Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Automatické aktualizace a instalace SW v unixové učebně (formou typu cvs)

Filip Borovec

Bakalářská práce

2008

Univerzita Pardubice,
Fakulta elektrotechniky a informatiky
Katedra informačních technologií
Akademický rok: **2007/2008**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Filip BOROVEC**

Studijní program: **B2646 Informační technologie**

Studijní obor: **Informační technologie**

Název tématu: **Automatické aktualizace a instalace SW v unixové učebně (formou typu cvs).**

Zásady pro vypracování:

Teoretická část:
Diskutujte možná řešení automatických aktualizací a instalací SW na jednotlivých PC podle vybraného PC.
Zohledněte časovou nezávislost spouštění aktualizací na různých PC.
Zaměřte se podrobněji na systém nezávislý na správě balíčků (SW), hledejte obecné řešení pro systémy UNIX.
Implementační část:
Vytvořte prostředky pro ukládání celého souborového systému tak, aby se z něj dal aktualizovat jiný počítač do identického stavu podle vzoru.
Pokuste se systém implementovat ve spolupráci se správcem učebny FEI v učebně ÚEI.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

Kolektiv autorů: LINUX dokumentační projekt, Computer Press 1998 (3. aktualizované vydání), ISBN 80-7226-761-2

Shah, S.: Administrace systému Linux – podrobný průvodce začínajícího administrátora, Grada Praha 2002, ISBN 80-7169-586-6

dokumentace a manuálové stránky programů CVS, rsync, SSH

Vedoucí bakalářské práce: **Mgr. Tomáš Hudec**
Ústav elektrotechniky a informatiky

Datum zadání bakalářské práce: **30. listopadu 2007**

Termín odevzdání bakalářské práce: **16. května 2008**

Univerzita Pardubice Česká republika

L.S.

doc. Ing. Simeon Karamazov, Dr.
děkan

V Pardubicích dne 29. dubna 2008

SOUHRN

Tato práce se zabývá problematikou synchronizace stanic a aktualizace softwaru na klientských počítačích s operačním systémem unixového typu. Má za cíl navrhnout takové řešení, které by mělo být nezávislé na systému správy softwarových balíčků. Na základě teoretických východisek jsou navržena a implementována řešení, která automatizují aktualizaci softwaru a synchronizaci klientských stanic v síti.

KLÍČOVÁ SLOVA

Unix, Linux, aktualizace, synchronizace, CVS, rsync

TITLE

Automatic updates and installations SW in the Unix classroom (cvs-like form)

ABSTRACT

This work deals with a synchronizations of a client stations and updates sw on the client computers with the Unix-like operation system. It aims to project solution which should be independent on the software package management system. Based on the theoretical starting points it is designed and implemented solution which automate updating software and synchronizing client stations on the network.

KEYWORDS

Unix, Linux, update, synchronization, CVS, rsync

Obsah

Úvod

V počítačových sítích s velkým počtem uživatelských stanic často vyvstává problém, jak tyto stanice udržovat jednoduše v aktualizovaném stavu. Tedy jak jednoduše nainstalovat, odinstalovat, popřípadě překonfigurovat nějaký software na všech těchto počítačích. Pokud je k síti připojeno více než 4 – 6 uživatelských stanic, je již pro správce této sítě časově nepřijatelné provádět veškeré změny několikrát na každém počítači zvlášť. Často je také kvůli snadnější správě celé sítě potřeba udržovat všechny uživatelské stanice v identickém stavu.

Tato práce má za cíl vyřešit uvedený problém obecně pro všechny počítačové sítě, na kterých běží operační systém unixového typu. Tedy například libovolný Linux, Solaris, BSD, IRIX, GNU HURD a další.

Při řešení je třeba zohlednit časovou nezávislost aktualizací prováděných na různých klientských stanicích tak, aby aktualizace nepřerušovala uživatele v jejich práci. Je tedy třeba aktualizovat různé uživatelské stanice v různou dobu, například při odhlášení uživatele ze systému.

Je zřejmé, že pokud má být tento systém automatických instalací a aktualizací softwaru použitelný na jakémkoli systému unixového typu, nebude možné využít správce balíčků (SW), který je na různých unixových systémech implementován různým způsobem. Naopak bude třeba vytvořit takový systém, který bude ukládat na server celý souborový systém z nějakého vzorového uživatelského počítače takovým způsobem, aby se z něj dal rychle aktualizovat jiný počítač do identického stavu podle této vzorové stanice.

Je také dobré, aby takový systém aktualizací měl možnost tzv. verzování. Tedy aby mohl administrátor sítě ukládat na server různě nakonfigurované verze systému s různým nainstalovaným softwarem a pokud se mu například nebude z nějakého důvodu líbit poslední konfigurace systému, kterou provedl, aby se mohl jednoduše vrátit k nějaké starší verzi uživatelského operačního systému.

1 Možná řešení automatických aktualizací

Aby bylo možné vytvořit systém, který aktualizuje počítače na síti podle vybraného vzorového počítače, je nejdříve nutné vybrat pro tyto účely co možná nejlepší nástroj pro synchronizaci souborů.

Tento nástroj by měl mít v ideálním případě tyto vlastnosti:

- Měl by umět synchronizovat adresáře po síti po zabezpečeném protokolu, protože při aktualizaci stanice se mohou přenášet po síti citlivá data.
- Synchronizace by měla probíhat rychle a neměla by příliš zatěžovat síť. Rozhodně je časově nepřijatelné, aby se při každé aktualizaci přenášel celý souborový systém nebo celé diskové oddíly například pomocí programu dd. Naopak nejlepší řešení by bylo takové, kde by se při aktualizaci po síti přenášely jen ty části souborů, které se změnily.
- Měl by umožňovat verzování souborů. A to buď přímo jako jednu z funkcí nebo tak, aby bylo možné tuto funkci jednoduše doprogramovat.
- Tento nástroj musí být schopen sesynchronizovat celý kořenový souborový systém, ne například jen nějaký adresář ve stromové struktuře s uživatelskými typy souborů.
- Nástroj nesmí být závislý na specifické vlastnosti nějakého systému (např. Linux). Musí fungovat stejně pod všemi operačními systémy unixového typu.

Nyní popíši vlastnosti jednotlivých synchronizačních a zálohovacích nástrojů, které připadají v úvahu.

1.1 CVS

CVS je zkratka z anglického Concurrent Version system. Již z názvu je tedy poznat, že tento nástroj umožňuje verzování adresářů. CVS byl vyvinut Dickem Grunem v 80. letech 20. století. Je to systém navržený pro verzování zdrojových kódů tak, aby mohli vývojáři vzájemně vzdáleně spolupracovat na jednom projektu a měli k dispozici vždy nejnovější verzi projektu.

CVS používá klient-server architekturu, přičemž na serveru uchovává záznam, který obsahuje veškeré změny vybrané množiny souborů. Na serveru se též uchovává aktuální verze vybraného adresáře (v tzv. modulech) a jeho historie (repozitář jednotlivých modulů).

Administrátor, který provede nějaké změny v klientském operačním systému, může jednoduše zapsat novější verzi systému na server. CVS umožňuje zápis do CVS repozitářů ochránit heslem tak, aby je nemohl měnit neprivilegovaný uživatel. Uživatelský počítač, který vyžaduje aktualizaci, se připojí k serveru a pomocí CVS příkazu update si zaktualizuje vybraný adresář (v tomto případě kořenový adresář) tak, aby měl nejnovější verzi systému.

CVS využívá delta-kompresi pro efektivní ukládání různých verzí stejných souborů. Tím šetří místo na serverovém disku.

Díky těmto vlastnostem by byl tento nástroj ideální pro účely aktualizace stanic na síti. Je to ovšem nástroj primárně určený pro verzování zdrojových kódů, z čehož plyne pro naše účely několik podstatných omezení:

- Omezená podpora textových souborů s kódováním Unicode a názvů souborů v jiné znakové sadě než ASCII.
- CVS neumožňuje zahrnout do svých repozitářů symbolické odkazy (symlinky[1]).

1 Textový odkaz na soubor nebo adresář (více v seznamu zkratek a pojmů).

- CVS pracuje s tvrdými odkazy (hardlinky[2]) jako s normálními soubory. Nezachová tedy strukturu hardlinků tak, jak je na vzorovém počítači pro aktualizaci.

Tato omezení by se při použití CVS musela velice složitě obejít. Nepodpora symbolických odkazů by se dala obejít na vzorovém počítači například vytvořením souboru, kde by bylo zapsáno jméno a umístění symlinku v adresářové struktuře a kam se odkazuje. Z tohoto souboru by se pak na aktualizovaném počítači musely symlinky obnovit.

Nepodpora hardlinků už by se obcházela poměrně obtížně. Na vzorovém počítači by se musely najít všechny soubory, které jsou hardlinky, zapsat je do nějaké struktury, z které by se obnovovaly na uživatelských počítačích.

Nepodpora některých znakových sad už se obejít v podstatě nedá.

1.2 CVSup

CVSup vychází ze CVS a přizpůsobuje ho pro synchronizování jakýchkoli typů souborů. Odstraňuje tedy pro naše účely všechny hlavní nevýhody CVS. Umí pracovat se symbolickými odkazy a umí též zachovávat tvrdé odkazy.

Je to ovšem projekt, který postrádá dokumentaci a ze stránek autora je patrné, že tento program funguje pouze pod systémy BSD (FreeBSD, NetBSD a OpenBSD). Nefunguje tedy pod všemi systémy unixového typu, není tedy možné pomocí tohoto nástroje splnit zadání práce.

2 Záznam v adresáři s odkazem na stejný i-uzel, na který odkazuje jiný soubor (více v seznamu zkratek a pojmů).

1.3 Git

Git je software určený pro správu velkých softwarových projektů. Vytvořil ho samotný Linus Torvalds a původně byl určen jako podpůrný nástroj pro vývoj linuxového jádra.

Git je navržen pro údržbu a vývoj velkých softwarových projektů. To znamená, že má vynikající podporu verzování zvoleného adresáře (v našem případě kořenového adresáře). Umožňuje efektivně spravovat mnoho větví, vytvořených z původní verze. Umožňuje též šifrovanou autentifikaci uživatelů pro přístup do historie repozitářů.

Je ovšem trochu pomalejší při synchronizaci (než v CVS nebo rsync). Musí totiž při určování toho, jestli se soubor změnil nebo ne, procházet celou větev historie. Ale hlavně neposkytuje metody, které jsou nutné pro ukládání celého kořenového souborového systému. Například neumožňuje snadno vyloučit soubory nebo adresáře z aktualizace. Také není nikde v dokumentaci zmíněno, jak je to se zachováním symbolických a tvrdých odkazů podle vzoru.

1.4 Rsync

Rsync je součástí většiny linuxových distribucí. Je to aplikace pro unixové systémy a je určen pro synchronizování souborů a adresářů. To buď z jednoho adresáře do druhého lokálně na jednom počítači nebo po síti na jiný počítač. Ještě před samotnou synchronizací najde rsync pomocí algoritmu Quick check soubory, které se změnily (změnila se jejich velikost nebo datum poslední změny) a které tedy mají být synchronizovány. Pokud se změnila pouze práva pro přístup k souboru, tak ta jsou aplikována přímo na cílový soubor.

Pro synchronizaci změněných souborů se používá algoritmus delta-transfer, aby se minimalizovalo množství dat přenášených po síti. Díky tomuto algoritmu se po síti přenáší jen změněné bloky dat ze zdrojového souboru do existujícího cílového souboru. Rsync může také použít kompresi zlib pro přenášená data, a tím dále zmenšit zátěž sítě.

Vzdálená synchronizace může probíhat přes jakýkoli udaný vzdálený shell (např. zabezpečené ssh nebo pomocí rsh) nebo pomocí démona rsync. Démon rsync je program, který naslouchá na nějakém portu na serveru a je nastaven tak, aby dokázal přes síť pomocí rsyncu synchronizovat soubory a adresáře do/z nastavených modulů. Tyto moduly jsou vlastně adresáře s nastaveným chováním. Například je možno povolit nebo zakázat zápis, nastavit uživatele, pod kterým se v modulu zapisují nebo čtou soubory. Je možno také použít chroot na zvolený adresář tak, aby se uživatel používající démona nemohl dostat z tohoto adresáře a zneužít toho. Démon rsync využívá nezabezpečený protokol, takže je dobré ho tunelovat například pomocí ssh, aby byla při případném odposlechu sítě znemožněna rekonstrukce souborů, které po síti putují při synchronizaci.

Rsync umí pomocí několika z mnoha voleb zachovat v cílové lokalitě jak symbolické odkazy, tak tvrdé odkazy podle vzoru. Též umí zachovat práva k souborům, vlastnictví souboru (pokud rsync běží pod právy roota), rozšířené atributy u souborů a access listy (ACL) od verze 3.0. Pomocí volby „--exclude“ umí vynechat ze synchronizace zvolené soubory nebo adresáře.

Démon rsync umožňuje autentizaci uživatelů k jednotlivým modulům. Bohužel v aktuální verzi 3.0.2 používá slabou 128 bitovou MD4 hash. To by mělo být řešeno v dalších verzích. Další nevýhodou rsyncu pro naše účely je to, že nepodporuje verzování.

1.5 Rdiff-backup

Rdiff-backup je software pro posixové systémy určený pro zálohování jednoho adresáře do jiného adresáře. Umožňuje jak lokální zálohy tak zálohování přes síť. Rdiff-backup využívá knihovny rsync (librsync verze 0.9.7 nebo novější).

Má tedy podobné vlastnosti jako rsync:

- Umí zachovat strukturu symbolických i tvrdých odkazů, přesně tak, jak jsou ve vzorovém adresáři.
- Umí zachovat práva k souborům, jejich vlastnictví (musí běžet s právy roota), čas poslední úpravy, rozšířené atributy a přístupové seznamy (ACL's).
- Používá delta encoding pro přenos souborů, takže se přenáší jen rozdíly, nikoliv celé soubory.

Oproti rsyncu přidává navíc ještě podporu verzování. Rdiff se snaží zkombinovat pro účely zálohování nejlepší vlastnosti plných a inkrementálních záloh. První provedená záloha se zkopíruje celá. Při druhé a další záloze se vlastně synchronizuje zdrojový adresář s předchozí zálohou, přičemž je do speciálního podadresáře cílového adresáře uložen reverzní rozdíl aktuální zálohy oproti předchozí záloze, takže je možno obnovit starší zálohu. Rdiff také umožňuje tyto inkrementy komprimovat pomocí gzip.

Obsluha (tvorba, mazání, obnova) těchto inkrementů probíhá plně v režii Rdiff--backupu. Při obnově se zadává datum, kdy byla provedena záloha, ze které chceme náš adresář obnovit. Je možné též zadat stáří zálohy, ze které chceme obnovu provést. Když se zadá „now“, tak se provede obnova z nejnovější zálohy.

Nevýhodou ovšem je, že při obnově starší zálohy, musí Rdiff spočítat postupně všechny diference všech změněných souborů oproti souborům v aktuální záloze. Některé inkrementy mohou být navíc ještě komprimovány a je nutno je dekomprimovat. Pokud se tedy chceme při aktualizaci operačního systému vrátit hodně hluboko do historie, bude taková aktualizace trvat poměrně dlouho.

Další nevýhodou Rdiffu je, že neposkytuje podobnou službu jako je démon rsync. Vzdálené spojení se realizuje pomocí roury, která je otevřena Rdiffem na klientské straně a pomocí ssh shellu (nebo jiného shellu) je na straně serveru spuštěn Rdiff--backup v serverovém módu, na který se tato roura připojí.

1.6 Rsnapshot

Rsnapshot je program pro periodické zálohování celých filesystémů. Je napsán celý v Perlu a funguje pod všemi moderními unixovými systémy. V konfiguračním souboru Rsnapshotu se nastaví počet záloh, které se mají udržovat a jejich perioda. Zálohování pak lze spouštět automaticky pomocí Cronu nebo ručně. Rsnapshot se nechal inspirovat článkem *Easy Automated Snapshot-Style Backups with Linux and Rsync* od Mika Rubela [1].

Je založen na rsyncu (může využívat služeb démona rsync) a na jednoduché, a svým způsobem geniální, myšlence používat tvrdé odkazy (hardlinky) pro plné zálohy, které ovšem zabírají na disku pouze rozdíl oproti předchozí záloze. To je provedeno tak, že se vždy před samotnou zálohou vytvoří nový adresář a v něm se vytvoří stejná struktura souborů a adresářů jako je v předchozí záloze. Soubory v tomto novém adresáři jsou ale hardlinky na soubory z předchozí zálohy, takže zabírají pouze několik bajtů nutných pro vytvoření odkazu. Poté je spuštěn rsync, který synchronizuje zvolený adresář s tímto nově vytvořeným. Pokud narazí na soubor, který se změnil, tak nejprve provede unlink tohoto souboru (odstraní hardlink) a poté ho synchronizuje se zdrojovým souborem. To, že rsync při synchronizaci změněných souborů nejprve provádí unlink cílového souboru, je klíčová vlastnost, díky které se neovlivní soubory z předchozí zálohy, které se v té aktuální změnily.

Výsledek této operace je tedy takový, že změněné soubory k sobě nemají žádný vztah typu hardlink a soubory, které se oproti předchozí záloze nezměnily, jsou hardlinky na jeden fyzický soubor na disku.

Jak již bylo zmíněno výše, pokud se oproti předchozí záloze změní pouze práva pro přístup k souboru, rsync tato práva aplikuje přímo na cílový soubor, bez provedení unlinku. To je tedy omezení Rsnapshotu, protože pokud se u nějakého souboru změní pouze práva a aktualizuje se celý souborový systém, pak je tato změna práv u tohoto souboru aplikována i na všechny předchozí zálohy. Tím dochází k nekonzistenci v historii aktualizací.

2 Výběr vhodného nástroje pro aktualizaci

Jako první nástroj, o kterém jsem uvažoval, že použiji k automatické aktualizaci a instalaci softwaru na klientských stanicích, byl CVS. Pomocí tohoto nástoje by se dal vytvořit systém s velmi propracovanou podporou verzování. Podpora synchronizace po počítačové síti je u CVS rovněž na velmi vysoké úrovni. Nicméně je to nástroj určený pro správu zdrojových kódu a jeho nepodpora symbolických odkazů a omezená podpora kódování znaků ve jménech souborů mě přinutila poohlédnout se po jiném nástroji.

CVSup odstraňuje všechny vlastnosti CVS, kvůli kterým je pro tuto práci nepoužitelný. Je ale velká škoda, že tento projekt postrádá dokumentaci a je určen pouze pro operační systémy typu BSD.

Git disponuje možná ještě propracovanějším systémem verzování než CVS. Ovšem, podobně jako CVS, je určen pro správu softwarových projektů. Pro účely automatických aktualizací celého operačního systému i s nainstalovanými programy to tedy není nejvhodnější nástroj.

Pro výběr nástroje, pomocí kterého budou automatické aktualizace SW implementovány, zůstává trojice: rsync, Rdiff-backup a Rsnapshot. Přičemž Rdiff využívá hlavní knihovnu rsyncu a Rsnapshot je na rsyncu přímo postaven. Lze tedy očekávat podobnou efektivitu využití přenosového pásma na síti, podobnou zátěž procesoru při synchronizaci i podobnou rychlost přenášení souborů po síti. Rozdíly jsou v možnostech nastavení a poskytované funkcionalitě.

Všechny tyto tři nástroje mají své výhody i nevýhody. Rsnapshot implementuje velice jednoduchý, efektivní a rychlý způsob verzování, ovšem ne zcela funkční v případech, kdy se změní pouze přístupová práva k souboru. To je zásadní nedostatek.

Rdiff-backup implementuje verzování zajímavým způsobem pomocí ukládání reverzních rozdílů oproti předchozí verzi. To je efektivní, co se týče obsazeného prostoru na disku, ovšem za cenu náročnější obnovy historie.

Nakonec jsem se rozhodl vybrat rsync z těchto důvodů:

- Má velké možnosti nastavení svého chování a přehlednou a kvalitní dokumentaci.
- Umožňuje synchronizaci pomocí démona, který běží na serveru. To je klíčová funkce pro zajištění bezpečnosti serveru. Není totiž potřeba nikde v systému klientské stanice uchovávat klíč, pomocí kterého je možno dostat shell na serveru (popsáno v kapitole 4.3) tak, jako by to bylo třeba, pokud by se používal Rdiff-backup.

Rsync sám o sobě nijak nezajišťuje verzování jednotlivých aktualizací, nicméně to lze dopsat pomocí shellových skriptů.

3 Postup Implementace

Nyní, když už je vybrán nástroj rsync, můžeme začít řešit otázky implementace. Praktická implementace bude provedena pomocí skriptů[3] bash (Bourne Again Shell). Bude rozdělena na dva hlavní skripty. První skript se bude starat o synchronizaci z klientské stanice na server, tedy aktualizace dat na serveru. Tento skript bude spouštět ručně administrátor sítě po provedených změnách v konfiguraci na dané stanici. Druhý skript se bude starat o synchronizaci ze serveru na stanici, tedy o aktualizaci klientské stanice. Tento skript bude spouštěn automaticky.

Nejprve je ovšem potřeba vyřešit jak správně nastavit rsync pro účely automatických aktualizací klientských stanic na síti.

3.1 Volby programu rsync

Nejprve je nutné v manuálové stránce rsync(1) najít, jaké volby použít tak, aby se dal operační systém zrekonstruovat zpět do stejného stavu z kopie kořenového adresáře vytvořené pomocí programu rsync. K tomu je nutné zajistit:

1. aby se správně kopírovaly symbolické odkazy, tedy aby se nenásledovala cesta kam odkazují,
2. aby se zkopíroval celý kořenový adresář rekurzivně i s podadresáři,
3. aby se zachovala přístupová práva k souborům a adresářům,
4. aby se zachoval čas poslední změny souborů a adresářů,
5. aby se zachovaly informace o vlastníkovi souboru a jeho skupině,
6. aby se zachovaly pevné odkazy (hardlinky), tedy pokud máme dva hardlinky odkazující na jeden fyzický soubor, nesmí se stát, aby se zkopírovaly jako dva fyzické soubory,

3 Posloupnost příkazů, které jsou vykonávány interpretem skriptu (v našem případě shellem bash). Skrip umožňuje větvení programu podle podmínek, vytváření cyklů, ...

7. aby se, pokud je to potřeba, zachovala rozšířená práva k souboru, popřípadě access listy, pokud je používáme.

Pro zajištění bodu 1 – 5 slouží volby „a“ (archiv mode) a „E“ (zachování spustitelnosti souborů). Pro zajištění bodu 6 volba „H“ a pro zajištění bodu 7 volby „X“ (zachování rozšířených atributů) a „A“ (zachování access listů). Zachování access listů podporuje rsync od verze 3.0.

3.2 Neaktualizované soubory a adresáře

Jelikož jako zdrojový adresář pro synchronizaci rsyncem směrem na server bude nastaven kořenový adresář („/“), je třeba pomocí volby rsyncu „--exclude“ správně nastavit soubory a adresáře, které budou z této synchronizace vynechány. Ze synchronizace by měly být vynechány následující soubory a adresáře:

- různé speciální soubory (např. soubory zařízení),
- soubory, které jsou unikátní pro daný počítač (např. soubor /etc/hostname),
- virtuální souborové systémy (např. /proc, který se vůbec nenachází na disku),
- síťové souborové systémy, jelikož ty se fyzicky nenachází na klientské stanici, tudíž nejsou předmětem aktualizace,
- další soubory a adresáře, u kterých si administrátor sítě nepřeje aby byly aktualizovány (diskutováno v kapitole 6.2).

Co se týče unikátních souborů, tak ty je třeba vyjmenovat explicitně v konfiguračním souboru skriptu, který provádí aktualizaci. Rovněž soubory a adresáře, které si administrátor přeje vynechat z aktualizací, je třeba uvést v konfiguračním souboru.

Síťové a virtuální souborové systémy, tedy adresáře a soubory, které se fyzicky nenachází na lokálním disku klientské stanice, je možno z aktualizace vynechávat automaticky pomocí výstupu příkazu „mount“. Příkaz mount poskytuje textový výstup ve formátu několika polí oddělených mezerou pro každý připojený souborový systém.

První tři pole vždy znamenají (v pořadí v jakém jsou uvedena):

1. připojené zařízení,
2. „on“ - konstantní textový řetězec (pro přehlednost výstupu),
3. adresář, do kterého je zařízení připojeno,

Z výstupu příkazu mount je tedy možné pomocí prvního pole rozpoznat, jestli se jedná o lokální, síťový nebo virtuální souborový systém. Pokud se jedná o lokální souborový systém, nachází se v prvním poli vždy řetězec „/dev/“. Je tedy třeba zpracovat všechny řádky z výstupu příkazu mount a do seznamu adresářů vyloučených z aktualizace je třeba přidat třetí pole (přípojný adresář) u řádků, v jejichž prvním poli se nevyskytuje řetězec „/dev/“.

Seznam souborů a adresářů, které byly vyloučeny ze synchronizace na server, je nutno nějakým způsobem předat skriptu, který provádí samotnou aktualizaci klientských stanic, tedy synchronizaci ze serveru na klientský počítač. Tento skript totiž musí tento seznam také vyloučit ze synchronizace, aby se na klientském počítači nesmazaly soubory a adresáře, které jsou v tomto seznamu. Jako způsob předání těchto parametrů jsem zvolil vytvoření textového souboru, do kterého se zapíše seznam souborů a adresářů vyloučených z aktualizace přímo ve tvaru, v jakém se zadávají rsyncu jako parametr. Tedy například „--exclude=/proc --exclude=/dev“. Tento soubor je vytvořen na klientské stanici ještě před provedením synchronizace na server. Tím pádem je při synchronizaci rsyncem nakopírován na server, odkud si ho skript pro aktualizaci ze serveru přečte a použije jako parametr rsyncu.

Z aktualizace klientské stanice jsou též vynechány adresáře, do kterých jsou na této stanici připojeny virtuální a síťové souborové systémy. To je provedeno opět pomocí výstupu příkazu mount.

3.3 Řešení bezpečného připojení na server

Je zřejmé, že pokud chceme ukládat celý souborový systém na server a poté s tímto uloženým systémem synchronizovat klientské stanice, je nutné najít vyhovující způsob připojení na server. Z hlediska bezpečnosti je nutné dodržet dvě základní priority:

1. Ochránit server. Tedy vytvořit takový systém, který neumožňuje získání kontroly nad serverem, na kterém budou aktualizace uloženy.

2. Ochránit klientské stanice a data, která jsou na klientských stanicích.

Podle prvního bodu je třeba dbát na to, aby vytvořený systém aktualizací nevytvářel bezpečnostní díru na aktualizačním serveru. Co se týká ochrany klientských stanic, je nutné znemožnit neautorizovaný zápis aktualizace na server, protože se tato aktualizace automaticky přenese na všechny klientské stanice na síti. Je též dobré, aby byl přenos dat po síti imunní proti odposlechu sítě, tedy posílat data po šifrovaném kanálu, protože na disku klientské stanice mohou být citlivá data v chráněných souborech.

3.3.1 Připojení pomocí ssh

První řešení toho, jak se bezpečně připojit na server, které jsem navrhl, bylo přímo pomocí rsyncu s využitím volby „-e“. Pomocí této volby jsem specifikoval, aby se rsync připojoval k serveru pomocí ssh. Jako cestu k aktualizacím na server pro rsync se tedy uvádí klasické „root@server:/adresář/s/aktualizacemi/“. To zajistí šifrovaný způsob připojení k serveru a znemožní případný odposlech dat, která putují po síti.

Aby správce (kterým nemusí být administrátor serveru) nezadával heslo roota, je třeba použít vygenerovaný pár klíčů pro přístup na server přes ssh, veřejný klíč umístit do authorized_keys v adresáři „/root/.ssh“ na serveru a soukromý klíč někam na klientskou stanici. Pro účely vytvoření nové aktualizace administrátorem a přenosu této aktualizace na server je možno tento klíč ochránit heslem, které bude znát administrátor.

Ovšem pro účely samotné aktualizace klientských stanic na sítí už klíč nelze ochránit heslem, jinak by aktualizace nebyly automatické. Na čtení tohoto klíče může mít práva pouze root (na dané klientské stanici). To je ovšem nedostatečná ochrana, protože kterýkoli uživatel jedné z klientských stanic by mohl nabootovat počítač například z liveCD nebo z flash disku se svým operačním systémem a klíč si přečíst. Tím získá shell roota na serveru, tedy plnou kontrolu nad serverem.

Tento způsob připojení na server, kdy rsync potřebuje přístup k shellu roota na serveru znamená zcela zásadní bezpečnostní díru. To lze řešit pomocí démona rsync.

3.3.2 Připojení pomocí démona rsync

Démon rsync je služba běžící na serveru. Pomocí této služby jdou synchronizovat adresáře na serveru s počítači na síti. Tyto adresáře jsou nastaveny v tzv. modulech v konfiguračním souboru démona rsync. K ovládání démona rsync a k samotné síťové synchronizaci není nutné, aby měl rsync spuštěný na klientském počítači k dispozici shell roota na serveru. Démon rsync totiž naslouchá na serveru na daném portu a rsync spuštěný na klientském počítači se k němu přes tento port připojí. Synchronizace pomocí démona rsync je ve výchozím módu realizována nezabezpečeným protokolem.

V globální části konfiguračního souboru démona rsync je pro naše účely dobré nastavit port, na kterém bude démon naslouchat a tzv. „PID file“. Port je dobré nastavit na nějaký jiný než standardní port démona rsync (873), tak aby mohl na serve-

ru běžet démon rsync určený pro jiné účely než aktualizace stanic na síti. PID file je soubor, do kterého se bude zapisovat process id spuštěného démona rsync. Toto id je užitečné pro ukončení konkrétní instance démona v době, kdy již není potřeba.

Pak je třeba v konfiguračním souboru nastavit jednotlivé moduly[4], které budou využívány. V modulech je pro účely aktualizací klientských stanic dobré nastavit:

- path - cestu k adresáři pro synchronizaci,
- uid – jméno nebo id uživatele pod kterým synchronizace poběží na serveru, v našem případě „root“,
- use chroot – nastavit path jako root adresář, aby se nemohl uživatel používající modul dostat někam výš v adresářové struktuře (např. pomocí odkazů),
- read only – povolení nebo nepovolení práva zápisu do modulu,
- host allow – seznam počítačů, kterým je povoleno používat modul.

To, že démon rsync přenáší po síti data nezabezpečeným protokolem, lze jednoduše vyřešit tunelováním portů pomocí ssh. S výhodou lze též vytvořit dva konfigurační soubory pro démona rsync, které se budou lišit v tom, jestli je povolen nebo zakázán zápis do modulů a v uvedených portech, na kterých démon naslouchá. Tím získáme možnost využívat dvě instance démona rsync . Jednu pro vytváření aktualizací na serveru (ta s povoleným zápisem do modulů). Tu bude vzdáleně spouštět pouze administrátor, takže je možné spouštění tohoto démona ochránit heslem. Druhá instance nebude mít povolen zápis do modulů a bude sloužit pro aktualizaci klientských stanic ze serveru. Ta bude vzdáleně spouštěna bez zadávání hesla, aby byl proces aktualizace automatický.

Jestliže je systém navržen tak, že se k démonu rsync stanice vždy připojují přes ssh tunel, lze v konfiguračních souborech démona nastavit host allow pouze pro localhost, tak aby démona nemohl využívat žádný počítač na síti přímo, ale pouze přes tunel. V souboru authorized_keys (na serveru), kde jsou uloženy veřejné klíče, je třeba před klíč z dvojice klíčů, pomocí kterých tunelujeme spojení na server, uvést řetě-

4 Adresáře na serveru s nastaveným chováním pro synchronizaci pomocí programu rsync.

zec command=" ", tak aby případný útočník pomocí tohoto klíče nezískal shell na serveru. Na serveru se po připojení klienta pomocí tohoto klíče provede pouze příkaz uvedený v uvozovkách a klient nedostane shell na serveru.

3.4 Verzování

Jelikož rsync sám o sobě neumožňuje verzovat provedené synchronizace a v případě potřeby se vracet do historie, je třeba tuto funkci dopsat např. pomocí skriptu bash. Při psaní tohoto skriptu jsem se nechal inspirovat článkem *Easy Automated Snapshot-Style Backups with Linux and Rsync* od Mika Rubela [1].

Rozhodl jsem se tedy vytvořit podobný systém verzování, tak jak je realizován ve výše popisovaném programu Rsnapshot. Tento systém verzování je totiž zcela transparentní pro skripty, které provádí aktualizace stanic. Navíc je velice rychlý. Malé zpomalení oproti systému bez verzování lze zaznamenat pouze při vytváření aktualizace na serveru. Na serveru se totiž ještě před samotnou aktualizací musí nalinkovat soubory z předchozí aktualizace. Tento skript ovšem spouští pouze administrátor po jeho zásahu do systému. U skriptu pro aktualizaci klientských stanic ze serveru, které se spouští automaticky poměrně často, není zpomalení žádné.

Vytvoření nové aktualizace na serveru tedy probíhá následovně: Ještě před samotným synchronizováním klientské stanice na server se na serveru vzdáleně spustí procedura, která „připraví“ adresář, do kterého bude tato synchronizace provedena. Poté se vzdáleně spustí démon rsync pomocí kterého je synchronizace provedena a následně se ukončí.

Ona příprava adresáře probíhá tak, že se na serveru v adresáři určeném pro aktualizace konkrétního systému vytvoří podadresář nazvaný podle aktuálního data a času ve formátu RRRR-MM-DD-HH:MM (tedy rok-měsíc-den-hodina:minuta). Do tohoto adresáře je nalinkován obsah adresáře s předchozí aktualizací, pokud existuje. Na adresář z předchozí aktualizace ukazuje symbolický odkaz „aktualni“. Pokud tento symbolický odkaz neexistuje, znamená to, že žádná předchozí aktualizace neexistuje, nic se tedy nelinkuje. Poté je smazán symbolický odkaz „aktualni“, pokud existuje, a

je vytvořen nový symbolický odkaz „aktualni“, který ukazuje již na nový adresář. Nalinkování probíhá pomocí příkazu „cp -al“, kde volba „-l“ říká, že se zdrojové soubory nemají kopírovat, ale místo toho se mají vytvořit pevné odkazy (hardlinky).

Rsync poté synchronizuje obsah klientské stanice na server do adresáře „aktualni“. Nejprve odstraní hardlinky u změněných souborů a zapíše změny. Nezměněné soubory ponechá nalinkovány na předchozí aktualizace. Tím je docíleno toho, že nová aktualizace zabírá na disku serveru pouze rozdíl oproti předchozí.

Ovšem, jak již bylo zmíněno, pokud se změní pouze přístupová práva k souboru, tak je rsync aplikuje přímo na cílový soubor a ovlivní tedy i předchozí aktualizace. Proto jsem udělal skript, který vytváří aktualizaci na serveru tak, že je tento způsob verzování (nalinkování na předchozí aktualizaci) provede pouze pokud je skript spuštěn s volbou „-l“ (jako link). Pokud tato volba zadána není, adresář s předchozí zálohou se nelinkuje. Místo toho se zkopíruje („cp -a“). Tím se tedy vytvoří další plná záloha systému.

Návrat do historie je velice jednoduchý. Administrátor pouze přepíše cíl symbolického odkazu „aktualni“ na nějaký starší adresář a klientské stanice se poté automaticky aktualizují na tuto starší aktualizaci. Mazat aktualizace je možné standardními prostředky pro mazání adresářů.

Pokud tedy administrátor udělá takový zásah do systému, který změní u některých souborů pouze práva, neměl by pro vytvoření aktualizace používat volbu „-l“. Po návratu do historie by se totiž mohlo stát, že nějaký program přestane fungovat, protože nemá některá potřebná práva k některým souborům. Mohlo by se také stát, že bude mít uživatel na klientské stanici více práv, než je nutné.

3.5 Bezpečné ovládání serveru

Nyní vyvstává potřeba spouštět a ukončovat na serveru vzdáleně démona rsync. Též je potřeba si ze serveru ještě před spuštěním aktualizace přečíst seznam souborů a adresářů vyjmutých z aktualizace a také je potřeba vzdáleně spustit nalinkování

souborů z předchozí aktualizace do adresáře určeného pro vytvoření nové aktualizace na serveru. Je tedy potřeba napsat skript, pomocí kterého lze na serveru bezpečně provádět tyto úkony. Tedy jakýsi skript pro vzdálené ovládání serveru, který bude ale umožňovat provádět pouze několik operací potřebných k funkci systému automatických aktualizací.

Jedna možnost, jak zajistit takové ovládání serveru, je pro každou akci, kterou je potřeba provést na serveru, napsat skript a spouštět ho pomocí ssh. Tedy přesněji pomocí připojení na server přes ssh s využitím dvojice klíčů. Hesla pro připojení na server pomocí ssh místo klíčů používat nemůžeme, protože by při aktualizaci bylo vyžadováno heslo od uživatele a aktualizace by tím pádem nebyla automatická. Před veřejným klíčem na serveru v souboru authorized_keys bude tedy napsáno: `command="/cesta/k/skriptu"`. Při připojení se na server pomocí soukromého protějšku takového klíče nedostane klient shell serveru, ale bude pouze proveden příkaz, který je uveden v uvozovkách. Při tomto způsobu ovládání serveru bude v souboru authorized_keys spousta klíčů a stane se značně nepřehledným.

Lepší způsob je vytvořit jeden skript a tomu nějakým způsobem předat parametry. Jelikož při připojení pomocí ssh nelze předávat standardní parametry skriptu, který je uvedem v příkazu command u veřejného klíče v authorized_keys, je nutno tyto parametry číst přímo ve skriptu na standardním vstupu pomocí příkazu read. Skript pro ovládání serveru tedy spustíme vzdáleně pomocí ssh spojení a přes rouru pošleme na standardní vstup tohoto skriptu parametry například takto: `echo "param1 param2" | ssh -i klic root@serveru`

Samozřejmě je potřeba ošetřit vstupní hodnoty parametrů tak, aby tento skript prováděl skutečně jen to, k čemu je určen. Tedy aby případný útočník tomuto skriptu nemohl podsunout parametry, díky kterým získá ze serveru citlivé údaje nebo díky kterým by na serveru provedl nějakou neočekávanou akci.

3.6 Metodika testování

Při postupné implementaci jednotlivých idejí popsaných výše je nutné vždy testovat, jestli konkrétní pracovní verze vytvořeného systému spolehlivě funguje. Též je nutné otestovat finální verzi tohoto produktu. K tomu je potřeba stanovit si nějakou testovací metodiku. Tato testovací metodika musí určit:

- zda systém po aktualizaci funguje,
- zda po aktualizaci fungují všechny přiinstalované a změněné součásti systému, stejně tak i aktualizovaný software,
- zda je v aktualizovaném systému uvolněno místo na disku po odinstalovaných součástech,
- zda je aktualizovaný systém konzistentní, resp. ve stejném stavu jako vzorový.

Testování by přitom mělo systém dobře prověřit. To znamená, že by se měly testovat různé situace, kdy se aktualizuje různý typ softwaru.

Ideálními podmínkami pro testování by bylo mít tři počítače. Jeden jako vzorová stanice, druhý jako server a třetí jako stanice, která bude aktualizována. Já jsem zpočátku testoval na dvou počítačích, přičemž jeden fungoval jako server a druhý počítač, který má dva disky, fungoval jako vzorová stanice i jako stanice aktualizovaná. To šlo zajistit tak, že když počítač fungoval jako vzorová stanice, tak se zaváděl systém z prvního disku (zavaděč byl též na prvním disku), a když fungoval počítač jako stanice aktualizovaná, tak se zaváděl systém z druhého disku (se zavaděčem na druhém disku). Měnilo se tedy v BIOSu pořadí disků při bootování.

Samotné testování probíhalo vždy následujícím způsobem: Nejprve jsem na vzorové stanici provedl určitou změnu v softwarové konfiguraci počítače. Poté jsem zkontroloval konzistenci systému na tomto vzorovém počítači. To je možné zabezpečit přepočítáním kontrolních součtů všech balíčků a všech konfiguračních souborů v systému. Například na systémech odvozených od Debianu pomocí příkazu

„debsums -as“, na RPM-based Linuxech (Fedora, Suse, Mandriva, ...) pomocí příkazu „rpm -Va“. Chybový výstup z tohoto programu jsem uložil do souboru. Poté jsem provedl aktualizaci a restartoval počítač.

Na aktualizovaném systému jsem následně zkontroloval funkčnost všech aktualizovaných částí systému. Zkontroloval jsem též, zda se smazal z disku odinstalovaný software. A provedl jsem kontrolu konzistence systému stejně tak jako na vzorové stanici. Poté jsem porovnal obsah souborů s chybovými výstupy (kontroly konzistence) obou systémů. Pokud se nelišily, aktualizace proběhla v pořádku.

Jak je zmíněno výše, rozsah aktualizace by měl dobře prověřit systém aktualizací. Na finální verzi systému automatických aktualizací byly otestovány následující situace:

- instalace, odinstalace a konfigurace softwarových balíčků,
- odebírání a přidávání uživatelů v systému,
- instalace nové verze programu rsync,
- aktualizace operačního systému Ubuntu Linux na operační systém Suse Linux.

Ve všech situacích byl aktualizovaný systém plně funkční podle zvolených kritérií a připraven na další automatické aktualizace. Dokonce i v posledním extrémním případě aktualizace úplně jiného systému, než byl nainstalován, se aktualizace povedla a po restartu byl nový systém funkční a konzistentní. Restart musel být v posledním případě proveden ručně, protože starý systém, který byl v paměti, již nenašel na disku potřebné programy pro restart.

4 Architektura systému

Tato kapitola popisuje architekturu vytvořeného systému pro automatické aktualizování stanic na síti. Tento systém se skládá ze tří hlavních skriptů. Každý skript má pro snadnější nastavování ještě svůj konfigurační skript. Všechny skripty, konfigurační soubory a ssh klíče jsou uloženy v adresáři „/etc/sync“. Pokud se administrátor rozhodne tyto skripty umístit jinam, je potřeba ve skriptech tuto cestu správně nastavit nebo použít interaktivní konfigurátor (popsáno v kapitole 9).

Všechny skripty jsou bohatě komentovány, takže v této kapitole nebudou popsány veškeré detaily týkající se konfigurace a způsobu implementace, ale jen princip funkce. Jak nakonfigurovat skripty popisuje kapitola 9.

4.1 Serverová část

Serverovou část reprezentuje skript „ovladani.sh“ se svým konfiguračním skriptem „ovladani.conf.sh“. Tento skript (ovladani.sh) je spouštěn výhradně vzdáleně pomocí ssh, kdy je před veřejnou částí klíče v souboru authorized_keys uveden příkaz `command="/etc/sync/ovladani.sh"`, popřípadě je uveden příkaz command="/etc/sync/ovladani.sh admin". Parametr admin udává, že skript spouští administrátor sítě pomocí klíče chráněného heslem. Pomocí tohoto parametru je tedy rozeznávána úroveň oprávnění. Vlastníkem tohoto skriptu je uživatel root, přičemž nikdo kromě roota nemá ke skriptu žádná práva.

Skript ovladani.sh po svém spuštění nejprve spustí svůj konfigurační skript, ve kterém jsou nastaveny proměnné udávající:

- adresář určený pro aktualizace na serveru,
- adresář se serverovými skripty a konfiguračními soubory pro démona rsync,
- adresář s klientskými skripty na klientských stanicích,

- názvy konfiguračních souborů démona rsync pro aktualizaci stanic a pro vytváření aktualizací na serveru,
- porty, na kterých naslouchá démon rsync pro aktualizaci stanic a pro vytváření aktualizací na serveru,
- názvy souborů, ve kterých je uložen počet klientů připojených k démonům pro aktualizaci stanic a pro vytváření aktualizací na serveru,
- názvy souborů s process id démonů pro aktualizaci stanic a pro vytváření aktualizací na serveru,
- cesta k programu rsync.

Po načtení konfigurace zjistí skript úroveň oprávnění pomocí parametru „admin“ a podle toho zvolí jakou konfiguraci používat. Poté jsou přečteny parametry ze standardního vstupu. Na standardní vstup jsou parametry zasílány pomocí roury na ssh spojení z klientských stanic. Podle těchto parametrů je provedena jedna z pěti akcí.

4.1.1 Spouštění a ukončování démona

První dvě jsou akce „daemonstart“ a „daemonkill“. Jak už název napovídá, jedná se o spouštění a ukončování démona rsync. Jelikož je spouštěn pouze jeden démon pro účely vytvoření nové aktualizace na serveru a hlavně pouze jeden démon určený pro aktualizaci stanic na síti a k tomuto démonu se může připojovat najednou více stanic, není možné, aby každá stanice spustila démona a pak ho ukončila. Mohlo by se totiž stát, že jedna stanice dokončí aktualizaci a vypne démona jiné stanici, která ještě aktualizuje.

Tato situace je řešena souborem, ve kterém je zapsán počet klientů využívajících služeb démona rsync (jeden soubor pro každého démona). Když skriptu přijde požadavek na spuštění démona, je podle oprávnění spuštěn příslušný démon pouze v případě, že je v souboru nula. Poté se inkrementuje obsah souboru. Pokud je v souboru číslo vyšší než nula, znamená to že je již démon spuštěn. Podobně při požadavku na ukončení démona je dekrementován obsah souboru a pokud je v souboru číslo nula,

je démon ukončen pomocí příkazu „kill“ s daným číslem procesu zapsaném v souboru udaném v konfiguraci démona rsync. Jelikož je takové spouštění a ukončování démonů z hlediska operačního systému tzv. kritická sekce, tak se po spuštění démona testuje, jestli je opravdu spuštěn a poslouchá na zvoleném portu. Pokud ne, je spuštěn znovu a tím je zajištěno, že démon bude zapnut vždy, když to je potřeba.

4.1.2 Příprava na vytvoření aktualizace

Další akce, kterou provádí skript ovladani.sh je akce „prepare“. Tuto akci může spustit pouze administrátor. Akce „prepare“ vytvoří nový adresář pro aktualizaci a nalinkuje do něho soubory z předchozí aktualizace (podrobně popsáno v kapitole 4.4). Této akci je předán parametr s názvem modulu démona rsync, pro který se má adresář připravit. Tento parametr je použit jako název podadresáře v adresáři určeném pro aktualizace. Protože tento parametr je poslán z klientské stanice, je nutné ho zkontrolovat, jestli se opravdu jedná o regulérně zapsaný adresář. Jde hlavně o to, aby případný útočník nemohl pomocí použití dvou teček a lomítek přepsat důležité soubory na serveru. Parametr je kontrolován na výskyt těchto znaků „./\ *?[“.

4.1.3 Získání seznamu neaktualizovaných souborů

Předposlední akcí je „getexclude“. Tato akce přijímá jako parametr ze standardního vstupu podobně jako akce „prepare“ název modulu. Tento parametr je použit stejně jako v akci „prepare“ a je kontrolován stejnou procedurou. Akce vypisuje na standardní výstup obsah souboru „exclude.txt“, v němž je seznam souborů a adresářů vyloučených z aktualizace. Tento seznam si následně přečte skript pro aktualizaci klientských stanic.

4.1.4 Příprava adresáře a spuštění démona

Poslední akcí je „prepare_and_daemonstart“. Tu může spustit pouze administrátor. Akce provádí pouze postupně akci „prepare“ a poté „daemonstart“. Její smysl je v tom, aby se nemuselo při vytváření aktualizace dvakrát za sebou navazovat ssh spojení a aby administrátor nemusel dvakrát za sebou zadávat heslo ke klíči pro toto spojení.

4.1.5 Konfigurace démona rsync

Na serveru se též nachází konfigurační soubory „rsyncd_read.conf“ a „rsyncd_write.conf“. Jsou to konfigurační souboru démona rsync určeného pro aktualizaci stanic a pro vytváření aktualizací na serveru. V těchto souborech se musí uvést stejný port a stejná cesta k „pid file“ (soubor s process id démona) jako je v „ovladani.conf.sh“. Po té se musí vyplnit údaje o modulu:

- path – cesta k adresáři pro aktualizace konkrétního systému, musí být stejná jako v souboru „ovladani.conf.sh“ doplněná ještě o podadresář se jménem stejným jako modul a další podadresář se jménem „aktualni“,
- uid – musí být nastaveno na „root“,
- use chroot – mělo by být nastaveno na „yes“ kvůli bezpečnosti,
- hosts allow – spojení je tunelováno přes ssh, takže je dobré kvůli bezpečnosti vyplnit pouze localhost (pokud nefunguje, tak adresy rozhraní, z kterých přichází požadavky na démona),
- read only – v souboru „rsyncd_read.conf“ nastaveno na „yes“, protože při aktualizaci stačí z modulu pouze číst, a v souboru „rsyncd_write.conf“ nastaveno na „no“.

4.2 Klientská část

Na klientských počítačích jsou standardně v adresáři „/etc/sync“ uloženy skripty pro vytváření aktualizací na serveru, samotnou aktualizaci klientských stanic a jejich konfigurační skripty. V tomto adresáři jsou též soukromé klíče potřebně pro připojení k serveru.

4.2.1 Skript pro vytváření aktualizace

Tento skript s názvem „vzor2server.sh“ a s jeho konfiguračním skriptem „vzor2server.conf.sh“ je určen pro administrátora, který vytváří na serveru novou aktualizaci. Skript umožňuje přijmout volbu „-l“, která zapíná verzování pomocí hardlinků. Pokud tato volba není uvedena, je verzování provedeno jako plná záloha (popsáno v kapitole 4.4).

Po spuštění tento skript spustí konfigurační skript, ve kterém jsou nastaveny například:

- volby rsyncu včetně souborů a adresářů vyloučených z aktualizace,
- název nebo adresa serveru,
- název používaného modulu démona rsyncu,
- cesta ke klíčům pro ssh spojení,
- port, na kterém naslouchá démon rsync pro účely vytváření aktualizací na serveru,
- lokální port pro ssh tunelování.

Potom jsou k seznamu souborů a adresářů vyloučených z aktualizace přidány adresáře, které jsou automaticky generovány z příkazu mount (popsáno v kapitole 4.2). Celý tento seznam je zapsán do souboru (standardně „/etc/sync/exclude.txt“), pro budoucí použití skriptem pro aktualizace klientských stanic. Následně je vzdáleně spuštěna na serveru akce „prepare_and_daemonstart“ v serverovém skriptu

ovladani.sh. Jako parametr je serverovému skriptu předán název používaného modulu démona rsync. Jako další parametr je nebo není podle volby „-l“ předán parametr „link“. Tedy například:

```
echo "prepare_and_daemonstart ${modul} link" | ssh -i $serverKey root@$kam.
```

Poté je pomocí ssh tunelován daný port na server. Před zahájením synchronizace pomocí rsync je nutno, pomocí cyklu while, čekat dokud nebude tunel ustaven. Následně je pomocí rsyncu provedena synchronizace s daným modulem démona rsync na serveru. Nakonec je ukončen ssh tunel a démon rsync na serveru.

4.2.2 Skript pro aktualizaci klientských stanic

Tento skript s názvem „server2stanice.sh“ a s jeho konfiguračním skriptem „server2stanice.conf.sh“ je určen pro automatickou aktualizaci klientské stanice na síti. K tomu aby byla aktualizace opravdu automatická, je potřeba umístit na tento skript symbolický odkaz do adresáře, kde se nachází skripty prováděné při přechodu do runlevelu 0 (většinou /etc/rc0.d), tedy před vypnutím počítače. Je možno umístit odkaz i do adresáře „rc6.d“, tedy spouštět aktualizaci před resrtartem. Nicméně když uživatel restartuje počítač, tak to znamená, že chce dále pracovat. A aktualizace by ho zbytečně zdržovala. Když uživatel počítač vypíná, znamená to, že již dále nechce pracovat a aktualizace ho již nebude zdržovat.

Symbolický odkaz na skript je ovšem třeba umístit tak, aby se před jeho spuštěním neukončily některé důležité služby, které by mohly být potřeba pro jeho funkci.

Po spuštění tento skript spustí konfigurační skript, ve kterém jsou nastaveny podobně jako v skriptu „vzor2server.conf.sh“ například:

- volby rsyncu včetně souborů a adresářů vyloučených z aktualizace,
- název nebo adresa serveru,

- název používaného modulu démona rsync,
- cesta ke klíčům pro ssh spojení,
- port, na kterém naslouchá démon rsync pro účely aktualizací na serveru,
- lokální port pro ssh tunelování.

Poté je testována dostupnost serveru. Pokud server není dostupný, je skript ukončen. Pak je smazán soubor „finished“, pokud existuje. Tento soubor funguje jako příznak, jestli se aktualizace dokončila. Následně je vzdáleně pomocí ssh a akce „getexclude“ serverového skriptu „ovladani.sh“ přečten ze serveru seznam souborů a adresářů vyloučených z aktualizace. Ten je přidán k seznamu z konfiguračního souboru. Poté je spuštěn na serveru vzdáleně démon rsync, ustaven tunel podobně jako ve skriptu „vzor2server.sh“. Následně je provedena aktualizace pomocí rsyncu a následně je ukončen ssh tunel a démon rsync na serveru. Nakonec je znovu vytvořen soubor „finished“, což signalizuje, že se aktualizace dokončila.

Klientská část obsahuje ještě skript „server2stanice_afterstart.sh“, na který se musí vytvořit symbolický odkaz v adresáři, kde se nachází skripty spouštěné po startu počítače (např. /etc/rc2.d). Většinou funguje, když ho umístíme jako první v pořadí. Tento skript pouze spouští skript „server2stanice.sh“ pokud nenajde soubor „finished“. Odkaz na tento skript můžeme umístit i do adresáře „/etc/rcS.d“ jako poslední v pořadí, ovšem potom by byl správce zdržován synchronizací při vstupu do záchranného single-režimu.

Tento mechanizmus je zde pro případ, kdy dojde k násilnému vypnutí počítače během aktualizace. V tomto případě se nevytvoří soubor „finished“ a aktualizace je po startu počítače spuštěna znovu.

5 Na co si dát pozor

Tato kapitola má za cíl popsat a zdůraznit některé záležitosti, na které by si měl dát administrátor pozor při konfiguraci skriptů a při vytváření aktualizací.

5.1 Vytváření aktualizace

První věc, kterou je třeba zmínit, je to, že po spuštění skriptu „vzor2server.sh“ probíhá vytváření aktualizace ze vzorové stanice, na které je skript spuštěn, a probíhá za běhu této stanice. To znamená, že při vytváření aktualizace by neměly běžet na této stanici žádné úlohy, které mění nastavení systému nebo programů v tomto systému. Mohlo by totiž dojít k nekonzistenci, kdy na serveru budou některé soubory odrážející staré nastavení systému a některé již modifikované odrážející nové nastavení systému. Nejlepší je, pokud běží pouze systém a skript pro vytvoření aktualizace.

Další věc, na kterou by si měl dát administrátor při vytváření aktualizace pozor, je použití volby „-l“, pokud dělá v systému takové změny, které ovlivní u některých souborů pouze přístupová práva. Tato volba by totiž zapříčinila změnu těchto práv i v předchozích aktualizacích (popsáno v kapitole 4.4).

5.2 Jaké soubory neaktualizovat

Další otázka, kterou je třeba pečlivě zvážit, je, jaké soubory a adresáře vynechat z aktualizace. Z aktualizace se automaticky vynechávají virtuální a síťové souborové systémy. Nicméně je třeba nakonfigurovat vynechání i dalších souborů a adresářů. Například soubor „/etc/hostname“, který je na každém počítači unikátní. Pokud nemají všechny stanice na síti přesně stejným způsobem rozděleny disky na oddíly nebo pokud je v systému použito pro označování oddílu UUID namísto jména zařízení (např. /dev/sda6), je nutné vynechat z aktualizace soubory „/etc/fstab“, „/etc/mtab“ a konfigurační soubor zavaděče, například „/boot/grub/menu.lst“. Pokud

bychom to neudělali, aktualizovaný systém by se již nemusel zavést. Jména těchto souborů se mohou lišit na různých systémech, například na BSD je místo souboru „fstab“ soubor „vfstab“.

Je třeba mít také na paměti, že aktualizace stanic se provádí pomocí programu rsync se zapnutou volbou „--delete“. Je tedy třeba určitě alespoň ve skriptu „server2stanice.conf.sh“ nastavit, aby se z aktualizace vynechal adresář, kam systém automaticky připojuje flash disky a podobné zařízení (např. adresář /media). Pokud bychom toto neudělali a uživatel by si zapomněl flash disk v počítači při jeho vypínání, byl by jeho obsah „aktualizován“, tedy smazán. Pokud je v tomto adresáři podadresář, který chceme aktualizovat, je možno ho do aktualizce zahrnout pomocí volby „--include“. Tedy například „--exclude=/media --include=/media/sda6“.

5.3 Aktualizace programu rsync

Rsync je základním stavebním kamenem celého systému. Pokud se administrátor rozhodne aktualizovat program rsync, je třeba, aby zajistil stejné umístění nového rsyncu v adresářové struktuře. Pokud to nezajistí, je potřeba zároveň s touto změnou změnit i proměnnou „rsync_path“ v konfiguračních skriptech, které se též aktualizují. Také je potřeba zjistit, jestli je nový rsync zpětně kompatibilní s dosud používanou verzí.

6 Selhání systému

Systém automatických aktualizací může selhat v případě, že administrátor udělá při vytváření aktualizace nebo při konfiguraci skriptů nějakou z chyb popsaných v kapitole 6. Další možností, kdy tento systém nebude klientské stanice aktualizovat, je situace, kdy se uživateli na klientské stanici nějakým způsobem „povede“ smazat program rsync nebo aktualizační skripty, k těm by ovšem na klientské stanici měl mít právo přístupu pouze uživatel root.

Reálné nebezpečí hrozí, pokud je aktualizace v jejím průběhu násilně přerušena, například kvůli výpadku elektrické energie nebo kvůli hardwarovému vypnutí počítače uživatelem. Pokud k takovému přerušení dojde, bude aktualizace spuštěna znovu při zapnutí počítače (popsáno v kapitole 5.2.2) a systém se uvede do konzistentního stavu. Pokud ovšem aktualizace mění důležité systémové soubory, může nastat i taková situace, že systém po násilném vypnutí již nenastartuje. Příkladem takovéto extrémní situace může být vypnutí počítače ve chvíli, kdy se aktualizuje jádro operačního systému a staré jádro je již smazané a nové ještě na disku není zapsáno.

7 Instalace systému na čistý počítač

Tento systém automatických aktualizací se dá využít i pro rychlou instalaci operačního systému na počítač, kde ještě žádný není.

Stačí nabootovat nějaký funkční systém unixového typu (např. z liveCD nebo ze sítě). V tomto systému rozdělit disk, pokud ještě není rozdělen. Pak vytvořit na nějakém diskovém oddílu souborový systém, na kterém se dá provozovat operační systém, který se bude instalovat. Tento diskový oddíl připojit do nějakého adresáře. Poté je třeba nastavit konfigurační soubor „server2stanice.conf.sh“ tak, aby byl cíl aktualizace shodný s adresářem, do kterého jsme připojili nově vytvořený diskový oddíl. Samozřejmě je nutné správně nastavit i adresu serveru a název modulu démona rsync, který chceme použít. Potom ve skriptu „server2stanice.sh“ nastavíme cestu ke konfiguračnímu souboru a skript spustíme. Provede se aktualizace do zvoleného adresáře.

Poté je třeba dodat soubory, které se v aktualizaci nenacházejí (byly vyloučeny z aktualizace). Tento krok nemusíme provést, pokud ve skriptu, který vytváří aktualizace, nastavíme, že nechceme vyloučit žádné soubory z aktualizace. Tyto soubory je možné vyloučit až ve skriptu, který aktualizuje klientské stanice.

Nakonec je třeba do MBR disku nainstalovat zavaděč tak, aby využíval konfiguračního souboru na nově vytvořeném diskovém oddílu, a upravit patřičně soubor „fstab“. Následně restartovat počítač a nechat zavést systém z disku počítače.

8 Rychlá konfigurace systému

Pro administrátory, kteří chtějí tento systém rychle zprovoznit na nějaké počítačové síti bez předchozího studia konfiguračních souborů, je tu interaktivní konfigurační skript „config.sh“. Tento skript stačí spustit na serveru, ze kterého se budou klientské stanice aktualizovat. V průběhu provádění skriptu je potřeba odpovědět na několik otázek spjatých s konfigurací systému automatických aktualizací. U většiny otázek je předvoleno nějaké výchozí nastavení, které stačí potvrdit stisknutím klávesy Enter. Po skončení tohoto skriptu je třeba podle instrukcí zkopírovat na klientské stanice kořenového adresáře jeden z adresářů, který tento skript vytvořil.

Pomocí tohoto skriptu je možno kompletně nakonfigurovat server i pro více různých aktualizací různých typů operačních systémů. Tedy přidat více modulů démona rsync. Pokud je potřeba aktualizovat na síti různé počítače různými aktualizacemi, je třeba na těchto stanicích změnit v konfiguračním souboru „server2stanice.conf.sh“ název modulu démona rsync v proměnné „modul“.

Před použitím tohoto konfiguračního skriptu a zprovozněním systému automatických aktualizací klientských stanic doporučuji přečíst si kapitolu 6.

9 Závěr

Všechny cíle stanovené v zadání práce se podařilo splnit. Tedy vytvořit systém, který umožňuje rychle aktualizovat jakýkoli systém unixového typu na klientských stanicích na síti. Tato aktualizace je přitom pro uživatele zcela transparentní a nijak ho neomezuje. Na síti mohou navíc existovat skupiny stanic, které se aktualizují různými aktualizacemi.

Podařilo se též efektivním způsobem implementovat verzování jednotlivých operačních systému a snadný návrat do historie vytvořených aktualizací.

Systém nepředstavuje bezpečnostní riziko pro aktualizační server ani pro uživatele klientských stanic. Umožňuje vytvoření nové aktualizace na serveru pouze privilegovaným uživatelům.

Vytvořený systém byl úspěšně testován na systémech Linux (Ubuntu a Suse) a na systému FreeBSD. Byl též úspěšně otestován v učebně FEI ve spolupráci se správcem učebny.

Seznam zkratek a pojmů

ASCII – American Standard Code for Information Interchange, kódová tabulka definující znakovou sadu obsahující anglické abecedy, a jiné znaky používané v informatice.

BASH – Bourne-again shell, unixový shell.

BIOS – Basic Input-Output system, základní programové vybavení osobního počítače, BIOS vytváří základní vrstvu pro vyšší programy.

bootování – Zavedení, tedy spuštění operačního systému.

CVS – Concurrent Version system, systém pro správu verzí projektu.

dekrementace – Zmenšení o nějakou (většinou jednotkovou) velikost.

démon – Program typicky běžící na pozadí, který poskytuje nějakou službu.

hardlink – Pevný (tvrdý) odkaz, je to odkaz na fyzická data na záznamovém zařízení (např. pevném disku). Na systémech unixového typu může být vytvořen příkazem „ln“.

inkrementace – Zvětšení o nějaký (většinou jednotkový) přírůstek.

klient-server – Síťová architektura, kde server poskytuje služby, které na vyžádání „konzumuje“ klient.

kritická sekce – Ta část kódu procesu nebo vlákna procesu, která operuje nad sdílenými daty a hrozí, že paralelně může jiný proces nebo vlákno procesu operovat nad stejnými daty. Důsledkem může být nekonzistence dat.

MBR – Master Boot Record, je to první sektor na záznamovém zařízení a nachází se zde kód zodpovědný za zavedení systému.

RPM – Red Hat Package Manager, správce balíčků (softwarových).

RPM-based Linux – Označení pro distribuci linuxu používající balíčkovací systém RPM.

rsh – Remote shell, program, který umožňuje spouštění programů jako jiný uživatel a na jiném počítači.

shell – Označení počítačového programu, který čte příkazy z terminálu nebo ze souboru (tzv. shell scriptu) a spouští je.

ssh – Secure shell, síťový protokol umožňující výměnu dat mezi dvěma počítači po zabezpečeném kanálu, často používaný pro přihlášení a provádění příkazů na vzdáleném počítači.

symlink – Symbolický odkaz, je to speciální typ souboru, který obsahuje textový odkaz na jiný soubor nebo adresář ve formě absolutní nebo relativní cesty. Na systémech unixového typu může být vytvořen příkazem „ln -s“.

UUID – Universally Unique Identifier, unikátní identifikátor pro různé objekty (např. diskové oddíly).

Použitá literatura

1. SHAH, Steve. *Administrace systému Linux: podrobný průvodce začínajícího administrátora*. Praha: Grada, 2002. 533 s. ISBN 80-7169-586-6.
2. Kolektiv autorů. *Linux dokumentační projekt*. 3. aktualiz. vyd: Computer Press, 1998. 1174 s. ISBN 80-7226-114-2.
3. RUBEL, Mike. *Easy Automated Snapshot-Style Backups with Linux and Rsync* [online]. 2004 [cit. 2008-05-06]. Dostupný z WWW: <http://www.mikerubel.org/computers/rsync_snapshots/>.
4. ŽÁČEK, Kamin. *Úvod do skriptování v BASH* [online]. c2000 [cit. 2008-05-06]. Dostupný z WWW: <http://docs.linux.cz/programming/interpreted/bashdoc-1.4/index.html>.
5. BÁRTA, Milan. *Pokročilé zálohování s Rsync*. Root.cz [online]. 2007 [cit. 2008-05-06]. Dostupný z WWW: <http://www.root.cz/clanky/pokrocile-zalohovani-s-rsync/>.
6. ŠŤASTNÝ, Jakub. *Úvod do skriptování v Linuxu*. *Root.cz* [online]. 2007 [cit. 2008-05-06]. Dostupný z WWW: <http://www.root.cz/clanky/uvod-do-skriptovani-v-linuxu/>.
7. Chovanec, Luděk. *Tao zálohování s pevnými odkazy*. Root.cz [online]. 2002 [cit. 2008-05-06]. Dostupný z WWW: <http://www.root.cz/clanky/tao-zalohovani-s-pevnymi-odkazy/>.
8. *Maximum RPM* [online]. c2000 [cit. 2008-05-06]. Dostupný z WWW: <http://www.rpm.org/max-rpm-snapshot/index.html>.
9. JELEN, Milan. *UNIX V*. Praha: Grada, 1995. 242 s. ISBN 80-7169-113-5.
10. *MD5 sums of Installed Debian Packages* [online]. 2007 [cit. 2008-05-06]. Dostupný z WWW: <http://www.linuxhaxor.net/2007/12/18/debsums-md5-sums-of-installed-debian-packages/>.
11. *Concurrent Versions System*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Concurrent_Versions_System>.
12. *Rsync*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Rsync>.
13. *Unison (file synchronizer)*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Unison_(file_synchronizer)>.
14. *GNU*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/GNU>.

15. *Git (software)*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Git_(software)>.

16. *List of delta encoding software*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/List_of_delta_encoding_software>.

17. *PowerFolder*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/PowerFolder>.

18. *Rdiff-backup* [online]. 2006 , 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://www.nongnu.org/rdiff-backup/index.html>.

19. *Rsnapshot* [online]. 2003 , 2007 [cit. 2008-05-06]. Dostupný z WWW: <http://www.rsnapshot.org>.

20. *Rsync* [online]. 2002 , 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://samba.anu.edu.au/ftp/rsync/rsync.html>.

21. *Remote Shell.* Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Remote_Shell>.

22. *Universally Unique Identifier.* Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/UUID>.

23. POLSTRA, John. *CVSup* [online]. c1997 , 2002 [cit. 2008-05-06]. Dostupný z WWW: <http://www.cvsup.org/>.

24. *ASCII.* Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/ASCII>.

25. *Symbolic link.* Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Symbolic_link>.

26. *Klient-server.* Wikipedie [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Klient-server>.

27. *Hard link*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Hard_link>.

28. *Secure shell.* Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Secure_shell>.

29. *Remote shell.* Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Remote_shell>.

30. *Shell.* Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Shell>.

31. *BASH.* Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Bash>.

32. *Daemon (computer software)*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Daemon_(computer_software)>.

33. BIOS. Wikipedie [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/BIOS>.

34. *Boot sector*. Wikipedie [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Boot_sector>.

35. *RPM Package Manager*. Wikipedie [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/RPM_Package_Manager>.

36. MICHL, Vladimír. *Linux a vlákna*. Linux.cz [online]. 1998 [cit. 2008-05-06]. Dostupný z WWW: <http://www.linux.cz/noviny/1998-0809/clanek11.html>.

37. *Universally Unique Identifier*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/UUID>.

38. *Master Boot record*. Wikipedia [online]. 2008 [cit. 2008-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Master_boot_record>.