinou, natož aby vznikl proces mající oprávnění dané skupiny),

Adresář:	**= 0**	**= 1**
SGID =	Když vytváříme nový soubor v tomto adresáři, přiřazenou skupinou se stává naše skupina.	Když vytváříme nový soubor v tomto adresáři, přiřazená skupina je zděděna od nadřízeného adresáře (toho s SGID bitem).
Sticky =	Pokud má uživatel k adresáři přiřazeno právo „`w`“, položky v adresáři může přidávat i mazat. Modifikovat položku (měnit obsah souboru) může jen tehdy, když má právo „`w`“ k této položce.	Pokud má uživatel k adresáři přiřazeno právo „`w`“, může přidávat (a případně modifikovat) položky, ale mazat je může pouze jejich vlastník nebo *root*.

Tabulka 4.3: Speciální práva pro adresáře

`rwxrwx--T` je nastaven Sticky bit (písmeno „`T`“ na konci), vlastník a členové skupiny mají všechna oprávnění, ostatní nemají žádná oprávnění (ani spuštění) – zde to není problém, Sticky bit nevyžaduje právo spuštění pro „zbytek světa“ ⇒ pokud v tomto adresáři chceme mazat soubory a podadresáře, musíme být jejich vlastníky nebo mít práva roota.

V grafickém rozhraní nastavujeme přístupová oprávnění přes kontextové menu daného souboru, položku *Vlastnosti*. V současné době ve většině desktopových prostředí nelze přímo nastavovat speciální příznaky v nástroji s grafickým rozhraním, ale v textovém režimu to jde (na postup se podíváme v příštím semestru). Obrázek na straně 69, na kterém je i možnost změny těchto příznaků, je staršího data.

Úkoly

1. Projděte si přístupová oprávnění (také vlastníka a skupinu) souborů a adresářů ve svém domovském adresáři. Srovnejte oprávnění u adresářů, dokumentů a spustitelných souborů.

   Vyberte si v domovském adresáři některý soubor a v kontextovém menu zvolte *Vlastnosti*. Najděte místo, kde se dají nastavovat přístupová oprávnění. Všimněte si, že oprávnění určujeme zvlášť pro vlastníka, skupinu a ostatní.

2. Rozkódujte tyto řetězce oprávnění:

   Soubory:
   - `rw-rw-r--`
   - `rwxr-x---`
   - `rwsr-xr-x`
   - `rwxr-S---`

   Adresáře:
   - `rwxr-x---`
   - `rwxrwS---`
   - `rwxrwsr--`
   - `rwxrwxr-t`

3. Ve své distribuci spusťte správce souborů a v nastavení zajistěte, aby se zobrazoval sloupec s oprávněními. Pravděpodobně se bude zobrazovat řetězec s 10 znaky místo 9, pak první z nich není oprávnění, ale určuje typ souboru (tj. oprávnění budou až od druhého znaku). Taky nechte zobrazit sloupec s vlastníkem a přiřazenou skupinou.
4. Zjistěte u příkazů v adresáři `/bin`, kdo bývá vlastníkem, jaká je přiřazená skupina a jaká jsou k nim přístupová oprávnění.
5. Najděte soubor `/usr/bin/passwd` a zjistěte, jaká oprávnění jsou k němu stanovena (včetně speciálních).

4.8.3 Atributy souborů

Každý soubor (míněno včetně adresářů a speciálních souborů) má atributy (vlastnosti), které ho plně určují (pozor – pod pojmem atribut zde rozumíme něco jiného než ve Windows). Tyto atributy v sobě zahrnují především vlastnosti související s identifikací a oprávněními souboru. Jsou to:

1. *typ* souboru – adresář, běžný soubor, symbolický odkaz, roura (pipe), apod.,
2. *mód* souboru – obsahuje nastavení přístupových oprávnění (číslo, ve kterém je zakódován řetězec, kterým jsme se zabývali v předchozím textu),

3. *počet pevných odkazů* na soubor,
4. *vlastník* (majitel) souboru (jeho UID),
5. *skupina* přiřazená souboru (její GID),
6. *velikost* souboru v B (pro adresář má vztah k počtu souborů v tomto adresáři),
7. *čas posledního přístupu* k souboru (nejen otevření souboru),
8. *čas poslední změny* obsahu souboru,
9. *čas poslední změny atributů* souboru (těchto atributů), mění se tedy například i při změně vlastníka nebo přístupových oprávnění.

První dvě položky probereme podrobněji, ostatní jsou zřejmé z předchozího textu.

Typ souboru: rozlišujeme

`d` adresář (directory)

`-` obyčejný soubor

`c` speciální soubor pro znakové zařízení (Character Device)

`b` speciální soubor pro blokové zařízení (Block Device)

`p` roura (Pipe) – lze vytvářet i pojmenované roury, které po svém použití nejsou zrušeny

`l` symbolický odkaz – link

`s` socket

Příklad

Typ souboru bývá ve výpisech uveden těsně před přístupovými oprávněními, například

- „`drwxr-xr-x`“ znamená, že jde o adresář (`d`), ke kterému má vlastník plná přístupová práva, a členům přidružené skupiny a ostatním uživatelům je dovoleno vypsat si obsah adresáře (`r`) a nastavit ho jako pracovní adresář (`x`),
- „`-rw-r--r--`“ je řetězec pro běžný soubor, jeho vlastník ho může číst i měnit, všichni ostatní pouze číst (pravděpodobně nepůjde o spustitelný soubor nebo spouštění není přímo dovoleno),
- „`lrwxr-xr-x`“ jde o symbolický odkaz (`l`).

Mód souboru je číslo složené ze čtyř číslic, první odpovídá speciálnímu příznaku souboru a ostatní tři běžným oprávněním. Tyto číslice dostaneme součtem určitých čísel (tj. vlastně pracujeme s binárním číslem, které pak převedeme na desítkové):

- první číslice:
  4 – set UID, nastaven příznak `SUID`
  2 – set GID, nastaven příznak `SGID`
  1 – set Sticky bit, nastaven příznak `Sticky`
- druhá číslice:
  4 – právo čtení pro majitele (r)
  2 – právo zápisu pro majitele (w)
  1 – právo spouštění pro majitele (x)
- třetí číslice: totéž pro skupinu
- čtvrtá číslice: totéž pro ostatní uživatele

Takže řetězec oprávnění rozložíme na tři části po třech znacích, v každé třetině si místo písmen dosadíme buď 0, pokud oprávnění není uděleno, nebo podle pozice v trojici $2^2/2^1/2^0$, sečteme a máme tři „hlavní“ číslice. Pokud je nastaven některý ze speciálních příznaků, pak před ně předsuneme ještě čtvrtou číslici vytvořenou podobně – pokud příznak není nastaven, dosadíme 0, jinak podle typu příznaku číslici $2^2/2^1/2^0$.

Příklad

Princip si ukážeme na několika příkladech:

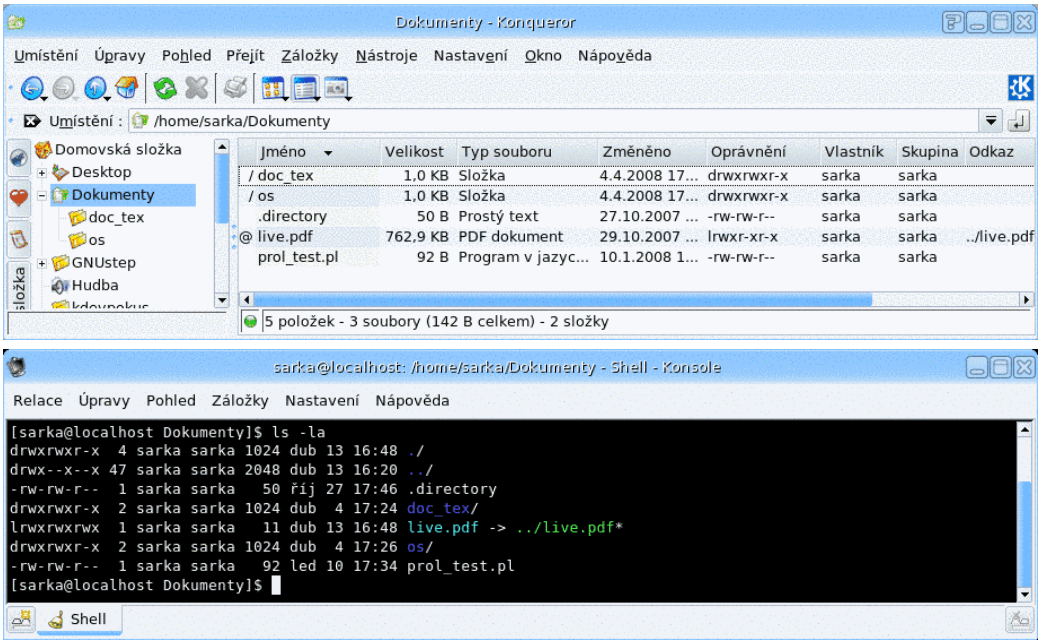
- řetězec „`rwxr-xr-x`“ s tím, že SUID, SGID i Sticky jsou 0, tento řetězec (včetně speciálních bitů, které umístíme zcela vlevo) je binárně $000\|111\|101\|101$, zakódujeme jako
  $(0+0+0)\|(1\cdot 2^2+1\cdot 2^1+1\cdot 2^0)\|(1\cdot 2^2+0\cdot 2^1+1\cdot 2^0)\|(1\cdot 2^2+0\cdot 2^1+1\cdot 2^0) =$
  $= (0+0+0)\|(4+2+1)\|(4+0+1)\|(4+0+1) = 0755$,
- řetězec „`rw-r-----`“ s tím, že SUID, SGID i Sticky jsou 0, je binárně (včetně speciálních bitů) $000\|110\|100\|000$, zakódujeme jako
  $(0+0+0)\|(4+2+0)\|(4+0+0)\|(0+0+0) = 0640$,
- řetězec „`rwxr-x---T`“ s nastaveným Sticky bitem bude zakódován následovně:
  $(0+0+1)\|(4+2+1)\|(4+0+1)\|(0+0+0) = 1750$,
- řetězec „`rwsr--r--`“ s nastaveným SUID bitem bude zakódován následovně:
  $(4+0+0)\|(4+2+1)\|(4+0+0)\|(4+0+0) = 4744$,
- mód 0710 znamená, že
  - ze speciálních bitů není nastaven žádný,
  - vlastník souboru má nastavena všechna přístupová práva, $4+2+1=7$, `rwx`,
  - skupina, jejíž GID je také v atributech souboru, má pouze právo spouštění (třeba programů), tj. $0+0+1=1$, `--x`,
  - ostatní nemají žádná přístupová práva ($0+0+0=0$, `---`),

  odpovídá to sekvenci „`rwx--x---`“,
- mód 2754 znamená, že
  - je nastaven SGID bit ($0+2+0=2$),
  - vlastník souboru má nastavena všechna přístupová práva, $4+2+1=7$, `rwx`,
  - skupina má pouze právo čtení a spouštění (zřejmě se jedná o adresář, tedy možnost nastavit adresář jako pracovní), tj. $4+0+1=5$, `r-x`,
  - ostatní mají právo čtení (zobrazit obsah adresáře, $4+0+0=4$, `r--`),

  odpovídá to sekvenci „`rwxr-sr--`“.

Na obrázku 4.10 vidíme výpis vlastností souborů v adresáři pro dokumenty uživatele, a to nejdřív ve správci souborů (Konqueror v KDE) a pak v terminálu. Oba výpisy jsou prakticky ekvivalentní, jen v grafickém režimu je vypnuto zobrazování pevných odkazů na nadřízený adresář a sebe sama („tečkové“ adresáře).

V okně Konqueroru vidíme z atributů hned za názvem velikost, čas poslední změny, přístupová oprávnění včetně typu souboru, vlastníka a skupinu. U symbolických odkazů se zobrazí také cílový soubor. V menu lze samozřejmě zvolit také zobrazování dalších atributů.



Obrázek 4.10: Výpis vlastností souborů v grafickém a textovém režimu

V okně terminálu jsme použili příkaz `ls -la`. Tento příkaz je obdobou příkazu `dir` ve Windows, vypisuje obsah adresáře. Parametry `-la` zajišťují „dlouhý“ výpis (*long*, včetně atributů), a to všech souborů (*all*, včetně skrytých). Oproti výpisu v Konqueroru máme navíc počet pevných odkazů na soubor (to číslo hned za řetězcem oprávnění), následuje vlastník, skupina, velikost a čas poslední změny (rok se zrovna nevypisuje, poslední změna tedy byla letos u všech souborů). Jako poslední údaj je uveden název souboru (je poslední, protože narozdíl od ostatních může být jakkoliv dlouhý a těžko se „zarovnává“ do sloupce).

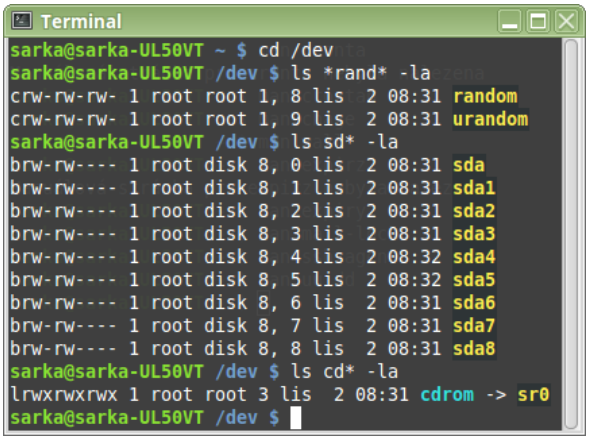
Jak vidíme, symbolický odkaz poznáme podle „`l`“ v na začátku řetězce oprávnění. Značit pevné odkazy podobným způsobem nemá smysl, protože jsou naprosto rovnocenné s jakoukoliv další cestou k souboru. Jejich existence se projevuje pouze v hodnotě čísla počtu pevných odkazů, které vidíme na obrázku z terminálu.

Příklad

Na výpisech jsme zatím viděli běžné soubory, adresáře a odkazy. Na obrázku 4.11 najdeme i jiné typy souborů (hvězdička zastupuje jakýkoliv řetězec, tedy například `sd*` zastupuje vše, co začíná podřetězcem `sd` a pak mohou následovat jakékoliv další znaky).

Jak víme, v adresáři `/dev` jsou speciální soubory zařízení. Zařízení jsou znaková nebo bloková, což se projeví také v typu speciálního souboru. Vidíme, že řádky pro zařízení `random` a `urandom` začínají písmenem „`c`“, jde o znaková zařízení (*character*) – virtuální vstupní zařízení generující náhodná čísla.

Speciální soubory začínající řetězcem `sd` náležejí (při použití udev) pevnému disku či jeho oddílům. Jak



Obrázek 4.11: Výpis vlastností některých speciálních souborů

vidíme, jde o bloková zařízení („b“ jako *block*), na disku jsou pouze tři oddíly. Poslední část výpisu obsahuje symbolický odkaz („l“) na speciální soubor nacházející se v tomtéž adresáři, k CD/DVD mechanice lze tedy přistupovat přes dva různé soubory, z nichž jeden je speciální (`/dev/sr0`), další je jen odkaz.

Určitě jste si všimli, že ve sloupci, kde by jinak byla velikost souboru, je u zařízení na obrázku 4.11 něco jiného – dvě čísla oddělená čárkou (srovnejte ve výpisu poslední řádek s předchozími). U speciálních zařízení obvykle nemá moc smysl vypisovat velikost souboru, ale zato je důležitý jiný údaj – čísla zařízení. Jádro UNIXových systémů nepracuje s názvy speciálních souborů, ale se zařízeními zachází pomocí *hlavního a vedlejšího čísla zařízení*.

Hlavní číslo označuje třídu zařízení. Svou třídu mají hlavní bloková zařízení (pevné disky, RAM disky apod.), další je pro vstupní zařízení (například myš), další mají síťová zařízení (síťové karty), atd., týká se to také virtuálních zařízení. Rozdělení do tříd zde odpovídá tomu, co jsme viděli v adresáři `/sys/class`. Vedlejší číslo pak odlišuje jednotlivá zařízení ve stejné třídě. Statický souborový systém pro zařízení devfs měl stanoveno, která hlavní čísla patří kterým zařízením, v dynamickém udev se hlavní i vedlejší čísla přiřazují dynamicky podle jejich připojování.

Na obrázku 4.11 vidíme, že v tomto konkrétním případě je pro virtuální zařízení (jako je třeba `/dev/random` hlavní číslo (tedy třída) 1, jednotlivá zobrazená zařízení mají vedlejší čísla 8 a 9. Pro paměťová média je hlavní číslo 8, vedlejší čísla mají jednotlivá zařízení různá.

Úkoly

1. Podle řetězce s přístupovými oprávněními `rwsr-xr-T` sestavte k němu ekvivalentní číselnou reprezentaci módu souboru.
2. Vytvořte řetězec s přístupovými oprávněními podle módu 1551.
3. Přiřaďte k řetězcům s přístupovými oprávněními k nim ekvivalentní číselnou reprezentaci módu souboru:

`rwxrwxr--`	`rwxrwsr-t`	`rw-rw-r--`	`rwxrwxr-x`	`rwsrwx---`	`rwxr-sr-x`

2755	0774	3775	0775	0664	4770

4. Prohlédněte si vlastnosti souborů a podadresářů v adresáři `/proc`. Z jakého důvodu mají všechny soubory nulovou velikost? Do kterých lze zapisovat a kdo zápis může provést? Co vyplývá z rozdílů ve vlastníkovi (a skupině) u podadresářů s číselným pojmenováním?
5. Spusťte si terminál (nebojte se, zatím si nemusíte pamatovat žádné příkazy). Najdete ho v hlavním menu prostředí. Pokud jste šťastně nalezli a spustili, vyzkoušejte tuto sekvenci příkazů:

   `cd ~` ..... přechod do domovského adresáře
   `touch pokus.txt` ..... vytvoříme soubor `pokus.txt`
   `ls po* -l` ..... podívejte se především na přístupová oprávnění
   `chmod u+s` ..... nastavíme SUID („u“ jako *user*, zde vlastník)

`ls po* -l` ..... jak se to projevilo v přístupových oprávněních?
`chmod u-s` ..... zrušíme SUID
`ls po* -l` ..... opět zkontrolujeme
`chmod g+s` ..... nastavíme SGID („g“ jako *group*, skupina)
`ls po* -l` ..... ověříme, jak se to odrazilo na přístupových oprávněních
`chmod +t` ..... nastavíme Sticky bit (nevztahuje se k vlastníkovi ani skupině)
`ls po* -l` ..... zkontrolujeme změnu v oprávněních
`rm pokus.txt` ..... smažeme soubor `pokus.txt`

Kapitola 5

Textový režim v Linuxu

Rychlý náhled: Tato kapitola je úvodem do správy operačních systémů s jádrem GNU/Linux (dále Linux) v textovém režimu, většina postupů s mírnou modifikací platí i pro jiné UNIXové (UNIX-like) systémy.

V UNIXových systémech včetně Linuxu je možné používat několik různých shellů. My se zde budeme věnovat pouze shellu `bash` – Bourne Again Shell.

Klíčová slova: Terminál, konzola, shell, příkaz, nápověda, manuálová stránka, info stránka, soubor, adresář, pevný odkaz, symbolický odkaz, přesměrování, filtr, proměnná, alias

Cíle studia: Po prostudování této kapitoly budete umět používat Linux v textovém rozhraní. Naučíte se základní příkazy pro práci se soubory a adresáři, proměnnými, aliasy.

5.1 Textové shelly a příkazy

5.1.1 Kdy, kde a jak používat textový shell

V UNIXových systémech včetně Linuxu máme vždy mimo grafické prostředí možnost provádět různé operace také v textovém režimu pomocí příkazů. Tento způsob práce se systémem se může zdát nepohodlný, ale je užitečný, pokud

- chceme operace provádět efektivně (nemusíme se dostávat do určitého okna s nastavením přes různá menu a jiná okna, máme všechno „pohromadě“, různá nastavení provádíme na jednom místě, v okně terminálu nebo konzoly),
- chceme automatizovat a zrychlit určité operace (pro často používané sledy příkazů napíšeme skript, ten vždy jen spustíme), hodí se také tehdy, když tatáž nastavení provádíme na více zařízeních,
- chceme, aby určité příkazy a programy spolupracovaly formou předávání vstupů a výstupů,
- nemůžeme najít v grafickém prostředí místo, kde se určitá věc nastavuje (v mnoha distribucích je grafické prostředí poněkud upraveno, navíc jsou výrazné rozdíly mezi KDE/GNOME/XFce/atd.),
- určitý program nepracuje v grafickém prostředí podle našich představ,
- máme alergii na grafická prostředí, atd.

Když zadáváme příkazy, pracujeme vždy v určitém shellu. *Shell* je příkazový interpret, je to především program (rozhraní), který slouží ke spouštění jiných programů většinou v textovém režimu. Pokud máme spuštěno grafické prostředí, dostaneme se do shellu spuštěním terminálu (závisí na používaném desktopovém prostředí) nebo spuštěním určitých programů, případně můžeme pracovat zcela mimo grafické prostředí (v textové konzoli). Reálně můžeme tedy v shellu pracovat takto:

- použijeme konzoli,
- použijeme některý program, který poskytuje rozhraní k shellu, který potřebujeme (terminál).

Původní shell v UNIXu byl *Bourne Shell* (vznikl roku 1978 a označoval se jednoduše `sh`), jehož autor je Stephen Bourne. O něco později (v době přepisu UNIXu do jazyka C) vznikl jako alternativa *C Shell* (označoval se `csh`). Zatímco Bourne Shell byl určen spíše pro psaní skriptů a umožňoval lepší spolupráci programů (včetně směrování), C Shell hodně inspirovaný jazykem C se orientoval hlavně na interaktivní práci přímo v příkazovém řádku (například používání historie pomocí speciálních příkazů) a také přinesl pokročilejší správu úloh.

V současné době se nesetkáme přímo s Bourne Shellem a C Shellem, ale spíše s jejich potomky:[1]

- *Toronto C Shell* (`tcsh`) je rozšíření `csh` o další možnosti, například používání šipek při práci s historií příkazů, `tcsh` také odstranil některé chyby související s činností ve skriptu, které se vyskytovaly v `csh`, je obvykle nainstalován v adresáři `/bin/tcsh` nebo `/usr/bin/tcsh` nebo `/usr/local/bin/tcsh` (záleží na distribuci),
- *Korn Shell* (`ksh`) – syntaxe příkazů odpovídá Bourne Shellu, ale jinak většinu vlastností přejal z `tcsh`, jde vlastně o hybrid shellů `sh` a `tcsh`, je nainstalován v adresáři `/bin/ksh` nebo `/usr/bin/ksh`,
- *Bourne Again Shell* (`bash`) je nejobvyklejším shellem v Linuxu a je velmi podobný `ksh` (je jednodušší, například oproti `ksh` neobsahuje podporu racionálních čísel a vícedimenzionálních polí), je nainstalován v `/bin/bash` nebo `/usr/bin/bash` nebo `/usr/local/bin/bash`,
- *Debian Almquist Shell* (`dash`) je potomkem shellu `ash` (Almquist Shell) ze systému FreeBSD, jak název napovídá, můžeme se s ním setkat u Debianu a jeho potomků (včetně Ubuntu); je podobný shellu `bash`, ale mírně osekaný (některé vlastnosti shellu `bash` nepodporuje) a rychlejší,
- *Z Shell* (`zsh`) je naopak vybavenější než `bash`, a to směrem k vědeckým výpočtům (je srovnatelný spíše s shellem `ksh`).

V Linuxu je vždy (nebo téměř vždy) nainstalován shell `bash`, a kromě něho i několik dalších. Seznam použitelných shellů (těch, které máme k dispozici) je v souboru `/etc/shells`.

Poznámka:

Vzpomeňte si na soubor `/etc/passwd` (seznam uživatelů) – v posledním „sloupci“ je výchozí shell pro daného uživatele. U „lidských“ uživatelů to v Linuxu bývá `/bin/bash`.

5.1.2 Konzoly

I tehdy, když funguje pouze textový režim a tedy nepoužíváme žádného správce oken (vlastně ani žádná okna), máme k dispozici standardně 6 až 8 konzol. Mezi nimi se přepínáme stiskem klávesových

[1] Pěkné porovnání shellů (dokonce včetně Příkazového řádku ve Windows) najdeme například na stránce http://en.wikipedia.org/wiki/Comparison_of_command_shells.

kombinací Alt+F1 (první konzola), Alt+F2 (druhá), ..., Alt+F6 (šestá konzola). V každé konzole můžeme mít spuštěn nějaký program na popředí a více programů na pozadí. V kterékoliv konzole můžeme mít také spuštěn program Midnight Commander (spouští se příkazem `mc`, pokud je nainstalován) pro snadnější práci se soubory, sítí a dalšími možnostmi.

Z grafického režimu se do konzol dostaneme tak, že s příslušnou klávesovou zkratkou stiskneme také klávesu Ctrl, tj. celkově Ctrl+Alt+F1 až Ctrl+Alt+F6 (pak při přímém přepínání mezi konzolami již Ctrl netiskneme). Zpět do grafického režimu se dostaneme klávesovou zkratkou Alt+F7 (grafická prostředí běží na poslední konzole, tedy pokud je jich celkem sedm, tak sedmé). Pokud máme více konzol, pak číslo pro horní hranici samozřejmě patřičně posuneme.

Úkoly

1. Spusťte některý terminál, emulátor konzole nebo se přesuňte do některé z konzolí (tam se přihlašte).
2. Vyzkoušejte si přepínání mezi konzolami – přepněte se postupně na první konzolu, pak druhou, případně další, potom se přesuňte zpět do grafického prostředí na poslední konzolu.
3. Pokud se vám podaří se v některé konzole přihlásit, vyzkoušejte program `mc` (pokud je instalován) a projděte si v něm svůj domovský adresář.

5.1.3 Struktura příkazů

Příkazy se skládají z těchto částí:

- *název příkazu* nebo případně název spustitelného souboru, který tady používáme jako příkaz, může obsahovat i cestu, v opačném případě je hledán v adresářích uvedených v příslušné proměnné prostředí (nehledá se v pracovním adresáři, proto když se právě tam nachází, píšeme `./název příkazu`),
- *volby* (přepínače) začínají obvykle znakem pomlčka, ve většině UNIXových systémů je můžeme „shrnout“ za jedinou pomlčku (bez mezer samozřejmě), pokud jsou jednopísmenné (volby delší než jedno písmeno musíme psát každou zvlášť, každá z nich bude mít vlastní pomlčku), volby obvykle řídí a upřesňují příkaz; existují i „vícepísmenné“ volby (například velmi užitečná je volba `-{}-help`), aby se odlišily od jednopísmenných, u některých příkazů začínají dvěma pomlčkami,
- *argumenty* nezačínají pomlčkou, obvykle příkazu říkají, se kterými daty má pracovat, například názvy souborů.

Například u příkazu

```
ls -la nejaky_adr
```

je názvem řetězec `ls`, volby jsou `l`, `a` (jednopísmenné, proto je můžeme napsat za jedinou pomlčku), argument je adresář `nejaky_adr`.

Interaktivní příkazy (komunikující s uživatelem, například filtry pro stránkování v textu) často pro své ukončení vyžadují stisknutí klávesy q. Terminál lze ukončit příkazem `exit` nebo případně zavřením okna. Zamrzlý nebo příliš dlouho pracující program ukončíme klávesovou zkratkou Ctrl+C.

Poznámka:

UNIXové systémy jsou case-sensitive, tedy rozlišují malá a velká písmenka, na což je třeba dbát nejen u zadávání názvů příkazů, ale i u názvů souborů a adresářů.

5.2 Zástupné znaky

Také v UNIXu používáme při práci se soubory a adresáři zástupné znaky, máme jich dokonce více než ve Windows. Jsou to:

`*` – jakýkoliv počet kterýchkoliv znaků, zde navíc bereme v úvahu, že přípona souboru a tečka před ní jsou součástí názvu souboru, a tedy `*` může reprezentovat i tečku,

`?` – kterýkoliv znak, a to právě jeden, včetně tečky v názvu,

`[znaky]` – jeden ze znaků uvedených v seznamu, například `[ABa]` znamená, že na to místo lze dosadit jeden ze znaků A, B, a; pokud chceme reprezentovat posloupnost znaků, napíšeme: `[A-Z]` znamená všechna velká písmena, lze kombinovat: `[A-Za-z0-9]` znamená písmeno nebo číslici,

`[!znaky]` – odpovídá jednomu znaku, který se liší od všech uvedených v závorce, například `[!a-kA-K]` znamená kterýkoliv znak kromě malých a velkých písmen v rozmezí od A do K, například písmeno M nebo některá číslice podmínku splňují,

`~` – domovský adresář, navíc sekvence `~uzivatel` (bez mezery) je odkaz na domovský adresář uživatele s přihlašovacím jménem uzivatel, ať se nachází kdekoliv.

5.3 Nápověda

5.3.1 Jak volat o pomoc

Nápovědu k příkazům můžeme získat více způsoby:

- zobrazením manuálové stránky příkazu, a to příkazem `man`, takto lze buď zobrazit nápovědu ke konkrétnímu příkazu nebo pomocí klíčového slova zjistit, jak se hledaný příkaz nazývá,
- někdy je implementován příkaz `apropos`, který použijeme, když nevíme, jak se příkaz nazývá,
- příkaz `whatis`[2] naopak použijeme, když jsme narazili na příkaz (spustitelný soubor), ale nevíme, co provádí (vypíše se krátká informace o příkazu),
- příkaz `info` vypíše krátkou informaci o příkazu,
- v grafickém režimu také mohou být stránky nápovědy,
- existují *dokumenty HOWTO* („jak na to“), a to buď přímo v jednotlivých distribucích nebo na internetu, obsahují přímo rady, jak postupovat v určitých situacích,
- také máme *dokumenty FAQ* (Frequently Asked Questions) pro často pokládané otázky,
- ve webovém prohlížeči stačí zadat podobný výraz jako když v textovém režimu hledáme manuálové stránky, například při zadání `man chsh` získáme odkazy na manuálovou stránku příkazu `chsh` na některém serveru přímo určeném pro manuálové stránky.

[2] Příkaz `whatis` používá vlastní databázi o příkazech. Tuto databázi je dobré občas aktualizovat, například po větší aktualizaci systému, což se provede příkazem `makewhatis`.

Další informace:

Na internetu je hodně stránek věnovaných shellu `bash`, například:

- http://www.gnu.org/software/bash
- http://tldp.rog/LDP/abs/html (skripty)
- http://www.abclinuxu.cz/clanky/show/46130

5.3.2 Manuálové stránky

Nápověda k příkazům, konfiguračním souborům, skriptům a funkcím je v UNIXových systémech standardně k nalezení především v manuálových stránkách.

Příkaz `man` je zkratkou z anglického slova „manual“ a je obdobou windowsovského příkazu `help`, i když je mnohem komplexnější. Základní tvar příkazu je následující:

`man příkaz` vypíše manuál (nápovědu) k zadanému příkazu

Nápovědu k příkazu `man` můžeme zobrazit příkazem `man man`.

Manuály k jednotlivým příkazům (ve skutečnosti existují i pro některé konfigurační soubory a další věci) jsou členěny do částí, každá část má určitý význam. Manuálová stránka obvykle obsahuje části uvedené v tabulce 5.1 (některé se nemusí vyskytovat, naopak jiné mohou být navíc).

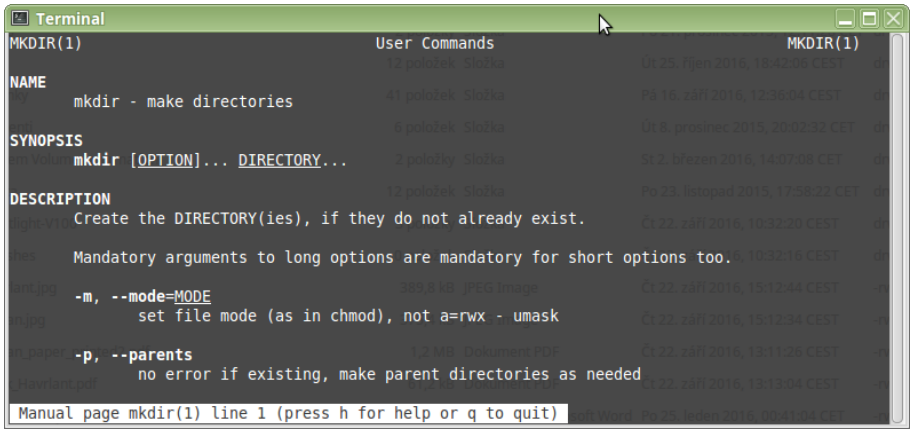
Část	Význam
NAME	Jméno příkazu
SYNOPSIS	Syntaxe příkazu, nepovinné parametry jsou uzavřeny do hranatých závorek, položky, mezi kterými si můžeme vybrat, jsou odděleny symbolem „|“, položky, které se mohou opakovat, jsou ukončeny třemi tečkami
DESCRIPTION	Detailní popis příkazu
FILES	Seznam systémových souborů, ke kterým má příkaz nějaký vztah (například je nějakým způsobem upravuje)
OPTIONS	Význam jednotlivých parametrů
EXAMPLES	Příklady použití příkazu
SEE ALSO	Odkazy na další manuálové stránky, ke kterým má tento příkaz nějaký vztah
DIAGNOSTICS	Význam chybových hlášení a návratové kódy příkazu
BUGS	Popis neočekávaného chování příkazu nebo kontakt, na který lze zaslat informaci o chybném chování programu
COPYRIGHT	licence, pod kterou je příkaz šířen
SEE ALSO	odkazy na manuálové stránky souvisejících příkazů nebo souborů

Tabulka 5.1: Části manuálových stránek

Příklad

Podíváme se na strukturu jedné z manuálových stránek. Otevře se automaticky pomocí „stránkovače“ včetně zobrazení formátování (to zde neuvidíme).

Po zadání příkazu `man mkdir` se nám zobrazí tato manuálová stránka (viz obrázek 5.1):



Obrázek 5.1: Výstup příkazu `man mkdir`

```
MKDIR(1)                          User Commands                          MKDIR(1)
NAME
  mkdir - make directories
SYNOPSIS
  mkdir [OPTION] ... DIRECTORY ...
DESCRIPTION
  Create the DIRECTORY(ies), if they do not already exist.

  Mandatory arguments to long options are mandatory for short options too.

  -m, --mode=MODE
    set file mode (as in chmod), not a=rwx - umask
...
GNU coreutils 8.21                   March 2016                          MKDIR(1)
```

Stránka byla zkrácena.

Manuálové stránky jsou sdružovány do *sekcí*, sekce obvyklé v Linuxu jsou v tabulce 5.2. Každá sekce sdružuje příkazy (ale také nápovědu ke konfiguračním souborům a funkcím), které mají něco společného, například sekce 1 obsahuje příkazy, které může používat i běžný uživatel, naproti tomu sekce 8 obsahuje příkazy určené pro administrátora (například roota). Užitečná (i pro běžného uživatele) je také sekce 5 obsahující formát konfiguračních souborů.

Příkaz `man -s n příkaz` vypíše manuálovou stránku příkazu v sekci číslo n (tentýž příkaz může být ve více sekcích, v každé je trochu obsah manuálové stránky), bez tohoto parametru vypíše první nalezenou manuálovou stránku. Pro tentýž příkaz může být víc manuálových stránek (zvlášť pro běžné uživatele a adminy), případně stejně se může jmenovat jak program, tak i konfigurační soubor (např. `passwd`).

Příkaz `man -f příkaz` vypíše seznam manuálových stránek příkazu (ze všech sekcí). Pozor, pořadí není podle čísel sekcí, například stránky ze sekce 5 mají přednost před stránkami ze sekce 1.

Úkoly

1. Zobrazte obsah manuálové stránky příkazu `mkdir` tak, jak je naznačeno v příkladu. Všimněte si prvního řádku stránky – hned tam máme informaci o tom, že stránka patří do sekce 1 (tj. příkazy pro běžné uživatele, uprostřed řádku to máme i slovně – „User Commands“). V části stránky

Číslo sekce	Obsah
0	Hlavičkové soubory (jazyka C), obvykle jsou k v adresáři `/usr/include`
1	Příkazy a systémové programy, které může používat i běžný uživatel
2	Služby jádra (volání jádra, systémová volání) a chybové (návratové) kódy
3	Dokumentace k systémovým knihovnám (dostupné funkce obsažené v knihovnách)
4	Speciální soubory v `/dev`
5	Formát konfiguračních souborů (včetně `/etc/fstab`)
6	Hry
7	Balíky maker, popis souborového systému, manuálů atd., nástroje pro práci s textem
8	Správa systému – příkazy a nástroje používané rootem
9	Dokumentace k jádru
n	Nezařazené stránky

Tabulka 5.2: Sekce manuálových stránek

NAME je stručný popis příkazu (vytváření adresářů), v části SYNOPSIS se dozvíme o pořadí zadávaných parametrů – nejdřív přepínače, pak až řetězcové parametry (v tomto případě třeba název adresáře, který chceme vytvořit). Z toho vyplývá, že:

- `mkdir -v novy` je syntakticky správně,
- `mkdir novy -v` je syntakticky špatně (ale konkrétně tento příkaz by se s tím vyrovnal).

Pokud je v popisu syntaxe parametr obklopen hranatými závorkami, znamená to, že je nepovinný (například u příkazu `mkdir` nemusíme zadávat žádné přepínače, stačí název adresáře, který chceme vytvořit).

Z interaktivního režimu příkazu `man` odejděte stiskem klávesy q.

2. Příkazem `man -f passwd` si ověřte, které manuálové stránky se takto nazývají. Všimněte si, že existují minimálně dvě (v některých distribucích jich může být více): v sekci 1 je stránka pro příkaz `passwd` (příkaz pro změnu hesla – `/usr/bin/passwd`), v sekci 5 je stránka popisující konfigurační soubor `/etc/passwd` se seznamem uživatelů.

   Proveďte příkaz `man passwd`. Která z těchto dvou stránek se zobrazila a je tedy „výchozí“? (Poznáte to podle prvního řádku, je tam číslo sekce). Zobrazte pak tu stránku, která není výchozí.

5.3.3 Znám příkaz, chci vědět, k čemu slouží

Pokud známe název příkazu a chceme zjistit, k čemu slouží a případně také jak se používá (parametry apod.), použijeme jeden z následujících příkazů:

`man příkaz` zobrazí se manuálová stránka příkazu v programu `man`, který zajistí zformátování stránky a pomocí stránkovače i možnost interaktivně se na stránce posouvat,

`info příkaz` jedná se o dokumentaci *GNU Info*, což je hypertextový systém, ale pokud pro daný příkaz neexistuje info stránka, zobrazí se manuálová stránka,

`whatis příkaz` takto zjistíme, ve kterých manuálových stránkách jsou k nalezení podrobnější informace o příkazu, zobrazí se také krátký popis příkazu; Totéž provádí příkaz `man -f příkaz`.

Příklad

Po zadání příkazu `man chsh` získáme výstup v podobě, kterou vidíme na obrázku 5.1 na straně 83 (jen pro jiný příkaz), zobrazila se manuálová stránka příkazu `chsh`.

Pokud zadáme `info chsh`, ve skutečnosti získáme něco velmi podobného (naprosto stejný obsah, jen trochu jinak formátovaný), protože pro příkaz `chsh` neexistuje samostatná info stránka a bere se manuálová stránka.

Jiná situace je však u používanějších příkazů, jako je například příkaz `ls` (to je obdoba příkazu `dir` ve Windows):

- jestliže zadáme `man ls`, získáme běžnou manuálovou stránku, která se především soustředí na vysvětlení funkce jednotlivých parametrů,
- po zadání `info ls` se zobrazí něco jiného – info stránka soustřeďující se především na vysvětlení funkce samotného programu, oproti manuálové stránce je poněkud „upovídanější“,
- když zadáme `whatis ls` (nebo `man -f ls`, získáme pouze krátkou informaci o určení příkazu, jde naopak o velmi stručnou informaci.

Manuálová stránka příkazu `ls` (`man ls`) začíná takto (ve skutečnosti je mnohem delší):

```
LS(1)                          User commands                          LS(1)
NAME
  ls - list directory contents
SYNOPSIS
  ls [OPTION]... [FILE]...
DESCRIPTION
  List information about the FILEs (the current directory by default). Sort entries
  alphabetically if none of -cftuvSUX nor --sort is specified.
  Mandatory arguments to long options are mandatory for short options too.

  -a, --all
      do not ignore entries starting with .
  -A, --almost-all
      do not list implied . and ..
```

Zjistili jsme, že jde o příkaz, který vypisuje obsah adresáře, dále že v syntaxi jsou hned za názvem příkazu volby (options, přepínače) a pak následují názvy souborů. Souborů? No ano, přece víme, že v UNIXových systémech platí *všechno je soubor*, včetně adresářů. Ze stručného popisu (description) se dále dovíme, že pokud neuvedeme žádný soubor, vypíše se obsah pracovního adresáře, položky jsou řazeny abecedně (pokud neurčíme jinak přepínači).

Následuje popis voleb (options). Je jich poměrně hodně, proto jsme výpis „usekli“. Zde na první pohled vidíme rozdíl mezi použitím malého a velkého písmene – pokud napíšeme „`-a`“, budou vypisovány také položky začínající tečkou (z předchozího semestru již víme, že jde o skryté soubory), pokud však napíšeme „`-A`“, chování bude zdánlivě podobné (také budou vypisovány skryté soubory), ale nevypíšou se dva adresáře, které jsou také skryté (odkaz na nadřízený adresář a odkaz na sebe sama).

Info stránka příkazu `ls` (`info ls`) začíná takto:

```
File> coreutils.info, Node: ls invocation, Next: dir invocation,
Up: Directory listing

10.1 'ls': List directory contents
==================================
The 'ls' program lists information about files (of any type, including
directories). Options and file arguments can be intermixed
```

```
arbitrarily, as usual.

For non-option command-line arguments that are directories, by
default ‘ls’ lists the contents of directories, not recursively, and
ommitting files with names beginning with ‘.’. Fot other non-option
arguments, by default ‘ls’ lists just the file name. If no non-option
argument is specified, ‘ls’ operates on the current directory, acting
as if it had been invoked with a˜single argument of ‘.’.
...
```

V záhlaví info stránky se především dozvíme název info souboru, který právě čteme a jeho pozici ve struktuře (info stránky tvoří stromovou strukturu). Následujícím uzlem ve stromě je příkaz `dir` (platí pro OpenSUSE, obecně tento příkaz není v Linuxu ani jiných UNIXových systémech podporován), nadřízeným uzlem je skupina příkazů pro výpis adresářů.

Dále zjistíme něco podobného jako v manuálové stránce, tedy že příkaz vypisuje informace o nejrůznějších souborech a je upřesněno, o jaké informace se vlastně jedná. Výchozím chováním je výpis obsahu adresářů, a to ne rekurzívně a bez skrytých souborů, standardně pouze jména souborů.

Pro doplnění – výstupem příkazu `whatis ls` je

```
ls(1)        - list directory contents
```

Tento výpis je velmi krátký a jeho účelem je usnadnit orientaci – narazíme na příkaz (nebo si na některý příkaz matně vzpomeneme) a chceme vědět, o co se vlastně jedná. Zjistili jsme, že tento příkaz je v sekci 1 (v manuálových stránkách) a slouží k výpisu obsahu adresáře.

5.3.4 Neznám příkaz

Pokud neznáme název příkazu, ale napadají nás klíčová slova související s příkazem (slova, která by se mohla vyskytovat na manuálové stránce, nejlépe někde na jejím začátku ve stručném popisu příkazu), můžeme použít některý z následujících postupů, podle toho, který příkaz je nainstalován.

`man -k řetězec` výpis všech dostupných témat, ve kterých se nachází zadaný řetězec

`apropos řetězec` totéž

`apropos -s` *`číslo`* `řetězec` určili jsme, že mají být prohledány pouze manuálové stránky v sekci zadané číslem

Příklad

Nemůžeme si vzpomenout na název příkazu, který vypisuje obsah adresáře. Vyzkoušíme několik možností:

`man -k directory` získáme seznam příkazů, které nějak souvisejí s adresáři, ale je jich opravdu hodně (několik stránek), takže vyzkoušíme něco jiného,

`man -k "directory contents"` tak to je lepší, vypsalo se už jen pár příkazů, mezi kterými si určitě vybereme,

`apropos "directory contents"` totéž,

`apropos -s 1 "directory contents"` ještě lepší – vypsaly se pouze příkazy patřící do sekce 1

Výsledný výpis:

```
dir (1)          - list directory contents
ls (1)           - list directory contents
vdir (1)         - list directory contents
```

(v různých distribucích Linuxu se výpis bude lišit).

 Užitečná může být také klávesa Tab. Pokud víme, jak příkaz začíná, ale nevíme jistě, jak dále pokračuje (angličtina :-), napíšeme začátek příkazu (nebo třeba názvu souboru, pro ty to taky funguje) a stiskneme klávesu Tab.

Příklad

Vyzkoušíme automatické doplňování příkazů pomocí klávesy Tab. Víme, že příkaz začíná pravděpodobně písmeny „`wh`“, ale nevíme, jak dál. Napíšeme

`wh` Tab

Zobrazí se:

```
whatis    whereis    which    while    who    whoami    whois
```

Úkoly

1. Zobrazte manuálovou stránku příkazu `ls`. Zjistěte, k čemu se tento příkaz používá, jaké lze zadat přepínače a podívejte se, pod jakou licencí je příkaz šířen. Všimněte si, že za názvem příkazu v záhlaví stránky je řetězec `(1)`, což znamená, že pracujete se stránkou ze sekce 1.
2. Vyzkoušejte ve svém domovském adresáři postupně příkazy

   ```
   ls
   ls -a
   ls -la
   ```

3. Zobrazte manuálovou stránku příkazu `mount`. Všimněte si čísla sekce, už to není 1, ale 8. Jaké typy příkazů patří do sekce 8? Tento příkaz slouží k připojování paměťových médií. Jak vidíme, manuálová stránka je velmi dlouhá (všimněte si, že jsou odlišné parametry pro různé typy souborových systémů). Na konci stránky se také podívejte, na které další stránky s podobnými tématy je odkazováno.
4. Zjistěte, ve kterých manuálových stránkách se píše o souboru `fstab` (pozor, nezajímají nás pouze stránky s tímto názvem, ale obecněji stránky, na kterých se tento řetězec vyskytuje).

5.4 Práce s adresáři a soubory

5.4.1 Adresáře

 Používané příkazy jsou

`pwd`

výpis názvu pracovního adresáře (zkratka z „print working directory“)

`cd adresář`

změna aktivního adresáře (bez argumentu: přesun do domovského adresáře)

`cd podrizeny` přesun do podřízeného adresáře `podrizeny`

`cd ..` přesun do nadřízeného adresáře (pozor, před tečkami by měla být mezera)

`cd ../..` přesun o dva adresáře nahoru, všimněte si, že pro oddělení adresářů na cestě (zde na cestě nahoru) používáme obyčejné lomítko, nikoliv opačné

`cd ~` (vlnka) přesun do našeho domovského adresáře (například pokud jsme přihlášeni jako uživatel `hanicka`, takto se přesuneme do adresáře `/home/hanicka`)

`cd /home/novak` přesun do domovského adresáře uživatele `novak`

`cd ~novak` totéž co předchozí, také se přesuneme do domovského adresáře uživatele `novak`

`mkdir adresář`

vytvoření nového adresáře

`rmdir adresář`

zrušení *prázdného* adresáře (předem musíme z adresáře odstranit všechny položky)

`ls [`*`volby`*`] [soubor]`

výpis obsahu adresáře (adresářů), některé volby:

`-l` provádí dlouhý výpis (i s atributy včetně práv a vlastníka),

`-a` vypíše také skryté soubory (které začínají tečkou, včetně `.` a `..`),

`--sort=size` seřadí položky podle velikosti místo podle názvu, také lze seřadit například podle času (`time`), přípony (`extension`) apod.,

`-r` seřadí v opačném pořadí (reverse),

`-R` provede příkaz rekurzívně (recursively), atd. (parametrů je velmi mnoho)

Příklad

Pro `ls -l` může být výstup

```
-rwxr-x---  2  novakj  elita  254136  Oct  12  2000  sezn.txt
-rwxr-x---  1  novakj  elita  254136  Jan  30  2004  moje poznamky
drwxr-xr--  1  novakj  users     150  Feb  14  1997  adr/
```

nebo třeba

```
drwxr-xr-x  4  uzivatel  users      4096  úno   9  00:10  Desktop/
drwxr-xr-x  4  uzivatel  users      4096  lis  28  22:31  Documents/
-rw-r--r--  1  uzivatel  users       150  lis  29  14:23  Pevný disk C
-rw-rw-r--  1  uzivatel  users  32223208  úno   2  00:23  pvysl.pdf
```

Na začátku řádku je typ souboru, přístupová oprávnění, následuje počet pevných odkazů na soubor, vlastník, skupina, velikost, čas poslední změny a konečně název souboru.

`echo` *`maska`*

příkaz `echo` je sice vnímán jako prostředek, který prostě na obrazovku vypíše to, co má jako argument, ale ve skutečnosti ho můžeme použít i pro výpis všech položek v adresáři odpovídajících zadané masce, takže tak trochu může „suplovat“ příkaz `ls`

Příklad

Předpokládejme, že v domovském adresáři máme několik položek začínajících písmenem „V“. Srovnejme výstup těchto svou příkazů:

- `echo V*`
  - vypíše se jednoduše seznam všech souborů či adresářů začínajících písmenem „V“, jednu položku za druhou, oddělené mezerou
- `ls V*`
  - u běžných souborů reaguje tento příkaz očekávaně (vypíše jejich seznam), ale pokud narazí na adresář vyhovující zadání (začíná písmenem „V“), vypíše k němu seznam vnořených položek (ale hlouběji ve struktuře nejde)
- `ls -R V*`
  - oproti předchozímu vypisuje u nalezených adresářů nejen jejich obsah, ale v rekurzi pokračuje i dále

Úkoly

1. Pokud nejste ve svém pracovním adresáři, přejděte do něj. Vytvořte podadresář `test`.
2. Přejděte do adresáře `test` a pro jistotu vypište název pracovního adresáře, abyste si ověřili, kde konkrétně v adresářové struktuře jste. Vytvořte podadresář `vnoreny`.
3. Přesuňte se do svého domovského adresáře a vypište jeho obsah tak, aby se vypsaly i skryté soubory a aby byl použit dlouhý (široký) výpis (včetně vlastností položek). Vyzkoušejte také rekurzi, porovnejte výpis těchto příkazů:
   - `ls -lR`
   - `ls -aR`
   - `ls *`
   - `ls -l *`
   - `ls -a *`
4. Smažte adresáře, které jste vytvořili v předchozích úkolech – `test` a `vnoreny`. Nezapomeňte, že zatím umíme smazat pouze prázdný adresář.
5. Vyzkoušejte příkazy uvedené v příkladu před těmito úkoly – srovnejte příkazy `echo` a `ls`.

5.4.2 Soubory

Protože v UNIXových souborech platí, že vše je soubor, téměř všechny dále uvedené příkazy lze použít také na adresáře.

`file soubor(-y)`

pokusí se zjistit vnitřní formát souboru (jestli se jedná o adersář, textový soubor pro určitou znakovou sadu, PNG nebo jiný obrázek, PDF, ..., příkaz je samozřejmě použitelný i na soubory bez přípony nebo s „nic neříkající“ příponou, pracuje především s obsahem souboru (to je velmi důležité, proto na to znovu upozorníme: v Linuxu není ani tak důležitá přípona, jako spíše skutečný obsah souboru, což je rozhodně bezpečnější)

`type soubor(-y)`

taky slouží ke zjištění typu souboru podle jeho obsahu, každý z těchto příkazů může mít u některých souborů trochu jiný výstup (oba příkazy si zapamatujeme tak, že si řekneme, co chceme: file type)

`cp soubor1 soubor2`

kopírování prvního souboru do druhého

`cp -a ~/Documents ~/zaloha` provede archivaci zvoleného adresáře, tedy kopíruje rekurzívně se zachováním atributů souborů (například vlastník nebo časové údaje)

`cp -uR ~/Documents ~/zaloha` provede rekurzivní update (jen soubory, které se od poslední zálohy změnily)

`cp -sR *.png ./obrazkyPNG` v podadresáři `obrazkyPNG` pracovního adresáře vytvoří symbolické odkazy na všechny soubory s příponou PNG

`mv s1 s2`

přejmenování nebo přesun souboru `s1` na `s2` (takto můžeme přejmenovávat či přesouvat i adresáře, nejen běžné soubory)

`rm soubor`

odstranění souboru

`rm soubor` odstraní zadaný soubor

`rm -fr adresář` rekurzívně odstraní vše v zadaném adresáři, a to bez upozorňování (force)

`cat soubor`

vypsání obsahu textového souboru; zobrazí obsah zadaného souboru (obdoba `type` ve Windows, ale `cat` má ve skutečnosti výrazně více možností použití – budeme se učit v příštím semestru)

`head -n soubor`

vypíše prvních n řádků souboru, tedy jakési „záhlaví“, podle začátku můžeme orientačně odhadnout zbývající obsah souboru

`head /etc/sensors.conf` zajímá nás, k čemu je asi tento soubor; některé konfigurační soubory mají na svém začátku v poznámce stručný popis, ten takto zobrazíme

`tail -n soubor`

vypíše posledních n řádků souboru, používá se při procházení log souborů, kde jsou nové záznamy přidávány právě na konec:

`tail -8 /var/vysledky.log` zobrazí posledních 8 řádků zadaného souboru

`cmp s1 s2`

porovnání souborů `s1` a `s2`, po nalezení rozdílu se běh ukončí

`diff s1 s2`

porovnání souborů, oproti předchozímu příkazu tento prochází celé soubory a pokouší se najít odlišnosti

`diff -iEBw soubor1 soubor2` díky všem čtyřem použitým parametrům jsou ignorovány odlišnosti ve velkých/malých písmenech, tabulátorech, prázdných řádcích a mezerách

`touch soubor`

soubor neotevře, ale změní datum a čas posledního přístupu na aktuální údaje („dotkne se“ souboru), pokud soubor neexistuje, vytvoří ho

Ve většině příkazů můžeme zadat i více souborů, a to třeba pomocí masky. Pak se provedou požadované operace pro všechny tyto soubory.

Úkoly

1. Vyberte si některý dostatečně velký textový soubor ze svého domovského adresáře. Nejdřív ho celý vypište, pak vypište prvních 5 řádků, a pak posledních 5 řádků.
2. Pak tento soubor zkopírujte do téhož adresáře, ale se jménem `pomocny`. Následně takto vytvořený soubor smažte.
3. Vyzkoušejte ve svém domovském adresáři vytvoření nového souboru příkazem `touch`.

5.4.3 Pevné a symbolické odkazy

 Z předchozího semestru víme, jaký je rozdíl mezi pevnými a symbolickými odkazy na soubor. Teď si ukážeme, jak se s těmito odkazy pracuje v textovém režimu.

`ln zdroj [cíl | cílový_adresář]`

vytvoření *pevného odkazu* `cíl` na zdroj (nebo v cílovém adresáři), ve výpisu pomocí `ls` se vytvořený odkaz projeví zvýšením čísla počtu pevných odkazů za přístupovými právy souboru

`ln /adr1/adr2/soub.pdf .` vytvoří pevný odkaz na zadaný soubor v pracovním adresáři (to je ta tečka na konci)

`ln prvni.txt druhy.txt treti.txt /home/uzivatel` vytvoří pevné odkazy na všechny zadané soubory v domovském adresáři uživatele s přihlašovacím jménem `uzivatel`, vytvořené pevné odkazy jsou v pracovním adresáři

`ln prvni.txt druhy.txt treti.txt ~` provede totéž (je to jiný způsob adresace domovského adresáře tohoto uživatele)

`ln -s puvodni [cil]`

vytvoření *symbolického odkazu*, ve výpisu pomocí `ls` se tento odkaz projeví typem souboru `l`

`ln -s /etc/X11/xinit/xinitrc.xfce ~/.xinitrc` změní desktopové prostředí, které je spouštěné automaticky po startu X Window na XFce (soubor – skript `.xinitrc` v domovském adresáři uživatele určuje, který soubor spouští desktopové prostředí nebo správce oken pro tohoto uživatele, soubor `xinitrc.xfce` je spouštěcí soubor pro XFce, případně to může být textový soubor, ve kterém je spouštěcí příkaz pro toto prostředí či správce oken)

Úkoly

1. Vytvořte ve svém domovském adresáři, v podadresáři `documents` (nebo `dokumenty`) symbolický odkaz na soubor `/etc/fstab` (pod původním názvem), dále symbolický odkaz na tentýž soubor s názvem `fst`.
2. Pak zde vytvořte pevný odkaz na soubor `.bashrc` ze svého domovského adresáře (`~/.bashrc`) pod názvem `bashrc` (bez tečky).
3. Vypište obsah svého adresáře `documents` v rozšířené formě, abyste viděli označení jednotlivých položek vytvořených v předchozím úkolu.

5.5 Přesměrování a filtry

5.5.1 Směrování a deskriptory

Také v UNIXu existuje standardní vstup, výstup a chybový výstup. Symboly `<`, `>`, `>>`, `|` fungují jako ve Windows. Programy, které přes roury přijímají vstupy a dokážou výstup poslat do roury, se nazývají *filtry*. Podobně jako ve Windows, i zde používáme deskriptory, které se píšou vždy nalevo od symbolu směrování. Deskriptory pro vstup, výstup a chybový výstup jsou `0 (STDIN), 1 (STDOUT), 2 (STDERR)`.

S číselným označením deskriptorů se pracuje podle tabulky 5.3 (máme o něco víc možností než ve Windows).

Směrování	Význam
`i> soubor`	přesměrování výstupu původně směrovaného na deskriptor `i` do souboru
`i>> soubor`	připojení výstupu místo na deskriptor `i` do souboru
`<&i`	vstup je přesměrován z deskriptoru `i` (vstup půjde z místa, které je nastaveno v `i`)
`i<&j`	vstup je směrován místo z deskriptoru `i` z deskriptoru `j`
`>&i`	výstup je přesměrován do deskriptoru `i`
`i>&j`	do výstupního deskriptoru `i` je přiřazen deskriptor `j` (`i` nasměrujeme tam, kam míří `j`)

Tabulka 5.3: Práce s deskriptory

Příklad

Následující dvojice příkazů jsou ekvivalentní:

```
ls -la > soubor          sort < soubor
ls -la 1> soubor         sort 0< soubor
```

(je důležité, aby mezi deskriptorem a směrovacím symbolem *nebyla mezera*)

Přesměrujeme chybový výstup prohledávání adresáře `/var` na zařízení `/dev/null`, tedy do „koše“:

```
find /var -name '*' -print 2> /dev/null
```

V příkladu jsme viděli využití zařízení `/dev/null`. Jedná se o „odpatkový koš“ ve stejném významu, v jakém známe zařízení NUL ve Windows. Můžeme používat například tato zařízení:

- `/dev/null` – výstupní zařízení pro odstranění výpisů (odpadkový koš)
- `/dev/random` – vstupní zařízení pro náhodná čísla
- `/dev/urandom` – totéž, jen je jiný algoritmus generování náhodného čísla
- `/dev/zero` – vstupní zařízení generující nuly
- `/dev/tty` – vstupní zařízení, které znamená terminál (z pohledu uživatele prostě klávesnice)
- `/dev/lp0` – výstupní zařízení představující tiskárnu

Pokud obvykle výstupní zařízení `/dev/null` použijeme jako vstupní soubor (například v příkazu

```
cat /dev/null > soubor
```

výsledný soubor je prázdný (nebo pokud neexistoval, vytvoří se prázdný soubor).

Příklad

Ukázky přesměrování deskriptorů:

`ls /adr 1>soubor 2>&1` přesměruje standardní výstup do souboru a standardní chybový výstup do deskriptoru, který je momentálně pod &1, tedy také do souboru

`ls /adr 2>&1 1>soubor` přesměruje chybový výstup na původní výstup a potom výstup na zadaný soubor (všimněte si změny pořadí směrovacích parametrů)

`ls -la >& cely_vystup.log` přesměrujeme chybový výstup na stejné místo jako standardní výstup, tj. do zadaného souboru (jinými slovy, standardní i chybový výstup budou v tomto souboru)

Poznámka:

Některé kombinace se mohou zdát nesmyslné, ale mějme na paměti, že přesměrování deskriptorů platí pro celou úlohu (pod tím si zatím můžeme představit posloupnost příkazů propojených rourami) a tedy hodnoty deskriptorů „putují“ postupně celou rourou doprava.

5.5.2 Filtry

 Co je to filtr, už víme. Podíváme se na několik nejdůležitějších programů, které mohou pracovat jako filtry.

`sort [přepínače] soubor(-y)`
setřídí řádky textového souboru

`sort -u soubor` pokud narazí na dva stejné řádky, v setříděném výstupu bude pouze jeden z nich (vynechává duplicitní řádky)

`sort -r soubor` reverzní třídění (setřídí v opačném pořadí)

`sort < souborzdroj > souborcil` program získal svůj vstup ze souboru směrováním, podobně výstup je směrován do dalšího souboru

`cat soubor | sort` získání vstupu přes rouru, příkaz používáme jako filtr

`wc [přepínače] soubory`
příkaz počítající počet řádků, slov a bytů souboru, používáme přepínače `-c` (vypíše počet bytů souboru včetně formátovacích jako je konec řádku), `-m` (počet znaků v souboru), `-w` (počet slov – řetězců oddelených bílými znaky), `-l` (počet řádků, ve skutečnosti počet symbolů konec řádku), `-L` (počet znaků v nejdelším řádku).

`wc -m soubor` vypíše počet znaků souboru

`wc -c soubor` vypíše počet B (oktetů) souboru, což není úplně totéž jako počet znaků (některé znakové sady kódují znaky do více než jednoho B)

`wc -l soubor` vypíše počet řádků (lines) souboru

`wc -w soubor` vypíše počet slov (tj. řetězců oddělených netisknutými znaky, třeba mezerami, tabulátory, konci řádků)

`wc -L soubor` (pozor, velké písmeno) vypíše délku nejdelšího řádku, tedy „šířku“ souboru

`wc -mclwL soubor` vypíše všechny výše uvedené údaje; ale pozor, údaje nejsou nijak označeny (vypíše se prostě řada čísel), pořadí není dáno pořadím parametrů v příkazu, ale je to „řádků slov oktetů znaků délka_řádku“ (pořadí najdete v nápovědě)

`wc -c soubor1 soubor2 soubor3` vypíše požadovaný údaj postupně pro všechny zadané soubory

`ls -la | wc -l` zjistíme počet položek včetně skrytých v pracovním adresáři (vypíše se číslo)

`pg, more, less`

programy (filtry) pro stránkování souboru; `pg` je nejstarší a nejjednodušší, obvykle používáme další dva (filtr `less` je ve většině distribucí interně používán pro zobrazování manuálových stránek příkazem `man`, lze používat i kurzorové klávesy)

`more -s soubor` více prázdných řádků za sebou zobrazí jako jediný prázdný řádek (squeeze multiple blank lines into one)

`more +`*`číslo`* `soubor` zobrazování začne až od řádku s tímto číslem

`ls / -R | less` stránkuje se výstup příkazu, který zobrazí obsah kořenového adresáře (tj. /), a to rekurzívně (výstup má velmi dlouhý), mezerníkem stránkujeme, kurzorovými klávesami se můžeme volně pohybovat po výstupu, program ukončíme klávesou [q]

Příklad

Stránkovací filtry vypisují svůj vstup po stránkách. Například

`ls -l | pg` `ls -l | more` `ls -l | less`

Ovládání filtru `pg`*:*

[Enter] další stránka

[+], [−] další stránka, předchozí stránka

[n] přechod na stránku n (číslo)

[$] přechod na poslední stránku

[q] ukončení

Základní ovládání filtrů `more` *a* `less`*:*

[Mezerník] další stránka

[Enter] další řádek

[h] vypíše se nápověda

[g], [G] přechod na začátek/konec textu

[q] ukončení

Kromě toho mají tyto filtry další možnosti ovládání, jak bychom zjistili z nápovědy. Podporují například odskoky v textu, vyhledávání, přeskoky mezi soubory, mají také vlastní příkazový prompt.

Příklad

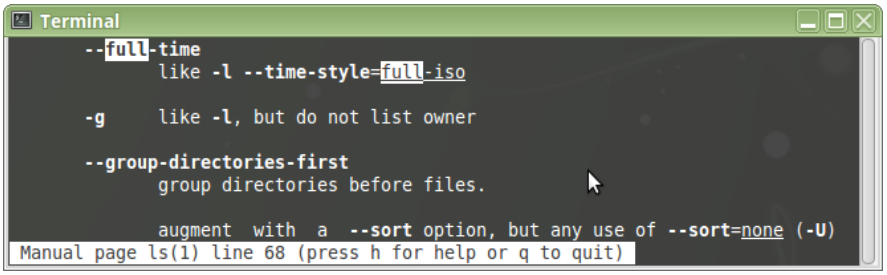
Ve většině distribucí je pro manuálové stránky používán stránkovač `less`, velmi pravděpodobně tak tomu bude i ve vaší distribuci. Z toho vyplývají i možnosti ovládání stránek, včetně prohledávání textu. Pokud chceme na manuálové stránce (nebo obecně ve výstupu příkazu `less` najít určitý řetězec, napíšeme lomítko, za něj hledaný řetězec a klepneme na [Enter].

Předpokládejme, že si nemůžeme vzpomenout, kterým parametrem se u příkazu `ls` vynucuje zobrazování úplného časového údaje. Nejdřív zobrazíme manuálovou stránku příkazu:

```
man ls
```

Kdybychom teď nevěděli jak dál, stačilo by stisknout klávesu [h] (ale my víme): chceme přejít k prvnímu výskytu řetězce „full“ (ten se dá předpokládat v popisu parametru nebo přímo v parametru):

```
/full
```



Obrázek 5.2: Hledání na manuálové stránce

Výsledek vidíme na obrázku 5.2 – všechny výskyty hledaného slova jsou zvýrazněny a zároveň jsme se přesunuli na jeho první výskyt. Takže teď už víme, že máme napsat `ls --full-time`.

Pokud nám nevyhovuje první výskyt (ani další, které by byly viditelné na téže stránce), můžeme se posouvat postupně na následující výskyty klávesou n (podle „next“ – další), nebo v opačném směru klávesou N.

Práci s manuálovou stránkou ukončíme klávesou q.

Některé z dříve probíraných příkazů se také dají použít jako filtry, například příkaz

```
ls -la | head -10
```

zobrazí pouze prvních 10 řádků výpisu prvního příkazu. Na další typ filtrů, vyhledávací, se podíváme v příštím semestru. Nejpoužívanějším vyhledávacím filtrem je `grep`.

Úkoly

1. Vytvořte soubor `prvni` s jakýmkoliv obsahem. Pak tento soubor setřiďte, setříděný výstup přesměrujte do souboru `druhy`.
2. Vypište obsah adresáře `/dev` včetně skrytých souborů, a to i s vlastnostmi položek (tj. široký výpis), použijte některý stránkovací filtr.
3. Spočítejte řádky předchozího výpisu (tj. zjistěte, kolik položek je v adresáři `/dev`).
4. Zjistěte velikost souboru `.bashrc` ve vašem domovském adresáři.
5. Zjistěte počet všech uživatelů definovaných v systému (nápověda: seznam uživatelů je v souboru `/etc/passwd`, každý záznam na samostatném řádku).
6. Zobrazte manuálovou stránku příkazu `ping` (tento příkaz slouží k ověřování dostupnosti zařízení v síti). Použijte postup pro vyhledávání na manuálové stránce z výše uvedeného příkladu, najděte na stránce parametr, kterým se určuje interval pro zasílání testovacích paketů (tj. hledáte klíčové slovo `interval`). Pozor, několik prvních výskytů není to, co hledáte, několikrát použijte klávesu pro přesun k následujícímu výskytu.

5.6 Základy práce s proměnnými

Proměnné jsou uloženy v souboru `.bashrc` nebo jsou předdefinovány systémem či v jiném konfiguračním souboru. Když vytvoříme proměnnou, bude existovat jen do ukončení práce v daném shellu,

takže když chceme, aby existovala déle, musíme příkaz pro její vytvoření dopsat například do souboru `.bashrc` v našem domovském adresáři.

Nejdůležitější proměnné jsou:

`HOME` domovský adresář uživatele

`TERM` typ terminálu

`SHELL` cesta k používanému shellu

`USER` přihlašovací jméno uživatele

`PATH` seznam adresářů, ve kterých se hledá spouštěný soubor, jednotlivé cesty jsou odděleny *dvojtečkou*

`PS1` prompt, výzva příkazového řádku konzoly nebo terminálu

`PWD` pracovní adresář

Kromě proměnné `PS1` existují obvykle ještě další proměnné určující prompt. Většinou se setkáme alespoň se sekundárním promptem `PS2`, který je zobrazován například při psaní víceřádkových příkazů.

Poznámka:

V proměnné `PATH` není ve výchozím nastavení cesta k pracovnímu adresáři a při spuštění programu také na rozdíl od Windows není spouštěný program hledán v pracovním adresáři! Proto pokud chceme spustit například program `mujprogram` umístěný v adresáři, který je právě naším pracovním adresářem, provedeme to takto:

```
./mujprogram
```

Teoreticky by se tento „problém“ dal vyřešit přidáním adresáře „`.`“ do proměnné `PATH`, ale toto řešení se z bezpečnostních důvodů nedoporučuje.[3]

Existuje také obdoba rozdělení proměnných na běžné a dynamické, s čímž jsme se setkali už ve Windows. Například proměnná `PS1` je dynamická.

Proměnné jsou buď *lokální* (platné pouze v rámci skriptu, příkazového shellu nebo bloku uvnitř skriptu), a nebo jsou exportovány do globálního prostředí, a tedy jsou viditelné i mimo oblast, ve které byly deklarovány. Když vytvoříme proměnnou, je pouze lokální, a až exportováním ji „zveřejníme“.

Příkazy pro práci s proměnnými:

`env`

vypíše proměnné s jejich obsahem (proměnné prostředí)

`set`

vypíše veškeré proměnné a funkce, které jsou definovány v dané oblasti, ve které pracujeme (výstup je poměrně rozsáhlý), je lepší používat raději `env`

`unset proměnná`

odstraní proměnnou

[3] Představte si, že ve vašem domovském adresáři (který velmi často bývá pracovním adresářem) bude podstrčen škodlivý software pojmenovaný stejně jako některý z běžně používaných příkazů, nejlépe příkazu obvykle se nacházejícího v adresáři `/sbin`, který často nebývá zahrnut v proměnné `PATH`. Pokud zařadíme do proměnné `PATH` tečku, můžeme tento škodlivý software omylem spustit (a pokud je jeho programátor dost chytrý na to, aby po dokončení své vlastní činnosti spustil původní program na správném umístění, tak to ani nezjistíme). Mohlo by se zdát, že stačí tečku přidat až na konec proměnné `PATH`, aby všechny dříve uvedené adresáře měly přednost, ale toto nefunguje, pokud některé jinak důležité cesty nejsou v této proměnné zahrnuty.

`echo $proměnná`

vypíše obsah proměnné (vyhodnotí ji), pracuje i s dynamickými proměnnými

`echo Můj domovský adresář je ${HOME}, jsem ${USER}` vypíše zadaný řetězec, přičemž za podřetězec `${HOME}` se dosadí domovský adresář a za `${USER}` přihlašovací jméno uživatele (všimněte si těch uvozovek)

`proměnná=výraz`

změní obsah proměnné

`read proměnná`

načte ze standardního vstupu řetězec do proměnné

Pokud chceme, aby námi vytvořená proměnná byla viditelná i mimo pracovní prostředí, ve kterém je vytvořena (například i mimo skript, nejrůznějším dalším programům apod., musíme ji *exportovat*:

```
export prom
```

Exportovat můžeme proměnnou i zároveň s jejím vytvořením a inicializací:

`export mojeprom=$PWD` (do proměnné jsme načetli název pracovního adresáře)

Znak $ vyhodnotí vše, co se za ním nachází (vše až po první mezeru nebo konec řádku považuje za název zpracovávané proměnné), proto když zadáváme za názvem proměnné ještě něco dalšího, uzavřeme tuto proměnnou do lomených závorek:

```
export PATH=${PATH}:/usr/TeX/bin
```

Příklad

Předpokládejme, že někde třeba ve skriptu používáme proměnnou. Nejdřív ji vytvoříme a inicializujeme, pak vypíšeme její obsah.

```
prom="A B   C"
echo $prom
echo "$prom"
echo \$prom
```

Druhý, třetí a čtvrtý příkaz nám postupně dají tyto výstupy (všimněte si důsledku použití uvozovek):

```
A B C
A B   C
$prom
```

Postup

Do proměnné nemusíme ukládat jen řetězec nebo číslo. Obrácené apostrofy mohou sloužit k vynucení vyhodnocení výrazu. Například:

```
prom="Výstup: ; `ls -la`"
echo $prom
echo "$prom"
```

V prvním příkazu jsme do proměnné uložili příkaz (vlastně dva příkazy oddělené středníkem, aby mohly být na jednom řádku) k interpretaci. Obrácené apostrofy uvnitř uvozovek jsou zde nutné.

V dalších příkazech vypisujeme obsah proměnné, čímž vyvoláme interpretaci příkazů v ní uložených. Vyzkoušením si můžeme ověřit význam uvozovek u posledního příkazu.

Postup

Další ukázky s proměnnými:

`export PS1='$PWD'` do proměnné `PS1`, tj. do momentálního promptu, si uložíme pracovní adresář

`PS1="\W"` pracovní adresář bez cesty, s cestou je `\w` (a také zobrazuje domovský adresář jako vlnovku)

`PS1="\w:"` promptem bude plná cesta k pracovnímu adresáři, na konci dvojtečka

`echo $PS1` ověříme si, co je momentálně nastaveno jako prompt

`PS1="\d"` promptem bude aktuální datum

`PS1="\A"` promptem je aktuální čas – jen hodiny a minuty, jiné možnosti pro čas jsou `\t` nebo `\T`

`PS1="\u\h\$ "` prompt je ve formě uživatel počítač$ (plus mezera)

`PS1="\u\h:\w> "` prompt je ve formě uživatel počítač: pracovní adresář> (plus mezera)

`PS1='pracuji tak dlouho: $SECONDS'` opět změníme prompt, proměnná `SECONDS` je dynamická

`mkdir $HOME/novy` využili jsme proměnnou uvnitř parametru příkazu; všimněte si složených závorek kolem názvu proměnné, zde jsou nutné

Příkaz `set` je ve skutečnosti mnohem silnější, nejde jen o vypisování seznamu proměnných a funkcí. Můžeme například do proměnné přiřadit výsledek funkce (podrobnosti najdeme v manuálové stránce příkazu `set`).

Úkoly

1. Vyzkoušejte příkazy vypisující proměnné. Vyberte si kteroukoliv proměnnou a vypište její obsah.
2. Vyzkoušejte příkazy z příkladu na straně 97.

5.7 Aliasy

Alias je zástupný název pro nějaký řetězec, obvykle příkaz nebo část příkazu. Pro práci s aliasy existuje příkaz `alias`, při použití bez parametrů vypíše všechny existující aliasy.

`alias` vypíše nadefinované aliasy

`alias dir='ls -la'` vytvoříme příkaz `dir`, který bude fungovat stejně jako `ls -la`

`alias md='mkdir'` vytvoří zkratku pro příkaz `mkdir`

`alias ls='ls -a'` od této chvíle se při zadání příkazu `ls` (případně i s dalšími parametry) budou vypisovat všechny položky včetně skrytých

`alias seznam='cat -n'` vytvoříme vlastní příkaz (vlastně ani ne tak příkaz jako obdobu makra), který vypíše zadaný soubor a zároveň očísluje řádky. Používá se takto:

```
seznam soubor.txt
```

`alias odstrel='kill -9'` vytvoříme příkaz, který ukončí zadaný proces, i kdyby byl například zamrzlý, použití pro proces s PID 2510:

```
odstrel 2510
```

`alias textove="ls -la | grep '.txt'"` tento příkaz vypisuje všechny textové soubory v pracovním adresáři, včetně skrytých

`alias c+='mount /mnt/sda1'` vytvořený příkaz `c+` připojí první oddíl prvního pevného disku

Poznámka:

Aliasy platí vždy jen do ukončení systému, takže pokud chceme některý zástupný řetězec používat trvale, musíme ho zapsat do některého skriptu, který se spouští hned po startu počítače, přihlášení nebo startu shellu (např. `~/.bashrc`).

Úkoly

1. Zjistěte, které aliasy jsou ve vašem systému definovány.
2. Vytvořte si některý z výše uvedených aliasů (nebo podle vlastní fantazie), vypište seznam aliasů (abyste si ověřili, zda byl opravdu vytvořen), vyzkoušejte (pokud není destruktivní).

5.8 Další příkazy

 Podíváme se na některé další jednoduché příkazy:

`date`

výpis data a času, parametry můžeme stanovit, co a v jaké formě bude vypsáno

`date +%d.%B` vypíše datum ve tvaru den.měsíc (měsíc bude vypsán slovně, podle jazykového nastavení systému)

`date +%x` vypíše datum ve tvaru měsíc/den/rok

`date +%T` vypíše čas ve tvaru hodiny:minuty:sekundy

`date >> log.dat` tento příkaz umístěný do `.login`, případně `.profile`, způsobí, že při každém přihlášení se do souboru `log.dat` přidá datum tohoto přihlášení, máme tedy přehled o tom, kdy se uživatel tohoto jména přihlašoval

`clear`

smazání obrazovky

`whoami`

takto zjistíme, pod kterým účtem právě pracujeme

`echo [-e] [-n] řetězec`

tento příkaz už známe. Jeho základní účel je stejný jako ve Windows, tedy vypíše svůj parametr (vše včetně mezer). Přepínač `-n` zamezí přechodu na nový řádek po vyhodnocení příkazu.

Pokud parametr nebo jeho část uzavřeme do uvozovek, můžeme používat speciální znaky (*escape sekvence*) (například `\n` způsobí přesun na nový řádek – zařádkování ve výstupu, `\t` je tabulátor), ale navíc musí být použit přepínač `-e`. Můžeme použít i apostrofy, pak obsah nebude interpretován (vypíše se tak jak je).

Příklad

Tento příkaz vytvoří následující tabulku:

```
echo -e "Nadpis1\tNadpis2\nObsah1\tObsah2"
    Nadpis1      Nadpis2
    Obsah1       Obsah2
```

Úkoly

1. Vyzkoušejte různé možnosti zobrazení data a času.
2. Zobrazte manuálovou stránku příkazu `echo` (v sekci 1). Zjistěte, jaké přepínače má tento příkaz a jaké formátovací řetězce lze používat mimo výše uvedených.