aktualizuji staré, opravuji chyby gramatické, stylizační i faktické, kterých se zde, vzhledem k rozsahu i šíři záběru najde víc než dost. Proto jsem na diskusním serveru mageo.cz založil k tomuto dokumentu diskusní fórum, které je přístupné i anonymně, tj. nemusíte se nikde registrovat. V něm mi můžete vytýkat chyby, pokládat doplňující dotazy, upozorňovat na nejasná místa, poskytovat doplňující informace a tak podobně, jak bude libo. Věřím, že to pomůže kvalitě tohoto dokumentu a tedy i nám všem.

úvod do systému Linux

Petr Mach alias Wraith
gsl@seznam.cz

"Děti, kdo ví kolik je 4 + 9?" "No, Aničko?"
"Prosím 5 jablíček, paní učitelko." "Kdepak. Honzíku, kolik to je?"
"Mmmm, 20 vrtulníků paní učitelko?" "Také ne. Co ty, Pepíčku?"
"To je prosím 7!" "Správně! Jak jsi to vypočítal Pepíčku?"
"To je jasný, 20 vrtulníků mínus 5 jablíček!"

Preambule

Úvod do systému Linux je dokument v jednom souboru, který je rozdělen do sekcí a tyto sekce do kapitol. Navigační menu zobrazuje seznam kapitol po jednotlivých sekcích, mezi nimi je potřeba se přepínat.

Tento text vzniká na popud mé nespokojenosti se seriálem o Linuxu pro začátečníky na serveru www.zive.cz. Jsem přesvědčen, že začátečník potřebuje jiný přístup a informace, než poskytuje živě. Seriál na živě je kontraproduktivní a začátečníky odrazující, protože Linux je představován skrze příkazovou řádku. To je z následujících důvodů chyba:

1. Naprostá většina začátečníků má již nějaké zkušenosti s Windows, které jsou orientovány výhradně na GUI (grafické rozhraní).
2. Linux umožňuje ovládání jak přes GUI tak přes příkazový řádek.
3. Je dobře známo, že GUI je pro člověka intuitivnější, a že se snadněji učí. Příkazový řádek v mnoha případech umožní efektivnější ovládání, ale jeho dobré zvládnutí je podstatně náročnější.
4. Linux a Windows se navzájem liší svou koncepcí. Spousta věcí v Linuxu funguje jinak. Začátečník musí tyto rozdíly vstřebat. Je potřeba mu je vysvětlit, je potřeba mu říct, proč je tohle takhle a takhle, jaké to má výhody a nevýhody, k čemu je to dobré. Jsem přesvědčen, že je zbytečné a hloupé mu to stěžovat ještě jiným ovládáním, než na jaké je zvyklý a když to není potřeba.

Je dobré začátečníka informovat, že existuje něco jako konzole, příkazový řádek a k čemu to je dobré. Ale je zbytečné mu zatěžovat hlavu příkazy pro kopírování, mazání a přejmenovávání souborů, pro vytváření adresářů, procházení a zobrazování obsahu disku a podobně, a to vše včetně různých obskurních přepínačů pro modifikaci jejich

chování, když stačí např. říct, že existuje správce souborů Midnight Commander nebo Nautilus, s kterým dokáže víc a snadněji, protože podobné nástroje zná už z Windows.

Příkazový řádek v Linuxu je kvalitní a užitečný, to bezesporu. Ale jeho ovládnutí není bezpodmínečně nutné a začátečník by do něj neměl být nucen. Je to pokročilejší téma pro ty, kteří chtějí Linux nejen nějak ovládat, ale chtějí to umět i efektivně a jsou ochotni tomu věnovat čas a píli, aby se to naučili. A to nemusí být cíl každého uživatele, někdo se může spokojit s méně efektivním ale jednodušším ovládáním. Proč ne, Linux to umožňuje a není na tom nic špatného.

Já ale nejsem spokojen s návody pro linuxové začátečníky celkově. To proto, že kladou příliš důraz na 'JAK něco udělat'. Podle mě je vhodnější klást důraz na 'JAK to funguje'. Věřím, že průměrně inteligentní člověk, který chápe podstatu věci, si už to 'JAK to udělat' dokáže zodpovědět sám, resp. stačí mu mnohem menší popostrčení. Bude samostatnější. Je to vhodné i proto, že Linux není unifikovaný jako Windows, jedna a ta samá věc se dá v Linuxu udělat mnoha způsoby a nástroji. Možností JAK to udělat je mnoho. Přitom JAK to funguje je jenom jedno.

Výše uvedené myšlenky mě vedou k vytvoření vlastního Úvodu do Linuxu pro začátečníky, ve kterém nebude primárně vysvětlováno jak se co dělá, ale jak to funguje, proč to tak je, jaké to má výhody a nevýhody, prostě pokec okolo, který by měl napomoci s pochopením o čem to ten Linux vlastně je. Myslím že se jedná o informace, které jsou pro začátečníky důležité a přitom nepříliš dostupné.

Pozadí Linuxu

Linux, fenomén naší doby. Milovaný i zatracovaný. Kdo by o něm už alespoň něco neslyšel? Nebo alespoň nevyužil jeho služeb? Jeho příznivci ho nekriticky chválí až do nebes, jeho odpůrci po něm nekriticky plivou a nenechají na něm nitku suchou. Mezi příznivci Windows a Linuxu hoří nesmiřitelný boj. Linux vyvolává vášně, není to obyčejný kus software. Při hádkách o Linux nejde ani tak o ten Linux, ale o princip jehož je hlavním představitelem. Linux je produkt zvláštní filozofie, možná ideologie, o které by člověk řekl, že nemůže fungovat. A přece funguje, a má se k světu, Linux je toho důkazem.

Je dobré o tom něco vědět, protože to má vliv na to jaký Linux je, jaké jsou jeho výhody a nevýhody, co může a co nemůže. Operační systém Linux je produktem hnutí Svobodného software (Free software). Nepleťte si to s volným software (freeware), což je software zadarmo. I svobodný software může být zdarma a často tomu tak je, ale není to u něj vyžadováno. Někteří lidé si myslí, že opakem svobodného software je komerční software, ale není tomu tak, Svobodný software může být i komerční.

Užívání software je umožněno licencí, pod kterou ho autor šíří. Existuje mnoho různých licencí, licence nesvobodného software je charakteristická tím, že chrání zájmy autora. Zakazuje uživateli šíření tohoto software, jeho modifikaci pro vlastní potřeby, zkoumání kódu a vůbec jsou sestaveny z řady zákazů a omezení uživatele. Licence Svobodného software (GPL) je přesný opak, jejím cílem je ochrana zájmů uživatele. Zajišťuje mu právo na svobodné nakládání s tímto softwarem, zaručuje mu, že uživatel může dotyčné sw dílo volně šířit (i prodávat), modifikovat a především, že k němu budou volně dostupné zdrojové kódy.

To je alfa a omega toho všeho. Volně dostupné zdrojové kódy a stovky a tisíce dobrovolných programátorů, nijak pevně řízených a organizovaných, kteří věří ve sdílení znalostí. Jejich cílem není zisk, ale vytvořit kvalitní operační systém a aplikace pro něj. Jen tak, pro radost, aby se člověk něco naučil a třeba si i jen něco vyzkoušel. Je opravdu podivuhodné, že taková dobrovolná činnost a v tak širokém (celosvětovém)

měřítku úspěšně funguje.

Tohle skutečně stojí za povšimnutí a za úvahu. Vezměme si na pomoc ekonomii. Ta zná dva způsoby řízení tvorby produktů a jejich spotřeby na úrovni společnosti. Centrální řízení (všem podle jejich potřeb, představitel je komunismus) a tržní řízení (neviditelná ruka trhu, představitel je kapitalismus). Centrální řízení je teoreticky lepší, šetrnější ke zdrojům a efektivnější. Potřebuje k tomu ale dokonalé lidi ve všech sférách společnosti, lidi kteří nejsou sobečtí, líní a chamtiví, nepodléhají svodům svěřené moci a tak podobně, takže prakticky je nepoužitelný, v praxi se svrhává v totalitu a ani v ní nefunguje. Ten tržní také nevede k ideálním výsledkům, posazují se v něm špatná řešení (např. drancování přírody), trpí poruchami (monopoly, které v příslušné oblasti trhu pak zavádí pro tržní regulační mechanizmus velmi nezdravé centrální řízení), je potřeba ho tedy chránit, řídit a tedy netržně regulovat, což opět vyžaduje na řídící úrovni dokonalé lidi. Ale i tak dosahuje lepších výsledků a je možné ho provozovat v demokracii.

Na free software je zajímavé to, že se nedrží ani jednoho z těchto schémat. Není centrálně řízený, naopak, je živelný a funguje na principu soutěže a konkurence, tak jako kapitalismus a možná ještě víc. Ale ne na trhu, ten je pro něj nezajímavý, což zamotává hlavu spoustu lidem, kteří nedovedou pochopit, jak tedy může fungovat. Tedy na trhu ekonomickém, Free Software má svůj vlastní trh. Bojuje o přízeň uživatelů, to je jeho jediné měřítko úspěchu a hodnoty. Je volně šiřitelný, každý si ho může stáhnout a dělat si s ním co chce podle svých potřeb. Staví na sdílení myšlenek a práce, v podstatě bez nároků na odměnu, každému podle jeho potřeb tak jak to předpokládá komunismus.

Free Software tak vlastně zavádí jakousi kombinaci kapitalismu a komunismu v té nejčistší esenciální podobě. Komunistické sdílení kódu (práce a myšlenek) paradoxně vede k velmi tvrdé kapitalistické konkurenci, které se samotný kapitalismus brání různými ochrannými opatřeními, jako jsou patenty a podobně. Přitom vše je dostupné všem. Vysvětlit tento jev nedovedu. Může to být anomálie, která časem zmizí, stejně tak to ale může být předzvěst nového paradigma ekonomie či fungování informační společnosti, kterou se pomalu, ale jistě stáváme. Paradigma, které změní náš pohled na duševní vlastnictví, patenty, autorská práva a vůbec nakládání s majetkem nehmotné povahy. Nebylo by to v historii západní kultury poprvé. A že současný stav není dobrý, že je nedostačující potřebám dnešní doby, že je v něm něco shnilého, to je evidentní. Stačí se podívat na kauzy typu Napster a dalších výmněných sítí a aktivit organizací typu RIAA či OSA, nebo na kauzy vyvolané společnostmi EOLAS či SCO. Změna je nutná, ale zatím není v zájmu mocných.

Zajímavé může být také zamyšlení nad tím, co vyvolalo takový úspěch Free Software. Podle mě je to to výsledkem úspěchu a agresivního chovaní Microsoftu. Microsoft se na významné části IT trhu stal extrémně dominantní a přesto nepřipustil žádnou konkurenci. Konkurenční firmy nelikviduje kvalitou svých produktů, ale marketingem a ekonomicky. Viz nejznámější kauza Netscape. Navíc ani není schopen či ochoten spolupracovat s konkurencí, úmyslným nedodržováním standardů je pověstný. Jako dominantní firma si to může bohužel dovolit. To podle mě stojí za úspěchem Free Software, protože Free Software se nedá zničit ekonomicky, není to ekonomický subjekt. Jako takový dokáže jako jediný úspěšně konkurovat Microsoftu.

Free Software ale není nijak zvlášť řízen či organizován. Je to spíš hnutí, které stojí na jakémsi společném světonázoru. Není jednotné, uvnitř něj jsou různé myšlenkové proudy. Jeden se dokonce oddělil a existuje samostatně jako Open Source. Tyto dva proudy se shodly na tom, že mají společný cíl, ale rozdílnou cestu k němu. Mnozí lidé mezi Free Softwae a Open Source vůbec nerozlišují. "Hloupému uživateli" to je vcelku jedno, ať si myslí co chtějí, hlavně že z obou vypadávají programy zadarmo :-). Přesto

jsou ty myšlenky důležité, právě ony stojí za jakousi samoorganizací neřízeného vývoje a úspěchu Free Software.

Některé firmy které nedokáží konkurovat Microsoftu (ekonomicky) přímo (na trhu) mají snahu se uplatnit a tedy mu konkurovat mimotržně právě skrze Free Software nebo Open Source. Nevím nakolik to pomůže těm firmám, ale Free Software to pomáhá určitě. Vznikají tak produkty, na které by se možná sám nezmohl. Třeba Mozilla nebo OpenOffice.org. Ale ty by na druhou stranu nevznikly, kdyby Free Software už silnou pozici neměl, kdyby neměl silné zázemí jak ve vývojářích či vývojových nástrojích tak v uživatelích. Takže úspěch Free Software nelze oddůvodnit jen těmito firmami. Ostatně mnoho jiných úspěšných produktů vzniklo jinak, třeba Apache, GNOME nebo samotný Linux.

Úspěchy těchto produktů nepřehlédl samozřejmě ani komerční svět. Takže dochází i k opačnému jevu, a mnohé firmy se v současné době snaží na Free Software vydělat, tedy jej využít komerčně. Takže se nedivte, když někdo vezme nějaký produkt Free Software, něco k němu přidá a snaží se to prodat. Pokud při tom neporušuje licenci toho produktu, je to legální. A nikomu to ani nevadí, jak jsem již uvedl, Free Software může být i komerční. Proto i Linux, který je obecně zadarmo, může někdo nabízet za peníze a je jen na vás, zda na takovou nabídku přistoupíte.

Strategií jak vydělat na Free Software, potažmo na Linuxu, je hodně. Nebývají však příliš úspěšné, nějaký zaručený obchodní model jak na něm vydělávat asi neexistuje. Přinejmenším jak na něm rychle a hodně vydělávat. Jinak třeba RedHat, čistě linuxová firma si nevede nijak špatně. IBM si své investice do Linuxu také pochvaluje. Corelu to na druhou stranu nevyšlo, o SCO, která jednu dobu nabízela vlasní distribuci Linuxu škoda mluvit. Uvidíme jak si povede Novell, který do Linuxu mohutně investoval, koupil SUSE (linuxová distribuce) a firmu Ximian (výrobce komerčního desktopového prostředí založený na GNOME a OpenOffice.org a tvůrce projektu Mono, což je linuxová implementace .NET). Novell by se mohl stát nejvýznamnější linuxovou firmou, je to známá a bohatá firma a má dobře našlápnuto. Jak vidno, ani na komerční bázi není Linux žádný troškař, zajímají se o něj nejvýznamnější IT firmy.

Někteří lidé se této komercionalizace linuxu obávají. Myslí si, že to Linux poškodí. Já si myslím, že plná komercionalizace Linuxu nebo jeho vykradení nehrozí a že komerční i nekomerční Linux vedle sebe mohou docela dobře c symbióze koexistovat. Úspěšně to předvádí například RedHat s projektem Fedora.

Snad právě to je důvod, proč se Linuxu obává Microsoft. Ekonomicky ho zničit nemůže, nemůže ho ani koupit, FUD taktiky (označování GPL rakovinou a podobné výpady hlavních představitelů Microsoftu na různých konferencích) také nebyly úspěšné a tak Microsoftu nezbývá, než bojovat proti Linuxu kvalitou a cenou svých produktů. Jsem pevně přesvědčen, že nebýt Linuxu, nejsou Windows ani zdaleka tak kvalitní a levné, jako jsou teď, a že tedy každý, ať Linux používá nebo ne z něj má prospěch.

Předchozí odstavec musím poopravit, zdá se, že se Microsoft s Linuxem snaží bojovat skrze SCO a právní systém USA. Ale je to jen spekulace. Fakta jsou taková, že SCO iracionálně útočí na významné linuxové uživatele (velké firmy v USA), vyžaduje po nich licenční poplatky, ale legálnost tohoto nároku neprokázal a ani se o to nepokouší. Těm co neplatí hrozí žalobami o astronomické částky. SCO je firma na pokraji krachu, která neúspěšně podnikala s Linuxem. Nedávno se provalilo, že ji Microsoft přímo i nepřímo financuje desítkami milionů dolarů (není jediný) a tím ji uržuje při životě. Nazdá se však, že by to nějak Linux poškozovalo. Možná že by rostl rychleji než roste, ale to se dá těžko odhadnout.

Pokud přijmeme tezi, že úspěch Free Software je anomálie vyvolaná Microsoftem (anomálie na trhu IT), tedy jakási reakce na akci, tak lze předpokládat, že tato

anomálie zmizí společně s Microsoftem tak jak ho známe dnes. V tomto kontextu by byl boj Microsoftu proti Free Software legrační, protože by to představovalo boj sama proti sobě, boj na kterém by se Microsoft mohl docela dobře vyčerpat.

Linux versus Windows

Co je lepší? To je hodně ošemetná otázka, na kterou nelze odpovědět, neb je to věc názoru a osobních preferencí. S jistotou lze pouze říci, že to jsou výrazně rozdílné operační systémy a každý má své přednosti a nedostatky. S jistotou lze také říci, že některým lidem vyhovují Windows a některým Linux. Tahle kapitola vám tedy neřekne co je lepší, spíš uvede na pravou míru to, co se o Linuxu povídá, co vás v něm asi tak čeká a na závěr se pokusím i uvést nějaké obecné výhody a nevýhody.

Linux je trochu v nevýhodě, protože Windows jsou dominantní, kdežto Linux viděl, natož zná jen málokdo. To vede k tomu, že většina lidí (asi všichni), kteří zkouší Linux už má zkušenosti z Windows. Jsou zvyklí na Windows a na jeho způsob a styl práce. Potom i od Linuxu vyžadují takové chování, protože se jim zdá správné. Když to Linux dělá jinak, přijde jim to hloupé, ne-li přímo blbé. Ne nadarmo se říká, že zvyk je železná košile.

Je dobré mít na mysli, že Linux je jiný operační systém a že vyznává jiné principy (filozofii) práce než Windows. Linux je velmi podobný UNIXovým operačním systémům a funguje podobně jako oni. Dále se také zpočátku etabloval jako serverový operační systém, a tak jsou některé věci tomuto poplatné. Jako uživatelský (desktopový) operační systém se začal prosazovat později a jeho přizpůsobování těmto potřebám je na dobré úrovni, ale ještě není ukončeno. To znamená, že některé věci se v něm řeší sice flexibilně a robustně, ale z hlediska potřeb laického uživatele zbytečně těžkopádně.

O Linuxu kolují mnohé pomluvy a předsudky. Ty mají mnoho příčin. Jednou z příčin je frustrace uživatelů, kteří Linux zkusili, ale neporozuměli si s ním. Mnoho lidí si totiž myslí, že když znají Windows, tak jsou dobří a Linux pro ně bude hračka, že ho zvládnou levou zadní, bez námahy a nutnosti se učit. Jenže to není pravda. I velmi zkušený uživatel Windows bude v Linuxu naprostý a bezradný začátečník a bude se v něm těžko orientovat. Když říkám, že Linux je jiný operační systém než Windows, tak tím skutečně myslím, že to je jiný operační systém. Dokonce si myslím, že méně zkušení uživatelé Windows na tom jsou lépe. Čím hlouběji člověk vidí pod povrch, tím větší uvidí rozdíly. Ti nejméně zkušení uživatelé, kteří jsou rádi, že si umí spustit nějaký program a v něm něco udělat budou pozorovat rozdílů nejmíň.

Zkušenému uživateli, který je najednou degradován na začátečníka, se to samozřejmě nelíbí, někteří se tím dokonce mohou cítit frustrováni a potřebují si nějak ulevit, najít viníka, no jasně, blbý Linux. Z takových lidí se pak stanou nesmiřitelní nepřátelé Linuxu na život a na smrt. Normální lidi to samozřejmě takhle nebere, buď si řeknou, tohle není pro mě a nebo zatnou zuby a těžkými začátky se prokoušou.

Já jsem na Linux přecházel ještě z DOSu, který jsem znal jak své boty a dobře si pamatuji, jak jsem obtížně začínal. Chudák začátečník totiž neví nic. A řešení jednoho problému vyvolává řadu dalších, člověk shání informace po všech čertech, dílčí úspěchy jsou těžce vybojovány, řešení toho původního problému v nedohlednu. Dnes je to už přeci jen snazší, protože třeba ovládání GUI programů je v Linuxu i ve Windows stejné, existují intuitivní grafické konfigurační nástroje a hlavně, už je koho se zeptat, když si člověk neví rady. Když člověk vytrvá, naučí se pár základních věcí, pochopí jak to funguje, tak najednou přijde zlom, kdy už jde všechno prakticky samo, kdy vám vše najednou přijde intuitivní a pochopitelné a už se můžete jen smát tomu, jak jste z

počátku tápali. Pochopili jste, jak funguje Linux. Jestli jsem vás teď vyděsil, tak to je dobře, lepší očekávat horší a pak být příjemně překvapen lepším, než naopak. Používat Linux se skutečně budete muset naučit, samo od sebe to nepůjde. Ale abych vás zase neodradil. Osobně znám jen pár lidí, kteří používají Linux, určitě jsem výjimka, ale nadpoloviční většina z těch lidí jsou holky. A všichni víme, že holky, ač nejsou hloupé, tak složité technické věci radši přenechávají na hraní klukům. Takže řeči o přílišné složitosti Linuxu jsou jen výmluvou pro lenochy. Když se chce, tak o jde. Kdo nechce, ať zůstane u Windows, to je jeho rozhodnutí.

Další příčinou pomluv je zastaralost informací. Kdysi dávno bylo možno Linux ovládat jen z textového rozhraní přes příkazový řádek. Grafické rozhraní existovalo, ale za moc nestálo a prakticky v něm nešlo nic dělat, protože pro něj nebyly grafické nástroje. šlo si tam spustit emulaci terminálu s příkazovým řádkem a to je asi tak všechno. To bylo tak nějak před osmi lety. Od té doby Linux výrazně pokročil, zvláště v posledních letech, kdy se cca v roce 2000 začal Linux orientovat na desktop. Dovolím si tvrdit, že dnes má Linux grafické rozhraní vyspělejší a lepší než má Windows. Nicméně předsudek, že Linux má mizerné grafické rozhraní přetrvává dodnes. A tak je to i u jiných věcí.

Třeba lokalizace. Ta bývá Linuxu často vyčítána, paradoxně, podobně jako u GUI je v Linuxu na vyšší úrovni než ve Windows. Lokalizaci můžeme rozdělit na dvě oblasti, na technologickou a kvalitativní. Technologie lokalizace je v Linuxu daleko napřed. Nejdůležitější na lokalizaci je pro každého asi překlad programů do mateřského jazyka, u nás tedy do češtiny. Lokalizace aplikací v Linuxu je standardizovaná, existují pro ni kvalitní nástroje a je velmi jednoduchá. Základem je, že všechny texty alikace se (automaticky) udržují ve zvláštním souboru. Pro lokalizaci aplikace pak stačí tento soubor přeložit. To může udělat kdokoliv, ne jen programátor aplikace. K aplikaci lze přiloižt více překladů najednou. Dále v Linuxu existují standardní systémové proměnné LANG a LC_* podle kterch si aplikace automaticky, vybere jakým jazykem má s vámi hovořit.

Když máte nastavenu češtinu, aplikace se při spuštění podívá, zda pro ni existuje soubor s českým překladem a když ano, použije jej a aplikace na vás mluví česky. Ve stejnou dobu může být k vašemu počítači přihlášen jiný uživatel (to je v Linuxu normální, další věc, kde má Linux nad Windows navrch, protože je skutečně multiuživatelský), který může mít nastaven jiný jazyk. Když spustí tu samou aplikaci, bude s ním hovořit jeho jazykem. Ve windows věc nevídaná a to ani nemusí jít o současný běh aplikací, stačilo by, kdyby s Windows mohli postupně pracovat různí cizinci ve vlastním jazyku. Nebylo by pěkné, kdybyste byli v Německu nebo Francii na dovolené a tamější počítače na vás mluvily česky? Linux to umí. Pokud se nepletu, ani speciální multilinguální verze Windows nic takového nezvládá, v Linuxu je to běžná věc. Už při přihlašování si můžete zvolit, jakým jazykem s vámi má počítač mluvit.

Nejdůležitější na tom je, že ty překlady může dělat kdokoliv. To znamená, že nejste odkázáni na to, až vám program přeloží do češtiny autor programu. Překladatel ani nemusí být programátor, překlad se kódu aplikace nijak nedotýká. Díky tomu v Linuxu mluví česky překvapivě mnoho programů, jejichž autor o česku možná ani nikdy pořádně neslyšel. Stačí aby použil standardní lokalizační nástroj Linuxu - gettext. To je jen příklad toho, že Linux je v některých oblastech technologicky na výši a že to není jen ubohé obkoukávání ostatních (Windows), a že nic vlastního nepřináší, že Free Software není inovativní, jak o něm někteří lidé tvrdí. Není to pravda.

Tím ale nechci říci, že všechno špatné co se o Linuxu doslechnete, je lež a pomluva. Linux má svá problémová místa. Kupodivu se o nich ani moc nemluví, místo toho se zveličují nepodstatné banality. Těmto problémovým místům se věnuji na řadě míst tohoto dokumentu.

A pak ještě také existují sporné záležitosti. To jsou záležitosti, o kterých jedni tvrdí, že jsou fajn, že to je výhoda Linuxu, a druzí, že jsou otřesné a že Linux je kvůli nim nepoužitelný. Záleží na vašich preferencích, jaký k tomu zaujmete postoj. Jestli negativní, pak jste člověk, kterému Linux svou koncepcí vyhovovat nebude. Než ho ale odmítnete, zamyslete se, zda je to pro vás skutečně takový problém a nebo zda jste jen někým naočkovaní a jestli je to opravdu tak špatné.

Typický příklad jsou textové konfigurační soubory vs. binární registry Windows. Já si cfg. soubory nemůžu vynachválit, mají mnoho praktických výhod, třeba snadná možnost jejich zálohování a kopírování na sousední stroj, takže po reinstalaci nebo nové instalaci aplikace kdekoliv ji není potřeba znovu konfigurovat. Častý problém ve Windows, zvláště když reinstalujete samotný operační systém. Díky možným zálohám také můžete dělat s konfigurací různé pokusy bez obav, že se nedovedete vrátit do původního stavu a podobně. Častý protiargument, obtížnost jejich editace, není pravdivý, jejich ruční editace v textovém editoru není nezbytná, i pro konfigurační soubory může klidně existovat grafické rozhraní, stejně jako ve Windows není nutné používat pro konfiguraci rgedit. Problém vzniká pouze tehdy, když takový nástroj neexistuje, ale to není primárně problém konfiguračního souboru. Stejně tak i ve Windows musíte občas použít přímou konfiguraci pomocí regeditu. Konfigurační soubory jsou proti registrům srozumitelnější a přehlednější. Často doplněné vysvětlujícími komentáři.

A upřímně řečeno, já dávám obvykle přednost té ruční editaci. Je to totiž zpravidla efektivnější a někdy také flexibilnější. Obojí vysvětlím. Třeba přidání uživatele do nové skupiny znamené, že v textovém editoru otevřete soubor /etc/group, najdete řádek, který začíná jménem uživatele a na jeho konec připíšete název nové skupiny. Tohle je samozřejmě možné si i naklikat, ale bude trvat déle, než se tím proklikáte. A k té flexibilitě. Konfigurační soubor pro XFree86 (grafické rozhraní Linuxu) umožňuje nastavit mnoho věcí a poměrně komplexních. Třeba vlastní atypické rozlišení nebo zobrazovací frekvenci, různé layouty pro rozličné potřeby, různá speciální nastavení klávesnice a myši a další a další záležitosti závislé třeba i na vlastnostech ovladače grafické karty. Samozřejmě že lze udělat grafický konfigurační nástroj, který by tohle všechno umožnil nastavit, ale byl by hrozně složitý a uživatel by se v něm stejně ztrácel. Proto grafický nástroj pro nastavení XFree86 umožňuje jakési základní nastavení a kdo potřebuje něco speciálního, může si to zapsat do toho konfiguračního souboru přímo, ten tak umožňuje bohatší možnosti a větší flexibilitu.

Mám-li to nějak shrnout, tak ve Windows jsou skutečně některé věci řešené líp. Stejně tak ale jsou některé věci řešené lépe v Linuxu. Linux je mladý, zvláště na desktopu, tam teprve dozrává. To znamená, že se postupně staré nevyhovující technologie nahrazují za modernější. Hlavně co se týče GUI. Přechod na nové techologie je nenásilný, zavede se nová, počká se až se ujme a stará se opustí. To občas působí i problémy, hlavně začátečníkům, kteří tyto trendy nevnímají a matou je. Například v současné době (duben 2004) se ujímá nový způsob práce s fonty. Některé aplikace používají ještě starý, některé nový. Z uživatelského pohledu se to projeví při přidávání nového fontu, je o vlastně potřeba udělat nadvakrát, pro starý a nový systém. Je to také důvod, proč některé aplikace mají vyhlazené fonty a některé ne. Stará technologie vyhlazování totiž neumožňuje. Občas se také setkáte s problémy způsobené přechodem na Unicode (univerzální kódování, u kterého není potřeba rozlišovat jazyky). Tohle jsou ale problémy přechodné a možná se s nimi ani nesetkáte. Jsou to drobné vady na kráse, věci co zamrzí, ale nic co by znemožňivalo Linux používat.

Zrovna v uvdených příkladech, fonty a Unicode, jsou Windows dál, a nikdo to nezpochybňuje. Až se nové techologie ujmou a stanou se standardní, bude na tom

Linux v těchto oblastch přinejhorším stejně jako Windows. Jak jsem již psal, Linux je na desktopu mladý a rychle se na něm vyvíjí. Kdo zná situaci před třemi lety a dnes, tak mi dá za pravdu. Za tu dobu Linux učinil neuvěřitlný vývojový skok vpřed. Proto si dejte pozor, abyste si nenainstalovali něco starého. Já pořád občas vidím v obchodech prodávat řadu let starý RedHat, chudák ten, kdo si to koupí. Tento odstavec jsem psal v dubnu 04, dnes (říjen 04) mám už jiný dojem, a to že s Unicode se lépe pracuje a je použitelnější na Linuxu. Windows se od toho kódování CP1250 ne a ne odprostit a řada win programů, třeba i textových editorů, s Unicode pracovat neumí.

Základy

Tato sekce obsahuje obecné úvodní pojednání o Linuxu. Její cíl je uvést Vás do světa Linuxu, naznačit o čem to ten Linux je, jaké má silné stránky i s čím má problémy, na jakých principech staví, jak přibližně funguje, jaké jsou jeho hlavní části a tak podobně.

Linux, jádro a další neznámé pojmy

Je těžké definovat co je to Linux, protože neexistuje žádná jednotná a obecně uznávaná definice. Linux je fenomén, a různí a někdy i titíž lidé si pod tímto pojmem představují a myslí různé věci v závislosti na svých zkušenostech a znalostech nebo kontextu. Proto uvedu svou vlastní definici, s kterou nebudou mnozí souhlasit.

Linux je operační systém, který je postaven na jádře (kernel) operačního systému, které vytvořil a stále vyvíjí Linus Torvalds a které se také nazývá Linux.

Někteří jedinci prosazují pro operační systém Linux název GNU/Linux, ale obecně se to neujalo. Bližší informace o této záležitosti jsou na www.gnu.cz. To je ale jen slovíčkaření, já se budu radši věnovat podstatnější záležitosti, jádru.

Jádro je takový mozek operačního systému, podobně jako je procesor mozek počítače. Operační systém je také rozhraní mezi hardware a software, řídí systémové zdroje (paměť, výpočetní výkon) a přiděluje je mezi procesy (programy), poskytuje mnohé služby (práci se soubory, sítí) ostatním programům, zajišťuje a kontroluje přístupová práva a tak dále a tak dále. Jádro, které je centrum toho všeho udává charakter celému OS.

Jádro, tedy to co dělá Linux Linuxem, je jen jedno, ale je v mnoha verzích a paralelních vývojových větvích, které mu přidávají ty či ony vlastnosti. Jádro z hlavní vývojové verze se označuje (ne nazývá) jako vanilla. Toto jádro je dostupné na www.kernel.org.

Jádro je v Linuxu věc důležitá, protože dělá Linux Linuxem a spousta věcí se od něj odvíjí. Třeba podpora mnohého hardware. Když vám nebude něco fungovat a budete chtít radu, je nutné říct jakou verzi jádra máte. Tak jako ve světě Windows je důležité vědět (když máte problémy), jestli máte Windows 98 nebo 2000, je v Linuxu potřeba vědět jakou máte verzi jádra.

Verze jádra má podobu X.Y.Z, kde X, Y a Z jsou jakási čísla.

X - je tzv. major verze. Mění se jen mimořádně při velmi významných a rozsáhlých změnách celé architektury jádra. Není tedy divu, že v současné době je to jen číslo 2.

Y - je tzv. minor verze. Sudá čísla představují stabilní verze jader, lichá pak vývojová.

Funguje to tak, že vývojáři jádra pracují na nové verzi jádra, které se označuje nějak jako X.1.Z. Až ji mají hotovou, označí ji jako X.2.0. Tato by měla být bez chyb a bez problémů. Ve skutečnosti, novou stabilní verzi začne zkoušet mnoho uživatelů a je ohlášeno mnoho drobných chyb a problémů, takže následuje přechodné období, kdy se nové jádro intenzivně ladí a odstraňují různé nekompatibility s hw a další problémy. Nové stabilní jádro se stává skutečně stabilní až tak někdy od verze (Z) 5 až 20, podle kvality předchozí testovací fáze. Po přechodném období vývojáři vytváří další verzi, X.3.0 na které opět pracují. Současně tak existují dvě verze jader, stabilní a vývojová. Nové stabilní verze vznikají v cca 2 až 4 letech. Tedy změna tohoto čísla je přibližně ekvivalent vydání nové verze Windows.

Z - budu nazývat jako opravné. Představuje každou nově vydanou verzi jádra. U stabilního jádra to obvykle představuje opravu nalezené chyby, nebo přidání nových důležitých vlastností, s kterými je nevhodné čekat na nové stabilní jádro (podpora nového hw), mnohdy formou backportu z vývojové verze. U vývojové verze pak toto číslo představuje stupně vývoje nového jádra. Toto číslo se mění nejčastěji, někdy ze dne na den, někdy po měsíci (u stabilních verzí se perioda s rostoucím číslem prodlužuje). Za tímto číslem pak mohou následovat ještě různé přípony, jako třeba test1 či rc1 (release candidate), které označují různé vývojové stavy.

V současné době (prázdniny 2003) je aktuální stabilní vrze 2.4.21 (stará cca dva roky), vývojová 2.6.0-test2. Ale stále se ještě udržují předchozí stabilní verze, které jsou aktuálně 2.2.25 a 2.0.39. Již brzy by tedy mělo vzniknout nové stabilní jádro 2.6.0.

Doplnění, dnes (březen 2004) je aktuální jádro 2.6.3, vývojová verze zatím nebyla vytvořena, vývojáři intenzivně pracují na odladění jádra 2.6.

K tomuto vanilla jádru existuje mnoho neoficiálních úprav. Úpravy mohou do jádra přidávat nové vlastnosti, drobné i významné, odstraňovat některé nedostatky a podobně. Tyto úpravy se mohou časem dostat i do vanilla jádra. Úpravy se poskytují pomocí tzv. patchů. Patche mohou být i oficiální, třeba pro přechod z jedné verze jádra na vyšší, oprava chyby v jádře a podobně. Proto se tento pojem někdy nepřesně překládá jako záplaty.

Obecne je patch datový soubor, který obsahuje informace o rozdílech mezi dvěma zdrojovými kódy. Např. mezi zdrojáky upraveného a vanilla jádra. Tento datový soubor se pomocí programu/příkazu, který se nazývá patch, aplikuje na původní originální zdrojové kódy a tím z nich udělají ty upravené. Upravené zdrojové kódy se přeloží překladačem jazyka C do spustitelné podoby a vzniká tzv. opatchované jádro. Je samozřejmě možné např. stáhnout celé nové zdrojové kódy jádra s opravenou chybou, ale tyto mají v případě jádra mnoho desítek MB a patch, který opravuje třeba i velmi závažnou chybu, může být velký jen jeden KB a to už je rozdíl, při kterém se vyplatí se zabývat patchováním.

Nemyslím si, že by začátečník hned začal patchovat a překládat své vlastní jádro (pokud k tomu nebude donucen okolnostmi), ale opatchované jádro je v Linuxu běžná záležitost, a i začátečník by měl vědět co to znamená. Například snad všechny distribuce používají opatchovaná jádra.

Distribuce

Operační systém Windows dělá Microsoft, operační systém AIX dělá IBM, operační systém Solaris dělá SUN, operační systém Linux dělá...? Nikdo. Operační systém Linux je obecný pojem, něco jako pojem automobil, není to konkrétní výrobek. Je to operační systém (dále již jen OS), který má linuxové jádro. Tak jako existuje mnoho výrobců

aut různých značek a čistě teoreticky si můžete vyrobit i vlastní, tak existuje mnoho tvůrců OS Linux a čistě teoreticky si můžete vytvořit i vlastní (a snadněji než ten automobil).

Přijmu-li zjednodušenou definici operačního systému, že se jedná o základní software, který umožňuje a řídí používání počítače a jeho zdrojů, pak musím konstatovat, že většina tvůrců OS Linux vytváří něco, co je mnohem víc, než "jen" OS Linux. To něco se nazývá (linuxová) distribuce.

Linuxová distribuce je rozsáhlá sada software, jejíž základem je i OS Linux. Jak moc rozsáhlá, záleží na distribuci. Například RedHat Linux (zkráceně jen RedHat, což je také název firmy, která tuto distribuci vytváří) se dodává v zásadě na 3 CD (+2 CD se zdrojovými kódy), SUSE (komerční) na 7 CD, Slackware (nekomerční, nejstarší) na 2 CD a Debian (nekomerční) na 7 CD. Na těch CD není žádná vata jako 500 MB velký film průvodce instalace, jsou nabita kvalitním softwarem.

Samozřejmě ani zdaleka neinstalujete všechno, při instalaci si vybíráte, které programy z těch CD chcete nainstalovat. Mnohdy je na nich několik alternativ stejné věci, třeba textových editorů. Můžete si nainstalovat všechny, nebo také žádný, jak je vám libo. Zde to má začátečník opravdu těžké, protože sice při výběru ví že tohle je textový editor a tohle je správce oken, ale to je asi tak všechno. Neví který z těch textových editorů nebo správců oken mu bude vyhovovat a který ne. Mnohdy ani neví, k čemu ta věc (třeba ten správce oken) je zač. Naštěstí většina distribucí má při instalaci možnost volby defaultní instalace nebo defaultní desktop, při které je připraven vhodný výběr komponent.

Při kvalitní nabídce aplikací v distribuci máte po instalaci distribuce hotovo. A můžete se pustit do konfigurace systému. To je zásadní rozdíl oproti Windows, kdy po jejich instalaci musíte instalovat pracně hromadu dalšího software (kancelářský balík, grafický editor, total commander a hromadu dalších nezbytností. Dále ovladače pro svůj HW a podobně. V ideální distribuci je už tohle všechno přítomno a tedy i nainstalováno. Ideální distribuce samozřejmě neexistuje, ale některé se tomu blíží a práce po nainstalování linuxové distribuce je skutečně mnohem méně, než po nainstalování Windows.

Kompatibilita a rozdíly mezi distribucemi

Hardware

Rozdíly mezi distribucemi jsou i nejsou velké. Záleží na tom, jak hluboko uživatel do Linuxu pronikl a na jaké úrovni se v něm orientuje. Začneme odspoda. Nejníže je jádro, které dělá Linux Linuxem. Všechny distribuce používají stejné jádro. Možná trochu odlišnou verzi (jádro lze snadno upgradovat, jestliže máte potřebu novější verze), podle doby vydání, možná trochu jinak opatchované, ale pořád jedno a to samé jádro. Zde jsou rozdíly minimální. To je důležité vědět. V Linuxu je podpora mnohého hardware závislá na jádře. Začátečníci se občas ptají, v které že distribuci je podporována jejich XYZ síťová karta. To je špatná otázka. V 99,99 % případů to je ve všech nebo v žádné. Je nutno se ptát zda a od které verze jádra je podporovaná.

Ano, může se stát, že v jedné distribuci je o chlup novější jádro než v druhé a že tento chlup stačí k tomu, aby v jedné distribuci ta karta fungovala a v druhé ne, ale to nevadí. Jádro se dá velmi snadno vyměnit za novější, tohle by tedy nemělo být rozhodující pro výběr, kterou distribuci použít. Mezi distribucemi jsou jiné, zásadnější rozdíly, dle kterých bychom měli vybírat.

Ještě k tomu dodám jednu věc. Může se stát, že v jedné distribuci je nějaký kus hw

automaticky nalezen a použit a v jiné ne. To neznamená, že v této distribuci nefunguje, jen nebyl automaticky nalezen. To může mít různé příčiny. Třeba nedokonalá konfigurace jádra. V Linuxu také existuje program, který hlídá zda se v počítači neobjevil nový hardware (především různé periferie, základní hw nutný k chodu počítače si jádro musí obsloužit samo) a který jej i konfiguruje. Někdy automaticky, někdy za pomoci uživatele. Různé distribuce používají různé tyto programy a nebo taky žádné. V RedHatu se tento program nazývá kudzu. Je jasné, že různé programy mají různé schopnosti detekce hw a tak to že někde nebylo něco samo nalezeno opravdu nic neznamená, a pokud je zařízení podporováno, lze jej přinejhorším zprovoznit ručně. Praktický je to tak, že v každé distribuci vám může fungovat kterýkoliv hardware který funguje v jiné, jen v některé může jeho zprovoznění stát větší úsilí (např. zmíněný upgrade jádra, a nebo také jen připsání jednoho řádku do /etc/modules.conf). Rád bych zdůraznil, že ne všechen hardware je závislý na jádře. Např. s grafickými kartami toho jádro moc společného nemá, ty má na starosti X Server a podpora vaší grafické karty je tak závislá na její podpoře v X Serveru. Hardware v Linuxu se budu věnovat ještě samostatně.

Správa

Nad jádrem je operační systém. To je vlastně taková softwarová výbava která poskytuje různé služby uživatelský aplikacím, a která umožňuje počítač konfigurovat a ovládat. Základní uživatelské rozhraní Linuxu je konzole a v ní příkazový řádek. Konzoli (tedy obrazovku v textovém režimu a klávesnici) obhospodařuje samo jádro, příkazový řádek je program, který se také označuje jako shell. Pokud znáte základní příkazy pro správu a konfiguraci Linuxu, můžete na této úrovni Linux docela dobře ovládat. Na této úrovni se distribuce od sebe navzájem také příliš neliší. Mohou nabízet vlastní příkazy, které práci usnadňují a jinde se nevyskytují, ale rozhodně na nich nebudete závislí. Existuje základní sada standardních příkazů, která je všude a ovládáte-li ji, umíte ovládat i ostatní distribuce. U některých distribucí (typicky různých mini) to je dokonce jediný možný způsob ovládání.

Ovládat kompletně Linux skrz příkazový řádek je značně náročné na znalosti. Musíte hluboce rozumět Linuxu, sítím a všemu ostatnímu. Proto vznikla řada utilit, jak s textovým tak grafickým rozhraním, které vám to usnadňují. A zde jsou už rozdíly mezi distribucemi zásadní.

Některé utility (xf86config ??) jsou součástí programů (X Server) a ty naleznete také v každé distribuci. Jiné utility, především ty, které slouží ke konfiguraci samotného OS, byly ale vytvořeny samotnými tvůrci distribucí (redhat-config-keyboard) a ty jinde než v konkrétní distribuci nenajdete. Ne vždy musí být pojmenovány tak inteligntně, aby bylo poznat co je univerzální utilita a co utilita specifická jen pro nějakou distribuci. Utility dostupné jen v nějaké distribuci komplikují lehce uživatelům život, protože to že si umíte levou zadní nakonfigurovat RedHat ještě neznamená, že dokážete nakonfigurovat SUSE, ale neznamená to ani, že to nedokážete. Bude záležet na tom, na jaké úrovni (pomocí kterých nástrojů) ho umíte Linux nakonfigurovat. Pokud umíte síťové rozhraní nakonfigurovat pomocí příkazu ifconfig, dovedete ho nakonfigurovat ve všech Linuxech, protože tento příkaz je součástí všech distribucí, je to skoro součást jádra. I proto zkušení uživatelé dávají přednost příkazovému řádku.

A nakonec, třeba zmíněná grafická utilita redhat-config-keyboard nakonec stejně kromě jiného (úpravy cfg. souborů) používá univerzální příkazy setxkbmap, loadkeys a sysfont. Její výhodou je, že je pohodlnější, že ji stačí spustit a kliknout, že nemusíte vědět jaké konfigurační soubory je potřeba upravit, jaké příkazy použít aby se změna provedla ihned a dokonce ani, že na konzoli se klávesnice konfiguruje jinak než v GUI. Nevýhodou je, že když přijdete do styku s jinou distribucí, tak budete muset pátrat, jak se to dělá v ní. Vysvětlit proč každá distribuce má více (RedHat/Debian) či méně

(RedHat/Aurox (v Polsku lehce upravený RedHat)) jiné utility není jednoduché, ale zkusím to.

Konfigurace bývá někde uložena, ve Windows to bývají registry, v Linuxu konfigurační soubory. Když se spouští nějaký program, načte konfigurační soubor a podle něj se nastaví. To samotné se děje i když se spouští operační systém. Spouštění operačního systému představuje spuštění jádra, jeho konfiguraci a spuštění řady podpůrných aplikací. Jádro ale nemá žádný konfigurační soubor. Jádro se konfiguruje jen pomocí příkazů. Původně to bylo tak, že při najíždění (UNIXového) OS se spustil skript, v kterém byly tyto konfigurační příkazy a také příkazy pro spuštění různých podpůrných aplikací, serverů a podobně. (Skript je textový soubor, v kterém jsou zapsány příkazy, které se postupně vykonávají.)

Později se pro vyšší přehlednost takový skript rozdělil na víc částí. Dejme tomu na konfigurační, a ten co spouští podpůrné programy. Ty se pak zase mohli dělit na části, jako je konfigurace sítě, souborových systémů a podobně.

Takto nějak vznikla jednoduchá soustava skriptů, která měla na starosti najíždění operačního systému. Mělo to ale jeden velký háček, skript lze jen velmi obtížně upravovat automaticky nějakým programem. Takže každá, i drobná, změna konfigurace vyžaduje ruční editaci člověkem. Zkušeným uživatelům to docela i vyhovuje a proto to některé systémy pro profsionální nasazení používají dodnes. Ty proto, že když víte jak věci fungují, tak ty skripty jsou průhledné a velmi flexibilní, máte tak dokonalý přehled a kontrolu nad tím, co se ve vašem systému děje.

Problém je to pro lidi, kteří neví jak věci fungují, ti dávají přednost konfiguračním nástrojům, které to ví za ně. Nejsou sice tak flexibilní, ale to je něco za něco. Aby bylo možno konfiguraci měnit automatický pomocí konfiguračních nástrojů, musely být ty skripty výrazně předělány. Byly rozděleny na mnoho malých skriptů, konkrétní konfigurační hodnoty nejsou přímo v nich, ale v dalších datových souborech, které lze programově měnit snadněji. Navíc ty skripty musí být inteligentnější, musí mít ty znalosti, které chybí uživateli a musí se automaticky a logicky rozhodovat na základě různých údajů. Díky tomu jsou velmi složité (jsou to už vlastně víc programy než skripty) a nepřehledné i pro zkušeného uživatele. Už nejsou určeny pro uživatele. Ale datové soubory, které používají, lze snadno měnit pomocí grafických utilit, jenž mohou snadno používat i nezkušení uživatelé.

To všechno se nestalo najednou, ale postupně. Mezi jednotlivými distribucemi hořel a dodnes hoří (přátelský) boj o uživatele. Mnohé distribuce, zvláště ty komerční, se předhánějí v tom, kdo nabídne uživatelsky nejpřívětivější možnosti správy své distribuce. Proto spolu nespolupracují a výsledkem je každá ves jiný pes. Navíc různé distribuce zastávají různé filozofické náhledy na to, co je nejlepší, takže to ani nelze sjednotit, vždy to bude někomu nevyhovovat. Možnosti a způsob konfigurace OS je jeden z největších rozdílů mezi linuxovými distribucemi. Je to jedno z hlavních kritérií, podle kterých byste si měli tu svou "nejlepší" vybírat. Pozor ale, není jediné.

Slackware zůstal u jednoduchého systému skriptů a dělá dobře. Zkušení administrátoři to vítají, Slackware je tak rychlý a flexibilní OS. RedHat dospěl ke složitým, těžko srozumitelným skriptům, které se snadno automaticky konfigurují, třeba i při instalaci nového software. RedHat měl také komplexní konfigurační nástroj LinuxConf, který se mu ale neosvědčil a tak se vrátil se k jednoduchým konfiguračním utilitkám, které lze spouštět samostatně s textovým i grafickým rozhraním, nebo z ovládacího centra podobnému tomu ve Windows. To udělal moc dobře, na vzdálenou správu je pro administrátory vhodný ten textový režim a pro ostatním je tu ten grafický. Firma SUSE dospěla k podobě složitým skriptům jako RedHat, ale nezůstali jen u toho, vytvořili komplexní ovládací centrum Yast2, které umožňuje jednotně konfigurovat nejen

samotný OS, ale i mnoho dalších aplikací. To je dobře, protože to velmi usnadňuje používání Linuxu i těm největším začátečníkům. Něco podobného vytvořil i Mandrake, kterého doporučují spíš než SUSE, protože SUSE se k uživatelům chová občas dosti divně a pro Mandrake existuje na internetu asi nejvíc připravených instalačníc balíčků s aktuálním software.

Jak jste si asi všimli, pochválil jsem všechny. Protože všichni to mají uděláno dobře, jen každý pro jiné potřeby/uživatele. Můžete si vybrat, který přístup vám vyhovuje. Je to jeden z klíčových prvků, podle kterého se vybírá distribuce. Osobně se domnívám, že RedHat zvolil rozumný kompromis, zlatou střední cestu, ale to je jen moje preference. Každopádně vezměte na vědomí, že administrace na této úrovni se distribuce od distribuce výrazně liší. To že si můžete vybrat přístup který vám vyhovuje je dobrá věc. Je jasné, že domácí uživatel a profesionální administrátor, který má na starosti mnoho vzdálených systémů mají odlišné potřeby. To že laický uživatel RedHatu pak nedokáže poradit uživateli SUSE jak si nakonfigurovat síť je stinná stránka této věci. Linuxový začátečník by to měl brát v úvahu výběru své distribuce. Jestliže všichni okolo mají zkušenosti s RedHatem, měl by k tomu přihlédnout, usnadní si život, s Debianem mu tito lidé moc nepomohou.

Software

Nad operačním systémem běží různé programy. U některých programů se těžko určuje zda to je ještě součást OS nebo už ne. Zde budu liberální a za software běžící nad OS budu považovat i některé systémové věci jako grafické rozhraní, které je v Linuxu narozdíl od Windows nepovinné a je zde navíc volba různých desktopových prostředí a podobně. Samozřejmě sem patří i aplikace. To je to, kvůli čemu uživatelé počítače vlastně používají. Tedy programy které tu nejsou proto aby ten počítač fungoval a dal se používat, ale proto, že něco užitečného dělal. Tedy různé textové a grafické editory, webové prohlížeče, poštovní programy, hry, přehrávače hudby a videa a tak dále a tak podobně.

Distribuce v tomto ohledu samozřejmě nejsou identické, ale ani extrémně rozdílné. Tedy, srovnatelné distribuce. Distribucí je obrovské množství, určitě přes 100 ??odkaz na seznam??. Linuxová distribuce může být jednodisketová, třeba specializovaná jako router, i univerzální uživatelská, která se nachází na 7 CD jako SUSE. Je jasné, že rozdíly v jejich aplikační i další výbavě jsou maximální a nikoho to nepřekvapí. Když tedy srovnávám distribuce, míním tím ty vhodné pro většinové uživatele do kanceláří a domů (nazývají se také desktopové).

Těch zase tolik není. U nás (ČR) jsou na desktopu obvyklé Debian, Mandrake, RedHat a SUSE. V roce 2003 se nekomerční část RedHatu (to co používala většina domácích uživatelů) přetransformovala na Fedora Projekt, který vydává pravidelně Fedora Core, což je distribuce, na které RedHat zpětně hodlá stavět svou komerční distribuci (Fedora vlastně funguje jako testovací verze) a na oplátku ji za to sponzoruje. Takže když tu mluvím o RedHatu, tak to platí i pro Fedoru. V Polsku zase udělali klon RedHatu, který se jmenuje Aurox. Je to v podstatě RedHat9, ale s aktuálními opravnými balíčky a přidanými aplikacemi (především multimédia), takže je to lepší něž RedHat proto6e je to RedHat + něco navíc. Aurox používám i já sám. Když tedy mluvím o RedHatí distribuci, myslím tím nejen Fedoru, ale i Aurox a možná další klony, které ani neznám. RedHat je linuxový leadr a tak má mnoho různých klonů. Ostatně na RedHatu byly založeny i distribuce jako SUSE nebo Mandrake. Je to už ale dávno, takže se stačili dosti odlišit. Přesto si jsou RedHat, Mandrake a SUSE podobnější, než třeba distribuce Debian nebo Slackware, které jdou svou vlastní cestou od začátku.

Na servery se používají často Debian, RedHat a Slackware. Poslední dobou také občas slýchávám o Gentoo, který si jde také vlastní cestou. Všechny tyto distribuce jsou

velké, největší je SUSE, Mandrake a Aurox se 7 CD, Debian a RedHat mají po 5 CD, u RedHatu z toho jsou 2 CD s dokumentací a zdrojovými kódy, Aurox má 3,5 CD se zdrojáky a dokumentací (u distribucí kde to neuvádím to nevím), Slackware má "jen" 2 CD (bez zdrojáků). Uživatele Windows mohou tyto počty překvapit, a to i když už ví, že distribuce není jen samotný OS, ale i aplikační výbava.

Je samozřejmé, že nemusíte instalovat všechno. Při instalaci Linuxu si vybíráte, jaký software si k němu nainstalujete. Poté můžete kdykoliv další nainstalovat, nebo ten stávající odinstalovat, není-li to nezbytná součást systému. Nezbytné části systému se mezi distribucemi liší. U Slackware je to prý (bez záruky) 40 MB, u RedHatu je to desetkrát víc, zkušený uživatel si ale poté může "zbytečné nezbytnosti" jako je dokumentace smazat.

Takové ty základní nebo dobré a oblíbené linuxové aplikace naleznete ve všech distribucích. Co tam nenajdete, to si obvykle můžete stáhnout z internetu. Některé komerční aplikace, jako třeba databázový server Oracle je podporovaný jen na některých distribucích. Ten Oracle myslím jen na RedHatu. V tomhle směru je RedHat nejlepší volba, v komerčním světě je uznávaný a nejvíce podporovaný. Neznamená to, že ten Oracle nebude fungovat i jinde, jen vám to nikdo nezaručí a výrobce vám neposkytne žádnou podporu.

Zde je nutno zmínit jeden velmi nepříjemný fakt. Kompatibilita aplikací mezi různými distribucemi je velmi špatná a nekompatibilní způsoby instalace aplikací tento stav ještě zhoršují. Dokonce i kompatibilita aplikací mezi různými verzemi jedné distribuce není nijak slavná. Windows nemají různé distribuce, ale verze ano. Ani u nich není kompatibilita ideální, ale Linux je v tom nesrovnatelně horší. Tohle je asi největší chyba a brzda Linuxu.

Aby bylo jasno, mluvím o binární kompatibilitě. Na úrovni zdrojových kódů je kompatibilita dobrá, na stejné úrovni jako ve Windows ta binární. Důsledkem toho je, že vývojáři programů musí pro každou distribuci a popřípadě i její verzi kompilovat program zvlášť. To je samozřejmě pracné, je také potřeba aby měl příslušnou distribuci nainstalovánu, takže není divu, že to nedělají pro všechny distribuce, ale jen pro některé. K tomu poskytnou zdrojové kódy a uživatelé ostatních distribucí si musí program přeložit sami nebo hledat na internetu, zda už to někdo pro jejich distribuci neudělal a výsledek své práce nevystavil na internetu. Nejlíp jsou na tom uživatelé RedHatu, pro ty většinou program přeloží už samotný autor a když ne, tak je na internetu hodně míst, kde se dají přeložené programy sehnat.

Jak již jsem se zmínil, kompatibilní nebývají ani jednotlivé verze stejné verze. Např. v RedHatu je to tak, že verze jedné řady jsou navzájem binárně kompatibilní. Tedy verze 6.0, 6.1 a 6.2 jsou navzájem binárně kompatibilní, zrovna tak i verze 7.0, 7.1, 7.2 7.3 (jednotlivé verze vychází pravidelně po půl roce). Přitom do RedHatu 7.x je možno nainstalovat programy určené pro 6.x (je zachována zpětná kompatibilita), ale musíte mít pro to nainstalovánu speciální podporu (je součástí distribuce, v podstatě se jedná o starší verze systémových knihoven). U následující verze změnil politiku číslování, takže tato má číslo 8 a další (poslední) 9. Nevím jestli jsou kompatibilní, já používám stále verzi 7.3, ale asi ano. Programy se na internetu zatím obvykle objevují pro verzi 7.3 i 8, ale čím dál častěji budou jen pro 8 a já tak budu nucen si programy překládat sám a nebo upgradovat na vyšší verzi.

Binární kompatibilita je dána knihovnami, které v Linuxu čas od času prochází reedicí a přestávají být kompatibilní. Jsou číslovány podobně jako distribuce RedHat, takže je zřejmé které jsou kompatibilní a které už ne. V Linuxu (na rozdíl od Windows) je možno do systému nainstalovat víc verzí stejné knihovny, takže až na újmu na diskové kapacitě to není problém. Ve Windows se to dá omezeně řešit tím, že každá

aplikace může mít svou verzi knihovny. Tuto ale nemůže už sdílet s jinými aplikacemi které ji potřebují, takže to stojí ještě více diskové kapacity. Jedna a ta samá verze je nainstalována víckrát. To by ani tak nevadilo, horší je, že to spotřebovává i víc paměti RAM, protože se do ní stejná knihovna zavádí víckrát. Dále je binární kompatibilita dána překladačem, který binární kód produkuje. Ten je také neustále vylepšován a čas od času vznikají verze, které produkují nekompatibilní kód. S tím se toho moc dělat nedá, jedině používat pořád starý překladač, což je také možné, ale nepraktické. Ve Windows je tohle řešeno asi lépe a k něčemu takovému nedochází.

Proč tomu tak je? Má to minimálně dvě příčiny. Za prvé je nutno říct, že Linux nemůže nikdy být plně binárně kompatibilní, dokud tu bude existovat více architektur procesorů a Linux je bude podporovat. Binární podoba programu je totiž podoba, které rozumí procesor počítače. Existuje několik různých architektur procesorů, Windows podporují jenom jednu x86 (no dobře, tak dvě, ještě ia64, nový nekompatibilní procesor Itanium od Intelu), Linux jich podporuje několik, (cca 7??). Program přeložený pro jednu architekturu nemůže běžet na jiné, protože tyto jsou také nekompatibilní (to se týká i těch Windows, pro ia64 potřebujete speciální verzi Windows a veškerého software, který je pro tento procesor přeložený, takže tam toho software moc není, je to určeno pro servery, ne desktop). Za druhé, Linux je UNIXová platforma, na zdrojové úrovni je vcelku dobře kompatibilní s jinými operačními systémy typu UNIX. To jest napíšete program pro Linux a ejhle, on bude fungovat i na Solarisu, FreeBSD, AIXu, a podobně. Stačí ho zde přeložit. No, zas tak snadné to není, jde napsat i nekompatibilní program, ale při troše snahy a šikovnosti to možné je a mnohdy se tak i děje. Při hodně snahy a šikovnosti to bude fungovat po překladu i ve Windows.

Díky tomu může jeden linuxový program fungovat na mnoha operačních systémech a v nich na mnoha procesorových architekturách, které nejsou binárně kompatibilní a tak chtěj nechtěj je nutno poskytnout k zdrojové kódy, protože nikdo nedokáže udělat tolik různých binárek. Vzhledem k tomu že Linux je postaven na sdílení zdrojových kódů, tak to nikomu nevadí. A snad právě proto se pak nehledí na binární kompatibilitu ani tam, kde by být mohla. Vždyť přeci máte zdrojáky a můžete si je přeložit, ne?

Obávám se že ne. Zaprvé, překlad programu není triviální záležitost, zvláště těch rozsáhlých jako je GNOME, KDE, XFree86, OpenOffice.org a podobně. Za druhé, člověk který je zvyklý párkrát kliknout si nepřeloží vůbec nic. Za třetí, to nedělá dobrotu v systémech, které si hlídají závislosti (o tom budu psát ještě podrobně). Za čtvrté kvůli tomu musíte mít nainstalovány rozsáhlé vývojové nástroje, které zabírají stovky MB diskové kapacity. Za páté překlad programu dlouho trvá, u velkých programů to jsou řádově hodiny, a to je špatné.

Situace ale zase není až tak černá jak jsem ji teď vylíčil, třeba OpenOffice.org nemají binárky pro každou linuxovou distribuci zvlášť a přesto v nich fungují. Mezi distribucemi ze stejného období nejsou zase až tak velké rozdíly. Používají stejný překladač, sdílené knihovny a podobně. OpenOffice.org tedy sdílené knihovny moc nepoužívá, protože to je multiplatformní software, který si nemůže dovolit závislost na platformě (OS), platí za to ale velkou spotřebou paměti, takže to není úplně nejlepší přístup. To samé se týká třeba i Mozilly. Největší problémy asi působí nekompatibilitou samotné instalace software v různých distribucích. V případě jednoduchých programů je možno např. v RedHatu nainstalovat program z instalačního balíčku pro Mandrake (oba systémy používají RPM systém), ale obecně to není dobrý nápad, zvláště u systémových záležitostí nebo u balíčků, které používají pre a post instalační skripty.

Potíže s kompatibilitou software je podle mně největší hřích Linuxu. Obtěžuje jak uživatele (a tím brzdí rozšíření Linuxu) tak vývojáře (čímž brzdí vývoj softvare pro

Linux). Významně to omezuje komerční tvůrce software, kteří nejsou ochotni se o své produkty a know how dělit zdarma a tedy zveřejňovat zdrojové kódy programů. Extrémně se to týká ovladačů, které lze v binární podobě distribuovat jen velmi omezeně.

Výrobci hw nejsou ochotni dodávat ovladače v podobě zdrojových kódů a dodávat binární verzi pro každou verzi jádra a distribuci se jim nevyplatí. Zde to tedy není tak úplně chyba Linuxu, ale záměr vývojářů jádra. Mohli by vytvořit v jádře rozhraní pro binární ovladače jako to mají Windows, ale vývojáři jádra to z ideologických či politických důvodů odmítají udělat, ač o to byly mnohokráte žádáni. Nechtějí aby součástí jádra byly černé skříňky o kterých nikdo neví co dělají.

Instalace software

Linux nevytváří nějaká konkrétní společnost, naopak, je vyvíjen tisíci programátory z celého světa. Shánět a kompilovat si všechny programy by bylo velmi náročné. Proto také vznikly distribuce, abyste toho byli ušetřeni. Ale i když program máte a máte ho přeložený, není všemu konec. Ještě ho musíte nainstalovat. V DOSu ho stačilo zkopírovat do nějakého adresáře a bylo to. V modernějších OS to už tak jednoduché není. Všechny programy dělají některé stejné věci, otevírají soubory, pracují s pamětí a tak podobně. Programátor takovéhle věci neprogramuje přímo, děsně by se u toho nadřel a nic by pořádně neudělal. Místo toho používá knihovny funkcí. Pak, místo toho aby celou záležitost pracně programoval jen použije připravenou funkci, která udělá všechno potřebné.

V DOSu se taková knihovna připojila přímo k programu. DOS byl jednoduchý, knihoven funkcí bylo málo a byly malé, navíc v něm mohl být spuštěn jen jeden program. Dnes je jiná doba, knihovny zabírají desítky MB (umí toho víc a jsou navíc 32 nebo i 64 bitové, což jejich velikost zvetšuje 2x nebo 4x), programů je hodně a hodně jich bývá i spuštěno současně a paměť počítače není nekonečná. Proto je nutno jejich velikost co nejvíc minimalizovat. Děje se tak používáním sdílených knihoven. Místo aby se knihovna připojila k programu, je v samostatném souboru. Když je knihovna potřeba, nahraje se z disku do paměti a programy které její funkce potřebují ji sdílí, je tak na disku i v paměti jen jednou a nezabírá tolik místa. To je dobrá věc.

Ale nic není zadarmo. Používání sdílených knihoven přináší své problémy. První problém je instalace programů, druhý kompatibilita jednotlivých verzí knihoven. Když dva programy používají stejnou knihovnu, jak je nainstalovat? V Linuxu se programy dodávají bez sdílených knihoven, knihovny se dodávají a instalují samostatně.

To má své dobré důvody. Autor programu je obvykle někdo jiný než autor knihovny. Uživatelé Linuxu často stahují programy z internetu a knihovny mohou zabírat postatnou část a někdy i většinu velikosti programu. Proč je tedy stahovat neustále dokola s každým programem, když to stačí jednou.

Ve Windows se obvykle knihovny přikládají ke každému programu. Když v systému není, nainstaluje se, když ne, tak záleží na tom. Instalátor si může zjistit zda je novější verze a když ano, přeinstaluje tu starší, nebo se zeptá uživatele, který obvykle neví oč kráčí a nebo prostě bez pardonu přepíše tu starší. Různé verze knihoven spolu nemusí a často nejsou kompatibilní, takže jedna z těch aplikací (která nemá tu svoji verzi knihovny) pak může být nestabilní. K tomuto skutečně dochází, projevit se to může známou chybou Aplikace vykonala nepovolenou instrukci.

Také je možno dotyčnou knihovnu nainstalovat do adresáře té aplikace, ale pak jsou veškeré výhody sdílených knihoven v trapu. Říká se tomu dll hell (dll peklo, dll je přípona souboru sdílených knihoven ve Windows, je to od dynamic link library - dynamicky připojované knihovny). V Linuxu se nic takového neděje, nejen že se

sdílené knihovny instalují zvlášť, ale hlavně na to má jednoduchou, ale účinnou metodu.

Číslo verze knihovny je součástí názvu souboru, takže dvě různé verze si nekolidují a mohou být v případě nekompatibility nainstalovány společně. Pak může padesát programů používat jednu verzi a padesát druhou. Je to lepší než když jich padesát používá jednu a dalších padesát každou svoji, což je jediné možné řešení dll hell ve windows.

Linux se tak vyhnul so hell (so jsou přípony souborů sdílených knihoven v Linuxu, je to od shared objects - sdílené objekty), ale také má své vlastní peklo, dependancy hell. Peklo závislostí. Jde o to, že program se nešíří jako jeden balíček, ale několik balíčků. Když si nainstalujete program bez knihovny, nebude fungovat. Znamená to, že program je na knihovně závislý. V praxi programy používají více různých knihoven, takže nemusí být snadné je všechny podchytit. Zvláště když každá knihovna má jiného autora a je na internetu někde jinde a musí se přeložit a přitom sama zas potřebuje jiné knihovny a některá z těchto jiných může potřebovat ještě něco jiného. Z instalace jednoho prográmku se pak může stát dlouhotrvající boj při kterém dostane k instalaci desítek MB jiných balíčků. Není to úchylné, jestliže např. máte nainstalován Linux jen s textovým rozhraním a chcete si nainstalovat nějaký GUI prográmek, je logické, že to bude vyžadovat instalaci celého grafického prostředí včetně fontů a podobně a to se nasčítá.

Jaké knihovny program potřebuje se dá zjistit snadno, jiné závislosti se zjišťují už hůř. Normální uživatel to není schopen podchytit. Právě proto vznikly linuxové distribuce a s nimi různé systémy instalace programů. Ano, slyšíte dobře, různé systémy, samozřejmě že nekompatibilní. Jedno ale mají společné, programy se neinstalují sami jako ve Windows. Program je zabalen do jednoho nebo více instalačních balíčků, jakýchsi datových souborů. V Linuxu je pak speciální program, který tyto balíčky instaluje sám. Je to něco jako ve Windows zkomprimované balíčky zip, které se rozbalují nějakým programem, např. WinZipem, ale instalační balíčky jsou složitější.

První s tím přišel Slackware (nejstarší distribuce). Jeho balíčkovací systém ale byl a je velmi primitivní. Závislosti např. nijak neřeší. Jeho instalační balíčky jsou v podstatě jen přeložené a zkomprimované programy. Nástroj na jejich instalaci je (a jejich obsah) registruje a umí je proto i odinstalovat. Ve své době to byl velký pokrok, už to že jste si nemuseli program překládat a šel snadno odinstalovat byl úspěch.

S dalším řešením přišel RedHat. Vytvořil RPM (RedHat package manager, RedHatí správce balíčků) a dal ho volně k dispozici. Dnes je nejrozšířenější a používají ho téměř všechny ostatní distribuce. RPM je lepší, umí hodně věcí, hlídá závislosti, udržuje databázi všech nainstalovaných souborů, takže se můžete RPM dotázat do kterého balíčku patří tenhle soubor a jaké soubory k tomuto balíčku ještě patří a kde jsou (dobré, když třeba nevíte kde je dokumentace k nějakému programu), umí balíčky digitálně podepisovat a podpis při instalaci ověřovat. Balíčky mohou být binární i zdrojové, tj. software lze vcelku pohodlně instalovat i ze zdrojových kódů (hlídání závislostí pořád aktivní) a podobně.

Své vlastní řešení má i debian. Ten odmítl RPM z ideologických důvodů. Je zásadně nekomerční a odmítá používat produkty komerčních firem a to i když jsou volně k dispozici. Tímto se ale hloupě omezuje a já ho proto nemám rád. Jeho balíčkovací systém je velmi podobný RPM. Dlouhou dobu se mohl chlubit výborným řešením závislostí. Mohli jste mu říct nainstaluj tento balíček a on už k němu dohledal všechny potřebné balíčky a také je nainstaloval. RPM umělo říct jenom, že tento balíček ke své činnosti potřebuje to a to a sehnat a nainstalovat jste to museli sami. Debian tohle umožňuje díky centralizovanému archivu všech balíčků na internetu. Lidé s pomalým

nebo drahým připojením toho tedy moc nevyužijí. V současné době podobné nástroje existují už i pro RPM, je to ale hodně závislé na distribuci, protože to vyžaduje centrální archiv balíčků a každá distribuce má vlastní instalační balíčky. Debian je v tom určitě nejdál.

Tyto nekompatibilní instalační systémy jsou další důvod, proč je pro každou distribuci překládat program zvlášť. Tedy, přesněji řečeno musíte pro něj udělat vlastní balíček. A to i pro Mandrake nebo SUSE, kteří používají RPM. Instalační balíček může totiž obsahovat konfigurační skripty, které se distribuce od distribuce liší, distribuce mohou mít rozdílné umístění souborů a nebo být binárně nekompatibilní.

Ve všech distribucích pořád můžete problémový program přeložit ze zdrojových kódů a nainstalovat jej přímo. To ale není dobré, protože tím obejdete instalační systém. Za prvé už pravděpodobně nepůjde odinstalovat, leda ručním smazáním souborů, když budete vědět které to jsou. Za druhé vám přestane fungovat kontrola závislostí. Protože ta se kontroluje v databázi nainstalovaných programů, kam se přímo nainstalovaný program nedostane.

Dejme tomu, že jste si takto nainstalovali knihovnu, kterou jste jinak nemohli sehnat. Program který ji potřebuje ale i nadále nepůjde nainstalovat, instalátor neví, že ta knihovna už někde v systému je. Můžete mu přikázat, že má nevyřešené závislosti ignorovat a program nainstalovat "násilím". A všechno bude fungovat. Možná. Možná že ten program potřeboval ještě jinou knihovnu, což jste netušili a která v systému skutečně chybí. Když ten program nepůjde spustit, bude to ten lepší případ, horší bude, když bude jen občas padat a vy ztrácet data. Zvláště začátečníci by neměli programy instalovat a odinstalovávat násilím, protože ti nevědí co dělají. Přitom právě oni to nejčastěji, často bezelstně dělají. Když se jich zeptám proč, tak proto, že jim to někdo poradil aniž by jim vysvětlil důsledky, a že jim to bez toho nešlo nainstalovat. A že to pak nefunguje? Blbej program nebo rovnou Linux :-). Proto se snažte pokud možno dodržet následujících doporučení:

- Všechno instalujte jako balíčky. Dejte si tu práci a najděte si je na internetu, na 90 % někde budou, používáte-li RedHat, Mandrake nebo Debian.
- Nikdy, a to opravdu nikdy, nevypínejte kontrolu závislostí. Budete-li se držet prvního pravidla, vždy se bez toho obejdete.
- Nainstalované programy z disku nemažte. Vždy je odinstalujte. Samozřejmě s dodržením závislostí.
- Když už si budete program sami překládat, na jeho instalaci použijte geniální Check Install, který z přeloženého programu udělá balíček. Podporuje RPM, Debian i Slackware.
- Přeložené programy neinstalujte do systému, ale na nějaké speciální místo. Třeba vašeho domácího adresáře nebo do adresáře /opt.

Pro zkušeného uživatele je to normální, ale začátečník si dokáže tak zprasit systém, že škoda mluvit. Tímto způsobem se dá ze skutečně stabilního systému udělat systém velmi nestabilní. Proto je vhodné uvedená pravidla dodržovat, vyplatí se to.

Souborové rozhraní

Tato kapitola se vám bude snažit vysvětlit jednoduchou skutečnost, a to tu, že v Linuxu, po vzoru UNIXu, se v podstatě všechno tváří jako soubor. Soubor představuje jednoduché a univerzální rozhraní k mnoha rozličným věcem. Například k sériovému portu. Ve Windows je sériový port znám např. jako jakési COM1 visící ve vzduchu, s kterým nelze pořádně pracovat, lze jej zadávat pouze jako parametr v některých konfiguračních dialozích.

V Linuxu je znám tento port jako speciální soubor /dev/ttyS0. Také ho můžete zadávat jako parametr v konfiguračních dialozích. Ale kromě toho s ním můžete pracovat jako se souborem. Můžete ho otevřít a zapisovat do něj, tím budete zapisovat data na tento port. Nebo ho naopak můžete číst a tím budete číst to, co na něj zapisuje nějaké zařízení k němu připojené, třeba myš. Takhle jednoduše funguje ovladač myši. Jeho programátor otevře tento soubor jako každý jiný a z přicházejících dat (pokud zná protokol, kterým myš komunikuje, tedy zná formát dat, které myš na port zapisuje) rozpozná, co to s tou myší děláte.

To byl jen příklad, o souborech reprezentující různá zařízení budu podrobněji ještě psát. Další příklad je samotné jádro. To s uživatelem nebo programátorem komunikuje pomocí spousty souborů ve virtuálním adresáři /proc. Podívejte se třeba na /proc/cpuinfo, dozvíte se z něj, jak jádro vnímá váš procesor a co o něm ví. Když programátor potřebuje zjistit frekvenci vašeho procesoru, přečte si to taky tady. Linux je v tomhle směru velmi jednoduchý a průhledný. To že je /proc virtuální znamená, že neexistuje nikde na disku, je jen v paměti. Obsahy těch souborů nejsou nikde, jádro je vygeneruje až při pokusu o jejich čtení podle aktuálních hodnot.

O všem budu podrobněji hovořit později, teď stačí, když vezmete na vědomí, že v Linuxu nemusí být to, co se na první pohled jeví jako soubor nebo adresář skutečný soubor nebo adresář tak, jak je znáte z Windows. To, že se mnoho věcí prezentuje jako soubor je jeden ze základních stavebních kamenů Linuxu (a UNIXu obecně). Později uvidíte, že v Linuxu existuje mnoho speciálních druhů souborů. Linux na rozhraní souborového systému staví.

Úvod do souborových systémů

Počítače pracují s daty. Data jsou uložena na nějakých zařízeních či médiích. Zajisté je znáte, jsou to pevné disky, diskety, cd nebo dvd média, paměťové karty, usb disky, fotoaparáty a tak dále a tak podobně. Všechny vyjmenované věci mají něco společného, souborový systém. Linux narozdíl od Windows podporuje mnoho souborových systémů. Uživatel Linuxu musí vědět o souborových systémech víc než uživatel Windows, v Linuxu je souborový systém základní a důležitá část systému, více než ve Windows. Uživatel Windows také musí něco vědět o souborovém systému, mnohdy však ani neví, že o něm něco ví. Ve Windows slouží souborový systém k ukládání dat. V Linuxu plní ještě další neméně důležité úkoly. Funguje jako obecné rozhraní, skrze které uživatel a programátor komunikuje se systémem i s různými zařízeními a je na něm závislá i bezpečnost systému. Proto věnují souborovým systémům hodně prostoru. V Linuxu skoro všechno nějak souvisí se soubory a souborovým systémem.

Vysvětlit pořádně co to je souborový systém není jednoduché, proto přijměte prosím jen takové zjednodušující vysvětlení. Předpokládám že každý ví co to je soubor a adresář (ve Windows se tomu začalo říkat složka). Mohu si vytvořit adresář fotky, v něm podadresáře 2002 a 2003 v nich podadresáře 01 až 12 a do nich dávat fotky z příslušného období. To je logická organizace souborů na disku. Je vhodná pro člověka, aby se v "tom" vyznal. Neříká nic o tom, kde tyto soubory jsou fyzicky na disku umístěny, které jeho sektory zabírají a pod. Člověk to vědět ani nepotřebuje, ale počítač ano, aby nám ty fotky mohl ukázat, když budeme chtít. Proto existuje i jakási fyzické organizace souborů (dat) na disku. A to je právě ten souborový systém, fyzická organizace dat na disku, paměťové kartě a podobně. Kromě toho souborový systém o souborech uchovává některé důležité informace. Linuxové (UNIXové) souborové systémy mají navíc speciální druhy souborů, které slouží k dalším užitečným věcem. Souborový systém na nějakém médiu se vytváří procesem známým jako formátování disku.

Existuje mnoho různých souborových systémů. Z Windows jsou známy FAT (kterých je několik variant, v Linuxu se označuje vfat) a NTFS. Nativní linuxový souborový systém je ext2 a ext3, ale Linux jak jsem již uvedl, podporuje mnoho dalších (cizích) souborových systémů. Podporuje všechny varianty FAT i NTFS (ten umí ale jen číst), takže z Linuxu je snadný přístup k datům uloženým pod Windows. Dále podporuje další unixové souborové systémy, jako ReiserFS, XFS (od SGI), JFS (od IBM) i některé ne unixové jako HPFS (od IBM pro OS/2) a pak také speciální, jako NFS (síťový, pro sdílení disků na síti od SUNu), CODA (síťový distribuovaný) a pod.

Sluší se poznamenat, že IBM a SGI své souborové systémy v Linuxu podporují (na jejich podpoře v jádře se podílejí jejich vývojáři), kdežto Microsoft dělá obstrukce (neposkytuje žádné informace a vývojáře NTFS ovladače dokonce zažaloval). Pokud vás zajímá, co znamená to FS, které se objevuje v mnoha zkratkách, tak to je File System, česky souborový systém.

Souborový systém má tedy za úkol ukládat na disk soubory. Možná vás to překvapí, ale souborů existuje vícero druhů. Nejznámější druhy souboru jsou obyčejný datový soubor (který může obsahovat libovolná data) a adresář (ano, adresář je z pohledu souborového systému také jen soubor). Windows (a jeho souborové systémy) už žádné jiné druhy souborů neznají (pokud je mi známo), unixové souborové systémy ale ano. Nejčastěji je možno se setkat se symbolickými linky a pojmenovanými rourami (podrobnosti se tím zabývám v systémové sekci).

Obecná představa souboru je, že to jsou nějaká data uložená na disku, třeba obrázek. Kromě těchto dat se na disk ukládají ještě další data, různé informace o souboru. Např. jeho název, datum a čas poslední změny a podobně. Tyto metadata se nazývají atributy souboru. Různé souborové systémy ukládají různé atributy. Některé z nich jsou společné všem souborovým systémům, jiné jsou ojedinělé.

Souborové systémy unixového typu podporují shodnou standardizovanou množinu atributů. Díky tomu jsou snadno zaměnitelné a je jedno, jestli provozujete Linux na ext3, ReiserFS nebo XFS. S jinými souborovými systémy už to tak snadné není a jejich použití, třeba FAT je omezené.

Mezi důležité "unixové" atributy souboru patří např. id majitele, id skupiny nebo přístupová práva. Na nich je postavena bezpečnost v operačním systému Linux. Souborové systémy FAT a NTFS tyto atributy nepodporují. Ne že by podpora FAT byla v Linuxu zbytečná, díky ní můžete přistupovat k datům uloženým pod windows, ať na disketě nebo na diskovém oddílu Windows. Ale určitě není vhodné na FAT instalovat třeba samotný Linux, nebo na něm mít domácí adresáře uživatelů a podobně.

V případě FAT by problémy dělalo i to, že rozlišuje pouze dva druhy souboru, datový a adresář. V unixech jich existuje víc a nejen existuje, ale i používá. Přesto některé distribuce umožňují Linux nainstalovat na FAT pomocí speciálních ovladačů. Ale berte to jako raritu, na takovou nouzovku, jak se podívat, co to ten Linux je bez nutnosti formátovat fisk na jiný souborový systém, pro vážnější použití to není vhodné.

Rozdíly mezi FS Linuxu a Windows FS

Z předchozí kapitoly vyplývá, že rozdíly mezi souborovým systémem Linuxu a Windows jsou značné. A to jsem o nich hovořil jen obecně. Tato kapitola se zaměřuje právě na jejich rozdíly.

Jeden z nejviditelnějších rozdílů je odlišný způsob připojování souborových systémů. Ve Windows se souborové systémy zpřístupňuje skrze (diskové) jednotky označená A:, B:, C:, D: a tak dále. Přitom na/v dotyčném zařízení souborový systém nemusí být

vůbec k dispozici. Třeba proto, že v disketové mechanice není disketa a nebo že disk není naformátován.

Linux na to jde jinak. Linux nemá žádné jednotky A:, C: atd. Linuxu má tzv. kořenový (root) adresář, který se označuje / a k tomu se připojí disk. Dalo by se říct, že to co je ve Windows C: je v Linuxu /. Soubor C:\pohadky\rumcajs.txt ve Windows bude v Linuxu /pohadky/rumcajs.txt. Zatím nic světoborného. Zajímavé to bude u dalších disků.

Třeba u diskety. Soubor na disketě bude ve Windows dostupný pravděpodobně jako A:/soubor.txt, kdežto v Linuxu např. jako /a/soubor.txt nebo třeba /mnt/floppy/soubor.txt či /media/floppy/soubor.txt. Přitom /a nebo /mnt/floppy nejsou nějaké jednotky, ale obyčejné adresáře na disku připojeném ke kořenovému adresáři.

V Linuxu nejsou žádné (diskové) jednotky, souborové systémy se připojují k obyčejným adresářům. Hlavní, systémový, souborový systém (disk) se připojí ke kořenovému adresáři, který poskytuje jádro jako jediný přípojný bod a všechny ostatní souborové systémy se připojují k libovolným adresářům v tomto souborovém systému (nebo i jiného již připojeného disku, bylo by-li to potřeba). Když k adresáři připojíte disk, tak obsah toho adresáře je obsahem toho disku. Kdyby snad náhodou ten adresář měl už nějaký vlastní obsah, tak se k němu nedostanete, dokud ten souborový systém zase od adresáře neodpojíte. Ale to se nestává, nikdo normální do adresářů určených k připojování souborových systémů žádný obsah nedává. Ale byla by to prima skrývačka :-)).

To má dva zajímavé důsledky. Na první pohled nepoznáte, s kterým diskem vlastně pracujete. Třeba v adresáři /home jsou domácí adresáře uživatelů (v nich mají svoje osobní soubory). Tento adresář je na systémovém disku a ty data tedy mohou být na něm. Také k adresáři ale může být připojen jiný disk a ty data mohou být na tom jiném disku. A dokonce k němu může být připojen nějaký síťový disk a ty data mohou být reálně někde na serveru na síti. Takže ty data mohou být vlastně kdekoliv. Ale to je jedno, uživatel s nimi pracuje pořád stejně, nemusí ho to zajímat. Pokud by ho to ale přeci jen zajímalo, tak kde jsou můžete zjistit nahlédnutím do seznamu připojených souborových systémů. Pokud bude k adresáři /home připojen nějaký souborový systém, dozví se to z něj, včetně toho, co to je a kde se nalézá.

Všimněte si, že ve Windows jednotka reprezentuje zařízení, třeba disketovou nebo CD mechaniku, kdežto v Linuxu adresář ke kterému je připojený souborový systém reprezentuje tento souborový systém. K adresáři se nepřipojuje mechanika, dokonce ani médium v ní, ale souborový systém na tom médiu. Například nenaformátovanou disketu nepřipojíte.

Připojování souborových systémů je docela složitý proces (ve smyslu pochopení co jak a proč, prakticky se to provede třeba kliknutím na ikonku), takže mu věnuji samostatnou kapitolu v systémové sekci. Nastudujte si ji, budete to potřebovat. Souborový systém u výměnných médií se totiž nepřipojuje automaticky, musí se připojit a poté i odpojit ručně. To je pro začátečníka docela frustrující a proto, v rámci souborových systémů, asi nejvetší (viditelný) rozdíl oproti Windows.

Není to ale jediný rozdíl. Dalšího už jste si možná povšimli sami. Linux v cestě používá normální lomítka / kdežto Windows používá lomítka zpětná \. Než začnete na Linux nadávat, tak vezměte v úvahu, že Linux tento způsob převzal od UNIXů a ty tu jsou mnohem déle než Windows i DOS. Proč Microsoft dělá všechno naopak nevím, ale myslím že to je záměr.

Další rozdíl je, že Linuxu v názvech souborů rozlišuje velikost písmen. Soubor petr.txt a Petr.txt jsou v Linuxu dva různé soubory a mohou být spolu v jednom adresáři. Ve

Windows ne, ten to bere jako jeden soubor a nedovolí vám v jednom adresáři udělat tyto dva soubory. Je na to potřeba dávat pozor, když v Linuxu vytváříte soubory na FATce, na kterou potom budete chtít přistupovat z Windows.

Další rozdíl s podobnými důsledky jsou povolené znaky v názvu souborů. Ve Windows je mnoho znaků zakázaných. V Linuxu je zakázán pouze znak /, ale u mnoha dalších je doporučeno je nepoužívat. Jedná se o znaky se speciálním významem na příkazové řádce (viz. sekce CLI) třeba znak * nebo ? (ve Windows jsou zakázané). V Linuxu je doporučeno nepoužívat i mezeru, která je ve Windows používaná běžně. Ve vlastním zájmu ji nepoužívejte. Mezera se používá jako oddělovač parametrů a když budete spouštět třeba textový editor s parmetrem názvu souboru petr mach.txt, tak se je ten editor bude pokoušet otevřít místo toho soubor petr a soubor mach.txt, které neexistují.

S tím souvisí i diakritika v názvech souborů. Linuxu obecně nevadí, ale protože Windows (samozřejmě) používá jiné kódování češtiny než UNIXy, tak je to mezi systémy nepřenositelné. Tento problém je v Linuxu řešitelný, dozvíte se o tom v kapitole o připojování souborových systémů. Ale ve Windows si asi s Linuxovým kódováním názvů souborů neporadíte, takže je lepší se tomu vyhnout.

Textová orientace Linuxu

Teď už máte o Linuxu jakousi představu. Víte, že grafické rozhraní je pro něj něco navíc, jenom jakási nadstavba. Že základní uživatelské rozhraní (ve smyslu komunikace uživatele se systémem) je textové rozhraní a příkazový řádek. Toto ovládání je podpořeno tím, že prakticky cokoliv se v Linuxu prezentuje jako soubor, s kterými se na příkazovém řádku snadno pracuje. Tento obrázek ale není kompletní. Další významná charakteristika Linuxu je, že si zakládá na textových souborech, s kterými uživatel přichází přímo do styku. Tím se liší od Windows, které preferuje spíše binární soubory, s kterými uživatel přímo do styku nepřichází.

Základní rozdíl mezi textovými a binárními soubory je ten, že textové soubory jsou určeny pro lidi, kdežto binární pro počítač. Textové soubory ukládají informace v lidem čitelné podobě, ale programům (a programátorům) dá jejich zpracování (parsování) více práce. V binárním souboru jsou informace uloženy způsobem, který se programům snadno čte (to znamená, že je zpracuje rychleji a programátorovi to dá méně práce), ale člověk vnímá jejich obsah jako tzv. "rozsypaný čaj" a nemůže s nimi přímo pracovat, musí k tomu použít nějaký program, který mu obsah souboru přetransformuje do čitelné podoby. Textové soubory také mohou mít různé formáty a mohou pro ně existovat speciální programy na jejich zpracování, ale jejich existence ani použití není povinná, vystačíte si s univerzálním textovým editorem.

Tento rozpor Linux/Windows

Je jasné, že v některých případech je jednoznačně lepší použít textový formát a ten je pak použit v Linuxu i Windows, v jiných případech je zase jednoznačně lepší použít binární formát a ten je použit v obou systémech, ale v některých případech to tak jasné není. Nelze jednoznačně říci co je lepší a co horší, obojí má své výhody i nevýhody. Záleží na preferencích, na tom co kdo považuje za více či méně důležité. Linux a Windows stojí na jiných preferencích, jsou to systémy s různým pohledem na svět a to se mimo jiné odráží i v tom, že jeden preferuje spíš binární a druhý spíš textové soubory v nejednoznačných situacích.

Tento rozpor ale není jen v rámci Linux/Windows na úrovni operačních systémů, ale i třeba v rámci datových formátů. Nevýhodou textových datových formátů je, že jejich zpracování je pomalejší a že tyto soubory jsou obvykle objemnější. Ale s rostoucím

výkonem počítačů, přenosových rychlostí i úložných kapacit je to čím dál tím zanedbatelnější. Z lidského hlediska vnímání času je dnes zpracování textového i binárního datového souboru běžné velikosti stejně rychlé. Kapacita již také není problém a při použití komprimace jsou na tom textové soubory (za cenu další trošky výkonu) dokonce lépe. Velký průlom v tomto směru způsobil internet a textový formát html stránek. Ten ukázal nejen že to jde, ale že to má i své nemalé výhody, především snadné kooperaci a výměny dat mezi různými systémy, snazší rozšiřování a robustnosti formátu, neexistence binárních nekompatibilit a podobně. Na základě toho vzniknul univerzálnější XML formát pro výměnu dat, který se stal velmi populárním a který díky promyšlenosti a množství různých podpůrných technologií odstranil obtížnost zpracování textových formátů. S XML formáty se programátorům pracuje stejně snadno a možná i snadněji než s binárními formáty. XML se stalo okamžitě velmi úspěšné a každý už o něm už určitě slyšel, i když možná neví přesně co to je. XML je úspěch UNIXových principů, tedy principů, na kterých stojí i Linux.

Světovým trendem je ústup od binárních datových souborů a nástup textových. Pomalu, ale jistě se mu podřizuje i Microsoft a to nám dává neději, že svět Linuxu by se mohl se světem Windows stát o něco kompatibilnější. U textových souborů se bude Microsoftu obtížněji udržovat nekompatibilita. Mimochodem OpenOffice.org, Abiword či Gnumeric (kancelářské aplikace používané nejen na Linuxu) používají XML nativně a už dlouho. Zde má konkurence před Microsoftem také značný náskok. Nedovede to však marketingově využít.

Není to důležité, ale pro zajímavost, textové jsou i mnohé síťové protokoly, z těch známějších http, ftp, pop3 či smtp, jejich UNIXový původ se nedá zapřít. Když průběh komunikace budete zapisovat do souboru, vznikne vám obyčejný textový soubor, který lze snadno číst a z kterého i snadno poznáte co se děje (jestli umíte anglicky). Podívejte se třeba na FTP klienta gftp, v nejspodnějším okně uvidíte záznam komunikace s FTP serverem. O html a xml jsem už mluvil. Kdyby internet vymýšlel Microsoft, bylo by to všechno binární a byly by tu věčné problémy s kompatibilitou.

A jak se tedy preference textových souborů projevuje v Linuxu? Jaké z toho plynou rozdíly oproti Windows?

Nejvýznamnější je asi existence konfiguračních souborů. Ve Windows se konfigurace systému, aplikací i uživatelského nastavení ukládá obvykle do binárního souborů zvaný registr (ve skutečnosti jich je několik, ale aplikacím i uživateli se jeví jako jeden). Pro práci s registrem potřebujete speciální programy. Převážně mají podobu konfiguračních utilit nebo konfigurační dialog nějaké aplikace. Kromě toho existuje pro nouzové účely univerzální regedit.

V Linuxu se konfigurace ukládá do konfiguračních souborů. Každá aplikace má jeden nebo i víc svých vlastních. Obecně se editují textovým editorem. Dříve to bylo vcelku běžné. Dnes Linux disponuje i grafickými konfiguračními nástroji. Přímá editace konfiguračních nástrojů je ale i nadále možná a není výjimkou, že grafický konfigurační nástroj v Linuxu umožňuje jen základní nastavení, kdežto přímá editace je flexibilnější a umožňuje nastavit věci lépe. Analogická situace tohoto ve Windows je, když vám nezbývá nic jiného, než vzít regedit a nastavit nějaké registry přímo.

Konfigurační soubory mají tu nevýhodu, že jim musíte rozumět, musíte vědět co kam napsat. Naštěstí bývají okomentovány a existuje k nim dokumentace, takže to není zas tak strašný, když víte co chcete (což u začátečníků není vždy pravda). Z tohoto pohledu se grafické nástroje používají pohodlněji, ať zapisují do registrů nebo konfiguračních souborů.

Na druhou stranu konfigurační soubory lze snadno kopírovat. Před jejich úpravou si můžete snadno udělat zálohu a když se něco nepodaří nebo i z jiných důvodů, se

snadno a spolehlivě vrátit k předchozímu stavu. Hned, po týdnu i po roce. Záloh můžete mít i víc. Také můžete konfigurační soubor poslat kamarádovi nebo naopak on vám. Když se vám něco nedaří nastavit, můžete se na něj zeptat někoho na internetu a podobně. Obecně je dobré mít zálohovány všechny konfigurační soubory a nebo alespoň ty ručně upravované. Obnova systém či instalace nového na další počítač se tím významně urychlí. Nastavení pomocí konfiguračního souboru vám dá víc práce, ale jen jednou. Některé své konfigurační soubory uchovávám a vylepšuji už dlouhá léta. Nakonfigurovat si připojení k internetu přes dialup? Neumím, dělal jsem to jednou někdy před třemi lety, ale od té doby je mám schované, když někde někomu něco instaluji nebo konfiguruji, stačí je zkopírovat, upravit číslo, přihlašovací jméno a heslo a je to, otázka třiceti vteřin, jak dlouho to budete dělat ve windows? Jak dlouho vám bude trvat, než mi nakonfigurujete připojení k internetu přes můj mobil co si k vám přinesu? Na Linuxu, když to už jednou máte, to je jen otázka zkopírování pár souborů. Na Linuxu se konfiguruje jen jednou. Z tohoto pohledu jsou pohodlnější zase konfigurační soubory.

Konfiguračním souborům a konfiguraci se věnuje podrobněji samostatná kapitola. ?? ale ještě není napsaná :-)

Ale nejen konfiguračními soubory je živ člověk. Občas se stane, že nějaký program který jede na pozadí narazí na nějakou chybu. Ve Windows vám vyhodí hlášku, že se stalo to a to, ať se ti to líbí nebo ne, stiskni OK. Na Linuxu to mají programy jedoucí na pozadí složitější. Jak už víte, v Linuxu nemusí být grafické rozhraní vůbec přítomné. Navíc se Linux původně profiloval jako serverový operační systém, a to znamená, že u počítače nemusí být ani žádný člověk. Proto se na linuxu řeší podobné problémy způsobem serverovým, zápisem do logu. Log je textový soubor, do kterého aplikace zapisuje informace. Aplikace může mít vlastní log, ale kromě toho existuje také log systémový.

Systémový log nefunguje přímým zápisem aplikace do souboru, logování je v Linuxu propracováno. Aplikace pro zápis do logu používají systémové rozhraní kterému předávají informace a o samotný zápis dat do souboru se nestarají. O to se stará systém, který je konfigurovatelný. Můžete si určit, že se vše zapisuje do jednoho souboru, nebo mít souborů víc, dokonce je možno nechat logy posílat na jiný počítač, což se doporučuje tam, kde záleží na vysoké bezpečnosti (do logu se zapisují i bezpečnostní incidenty, třeba pokusy hacknout váš stroj, když se to hackerovi podaří, vymaže tyto informace z logu, zanechá vám na počítači nějaký backdoor, potichu se vypaří a vy nic nevíte, když jsou logy na jiném počítači, musel by hacker hacknout i tento počítač a čím víc překážek mu postavíte do cesty, tím lépe jste proti němu chráněni).

Když budete pořád zapisovat něco do logu, budou tyto čím dál větší a nakonec vám zcela zaplní disk. Tomu se můžete bránit jejich pravidelným odmazáváním, samozřejmě automatickým, kdo by to dělal ručně. Tato činnost se nazývá rotování logů. Také se to dá/musí nakonfigurovat. Když si nainstalujete nějakou desktopovou distribuci Linuxu, tak tam tohle všechno máte, pochopitelně už předkonfigurované. Nemusíte se o to starat, ale měli byste vědět, že tam něco takového máte.

Když máte nějaké záhadné problémy s Linuxem, třeba vám najednou havaruje grafické rozhraní při startu počítače, obvykle se příčinu, nebo aspoň něco, co vás nakopne tím správným směrem, dozvíte v lozích. Linuxový začátečník tápe, protože to neví, udělá mu to blik, tma, příkazový řádek a nic, je bezradný, neví co má dělat. Přitom kdyby nahlédnul do /var/log/XFree86.log tak by se třeba dozvěděl, že X Server nemůže nalézt žádné fonty a že se tedy nespustí. Kdyby si to přečetl, plácl by se do čela, no jo, vždyť já jsem ten adresář s fonty přejmenoval, opravil by název a zase by mu všechno jelo. Jenže protože to neví, přistoupí k tomu jako k Windows a celý Linux

zbytečně reinstaluje (v lepším případě) a nebo smaže (v tom horším). Pozn. není to moc dobrý příklad, protože on by to ten X Server v tomto případě napsal i do příkazového řádku v kterém by uživatel skončil, takže by věděl co se děje, ale nic lepšího mě zrovna nenapadlo.

Může se stát, že třeba tomu co se píše v logu nerozumíte, že nevíte co to znamená, ale to nevadí, bude to vědět někdo jiný. Když se zeptáte na nějakém fóru na internetu, nejede mi grafické rozhraní a v logu jsem našel toto, tak vám někdo určitě poradí. Je to lepší, než když se zeptáte, nejede mi grafické rozhraní, co s tím budete dělat? :-) Ale pokud tam bude slušné osazenstvo, tak vám někdo řekne, vystav někde výpis z logu, my se na to podíváme. Takového přátelského přijetí se dočkáte třeba na www.nyx.cz v klubu pro linuxové začátečníky.

Obecně se dá říct, že v Linuxu se problémy řeší lépe, protože je k nim více informací a systém je průhlednější. Ale chce to o Linuxu a o tom jak funguje něco vědět. Textové logy k tomu významně přispívají.

O virtuálním souborovém systému /proc už jsem se už několikrát zmiňoval, takže víte, že to je rozhraní jádra, skrze které komunikuje s uživatelem, skrze které se můžete podívat dovnitř jádra. Sluší se napsat o něm i zde, protože tak činí pomocí virtuálních textových souborů. Třeba v /proc/meminfo se dozvíte něco o paměťovém subsystému.

Z pohledu příkazového řádku je Linux na textové soubory také výborně vybaven. Má hodně příkazů a nástrojů pro práci s textovými soubory. Od jednoduchého wc (word count, s onou místností to nesouvisí), které počítá počet znaků, slov a řádků až po výkonné jako je awk, který může fungovat i jako jednoduchý programovací jazyk a je specializovaný na práci se sloupcově orientovanými textovými soubory. Potřebujete sečíst všechna čísla ve třetím sloupci u kterých text v prvním sloupci je 08.03.2004? Žádný problém. Jeden můj kolega potřeboval něco podobného dělat u výpisu z telefonní ústředny. Měl dva soubory. V jednom bylo mimo jiné sloupec tel. číslo a sloupec cena hovoru a v druhém mimo jiné sloupec tel. číslo a sloupec jméno tel. účastníka. Potřeboval sečíst všechny ceny hovoru pro všechna tel. čísla a tyto čísla nahradit jménem tel. účastníka. Ne že by to nešlo naprogramovat v normálním programovacím jazyku, ale v awku je to skutečně srandička.

A takových nástrojů je v Linuxu skutečně mnoho a kromě toho v něm jsou všudypřítomné interpretované jazyky jako Perl nebo Python, které jsou také velmi dobře vybaveny pro práci s textovými soubory a texty. A samozřejmě v něm nechybí ani kvalitní a výkonné textové editory, ty nejlepší jsou Vim a EMACS, ale to není nic pro začátečníky. Docela dobrý jednodušší editor, který je součástí GNOME je gedit, v KDE je kedit. Ty zastávají úlohu notepadu. Něco na střední úrovni by mohl být bluefish nebo quanta. V textovém režimu se začátečník najde asi v editoru, který je součástí Midnght Commandera. Dá se spustit i samostatně příkazem mcedit. Zdá se vám toho hodně? Není to určitě všechno, ale ne vše musí být součástí každé distribuce i když většina asi ano. Linux je textově orientovaný a textový editor v něm hraje důležitou roli.

Začátečníci se snaží Linux používat jako Windows, snaží se v něm dělat a chovat tak, jak jsou zvyklí z Windows. To je asi nevyhnutelné, bohužel. Linux ale není Windows, a pokusy používat ho jako Windows nevyhnutelně vedou ke konstatování, že není dobrý jako Windows. To je pravda, Windows zase nejsou tak dobré jako Linux, to zjistíte v zápětí, kdy je zkusíte používat jako Linux. Člověk se diví, že vůbec někdo na Linux z Windows přechází, začátečník se nutně střetává se slabinami Linuxu a jeho silné stránky mu zůstávají skryty. Jedna z jeho silných stánek je právě jeho textová orientace a všeobecně dostupné silné nástroje pro práci s texty. Začátečník (míněný ten, co přestupuje z Windows, kde klidně může být velmi zkušený uživatel), to však

neumí ani ocenit, ani využít.

Kromě příkazů, které s texty pracují, jsou v Linuxu také příkazy, které texty produkují. Jsou to různé informační příkazy, které vypisují různé informace. Defaultně je vypisují na obrazovku. Ale lze je snadno přesměrovat do příkazu, který je rovnu zpracovává. Jen pro příklad, dejme tomu, že potřebuji do nějakého skriptu zjistit, kolikrát jsem do svého Linuxu přihlášen. Seznam přihlášených uživatelů produkuje příkaz who. Z něj potřebuji vyfiltrovat všechny ostatní, to zařídí příkaz grep a počet řádků vyfiltrovaného výpisu odpovídá tomu, kolikrát jsem přihlášen. Na jejich sečtení mám příkaz wc. Takže tato kolona příkazů zadaná najednou:

```
who | grep wraith | wc -l
```

Mi poskytne kýžený výsledek. Jednoduché, elegantní, krásné, leč pro standardního uživatele windows obskurní a archaické, přece nebude používat takové primitivní způsoby nebo si dokonce psát skripty. On chce jen klikat, klik, klik, klik, vidím to na svých kamarádech :-). Je to na první pohled těžkopádné a neefektivní. To však není pravda, tyto příkazy můžete psát do skriptů a psaním skriptů, si můžete zautomatizovat, zjednodušit a zefektivnit řadu rutiních činností. Třeba kopírování fotek z USB fotoaparátu do nějakého adresáře s názvem podle aktuálního data si můžete zkrátit na jedno kliknutí.

Mimochodem, skripty jsou další forma textových souborů, s kterými se v Linuxu můžet vcelku běžně setkat. Skripty jsou jakési záznamy příkazů na příkazovém řádku doplněné o práci s proměnnými a takovými věcmi jako jsou podmínky a cykly, tedy takové hodně jednoduché programování. Z pohledu souborového systému se jeví jako ostatní programy, teprve nahlédnutím do nich poznáte, že se jedná o skript. Takže začátečníci je možná i používají, aniž o tom ví. Dobré je na nich to, že si je můžete snadno upravit, když potřebujete.

Grafické rozhraní a příkazový řádek

V této kapitole volně navážu na tu předchozí, budu se v ní věnovat nakousnutému GUI versus CLI, tedy grafické (klikací) rozhraní versus příkazový řádek. Zdůrazňuji klikací, protože příkazový řádek lze používat i v GUI. Nejde ani tak o to, jestli jste v grafickém nebo textovém režimu, ale o způsob ovládání počítače. Je to spíš klávesnice versus myš. Ale ani to není přesné, protože i grafické rozhraní lze ovládat pomocí klávesnice. Jak vidno, je to zapeklitý problém, a proto mu věnuji celou kapitolu. A mimochodem, každý ví, že lepší je klávesnice, myš má jen tři tlačítka :-).

Když spustíte Windows, naběhne vám grafické rozhraní, skrz které můžete počítač ovládat. Můžete si třeba spustit DOSové okno s příkazovým řádkem. V Linuxu je tomu přesně naopak, po jeho spuštění se dostanete do konzole s příkazovým řádkem, z kterého si můžete mimo jiné spustit grafické rozhraní. Nutno podotknout, že Linux lze nakonfigurovat tak, aby se toto grafické rozhraní spouštělo automaticky a uživatel pak konzoli ani nezahlédne (přesto na něj kdesi v hlubinách OS stále trpělivě čeká). Z grafického prostředí se pak můžete snadno přepnout do konzole (a zase zpět, samozřejmě) nebo si tam můžete spustit terminál s příkazovým řádkem v okně. Ale ve Windows je systémově výchozí GUI a příkazový řádek si můžete spustit jako něco navíc a v Linuxu je výchozí příkazový řádek a GUI si můžete spustit jako něco navíc.

Proč je to naopak? Ve Windows je grafické rozhraní součástí jádra. Příkazový řádek je pak jakási emulace DOSu pro zpětnou kompatibilitu, jen jeden z programů. Jádro Linuxu nemá s grafikou ani grafickou kartou nic společného. V jádře Linuxu není nic, co tam být nemusí. Grafický subsystém s podporou různých grafických procesorů, různých akcelerací, kešování a podobně je rozsáhlá a složitá záležitost. Snadno se tam udělá

nějaká chyba. Kvalitní jádro se umí chránit proti chybám v aplikacích, ale neumí se chránit proti chybám v sobě samém (je to principiálně nemožné). Proto čím méně toho v jádře je, tím lépe. Zvyšuje se tím stabilita systému. Grafický subsystém proto v Linuxu zajišťuje X Window System, hardware v něm má na starost "obyčejná" aplikace, X Server. Podobně tomu je i v jiných UNIXech.

Pravda, když je chyba v X Serveru a ten spadne, tak s ním jdou k čertu i všechny aplikace, které na X Server závisí (to nejsou všechny, ale určite to jsou ty, co mají grafické rozhraní, tedy ty, které uživatel vidí). To také moc nepotěší, ale je to lepší než pád celého operačního systému. Dnes už asi každý ví, že před vypnutím počítače (což je i pád OS) by s měl korektně ukončit činnos operačního systému. Násilné přerušení chodu operačním systémům nesvědčí a v Linuxu pád grafického rozhraní nenaruší chod operačního systému.

Výhodné to je pro různé síťové servery, které (v Linuxu) nejsou závislé na grafickém rozhraní. Grafické rozhraní dokonce nemusí být vůbec spuštěné, což vám ušetří trochu výkonu a docela hodně paměti. A hlavně, zvýší to stabilitu systému. Chyby v ovladačích grafických karet se uvádí jako nejčastější příčina pádů Windows (ve Windows je ovladač grafiky součástí jádra, jádro se nedovede chránit proti chybám v sobě, ovladač grafické karty je složitá záležitost a ovladače dělá kde kdo).

Zjednodušenš řečeno, při startu počítače jádro Linuxu grafickou kartu inicializuje v standardním textovém režimu (to umí všechny grafické karty a dělají to stejným způsobem, takže na to nejsou potřeba žádné ovladače), dále zpřístupní klávesnici a spustí příkazový řádek. To je vše, operační systém je připraven vám sloužit. V Linuxu je příkazový řádek plnohodnotné uživatelské rozhraní, pomocí kterého lze počítač dobře ovládat a také pro něj existuje hodně aplikací, které umí pracovat (i) v textovém režimu. Třeba Midnight Commander (správce souborů), Vim (textový editor), links (webový prohlížeč) a další. Z příkazového řádku lze Linux i plnohodnotě konfigurovat. Včetně takových věcí, jako je hlasitost zvukové karty a pochopitelně je zde možno přehrávat třeba mp3. Práci v tomto textovém režimu označujemé jako práci v/na konzoli.

Grafický režim v Linuxu představuje toliko alternativní ovládání. V pradávných dobách bylo omezené a většina věcí se musela dělat přes příkazový řádek, nebo v aplikacích používající textové rozhraní. Dnes tomu už dávno není pravda, já osobně považuji Linuxové GUI za lepší než to we Windows (které má ale zase třeba širší nabídku aplikací, takže člověk si nevybere). Přesto se můžete dostat do situace, kdy se bez příkazového řádku neobejdete. Nemůžu si zrovna na žádnou tyickou situaci vzpomenout, kromě té samozřejmé, že vám přestalo fungovat grafické rozhraní a musíte ho opravit. Jinak ale já příkazový řádek používám často, na spoustu věcí mi přijde pohodlnější (a na spoustu zase ne). Ale vím, že i ve windows jsou situace, které lze řešit jen pomocí příkazu na příkazovém řádku, jedná se ale o speciální záležitosti spojené s administrací systému.

Někteří lidé řeší otázku, zda je lepší GUI (grafické rozhraní) nebo CLI (příkazový řádek). To je nesmysl, některé věci jdou dělat lépe v GUI a jiné v CLI. Například grafický editor je jasný případ GUI aplikace, upravovat obrázky přes CLI je v omezené míře sice také možné, ale není to jaksi ono. Na druhou stranu, CLI je výborné na automatizování často se opakující činnosti. O skriptech jsem už psal, to je ono. Vlastně při tom ani příkazovou řádku přímo nepoužíváte, jen ve skriptu používáte příkazy pro ni. Vzhledem k tomu, že na příkazové řádce lze dělat tařka všechno, lze si tařka všechno naskriptovat. Myslím že každý uzná, že tohle, přinejmenším pro někoho, to přínos má. Ve skutečnosti pro každého, protože i když uživatel Linuxu nepíše vlastní skripty, tak ten systém používá mnoho systémových, které napsali třeba autoři distribuce a které uživateli zjednodušují používání Linuxu.

Na druhou stranu, příkazů pro příkazovou řádku existuje skutečně velké množství a jejich zvládnutí (byť i jen nějaké té základní sady) není jen tak. GUI je oproti tomu mnohem intuitivnější a snáze se učí. Jak jsem již psal, kdysi bylo možno Linux ovládat jen skrze CLI, jak se Linux popularizoval bylo GUI stále významnější a tak pro ovládání začali vznikat GUI alternativy. Mnohdy taková GUI alternativa je jen grafická nadstavba nad příkazovým řádkem. Stiskli jste tlačítko a grafická utilitka někde na pozadí spustila nějaký příkaz. Chování CLI příkazů lze modifikovat tzv. volbami a parametry, někdy obrovskou spoustou voleb a parametrů. Grafická nadstavba může používat jen vybranou sadu parametrů, tedy ne vše, co umí samotný příkaz.

Domnívám se, že tohle je ideální systém. Začátečník dostane snadno použitelnou grafickou utilitku s kterou zvládně základní nebo standardní věci. Zkušený uživatel nezůstane ochuzen možnost psaní skriptů a automatizace prací, nemusí ani znát všechny možné volby a parametry a vědět k čemu slouží. A odborník má po ruce komplexní nástroj který dovede splnit i velmi specializované přání. Přitom se o to všechno stará jeden a ten samý program.

Vsuvka: Přesto to uživatel Linuxu nemá tak snadné jako ve Windows. Sice nemusí používat příkazovou řádku, jako dřív, nicméně stále se občas (hlavně v případě obtíží) nevyhne ruční editaci konfiguračních souborů. Zde Linux ještě tak daleko nepokročil, ale i v tomhle dělá pokroky a jednotlivé desktopové distribuce se předhánějí v tom, která nabídne lepší konfigurační nástroje. Plnou moc a kontrolu nad systémem má jen uživatel, který ovládá příkazovou řádku, ruční editaci konfiguračních souborů a ví jak celý systém pracuje a funguje.

Bezpečnost

?? doplnit info o rootovi, co je to zac

Když už píši o souborech, nemohu se nezmínit o uživatelských (nebo také přístupových) právech k souborům a bezpečnosti v Linuxu jako takové. Především pro uživatele Windows 9x a ME je existence něčeho takového překvapující.

V Linuxu má každý uživatel svůj číselný identifikátor, tzv. UID (User ID tj. ID uživatele) ke kterému přiřazeno uživatelské jméno. Uživatelské jméno je zavedeno jen pro "lepší pocit", lépe se to pamatuje než číslo. Každý uživatel má jedinečné UID, kdyby se stalo, že jedno UID budou mít dva uživatelé, pak oba budou mít shodná přístupová práva a budou si moci "lézt do zelí".

Kromě uživatelů podporuje Linux také uživatelské skupiny. Skupina opět není nic jiného než nějaké číslo a k němu přiřazené jméno. Toto číslo se označuje jako GID (Group ID tj. ID skupiny). GID je opět jedinečný. Každý uživatel může být zařazen do libovolného počtu skupin.

Každý soubor v Linuxu patří nějakému uživateli a skupině. Na linuxovém (UNIXovém) souborovém systému má každý soubor kromě jména, času a pod. i atribut uid a gid. Takže když je uid souboru nastaven na 500, patří uživateli s UID 500. Když je uid souboru nastaven na 0, patří uživateli root, protože root má UID = 0. Na nelinuxovém souborovém systému, který atributy uid a gid nepodporuje, jako je třeba FAT ve Windows pak všechny soubory dotyčného disku patří jednomu uživateli. Defaultně tomu, kdo disk připojuje, ale při připojování disku lze určit i jiného uživatele. To platí i o souborovém systému iso9660 (používají ho datová CD), pokud nemá RockRidge rozšíření, což (není-li vypalováno přímo pro Linux, jako linuxová např instalační CD) obvykle nemá.

?? doplnit info o změně uid a gid

Každý soubor v Linuxu má určeny uživatelská práva. Stejně jako u uid a gid, linuxové souborové systémy mají pro každý soubor speciální atribut, na nelinuxových se pak nastavují hromadně pro všechny soubory na disku.

Uživatelská práva

- Uživatelská práva jsou trojí, právo na **čtení**, právo na **zápis** a právo na **spuštění** souboru.
- Tyto práva se určují zvlášť pro **majitele** souboru, **skupinu** souboru a **ostatní**.

Nelze tedy nastavit každému souboru práva zvlášť pro každého uživatele. Normálně se uplatňují u majitele souboru práva pro majitele a u ostatních uživatelů práva pro ostatní. Nevyhovují-li pro nějakou skupinu uživatelů práva pro ostatní, tj. chceme vybraným uživatelům povolit např. právo zápisu do souboru a ostatním ne, pak můžeme vytvořit skupinu (dejme tomu autori) a zařadit příslušné uživatele a soubory do této skupiny. Jednoduché jako facka. Kdybyste ale měli dvě různé speciální skupiny (ostatní nesmí nic, první skupina jen číst a druhá i zapisovat) máte v podstatě smůlu. Nouzově to lze řešit hardlinkem (viz. dál), ale není to hezké řešení. Naštěstí něco takového není obvykle potřeba. Když ano, je možno některé souborové systémy v Linuxu rozšířit o ACL (Access Control List), které umožňuje jemnější nastavení, ale není to triviální, je kvůli tomu potřeba opatchovat jádro.

Ta samá přístupová práva se dají nastavit i adresáři, ale znamenají zde něco trochu jiného. Stačí si uvědomit, že i adresář je ve skutečnosti soubor, datový soubor, seznam adresářů a souborů které obsahuje. Právo čtení povoluje přečíst tento seznam, tedy přečíst obsah adresáře (třeba příkazem ls). Právo zápisu umožňuje měnit obsah tohoto seznamu, tedy mazat v něm soubory a podadresáře nebo zakládat nové. Právo spuštění povoluje aby se adresář stal pracovním, tedy aby bylo možno do něj vstoupit. Nevím sice k čemu je dobré povolit čtení obsahu adresáře a přitom do něj zakázat vstoupit nebo naopak, ale jde to.

Pozor na častý omyl začátečníků. To že zakážete zápis do souboru petr.txt ještě neznamená, že vám ho nemůže někdo smazat. Smazání souboru zakážete zákazem zápisu do adresáře, ve kterém se soubor nachází. Z tohoto důvodu nelze nastavit, že v jednom a tom samém adresáři některé soubory smazat půjdou a některé ne. Zákaz zápisu do adresáře se vztahuje vždy na celý adresář. A platí to i naopak, to ž zakážete zápis do adresáře ještě neznamená, že vám nemůže někdo přepsat obsah souboru který je v něm. Tomu zabráníte jen zákazem zápisu do souboru.

Ve Windows se to, zda je program spustitelný, rozlišuje podle přípony. V Linuxu lze spustit každý soubor, kterému dáte právo pro spuštění. Není-li ve skutečnosti spustitelný, jeho spuštění selže stejně, jako byste ve Windows dali textovému souboru příponu exe.

Když v Linuxu spustíte normální program, běží s právy uživatele který ho spustil. To jest může číst ty soubory, které může číst dotyčný uživatel a podobně. Někdy můžeme potřebovat, aby program běžel s právy jiného uživatele, než který jej spustil. Proto lze programu nastavit tzv. suid bit (říká se suidnout program). Takový program se pak spouští s právy majitele souboru, ne uživatele který jej spouští. Obdobně lze nastavit sgid bit a spouštět program s právy skupiny. Velmi se nedoporučuje nastavovat suid bit programům, které patří rootovi. Ty pak běží s právy roota, což je velké bezpečnostní riziko. Pokud program není napsaný s ohledem na toto spouštění, tak je v celku pravděpodobné, že v něm bude nějaká chyba, která útočníkovi umožní program zmást tak, že provede něco co by dělat neměl. Pokud je potřeba aby běžný uživatel spouštěl nějaký program s právy roota, pak existují bezpečnější řešení, třeba spouštění přes sudo, kterému lze nastavit jemně kdo co může spustit a jak. Ani sudo

ale nezabrání zneužití chyby v programu.

Z bezpečnostních důvodů lze suid a sgid bit nastavovat jen binárním programům. Ne skriptům a programům v interpretovaných jazycích jako je ruby, python a nebo perl. Tedy přesněji, i jim je lze nastavit, ale přesto se spustí s právy spouštějícího uživatele, ne majitele souboru.

Dokumentace

Dokumentace a nápověda k Linuxu má několik podob. Hned na začátku je nutno poctivě říct, že ačkoliv programy jsou obecně lokalizované celkem dobře, tak to obvykle neplatí pro dokumentaci a nápovědu. Ta často zůstává v anglické verzi. Zřejmě to souvisí s obecnou nechutí číst takové věci a tedy nízkou motivací k překladu a také náročností udržovat překlady v aktuální podobě při rychlém tempu vývoje linuxových programů.

Základní uživatelské rozhraní Linuxu je textové, a tak by nikoho nemělo udivit, že základní nápověda je dostupná v první řadě skrz toto rozhraní. Touto nápovědou jsou manuálové stránky. V těchto stránkách jsou především popsány příkazy příkazového řádku, ale také některé programy, konfigurační soubory, funkce některých programovacích jazyků a podobně. Některé manuálové stránky jsou přeloženy do češtiny.

Manuálové stránky nemají výukový charakter jako učebnice, návody a podobně. Jsou obvykle charakteru referenčního. Proto se v nich nevysvětluje proč a nač a jak, nejsou uváděny příklady použití a podobně. Jsou to jen stručné reference. Manuálové stránky jsou něco jako třeba česko-anglický slovník, který obsahuje slovíčka a popisuje je, ale nefunguje jako učebnice, podle které by se člověk učil angličtinu.

Proto když začátečník z počátku tápe v Linuxu a potřebuje se ho naučit používat, tak je chybou ho odkazovat na manuálové stránky, ať si to nastuduje a neotravuje. Je potřeba mu věc nejdříve vysvětlit, ať přímo nebo odkazem na vhodný výukový text a teprve pak mu poradit, že podrobnosti si už sám může dohledat v té a té manuálové stránce.

K prohlížení mauálových stránek slouží příkaz man, existují jeho GUI alternativy, třeba xman nebo yelp (to je obecnější prohlížeč dokumentace, prohlížení manuálových stránek je jeho součástí). O příkazu man si vyvoláte nápovědu příkazem man man. Příkaz man zajišťuje přeložení manuálové stránky z jejího zdrojového kódu do čitelné podoby a její zobrazení v nějakém pageru (stránkovač, protože většina stránek se nevejde na obrazovku a obsahem je potřeba listovat).

Kdysi dávno se jako pager používal příkaz more (česky to znamená více, určitě známý i některým uživatelům Windows), posledních pár let se snad všude používá jako pager podstatně lepší příkaz less (česky to znamená méně a zde skutečně platí, že méně je někdy více). Pager je jakýsi prohlížeč textu. O tom jak ho ovládat se dočtete, když zadáte příkaz man less. Úplně základní je q = quit (konec), / = vyhledávání v textu, n = next (další hledaný výraz). Mějte na paměti, že toto ovládání platí pro pager less.

Manuálových stránek je opravdu hodně. Proto jsou rozděleny do několika sekcí. Z těch důležitějších, v sekci 1 jsou popsány uživatelské příkazy a programy, v sekci 8 jsou popsány administrátorské příkazy a programy, v sekci 5 jsou popsány konfigurační soubory. Klíčová slova v různých sekcích mohou být shodná. Třeba slovo sleep je jak uživatelský příkaz, tak funkce v jazyce C. Když dáte příkaz man sleep, zobrazí s první nalezená manuálová stránka, v tomto případě tedy ta ze sekce 1. Když budete chtít manuálovou stránku z jiné sekce, třeba druhé, musíte dát příkaz man 2 sleep. Když

nevíte v které sekci co je, můžete dát příkaz man -a sleep a budou vám postupně zobrazeny všechny existující manuálové stránky sleep.

Manuálová stránka nemá charakter učebnice, ale reference, předpokládá se tedy, že víte co chcete a který příkaz vás zajímá. To ovšem u začátečníků není moc časté. Oni často ani neví, jaký příkaz použít. V takovém případě můžete použít jednoduché vyhledávání. Každá manuálová stránka má krátký popisek o čem je a je možno si nechat zobrazit seznam manuálových stránek, jejichž popisek obsahuje klíčové slovo. Například seznam manuálových stránek, které mají něco společného s diskem, vyvoláme příkazem man -k disk. To nám dá pravděpodobně velmi dlouhý seznam, který se ani nevejde na obrazovku.

Takový dlouhý seznam můžeme buď zkrátit filtrováním na další klíčové slovo pomocí příkazu grep. Např. man -k disk ' grep -i partition nám poskytne seznam manuálových stránek, které mají pravděpodobně něco společného s partition tabulkou disku. Druhou možností je poslat seznam do, nám už známé, prohlížečky textu less příkazem man -k disk | less.

Tohle jsou jakési základní možnosti. Man toho umí víc, ale to už si když tak přečtete v man man.

Další nápovědní systém je sytém info. Ten si v podstatě konkuruje s manuálovými stránkami. Manuálové stránky jsou starý UNIXový standard, podle mě dodnes dobře funkční. Info přináší několik vylepšení. Je také v textovém režimu, ale všechny informace jsou hypertextově provázány v jeden celek. Obsahuje seznam témat, o několika úrovních, který lze procházet tam a zpátky a podobně. I info je uděláno pro textové rozhraní, existují i grafické nadstavby??konqueror?. Textové rozhraní bude uživatelům zvyklých na gui připadat (oprávněně) neohrabané. O infu se soudilo, že nahradí manuálové stránky a mnoho jich také bylo do infa převedeno. Proto místo man sleep můžete dát i info sleep. O infu samotném se dozvíte více příkazem `info info`.

Kromě toho, že jsou do infa převedeny manuálové stránky (ne všechny, stránky píší autoři příkazů a je jen na nich, jestli je napíší pro man, info nebo obojí) tak v něm jsou i rozsáhlejší dokumentace k programům, třeba k gcc. Program info umí číst i manuálové stránky, takže bude asi lépe, když si zvyknete používat info.

Kromě toho bývá občas přiložena normální dokumentace k programům v různých formátech. Čistém textovém, HTML, PS, PDF a podobně. Může mít libovolný charakter, i výukový, takže by ji začátečník rozhodně neměl přehlédnout. Standardně bývá umístěna do adresáře ../share/doc. Konkrétní cesta se liší podle toho, kam je aplikace instalována, nejběžnější cesty jsou /usr, /usr/local, /opt, takž z toho pak vychází /usr/share/doc a podobně. Nahlédněte tam a uvidíte kolik tam toho je, mnohdy velmi obsáhlého. Třeba databáze tam může mít kompletní příručky pro databázového uživatele, programátora i administrátora.

Tato dokumentace prakticky nikdy není přeložena.

Další nápověda bývá přímou součástí (především) GUI aplikací, kterou si vyvoláte přes menu, tak jako to je obvyklé i ve Windows. Tato také nebývá moc často přeložena, ale třeba u takového OpenOffice.org, což je asi pro mnoho uživatelů to nejdůležitější, přeložena je, přestože je velmi rozsáhlá.

Dále existuje pro Linux rozsáhlá sada návodů HOWTO (jak udělat), které se snaží komplexně popsat nějaké vybrané téma, třeba vypalování CD a podobně. Jsou v HTML a primárně jsou dostupné na internetu. Některé distribuce je obsahují i na instalačních CD. Mohou pak být nainstalované do /usr/doc/HOWTO. Tyto návody také ve většině případů bývají v angličtině. Ale třeba CZECH-HOWTO ??ověřit přesný název? je v

češtině. Každý takový návod teoreticky tvoří nějaký člověk nebo skupina a udržuje ho v aktuálním stavu, který reflektuje současnost. V praxi tomu tak vždy není, takže je dobré si pohlídat, kdy byla provedena poslední změna takového dokumentu. Ono přece jen návod z roku 1997 není už moc co k čemu. A i v udržovaném návodu mohou být pasáže, které řeší už dávno neexistující problémy. Zrovna CZECH-HOWTO je v tomhle ostudné, takže se jím nenechte vystrašit. Různé hackování knihoven, aby bylo možno v GUI aplikacích používat češtinu je už řadu let pasé.

Potom také existuje LDP, Linux Documentation Project, je přeložen do češtiny a vychází i u nás knižně, aktuální je (březen 2004) 3. vydání. Kniha má přes tisíc stran a je volně stažitelná z internetu. Její součástí jsou i některá přeložená HOWTO.

Další významný zdroj informací je i internet. Ať domácí stránky jednotlivých projektů či distribucí, stránky podobné této, diskuzní servery, linuxové servery, emailové konference v mnoha jazycích včetně češtiny. Tyto mají často tu výhodu, že si v nich můžete o problému s někým zkušeným pohovořit. Je důležité toho nezneužívat a nejdříve se problém pokusit vyřešit sám. Minimální slušnost je prohledání archivu diskuzního fóra, protože skutečně nikoho nebaví vysvětlovat někomu něco, co vysvětloval včera i předevčírem. šetřete čas linuxových guru, ať se mohou věnovat podstatným záležitostem. Je velmi pohodlné, když mám problém, se někoho zeptat a nechat si to po lopatě vysvětlit, ale pamatujte, je pohodlné i vůbec neodpovídat. Vážte si lidí, kteří vám jsou ochotni poradit. Nemyslete si, že to je jejich povinnost a skutečně toho nezneužívejte, ve vlastním zájmu. Někteří guru (a i různá hovada, kteří sice také nic neví, ale proč by si někoho neokřikli) na to reagují podrážděně a někdy pak to líné ťululum ani nechápe, proč jsou na něj všichni oškliví a píší jakési záhadné RTFM. Chyba bývá obvykle na obou stranách, ani guru nejsou svatí. Nejhůř se obvykle chovají ti, kteří toho vlastně ani zas tak moc neví, jsou třeba jen o krok napřed před tazatelem, ale dává jim to pocit mistrů světa a dávají to tazatelům sežrat. Z toho si nic nedělejte a nenechte se tím odradit. Chovejte se slušně, ale ne servilně, blbci jsou všude. Většina fór má určena nějaká pravidla diskuze, zjistěte si je předem a držte se jich.

Upozorním vás na některé významné české zdroje informací:

http://www.root.cz
http://www.abclinuxu.cz
http://www.nyx.cz
http://www.mageo.cz/??
http://

O systému podrobně

Druhy souborů

Jak bylo uvedeno v základech o souborovém systému, Linux podporuje různé druhy souborů. Tato kapitola se jim bude věnovat podrobněji, ale zas ne úplně do mrtě, protože pro exaktní popis by bylo nutno objasnit detailně, jak funguje UNIXový souborový systém, co to jsou i-nody a tak dále. To v úvodu pro začátečníky není asi vhodné. Koho by to zajímalo víc, nalezne dobrý popis v češtině třeba v LDP (Linuxový Dokumentační Pojekt) a to nejn o tomto.

Úplný výčet druhů souborů seřazený podle pravděpodobné četnosti užívání následuje:

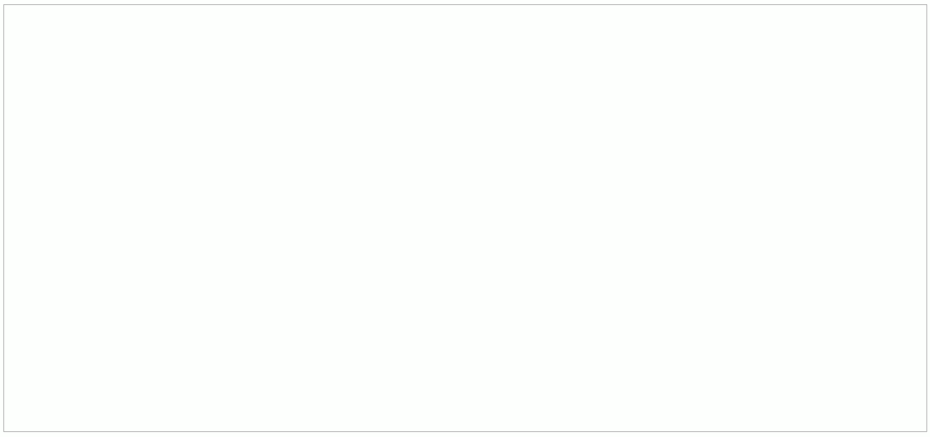
- datový soubor

- adresář (ve Windows známý také jako složka)
- symbolický link
- blokové zařízení
- znakové zařízení
- pojmenovaná roura
- pojmenovaný socket
- pevný link (*)

(*) Pevný link není ve skutečnosti samostatný druh souboru, ale má s nimi úzkou souvislost, takže ho vysvětlím s nimi. Viz dále.

Vzhledem k většímu množství různých druhů souborů je potřeba je od sebe navzájem nějak odlišit, aby uživatel věděl s čím má tu čest. Jejich rozlišení je závislé na správci souborů, každý to dělá jinak, a nejde to tedy popsat nějak univerzálně. Proto uvedu jen pár příkladů.

Tohle je Nautilus, grafický správce programů desktopového prostředí GNOME. Jednotlivé druhy souborů jsou odlišeny ikonkou nebo, v případě symbolického linku, emblémem. Pevný link není nijak odlišen. Mě osobně to nepřijde moc intuitivní, protože různé typy dokumentů mají také různé ikonky, takže tyhle se mezi nimi lehce ztratí. Uvítal bych radši u všech speciálních druhů nějaký emblém jako má symbolický link. Je potřeba vzít v úvahu, že GNOME je skinovatelné, tj. umí snadno na požadavek uživatele změnit svůj vzhled a to i včetně ikonek, takže u vás to může vypadat více či méně jinak.



Tohle je několik různých výpisů příkazu ls na příkazové řádce. Jak ukazuje první výpis, tak soubory nemusí být rozlišeny vůbec, nepoznáte ani co je datový soubor a nebo adresář.

Druhý výpis ukazuje rozlišení pomocí pomocného znaku, není to příliš dokonalé, nerozlišíme tak všechny druhy souborů. Navíc to laika může zmást, mohl by si myslet, že ten pomocný znak je součástí názvu souboru.

Třetí výpis ukazuje rozlišení pomocí barev, to už je mnohem lepší. V RedHatu je to defaultně zapnutý způsob, nemusíte v něm kvůli tomu psát za příkaz žádný parametr, jako to dělám v ukázce. Pro úplnost musím říct, že příkaz ls barevně rozlišuje ještě některé datové soubory i když jsou stejného druhu. Např. ty co mají právo pro spuštění (jsou to tedy programy) vypisuje zeleně a podobně.

Čtvrtý výpis je podrobný výpis, každý soubor je popsán na jednom řádku a je o něm vypsáno mnoho informací. Typ souboru zcela jednoznačně určen hned prvním písmenkem na každém řádku.

A pro úplnost jak to dělá Midnight Commander. Jak je vidět, každý program má svůj vlastní způsob, ale dělá to, je to důležitá informace.

Datový soubor

je obyčejný soubor, který každý zná. Je to textový soubor, obrázek, program a tak podobně. Datových souborů je jednoznačně nejvíc a používají se nejčastěji. Jak už z názvu vyplývá, jeho obsahem jsou data. Máme mnoho druhů dat. Tyto jednotlivé druhy dat se často rozlišují příponami souborů, a ikonkové správce souborů pak dokonce podle těchto přípon používají různé ikonky (ti chytřejší dokonce dokáží soubory rozlišit jen podle obsahu). A třeba takový nautilus dokonce u obrázkových souborů místo příslušné ikonky zobrazuje rovnou náhledy na obrázky. Z pohledu souborového systému se mezi těmito soubory ale nijak nerozlišuje, pro souborový systém jsou všechny tyto soubory jeden druh.

Adresář

je z pohledu souborového systému taky jen soubor, ale se se speciálním významem. Jsou v něm obvykle odkazy na jiné soubory a adresáře umístěné na disku. Pomocí adresářů může uživatel organizovat soubory v logické hierarchické stromové struktuře tak, aby se mezi nimi lépe orientoval. Čistě teoreticky nejsou adresáře vůbec potřeba, všechny soubory by mohly klidně být pohromadě, počítači by to nevadilo, ale chudák člověk by se v tom nevyznal.

Jak adresáře fungují asi není potřeba nějak zvlášť vysvětlovat. Disk má jeden kořenový adresář a v něm mohou být odkazy na soubory a další adresáře. Tyto adresáře mohou obsahovat další odkazy na další soubory a další adresáře a tak dále. Nemůže to tak být ale do nekonečna, adresář je soubor, takže zabírá na disku místo, a tedy s každým novým adresářem nám místo na disku ubývá a jednou se musí vyčerpat. Navíc některé souborové systémy mohou mít i jiné limity, které omezují počet adresářů nebo jejich úrovní, ale při běžném provozu se s nimi nesetkáme, bývají dostatečně dimenzované.

Symbolický link

U symbolických linků musím udělat největší zastávku. Jsou velmi užitečné, v Linux jsou běžné a hojně užívané a uživatelům Windows neznámé.

Symbolický link je něco jako zkratka, principem je velmi podobný odkazu v html souboru, ale funguje v rámci souborového systému. Je velmi vzdáleně podobný zástupci ve Windows, což je nedokonalá napodobenina symbolického linku. Zástupce ve Windows je obyčejný datový soubor, který některé programy speciálně interpretují a jiné nikoliv. Tj. podpora zástupců je záležitost jednotlivých aplikací. Symbolický link je záležitost podstatně nižší úrovně, je to součást souborového systému a jeho interpretaci má na starost jádro. Symbolický link je speciální soubor stejně jako je speciální soubor adresář.

Jak jsem již uvedl, adresáře slouží k vytváření hierarchické stromové struktury logické organizace souborů. Symbolické linky slouží de fakto k boření hierarchie této struktury :-). Umožňují vytvářet různé zkratky. V jednom adresáři můžete mít odkaz na soubor (libovolného druhu) v jiném adresáři ať je tento kdekoliv a dokonce i dočasně nedostupný (třeba na vyjmutém CD). Kromě toho slouží k vytváření alternativních názvů souborů v rámci jednoho adresáře, to se hodně používá z různých důvodů, viz. dále.

Symbolický link si můžete představit jako normální soubor, jehož obsahem je adresa (cesta, path) na jiný soubor (i speciální, tedy adresář, rouru a podobně). Přesně to totiž symbolický link je. Akorát že to není normální soubor a ta cesta není uložena v datové oblasti (symbolický link nic takového nemá) takže ho třeba nemůžete otevřít v txtovém editoru a podívat se na ni, ale jinde. Tato cesta může být jak relativní (vztažená k umístění symbolického linku, tedy např. ../jinySoubor.txt) tak absolutní (tedy vztažená ke kořenovému adrsáři např: /home/wraith/jinySoubor.txt). Rozdíl je ten, že když zkopírujete symbolický link s relativní cestou, tak může ukazovat jinam, kdežto když zkopírujete symolický soubor s absolutní cestou, bude ukazovat pořád na to samé místo. (pro ty, kteří rozumí html: je to stejné, jako u relativních a absolutních html odkazů)

Práce se symbolickým linkem je prakticky přirozená, nejlépe to bude vidět na pár příkladech. Mějme textový soubor text.txt a symbolický link link.txt, který ukazuje na ten text.txt.

a) Když v textovém editoru otevřete soubor link.txt, otevře se ve skutečnosti soubor text.txt a budete pracovat přímo s ním. Když totiž editor požádá operační systém (jádro) o otevření souboru link.txt, tak tento ví, že se jedná o symbolický link a místo něj otevře soubor na který symbolický link ukazuje. Tj. jádro se o to postará. Takto to ale funguje (naštěstí) jenom při otvírání symbolického linku!

b) Při jakékoliv jiné činnosti, tj mazání, přejmenování, přesouvání či kopírování symbolického linku, pracujete s ním, tj. pak skutečně smažete, přejmenujete, přesunete a nebo zkopírujete ten link a ne ten soubor na který ukazuje. Je to intuitivní a přirozené, ani vás nenapadne o tom nějak přemýšlet.

V případě přesunutí nebo kopírování, když symbolický link obsahuje relativní adresu, tak se upraví, aby odpovídala novému umístění a pořád ukazovala na ten samý soubor. ???!!!??? boze, co to placam ???!!!??? To všechno opět provede operační systém automaticky, uživatel ani programátor správce souborů se tím nemusí zabývat. Takže když chcete smazat symbolický link, tak ho jednoduše smažte, nemusíte se bát, že si smažete soubor na který ukazuje. Se souborem na který symbolický link ukazuje se pracuje jen při otvírání linku pro čtení a zápis.

c) V některých případech můžete vyžadovat jiné chování. Například když budete kopírovat nějaký adresář někam (na disketu, budete ho chtít zálohovat na CDR a pod), tak můžete chtít, aby se symbolický link nahradil souborem na který ukazuje. Tohle speciální chování musí ošetřit programátor alikace a umožnit vám takové speciální chování zapnout. Třeba v Midnight Commanderu při kopírování souborů máte možnost zatrhnout volbu následovat linky, která právě toto zajišťuje. Podpora takového speciálního chování je tedy už aplikačně závislá.

Symbolický link nemusí ukazovat jen na datové soubory. Může ukazovat ne libovolný druh souboru, třeba na adresáře. Je užitežné (alespoň já to dělám a dělám to všem lidem, kterým instaluji a konfiguruji Linux) si ve svém domácím adresáři udělat adresář který obsahuje hromadu symbolických linků, které ukazující na často navštěvované adresáře rozmístěné po různých discích co máte v počítači jako zkratky, které umožňují rychlý přístup k nim. Do těchto frekventovaných adresářů se ze všech aplikací pak dostanete nanejvýš na dvě kliknutí, protože domácí adresář je v Linuxu výchozí bod, od kterého se pokračuje dál. A aby to nebylo málo, můžete si uděla symbolický odkaz na tento adresář třeba i ze své plochy. Jak jsem řekl v úvodu, symbolické linky jsou užitečné a hojně používané.

Další obvyklé použití symbolických linků je unifikace názvů zařízení. V Linuxu je většina zařízení prezentována jako speciální soubor (k tomu se také dostanu). Třeba CD mechanika je podle toho kde je připojena reprezentována jako soubor zařízení (viz. dále) /dev/hdb /dev/hdc /dev/scd1 a podobně. Takže programům které s CD mechanikou pracují je potřeba říct, kde se nalézá. Může se stát, že si koupíte nový disk a připojíte ho na místo, kde byla CD mechanika a tu zapojíte jinam. Pak musíte překonfigurovat všechny programy, které s touto CD mechanikou pracují. To je pochopitelně děsně nepraktické. Mohem praktičtější je udělat symbolický link /dev/cd, který ukazuje na příslušný soubor, jenž reprezentuje CD mechaniku a při změně zapojení pak stačí změnit jen tento symbolický link. Aplikace, které používají teno symbolický link pak změnu zapojení ani nepoznají. Takto to dělají automaticky snad všechny distribuce Linuxu. A nejen co se týče CD mechaniky, najdete tak i symbolické linky /dev/mouse, /dev/modem a podobně, které se i lépe pamatují.

Další obdobné použití souvisí s přejmenováváním programů. Chování programu můžete modifikovat parametry na příkazové řádce. Např. textový editor můžete spustit příkazem "editor text.txt" nebo "editor -r text.txt". V druhém případě ale soubor bude otevřen jen pro čtení (r = read), tj. bude možno ho jen prohlížet, ne měnit. Programátor editoru může jednoduše zjistit, jak se nazývá spouštěcí soubor programu a když program spustíte přes symbolický link, tak tento název bude odpovídat názvu symbolického linku. Proto se v Linuxu občas používá ještě jeden způsob modifikace chování programů, a to spouštění přes symbolické linky. Když soubor "editor" spustíte přes symbolický link "prohlizec", tak program "o tom ví" a chová se jen jako prohlížeč. Musí to v něm být takto pochopitelně naprogramováno. Používá to např. textový editor vim na který může ukazovat mnoho různých symbolických linků, které modifikují jeho chování. Namátkou gvim (zapíná grafické rozhraní) nebo view (zapíná režim jen pro čtení). Takto si můžete lehce ošetřit i vlastní skripty, až si je budete psát.

Tohle nejsou všechna možná použití symbolických linků, jen příklady, které ukazují rozmanitost využití a jejich užitečnost. Nebojte se jich, naučte se je kreativně používat, ušetří vám mnoho práce i klikání.

Symbolickým linkům nelze nastavit přístupová práva, vždy se používají práva, která má soubor, na nejž symbolický link ukazuje.

Když smažete soubor, na který ukazuje symbolický link, tak symbolický link poté ukazuje do nikam. Pokus o jeho otevření pro čtení nebo zápis se chová stejně jako

pokus o otevření neexistujícího souboru. Správci souborů takové symbolické linky často výstražně zvýrazňují, třeba červenou barvou. Viz ukázky na začátku kapitoly, symbolický link ukazující do nikam zajisté poznáte sami.

Blokové a znakové zařízení

slouží jako rozhraní pro přístup k většině hw zařízení. Např. disku, klávesnici, zvukové kartě a podobně. V Linuxu se zařízení rozlišuje na dva základní typy, blokové a znakové, každé má svůj speciální druh souboru. Blokové zařízení jsou zařízení, k jejichž datům se dá přistupovat náhodně a načítají se či zapisují po blocích, to jest disky, cd romy a podobně. Znakové zařízení jsou zařízení, u kterých se k datům přistupuje jako k proudu a po bajtech, to jest např. myš, tiskárna a podobně.

Čtením těchto souborů čtete data ze zařízení, zapisováním do nich zapisujete data na zařízení. Takto komunikují programy v Linuxu téměř s každým zařízením. Výjimkou jsou síťové karty/zařízení, pro které má jádro speciální rozhraní a programátoři k nim tedy také musí přistupovat speciálně, ale to jen naokraj.

Jejich pojmenování se řídí vcelku složitými pravidly, které popisuji v následující kapitole názvy zařízení (!!! link). Tyto názvy a pravidla jsou ale jen pro lidi. Operační systém se jimi neřídí, z jeho pohledu mohou být názvy těchto souborů libovolné. Operační systém (Linux) se řídí tzv. čísly zařízení, ta jsou dvě, major a minor, major číslo označuje druh zařízení, minor pak bližší určení. Např. major číslo X znamená, že jde o pevný disk a minor číslo Y, že jde o druhou partition master disku na sekundárním řadiči. Pro lidi je pojmenován jako hdc2. Major a minor číslo mimochodem můžete vidět na ukázce u podrobného výpisu příkazu ls. Jsou napsána na místě, kde se u datového souboru vypisuje jeho velikost. Ty soubory jsem vytvořil sám, a zadal jsem je jako nuly.

Tyto speciální soubory mohou být teoreticky umístěny všude, ale prakticky je naleznete jen v adresáři /dev (od device = zařízení) a jeho podadresářích. Těch souborů je tam ohromné množství a jsou snahy tento adresář zvirtualizovat. Tj. udělat z něj virtuální souborový systém, podobně jako je tomu u adresáře /proc. První pokus nebyl moc úspěšný a nepoužívá se, ale prý bude učiněn druhý pokus. Uvidíme, co nám vývojáři jádra připraví. Výhodou by mělo být, že v takovém adresáři nebude ta ohromná spousta souborů, ale jen soubory reálně existujících zařízení v příslušném počítači. Tyto soubory by se navíc měly vytvářet dynamicky, např. po připojení digitálního fotoaparátu k počítači přes USB.

Pojmenovaná roura

slouží k výměně dat mezi programy. Uživatel neprogramátor s nimi moc do styku nepřijde, jen je sem tam občas uvidí. Bohužel bývají často umístěny v adresáři /tmp, takže zde není vhodné bezhlavě mazat soubory. Vedlo by to k nestabilitě některých programů, třeba samotného X Sereru, který má na starost grafické rozhraní. To by byl docela závažný problém. On si ho sice při startu umí sám vytvořit, ale když mu ho smažete během jeho činnosti, tak se mu to moc nelíbí.

Pojmenovaný socket

Z pohledu neprogramátora je to to samé, co pojmenovaná roura, používá se to k těm samým účelům. Nazývá se také UNIX socket.

Pevný link

Je na první pohled i podle názvu obdoba symbolického linku. je to ale něco úplně

jiného. Jak jsem již uvedl, adresář je speciální soubor, který obsahuje odkazy na soubory (data) které se nalézají na disku (tedy datové soubory, jiné adresáře, symbolické linky a tak dále). Pevný link není nic jiného, než další takový odkaz na již existující soubor.

Takže to vlastně není ani speciální druh souboru, jen další položka adresáře. Ve své podstatě je každý soubor pevný link. Zmiňuji se o něm proto, že v Linuxu může na jeden soubor vést několik takových odkazů z různých adresářů.

Z toho plynou určite vlastnosti pevných linků, kterými se liší od symbolických.

a) Mohou fungovat jen v rámci jednoho souborového systému (disku). Symbolické linky jsou definovány cestou k souboru a tak mohou ukazovat i na soubory na jiných souborových systémech a dokonce i na (třeba jen dočasně) neexistující soubory. To u pevných linků není možné.

b) Po vytvoření pevného linku se na jeden soubor odkazují dva názvy, tj. pevné linky. Nelze rozlišit který název je původní a který dodaný později, oba mají stejné vlastnosti.

Příklad. Mám soubor bonz.txt a na něj si udělám pevný link hlaseni.txt. Oba soubory budou mít totožný obsah a když ho změním u jednoho, změní se i u druhého, protože jde o jedny a ty samá data (obsah). Mohu smazat kterýkoliv z těchto souborů, obsah druhého bude zachován. Teprve po smazání i druhého dojde k odstranění dat (obsahu) souboru z disku. Funguje to tak, že hlavička dat na disku má v sobě čítač pevných linků, které na ně ukazují a k odstranění dat dojde až když klesne na nulu.

!!! Zjistit jak je to s nastavenim prav, zda je zdileno ci ne.

Pevné linky se moc nepoužívají, symbolické linky jsou pružnější a také přehlednější.

Názvy zařízení

V adresáři /dev se nalézají soubory reprezentující různá zařízení. Takových zařízení je mnoho, tedy i souborů v /dev je mnoho. Já jich tam mám přes 14 000. V tom je těžké se orientovat. Naštěstí v praxi je potřeba jich znát jen minimum. Je to potřeba znát především při konfiguraci systému a připojování souborových systémů.

- **ttyS – sériový port**
  - /dev/ttyS0 – první (ve win COM1)
  - /dev/ttyS1 – druhý (ve win COM2)
- **lp – paralelní port**
  - /dev/lp0 – první (ve win LPT1)
  - /dev/lp1 – druhý (ve win LPT2)
- **psaux – ps/2 port**
  - /dev/psaux – ps/2 port (ve win ??)
- **fd – disketová mechanika**
  - /dev/fd0 – první (ve win a:)
  - /dev/fd1 – první (ve win b:)
- **hd – IDE zařízení (hard disk, cd, dvd)**; man 4 hd
  - /dev/hda – primary master
  - /dev/hdb – primary slave
  - /dev/hdc – secondary master
  - /dev/hdd – secondary slave
  - /dev/hda1 – první primary partition na /dev/hda
  - /dev/hda2 – druhá primary partition na /dev/hda
  - /dev/hda5 – první logický disk v extended partition na /dev/hda

    - /dev/hda6 – druhý logický disk v extended partition na /dev/hda
- **sd – SCSI zařízení**; man 4 sd
    - /dev/sda – první
    - /dev/sdb – druhé
    - /dev/sda1 – první primary partition na /dev/sda
    - /dev/sda5 – první logický disk v extended partition na /dev/sda

Označování disků je poněkud komplikované. Vychází ze skutečného hardwarového provedení. V domácích počítačích, v současné době, jsou převážně IDE disky. Tyto disky se připojují k IDE řadiči. K jednomu IDE řadiči lze připojit jen dva disky. Kromě disků se k nim připojují i IDE CD (-ROM, -R, -RW) a DVD (-ROM, atd) mechaniky. Jedno z těchto zařízení se musí mechanicky (přepínači) označit jako master (pán, hlavní) a druhé jako slave (otrok, vedlejší). Protože dva disky jsou málo, bývají v soudobých počítačích dva až tři IDE řadiče, ty se označují jako primary (primární), secondary (sekundární) atd.

Takže zařízení připojené k sekundárnímu řadiči a nastavené jako hlavní (primary master) je označeno /dev/hdc. Když je toto zařízení CD-ROM, pak se připojuje jako /dev/hdc. Když je toto zařízení pevný disk, je to složitější.

Pevné disky se softwarově rozdělují na jednu až čtyři částí. Tyto části se označují jako 1. až 4. primary partition. Tyto se číslují jako /dev/hda1 až 4. Z historických důvodů nelze disk rozdělit na víc partition. Protože je to ale občas potřeba, může být poslední partition (ne nutně ta čtvrtá) označena jako extended (rozšířená) partition a ta se pak dá rozdělit na 1 až 28 (nebo 60??) logických disků. Tyto se pak označují jako /dev/hda5, /dev/hda6 atd. Takže když máme disk rozdělen na jednu primární a jednu rozšířenou partition a tu rozšířenou na 4 logické disky, tedy z laického pohledu na pět disků, tak jsou tyto disky označeny jako /dev/hda1, 5, 6, 7 a 8. Pevný disk lze ale rozdělit na pět částí i v jiné kombinaci, třeba /dev/hda1, 2, 5, 6, 7. Jak je to konkrétně na vašem systému se můžete podívat do /proc/partitions, sloupec name.

Prosím ostatní zkušené uživatele aby mi pomohli doplnit seznam názvů různých zařízení, které považují za důležité, často potřebné. Já s jinými nemam zkušenosti.

V adresáři /dev bývají také symbolické linky se srozumitelnějšími jmény jako /dev/cdrom, /dev/mouse nebo /dev/modem, které ukazují na konkrétní zařízení. Když vám nefunguje např. myš a v konfiguračním souboru ji máte vedenou jako zařízení /dev/mouse, pak je mimo jiné potřeba ověřit, jestli ukazuje na na správný port.

kompletní popis

Připojení souborového systému – mountování

Obecně jsem o připojování souborových systémů již mluvil v sekzi základy v kapitole o souborových systémech. Na nejnižší úrovni se připojování provádí příkazem mount a proto se tato činnost nazývá také mountování, čehož se budu držet.

Mountování je činnost ve Windows tařka nevídaná, i když prý v novějších Windows teoreticky také možná. Nejbližší běžný ekvivalent mountování ve Windows asi je připojování sdílených (síťových) disků. S tím se někteří z vás možná setkali. Mým kolegům v práci tato občasná činnost dělá velké problémy, takže si nedělám žádné iluze o začátečnících v Linuxu. Mountování bude činnost neoblíbená a někdo kvůli tomu moná i Linux zatratí.

Uživatel Windows tedy připojování disků nezná, buď má disk přístupný pod nějakým

písmenkem a nebo ne. Výměnná média se tam pod písmenka mountují samy. V Linuxu je tomu jinak a upřímně řečeno, je to docela pruda a jsou s tím spojeny určité problémy. Ne obecně, ale ve spojení s výměnnými médii, především disketami. Naštěstí se dnes už tak moc nepoužívají.

Mountování není zas až tak nesmyslné, má své opodstatnění. Vlastně k němu dochází i ve Windows, ale automaticky. V Linuxu se to musí dělat ručně a je to složitější. Tedy, po pravdě řečeno, i pro Linux existují různé automountery (automount, supermount), ale já s nimi nemám skoro žádné zkušenosti a ty co mám jsou z kdysi dávno a nejsou moc dobré, takže možná nejsem tak úplně způsobilý k napsání této kapitoly. Ale při dodržení pár zásad není ani ruční mountování zas tak hrozné a v případě problémů je dobré vědět co kdy jak a proč, takže to co zde napíšu snad nebude příο k zahození.

Ve Windows je to jednoduché, na disketě musí být souborový systém FAT12, Windows jiný nepodporují. Navíc tam nejde zvolit jaké se má používat kódování češtiny, bezpečnostní nastavení nebo režim jen pro čtení a podobně. To se to pak automaticky mountuje docela snadno. V Linuxu je možno použít mnoho různých nastavení. Takže při použití nějakého toho automounteru se nevyhnete prvotnímu nastavení.

Souborový systém (libovolný a z libobovolného zařízení) se mountuje příkazem mount. Všechny volby lze zadat jako parametry toho příkazu. To je ale zbytečně pracné, zvláště když ty parametry jsou až na výjimečné situace u jednoho zařízení prakticky pořád stejné. Proto existuje konfigurační soubor /etc/fstab v kterém si můžete mountování často používaných zařízení (souborových systémů) předkonfigurovat. Potom už stačí příkazu mount zadat jen přípojný adresář nebo soubor reprezentující zařízení. Z toho si příkaz mount odvodí co chceme přimountovat a ostatní parametry si už doplní z konfiguračního souboru sám.

Tento konfigurační soubor používají i při jiných přílažitostech a dokonce se podle něj orientují i jiné programy. Například se z něj pozná, které souborové systémy se mají přimountovat automaticky při startu počítače. Orientují se podle něj desktopová rozhraní, jenž pak umožní jednoduché mountování cédéčka na jedno kliknutí nebo dokonce automaticky při vložení média do mechaniky. Používá ho také Gkrellm, který umožňuje velmi pohodlné mountování i začátečníkům (popisuji ho v sekci programy) a mnoho dalších. Tento konfigurační soubor by se tedy měl naučit používat každý uživatel, který si sám administruje Linux.

Jen pro zajímavost, Linux umožňuje vzít soubor, ten naformátovat nějakým souborovým systémem a pak ho přimountovat jako disk. Používá se to často se souborovým systémem iso9960 (cd rom). Když si z internetu stáhnete iso image cd romu (což není nic jiného než soubor naformátovaný jako cd), ten přimountovat a pracovat s ním jako s normálním cédéčkem, nemusíte si ho napřed vypalovat. A nebo naopak, při vypalování dat si můžete nejprve vyrobit tento iso image a před vypálením ho zkontrolovat, zda je v pořádku.

Ty parametry které se zadávají nejsou v zásadě nic světoborného. Základní jsou tři. ODKUD, tedy na jakém zařízení se souborový systém nachází. Zařízení je reprezentováné speciálním souborem zařízení, viz přdchozí dvě kapitoly. KAM, tady ke kterému adresáři se souborový systém má přimountovat. A CO, tedy co to je za souborový systém. Potěšující je, že na moderním Linuxu stačí zadat auto a pokud to není exotický souborový systém, tak se rozpozná sám. Nepovinně je možno zadávat ještě různá nastavení, kterých je spousta a mohou se lišit podle druhu souborového systému. Viz man mount.

Zatím to nevypadá nijak složitě. Problémy začnou při pokusu odpojit souborový sstém. To se totiž nemusí zdařit. Nezdaří se to tehdy, když nějaký program (třeba textový

nebo grafický editor) má na tom souborovém systému otevřený soubor nebo když nějaký adresář na tom souborovém systému slouží jako pracovní adresář toho programu (typicky bash nebo mc). Stačí ten soubor zavřít, přesunout se do jiného adresáře nebo ten program ukončit a souborový systém půjde odpojit, ale je to komplikace. Zvláště když nevíte který program to je a třeba i spěcháte.

Přitom ten souborový systém musíte odpojit. Bez toho vám u cd rom mechaniky nepůjdou otevřít dvířka (to je ten lepší případ) nebo, u diskety a podobných zařízení, kde fyzickému odpojení nejde zabránit, můžete přijít o data a navíc už v daném zařízení žádné nové médium nepřimountujete až do restartu počítače.

Ke ztrátě dat může dojít proto, že Linux používá vyrovnávací diskovou paměť (buffer, cache) a k zápisu dat na médium nedochází okamžitě, ale až po čase. Práce se zařízením je tak mnohem rychlejší. Když dáte pokyn k odpojení souborového systému, tak se nejdříve všechna data uloží a teprve potom dojde k jeho odpojení. Když vyáhnete disketu (USB disk a podobně) před odpojením, tak k uložení obsahu vyrovnávací paměti na médium nedojde a tyto data jsou ztraceny. Přitom nemusí jít jen o samotný obsah souborů, ale i o data samotného souborového systému u nějž tak může dojít k částečnému poškození.

Tento problém nastane (a nejen v Linuxu) i když špatně vypnete počítač, třeba výpadkem elektřiny. Proto se při následném startu počítače kontroluje integrita souborových systémů. U žurnálových souborových systémů jsou rizika poškození tímto způsobem minimalizovány.

Když tedy vytáhnete disketu z mechaniky bez odpojení, koledujete si o ztrátu dat. A nejen to, když do mechaniky dáte novou nebo tu samou disketu, tak už se vám nepodaří znovu přimountovat, mechanika zůstane zablokovaná. Nejsem si jist, jestli to platí i u USB zařízení. Tento jev jsem nezkoumal, ale u kamaráda jsem pozoroval, že se mu vypnul automaticky fotoaparát který byl zrovna přimountován jako USB disk. Po jeho opětovném zapnutí šel opět normálně přimountovat jako to samé zařízení. Možná že USB rozhraní je oddolnější. Možná ale také foťák před svým vypnutím USB rozhraní o tomto informoval a to ho odpojilo automaticky, nevím.

Velké zkušenosti mám s pcmcia redukcí pro CF paměťové karty. Ta karta se prezentuje jako virtuální zařízení. Při špatném odpojení bez odmountování se to virtuální zařízení zablokuje podobně jako disketová mechanika. Karta lze poté znovu přimountovat, ale jako jiné virtuální zařízení. Převedeno do řeči windows, jako jiné písmenko. To je lepší než nic, ale také to nijak neohromí. Všechny tyhle problémy by mělo řešit hotlug rozhraní, ale do toho zatím moc nevidím, možná že ten fotoaparát přimountovaný přes USB ho už používá. Do progresivních linuxových technologií zas až tak moc nevidím.

Z toho vyplývá jedno. Mountování není nic nepochopitelného a tragicky nesrozumitelného, ale je to pruda a vyžaduje to disciplínu při zacházení s výměnnými médii. Dokud ji Linuxový začátečník nezíská, bude s tímhle jevem zápasit.

Když se dostanete do situace, že vám nějaký souborový systém nejde odpojit, a vy nemůete zjistit proč, pak si můžete pomoci příkazem fuser nebo ???. Viz moje stránky LinuxPoint.

Konkrétním příkladům mountování souborového systému se věnuje v sekci konfigurace kapitola mountování. Pěkný grafický prográmek který vám mountování usnadní se jmenuje Gkrellm, který je popsán v sekci programy.

Konfigurační soubory

Tato kapitola má za úkol popsat systém konfiguračních souborů v Linuxu.

GUI

Tato sekce se věnuje GUI v Linuxu. GUI je zkratka z Graphics User Interface, česky grafické uživatelské rozhraní. Bude zde popsáno, jak vlastně GUI v Linuxu funguje, od hardware, přes fonty a podobné záležitosti až po nejčastěji používané knihovny, které programátoři používají při tvorbě aplikací s gafickým rozhraním a které mají velký vliv na to, jak aplikace v Linuxu vypadají a co umí nebo neumí.

Úvod

Z předchozích kapito víte, že v Linuxu si můžete vybrat, zda ho budete používat s grafickým nebo textovým rozhraním. Obojí má své výhody i nevýhody a hodí se pro různé situace. Při použití počítače jako pracovní stanice, při kancelářském nebo domácím použití, při práci s grafikou a podobně dá naprostá většina uživatelů přednost grafickému rozhraní. Grafické rozhraní může mít více podob.

Za normální se považuje tzv. okenní systém. Obrazovka představuje plochu na které jsou umístěna okna, která představují rozhraní mezi uživatelem a spuštěným programem. Program může být reprezentován jedním i více okny (nebo i žádným, pak říkáme, že běží na pozadí. Takový program může mít okno jen schované a nebo se jedná o program, který grafické rozhraní vůbec nepodporuje). Je možné a obvyklé mít spuštěno více programů současně a pak i jejich okna mohou být na obrazovce současně. Takto fungují moderní operační systémy jako Windows (které prakticky nic jiného ani neumožňují) a nebo X Window System, který jako volitelné grafické rozhraní používá Linux (a řada dalších UNIXových OS).

Je možný i jiný, primitivnější, model, který se používal třeba v DOSu. Samotný operační systém běží v textovém režimu, ale některé aplikace mohou přepnout grafickou kartu do grafického režimu a samy o sobě poskytovat grafické rozhraní. Rozhraním mezi uživatelem a programem je pak celá obrazovka. Okna programů neexistují, takže není možné mít na obrazovce rozhraní k víc programům najednou. Linux samotný, jak zde už mnohokrát zaznělo, umožňuje běh v textovém režimu a existují i některé linuxové aplikace, které si samy umí poskytovat grafické rozhraní, tak, jako tomu bylo v DOSu. Ty se pak obejdou bez X Window Systému. Tyto aplikace používají buď SVGAlib nebo frame buffer. Nepoužívá se to příliš často, ale možné to je. Je to vhodné obzvláště na slabších konfiguracích, protože je to systémově méně náročné. Dokonce i samotná konzole (příkazový řádek) může být poskytována přes frame buffer a textové rozhraní tedy jen emulovat v grafickém.

Jestli se ptáte k čemu je dobrá taková ptákovina jako je používání textového rozhraní v textovém režimu emulovaném v grafickém režimu, tak např. v tom, že si můžete nastavit rozlišení a zobrazovací frekvenci monitoru (to v textovém režimu grafické karty nejde, tam je to napevno) a nebo že v textovém režimu lze zobrazit max. 512 různých znaků, což je dostatečné pro 8 bitové kódování (256 různých znaků), ale už ne pro 16 nebo 32 bitové Unicode (až 65 tisíc nebo 4 miliardy) různých znaků. To je tedy (minimálně teoretická) cesta, jak i v textovém režimu moci používat na Linuxu Unicode, myslím že se tohle zatím bohužel nepoužívá. Ale hlavně to umožňuje na konzoli zobrazit obrázek, třeba při startování Linuxu. Jde tedy hlavně o efekt, než nejaký praktický přínos.

Tento druhý model je spíš rarita než cokoliv jiného. Když někdo bude mluvit o grafickém rozhraní (GUI) Linuxu, bude tím myslet X Window Systém. Pak ještě existují další modely, či spíše alternativy, s těmi se ale na PC nesetkáte. Existuje např. velmi

odlehčená náhrada X Window Systému, která se používá na PDA, na kterých běží Linux (třeba Zaurus). Zbytek sekce se věnuje X Window Systému, který na desktopu jednoznačně převažuje.

X Window Systém

Grafické rozhraní v Linuxu není povinné, ale volitelné. Kromě toho není ani součástí jádra, jako ve Windows 95+, ale je realizováno obyčejným programem. Když ho spustíte, poskytne vám grafické rozhraní. Tak nějak fungovaly třeba Windows 3.x, které poskytovaly grafické rozhraní nad DOSem. V Linuxu je ale mnohem vyspělejší, Windows jsou proti tomu slabý odvar, jak se dozvíte.

Grafické rozhraní Linuxu je standardizováno a je shodné s grafickým rozhraním ostatních UNIXů. Tento standard se jmenuje X Window System. Implementace tohoto standardu používaná v Linuxu se jmenuje XFree86. Je jedním z mála jasně zavedených a všeobecně přijímaných linuxových programů, který nemá žádné významné alternativy a konkurenty.

V současné době (duben 2004) se ale situace začíná trochu komplikovat. Před nedávnem XFree86 změnilo licenční politiku (u dnes aktuální verze 4.4), která se nelíbí tvůrcům linuxových distribucí. Tvrdí, že za nových licenčních podmínek nemohou XFree86 do distribucí zařazovat. Takže nové distribuce které právě teď vznikají používají stále verzi 4.3 a jak to bude do budoucna se uvidí. Buď XFree86 opět změní licenční podmínky a nebo, jak se zdá, tu existují přinejmenším dva použitelné klony, které by bylo možno použít. Jeden z klonů má na starost jeden z bývalých vývojářů XFree86 a zavádí do něj novinky jako podpora průhlednosti a podobně, takže to nemusí být ani zas až tak moc na škodu. Při dodržení standardu X Window System by ani nemělo dojít ke stavu, že aplikace pro svou funkci bude potřebovat tu a nebo tu konkrétní implementaci, což by bylo velmi nepříjemné. (Červenec 2004 - praktick všechny hlavní distribuce přešly z XFree86 na X.org, které vzniklo jako paralelní vývojová větev z poslední verze XFree86 (4.4rc2), která výšla pod licencí GPL.)

Standard X Window System se začal vyvíjet v roce 1983. Takže to v žádném případě není kopie Windows, jak si někteří myslí. Byl tady dřív a je od Windows také zásadně odlišný, nedejte na první dojem. Několik prvních let se standard X Window Systém překotně vyvíjel a v rychlém sledu vznikaly nové a nové verze. Od roku 1987 vznikla verze X11, která je aktuální dodnes. Ne že by se tím vývoj zastavil, ale standard se už nemění, jen rozšiřuje. Vznikají takzvaná vydání (release), dnes je aktuální release 6.4??overit?.

X Window System není, jak už vyplývá z názvu jen nějaký program, je to celý poměrně hodně komplikovaný systém, který já nazývám grafický subsystém. Tento systém je definován obsáhlou řadou dílčích standardů. Které definují architekturu, API, podpůrné aplikace, způsob komunikace a přenos dat mezi aplikacemi a řadu dalších podobných věcí. Není potřebné a ani možné to všechno nějak dopodrobna znát. Některé věci jsou důležité jen pro programátory nebo administrátory, ale i uživatelé by měli znát třeba základní architekturu, protože jim to umožní pochopit jak XFree86 funguje a co jim díky tomu může nebo nemůže nabídnout. Některé věci jsou skutečně zajímavé a ve Windows nevídané. Stačí je jen znát a umět používat.

Základní architektura X Window Systému

X Window Systém má modulární architekturu. Rozhraní mezi těmito moduly je jednoznačně standardizováno, což zaručuje dobrou kompatibilitu a možnost jak

výměny modulů, tak i bezproblémovou komunikaci mezi různými platformami, které používají X Window System.

X Server a klienti

X Window Systém má architekturu klient/server. X server se je program, který poskytuje vlastní grafické rozhraní a který ovládá grafickou kartu. Je to tedy program, který běží na počítači u kterého sedíte (obvykle je server nějaký vzdálený počítač, takže některým lidem dělá obtíže tohle pochopit). Klienti jsou pak programy, které toto grafické rozhraní používají. Teda třeba grafický editor, nějaká hra, www prolížeč a podobně.

X Server je velmi důležitý prvek. Je to takový malý operační systém sám o sobě. Za prvé přímo ovládá grafickou kartu. To znamená, že ji musí znát a to znamená, že pro ni musí mít ovladače. A skutečně, jádro Linuxu se o grafickou kartu prakticky nestará. Narozdíl od Windows běží grafické rozhraní Linuxu v uživatelském prostoru, ne režimu jádra. Že to nemusí být tak úplně pravda vysvětlím později, až budu vysvětlovat jak to je implementováno v Linuxu. Za druhé kromě toho že má na starost výstup, tedy grafickou kartu, tak má na starost také vstup, tedy např. myš a klávesnici (ale i všechno ostatní, jako touchpad, joysick, tablet a co vás napadne). Bohužel se to netýká zvukových karet, o ty se stará jádro. X Server má tedy na starost kompletní komunikaci s uživatelem, kreslí mu na obrazovku obrázky a přijímá od něj pokyny z různých vstupních zařízení.

X Protokol

Klient komunikuje s X Serverem pomocí standardizovaného X Protokolu. Komunikace probíhá oboustranně. Klient dává X Serveru požadavky jako od bodu A se souřadnicemi x1, y1 do bodu B s souřadnicemi x2, y2 nakresli obdélník vyplněný bílou barvou. X Server naopak klienta informuje o událostech jako uživatel stisk druhé tlačítko myši na souřadnicích x, y. Velmi důležité je, že tato komunikace probíhá přes síťové rozhraní. To znamená, že klient může klidně běžet na jiném počítači než X Server. V praxi to znamená, že na svém počítači můžete ovládat programy, které běží na jiném počítači tak, jako by běžel na vašem počítači. Není na tom nic zvláštního, používám to často, přihlásím se ke vzdálenému počítači třeba přes ssh (nebo telnet), spustím na něm grafický program a jeho grafické rozhraní se objeví na mém počítači. Není problém ani ovládat z vašeho počítače několik programů spuštěných na různých počítačích. Komunikace přes síťové rozhraní probíhá i když X server i klient běží na jednom a tom samém počítači. X Serveru to je jedno, kde klient běží, ke všem klientům s chová stejně. U tohoto pravidla je možno uplatnit výjimku tam, kde je nutno dosáhnout velmi vysokého grafického výkonu. Např. při přehrávání filmů. Tomuto se budu ještě později věnovat.

Jestli hned teď zkoušíte vzdálené spuštění programu, nemělo by to jít (záleží na konfiguraci), měli byste obdržet chybovou hlášku ??o chybne autorizaci?. Ale stačí spustit na vašem počítači (tedy na tom na kterém chcete mít grafické rozhraní) příkaz `xhost adresa`, kde adresa je adresa vzdáleného počítače. To je bezpečnostní opatření, aby kdejaký hejhula neotvíral na vašem počítači okna programů běžící někde jinde. Tím příkazem dáte X Serveru na vědomí, že programy běžící na počítač se zadanou adresou to mají povolené. Pak už by to mělo chodit, pokud se vám do cesty nepostaví ještě nějaké další omezení, jako je firewall, maškaráda (NAT) a podobně.

Xlib a toolkity

Programátor klienta (grafické aplikace) nemusí znát X Protokol a nemusí ho

implementovat do aplikace (to by se pěkně nadřel), ten se používá transparentně. Je implementován v knihovně Xlib, jejíž funkce může programátor volat. Tato knihovna je ale nízkoúrovňová, programování v ní je obtížné podobně jako programování v asembleru. Umožňuje kreslit na obrazovku základní primitiva jako čtverec, oblouk a podobně či nízkoúrovňovou obsluhu událostí. Takto vytvářet grafické uživatelské rozhraní by bylo velmi pracné. Proto nad Xlib existují nadstavby, které vytváření GUI významně ulehčují. Umožňují např. snadno vytvářet a používat rovnou celé prvky GUI jako jsou tlačítka, menu a podobně i s tím, že se třeba tlačítko automaticky vysvítí když je nad ním kurzor a podobně. Kdyby tohle měl člověk dělat přímo v Xlib, tak se u toho pěkně nadře. Tyto nadstavby se nazývají grafické toolkity a existuje jich několik.

Správce oken

X Server se stará o malování oken aplikací na obrazovku, ale nestará se o jejich ovládání. Tedy jejich přesouvání, změnu velikosti a podobně. Nestará se o to ani toolkit ani programátor aplikace. V X Window Systému na to existuje specializovaný program, který se nazývá správce oken. Ten má na starost ovládání všech existujících oken. Kromě toho převzal i úlohu grafického shellu. To znamená, že má na starost i základní grafické rozhraní mezi uživatelem a operačním systémem. To znamená, že umožňuje základní ovládání počítače, jako spouštění programů a tak. Správců oken je také mnoho, šíře služeb a jejich možnosti a schopnosti jsou různorodé. Některé pak přerostl až v desktopové prostředí.

Další správce

Pro úplnost musím dodat, že samostatně existují i správce plochy. Ty mají na starost především ikonky na ploše. Umožňují i nastavit třeba obrázek na pozadí a podobně, ale to umí i X Server samotný. Je zvláštní, že správce oken neposkytují správu plochy samy. Vzhledem k tomu co všechno mnohdy dělají to je už maličkost. Nic jim v tom nebrání, ale tak nějak se to vyvrbilo, a správa plochy zůstává na jiném programu. Není povinný, lze se docela dobře obejít i bez něj. Já osobně třeba ikonky na ploše nepoužívám a nijak mi nechybí. Ani ve Windows.

Pak také existuje správce přihlášení. Ten má na starost grafické přihlášení uživatele k počítači. Také není povinný, je potřeba jen tehdy, když se chcete k počítači přihlašovat v grafickém režimu jako ve Windows. Je možné se přihlásit i v textovém režimu a grafické rozhraní spustit až v případě potřeby. Pak už jste samozřejmě přihlášeni a správce přihlášení není potřeba. Kromě toho správce přihlášení může (opět jich existuje víc) nabízet třeba výběr desktopového prostředí, které se má spustit, nebo jazyk, ve kterém má s vámi operační systém komunikovat nebo restartovat či vypnout počítač.

Desktopová prostředí

Již několikrát jsem se zmínil o desktopovém prostředí. To je prakticky ucelená snůška výše uvedených částí grafického rozhraní, tedy nějaký X Server, nějaký grafický toolkit, nějaký správce oken a plochy. To je základ. K tomu ještě obvykle obsahují různé podpůrné knihovny pro psaní grafických aplikací, které třeba usnadňují výměnu dat mezi aplikacemi, nějaký komponentní systém, různé konfigurační nástroje a řadu základních i pokročilých aplikací, od spávce souborů až po kancelářský balík (podle komplexnosti desktopového prostředí), které dohromady vytváří ucelené a jednotné pracovní prostředí pro uživatele.

I desktopových prostředí existuje víc. Opět je možno používat aplikace napsané pro různá desktopová prostředí současně (a mnoho aplikací není napsáno pro žádné desktopové prostředí). Není to žádný problém, jen je nutno mít tato desktopová

prostředí nainstalována. A také je nutno počítat se zvýšenými paměťovými nároky, protože v paměti počítače bude načteno víc různých knihoven.

XFree86

Jak již bylo řečeno, implementace tohoto standardu pro Linux se nazývá XFree86 a dnes je aktuální verze 4.4. Verze 4 je velmi odlišná od verze 3, s kterou je možno se ještě občas setkat. Verzi 2 nepamatuji.

Fonty

Pochopit jak to je s fonty v Linuxu není jednoduché. Ony samy fonty nejsou jednoduché a navíc v Linuxu se právě zavádí nová technologie práce s fonty, což vede k tomu, že se souběžně používá stará i nová, což situaci dále komplikuje. A to nemluvím o tom, že v Linuxu přežívají stále jakési prehistorické fonty. Takže nejdřív trochu teorie.

Trocha teorie

Font je datový soubor, který nějak popisuje kresbu písma. Toho nějak existuje více způsobů, tedy existuje několik formátů fontů. Linux (XFree86) jich většinu podporuje. Existují dva základní druhy popisů. Bitmapové a vektorové. Bitmapové jsou jednoduché. V nich každé písmenko představuje bitmapový obrázek (obrázek skládající se z matice bodů). Protože se bitmapové obrázky nedají zmenšovat a zvětšovat bez ztráty kvality, je nutno mít pro každý stupeň velikosti samostatný font. Velikost mezi není možná (tedy je, ale vypadá to hodně ošklivě). Bitmapové fonty se považují za zastaralé a prakticky se nepoužívají, setkat se s nimi ale můžete. Proto se jimi nebudu nějak zvlášť zabývat.

Vektorové fonty jsou popsány vektory. Prakticky tedy je každé písmenko definováno sadou rovnic křivek. Před zobrazením vektorového fontu je potřeba ho převést na bitmapový (protože obrazovka je také matice bodů). Tento proces se nazývá rasterizace a provádí ho nějaký (program) rastrizér během vaší práce s počítačem. Výhodou je, že si můžete velikost kresby, na kterou se má font rastrovat. To jest, všechny možné velikosti jsou "skryty" v jednom souboru a neboli fontu lze plynule měnit velikost. Nevýhodou je, že to špatně funguje u malých velikostí (obrázek písmenka se skádá jen z pár pixelů na výšku i šířku). Dříve se to řešilo tak, že pro hrubozrnou obrazovku se používalo hezké bitmapové písmo a pro tisk praktické vektorové. Dnes se to řeší tchnikou nazvanou hinting (v souboru fontu jsou speciální informace, které rastrizeru radí, jak hezky rastrovat i pro malé velikosti), kterou vymyslel Microsoft s Apple. Tato technika je patentovaná, což činí Linuxu drobné potíže.

Linux má z tohoto důvodu místo standardního hintingu autohinting, který ale nefunguje tak dobře. Samotný rasterizér umí i klasický hinting. Tato vlastnost je v něm ale vypnuta. Autoři to oddůvodňují tak, že když se např. autoři distribuce Linuxu rozhodnou zaplatit licenční poplatky za příslušné patenty, mají pak možnost kvalitní hinting zapnout. Nic samozřejmě nebrání tomu, abyste si jej nezapli sami. Zapnutí se dělá ve zdrojovém kódu rastrizéru, kde musíte nastavit jednu proměnnou. To znamená, že potom musíte zdrojov kód přeložit do binární podoby. Tímto způsobem se autoři rasterizéru brání případné soudní žalobě za porušování patentů. V některých distribucích je hinting zapnutý, navím nakolik se souhlsem vlastníka patentů.

Bitmapových i vektorových fontů eistuje více formátů. Bitmapovými se zabývat

nebudu. Nejznámější vektorové jsou T1 od Adobe, které jsou nejlepší na tisk, a TTF od Microsoftu/Apple, které jsou nejlepší na obrazovku. Linux dříve podporoval jen T1, dnes už je podpora TTF standardní záležitostí.

Stejně jako s Windows se i s Linuxem dodávají nějaké volně šiřitelné fonty. Smutnou skutečností je, že většinou nejsou moc kvalitní, spíše naopak. Ale třeba ty od Adobe jsou dobré. Pokud si ale aplikace vybere nějaký špatný, vypadá to otřesně. Lepší takové fonty z Linuxu radši odstranit. Nejlepší řešení tohodle problému je do Linuxu nainstalovat fonty z Windows. A ne jen z tohoto důvodu. Pokud budete používat kancelářský balík OpenOffice.org a v něm budete chtít číst dokumenty z Wordu, je to tařka nutnost. Stejně tak na tyto fonty spoléhá mnoho tvůrců webových stránek. Takže tento krok se vyplatí v každém případě. Jestliže nemáte Windows, nevadí, Microsoft je nabízí volně ke stažení na ??adresa1? ??adresa2?. Jak je nainstalovat poradím později.

Už jsem se zmínil, že Linux přechází na nový způsob práce s fonty, a že se v současné době používají původní i nový. A ten původní byly navíc dva různé. Každý používá svou vlastní konfiguraci, takže je to pro začátečníka docela zmatek. Popíšu proto všechny způsoby. Nejdříve oba původní a poté i nový, který sám přesně neznám.

X Server

V první řadě se o fonty a jejich rasterizaci může starat samotný X Server. Konfigurace je v cfg. souboru X Serveru (nejčastěji /etc/X11/XF86Config) a je jednoduchá. V sekci ??nazev sekce? je potřeba uvést pouze seznam adresářů v kterých se mají hledat fonty. Vypadá to nějak takto:

```
??ukazka?
```

??instalace fontu?

X Font Server

Modifikace tohoto způsobu je, že se o fonty a jejich rasterizaci stará samostatný X Font Server (xfs) s kterým X Server automaticky komunikuje přes síťové rozhraní. Výhodou tohoto řešení je, že se o fonty může starat jediný počítač, který poskytuje tuto službu všem počítačům na síti. Nemusíte tak font instalovat na všechny počítače, ale na jeden, což ocení hlavně ti, kteří vědí, jak jsou profi fonty drahé. Kromě toho se tím zjednodušuje instalace fontů a třeba v RedHatu je tohle defaultní způsob. Z hlediska aplikace je tento a přdchozí způsob totožný. Aplikace požádá X Server o vykreslení textu XXX na to a to místo tím a tím fontem. Kde ten font X Server vezme a kdo ho vyrastruje je aplikaci a jejímu programátorovi jedno. Konfigurace je mírně složitější. Do cfg. souboru X Serveru se na místo seznamu adresářů uvede spojení na X Font Server. To vypadá nějak takto:

```
??ukazka?
```

Konfigurační soubor samotného X Font Server (nejčastěji ??/etc/X11/xfs/config?) je složitější. Kromě jiného obsahuje také seznam adresářů, kde se mohou nacházet fonty. A nic jiného jsem v něm nikdy měnit nemusel. Ukázku sem nedám, považuji to za zbytečné.

??Musím se přiznat, že nevím, jak se dělá na nejnižší úrovni. V RedHatu to je jednoduché. Zkopíruji font do nějakého adresáře, který je v seznamu v cfg. souboru (nový adresář je možno doplnit) a poté jako root stačí restartovat X Font Server příkazem:

```
service xfs restart
```

A X Font Server si už všechno zařídí sám.

Xft + FreeType + FontConfig

Tím jak funguje nový způsob si nejsem jist, je to komplikovanější. Využívá se při něm nového rozhraní X Serveru, které se nazývá Xft. Takže tento způsob musí podporovat sama aplikace, musí být přeložena s podporou Xft. Za druhé se používá rasterizer ne ve formě programu, ale knihovny, tj. prakticky si fonty rastruje každá aplikace sama (díky tomu je umžněno vyhlazování fontů, pro které je potřeba vědět, co je na pozadí pod písmem). Tato knihovan se jmenuje FreeType. Kromě toho se změnilo i názvosloví fontů na trochu lidštější způsob (viz. dále).

Konfigurační soubory jsou tři. Jeden základní (/etc/fonts/??.conf), jeden systémový (/etc/fonts/local.conf) a jeden uživatelský (~/.??). Základní by se zřejmě neměl měnit, k tomu je určen systémový. Ten může měnit pouze root a je to nastavení společné pro všechny uživatele. Navíc si každý uživatel může provést svoje vlastní nastavení, to je novinka, to předtím nešlo. Konfigurační soubor je v XML formátu. Lze v něm nastavit obligátní seznam adresářů, kde se mají hledat fonty a další nastavení, jako třeba náhrady fontů, které asi hned tak někdo měnit nebude.

Instalace fontu je jednoduchá. Stačí ho zkopírovat do adresáře ze seznamu v cfg. souboru a poté spustit příkaz fc-cache. Výhodou je, že to nemusí dělat jen root, každý uživatel si teď může nainstalovat fonty sám k sobě do HOME adresáře, přístup k nim bude mít ale jen on sám. Prohlížet nainstalované a instalovat nové fonty umožňuje v GUI třeba Nautilus podobným způsobem jako je tomu ve Windows.

Názvosloví fontů

Fonty a kresba písma mají mnoho vlastností. Font může být tučný, skloněný, má nějakého výrobc (písmolajnu), patří do nějaké rodiny, má nějaké kódování a podobně. Všechny tyto informace jsou uvedeny ve fontu, ale do toho člověk nevidí. Proto se v Linuxu zavedlo standardizované pojmenování fontů, které obsahuje všechny tyto informace o fontu. Výsledkem jsou dlouhé názvy fontů, které vypadají nějak takto:

```
??ukazka nazvu fontu?
```

Jak je vidět, název se skládá z hodnot vlastností, které jsou odděleny spojovníkem (většina lidí mu říká pomlčka). O jakou vlastnost se jedná se pozná z pozice. Není-li nějaké vlastnost při výběru důležitá, lze místo ní uvést hvězdičku.

To je na jednu stranu praktické, na druhou stranu nikoliv. Především se nedá čekat, že by takto své fonty pojmenovával každý jejich tvůrce, třeba Microsoft. Takže tento název fontu je jeho názvem v grafickém systému (v XFree86) a soubor fontu se může jmenovat jak chce. Proto každý adresář s fonty v Linuxu obsahuje soubor ??nazev souboru?, který obsahuje aliasy (překlady) názvu souboru na systémový název fontu.

Pozornému čtenáři už asi došlo, že tohle je stavěné ještě na staré bitmapové fonty, protože vektorové fonty nemají žádnou skutečnou velikost. Proto soubor s vektorovým fontem má rovnou několik aliasů, pro jednotlivé stupně velikosti. Tohle je ale jen z důvodu kompatibility pro staré X aplikace, které umí používat jen bitmapové fonty se systémovými názvy. Díky aliasům těmto starým aplikacím umožňujeme používat i vektrové fonty, na oko z nich děláme vlastně sadu bitmapových. Moderní aplikace se tímto nenechají omezovat a umí si od renderu nechat vygenerovat fonty s libovolnou vlikostí.

Při normální práci se s těmito názvy ani nesetkáte. Dialog pro výběr fontů je podobný

tomu z Windows. Setkat se s ním můžete například při ruční editaci nějakého konfiguračního souboru, nebo ho lze zadat na příkazovém řádku např. při spouštění xtermu jako parametr -fn (od font name), čímž můžete xtermu říct, jaký font má použít a podobně. Začátečník se s tím asi hned tak nesetká. Probírat se těmito systémovými názvy fontů můžete např. pomocí programu xfontsel, který je součástí XFree86.

Nové rozhraní pro práci s fonty, Xft, opouští tento historický systém názvosloví a zavádí nový, zjednodušený a srozumtelnější. Uvádí se jen ty vlastnosti které vás zajímají, a nejsou dány pozicí, ale jménem vlastnosti. O tomto názvosloví vím jen teoreticky, prakticky jsem se s ním nikdy nesetkal. Nechci kecat, ale zřejmě je to tím, že Xft je zpětně kompatibilní a umožňuje používat i staré názvy. ??overit v cfg souboru pro GTK2?

Toolkity GTK a Qt

Jak jsem již uvedl, toolkitů je mnoho. V Linuxu dominují dva, GTK a Qt. GTK je určeno pro jazyk C a Qt je pro jazyk C++. Slyšet jste mohli třeba i o Motifu, který se používá v komerčních UNIXech.

GTK používá desktopové prostředí GNOME a mnoho samostatných aplikací, jako je třeba známý grafický editor Gimp a mnoho dalších. GTK lze snadno používat i z interpretovaných jazyků jako Python či Perl a dokonce i PHP. Krom toho je portováno i do Windows, takže lze pomocí něj dělat programy s grafickým rozhraním funkční v Linuxu i ve Windows, viz třeba ten Gimp. GTK má dvě verze. Staré GTK1 a nové, moderní GTK2. Bývají nainstalované obě verze, protože se lze stále ještě občas setkat s aplikací která používá GTK1, ale tento toolkit je zřetelně na ústupu. Výrazným neduhem GTK1 je nepodpora Unicode a vyhlazených fontů, ale i dalších, méně zřetelných technologií, jako podpora přístupnosti (accessibility, usnadnění) pro různě handicapované uľivatele a tak. GTK2 je naopak velmi moderní a toto všechno podporuje.

Qt používá neméně známé desktopové prostředí KDE, myslím že ho používá i vývojové prostředí Kylix od Borlandu. Také je portováno pro Windows, ale tam není zadarmo a dokonce je tam hodně drahé, takže nevím o žadné aplikaci, která by toho využívala. Já osobně preferuji aplikace používající GTK a i při programování používám GTK, takže o Qt toho moc nevím, ale bez pochyby to je také kvalitní toolkit.

O zmíněných desktopových prostředích GNOME a KDE pojednává samostatná kapitola.

Nečiní žádný problém používat současně aplikace, které používají rozdílné toolkity. Stačí mít nainstalované příslušné knihovny. Proto je dobré při instalaci Linuxu nainstalovat jak GNOME tak KDE. S nimi se nainstalují i GTK a Qt. Oba toolkity mohou aplikace používat i samostatně, bez vazby na GNOME nebo KDE. U Qt se to ale moc nevyužívá, u GTK je tomu naopak a existuje hodně samostatných (ne GNOME) aplikací, používající GTK. Např. ho používají tak známé aplikace jako Mozilla (na Linuxu, webový prohlížeč), Gimp (bitmapový grafický editor), Vim (textový editor), GQView (prohlížeč obrázků), Inkscape (vektorový grafický editor), Scribus (DTP nástroj pro sazbu), gFTP (FTP klient) a řada dalších.

??doplnit o skiny a jednotny vzhled??

Správce oken

Správce oken jsou v Linuxu velmi pokročilé a nabízí řadu možností a funkcí, které jsou

pro uživatele Windows neznámé nebo přinejmenším neobvyklé. Mezi to patří například pokročilejší možnosti ovládání oken. Každé okno je možno např. nechávat trvale nahoře. Třeba kalkulačku nebo slovník nad textovým editorem, to je věc, kterou ve windows umí jen některé aplikace a kterou v nich zoufale postrádám. Ale v Linuxu to jde ještě dál, ve skutečnosti je možné okno zařadit do několika úrovní, ta normální je uprostřed, takže můžete také okno nechat naopak trvale dole. To se hodí např. u grafického editoru Gimp, okno s obrázkem necháte trvale dole, takže všechny palety s nástroji a dialogy jsou automaticky nad obrázkem, nezmizí vám pod ním. Ve Windows musí aplikace nekvalitu ovládání oken řešit programátoři aplikací a tak ty složitější aplikace jsou každá správcem svých oken sama o sobě a řeší si to po svém. V Linuxu je ovládání oken jednotné a hlavně, každému vyhovuje jiný způsob ovládání (pro uživatele je něco takového těžko představitelné) a v Linuxu si každý může vybrat co mu vyhovuje. Další z užitečných drobností je možnost zabalení okna. Zabalené okno vypadá tak, že z něj zbude jen titlebar (horní titulková lišta), k zabalení a rozbalení stačí např. dvojklik na titlebar.

Ale nejen ovládáním oken je živ člověk. Další obecně podporovanou a užitečnou věcí jsou virtuální obrazovky nebo desktopy. Existuje více koncepcí virtuálních desktopů. Mně se líbí ta nejjednodušší. Správce oken vytváří několik virtuálních obrazovek (obvykle bývají přednastavené 4, já si jich nastavuji 8), každá virtuální obrazovka zabírá celou plochu skutečné obrazovky. Najednou vidíte jen jednu virtuální obrazovku. Mezi virtuálními obrazovkami se pak můžete přepínat (např. tlačítky 1-8 v icewm nebo pomocí schématického pageru v panelu GNOME). Když spustíte program, jeho okno se otevře ve virtuální obrazovce, ve které právě jste. Když se přepnete na jinou, okno zmizí. Když se přepnete zpátky, okno se objeví.

Na každé virtuální obrazovce můžete mít otevřeno několik souvisejících oken. Na první třeba www prohlížeč, poštovní program, TP klient, icq a vůbec věci související s internetem. Na druhé textový editor, www prohlížeč, správce souborů a další věci, které souvisí s vaší web designerskou prací. Na třetí můžete mít spuštěn správce obrázků a grafický editor s mnoha otevřenými obrázky, ve kterém můžete třeba připravovat grafiku pro své stránky. Na čtvrté obrazovce můžete mít spuštěn přehrávač mp3, ekvalizér, mixér, přehrávač CD nebo co používáte při relaxaci nebo na pozadí při práci. Na páté virtuální obrazovce (Cože, že máte jen čtyři? Tak si jich nakonfigurujte víc.) můžete mít spuštěnu třeba nějakou hru, kancelářskou aplikaci, jiný grafický editor (třeba vektorový), nějaké vývojové nástroje, jestli je používáte, terminál s příkazovým řádkem, nějaké konfigurační nástroje, prostě cokoliv co potřebujete a používáte. Já sám obvykle dělám na několika věcech současně a sem tam si oddychnu tak, že se mrknu na net a nebo si dám partii šachů, zahraji chvíli Zangband nebo tak něco.

Virtuální obrazovky vám umožní se mezi jednotlivými pracovišti snadno přepínat a neztrácet se v chaosu hromady překrývajících se oken, jako je tomu ve Windows. V práci, kde také musím dělat několik věcí najednou, musím používat Windows, a když mám spuštěno deset aplikací, začínám se v tom ztrácet, a to je ve Windows každá náročnější aplikace svým vlastním správcem oken. Virtuální obrazovky v nich také citelně postrádám. Ale Longhorn je snad prý už také bude mít. Já mám v linuxu spuštěno klidně dvacet, třicet programů, které mají ještě více oken a v pohodě se mi s tím pracuje. Virtuální obrazovky umožňují jiný styl práce, který mě přijde pohodlnější, zvláště když mi počítač běží řadu týdnů bez přerušení.

Ale i když nepočítáte s tím, že byste spouštěli tolik aplikací, dá se virtuálních obrazovek s výhodou využít. Já se mezi nimi mohu přepínat horkou klávesou Ctrl-Alt-1 až 8. Když budete mít na každé obrazovce spuštěnu jednu aplikaci, můžete se mezi nimi rychle a jistě přepínat. Je to pohodlnější, než skrolování aplikacemi pomocí Alt-Tab.

K virtuálním obrazovkám pak existuje řada ovládacích možností, jako např. možnost nechat okno zobrazit na všech obrazovkách současně, přesouvání oken mezi obrazovkami a podobně. Správce oken k tomu také poskytují bohaté konfigurační možnosti. Např. si můžete zvolit, zda se mají v taskbaru (lišta spuštěných úloh) objevovat tlačítka vších aplikací, nebo jen aplikací v právě aktivní virtuální obrazovce. Záleží na vás, co vám bude vyhovovat, zřejmě podle toho, zda obvykle budete spouštět spíš hodně nebo spíš málo aplikací. A tak podobně. A pokud by se tohle někomu nelíbilo, tak může používat jen jednu virtuální obrazovku a být omezený jako ve Windows. Není na tom nic špatného, začátečníci zvyklí na styl práce ve Windows tak mnohdy z počátku činí. I tak je ale dobré je informovat o jiných možnostech.

Tohle není ani zdaleka všechno, co správce oken umožňují, je to spíš kapka v moři. Některé věci umí všechny správce oken, některé chuťovky jen některé. Rozdíly jsou mezi nimi docela velké, protože existuje několik různých konceptů a přístupů k tomu, jak základnímu uživatelskému rozhraní má vlastně fungovat a vypadat. Nelze říci který je nejlepší, každému vyhovuje něco jiného. Není tedy divu, že mezi linuxáky pak panují spory, který správce oken je nejlepší. Je to podobné jako spory Sparta-Slávie, Amiga-Atari, Linux-Windows a podobně, jen jsou vedeny v přátelském duchu. Začátečníci používají to, co se jim tam nainstaluje, zkušení uživatelé si vybírají. Není v mých silách popsat všechny správce oken, ostatně, neumím je všechny ani vyjmenovat. Proto se budu věnovat jen jednomu, tomu který já osobně považuji za nejlepší a o kterém si myslím, že by mohl vyhovovat i uživatelům zvyklým na Windows a který dobře znám. Jmenuje se Icewm ??overit velikost pismen v nazvu? a jeho popis se nalézá sekci programy.

Desktopová prostředí

Zde bude popsano co to jsou desktopova prostredi a stručný popis těch nejobvyklejsich (GNOME, KDE, XFCE).

Programy

Tato sekce se věnuje popisu různých programů, které mi přijdou užitečné a nebo zajímavé.

Správce souborů

Obsah

Úvod

Správce souborů je program, který uživateli umožňuje řádit mezi svými soubory na disku. Lepší správci pak umožňují pracovat jednoduše i s archivy, soubory na FTP serveru a podobně. Ve Windows je takovým programem Total Commander (dříve

pojmenovaný Win Commander, než to Microsoft zakázal). Total Commander je tzv. killer aplication (zabijácká aplikace). Je tak dobrý, že ostatní programy tohoto typu živoří na okraji zájmu uživatelů. Přitom mně se dobrým začal jevit až teprve nedávno, pro mně žádný skutečně dobrý neexistoval. Se slzou oku jsem vzpomínal na vynikající DOS Navigátor, který ale skončil s érou DOSu, nepodporoval dlouhé názvy souborů zavedené ve Windows, byl jen pro textový režim a podobně. Jeho žezlo ve Windows tedy převzal Total Commander, ale tak dobrý jako DOS navigátor ještě dlouho nebyl. Ale po cca šesti letech dalšího vývoje se naučil většinu z toho co uměl DOS navigátor. Ale ne vše, třeba nemá propracovaný systém maker, nebo editaci souborů se schopností zvýrazněné syntaxe. Přesto je Total Commander výborný a samozřejmě zas umí věci, které neuměl DOS Navigátor.

Tímto úvodem jsem chtěl jen ukázat, že udělat správce souborů, který vyhoví všem požadavkům všech lidí je mimořádně náročný úkol a stojí za ním léta vývoje. Total Comander se dodnes učí nové a nové věci a celkově vzato, za správci souborů je hodně přes deset let vývoje. Však tyto programy jsou také základní uživatelská rozhraní, které umožňují s počítači pracovat. Proto jsou na ně kladeny mimořádně vysoké nároky na schopnosti, uživatelskou přívětivost a efektivitu ovládání.

Existují tři základní koncepce správců souborů. Tradiční dvoupanelový, vycházející z legendárního Norton Commanderu. Budu jej označovat jako typ T2. V obou panelech je obsah adresářů, tedy seznam souborů, obvykle včetně jejich atributů, co řádek, to jeden soubor. Druhou koncepci reprezentuje Průzkumník z Windows, také má dva panely. V jednom panelu je strom adresářů, v druhém obsah adresáře. Režim zobrazení je volitelný, soubory zde mohou být jako ikonky i jako řádkový seznam. Budu jej označovat jako typ T1.5. Poslední koncepce je jednoduché okno s obsahem adresáře. I zde je možno zvolit režim zobrazení jako u T1.5. Na levé straně se může vyskytovat pruh s různými informacemi, kontextovým menu nebo miniaplikacemi. Tento budu označovat jako typ T1. Osobně preferuji T2, myslím že ostatní jsou na práci se soubory nevhodné a mají nepohodlné ovládání. Hodí se akorát k nalezení a spuštění nějakého souboru.

V Linuxu lze samozřejmě se soubory pracovat z příkazové řádky, ale to je pro interaktivní práci nepohodlné. Možnosti příkazové řádky oceníme až při rozsáhlých hromadných pracích a nebo při automatizování nějaké opakující se činnosti. Není proto divu, že i pro Linux jsou k dispozici správci souborů. Se smutkem v srdci musím prozradit, že žádný nedosahuje úrovně Total Commanderu. V Linuxu žádná taková killer application není. Existuje jich několik v celku dobrých, jenž v něčem vynikají, třeba i nad tím Total Commanderem, ale také mají něco, co jej degraduje. Totall Commander je vlastně dobrý hlavně tím, že nemá žádnou výraznější slabinu.

Poznámka: Souborový manažer je základní aplikace, protože jich je několik dobrých a každý má oblíbené něco jiného, lze kolem nich vést náboženské války. Proto vše co zde o nich budu říkat je jen můj osobní názor, jiní lidé se na to mohou dívat úplně jinak.

Zpět na obsah Souborových manažerů

Midnight Commander (mc)

Midnight Commander

Asi nejlepší správce souborů v Linuxu. Je je vidět, je to program typu T2. Umí v podstatě všechno co se od takového programu očekává. Jeho výraznou charakteristikou je, že to je konzolová aplikace (s textovým rozhraním). To je výhoda i nevýhoda. Výhoda je, že funguje i na konzoli, že ke svému chodu nepotřebuje grafické rozhraní, to pro mnohé může být rozhodující. Nevýhoda je, že díky tomu má horší ovládání, sice lze částečně ovládat myší, ale věci jako drag&drop nepodporuje.

Ale mc má horší ovládání vůbec, a to už nelze svádět na textové rozhraní. Jeho tvůrci na efektivitu ovládání prostě moc nemysleli. Pro mnohé akce nemá horké klávesy a tak i pro tak základní věci, jako je změna třídění souborů podle nějakého kriteria musíte do menu, a to ke všemu až do menu druhé úrovně. Stejně tak zobrazování/skrývání skrytých souborů lze dělat jen skrze menu konfigurace, které je také až na druhé úrovní (musíte se k němu proklikat, vyvolat přímo nejde) a toto nastavení pak platí pro oba panely, nelze v jednom panu je mít skryté soubory zobrazené a v druhém ne. A tak by se dalo pokračovat. Ale pomalinku se v ovládání zlepšuje, poslední verze už dokonce má horkou klávesu pro zobrazení aktuálního adresáře v sousedním panelu. FTP adresu nemusíte pokaždé psát, lze dát do záložek a tak podobně.

To je jediná slabina Midnight Commanderu, mít lepší ovládání, je srovnatelný s Total Commanderem a v lecčems by ho i předčil. Mluví plně a plynně česky, což je pro mnohé důležité. Umí pracovat s archivy, dokonce i s rpm balíčky, umí FTP a dokonce i SCP (Aspoň doufám, v menu je to nazváno Secure Shell a chová se to úplně stejně jako FTP, ale funguje to přes šifrovaný bezpečný SSH. Nejsem odborník na SSH, možná má SSH i jiné možnosti kopírování souborů mezi počítači než SCP. Nechci uvádět zavádějící informace.). Součástí mc je i příkazová řádka s možností zobrazení terminálu, který je trvale na pozadí a přečíst si tak výstup příkazu.

Midnight Commander má výborné uživatelsky konfigurovatelné menu, z kterého můžete spouštět různé příkazy. Těmto pak lze předávat např. právě označené soubory a podobně. Nemusí jít jen o jednoduchý příkaz, ale i složitější skript, s podmínkami a pod. se syntaxí známého bashe. Toto menu už je předkonfigurované, jsou tam nabídky pro komprimaci souborů a adresářů a podobně. Podobným způsobem lze nadefinovat i akce Prohlédnout, Editovat a Otevřít k jednotlivým typům souborů. Většina známých je předkonfigurovaná. Díky podmínkám také lze spouštět různé programy (třeba hudební přehrávač) podle toho zda jste v grafickém nebo textovém režimu a podobně, je to velmi flexibilní.

Součástí mc je i kvalitní textový editor mcedit. Kdysi dávno to byla záchrana nových uživatelů Linuxu, protože pro Linux existovali jen dva editory, VI(M) a EMACS. Oba s tak neobvyklým ovládáním, že je začátečník nebyl schopen vůbec používat. Editor z Midnight Commanderu, s klasickým DOSovým ovládáním byl požehnáním z nebe. Jako

bývalý DOSista a nadšený uživatel VIMu to neumím příliš dobře posoudit, ale obávám se, že stávajícím uživatelům Windows už i tento editor přijde přinejmenším neobvyklý. Nevadí, pro grafické rozhraní v současné době existuje řada textových editorů, které jim budou vyhovovat. Mezi věci které mcedit ovládá patří i zvýrazněná syntaxe pro některé jazyky, třeba html a automatické odsazování. Mnozí nepotřebují víc.

- \+ archivy a RPM jako adresáře
- \+ nenáročný, rychlý
- \+ podpora FTP a SSH
- \+ textové rozhraní
- \+ plnohodnotný shell
- \+ undelete na ext2
- \+ textové cfg. soubory
- \- nepohodlné ovládání
- \- textové rozhraní
- \- konfigurace menu a typů souborů možná jen v cfg. souborech

Zpět na obsah Souborových manažerů

Alternativy už nejsou tak dobré. Před rokem (září 2002) jsem si stáhnul všechny správce souborů pro Linux, které jsem našel. Většinou to nebylo nic moc. Největší potíž dělá ovládání kurzoru z klávesnice. Grafické toolkity pro Linux (GTK, Qt a pod.) tohle neumí dobře a žádný programátor, který by si na to udělal vlastní komponentu.

Worker

obrázek: Worker

Jeden si ale asi udělal vlastní celý toolkit. Tím by se dal vysvětlit ošklivý vzhled Workeru a výborná ovladatelnost i z klávesnice. Možná že by stačila drobná změna barev, aby nevypadal tak humpolácky. Jeho konfigurovatelnost je dobrá, jak co do snadnosti, tak co do možností. Neovládá vyšší funkce souborových manažerů, nemá příkazový řádek spojený s shellem (příkazy ale umožňuje spouštět), neumí pracovat s archivy jako s adresáři (ale jinak ano) ani s FTP. To je jeho největší slabina, ale nalezl jsem elegantní řešení.

Minight commander lze spustit tak, že hned po spuštění se v jeho panelech zobrazuje obsah zadaných adresářů nebo i obsahy archivů či rpm balíčků. A Worker lze snadno nakonfigurovat tak, že při poklepání na archiv spustí mc tak, že v jednom panelu bude obsah toho archivu a v druhém obsah stejného adresáře, jako ve Workeru. Interaktivní práci s archivy tak budu pohodlně dělat přes mc (stejně to lze zařídit i s ftp, ale na to bude myslím vhodnější použít specializovaného FTP klienta), a ostatní běžné záležitosti už worker zvládne komfortněji sám. Lze se na to dívat tak, že mc funguje jako externí nástroj na práci s archivy. Co na tom, že je to zatraceně dobrý nástroj, až

by jeden řekl, že to je správce souborů. Důležité je, že ovládání zůstává pohodlné.

Worker není úplně klasický souborový manažer ala ?? Norton Commander. Od této koncepce se, jak je vidět, docela odlišuje. Ze stránky autora vyplývá, že tuto koncepci zřejmě převzal z oblíbeného správce souborů na amize. Zachovány zůstaly levý a pravý panel, tedy ta hlavní část. Horní menu zmizelo, ano, takové to menu nahoře co má každý GUI program (a dokonce i mc) worker nemá. Příkazový řádek je také fuč, ale zato spodní řada tlačítek se pěkně rozrostla. Místo jednoho řádku s tlačítky je zde pole tlačítek.

Je plně konfigurovatelné, defaultně má 8 sloupců a 6 řádků. To je 48 tlačítek. Tyto jsou dvouvrstvé, nebo dvoufunkční (jedním tlačítkem myši vyvoláte jednu akci, druhým druhou). To je vlastně 96 tlačítek (ne všechny tlačítka jsou ale obsazená, některá jsou prázdná úplně a některá jsou jen jednofunkční (pozná se podle oslího růžku)). A to není všechno, to celé je jen jedna sada, jednoduchým kliknutím se můžete přepínat mezi čtyřmi sadami (defaultně, lze je smazat i přidat další). To je celkem 192/384 tlačítek. Chybějící horní menu to plně nahradí a ač je to nezvyklé, používá se to lépe, protože se vám to pořád nezavírá jako menu.

Kupodivu je to i přehledné, funkce jsou rozmístěny logicky. První sada jsou běžné funkce, kopírování/přesun/mazání souboru, jejich označování, lze zde také připojit a odpojit CD ROM a disketu a podobně. Další sada je vyhrazena pro práci s různými druhy archivů (gzip, bz2, zip, rar), další má na starost různé konverze souborů (DOS/UNIX konce řádků, různé formáty obrázků, wav/mp3, ...) a poslední je pro nějaké programátorské záležitosti.

Na Worker je možno se dívat jako na jednoduchou a dobře ovladatelnou grafickou nadstavbu nad příkazovým řádkem. Mnoho jeho funkcí volaných přes tlačítka jsou příkazy příkazového řádku. Pro volání těchto příkazů je velmi dobře vybaven. Příkaz umí vykonávat na

Vyloženě mě nadchlo, že worker umí, stejně jako kdysi DOS Navigátor, vypočíst velikost obsahů adresářů a zobrazit to přímo v seznamu a pak podle této velikosti seznam seřadit, to jsem citelně postrádal. V Linuxu jsem na to měl aspoň příkaz pro příkazový řádek, ve Windows nic. Worker si navíc ty velikosti pamatuje a nepřepočítává je při každé změně adresáře, což je časově náročné. Takže při úklidu disku je to velmi dobrá pomůcka.

Nezvyklá je práce s myší v panelech. Levé tlačítko přesouvá kurzor a zároveň položku označí/odznačí, prostřední jen přesouvá kurzor, pravé nedělá nic. Hromadné označování pomocí Shift nefunguje, ale stačí stisknout tlačítko myši a přetáhnout kurzor přes soubory které chceme označit, na označení všech pak máme po ruce tlačítko. Jednotlivé označení přes Ctrl není potřeba, to se dělá defaultně při stisku levého tlačítka. Není to nijak omezující, ve skutečnosti se mi to zdá dokonce pohodlnější, protože si můžete jednoduše označit několik skupinek souborů, u klasického ovládání uděláte jednu a ostatní musíte pracně naklikat přes ctrl. Jenže zvyk je železná košile, takže to alespoň z počátku vadí. Uvítal bych přinejmenším, kdyby se pravé tlačítko chovalo jako levé a levé jako prostřední.

Dále je nespolehlivá klávesa F4, např. spuštění editoru na nějaký soubor. Někdy je potřeba tuto klávesu stisknout dvakrát, ale to možná bude mou špatnou překonfigurací této akce. To jsou zatím jediné negativní zkušenosti z krátkodobého soužití s ním.

- pohodlné ovládání
- nenáročný, rychlý
- dobře konfigurovatelný
- pracuje s velikostí adresářů

- ✚ šířky sloupců se automaticky přizpůsobují
- ✚ panel lze přepnout do režimu zobrazovaní obrázků
- ➖ na ovládání je třeba si trochu zvyknout
- ➖ nepodporuje archivy a rpm jako adresáře

Zpět na obsah Souborových manažerů

Gentoo

obrázek: gentoo

Zaujal mě především správce souborů Gentoo (neplést se stejnojmennou distribucí Linuxu). Je postaven na GTK a tak má divné ovládání kurzoru. Má fantastické možnosti konfigurace a podařilo se mi jej nakonfigurovat tak, že i ta práce s kurzorem je jakž takž. Jedná se o jednoduššího správce souborů, neumí pracovat s archivy jako s adresáři, FTP a tak podobně. Je velice rychlý, spouští se okamžitě, ani nestihnu dát prsty z horkých kláves přes které ho spouštím a už je k dispozici.

Jak již jsem uvedl, možnosti jeho konfigurace jsou fantastické a promyšlené. Třeba nastavení typu souborů. Normálně ke každému typu souboru (příponě) můžete nadefinovat, co se má stát když soubor chcete vidět (F3), editovat (F4) nebo otevřít (Enter). U těch lepších pak i kontextové menu s dalšími akcemi. Každý typ se nastavuje zvlášť, což je pruda, zvlášť když chcete třeba u dvaceti typů chcete změnit textový editor v kterém se mají otevírat. Gentoo na to jde chytřeji. Nadefinujete si takzvané obecné typy, text, obrázek, hudba a tak podobně včetně zmíněných akcí a kontextového menu. Přípony souborů se pak jen přiřadí k těmto obecným typům. Změny můžete tak provádět jednoduše hromadně, stačí změnit obecný typ. Ale to není všechnu. Tyto obecné typy mají hierarchickou dědičnou strukturu. To znamená, že z obecného typu text můžete odvodit specializované typy html a python. Těm předefinujete akci otevření, html budete otevírat ve www prohlížeči, python v interpretru python, zbytek konfigurovat nemusíte, převezme se z nadřazeného typu text. Můžete mít libovolně složitou strukturu, třeba můžete mezi text a html s py vložit ještě úroveň jazyky. Této úrovni pak můžete dát do kontextového menu položku pro převod jazyku do html se zachováním zvýrazněné syntaxe a tuto položku pak budou mít hned všechny jazyky. Takto si můžete připravit velmi příjemné pracovní prostředí a bez námahy, kterou byste tomu museli věnovat jinde.

Zpět na obsah Souborových manažerů

Endeavour Mark II

obrázek: Endeavour Mark II

Dále mě zaujal Endeavour Mark II. Už jen tím jménem. Je to souborový manažer typu průzkumník. Je postavený na GTK a tak vypadá dobře. Protože mi tento typ nikdy nepřirostl k srdci, nedovedu posoudit zda je dobrý a funguje jak by se od něj čekalo. Z mého laického pohledu ano.

Zpět na obsah Souborových manažerů

Gnome Commander

obrázek: Gnome Commander

Zpět na obsah Souborových manažerů

Nautilus

obrázek: Nautilus

Nautilus je defaultní správce souborů desktopového prostředí GNOME, jehož je součástí. Jako správce souborů nijak nevyniká, ale uživatelům Windows by se mohl líbit, je podobný správci souborů (explorer) z Windows.

Jeho výhodou/nevýhodou je integrace do GNOME. Pokud preferujete GNOME prostředí nebo GNOME aplikace, bude vám užitečným pomocníkem. Správa souborů je jen jeden z jeho úkolů. Dále má na starost třeba plochu (ikonky a pozadí, včetně nastavení). Já osobně GNOME jako takové nepoužívám, dávám přednost jednoduchému, nenáročnému a rychlému správci oken icewm, ale k němu si právě spouštím nautilus, aby se mi staral o plochu.

K tomu nautilus umožňuje přístup k osobnímu i některému systémovému GUI nastavení, vypalování souborů (nezkoušel jsem), ftp a podobně. U souborů s obrázky umí dělat ikonky jako náhledy těchto obrázků a dokonce má speciální režim pro prohlížení adresářů s obrázky nebo hudebními soubory. Občas bývá trochu nestabilní, hlavně ve speciálním režimu prohlížení hudebních souborů, který je možná právě z tohoto důvodu defaultně vypnutý. Nautilus ze prozkoumání určitě stojí, je to jednoduchý pohledný multimediální správce souborů integrovný do GNOME, jeho největší nevýhodou je jeho velká systémová náročnost. Nedoporučuji používat na počítačích s procesorem pod 1 GHz nebo s nedostatkem paměti RAM.

Screenshot byl dělán později na počítači s jiným skinem, proto vypadá trochu jinak, než předchozí správci souborů.

Zpět na obsah Souborových manažerů

Správce oken Icewm

Icewm bohužel nepatří mezi nejrozšířenější správce oken. Třeba do RedHatu je potřeba doinstalovat, Mandrake ho už obsahuje, ale není v něm nějak úžasně nakonfigurovaný (to jest vůbec). Přitom konfigurovatelnost Icewm je jedna z jeho výhod. Umožňuje uživateli aby si ho solidně přizpůsobil svým potřebám. Používá jednoduché konfigurační soubory, které lze snadno ručně upravovat, zálohovat a kopírovat. Existují k němu také grafické konfigurační nástroje (nejsou součástí Icewm, ale dají se doinstalovat).

Jeho další výhodou je možnost efektivního a kompletního ovládání z klávesnice. Kvůli tomu jsem ho začal používat na notebooku a jeho ostatní schopnosti mě přesvědčili natolik, že zakrátko putoval i na stolní počítač. Mnozí určitě i ocení rychlost a nízké paměťové nároky. Jestli pamětí šetřit nemusíte, pak zase oceníte, že je možné měnit jeho vzhled pomocí skinu. Některá témata vypadají opravdu hezky. A jistě se najde i pár těch, které potěší, že je lokalizován do češtiny.

Užitečné jsou i jeho některé speciální schopnosti. Například umožňuje spouštět automaticky okna vybraných aplikací se specifickým nastavením. Třeba vždy na té a té virtuální obrazovce, ať je aktuální jakákoliv, nebo na všech obrazovkách najednou, nebo maximalizované či naopak minimalizované. Bez titlebaru nebo tlačítka v taskbaru a tak podobně. Někdy se to skutečně hodí a člověk má pak radost, že to Icewm umí zařídit.

Pro mě osobně je také velmi důležitá a užitečná vlastnost možnost nastavit spouštění často používaných aplikací přes horké klávesy. Pokaždé když zkouším nebo používám jiného správce oken, tak se bez téhle schopnosti cítím jako ryba bez vody.

Napadá mě jediná věc, která se lidem zvyklým na windows nebude líbit. A to je menu s aplikacemi. Ne že by menu samotné bylo špatné, líbit se nebude tvorba jeho

obsahu, člověk z Windows je zvyklý, že si do něj nainstalované aplikace přidávají položky automaticky. V Linuxu tvorba menu nebyla standardizovaná, tj. každý správce oken si to dělá po svém a to se pak dá těžko zautomatizovat. V Icewm je menu definováno konfiguračním souborem s jednoduchou, ale speciální syntaxí. Automatická úprava takového souboru je ještě složitější.

To je samozřejmě špatné a protože se Linux začal orientovat na běžné uživatele, muselo se to nějak začít řešit. Desktopová prostředí GNOME a KDE zavedly jednoduchý standard, jak automaticky přidávat položky do menu při instalaci programu. Každá položka menu představuje samostatný soubor, který se umisťuje do speciálního adresáře v Linuxu. Obsahem tohoto souboru je cesta k aplikaci, která se má spustit a názvy položky v jednotlivých jazycích (kódování souboru je Unicode). To je chytré řešení, lepší než má Windows, protože Linux umožňuje snadno přepínat jazyk kterým s vámi komunikuje a s přepnutím jazyka se tak změní i jazyk menu.

Icewm tuto technologii implementuje tak na půl. Pořád používá konfigurační soubor pro menu, ale nyní se už může skládat z více souborů. Kromě toho je k němu dodáván prográmek, který se spouští automaticky a který podle souborů toho standardního menu vygeneruje cfg. soubor menu pro Icewm. V menu Icewm se pak toto standardní menu objevuje jako podmenu GNOME a podmenu KDE (každé desktopové prostředí má vlastní rozvržení, to je vcelku logické, jedno preferuje a dává do popředí GNOME aplikace a druhé ty KDE). Má to ale jeden háček, ten prográmek který generuje menu pro Icewm nebere ohled na nastavený jazyk a menu vygeneruje vždy v angličtině. To je hloupé, ale oprava toho prográmku pro generování menu by neměla být zas tak těžká, snad to někdo udělá.

Já to nebudu, protože mě osobně to netrápí. Já menu v Icewm vůbec nepoužívám, pouľívám jen toolbar Icewm (panel nástrojů) na kterem mam ikonky programů, které používám, to je rychlejší. Nejčastěji používané programz pak spouštím dokonce přes horké klávesy. Toolbar a programy na něm si upravuji ručně podle svých potřeb. Jeho konfigurační soubor používá tu samou syntaxi, jako menu. Není to sloľité. Přidání programu znamená do cfg. souboru přidat řádek:

```
prog "název" ikonka program
```

Kde prog je klíčové slovo, že se jedná o odkaz na program, název je název položky v menu nebo toolbaru, ikonka je název ikonky (obrázku) bez přípony, který se má zobrazit u položky a program je příkaz, kterým se program spouští. Lze za něj přidat i případné parametry. To je všechno, takže nějaké příklady:

```
prog mozilla         mozilla  mozilla
prog "X Terminál"    term     xterm -geometry 100x50


menu "Icewm Config"  folder {
  prog menu          edit     gedit /home/wraith/.icewm/menu
  prog toolbar       edit     gedit /home/wraith/.icewm/toolbar
  prog nastavení     edit     gedit /home/wraith/.icewm/preferences
}
```

Není to tak pohodlné jako automatické přidání položky, ale ani nijak extrémně náročné a máte to plně pod kontrolou. Konec ukázky předvádí přidání celého podmenu s položkami, které spouští textový editor gedit s jednotlivými konfiguračními soubory icewm.

No a to je to nejhorší a jediné špatné, co mě na Icewm napadá a mě osobně i tohle vyhovuje, protože cfg. soubory lze zálohovat nebo třeba zkopírovat kamarádovi, kterému instaluji Linux. Chápu ale, že někomu to může vadit a tak na to upozorňuji.

Kromě spouštění programů přes menu, toolbar nebo horké klávesy lze také programy spouštět přes příkazový řádek, který je přímo v toolbaru.

Gkrellm – informační a ovládací panel

My, staří linuxoví psi, jsme si už dávno zvykli na to, že v Linuxu se musí souborové systému mountovat a to klidně i na příkazovém řádku a ani nám to nepřijde. Linuxový začátečník s tím má větší problémy. Těžko se mu smiřuje už s tím, že to musí vůbec dělat, takže aspoň hledá co nejsnazší cestu. V moderních desktopových prostředích jako GNOME nebo KDE to má docela jednoduché, může si to snadno naklikat. Ale obě tyto prostředí jsou hodně náročná na výkon počítače, nic pro slabší kousky, takže co s tím?

Jedno z možných řešení je Gkrellm, který vidíte nalevo. Je to šikovný informační a ovládací panel. Jeho informační úloha je stěžejní, to je jeho pravý nebo spíš původní účel. Ale jeho monitory lze klidně vypnout a nechat tam jen tu ovládací část.

Horní dvě třetiny zabírají monitory, které nás informují o zatížení procesoru, loadu, vytížení disků, síťových rozhraní a paměti. Skrze ně má člověk pěkný přehled co se mu v počítači zrovna děje, ale touto funkcionalitou Gkrellmu, možnostmi nastavení, alarmů a podobně se nebudu zabývat. Momentálně mě zajímá ta část pod tím.

To je vybraný seznam souborových systémů. Objevit se zde může cokoliv, co máme nakonfigurováno ve fstabu (viz. sekce konfigurace, kapitola mountování). U souborovbých systémů které jsou trvale připojené, jako např. root souborový systém, zde máme informaci o obsazeném prostoru (lze nakonfigurovat) nebo názvu zařízení (přepíná se kliknutím na číslo nebo název). To není ještě nic moc zajímavého.

Zajímavé to začíná být u souborových systémů, které se připojují na požádání. Na obrázku je to cdrom a čtečka. U nich je malé tlačítko, pomocí kterého lze souborový systém připojit a zase odpojit. Podle zaškrtnutí je navíc vidět, co máme zrovna připojeno (připojení) se zobrazí, i když ho provedeme mimo gkrellm, třeba z přík. řádku.

U cdromu se dokonce po najetí kurzoru nad tlačítko toto změní na dvě, jedno slouží k připojení/odpojení, druhé k vysunutí média.

Konfigurace Gkrellmu je snadná, stačí napsat jmenovku, vybrat adresář ke kterému se má souborový systém připojovat, zatrhnout, že údaje se mají brát z /etc/fstab a u cd ještě dát na vědomí, že médium je možné vysunout. Obrázek to asi popíše lépe.

ukázka konfigurace

A to je vše. Pod souborovými systémy je ukazatel obsahu poštovní schránky (lze nakonfigurovat místní i vzdálenou, dostupnou přes pop3 protokol). Pro gkrellm existuje také mnoho pluginů, na obrázku jsou vidět dva, jeden má na starost ovládání hlasitosti a druhý xmms (něco jako winamp ve Windows). Ale to jen pro pořádek.

Myslím že Gkrellm dovede začátečníkovi zpříjemnit práci v Linuxu a proto by neměl být

přehlédnut. Gkrellm se může chovat jako panel, to znamená, že např. při maximalizaci nějakého okna nedojde k jeho překrytí a je tak pořád vidět a rychle dostupný. Kromě toho lze také nastavit, zda má být Gkrellm na všeh plochách nebo jen na jedné. Používám ho s icewm a mám s ním jen dobré zkušenosti.

Gkrellm je skinovatelný, ale skiny pro něj se mi moc nelíbí. To na obrázku mi přijde jako jediný použitelný skin (nazývá se Monkey Lovers) a i ten jsem si sám trochu upravil.

CLI

CLI je zkratka od Command Line Interface, což znamená rozhraní příkazové řádky. Tato sekce se mu bude podrobně věnovat.

V preambuli tohoto dokumentu jsem se rozčiloval, a vysvětloval, jak je nevhodné začátečníka seznamovat s Linuxem skrze příkazovou řádku. To ale neznamená že bych ji zatracoval, naopak. Příkazová řádka je mocný nástroj a rozhodně by neměla být ani ignorována.

Navíc hodně příkladů budu ukazovat právě na ní. Mám k tomu dobré důvody. Za prvé příkazy jsou univerzálnější, lze je používat na většině (ne-li všech) linuxových distribucích, kdežto grafické utilitky se distribuce od distribuce různí. Za druhé je to jen text a ten se do stránky snadněji vkládá a zabírá mnohem méně místa než obrázky.

Proto nemohu a ani nehodlám příkazovou řádku zanedbat a věnoval jsem ji hned rovnou dvě sekce. Tato sekce ji popisuje obecně a je určena lidem, kteří vůbec nevědí o co jde. Ta druhá se pak bude věnovat jednotlivým příkazům.

Příkazová řádka

Příkazová řádka slouží k interaktivnímu zadávání příkazů. Napíšete příkaz, odešlete a ten se vykoná. Převážná většina příkazů není nic jiného než jméno programu. To znamená, že z příkazové řádky můžete spouštět libovolné programy. I ty pro grafické rozhraní. Takže můžete zadat příkaz:

```
editor
```

A spustí se program editor. To je to nejjednodušší použití. Spuštěným programům lze předávat parametry. Parametry jsou různá slova, písmenka a znaménka oddělená mezerami. Několik příkladů:

```
editor poznamka.txt
     editor -r poznamka.txt
     editor poznamka1.txt poznamka2.txt
```

První příklad spustí editor a otevře v něm soubor poznamka.txt.

Druhý příklad udělá to samé, ale spustí se v režimu jen pro čtení. Parametry ovlivňující chovaní programu (jako je -r) se nazývají optiony.

Třetí příklad spustí editor a otevře v něm oba dva zadané soubory.

To všechno bude ale fungovat jen tehdy, pokud to ten editor umí. Kdybychom to zkoušeli třeba s notepadem, bude fungovat jen první příklad, protože notepad neumí otevřít víc souborů najednou a ani se neumí spustit v režimu jen pro čtení. Jiný editor

zas může na parametr -r reagovat jinak. Jaké parametry program rozeznává a jak na ně reaguje je jeho individuální záležitost. Vždy je potřeba se poradit s dokumentací dotyčného programu.

Přesto však lze uvést nějaké obecné konvence. V Linuxu je mnoho jeho příkazů součástí projektu GNU. Tam jsou stanoveny tyto pravidla.

- Optiony se píší hned za příkaz, soubory kterých se to eventuálně týká se píší nakonec. Tedy editor -r soubor a ne editor soubor -r.
- Optiony jsou dlouhé a zkrácené. Zkrácené jsou jednopísmenové a uvozují se pomlčkou, dlouhé jsou ty ostatní a uvozují se dvěma pomlčkami. Tedy zkrácený je např. -r a jeho dlouhý ekvivalent je --read-only (přeloženo --číst-pouze). Není povinné, aby každý program měl option v dlouhé i zkrácené podobě, může mít i jen jeden z nich. Když má obojí, tak ty zkrácené se obvykle používají při interaktivním psaní na příkazový řádek, aby se člověk příliš neupsal. Když se příkaz zapisuje někam trvale, je vhodné použít delší formu, náhodným čtenářům je pak jasnější o co jde.
- Zkrácené optiony lze spojovat, tedy optiony -r -o -x lze spojit v -rox.
- Ne všechny programy se těchto konvencí drží, dokonce ani ty GNU. Například některé programy uvozují optiony znaménky + a -, kde plus znamená zapnout volbu a minus vypnout. Příkaz find je zase interpretuje ve smyslu větší nebo menší, např. jestli má hledat soubory větší nebo menší než je zadaná velikost.

Takhle obecně řečeno to platí jak pro Linux, tak i pro Windows. Nenechte se mýlit, i Windows používá příkazovou řádku. Podívejte se třeba na vlastnosti ikonky, která spouští nějaký program, uvidíte tam jednoduchou příkazovou řádku. Zrovna tak, když se podíváte do editoru asociací s příponami, ve kterém se definuje s jakým programem se má který druh souboru otevírat, zjistíte, že se to definuje na příkazové řádce i s těmi parametry. Příkazová řádka se používá stále, nebyl vynalezen lepší způsob pro spouštění programů, v OS s grafickým rozhraním je akorát více skryta. Člověk je tak ušetřen zdlouhavého psaní, ale je zas omezena flexibilita v možnostech zadávání parametrů. Je to něco za něco.

Alternativou k těmto skrytým příkazovým řádkům je **shell**. To je program, který umožňuje interaktivní zadávání a spouštění příkazů. Ve Windows se tyto programy jmenují command.com nebo cmd.exe a jsou velmi primitivní a uživatelsky nepřívětivé. Ve Windows se s prací na příkazovém řádku moc nepočítá a tak je tam nepohodlná. Linux je velmi dobře přizpůsoben pro práci s příkazovou řádkou a jeho shelly jsou luxusní nástroje, které ji ještě víc usnadňují. Jednomu z těchto shellů se věnuji v následující kapitole.

Ve Windows si můžete spustit i linuxový shell, ale moc si tím nepomůžete, protože ta přívětivost je záležitost celého systému, nejen shellu. V Linuxu jsou např. díky jiné organizaci dat všechny programy v cestě, takže stačí napsat příkaz editor a tento se spustí. Ve Windows se buďto musíte přepnout do adresáře ve kterém je editor nainstalován a nebo k němu musíte uvést kompletní cestu. Neberte to dogmaticky, výjimky samozřejmě existují. Některé systémové programy jsou i ve Windows v cestě a lze je spouštět jednoduše a v Linuxu lze program nainstalovat mimo cestu. Windows ani zdaleka nenabízí takovou paletu příkazů jako Linux (o tom budu psát ještě dál). Dále umožňuje lepší práci s příkazovým řádkem v Linuxu souborové rozhraní a jeho textová orientace. Jednoduše a prostě řečeno, linuxový příkazový řádek je o něčem jiném.

Další výhodou linuxového příkazového řádku je, že jednotlivé distribuce Linuxu (a víceméně i UNIXu) se od sebe téměř neliší a že poskytují všechny stejnou sadu (alespoň tu základní standardizovanou) příkazů. Díky tomu se můžete snadno

administrovat různé distribuce Linuxu. Když znáte jeden, znáte i ty ostatní. To se o administraci skrze GUI rozhodně říci nedá, tam to platí spíše naopak. To je i jeden z důvodů, proč většinu příkladů administrace Linuxu uvádím právě pro příkazový řádek, bude to fungovat na větším množství systémů. Další důvod je, že se snadněji řekne zadej příkaz xyz než se popisuje kde na jaké menu či ikonku kliknout a kam a co do jaké kolonky napsat.

Linuxová příkazová řádka má ale i své záludnosti. Dejte si pozor na jednu věc při spouštění GUI programů. GUI program se od ní může buď odpoutat a nebo k ní může zůstat připoután. V prvním případě se to chová stejně jako ve Windows, v druhém případě dojde k dočasnému zablokování příkazové řádky (nelze na ni zadávat další příkazy) dokud není dotyčný program ukončen. Má to své dobré důvody, které jsou ale trochu vyšší dívčí, takže je nebudu objasňovat. Jednoduše řečeno, díky tomu umí příkazová řádka Linuxu některé užitečné věci, které ta ve Windows neumí. Nepříjemným důsledkem toho je, že když násilně ukončíte takto zablokovanou příkazovou řádku (třeba zavřením okna terminálu, což je hned), tak dojde i k násilnému ukončení programu, který je k tomu příkazovému řádku připoután. Linuxoví začátečníci se tak párkrát připraví o neuložená data, než se poučí, že tohle se dělat nemá.

Normálně to není problém, protože rozumná aplikace, která to nepotřebuje se od příkazového řádku odpoutá a je to, bohužel některé aplikace nejsou rozumné, třeba grafický editor Gimp.

Spuštění GUI programů z shellu je jen takový bonus, lze je spouštět i jinak, třeba z menu, kliknutím na ikonku datového souboru a podobně. Smysl a účel shellu je jiný, umožňuje (v Linuxu) kompletní ovládání počítače, bez GUI. Kdysi dávno to dokonce byla jediná možnost. Linux se dodnes dá administrovat bez GUI. Spousta aplikací (jako různé serverové programy) nepoužívají GUI dodnes. Pro Linux dokonce existuje mnoho normálních aplikací, které se obejdou bez GUI. Namátkou webový prohlížeč, poštovní klient, icq, mp3 přehrávač, mixér, samozřejmě textový editor, dokonce i přehrávač videa a tak podobně.

Zmíněné aplikace spadali většinou do kategorie TUI. Tedy s textovým uživatelským rozhraním a kromě schopnosti fungovat v textovém režimu moc výhod nepřinášejí. Pracovat v textovém režimu může být někdy výhodné, hlavně díky mnohem nižším systémovým nárokům, třeba při vzdáleném spouštění aplikací přes modem, jinak je ale obvykle pohodlnější a přehlednější pracovat v GUI.

Pak je tu ještě jedna kategorie programů, pro příkazovou řádku ta nejdůležitější. Programy této kategorie se vyznačují tím, že nemají žádné uživatelské rozhraní kromě samotné příkazové řádky. Tedy ovládají se jen pomocí zadaných parametrů. Po spuštění něco provedou, eventuálně vypíší na obrazovku a ukončí se. O těchto programech pak neuvažujeme jako o programech, ale o příkazech. Tyto příkazy dělají obvykle jednoduchou konkrétní věc. Velká síla těchto příkazů spočívá v tom, že se dají navzájem kombinovat a také že jejich sekvence lze zapisovat do textových souborů (skriptů) a posléze je opakovaně spouštět najednou. V těchto skriptech lze používat i základní programátorské techniky jako je používání proměnných a podmínek, takže možnosti jsou opravdu bohaté. Dostáváte k dispozici neomezené množství velmi rozmanitých kostiček lega a co si z nich postavíte je už jen a jen na vás.

Pro úplnost doplním, že některé základní příkazy jsou v mnoha shellech vestavěné, takže tyto příkazy pak provádí sám shell, nespouští žádné externí programy, i když tyto mohou být také k dispozici. Je to důležité vědět hlavně kvůli nápovědě. Např. příkaz cp (kopírování souboru) existuje jako vestavěný příkaz shellu bash i jako program, každý z nich však rozeznává jiné parametry. Začátečníka určitě zmate, když

si přečte manuálovou stránku k programu cp (man cp) a pak zjistí, že příkaz cp se chová jinak. Neví, že si musí přečíst manuálovou stránku programu bash (man bash) či spíš její pasáž věnující se vestavěnému příkazu cp. Teda pokud začátečník kopíruje soubory pomocí příkazové řádky, což není moc pravděpodobné.

Terminál a konzole

Teorie

Při psaní o příkazovém řádku se jen těžko mohu vyhnout zmínce o terminálu nebo konzoli, takže bude vhodné, když vysvětlím co to je.

Kdysi dávno neexistovaly osobní počítače, místo nich byl jeden velký (třeba i na celou firmu), myslím že se nazýval mainframe (sálový počítač) a s ním mohlo pracovat několik lidí najednou. Byly totiž k němu připojeno několik monitorů a klávesnic. Jedna klávesnice a monitor tvořily celek, jakousi krabici, která se nazývala terminál. Z počátku byl terminál úplně hloupý, později získal jakousi inteligenci, ale pořád to nebyl samostatný počítač, bez mainframe nefungoval, bylo to jen rozhraní pro člověka, skrze který mohl komunikovat s velkým počítačem. Člověk stiskl klávesu X a terminál poslal tuto informaci poslal mainframe, který na ní příslušně zareagoval. Když chtěl zdělit mainframe něco člověku, řekl terminálu na řádku 5 a sloupci 8 zobraz znak X a terminál to udělal. Mainframe tedy neposílal videosignál, jen změny.

Potom přišly osobní počítače, které fungují trochu jinak. Nebyly určeny k tomu, aby na nich pracovalo několik lidí současně (Linux to ale umožňuje), jsou přece osobní. Proto bylo zbytečné, aby se k nim připojoval terminál, osobní počítač jeho funkci zvládá sám. Je v něm grafická karta ke které se monitor připojí přímo a zrovna tak se k osobnímu počítači přímo připojuje samostatná klávesnice. Funkce terminálu je pak zajištěna softwarově operačním systémem. Takto emulovaný terminál se v Linuxu nazývá konzole. S konzolí se setkáte, když vám Linux nabíhá do textového režimu.

Samotná konzole vám ale ještě k ničemu není, je to černá prázdná obrazovka, na kterou umí Linux vypisovat (barevný) text. Aby byla konzole prakticky použitelná, musí v ní běžet nějaký program, který umožní interakci. Tento program je zpravidla login, který vám umožní se přihlásit do systému a který po přihlášení místo sebe spouští nějaký shell. Ale může to být i jiný program, záleží to čistě na konfiguraci OS.

Protože konzole je softwarově emulovaný terminál, není nijak velký problém jich emulovat víc najednou. Nevím kolik jich Linux emuluje, obvykle se v něm používá 6 konzolí (spouští se 6 loginů), pokusně jsem si nastavil aby se jich spouštělo 12 a pak jsem mohl používat 12 konzolí. Mezi konzolami se můžete přepínat horkými klávesami Alt-F1, Alt-F2, Alt-F3 atd. nebo Alt-Vlevo a Alt-Vpravo.

Většina lidí ale už nepracuje s linuxem čistě v textovém režimu, to je vhodné třeba pro servery, ale ne pro domácí použití nebo v kanceláři. Zde je mnohem oblíbenější grafické prostředí. Ani v něm nejsme připraveni o příkazový řádek. Existují dvě možnosti jak se k němu dostat.

Zaprvé se můžeme přepnout na konzolu, tedy do textového režimu, stisknutím horké klávesy Ctrl-Alt-F1, Ctrl-Alt-F2 atd. Z textového režimu se samozřejmě můžeme opět přepnout zpět na grafický režim. Obvykle horkou klávesou Alt-F7. To proto, že, když se spustí grafický režim, tak ten obsadí první volnou konzoli. Obvykle je jich obsazeno 6, takže grafické rozhraní běží pak na sedmé a přepnutím se na sedmou konzoli dojde zároveň k přepnutí zpátky na grafický režim.

Za druhé je možno spustit program, který terminál emuluje. Takových programů

existuje pro Linux mnoho. Nejobvyklejší je asi xterm, který je opravdu výborný. Mimo jiné podporuje i unicode. Další jsou např. rxvt, gnome-terminal nebo konzole (není tím myšlena ta textová, ale GUI program shodného jména, vlastní emulátor terminálu desktopového prostředí KDE). Takovýchto terminálů si můžete spustit několik a mít tak k dispozici několik příkazových řádků.

V těchto terminálech lze i v GUI spouštět programy určené pro textové rozhraní, jako je správce souborů Midnight Commander nebo textový editor Vim. Narozdíl od Windows lze terminálům v Linuxu měnit libovolně velikost a linuxové programy se tomu umí přizpůsobit.

Pokud by někdo řešil otázku, zda je lepší se přepínat na textovou konzoli nebo si radši spustit emulátor terminálu, pak bych doporučil ten emulátor. Zaprvé textové rozhraní má obvykle jiné rozlišení než to grafické a změna rozlišení monitorům trvá poměrně dlouho a nedělá jim příliš dobře, za druhé programy na konzoli nedovedou komunikovat skrze schránku s programy v GUI. Na druhou stranu, emulátor terminálu je program, který si ukousne kus paměti, takže na velmi slabých hw konfiguracích se může vyplatit používání konzole. A především, Linux je mimořádně stabilní systém a téměř nepadá. To už se ale nedá říci o jeho grafickém rozhraní, to je jen stabilní a sem tam může mimořádně spadnout. S ním pak odejdou k čertu všechny GUI programy, ty na konzoli pojedou klidně dál.

Se spuštěním terminálu dojde automaticky ke spuštění nějakého shellu, takže je v něm k dispozici příkazový řádek. To je ale jen defaultní chování, pomocí parametru lze s terminálem spouštět jiný program, třeba mc, top a podobně. Po ukončení tohoto programu také dojde k ukončení terminálu. Pomocí parametrů lze ovlivňovat i další věci, pár základních věcí představím.

Praxe

Bash – základy

Shellů pro Linux existuje několik, já se budu věnovat tomu, který se jmenuje bash. Je nejčastěji používán a je přítomen ve všech mě známých distribucích Linuxu. Příkazový řádek se nemusí nijak spouštět, spouští se automaticky s terminálem. Pokud se do Linuxu přihlašujete na konzoli, máte ho k dispozici hned po přihlášení, bez něj byste nemohli počítač vůbec ovládat. Pokud se přihlašujete v grafickém režimu, tak je k dispozici po spuštění emulátoru terminálu. Běžný uživatel má spojeno terminál = příkazový řádek a ani si neuvědomuje, že to jsou vlastně dva různé programy.

Bash se dá ukončit příkazem exit, po jeho ukončení dojde na konzoli k automatickému odhlášení uživatele, v GUI dojde k zavření terminálu. Je možno také spustit bash z bashe, druhá instance nenahrazuje první, ale pracuje nad ní, takže po ukončení druhé instance se vrátíme zpět do první a teprve po ukončení té první dojde k odhlášení nebo ukončení terminálu.

Psal jsem, že linuxové shelly jsou luxusní nástroje, které práci na příkazovém řádku usnadňují. Je to pravda, ač se to na první pohled nezdá, bash je složitý program s mnoha možnostmi a jen málokdo ho zná kompletně. Já určitě ne. Seznámím vás tedy s jen s věcmi které jsem si osvojil a které mi přijdou užitečné.

Historie příkazů

Ta nejjednodušší věc je **historie příkazů**. Bash si pamatuje příkazy které odešlete, umožní vám je vyvolat, editovat a spustit znovu. Historie příkazů se dá jednoduše

procházet pomocí kláves šipka nahoru a dolu. Lze s tím dělat i mnohé další věci jako příkazy vyhledávat, přebírat jenom parametry a podobně, ale to jsem se nenaučil používat, jednoduché procházení mi docela stačí. Příkazy se pamatují i po ukončení bashe a po jeho opětovném spouštění. Ukládají se do souboru .bash_history v domácím adresáři.

Doplňování

Další zjednodušení práce umožňuje **doplňování**. Doplňuje se stisknutím klávesy tabelátor a doplňují se různé věci podle kontextu. Především název příkazu (první slovo na příkazovém řádku) nebo souboru (možno i s cestou, další slova na příkazovém řádku). Dále lze doplnit jméno uživatele a název počítače nebo nějaké systémové proměnné (každé z těchto slov začíná speciálním znakem ~, @, $).

Doplňování funguje tak, že se napíše několik počátečních písmen výrazu a stiskne se TAB. Bash si zjistí všechny možnosti a podle nich výraz doplní kam až může. Hranice je dána jedinečným výskytem doplňované části. Např. mám v adresáři dva soubory, dlouhy_nazev1.txt a dlouhy_nazev2.txt. Potom stačí napsat za nějakým příkazem třeba dl, stisknout TAB a název se doplní na dlouhy_nazev a počítač pípne. Tím nás informuje, že doplnění nebylo kompletní, ale že neví jak dál. Můžeme stisknout znovu TAB a tím se nám na obrazovku vypíší všechny možnosti, v tomto případě názvy obou souborů, vidíme tedy, že jeden pokračuje znakem 1 a druhý znakem 2, doplníme tedy třeba 2 a stiskneme znovu TAB. Název souboru se doplní až do konce. Takhle můžeme doplnit i název podadresáře, za něj dát lomítko / a pokračovat doplňováním názvu souboru v tomto podadresáři. Název nadadresáře je .. a lze také použít při doplňování.

Zástupné znaky

Další zjednodušení jsou zástupné znaky. Dejme tomu, že v adresáři máme 10 souborů které končí příponou .jpg a chceme je všechny otevřít v grafickém editoru gimp. Můžeme napsat příkaz gimp a za něj uvést všech deset názvů, doplňování nedoplňování, dá to práci. Mnohem jednodušší je napsat příkaz gimp *.jpg a je to. Hvězdička je zástupný znak, který znamená – jakýkoliv text – a bash jej pak nahradí všemi odpovídajícími názvy souborů. Je důležité pochopit, že to dělá bash, ne gimp, gimp jako parametry už dostává konkrétní názvy souborů.

Hvězdička není jediný zástupný znak, je jich víc.

*	Zastupuje libovolně dlouhý (tedy i žádný) text. **a*.txt** = a.txt, ahoj.txt, ahoj.txt.txt
?	Zastupuje jeden libovolný znak. **a?.txt** = aa.txt, ab.txt, a_.txt
[ ]	Zastupuje jeden ze znak. Lze udávat intervaly jako 0-9, A-Z nebo c-i. **a[1-3x].txt** == a1.txt, a2.txt, a3.txt, ax.txt **a[0-9][0-9].txt** = a00.txt až a99.txt
[!]	Jako [ ], ale zastupuje znaky které nejsou uvedené v závorkách. **[!A-Z]*.txt** = všechny soubory s příponou .txt, které nezačínají velkým písmenem.
~ (vlnovka)	Domácí adresář uživatele který příkaz spouští. U uživatele boxer bude ~/.bashrc pravděpodobně znamenat /home/boxer/.bashrc. Ale nemusí,

	záleží to na nastavení systému a tom kde jsou domácí adresáře umístěny, uvedený příklad je jen nejobvyklejší místo. V každém případě to však bude do domácího adresáře uživatele boxer, ať je umístěn kdekoliv. Samozřejmě, když se uživatel boxer přehlásí na jiného uživatele, třeba root pomocí příkazu su, tak pak vlnovka bude ukazovat do domácího adresáře roota.
~xyz	Domovský adresář uživatele xyz.

Zástupné znaky v Linuxu jsou trochu lepší než ve Windows, které umožňují používat jen * a ? a hvězdička nemůže být na začátku nebo uprostřed textu, jen na konci. Další rozdíl je, že Windows bere název souboru a jeho poslední příponu jako dva texty, Linux jako jeden včetně oddělovací tečky. Ve Windows se proto musí výraz jakýkoliv soubor zapsat jako *.* kdežto v Linuxu stačí samotná hvězdička. V novějších verzích Windows to může být již vylepšeno, ty ale neznám, tak to berte trochu s rezervou.

Zástupné znaky, taktéž globing, si nepleťte s regulárními výrazy, které vypadají podobně, ale mají jinou syntaxi a třeba [!A-Z]*.txt v nich znamená něco úplně jiného.

Aliasy

Další usnadnění jsou **aliasy**. Alias neboli přezdívka je zástupné jméno pro nějaký (pravděpodobně dlouhý) příkaz. Například příkazy pro připojení a odpojení diskety mohou být mount /mnt/floppy a umount /mnt/floppy. Je to pruda pořád psát takový příkaz a některé mohou být mnohem delší. Stačí ale dát příkaz alias mf="mount /mnt/floppy" a alias uf="umount /mnt/floppy" a můžeme disketu připojovat a odpojovat příkazy mf a uf.

Také tak můžeme změnit defaultní chování příkazu. Třeba příkaz ls vypisuje obsah adresáře. Černobíle, s parametrem --color to dělá barevně, což je lepší, protože tak hned poznáme, co je adresář, co spustitelný soubor, co symbolicky link, co normální a podobně. Ale je to pruda tam parametr psát pořád dokola. Stačí však dát příkaz alias ls="ls --color" a voila, ls používá barvy sám od sebe. Při používání aliasu ls samozřejmě můžeme za něj dávat další parametry, které se předají skutečnému ls.

Aliasy nám tedy usnadní práci, ale přeci jen, pořád je pruda je definovat pokaždé na začátku práce. Hodilo by se, kdyby si je bash uměl uložit. To bohužel ale neumí. Naštěstí řešení existuje, bash při svém spouštění automaticky načítá několik různých souborů, do kterých je možno ty aliasy zapsat a ty se pak definují automaticky. Tyto soubory jsou tři, každý má svůj účel a svou podstatou to jsou skripty, které bash spouští.

/etc/profile
: Načítá se pouze při přihlášení, je společný všem uživatelům. Editovat ho může pouze root. Používá se pro defaultní systémovou konfiguraci.

~/.bash_profile
: Načítá se pouze při přihlášení, každý uživatel má svůj vlastní. Používá se pro osobní systémovou konfiguraci, když ta defaultní nevyhovuje.

~/.bashrc
: Načítá se při každém spuštění bashe a každý uživatel má svůj vlastní. Tento je pro definici aliasů nejvhodnější. Používá se pro konfiguraci samotného bashe.

Tak, to by byly ty jednodušší záležitosti. Nyní bude následovat výklad o kombinování příkazů, které pro uživatele znajícího dosud pouze klikání na ikonky obtížněji pochopitelné. Přitom to je jedna ze dvou věcí, která dělá z příkazového řádku mocný nástroj. Ta druhá je skriptování, kterému se budu věnovat později.

Bash – kombinace příkazů

Roury

Už víte, že z příkazového řádku se dají spouštět libovolné programy. Následující text se bude věnovat ale jen těm, které označuji jako příkazy (až na nějaké výjimky). To jest těm, které nepoužívají žádné uživatelské rozhraní kromě samotné příkazové řádky. Zvláštní kategorií příkazů jsou tzv. filtry. Tyto jsou charakteristické tím, že načítají text ze standardního vstupu, nějak ho upraví a upravený ho pošlou na standardní výstup. Ostatní příkazy používají jen jedno z toho a nebo vůbec nic. Normálně je standardní vstup klávesnice a standardní výstup obrazovka, ale bash umožňuje standardní vstup i výstup přesměrovat.

Snadno pochopitelné je přesměrování standardního výstupu do souboru. Výsledky se nezapíší na obrazovku, ale do zadaného souboru. Obráceně lze přesměrovat obsah souboru příkazu, ale to se moc často nepoužívá. Jiná možnost přesměrování je spustit dva příkazy najednou a propojit standardní výstup prvního se standardním vstupem druhého. Toto propojení se děje přes tzv. rouru (pipe, jeden z prostředků IPC (meziprocesové komunikace) které OS Linux podporuje). Pro úplnost a programátory mohu dodat, že standardní vstup je to samé co deskriptor souboru 0, standardní výstup je totožný s deskriptorem souboru 1.

Záležitost trochu komplikuje skutečnost, že kromě standardního výstupu existuje ještě chybový výstup (deskriptor souboru 2). Ten je normálně také propojen s obrazovkou. Je zcela nezávislý na standardním výstupu, takže po přesměrování standardního výstupu do souboru se na obrazovku pořád mohou vypisovat chybová hlášení. To je také smysl chybového výstupu. Někdy se ale hodí do souboru přesměrovat i chybový výstup. Bash ho dovede přesměrovat jak do samostatného tak do společného se standardním výstupem.

Syntaxe přesměrování

PRIKAZ 1> soubor.txt
: Přesměrování standardního výstupu příkazu do souboru. Pokud takový soubor již existuje, bude přepsán. Jedničku lze vynechat.

PRIKAZ 1>> soubor.txt
: Přesměrování standardního výstupu příkazu do souboru. Pokud takový soubor již existuje, výstup bude přidán na jeho konec. Jedničku lze vynechat.

PRIKAZ 2> soubor.txt
: Přesměrování chybového výstupu příkazu do souboru. Dvojku nelze vynechat.

PRIKAZ &> soubor.txt
: Přesměrování standardního i chybového výstupu příkazu do souboru.

PRIKAZ 2>&1
: Přesměrování chybového výstupu příkazu do standardního.

PRIKAZ < soubor.txt
: Příkaz čte data ze souboru místo klávesnice.

PRIKAZ1 | PRIKAZ2
: Propojení dvou příkazů rourou. První posílá data druhému.

PRIKAZ1 | PRIKAZ2 | PRIKAZ3 > soubor.txt
: Kombinace přesměrování.

Nejčastěji používaná přesměrování jsou >, >> a |. Ostatní se používají jen výjimečně. Lze použít i jiná čísla deskriptorů souboru, pokud s nimi příkaz pracuje, ale v praxi jsem se s tím nikdy nesetkal.

Samozřejmě, že kombinované příkazy mohou být s různými parametry a podobně,

stejně jako samostatné příkazy. Nějaký reálný příkaz může vypadat např. takto:

```
rpm -qil | less
```

Pár poznámek k filtrům

Některé příkazy (a není jich málo) se umí chovat jako příkazy i jako filtry. Když jim jako parametr předáte název jednoho či více souborů, pracují s těmito soubory. Když jim žádný soubor jako parametr nepředložíte, budou číst data ze standardního vstupu. Jiné příkazy zase začnou číst data ze standardního vstupu (a tedy se chovat jako filtr), když jim jako parametr předáte znak - (pomlčka).

Když spustíte filtr jen tak (třeba omylem), bude očekávat data ze standardního vstupu, který bude připojen ke klávesnici. Můžete pak psát cokoliv na klávesnici a filtr to bude hladově požírat. Abyste se ho zbavili, musíte stisknout Ctrl-D. Jakmile ho filtr obdrží, ukončí načítání dat ze standardního vstupu, udělá si svoji práci a ukončí se. Znak ^D totiž znamená konec souboru. Můžete si to zkusit třeba s příkazem grep.

Nahrazování

Roury jsou jen jeden ze způsobů kombinace příkazů. Další způsob je nahrazování příkazu jeho výsledkem. Hypotetický příklad, máme příkaz SMAZ, který smaže zadaný soubor a NEJVETSI, který vypisuje název největšího souboru. Normálně můžeme spustit příkaz NEJVETSI, přečíst si název souboru, např. velky.txt a poté spustit příkaz SMAZ velky.txt. Bash dovede vzít jeden příkaz, spustit ho a jeho výsledek předat jinému příkazu sám, takže příkaz:

```
SMAZ `NEJVETSI`
```

udělá to samé. Příkaz v obrácených apostrofech se spustí jako první, výsledek se zapamatuje a pak bash spustí celý příkaz s tím, že text v obrácených apostrofech včetně nich samých nahradí tímto výsledkem. Alternativně (po novu) to lze napsat také takto:

```
SMAZ $(NEJVETSI)
```

Tato novější syntaxe má tu výhodu, že umožňuje nahrazování vnořovat.

Podmíněné spouštění

Každý program při svém ukončení vrací chybový kód. Konvencí je, že nedojde-li k chybě, vrací příkaz kód 0 a jinak kód vetší než 0, který chybu zhruba specifikuje. Bash umí tento kód logicky (tj. rozlišuje stav ANO/NE, kde ANO jsou kódy různé od nuly) vyhodnotit a na základě toho spustit další příkaz.

```
PRIKAZ1 && PRIKAZ2
PRIKAZ1  || PRIKAZ3
```

Příkaz 2 se provede jen tehdy, když příkaz 1 proběhne bez chyby. Příkaz 3 se provede naopak jen tehdy, když příkaz 1 proběhne s chybou. Je to klasické logické AND a OR (A a NEBO). Moc často se to nepoužívá. Nepopleťte si || s rourou.

Vícenásobné spouštění

Někdy se může hodit spuštění několika příkazů najednou. Stačí jednotlivé příkazy oddělit středníkem:

```
PRIKAZ1; PRIKAZ2
```

Bash – skripty

Když budete často používat příkazovou řádku, zjistíte, že mnohdy používáte stejnou sekvenci příkazů a že vás nebaví ji pořád opakovat. Také můžete zjistit, že tuto sekvenci používáte pravidelně. V takové chvíli uvítáte schopnost bashe automatizovat práci za pomoci skriptů. Není to věc okrajová, za pomoci skriptů např. startuje samotný operační systém. Mnoho prográmků a příkazů ve skutečnosti není nic jiného než právě skripty a tak podobně.

Pomocí skriptů lze rafinovaně kombinovat jednoduché příkazy, zabalit je pod jeden příkaz a vytvořit tak na míru komplexní nástroj řešící nějakou úlohu. Tato věc je ve skutečnosti považována za jeden z hlavních stavebních kamenů UNIXových operačních systémů, za jeden ze základních principů, na kterém jsou postaveny. Ve Windows je situace opačná, tam se daří mohutným programům, které řeší několik úloh najednou a neumí automaticky spolupracovat s ostatními programy. Windows jsou postavené na komponentové technologii, takže tam může přijít programátor a za použití různých komponent také vytvořit potřebný nástroj. Ale tato možnost je běžným uživatelům uzavřena, ti jsou odsouzeni k čekání na to, zda jim někdo potřebný nástroj nevytvoří. Pokud je často žádaný, tak ho samozřejmě někdo vytvoří, proto pro Windows existuje tisíce různých prográmků, které si můžete stáhnout nebo koupit. Příkazová řádka je oproti tomu určena pro běžné uživatele a zkušenější uživatel si dokáže potřebný skript napsat sám. Proto pro Linux neexistují tisíce prográmků, ale zato pro něj existují stovky jednoduchých a výkonných příkazů. Ty příkazy umí kde co, včetně třeba vypalování souborů na CD, neinteraktivní úpravy obrázků a podobně, příkazová řádka a skripty jsou v Linuxu skutečně mocný nástroj.

Pro lepší názornost proberu podrobně jeden příklad, který vystihuje p principiální odlišnost fungování Linuxu a Windows. Pokud vás to nezajímá, přeskočte následující dva odstavce. Klasická záležitost, komprimace souborů. Ve Windows nějakým způsobem vyberete/označíte soubory které chcete komprimovat a komprimační program z nich udělá zkomprimovaný balíček. Linux na to jde složitěji, protože vidí, že ve skutečnosti to jsou dvě úlohy a tak na to má dva příkazy. První úloha je spojit všechny soubory v jeden, druhá tento soubor zkomprimovat. Spojení souborů v jeden není zase až tak jednoduchá záležitost, jak by se mohlo zdát. Kromě jejich spojení a pamatování si, kde jeden končí a další začíná je potřeba s pamatovat atributy souboru. Tedy jejich jméno, čas vytvoření, čas modifikace, přístupová práva a mnohé další. Existují různé souborové systémy, které mohou mít různé atributy. Program který spojuje soubory dohromady musí dotyčný souborový systém znát. Proto existuje více různých takových programů. Především se používá tar, který obslouží všechny "UNIXové" souborové systémy, ale pokud používáte nějaké speciální rozšíření, třeba přístupová práva ACL, musíte mít speciální (upravenou) verzi taru, která si s nimi rozumí. Stejně tak existuje mnoho různých komprimačních algoritmů, ale nemusí jít jen o ty, takový soubor můžete chtít třeba jen transformovat do jiné formy, třeba BASE64 nebo UU a podobně. Udělat program, který podporuje všechny možné souborové systémy a jejich varianty a všechny komprimační algoritmy a transformace je prakticky nemožné. A nebo pohled z druhé strany, vymyslíte novou revoluční komprimační metodu souborů, třeba specializovanou na textové soubory, kde dosahuje vynikajících komprimačních schopností (takové věci se čas od času opravdu objevují). Kdyby nebylo odděleno spojování souboru do samostatného programu, museli byste tuto záležitost implementovat do svého nového komprimačního programu. Ale to že umíte vymyslet revoluční komprimační metodu ještě neznamená, že jste odborník na všemožné souborové systémy. Naštěstí v Linuxu se tím trápit nemusíte (ve Windows také ne, protože ty podporují jeden nebo max. dva různé souborové systémy, ale to už je zase jiná záležitost). Rozdělením těchto dvou úloh na dva různé příkazy dosahujete

vysoké flexibility. Se zatarovaným souborem, který spolehlivě uchovává obsah disku můžete dělat v podstatě cokoliv. A ani na používání to není složité, třeba v mc si můžete označit soubory a ty pak výběrem jedné položky v menu zabalit a zkomprimovat najednou. Na pozadí se pak spustí dva příkazy (mimochodem propojené rourou). Srovnejte to se situací ve Windows, tam existuje několik archivátorů, každý podporuje pár komprimačních metod, o souborové systémy se moc nestarají, ve svých funkcích se překrývají, takže některé komprimační metody jsou implementovány několikrát, ty pak občas trpí nekompatibilitou a na soubory zkomprimované vzácnějšími komprimačními metodami stejně volají po vzoru UNIXu utility příkazové řádky. Když se kdysi ve Windows objevila podpora dlouhých jmen v názvu souborů (tedy nová varianta souborového systému) mnoho komprimačních programů s tím mělo dlouho problémy, než jejich programátoři do nich implementovali podporu této nové varianty. V Linuxu se vám nic takového nestane, s novou variantou souborového systému přijde nová verze taru (kterou udělají autoři toho souborového systému, kteří mu rozumí a kteří to musí udělat, s novým souborovým systémem jednoduše musí poskytnout nástroje pro jejich formátování, zálohování (to je ten tar) a podobně). Programátoři komprimačních programů se o nový souborový systém vůbec nestarají, nemusí. To jak to funguje ve Windows je imho plýtvání. Není to flexibilní a ztěžuje to inovaci samotnému MS, o inovaci souborového systému ve Windows nějakou třetí stranou si pak zřejmě můžeme nechat jen zdát.

Předchozí vysvětlení bylo poněkud obšírné, ale myslím že dobře vystihuje rozdíly mezi tím jak fungují Windows a Linux. Že usnadní pochopení Linuxu lidem znající Windows. Že jim to ukáže, že ne všechno na co jsou z Windows zvyklí a chtějí to i po Linuxu, je ve skutečnosti dobré. Že Linux prostě je jiný operační systém a je potřeba k němu přistupovat jinak, že to není jenom jinak vypadající kopie Windows, ale že vychází z jiných principů. Tyto se nám mohou líbit a pak budeme Linux preferovat a nebo nebudou a pak dáme přednost něčemu jinému. Linux je flexibilnější systém a není to náhodou, je to cílený stav, na druhou stranu je ale náročnější se naučit s ním a v něm pracovat. Ale teď už rychle zpátky k příkazovému řádku a skriptování.

Podstata skriptů je jednoduchá. Příkazy místo na příkazový řádek napíšete do souboru obyčejným textovým editorem, každý příkaz na nový řádek. *Skript je tedy obyčejný textový soubor.*

Spouštění skriptů

Když máte skript napsaný, asi ho budete chtít také spustit. Základní způsob je spuštění nového bashe, kterému skript (textový soubor) předáte jako parametr. Nový bash bude načítat soubor řádek po řádku a provádět uvedené příkazy. To mnohdy stačí a více k životu netřeba. Když to nestačí, nabízí bash práci s proměnnými, větvení na základě podmínek a cykly. Pomocí těchto nástrojů můžete skriptu udělit malou inteligenci a vytvořit v podstatě jednoduchý prográmek, který sice sám neprovádí nic, ale spouští příkazy, které už něco dělají.

Aby byla iluze prográmku dokonalá, lze skript nastavit jako spustitelný soubor a když se na prvním řádku skriptu bude skrývat magická formulka, lze takový skript spouštět stejně jako jiné příkazy a programy. To jest nemusí se předávat bashi jako parametr. Ta magická formulka vypadá takto:

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

Tohle je univerzální způsob. Existuje mnoho programů, které umí provádět skripty napsané ve vlastním jazyce. Některé jsou specializované na nějakou konkrétní činnost jako např. awk, jiné jsou univerzální jako python, perl nebo ruby. Tyto jsou pak označovány jako interpretované programovací jazyky. Bash samotný je možno v tomto

kontextu považovat za velmi primitivní programovací jazyk specializovaný na spouštění jiných programů. Když máte nějaký skript (pro libovolný jazyk a program), kterému nastavíte právo na spuštění a spustíte jej, bash?? jej převezme, podívá se na první řádek, z něj zjistí pro jaký interpretr (program) je skript určen, ten spustí a skript mu předá jako parametr. V našem případě se předá programu /bin/bash. To by bylo všechno ke spouštění skriptů.

Proměnné

Proměnné v bashi jsou jakési zkratky za nějaký kus textu, kterému se správně říká textový řetězec. Této zkratce nejprve pomocí znaku '=' text přiřadíte a poté můžete tuto zkratku používat místo tohoto textu. Aby nedocházelo ke zmatení co je zkratka a co text, je nutno při použití před zkratku dávat znak '$'. Tedy to funguje asi takto:

```
soub="data.txt";
echo "Smažu soubor $soub";
rm $soub;
echo "Soubor $soub je smazán!";
```

Bash tento skript bude načítat řádek po řádku. Po přečtení prvního řádku založí proměnnou soub a přiřadí ji textový řetězec který je mezi uvozovkami. Po načtení druhého řádku nahradí řetězec $soub řetězcem data.txt a celý příkaz spustí. Příkaz echo vypisuje zadaný text na obrazovku (přesněji standardní výstup). Po načtení třetího řádku opět nahradí řetězec $soub řetězcem data.txt a příkaz spustí. Příkazu rm maže zadané soubory. Co udělá čtvrtý řádek už jistě víte sami. Příklad postrádá logiku, protože místo proměnné mohu klidně všude uvádět rovnou název souboru. Příklad ovšem nabude logiky po tom, co jej upravím na:

```
soub=`NEJVETSI`;
echo "Smažu soubor $soub";
rm $soub;
echo "Soubor $soub je smazán!";
```

Komu první řádek nic neříká, měl by si přečíst předchozí kapitolu, konkrétně pasáž o nahrazování příkazů. Zde máme jakýsi fiktivní příkaz NEJVETSI, který vrací název největšího souboru v adresáři. Nyní už tedy nemohu ve skriptu nahradit proměnnou přímo názvem souboru, protože nevím předem který soubor bude při spuštění skriptu největší. Nemohu proměnnou nahradit všude ani konstrukcí `NEJVETSI`, protože při provádění čtvrtého řádku už je ten největší soubor smazaný a tak by se nám vypisoval nesmysl (název jiného souboru). Teď už byste tedy měli tušit k čemu ty proměnné jsou vlastně dobré a jak se používají. Nepřehlédněte, že každý příkaz je ukončen středníkem.

Tak jako můžete předat parametry příkazům, můžete je předat i spouštěnému skriptu. K jednotlivé parametry budou schovány v proměnných $1, $2, ..., $9. Více parametrů lze použít, ale nebudou přímo přístupné. Můžete si akorát pomoci příkazem , po jeho zavolání se obsah $2 zkopíruje do $1, $3 do $2 a tak dále, ve $9 se tak objeví nový parametr a $1 bude ztracen. Je mo6no i zadat příkaz např. jako 3, pak dojde rovnou k posunu o 3 místa. Kromě toho proměnná $@ obsahuje všechny parametry jako jeden dlouhý text, $# obsahuje číslo udávající počet parametrů a $0 obsahuje název příkazu.

V bashi existuje mnoho dalších proměnných. Mimo jiné všechny systémové. Třeba systémová proměnná PATH je přístupná jako $PATH. Proměnné které si sami zakládáte jsou lokální a do systémových se automaticky nepromítají. Příkazem export $PROMENNA z lokální proměnné uděláte systémovou.

Podmínky

Uvedený příklad není dokonalý. Když jej spustíme v adresáři kde nejsou žádné soubory, bude fungovat nesmyslně. Bylo by dobré, kdyby bylo možné si mezi prvním a druhým řádkem otestovat, zda příkaz NEJVETSI skutečně vrátil název nějakého souboru a následující příkazy provedl jen tehdy, když ano. Tedy aby ten zbytek provedl podmíněně. Protože o tom mluvím, tak je to samozřejmě možné, dělá se to takto:

```
soub=`NEJVETSI`;
if test "$soub" = ""; then
  echo "Nemažu nic.";
else
  echo "Smažu soubor $soub";
  rm $soub;
  echo "Soubor $soub je smazán!";
fi
```

Jsou tu dvě nové věci. Příkaz test, který testuje zda proměnná $soub je prázdná nebo nikoliv. Když ano, vrací logické ANO, když ne, vrací logické NE. A dále tu je podmínková konstrukce if – then – else – fi. Mezi if a then se dává podmínka, která vrací logické ANO nebo NE. Mezi then a else se dávají příkazy, které se provedou když podmínka vrací ANO. Mezi else a fi se dávají příkazy, které se provedou když podmínka vrací NE. Klíčová slova této konstrukce vychází z angličtiny, do češtiny by se dala přeložit takto: když – potom – jinak – žydk. To poslední slovo je jako první, ale pozpátku, označuje to konec.

TODO: Popsat case

Cykly

TODO: Popsat for, while a until

Funkce

TODO: Popsat function

Ostatní

TODO: Popsat select

Bash – řízení úloh

Rozmýšlel jsem se, zda mám vůbec o řízení programů (v tomto kontextu se používá termín úloha, budu se toho držet) v bashi vůbec psát. Nakonec jsem se rozhodl že ano, i když je to věc v současné době v podstatě nepotřebná. Budu tak moci snáze vysvětlit, co to je program běžící na pozadí, což se používá docela běžně. Řízení úloh sloužilo především ke spouštění několika interaktivních programů najednou na jednom terminálu. To je v době grafických rozhraní už zbytečné, každou úlohu si můžete spustit ve vlastním okně. Co se týče textového rozhraní, je lepší druhý program spustit na vedlejší konzoli, než v té samé. A dokonce, i když se k nějakému počítači přihlásíte přes telnet nebo ssh a máte tedy k dispozici skutečně jen jeden terminál, je lepší pro řízení úloh použít program screen, který vytváří virtuální terminály.

Přesto se můžete s řízením úloh v bashi setkat, hlavně nedopatřením, když stisknete Ctrl-Z. Tato horká klávesa přesune úlohu na pozadí a zastaví ji. To jest, objeví se příkazový řádek, program zmizí. Nepoučenému laikovi to přijde jako by program spadnul. Nespadnul, pořád "tam někde je" a čeká až ho vyvoláte příkazem fg

(foreground = popředí) na popředí. Také můžete dát příkaz bg (background = pozadí). Potom zůstane na pozadí, tj. neuvidíte ho, ale poběží. Je jasné, že textový editor běžící na pozadí je k ničemu, ten tam může být schován pozastavený, ale kopírovaní souboru na pozadí už smysl má. Zatím co se soubor kopíruje můžete na terminálu dělat něco jiného.

Jestli si to budete chtít zvídavě vyzkoušet, pak k experimentům doporučuji příkaz mpg123, který přehrává mp3, tedy jestli máte zvukovku a nějaký mp3. Budete tak mít jasno co se právě děje i když program neuvidíte.

Spouštět úlohy na pozadí pomocí Ctrl-Z a příkazu bg je zbytečně pracné. Pokud už při spouštění programu víte, že chcete aby běžel na pozadí, dejte za něj znak '&' a on se spustí na pozadí rovnou. To se používá docela často, hlavně v různých skriptech a nebo také v mc při spouštění některých programů asociovaných s nějakým typem souborů. Příklad spuštění programu na pozadí:

```
mpg123 soubor.mp3 &
```

Přehled základních příkazů pro řízení úloh:

Ctrl-Z	Úlohu běžící na popředí (aktivní úlohu) zastaví a přesune na pozadí.
jobs	Vypíše seznam úloh
kill	Ukončí úlohu
stop	Zastaví úlohu
bg	Spustí zastavenou úlohu na pozadí
fg	Přesune úlohu na popředí

Úloha na popředí nemůže být zastavená. Všechny příkazy (kromě Ctrl-z) pracují jen s úlohami na pozadí (to je logický, když bude nějaká úloha na popředí, nemůžete tyto příkazy zadávat). Když máte na pozadí více úloh než jednu, je potřeba za příkaz přidat identifikátor úlohy aby příkaz věděl které se to týká. Identifikátor může být buď pid (?? doplnit odkaz na kapitolu popisující procesy) programu, číslo úlohy (zjistíte z jobs), nebo jednoznačný řetězec z příkazu. Kromě pid se všechno dává za procento, tedy např:

```
fg 12345    # úloha jejíž proces má pid 12345
fg %2       # úloha s číslem 2
fg %mpg     # úloha jejíž příkaz začíná textem 'mpg'
fg %?soubor # úloha jejíž příkaz obsahuje text 'soubor'
```

Bash – speciální znaky a názvy souborů

Linux umožňuje používat v názvech souborů téměř libovolné znaky. Jediný zakázaný znak který mě napadá je / kterým se oddělují části cesty. Mnoho znaků ale má speciální význam, což stěžuje jejich používání v názvech souborů a proto se jejich používání nedoporučuje. Např. když budete mít soubor nazván -r, tak tím zmatete náš fiktivní editor, bude si myslet, že jde o parametr. Soubor s názvem martin*.txt zase zmate bash (nehledě na to, že by mě zajímalo, jak by se na to zatvářily Windows, které tento znak přímo zakazují), který vrátí všechny soubory s příponou .txt a začínající textem martin. Nejlepší je, když se názvy souborů skládají jen z písmen anglické abecedy, čísel, podtržítek a teček. Nedoporučuje se používat ani ve Windows oblíbené mezery (protože mezera má funkci oddělovače parametrů) a písmena s diakritikou (protože Windows a Linux používají rozdílná kódování češtiny a Windows se s tím neumí vyrovnat)

I když se toho budete držet, stejně se můžete s takovými názvy souborů setkat. Např. na cédéčku z nějakého časopisu. Pak se vám bude hodit, když budete vědět jak vypnout speciální význam nějakého znaku. Lze to udělat buď tak, že se před dotyčný speciální znak dá obrácené lomítko \ a nebo se celý název dá do uvozovek. Lomítka se používají při interaktivním zadávání příkazů a vlastně se o to ani nemusíte moc starat, doplňování názvů tabelátorem to udělá za vás. Uvozovky se používají spíše při skriptování. Lze použít jednoduché ' nebo dvojité ". Jednoduché vypnou všechny speciální znaky kromě samotné jednoduché uvozovky ('). Dvojité vypnou všechny speciální znaky kromě samotné dvojité uvozovky ("), znaku pro nahrazování (`) a znaku pro proměnné ($).

Seznam speciálních znaků, které rozlišuje bash:

;	oddělovač příkazů
&	spouštění na pozadí
()	seskupení příkazů / spuštění v novém bashi
{}	blok příkazů (skriptování)
|	roura
< > &	přesměrování
* ? [ ] ! ~	metaznaky v názvech souborů
' " \	omezování významu jiných znaků
`	nahrazování
$	proměnné (skriptování)
mezera tabulátor	oddělovače parametrů
#	komentáře (skriptování)

Příkazy

Tato sekce se věnuje příkazům pro příkazovou řádku. Momentálně je uveden pouze přehled vybraných příkazů. Časem se tu snad objeví i popis těch nejzajímavějších. Popis některých příkazů bude i v sekci konfigurace.

Přehled vybraných příkazů

správa procesů

at, atq, atrm	Spouštění programů v zadaném čase.
batch	Jako at, není-li zadán čas, spustí program až průměrná zátěž systému klesne pod 0,5.
bash, sh, csh, tcsh	Různé interprety příkazů (příkazové řádky, shelly).
crontab	Nastavuje cron, který umožňuje pravidelné spouštění programů.
kill	Pošle signál zadaným (pid) procesům. Defaultně se posílá signál TERM, který procesu přikazuje, aby se ukončil. Proces ho může ignorovat. Signál 9 (KILL) ignorovat nemůže.
killall	Jako kill, ale místo pid se zadává název procesu. Dojde tak k poslání signálu všem procesům se stejným jménem. Jméno procesu je shodné s názvem binárního souboru programu (názvy skriptů se tedy nepoužijí, místo toho se použije název interpretru skriptu).
nice, renice	Mění prioritu procesu. Několik běžících programů se stejnou prioritou se dělí o výkon procesoru rovnoměrně. Změnou priority lze výkon procesoru rozdělovat nerovnoměrně. Běžný uživatel může prioritu jen snižovat a program tak zpomalovat (třeba pro nějaké výpočty nebo kopírování na pozadí), root ji může i zvyšovat.
pidof	Zobrazí pid zadaných procesů.
ps	Vypisuje informace o běžících procesech.
run-parts	V zadaném adresáři se naleznou všechny skripty, setřídí se a postupně jeden za druhým spustí.
top	Interaktivní informace o běžících procesech. Informace se neustále aktualizují, lze je různě třídit a podobně. Výstup lze přesměrovat do souboru a zjišťovat tak změny v čase.

jazyky a podobné záležitosti

bc	Jednoduchý jazyk a interpret pro matematické operace s neomezenou přesností. Lze používat i interaktivně.
gawk	Jednoduchý jazyk pro zpracování textových souborů se sloupci textu. Umožňuje vybírat sloupce nebo řádky, sčítat čísla ve sloupcích, měnit jejich obsah, různě přesouvat a podobně.
gs, ghostscript	Umožní zpracovat PostScriptové soubory. Umí je vytisknout i na ne PostScriptové tiskárně, převést na PDF, bitmapový obrázek a podobně. V X Window ho dovede i zobrazit, ale bez možnosti ovládání, proto je pro zobrazení lepší používat nadstavbu gv (GhostView).
m4	Univerzální textový preprocesor (makroprocesor). Lze použít například k snadnému generování statických html stránek. Tedy bez skriptování na straně serveru i bez toho, že byste do každé stránky museli pracně psát opakující se věci a při jejich změně pak všechny stránky

	opravovali.
make	Interpret pro řízení překladu programů ze zdrojových kódů do binární podoby. Používá se i pro jejich instalaci. Můžete jej použít ale k řízení jakékoliv činnosti, při které vznikají soubory. Třeba generování html stránek pomocí m4. Make umí automaticky nechat generovat jen ty soubory, kterým se změnily zdrojové soubory,
perl	Známý interpretovaný programovací jazyk s divokou syntaxí. Některé příkazy Linuxu jsou napsány v perlu. Perl je standardní součást většiny linuxových distribucí (výjimkou jsou jednodisketové distribuce a podobně).
python	Známý objektový interpretovaný programovací jazyk. Hodně grafických konfiguračních nástrojů je napsáno právě v něm. Podobně jako perl i python je standardní součástí většiny linuxových distribucí.
ruby	Plně objektový interpretovaný programovací jazyk.

zpracování textů

cat	Vypisuje obsah souboru na standardní výstup.
colrm	Odstraní zadané sloupce ze souboru. Sloupcem je míněn jeden znak (jejich řada pod sebou), lze zadávat rozsah.
column	Formátuje soubor nebo standardní vstup do sloupců.
comm	Porovnává řádky dvou zadaných souborů nebo soubor se standardním vstupem. Umožňuje vypsat řádky jedinečné pro jeden soubor nebo jedinečné pro druhý soubor nebo řádky oběma souborům společné.
csplit	Viz práce se soubory.
cut	Vybere zadané sloupce nebo pole ze vstupních souborů.
diff	Porovnává dva textové soubory, vypisuje ve speciálním formátu řádky, v kterých se soubory liší. Tento výpis lze programem patch aplikovat na první soubor a vytvořit z něj tak druhý. Používá se ke zjištění rozdílů mezi soubory a pro vytváření záplat (patchů) zdrojových kódů.
diff3	Porovnává tři soubory a vypisuje rozdíly mezi nimi. Umí spojit rozdíly ze dvou souborů, které mají stejného předchůdce.
grep, egrep, fgrep	Viz práce se soubory.
expand, unexpand	V zadaných souborech nahrazuje tabulátory mezerami a naopak.
fmt	Formátuje odstavce na zadanou délku. Odstavce musí být odděleny prázdnou řádkou.
fold	Zalamuje hrubě dlouhé řádky na zadanou velikost.
head	Vypisuje začátek zadaného souboru (defaultně 10 řádek).
merge	Synchronizuje změny dvou různých verzí souborů oproti originálu.
paste	Spojuje zadané soubory horizontálně do jednoho. Tedy obsahy zadaných souborů budou tvořit sloupce výsledného souboru.
rev	Vypisuje obsah souboru pozpátku
sed	Neinteraktivní textový editor. Umožňuje manipulaci s obsahem textového souboru ze skriptů. Třeba nahrazování textu, mazání řádků a

	podobně.
tac	Jako cat, ale řádky vypisuje pozpátku.
tail	Jako head, ale vypisuje konec souboru.
tr	Nahrazuje zadané znaky za jiné.
uniq	Odstraňuje duplicitní řádky.
wc	Počítá počet znaků, slov a řádků v souboru.

administrace

change	Zjišťuje a nastavuje datum vypršení platnosti hesla.
chfn	Nastavuje informace o uživateli v /etc/passwd.
chgrp	Nastavuje gid souboru (skupinu do které patří).
chmod	Nastavuje přístupová práva souboru.
chown	Nastavuje uid souboru (majitele kterému patří), umožňuje zároveň nastavit i gid (skupinu).
chsh	Nastavuje jaký shell (příkazový řádek) se má spouštět po přihlášení.
env	Nastavuje dočasně systémové proměnné.
hostid	Nastavuje ID počítače.
hostname	Nastavuje jméno počítače.
passwd	Změna hesla uživatele.

práce se soubory

chattr	Nastavuje speciální atributy souboru u souborového systému ext2.
cksum	Vypočítá kontrolní součet (CRC) souboru.
cmp	Porovná soubory nebo soubor a standardní vstup, zda jsou shodné.
cp	Kopíruje soubory.
csplit	Rozdělí textový soubor na části podle regulárního výrazu.
dd	Speciální příkaz pro kopírování souborů a jejich částí. Umí provádět konverzi ASCII/EBCDIC, konverzi velikosti bloků a podobně. Používá se např. při kopírování boot sektoru disku a podobných záležitostech.
grep, egrep, fgrep	Prohledává zadané soubory.
file	Viz informační příkazy.
ll	Vytváří pevné a symbolické odkazy.
ls	Vypíše obsah zadaných adresářů.
lsattr	Vypíše speciální atributy souboru u souborového systému ext2.
lsof	??
mkdir	Vytváří adresář.
mv	Přesouvá nebo přejmenovává soubory.
pwd	Vypisuje cestu k pracovnímu adresáři.

rename	Umožňuje částečné i hromadné přejmenování souborů.
rm	Maže soubory, root může i adresáře v případě, že nejsou prázdné. Při rekurzivním mazání se adresáře nejprve vyprázdní a teprve potom mažou, nepodaří-li se z nějakého důvodu adresář vyprázdnit (např. z důvodu zákazu zápisu do adresáře), obyčejný uživatel adresář nemůže smazat.
rmdir	Maže zadané prázdné adresáře.
split	Příkaz pro rozdělení souboru na části.
sum	Vypočítá kontrolní součet souboru podle BSD nebo System V verze algoritmu.
symlinks	Vypíše informace o symbolických odkazech v zadaných adresářích.
touch	Nastavuje čas u zadaného souboru. Buď na aktuální nebo na zadaný. Jestliže zadaný soubor neexistuje, vytvoří se.

ostatní

login	Příkaz umožňující přihlášení uživatele do systému na příkazové řádce.
lpq, lpr, lprm	Příkazy pro tisk pro tiskový systém.
man	Zobrazí manuálovou stránku (nápovědu) k zadanému příkazu.
newgrp	Příkaz pro dočasnou změnu defaultní skupiny uživatele.
script	Po zadání tohoto příkazu se až do ukončení příkazového řádku zapisuje do souboru všechno co se děje na obrazovce.
sleep	Příkaz nedělá nic, akorát čeká po zadanou dobu. Je tak možno vložit prodlevu mezi spuštění dvou příkazů ve scriptu.
su	Spouští nový příkazový řádek s jiným přihlášeným uživatelem. Vyžaduje jeho heslo. Lze zadat i příkaz, který se má provést (s právy zadaného uživatele) místo spuštění nového příkazového řádku. ??
tee	Načítá data z standardního vstupu a vypisuje je na standardní výstup a do zadaného souboru. Jde o jakési rozdvojení proudících dat.

práce s obrazovkou

clear	Smaže obrazovku.
echo	Vypisuje zadaný text na standardní výstup.
less	Není příkaz, ale často se s příkazy používá. Je to program který zachycuje standardní vstup a umožňuje ho interaktivně prohlížet. Používá se, když výstup nějakého příkazu na obrazovku je dlouhý a nevejde se na ní. Takový výstup se rourou přesměruje na program less. Less je tedy něco jako prohlížeč textů, má velmi propracované ovládání, vyhledávání a podobně.
mesg	Povolí nebo zakáže posílat ostatním uživatelům na váš terminál zprávy pomocí programu write.
more	Primitivní varianta less, používalo se před příchodem less.
printf	Vypisuje text v zadaném formátu. Formát se zadává stejně jako u funkce

	prinf programovacího jazyka C.
reset	Smaže obrazovku terminálu a jeho parametry nastaví na defaultní hodnoty.
write	Pošle zprávu zadanému uživateli na zadaný terminál. V čase GUI rozhraní to není moc užitečné a funkční.
xargs	Příkaz spouští opakovaně zadaný příkaz a jako parametry mu předává postupně to, co načte ze standardního vstupu. Používá se to ve speciálních případech popsaných jinde ??dopsat a dodat odkaz??.

informační příkazy

apropos	Vyhledává manuálové stránky které se věnují zadanému heslu. Neprohledává se celá manuálová stránka, ale jen její zkrácený (úvodní) popis.
date	Vypisuje datum v nastavitelném formátu.
df	Disk Free. Informuje o zabraném a volném místu na připojených discích.
du	Disk Usage. Informuje o využití disku, tj. velikost zadaných adresářů a jejich podadresářů.
printenv	Vypisuje nastavení systémových proměnných.
file	Identifikuje druh zadaného souboru podle jeho obsahu.
find	Vyhledává soubory podle mnoha různých kritérií. Umožňuje např. nalézt všechny adresáře staré max. týden, které patří rootovi a do kterých mají všichni právo zápisu. Dále umožňuje na každý nalezený soubor spustit nějaký příkaz, třeba změnu nastavení přístupových práv.
finger	Vypíše informace o zadaných uživatelích.
free	Vypisuje informace o využití paměti počítače.
fuser	Vypisuje které procesy používají zadaný soubor nebo souborový systém. Užitečné, když nejde odpojit disk z důvodu, že ho používá nějaký program, a vy nevíte který.
hostid	Vypisuje ID počítače.
hostname	Vypisuje jméno počítače nebo jeho ip adresu.
id	Vypisuje informace o zadaném uživateli.
locate	Vyhledává soubory. Hledání neprobíhá na disku, ale v databázi souborů. Díky tomu je hledání rychlé, ale nepřesné. Nezachycuje změny provedené po poslední aktualizaci databáze.
ls	Vypíše obsah zadaných adresářů.
lsattr	Vypíše speciální atributy souboru u souborového systému ext2.
lsof	??
pidof	Zobrazí pid zadaných procesů.
ps	Vypisuje informace o běžících procesech.
pwd	Vypisuje cestu k pracovnímu adresáři.
symlinks	Vypíše informace o symbolických odkazech v zadaných adresářích.
uptime	Vypisuje aktuální čas, jak dlouho je počítač spuštěn, počet přihlášených uživatelů a zatížení systému.

users	Vypisuje seznam jmen přihlášených uživatelů.
w	Vypisuje podrobné informace o přihlášených uživatelích včetně toho co právě dělají (kombinace uptime, who a ps -a).
whereis	Příkaz vypisuje umístění programu a manuálových stránek zadaného příkazu.
who	Vypisuje podrobné informace o přihlášených uživatelích.

práce s disky a souborovými systémy

fdformat	Nízkoúrovňový formát diskety. Nezaměňujte to s vytvářením souborového systému (ve Windows při formátování diskety dojde automaticky i k vytvoření souborového systému FAT, v Linuxu si můžete vybrat mezi několika souborovými systémy, které se vytvářejí samostatným příkazem).
quota	Informuje o využití disku a nastavených kvótách (nastavené omezení uživatele čerpat kapacitu disku).
stat	Vypisuje informace o i-uzlu souboru (atributech).
uname	Vypisuje informace o počítači a operačním systému.

síťové příkazy

finger	Viz informační příkazy.
ftp	Řádkový FTP klient. Lze jím ve skriptech automaticky stahovat nebo posílat soubory nebo jej lze ovládat interaktivně.
tload	Zobrazuje průměrné zatížení.
pppd	Zajišťuje připojení k internetu přes ppp. Konfiguraci tohoto prográmku se věnuje kapitola modem v sekci konfigurace.
dial	Příkaz pro komunikaci s modemem, používá ho např. pppd, příklad použití je v kapitole modem v sekci konfigurace.

zálohování, komprimace a konverze

bz2	Výkonný komprimátor, který možná časem nahradí gzip, protože je viditelně účinnější.
compress	Komprimátor, kdysi defaultní program pro komprimaci souborů na UNIXových systémech.
gzexe	Komprimuje spustitelné soubory
gzip, gunzip	Komprimátor a dekomprimátor. Gzip je v Linuxu standardní formát, podobně jako zip ve Windows. Gzip neumí komprimovat víc souborů najednou, můžete (najednou) zkomprimovat každý zvlášť a nebo je před komprimací spojit do jednoho programem tar. Používá ten samý komprimační algoritmus jako zip ve Windows a má podobnou účinnost.
iconv	Konvertor kódování textových souborů. Podporuje mimo jiné utf8 (formát unicode (univerzální kódování)), iso8859-2 (linuxové kódování češtiny) a cp1250 (Windows kódování češtiny).

mimeencode	Převede soubor do formátu base64 nebo quoted-printable a zpět. Tyto formáty se používají při posílání souborů elektronickou poštou.
pr	Převede textový soubor do podoby vhodné pro tisk.
shar, unshar	Ze zadaných souborů vytvoří samorozbalovací balík v textovém formátu, který je možno posílat poštou bez překódování. Na počítači kde se má balík rozbalit musí být standardní nástroje jako gzip nebo uuencode. Unshar umožňuje rozbalit balík ručně a to i když je obsahem nějakého textového souboru, jako je poštovní zpráva.
split	Viz práce se soubory.
tar	Archivátor. Ukládá zadané soubory do jednoho a s nimi ukládá i jejich atributy. Samozřejmě že v případě potřeby je umí obnovit. Používá se při zálohování disků, vytváření zkomprimovaných balíčků a podobně.
uuencode, uudecode	Konvertuje zadaný soubor do uu formátu. Tento je možno bezpečně vkládat do textových souborů jako je poštovní zpráva a pak je zase obnovit.
zcat, zcmp, zdiff, zgrep, zmore	Příkazy dělají to samé jako cat, cmp, diff, grep a more, ale pracují se zkomprimovanými soubory. Některé textové editory, jako např. VIM také umí automaticky pracovat se zkomprimovanými soubory, takže je možno je mít na disku zkomprimované a přitom s nimi pohodlně pracovat jako s normálními.
znew	Zkonvertuje soubory komprimované programem compress na gzip.

Seznam dalších, zatím neroztříděných příkazů.

```
hwclock
netdate
ntptimeset

dip

kbdrate

setserial
slattach

insmod
lsmod
rmmod
modprobe
depmod

fdisk
debugfs
dumpe2fs
fsck.*
mkfs.*
mkswap
mount
rdev
```

`rootflags`
`showmount`
`swapdev`
`swapon`
`swapoff`
`sync`
`tune2fs`
`umount`
`cron`
`dmesg`
`ldconfig`
`ifconfig`
`iptables`
`ping`
`route`
`traceroute`
`badblocks`
`chroot`
`halt`
`reboot`
`runlevel`
`shutdown`
`telinit`
`netstat`
`useradd`
`userdel`
`usermod`
`wall`
`service`

Konfigurace Linuxu

Tato sekce obsahuje ukázky z různých konfiguračních souborů, které se mi zdají potřebné, užitečné a nebo často používané. Píšu to vlastně pro sebe, občas někde někomu konfiguruji Linux a nemůžu si vzpomenout, jak se zrovna udělá tohle či onohle a nebo je to delší a nechce se mi to psát, když to jde snadno zkopírovat. Myslím že by to mohlo přijít vhod i někomu jinému a proto to doplňuji komentáři. Rozsáhlejší konfigurace, třeba pro textový editor Vim budou k dispozici jako balíček ke stažení.

Mountování – /etc/fstab

`/dev/fd0  /a  auto  noauto,user,showexec=no  0 0`

Umožní připojit libovolnému uživateli disketu příkazem `mount /a` k adresáři /a (musí existovat). Parametr showexec=no zajišťuje, že soubory na souborovém systému FAT nebudou mít nastaveny práva pro spuštění.

`/cesta/cd.img  /cesta/adr  iso9660  user,noauto,loop,ro  0 0`

Umožní připojit libovolnému uživateli soubor s obrazem CD disku k adresáři adr. Tento soubor lze získat např. zkopírováním CD média (je-li CD mechanika zařízení /dev/hdc, pak např. příkazem `cp /dev/hdc cd.img`) nebo jeho vytvořením příkazem mkisofs, který se používá při vypalování (lze si tak před vypálením obrazu CD otestovat jeho obsah a funkčnost). Je možno také mít na disku několik různých

obrazů CD médií, tyto mít trvale připojené k adresářům a tak mít okamžitý a rychlý přístup k často používaným cédéčkům.

```
/dev/scd0   /cd   iso9660   user,noauto,ro,iocharset=iso8859-2   0 0
```

Je známo, že různé systémy používají různé kódování češtiny. Proto by se čeština neměla používat v názvech souborů, zvláště na výměnných médiích. Když vám dá někdo vypálené CD, kde se čeština v názvech souborů používá, můžete si pomoci parametrem iocharset a cédéčko na vás bude mluvit hezky česky.

Připojení k internetu přes modem

V Linuxu je mnoho způsobů, jak se připojit k internetu. Já vám ukážu ten nejprimitivnější způsob, který by měl být dostupný a funkční v každé linuxové distribuci a to jak v grafickém tak i textovém rozhraní. Tento způsob obnáší vytvoření několika konfiguračních souborů a dvou skriptů, jeden pro připojení a druhý pro odpojení. Ukáži dvě konfigurace, jednu pro GPRS (připojení přes mobil k Eurotel Data Nonstop) a druhou pro dial up (připojení přes seznam.cz, které nevyžaduje registraci).

GPRS

/etc/ppp/options

```
lock
```

/etc/ppp/peers/gprs

```
connect    '/usr/sbin/chat -v -f /etc/ppp/peers/gprs_up'
disconnect '/usr/sbin/chat -v -f /etc/ppp/etg1/gprs_down'
/dev/gprs
115200
xonxoff
modem
default-asyncmap
defaultroute
noipdefault
idle 600
noauth
```

Parametr /dev/gprs říká, se kterým zařízením se má komunikovat. Já mám svůj mobil připojen pomocí sériového kabelu k prvnímu sériovému portu a proto /dev/gprs je u mě symbolický link na /dev/ttyS0 (ve Windows známo jako COM1).

/etc/ppp/peers/gprs_up

```
TIMEOUT 6
ABORT BUSY
ABORT ERROR
ABORT "NO CARRIER"
ABORT alarm
ABORT Failed
""
'ATZ' OK
AT+CGDCONT=1,"IP","internet" OK
"ATDT*99***1#" CONNECT
```

Aby tohle fungovalo, musíte mít gprs v mobilu správně nakonfigurováno.

/etc/ppp/peers/gprs_down

```
"" "+++ath"
```

gprs

```
#!/bin/sh
xterm -geom 40x15 -e /usr/sbin/pppd nodetach call gprs
```

Tohle je příkaz, který spuští připojení k internetu v gui. Na konzoli stačí dát přímo příkaz `/usr/sbin/pppd nodetach call gprs`. Pokud vynecháte parametr nodetach, připojení se provede na pozadí. Příkaz si můžete pojmenovat libovolně. Je nutno ho dát do cesty, třeba do /usr/bin nebo do /home/uzivatel/bin. Dále je potřba mu nastavit právo pro spuštění (`chmod a+x gprs`).

Program /usr/sbin/pppd musí být suidnutý, to znamená, že se musí spouštět s právy roota. Že je suidnutý poznáte např. příkazem `ls -l /usr/sbin/pppd`, když ve výpisu nastavení práv (`-rwsr-xr-x`) bude to písmenko s. Když ne, tak jako root můžete suidnout pppd příkazem `chmod u+s /usr/sbin/pppd`.

stopgprs

```
killall pppd
```

/etc/resolv.conf

```
order bind
nameserver 195.146.100.100
nameserver 195.146.100.5
```

DIAL UP

/etc/ppp/options

```
lock
```

/etc/ppp/peers/seznam

```
connect    '/usr/sbin/chat -v -f /etc/ppp/peers/seznam_up'
disconnect '/usr/sbin/chat -v -f /etc/ppp/peers/seznam_down'
/dev/modem
115200
crtscts
modem
default-asyncmap
ms-dns 195.146.100.5
ms-dns 195.146.100.100
defaultroute
noipdefault
ipxcp-accept-local
ipxcp-accept-remote
proxyarp
idle 600
noauth
name seznam
```

Parametr /dev/modem říká, se kterým zařízením se má komunikovat. Já mám svůj modem připojen pomocí sériového kabelu k druhému sériovému portu a proto /dev/modem je u mě symbolický link na /dev/ttyS1 (ve Windows známo jako COM2). Pro jiného providera je nutno nastavit jiné přihlašovací jméno (seznam).

/etc/ppp/peers/seznam_up

```
ABORT         'BUSY'
ABORT         'ERROR'
ABORT         'NO ANSWER'
```

```
ABORT          'NO CARRIER'
ABORT          'NO DIALTONE'
TIMEOUT 60     ''
ATZX3          'OK'
ATDT971101200 'CONNECT'
```

Pro jiného providera je nutno nastavit jiné tel. číslo.

/etc/ppp/peers/seznam_down

```
"" "+++ath"
```

/etc/ppp/pap-secrets

```
"seznam"   *   "seznam"
```

První seznam je přihlašovací jméno (musí souhlasit s name v /etc/ppp/peers/seznam), druhý seznam je heslo k účtu.

/etc/ppp/chap-secrets

```
"seznam"   *   "seznam"
```

Alternativní ověřovací protokol k pap-secrets, provideři používají jeden z nich.

seznam

```
#!/bin/sh
xterm -geom 40x15 -e /usr/sbin/pppd nodetach call seznam
```

Tohle je příkaz, který spuští připojení k internetu v gui. Na konzoli stačí dát přímo příkaz `/usr/sbin/pppd nodetach call seznam`. Pokud vynecháte parametr nodetach, připojení se provede na pozadí. Příkaz si můžete pojmenovat libovolně. Je nutno ho dát do cesty, třeba do /usr/bin nebo do /home/uzivatel/bin. Dále je potřba mu nastavit právo pro spuštění (`chmod a+x seznam`).

Program /usr/sbin/pppd musí být suidnutý, to znamená, že se musí spouštět s právy roota. Že je suidnutý poznáte např. příkazem `ls -l /usr/sbin/pppd`, když ve výpisu nastavení práv (`-rwsr-xr-x`) bude to písmenko s. Když ne, tak jako root můžete suidnout pppd příkazem `chmod u+s /usr/sbin/pppd`.

stopseznam

```
killall pppd
```

/etc/resolv.conf

```
order bind
nameserver 195.146.100.100
nameserver 195.146.100.5
```

Česká klávesnice

Je v Linuxu lehce problematická. V GUI se o ni stará X Server na konzoli jádro, takže pro konzoli se nastavuje jinak a jinde než pro GUI. To ale není problém, na konzoli česká klávesnice obvykle není potřeba a navíc co mám zkušenosti, tak se dobře nastavuje už při instalaci Linuxu, pokud si vyberete, že chcete českou klávesnici. Problém nastane, až když ji budete chtít změnit.

Na konzoli se používá klávesová mapa, která obsahuje rozložení kláves jak pro českou, tak pro anglickou klávesnici a mezi těmito mapami umožňuje přepíná klávesou Pause. Mapa klávesnice se nahrává příkazem loadkeys a dostupné mapy jsou umístěny v /lib/kbd/keymaps/i386/. Na konzoli se musí se změnou klávesnice nahrát občas i

nový font, např. azbuka nebo západo evropské kódování. Fonty se nahrávají příkazem setfont a dostupné fonty jsou umístěny v /lib/kbd/consolefonts/. Takto přímo se to ale normálně nenastavuje. Jak probíhá nastavení je ale hodně závislé na distribuci. Popíši, jak to funguje v RedHatu.

Nastavení fontu a klávesnice při spuštění počítače má na starost skript /etc/init.d/keytable, který se spouští automaticky při startu počítače. Můžete si ho prostudovat a zjistíte, jak přesně to probíhá. Mimo jiné se z toho dozvíte, že konkrétní klávesová mapa a font který se má nahrát je uveden v souboru /etc/sysconfig/keyboard a /etc/sysconfig/i18n. Pokud chcete natrvalo změnit klávesnici a font za jiný, můžete to provést editací těchto dvou souborů. Mnohem jednodušší ale bude, když spustíte utilitku **redhat-config-keyboard**, kde si jen vyberete z nabídky jazyků/klávesnic. Tato utilita provede jak okamžitou změnu (voláním uvedených příkazů), tak i změnu uvedených cfg. souborů. Tato utilita bohužel nastavuje nejen klávesnici pro konzoli, ale i pro GUI, což může být nechtěný efekt.

Nastavení klávesnice v GUI je možno provést příkazem setxkbmap. To ale není obvykle potřeba, klávesnice se nastavuje sama při startu X Serveru a údaje o nastavení se berou z cfg. souboru /etc/X11/XF86Config nebo /etc/X11/XF86Config-4 (máte-li tam oba a máte X Server verze 4.x, tak přednost má ten druhý). Zde je druhý problém, zaprve nějací čeští chytráci pořád šťourali do klávesových map a jejich snahou bylo zbavit se oblíbene mapy czsk a nahradit ji mapou cz podobnou ve Windows, což se jim podařilo, ale setkalo se to s takovým odporem, že linuxové distribuce oblíbené v čechách ji stejně přidávaly. Navíc od XFree verze 4.3 se změnilo pojetí práce s klávesnicemi, což přineslo změnu způsobu jejich konfigurace. Výsledkem je, že skoro co verze XFree86, to jiná česká klávesnice nebo způsob její konfigurace.

Klávesnicová část cfg souboru /etc/X11/XF86Config vypadá nějak takto:

```
  Section "InputDevice"
    Identifier  "Keyboard0"
    Driver      "keyboard"
    Option      "XkbRules"   "xfree86"
    Option      "XkbModel"   "pc105"
    Option      "XkbLayout"  "us,cz_qwerty"
    Option      "XkbOptions" "grp:shift_toggle"
    Option      "XkbCompat"  "group_led"
  EndSection
```

Tohle je podle mě optimální nastavení pro XFree86 4.3 (dnes aktuální) a možná i vyšší. Důležité jsou poslední tři řádky, ty ostatní nechte tak jak máte. Odpovídá to jednořádkovému příkazu:

```
setxkbmap 'us,cz_qwerty' -option 'grp:shift_toggle'
                                 -compat 'group_led'
```

Při pokusech je lepší používat tento příkaz, protože při změně cfg. souboru je potřeba restartovat X Server.

Layout nastavuje jaké se mají používat klávesové mapy. Tohle je novinka v 4.3, před tím bylo možno mít nataženu jen jednu. Do úvahy připadají nejčastěji mapy cz, cz_qwerty a us. Seznam dostupných map je v /usr/X11R6/lib/X11/xkb/symbols/. První uvedená je defaultní.

Options grp: nastavuje způsob přepínání mezi mapami. Představené shift_toggle znamená stisknutí obou shiftů najednou. Další možnosti lze vydedukovat z /usr/X11R6/lib/X11/xkb/symbols/group

Compat zapíná indikaci použité klávesnice pomocí LED diody Scroll Lock na klávesnici. Když se přepnete na druhou mapu, tak se dioda rozsvítí.

Nepříjemné je, že když si takhle pěkně ručně nastavíte klávesnici pro GUI a pak použijete již zmíněnou utilitku redhat-config-keyboard, tak se vám to nastavení přepíše. Tento nepříjemný problém lze obejít již také zmíněným příkazem setxkbmap, který s příslušnými parametry necháte automaticky spouštět po startu GUI. Tím se vlastně nastavení v cfg. souboru obejde. Tohle je, pokud vím, nejjednodušší možnost, jak docílit toho, aby na jednom počítači mohlo být víc uživatelů s tím, že každý má jinou defaultní klávesnici. Různá desktopová prostředí, jako KDE, která podobné věci ošetřují, neberu v úvahu.

Jenom pro úplnost, kdysi dávno (XFree86 3.x) existovala mapa czsk, která v sobě obsahovala několik variant. V cfg. souboru se volala jako "XkbLayout" "czsk(us_cz_qwerty)" a další možné varianty byly sk_us_prog a podobně. Fungovala až do XFree86 4.2, ale již předtím měli někteří lidé snahu ji z XFree86 vyhodit. Tento způsob ovládání klávesnic měl jednu nevýhodu, kombinace klávesnic byly dány napevno v mapě. Takže když chtěl člověk něco neobvyklého, dejme tomu českou klávesnici se španělskou, měl smůlu. No, neměl smůlu, mapy mohl přehazovat pomocí setxkbmap, ale to bylo nepohodlné.

Současný způsob je tedy v tomto ohledu lepší. Ale zas na druhou stranu, u staré mapy mohl člověk dočasně přepínat klávesnice pomocí pravého Altu. To bylo velmi užitečné a pohodlné, při programování člověk potřebuje anglickou klávesnici a když sem tam potřeboval napsat nějaké písmenko s diakritikou, tak ho napsal s Altem stejně jako se píše velké písmenko se Shiftem. To nyní není možné, bohužel.

Nejhorší bylo období, kdy ještě nešlo kombinovat klávesové mapy a přitom nějaký český chytrák nechal z XFree86 czsk vyhodit a místo ní tam dal mapu podobnou té ve Windows. Prostě aktivní blbec. Doufejme že od XFree86 jsou problémy s klávesnicí a přepínání mezi mapami už jen historií.

Stručný seznam rozdílů oproti Windows

- Kódování češtiny je iso-8859-2.
- Názvy souborů rozlišují velikost písmen.
- V cestě k souborům se používají normální lomítka /.
- Disky, cd disky a pod. se připojují a připojují se k adresářům.
- Programy se instalují pomocí správce balíčků.
- Konfigurace se ukládá do konfiguračních souborů.
- Grafické rozhraní je nepovinné, alternativní.
- Téměř všechno se tváří jako soubor.
- Linux staví na textových souborech.
- Přístupová práva rwx pro ugo.
- Selekce a schránka

Pojmy a zkratky

adresář	ve windows známé jako složka
AIX	UNIXový operační systém od IBM
CLI	Command Line Interface, rozhraní příkazové řádky
distribuce (linuxová)	Sada software zahrnující linuxové jádro, operační systém kolem něj vystavěný a sbírka aplikačního software připravená k použití nebo

	instalaci jako jeden funkční celek. Může se vejít na jednu disketu, ale může zabírat i několik CD.
ext2	nativní souborový systém Linuxu
ext3	jako ext2, rozšířený o žurnálovací schopnosti
FAT	Souborový systém. FAT12 diskety; FAT16 DOS a Windows 95; FAT32 Windows 98, ME, NT, 2000 a XP.
FS	File system (souborový systém) nebo Free Software
FSF	Free Software Foundation (Nadace Free Software)
GNU	GNU'S NOT UNIX, software pod GPL licencí
GPL	General Public Licence, licence kterou vydala FSF a kterou používá mnoho linuxového software, včetně jádra. Microsoft se ji obává a nazývá ji rakovinou.
GUI	Graphics User Interface, grafické uživatelské rozhraní
IBM	Velká IT firma, vyrábějící jak hardware, tak software, velmi příznivě nakloněna Linuxu.
IT	informační technologie
jádro	Každý OS má jádro nebo mikrojádro, které zodpovídá za řízení celého počítače. Zde je jím myšleno jádro, jehož vývoj řídí Linus Torvalds.
JFS	Souborový systém, které vyvinulo IBM pro AIX.
kernel	viz. jádro
konzole	Textový rozhraní linuxového jádra
Linus Torvalds	Čelní představitel linuxové komunity, hlavní vývojář jádra.
Linux	Operační systém unixového typu nebo jádro operačního systému Linux.
Microsoft	Velká IT firma známá především svým OS MS Windows, kancelářským balíkem MS Office a výrobou myší. Je k Linuxu velmi nepříznivě nakloněna.
mountování	připojování souborového systému k adresáři
NTFS	Souborový systém používaný ve Windows NT, 2000 a XP.
OS	Operační systém
OSS	Open Source Software
ReiserFS	Alternativní souborový systém Linuxu.
root	Super uživatel, uživatel v Linuxu, který má neomezená práva.
SGI	Nejznámější výrobce grafických pracovních stanic.
Stallman Richard	Čelní představitel FSF a GNU.
terminál	Mnoho významů. Typicky program který zpřístupňuje textové rozhraní v grafickém. Představitelem je např. xterm, který je součástí XFree86. Dříve se také používaly hardwarové terminály, jako hardwarový terminál může také sloužit jeden počítač připojený k jinému. Hardwerové terminály mohou být textové i grafické. V hardwarovém terminálu tak může běžet softwarový.
Windows	Prý nějaký operační systém. Neslyšel jste už o něm někdo něco?

XFree86	Linuxová implementace X Window
XFS	Souborový systém od SGI.
X Server	Jedna ze základních součástí X Window
X Window	Standard UNIXového grafického rozhraní. Viz. XFree86

Evidence změn tohoto dokumentu

2004-07-xx v0.014 (cca 277 kB html text, 360 kB obrázky)

Opět umožněno nastavení dokumentu. Lze nastavit barva a umístění menu, které může být buď vlevo a nebo nahoře. Umístění nahoře je určeno pro velmi malé obrazovky a má drobný problém. Prohlížeč menu nerespektuje (neví o něm), takže při kliknutí na kapitolu (odkaz) se kapitola zobrazí hned navrchu okna prohlížeče, kde už je menu, titulek kapitoly je tak schován pod menu a není vidět. U dostatečně velkých obrazovek tedy doporučuji mít menu umístěno nalevo.

Do sekce základy byla doplněna kapitola Dokumentace.

Do sekce programy byla doplněna kapitola Icewm.

2004-03-12 v0.013 (cca 270 kB html text, 360 kB obrázky)

Uprava kodu z navrcholu.cz, ktera znepristupnovala stranku v IE6.

2004-03-13 v0.012 (cca 267 kB html text, 360 kB obrázky)

Redesign menu a vytvoreni ukazatele pozice v dokumentu.

2004-03-10 v0.011 (cca 264 kB html text, 360 kB obrázky)

Velke upravy v engine, zrusena moznost nastaveni, zjednoduseni skriptu, nekompatibility IE reseny ohackovanim CSS.

Rozsirena a prepracovana kapitola o souborovych systemech v sekci zaklady a rozdlena na dve kapitoly, "Souborove systemy - uvod" a "Souborove systemy - rozdily Linux / Windows"

Kapitola o mountovan prepracovana a presunuta ze sekce zaklady do sekce system. Dobre tam koresponduje s kapitolami druhy souboru a soubory zarizeni. Neni potreba takto tyto veci zvlast vysvetlovat. Obecne informace o mountovani v zakladech zustaly v ramci kapitoly o souborovych systemech.

Provedena revize cele sekce Zaklady.

2004-03-02 v0.010 (cca 210 kB html text, 240 kB obrázky)

Drobna uprava designu.

Oprava mnoha překlepů pomocí sed scriptu, který mi zaslal Hynek Pichi Vychodil, za což mu tímto děkuji.

Doplněna sekce konfigurace o tři kapitoly (mountování, modem a klávesnice).

Do správců souborů doplněn Nautilus.

Do sekce programy doplněn Gkrellm.

2003-10-19 12:15 v0.009 (cca 182 kB html text, 129 kB obrázky)

Uprava designu, aby obešel chyby a byl funkční i v MSIE (předchozí v MSIE extrémně pomalu skroloval).

Upraveno Menu, dokumentu lze nyní nastavit i velikost písma. Nastavení písma i plovoucího menu se pamatuje.

2003-10-02 16:00 v0.008 (cca 182 kB html text, 129 kB obrázky)

Uprava vzhledu

Úprava obsahu, vzhledu a chování menu

Přidána kapitola: Programy / Správci souborů

Úprava kapitoly: Preambule 2

2003-09-13 06:00 v0.007 (cca 163 kB)

Stylistické a gramatické opravy.

Změněna koncepce a cíl dokumentu a tím i předpokládaný rozsah, přidáno mnoho nových sekcí, zatím prázdných.

Protože velikost dokumentu se zvětšila natolik, že jeho editace se stala nepřehlednou, byl rozdělen na mnoho dílčích souborů (po kapitolách), které je potřeba před publikací zpracovat textovým procesorem m4. Výsledná podoba dokumentu (po zpracování) zůstává stejná.

Přidána kapitola: CLI / Příkazová řádka

Přidána kapitola: CLI / Terminál a konzole

Přidána kapitola: CLI / Bash - základy

Přidána kapitola: CLI / Bash - prikazy

Přidána kapitola: CLI / Bash - skripty

Přidána kapitola: CLI / Bash - ulohy

Přidána kapitola: CLI / Bash - specialni znaky

Přidána kapitola: Příkazy / Přehled příkazů

Přidána kapitola: Základy Linuxu / bezpečnost

Kapitoly konfigurace1, instalace1, slabina a instalace2 sloučeny do kapitoly: Kompatibilita

2003-08-08 06:00 v0.006 (cca 73 kB)

Doplnění kapitoly Názvy zařízení

2003-08-08 06:00 v0.005

Přidána kapitola Názvy zařízení

2003-08-08 01:00 v0.004

Upraveno menu aby lepe fungovalo v MSIE.

2003-08-06 18:00 v0.003

Přepracováno menu, rozděleno na interaktivní sekce.

CSS upraveno pro lepší tisk dokumentu.

2003-08-06 09:40 v0.002

Jazykové korekce.

2003-08-05 15:34 v0.001

První (pracovní) veřejná verze.

Co plánuji udělat, dopsat nebo opravit

- Věnovat se mnoha dalším oblastem.
  - FSF, GNU, OSI a neb ideologie
  - Vytvořit systémovou sekci
    - FHS
    - průběh startování Linuxu
    - lokalizace, jak funguje i18n
    - základy sítí
    - dělení disku
    - značení zařízení (hda1, ttyS0, ...) - HOTOVO
    - ovladače a hw
    - Práce s dokumentací, jak a kde ji najít, jak s ní pracovat
  - Vytvořit GUI sekce system/office/multimédia
    - X Server
    - základní konfigurace
    - fonty
    - základní aplikace
    - správci oken
    - desktopová prostředí
  - Vytvořit CLI sekci - HOTOVO

        - příkazový řádek - HOTOVO
        - přehled příkazů - částečně, doplnit X, grafika, foto, hudba
        - popis důležitých příkazů
        - užitečné skriptíky
    - Vytvořit administrační sekci
        - XF86Config, klávesnice, myš, rozlišení a frekvence, ovladače
        - nastavení času
        - vytvoření spouštěcí diskety
        - lilo a grub
        - inittab
        - bash
        - netrc
        - nastavení domácího firewalu
- Doplnit/ověřit informace označené značkou '??'.
- Rozdělit text na víc kratších souborů - HOTOVO
- Text navzájem provázat odkazy.
- Doplnit do textu odkazy na podrobné informace.
- Vytvořit komentovaný seznam kvalitních informačních webů o Linuxu.
- Vytvořit seznam vhodné literatury.
- Upravit text po jazykové a gramatické stránce
- Reload stránky musí nastavit výběr oddílu na výchozí hodnotu.
- Sjednotit terminologii jako přístupová vs. uživatelská práva a pod.
- Dopsat kapitolu o bezpečnosti
- Dopsat kapitolu o správcích souborů