TEKNIIKKA JA LIIKENNE

Tietotekniikka

Ohjelmistotekniikka

INSINÖÖRITYÖ

VARMUUSKOPIOINTIJÄRJESTELMÄN SUUNNITTELU JA KÄYTTÖÖNOTTO

Työn tekijä: Daniel Suni
Työn ohjaajat: Risto Virkkala
Auvo Häkkinen

Työ hyväksytty: 10.11. 2010

Auvo Häkkinen
yliopettaja

ALKULAUSE

Tämä insinöörityö tehtiin EfiCode Oy:lle osana yrityksen sisäistä laadunparantamisprosessia. Työnantajan puolesta Risto Virkkala toimi ohjaajana. Lisäksi Juha Lehtonen ja Pauli Manninen antoivat arvokasta palautetta. Sam Grönblom antoi myös tukea git:iin liittyvissä asioissa. Ammattikorkeakoulun puolesta Auvo Häkkinen toimi työn valvojana. Haluan kiittää kaikkia projektiin osallistuneita.

Helsingissä 31.10.2010

Daniel Suni

Metropolia

TIIVISTELMÄ

Työn tekijä: Daniel Suni

Työn nimi: Varmuuskopiointijärjestelmän suunnittelu ja käyttöönotto

Päivämäärä: 31.10.2010 **Sivumäärä:** 44 s. + 1 liite

Koulutusohjelma: Tietotekniikka

Suuntautumisvaihtoehto: Ohjelmistotekniikka

Työn ohjaaja: yliopettaja Auvo Häkkinen

Työn ohjaaja: toimitusjohtaja Risto Virkkala

Insinöörityön aiheena oli korvata EfiCoden vanha varmuuskopiointijärjestelmä, joka ei enää täyttänyt kaikkia tarpeita uudella. Tämän tulisi sekä täyttää nykyiset tarpeet että turvata seuraavien vuosien odotettu kasvu.

Työn tutkimusvaiheessa kartoitettiin millaisia eri ratkaisuja on jo olemassa ja vertailtiin niitä keskenään, jotta pystyttiin muodostamaan käsitys siitä, mikä ratkaisu olisi EfiCodelle sopivin. Vertailun tärkein kriteeri oli tiedonsiirtotehokkuus, koska jotkut varmuuskopioitavista palvelimista eivät sijaitse samassa maassa kuin varmuuskopiopalvelin, eli data on siirrettävä Internetin yli. Tämä oli myös se osa-alue, jolla edellisellä ratkaisulla oli ollut suurimmat vaikeudet. Muita kriteerejä, joita käytettiin olivat laajennettavuus, helppokäyttöisyys ja arkistointimahdollisuudet.

Dirvish ja BackupPC katsottiin olevan lupaavimmat EfiCoden vapaan ohjelmiston kulttuuriin sopivat ehdokkaat. Näitä tutkittiin tarkemmin ja BackupPC implementoitiin asentamalla se omalle palvelimelleen ja integroimalla se EfiCoden infrastruktuuriin.

Kun sopiva sovellus oli löytynyt seuraava askel oli työn implementointivaihe. Kyseinen sovellus asennettiin ja integroitiin EfiCoden järjestelmään. Tähän sisältyi sekä yhteistyö varmuuskopioitavien palvelimien ylläpitäjien kanssa että sovelluksen adaptointia kyseisten järjestelmien ongelmakohtiin, kuten tietokannat ja versionhallintajärjestelmät. Näitä ongelmakohtia varten kirjoitettiin skripti, jonka voi asentaa jokaiselle varmuuskopioitavalle palvelimelle ja paikallisesti muutamaa muuttujaa muuttamalla konfiguroida kyseiselle ympäristölle sopivaksi. Vaikka itse skriptin kirjoittaminen ja testaaminen oli työlästä lopputuloksena oli kuitenkin ratkaisu, joka on helposti muokattavissa uusille palvelimille.

Uusi varmuuskopiointijärjestelmä toimii toivotulla tavalla, jättäen paljon varaa uusien asiakkaiden lisäämiselle. Se on helppokäyttöinen ja sillä on enemmän ominaisuuksia kuin vanhalla järjestelmällä. Kaikkein tärkein on kuitenkin se, että se käyttää huomattavasti vähemmän kaistanleveyttä. Tämän ansiosta kookkaatkaan varmuuskopiot Internetin yli eivät enää tuota ongelmia.

Avainsanat: varmuuskopiointi, tietokantojen varmuuskopiointi, BackupPC, Dirvish

Metropolia

ABSTRACT

Name: Daniel Suni

Title: The Planning And Deployment of a Backup System

Date: 31 October 2010 **Number of pages:** 44 + 1 appendix

Degree programme: Information Technology

Specialisation: Software Engineering

Instructor: Auvo Häkkinen, Senior Lecturer

Supervisor: Risto Virkkala, Managing Director, EfiCode Oy

The subject of the Thesis was to replace EfiCode's old backup system, which no longer satisfied all the requirements with a new one, that would satisfy both the current needs, as well as the projected growth for the next few years.

In the research phase the different alternatives available were surveyed and compared in order to form an opinion as to which solution would be the most appropriate for EfiCode. The most important criterion of the survey, was data transfer efficiency, since some of the servers are not situated in the same country as the backup server, meaning the data has to be transferred over the Internet. This was also the part where the previous solution had had the most significant problems. Other criteria that were used included scalability, ease of use and features for archiving.

Dirvish and BackupPC were chosen as the most promising solutions for the free software culture of EfiCode, to be studied further. Then BackupPC was implemented by installing it on a dedicated server and integrating it into the EfiCode infrastructure.

After the appropriate software had been found, the next phase was implementation. The application was installed and integrated into EfiCode's infrastructure. This involved both co-operation with the administrators of the servers to be backed up, as well as adapting the application to various backup pitfalls, such as databases and version control. To avoid these pitfalls a script was written that could be installed to each of the servers that would be backed up. It would be adapted to the local environment by editing a few variables. Even though writing and testing the script required some work, the result was a solution that is readily adaptable to new servers.

The new backup system works as expected, with plenty of leeway for additional clients. It is easy to use and has more features than the previous solution. Most importantly it uses far less bandwidth, meaning that even sizeable backups over the Internet are no longer a problem.

Keywords: backup, database backup, BackupPC, Dirvish

SISÄLLYS

1 JOHDANTO

Työn tarkoituksena on rakentaa varmuuskopiointisovellus, joka soveltuu varmuuskopioiden ottamiseen sekä sisäisen verkon että tarvittaessa tavallisen laajakaistayhteyden yli ja jota myös pystyy ohjaamaan verkon yli. Koska yrityksen työpöytäympäristössä on käytössä sekä Linux-, Windows- että Macintosh-koneita, niin ohjainsovelluksen käyttöjärjestelmäagnostisuus olisi suotavaa. Varmuuskopioinnin pääkohteina tulee olemaan erilaiset Linux-palvelimet tietokantoineen. Jos sovelluksella pystyisi varmuuskopioimaan myös työasemia, niin tämä voisi mahdollisesti tulevaisuudessa olla hyödyllinen toiminto.

Tarve syntyi siitä, että yrityksen nykyisellä varmuuskopiointisovelluksella on ongelmia suorittaa varmuuskopiointi hyväksyttävässä ajassa. Tämä johtuu siitä, että sovellus siirtää aina kaiken varmuuskopioitavan datan asiakkaalta varmuuskopiointipalvelimelle. Kun kyseinen sovellus otettiin käyttöön, se täytti tarpeensa varsin hyvin. Sen jälkeen varmuuskopioitavan datan määrä on kuitenkin kasvanut huomattavasti, mistä syntyi tarve sovellukselle, joka on tehtävään paremmin optimoitu.

Työn luvussa 2 selvitetään tarpeita. Luvussa kolme tarkastellaan millaisia tallennusmedioita on olemassa ja, miten hyvin ne vastaavat esitetyt tarpeet. Tämän jälkeen luvussa neljä tutustutaan erilaisiin ongelmakohtiin ja niiden mahdollisiin ratkaisuihin. Luvuissa 5 – 7 katsastetaan erilaisia varmuuskopiointiin tarkoitettuja sovelluksia edelleen analysoiden niitä samojen tarpeiden näkökulmasta. Analyysi aloitetaan alkeellisista perussovelluksista, mutta siirrytään koko ajan lähemmäksi pääasiaa ja päädytään ratkaisuun siitä, minkä sovelluksen ympärille tämä ratkaisu rakennetaan. Itse implementaatio on esitetty luvuissa 8 – 9.

2 TARPEET

Yrityksellä on tällä hetkellä käytössään noin tusinan verran Linux-palvelimia. Linux-jakelut ja versiot vaihtelevat, mutta suurin osa on joko Debiania tai CentOSia. Palvelimien määrä saattaa nykyisten arvioiden mukaan jopa kaksinkertaistua seuraavan viiden vuoden aikana. Näillä palvelimilla pyörivät monenlaisia sovelluksia. Varmuuskopiointisovelluksen kannalta oleellisimmat ovat MySQL, PostgreSQL, Openldap, Subversion ja Git. Mitään oleellisia Windows-palvelimia ei tällä hetkellä ole. Koska tulevista asiakkaista ei ole tietoa olisi kuitenkin positiivista, jos sovellus pystyisi käsittelemään myös sellaisia.

Osa palvelimista sijaitsee Suomessa samassa lähiverkossa varmuuskopiopalvelimen kanssa, mutta osa Saksassa. Varsinkin saksalaiset palvelimet ovat tuottaneet ongelmia vanhalle sovellukselle. Tämän tähden on tärkeää, että uusi sovellus siirtää dataa niin tehokkaalla tavalla, ettei internetyhteydestä muodostu pullonkaulaa. Talteenotettavan datan määrä palvelimilla vaihtelee noin 20 – 170 gigatavun välillä.

Koska osa palvelimista sijaitsee samassa rakennuksessa kuin varmuuskopiopalvelin, olisi toivottavaa, että sovelluksella olisi arkistointitoiminto. Tämän avulla voisi varmuuskopioita siirtää toiseen fyysiseen paikkaan kaiken varalta. Mikäli tämä arkistointi toteutetaan medialle, joka on helppo viedä mukaansa, kaikkien arkistojen tulisivat olla kryptattuja yrityssalaisuuksien turvaamiseksi.

Sovellusta tulee käyttämään useita ihmisiä, joten selkeä ja helppokäyttöinen käyttöliittymä on haluttu ominaisuus. Koska sovelluksen etäkäyttö pitäisi myös onnistua, nämä vaatimukset yhdessä puoltavat vahvasti webkäyttöliittymää. Lisäksi sovelluksen toimivuutta pitäisi testata niin huolellisesti, että sitä voi käyttää turvallisin mielin eikä palautuksen yhteydessä saa tulla ikäviä yllätyksiä.

3 TALLENNUSMEDIAT

Ennen kuin voi päättää mitään ohjelmistosta, pitää ensin päättää, millaiselle medialle varmuuskopiot on tarkoitus ottaa. Pääasialliset vaihtoehdot ovat magneettinauhat, kovalevyt, SSD:t (Solid State Drive) ja optiset mediat.

Näistä vaihtoehdoista optiset mediat voidaan käytännössä heti sulkea pois. Niille ei pysty tallentamaan tarpeeksi dataa, ja ne vaativat liian paljon manuaalista työtä täyttääkseen sovelluksen tarpeet.

SSD:t putoavat myös pois heti alkumetreillä. Ne toimivat tämän sovelluksen kannalta samalla tavalla kuin kovalevyt, mutta niiden tallennuskapasiteetti on pienempi, ja ne ovat huomattavasti kalliimpia kuin kovalevyt. Niiden suurin hyöty on se, että ne ovat kovalevyjä nopeampia. Etu ei ole tässä tapauksessa tarpeeksi painava, jotta niiden haittapuolet olisivat mitenkään perusteltavissa.

Magneettinauhatallennus on menetelmä, jota on käytetty varmuuskopiointiin jo vuosikymmenien ajan. Sitä käytettiin ensimmäisen kerran jo vuonna 1951 UNIVAC I -tietokoneen kanssa, ja sitä voidaan pitää kypsänä metodina, joka käytännössä on todettu toimivaksi. [1.]

Magneettinauhatallennuksen suurin etu on perinteisesti ollut edullinen hinta kapasiteettiin nähden. Tämä tilanne on muuttunut viimeisen vuosikymmenen ajan kovalevyjen kapasiteetin kasvaessa ja hintojen pudotessa. Nykyään 1,6 TB:n nauha maksaa 900-1000 € (LTO4 Ultrium) ja 1,5 TB:n kovalevy maksaa noin 80-100 €. Lisäksi kovalevyjä voi käyttää ilman mitään lisälaitteita. Nauhat vaativat nauha-aseman, joka maksaa mallista riippuen 3000-5000 € ja jota toimivuuden takaamiseksi pitäisi huoltaa vähintään kahdesti vuodessa. Kovalevyjen luotettavuutta ja kapasiteettia voi parantaa RAID-ohjaimella (Redundant Array of Independent Disks), joka maksaa noin 350-400 €. Tällainen järjestelmä pärjää ilman säännöllistä huoltoa.

Kovalevyjen suuri etu on niiden hinta/kapasiteetti-suhteessa. Haittapuoleksi voisi taas laskea kyseenalaisen luotettavuuden. Yksi syy, miksi varmuuskopioita ylipäänsä otetaan, on se, että kovalevyt silloin tällöin hajoavat. Kun kovalevy hajoaa, tiedostojen palauttaminen sieltä voi olla kallista tai jopa mahdotonta. Tämän ikävän ominaisuuden voi kuitenkin kiertää käyttämällä RAID:ia. RAID 1, 5 ja 6 tarjoavat kaikki redundanssia, joka tarkoittaa, että vaikka yksi (tai konfiguraatiosta riippuen jopa useampi) kovalevy hajoaakin, niin yhtään dataa ei menetetä. Kyseisen kovalevyn voi korvata toimivalla, ja sen sisältö voidaan rakentaa uusiksi toisten kovalevyjen sisältämien tarkastussummien perusteella. [2, s. 77-87.]

Näistä syistä päätettiin, että varmuuskopiointi tapahtuisi ensisijaisesti kovalevyille. Koska sovellukselle kuitenkin toivotaan jonkin verran kasvuvaraa, niin olisi kuitenkin positiivista, jos ratkaisu ei lopullisesti sulkisi pois magneettinauhojen käytön mahdollisuutta.

4 ONGELMAKOHDAT

Kun varmuuskopioita tehdään, on olemassa tiettyjä ongelmakohtia, jotka tarvitsevat jonkun – usein varmuuskopiointisovelluksesta erillisen – ratkaisun. Tämä johtuu siitä, että tiedostojen varmuuskopiointi ei automaattisesti takaa, että kyseiset tiedostot tosiaan sisältävät kaiken toivotun datan. Tämä luku käsittelee näitä ongelmia sekä millä tavalla niitä voi ratkaista. Lisäksi yritetään etsiä meidän sovelluksen kannalta järkevimmät ratkaisut näihin ongelmiin.

4.1 Linkit

UNIXin linkit ovat näppärä työkalu. Itse asiassa moni varmuuskopiointisovellus hyödyntää niitä. Ne ovat myös ongelma sikäli, että jos niiden kopioimiseen ei varaudu oikealla tavalla, niin saattaa kopioida tiedoston, johon ne osoittavat. Usein tämä on toivottua, mutta varmuuskopioissa ei. Siellä halutaan kopioida linkit linkkeinä, muuten järjestelmä voi hajota ikävillä tavoilla,

kun tiedostoja palautetaan. Linkit eivät kuitenkaan ole suuri ongelma, koska ne ovat olleet osa UNIXia niin kauan, että käytännössä useimmat alkeellisetkin ohjelmat osaavat niitä huomioida.

4.2 Metadata

Hyvä varmuuskopio ei ainoastaan sisällä dataa, vaan se sisältää myös dataa, joka sallii varmuuskopion suoran palauttamisen asiakkaalle häiritsemättä suuresti toimintaa. Jotta tämä onnistuisi, täytyy datan lisäksi ottaa huomioon metadata. Jos palautuksen jälkeen kaikki tiedostojen oikeudet ja omistukset ovat erilaiset kuin alun perin, niin palautus on epäonnistunut. On hyvin todennäköistä, että tällainen järjestelmä ei toimi ollenkaan. Myös tiedostojen modifiointipäivämäärä saattaa olla oleellinen tieto joillekin sovelluksille. Tässä pätee sama asia kuin linkkeihin: Suurin osa ohjelmista osaa huomioida tämän.

Ikävä poikkeus tähän huomioon ottamiseen on kuitenkin ACL (Access Control List), joka on tiedostojärjestelmään, eikä tiedostoon sidottu ominaisuus. Kun tiedosto siirretään toiselle tiedostojärjestelmälle ACL-tieto menetetään. [3.] Tämä koskee sekä Windowsin ACL-implementaatiota, Mac OS X:n implementaatiota että eräiden UNIXien käyttämää POSIX.1e-implementaatiota. Tämä ei ole suurikaan ongelma meille, koska meidän koneilla ACL ei ole käytössä, joten sen käsittely on jätetty tämän työn ulkopuolelle. Mainittakoon kuitenkin, että ACLt ovat tunnettu varmuuskopiointiongelma ja niitä varten on olemassa erikoistuneita sovelluksia, jotka ottavat pelkästään listasta varmuuskopion. [4.]

4.3 Tiedostojärjestelmän yhtenäisyys

Tiedostojärjestelmän varmuuskopiointi on prosessi, joka saattaa kestää tunteja. Palvelimella pyörivät prosessit, kuitenkin muokkaavat koko ajan tiedostoja. Tämä tarkoittaa, että tiedostojärjestelmä kokonaisuutena tulee olemaan erinäköinen, kun varmuuskopiointi on valmis, verrattuna siihen, mitä se oli,

kun kopiointi aloitettiin. Talteen otetut tiedostot tulevat olemaan jostain tältä väliltä. Riippuen siitä, millaisessa käytössä kyseinen kone on, tämä voi olla ongelma, tai sitten ei. On kuitenkin olemassa menetelmiä saada talteen tiedostot niin, että jokainen tiedosto sisältää sen datan, minkä sisälsi kopioinnin alkaessa. Tätä kutsutaan tiedostojärjestelmän valokuvaamiseksi (engl. snapshot), ja se on yhtymänimi monille eri tekniikoille saavuttaa tämä tulos. [5, s. 667.]

Eniten Linuxilla käytetty tekniikka on LVM-kuva (Logical Volume Manager). Se edellyttää, että LVM on käytössä ja että vapaata tilaa on tarpeeksi. Tässä tapauksessa ”tarpeeksi” tarkoittaa, että kaikki varmuuskopioinnin aikana tapahtuvat muutokset pitää mahtua tähän tilaan. Koska kuvan ottaminen pitää tapahtua muutamassa sekunnissa, se tarkoittaa, ettei dataa ehditä kopioida mihinkään, vaan tiedostojärjestelmän muutokset otetaan kuvan luontihetkestä eteenpäin talteen toiselle volyymille. Kun varmuuskopiointisovellus lukee kovalevyltä, sille tarjotaan jäädytetty valokuva, jota saadaan aikaan vertaamalla tiedostojärjestelmä tallennettujen muutosten kanssa. Muut sovellukset näkevät kaikki muutokset ikään kuin kuvaa ei olisi olemassakaan. Kun varmuuskopio on valmis, kuva voidaan yksinkertaisesti poistaa ja hävittää. [6.]

Toinen mielenkiintoinen ja uudempi tapa, tulee NILFS2-tiedostojärjestelmän mukana. NILFS2 on lokistrukturoitu tiedostojärjestelmä, ja kuten tällaiset tiedostojärjestelmät yleensä, se tukee itsessään valokuvien ottamisen. Tätä menetelmää tullaan tuskin tässä työssä käyttämään, sillä NILFS2 on hyvin tuore tiedostojärjestelmä. Se tuli kerneliin vasta versiossa 2.6.30 ja on edelleen ”kokeiluvaiheessa”. Tulevaisuutta ajatellen se voi kuitenkin olla mielenkiintoinen vaihtoehto. [7.]

4.4 Tietokannat

Tietokannat ovat oma ongelmansa, koska pelkästään tiedostojen kopiointi ei riitä takaamaan, että tietokanta kopioituu järjellisessä tilassa. Kaiken lisäksi

eri tietokannat tallentavat tietonsa eri tavalla – jotkut jopa vaativat käyttöönsä kokonaisen osion eivätkä käytä tiedostoja. Ne käyttävät myös eri hallintaohjelmia. Tämän takia jokaiselle tietokannalle on laadittava oma varmuuskopiointistrategia. [8, s. 415.] Meidän sovelluksen kannalta riittää jos pystymme hallitsemaan MySQL- ja PostgreSQL-tietokannat.

Tietokantojen varmuuskopioinnille on periaatteessa kaksi perusstrategiaa: kylmä ja kuuma. Kylmässä varmuuskopiossa tietokanta otetaan pois päältä, suoritetaan varmuuskopiointi ja tietokanta nostetaan uudelleen pystyyn. Tämän lähestymistavan hyvä puoli on yksinkertaisuus sekä varmuuskopioinnissa että palautuksessa. Huono puoli taas on siinä, ettei tietokantaa voi käyttää varmuuskopioinnin aikana. (Edellä mainittu valokuvaustekniikka voi kuitenkin auttaa tässä.) Kuuman varmuuskopioinnin etu on, että kanta on koko ajan käytettävissä – haittapuoli on taas siinä, että eri tietokannat käyttävät hyvin erilaisia menetelmiä ja työkaluja ja että palautus yleensä vaatii lisätoimenpiteitä. [5, s. 391 ; 9, s. 44.]

Varmuuskopiointitekniikat voi myös jakaa ”raakoihin” ja ”loogisiin”. Raaka varmuuskopio kopioi yksinkertaisesti tietokannan binääritiedostot, kun loogi-nen puolestaan tallentaa ne komennot, jota pitää ajaa, jotta kanta saadaan siihen tilaan, missä se varmuuskopiointihetkellä on. Kylmät varmuuskopiot ovat yleensä raakoja niiden yksinkertaisuuden takia. Raaka varmuuskopio voidaan palauttaa muiden tiedostojen mukana ilman erityisiä lisätoimenpiteitä. Kuumat varmuuskopiot ovat yleensä loogisia, mutta esimerkiksi PostgreSQL tarjoaa vaihtoehdon, joka on siltä väliltä (raaka + WAL, missä WAL on looginen komponentti joka sisältää muutoksia). [5, s. 636 ; 9, s. 44.]

4.5 LDAP

Suosittu tapa varmentaa ja valtuuttaa käyttäjiä nykyään on LDAP (Lightweight Directory Access Protocol). LDAP-implementaatioita on monenlaisia, ja sisäisestikin ne saattavat toimia hyvin eri tavoilla. Ne voivat myös tallentaa

tietonsa monella eri tavalla, mutta hyvin usein käytössä on jonkinlainen tietokanta. [10.] Meidän käytössämme on OpenLDAP, jonka backendinä toimii BerkeleyDB-tietokanta. Kaikki mitä luvussa 3.4 todettiin tietokannoista, pätee myös tähän ja tämä on otettava huomioon. OpenLDAPin mukana tulee kuitenkin kaikki tarvittavat työkalut varmuuskopioiden ottamiseen.

4.6 Versiohallintajärjestelmät

Versiohallinnan ongelma on pitkälti sama kuin tietokannoilla: miten varmistua siitä, että saa kopion järjestelmästä, joka on vakaassa tilassa, mieluiten ottamatta palvelua alas pitkäksi aikaa? Ongelma on kuitenkin astetta lievempi kuin tietokantojen kanssa osaksi siitä syystä, että versiohallintajärjestelmät ovat rakennettu tavalla, joka tekee niiden sisällön kopioimisen toiseen paikkaan yksinkertaiseksi. Esimerkiksi gitin kanssa tavallisin varmuuskopiointistrategia on peilin käyttäminen. Varmuuskopiointipalvelimelle pystytetään tietolähde, johon asiakas itse työntää varmuuskopiot cronjobin avulla. Tämä vaikuttaa olevan täysin pätevä strategia meillekin.

Käytössämme on gitin lisäksi, ainakin toistaiseksi, subversion. Subversionilla on työkaluja, joilla pystyy tekemään ”kuumia” kopioita sen tietolähteistä. Yksinkertaisin tapa käsitellä tämä on ottaa tällainen kopio esimerkiksi koukkuskriptin avulla ennen jokaista varmuuskopiota ja palauttaa kyseinen kopio palautuksen jälkeen. [11.]

5 PERUSTYÖKALUT

Koska tiedon varmuuskopiointi on melkein yhtä vanha keksintö kuin tiedon tallennuskin, niin ei ole yllättävää, että on olemassa suurehko määrä valmiita ohjelmia tätä tarkoitusta varten. Tämän luvun idea on käydä läpi tavallisesti käytettyjä perustyökaluja sekä arvioida niiden soveltuvuus tälle projektille. Koska sovelluksen pääasiallinen käyttötarkoitus tulee olemaan Linux-palvelimien varmuuskopiointi, niin on luontevaa tarkistaa Linux-pohjaisia varmuuskopiointiratkaisuja.

5.1 Dump / Restore

Dump ja sen vastakappale restore ovat vanhimpia varmuuskopiointityökaluja, joita vielä käytetään. Ensimmäinen versio ohjelmasta tuli AT&T UNIX v6:n mukana vuonna 1975. [12.] Se perustuu lohkojen käyttöön eli se toimi matalammalla tasolla kuin tiedostotasolla ja se on lähinnä tarkoitettu käytettäväksi magneettinauhojen kanssa. Tämän lisäksi, ohjelma on kiistelty, koska sen lohkokeskeisyys ei välttämättä sovi liitetyn median varmuuskopiointiin. RedHat virallisesti julisti työkalun vanhentuneeksi versiossa 9 ja Linus Torvalds on itse sanonut, että se, joka käyttää dumpia varmuuskopiointiin Linuxilla, ”pelaa venäläistä rulettia datansa kanssa”. Tämä ohjelma ei ole varteenotettava vaihtoehto sovelluksen osaksikaan. [5, s. 70.]

5.2 Dd

Dd on työkalu, joka toimii matalalla tasolla, käyttäen raakoja laitteita. Sillä pystyy muun muassa ottamaan varmuuskopioita kokonaisesta kovalevystä, vaikka sen osiointijärjestelmä tai tiedostojärjestelmä olisi täysin tuntematon. Se ei ole varmuuskopiointityökalu sanan varsinaisessa merkityksessä, mutta sillä voi tehdä asioita, joihin muut työkalut eivät pysty. Jos esimerkiksi pitäisi ottaa varmuuskopio sellaisesta tietokannasta, joka ottaa käyttöönsä kokonaisen osion, sen sijaan, että kirjoittaisi tavalliseen tiedostoon (esimerkiksi DB/2 tai Oracle), niin dd on yksi harvoista työkaluista, joka voisi auttaa. Tällaisia tietokantoja ei kuitenkaan ole yrityksellä käytössä, eikä näytä todennäköiseltä, että tulevaisuudessakaan olisi. On hyvä tietää, että tällainen työkalu on olemassa, mutta se ei todennäköisesti tule milloinkaan olemaan osa nyt rakennettavaa sovellusta. [5, s. 62.]

5.3 Tar

Tar on yksinkertainen varmuuskopiointiohjelma, jota käytetään myös laajasti ohjelmistojen paketoimiseen levitystä varten. Kun Linuxille hakee ohjelmia verkosta, niin ne ovat hyvin usein pakattuja tar-paketteja. Tar on lyhenne sa-

noista ”Tape ARchiver”, ja se on alun perin tehty ottamaan varmuuskopioita nauhoille. Sen suurimmat hyödyt ovat helppokäyttöisyys sekä ohjelman laajalle levinnyt käyttö UNIX-maailmassa.

Vaikka taristä helposti saisi väännettyä yksinkertaisen varmuuskopiointisovelluksen esimerkiksi käyttämällä sitä sshfs:n kanssa, tällä sovelluksella olisi todennäköisesti samanlaisia kaistaongelmia kuin nykyisellä sovelluksella. Koko varmuuskopioitava aineisto olisi siirrettävä jokaisen varmuuskopioinnin yhteydessä, mikäli valitsisi tällaista ratkaisua. Lähinnä tästä syystä taria ei tulla käyttämään ainakaan pääasiallisena komponenttina nyt kehitettävässä ratkaisussa. [5, s. 63.]

5.4 Cpio

Cpio on tarin kaltainen ohjelma, joka ei kuitenkaan ole saavuttanut tarin suosiota. Se pystyy varmuuskopioimaan stdin:stä tulevaa dataa, ja siten on mahdollista tehdä inkrementaalisia varmuuskopioita käyttämällä find- ja touch-komentoja. Sen helppokäyttöisyys ei kuitenkaan ole tarin tasoa. Inkrementaaliset varmuuskopioinnit hoituvat näppärämmin käyttämällä rsynciä, joten tätäkään ohjelmaa ei tulla käyttämään. [5, s. 61.]

5.5 Rsync

Rsync on ohjelma, joka pystyy synkronoimaan tiedostoja yhdestä paikasta toiseen siirtäen vain minimaalisen määrän dataa. Tätä varten rsync käyttää Andrew Tridgellin kehittämää delta-encoding-algoritmin muunnosta. Se toimii siten, että vastaanottaja pilkkoo päivitettävän tiedoston vakiopituisiin osiin ja laskee jokaiselle osalle MD5-tarkistussumman sekä pyörivän tarkistussumman ja lähettää ne lähettäjälle. Lähettäjä pystyy näiden avulla laskemaan, mitkä osat tiedostosta ovat muuttuneet ja lähettämään vain ne osat sekä informaation siitä, miten kyseiset osat pitää liittää vastaanottajan tiedostoon, jotta lopputulos olisi identtinen kopio lähettäjän versiosta. Tämän järjestelmän ainoa heikko puoli on se, että ei riitä, että rsync on asennettu

toisessa päässä, vaan molemmat osapuolet tarvitsevat sen. Tästä ei ole nykyään suurtakaan haittaa, koska käytännössä kaikilla Linux-jakeluilla on rsync valmiiksi asennettuna. Tästä seuraa kuitenkin se, että rsyncin kanssa voi tulla ongelmia, jos kone on NATin takana, eikä toinen pääse muodostamaan siihen suoraa yhteyttä. [13 ; 14, s. 219.]

Rsync on siis ideaalinen ohjelma sovellukselle, jonka pitää siirtää tiedostoja käyttäen mahdollisimman vähän kaistanleveyttä. Rsync siirtää vain *tiedostojen muutokset, ei muutettuja tiedostoja* kuten lähes kaikki muut ratkaisut. Siten se välttää käytännössä kaiken turhan toiston. Koska tarkistussummia on kaksi, virheellisen positiivisen todennäköisyys on häviävän pieni. UNIXin perustyökaluista rsync on selkeästi lupaavin ehdokas, minkä ympärille tämänkaltainen sovellus kannattaa rakentaa.

6 VALMIIT SOVELLUSPAKETIT

Perusohjelmien lisäksi on olemassa laajahko valikoima pidemmälle kehittyneitä sovelluksia. Jotkut näistä käyttävät taustalla edellisessä luvussa mainittuja perussovelluksia, jotkut ovat taas täysin itsenäisiä ratkaisuja. Suurella osalla on valmiiksi kehitetty graafinen käyttöliittymä ja, jotkut voidaan ohjata valmiin webliittymän kautta. Tämän luvun tarkoitus on arvioida näiden ohjelmien soveltuvuutta tälle projektille.

6.1 AMANDA

AMANDA (Advanced Maryland Automated Network Disk Archiver), on kehitetty Marylandin yliopistolla vuonna 1991. Sen päätarkoituksena on pystyä varmuuskopioimaan suuri määrä työasemia yhdellä ainoalla varmuuskopiointipalvelimella.

AMANDA on laajasti käytetty, pitkälle kehitetty varmuuskopiointisovellus. Se on vuosien saatossa ehtinyt hioutua hyvin toimivaksi tuotteeksi. AMANDAlle on jopa saatavissa kaupallista tukea Zmanda-nimisen yrityksen kautta.

AMANDA toimii ”disk-to-disk-to-tape” -periaatteella, mikä tarkoittaa, että se ensin kopioi varmuuskopioitavan datan palvelimen omalle kovalevylle, josta se siirtyy magneettinauhalle. Tämä tehdään, jotta päästään eroon ”kengänkiillotusefektistä”. Kengänkiillotus tarkoittaa tilannetta, jossa nauha-asema ei saa dataa sisään niin nopeasti kuin mitä se nauhalle kirjoittaa. Silloin asema joutuu pysäyttämään nauhan, kelaamaan sen takaisin vähän matkaa ja jatkamaan sieltä. Jos tätä tapahtuu jatkuvasti, niin lopputulos on, että asema jatkuvasti hinkkaa nauhaa edestakaisin vähän kuin kengänkiillottaja rättiään. Tämä kuluttaa sekä aseman lukupäitä että nauhaa. Nykyiset asemat ovat niin nopeita, että näin käy lähes varmasti, jos dataa siirretään asemalle suoraan verkon yli. Paikallisen kovalevyn käyttö väliaikaisesti takaa tarpeellisen siirtonopeuden. [5, s. 133, 260.]

Nyt rakennettavan sovelluksen kannalta AMANDA on kiinnostava, mutta samalla turhan raskas. Vaikka se osaakin käyttää kovalevytallennusta, niin se on juuriltaan nauhasovellus, ja se on rakennettu nauhan ympärille. Se on kuitenkin sovellus, joka kannattaa pitää mielessä, jos järjestelmäämme lähdetään jossain vaiheessa laajentamaan magneettinauhojen tukemiseksi. Koska AMANDA ottaa ensin varmuuskopionsa kovalevylle, niin se pystyy jossain määrin tekemään yhteistyötä täysin kovalevypohjaisten sovellusten kanssa (esimerkiksi BackupPC). [15.]

6.2 Bacula

Bacula on varmuuskopiointisovellus, jonka kehitys alkoi vuonna 2000 ja joka julkaistiin vuonna 2002. Ohjelman tarkoitus on olla mahdollisimman hyvin skaalautuva, jotta sitä voisi käyttää myös hyvin suurissa yrityksissä. Tämän saavuttaakseen Baculan kaikki komponentit ovat niin eriteltyjä, kuin vaan on ollut mahdollista. Tiedostopalvelin on eroteltu tallennuspalvelimelta ja varmuuskopiointipalvelin on eroteltu molemmilta näistä. Tämä tarkoittaa, että skaalautuvuus on erinomainen, mutta se tarkoittaa myös, että asennus ja ylläpito on kaikkea muuta kuin yksinkertaista. Tämän lisäksi Bacula käyttää ai-

van omaa tallennusmuotoa, joka ei ole yhteensopiva minkään muun ohjelman kanssa. Baculalla otetut varmuuskopiot täytyy myös palauttaa Baculalla. Muun muassa näistä syistä Baculan käyttöönottoa ei missään vaiheessa vakavasti harkittu. [16 ; 5, s. 159.]

6.3 Dirvish

Dirvish on yksinkertainen ja kevyt rsynciin perustuva varmuuskopiointityökalu, joka on toteutettu perlillä. Se toimii täysin komentorivipohjalta. Mikäli tämän ratkaisun valitsee, niin graafista käyttöliittymä pitää rakentaa itse, jos sellaisen haluaa. Tästä huolimatta Dirvish on täysin vakavasti otettava vaihtoehto. Nimensä Dirvish on saanut dervisseistä (engl. dervish), jotka ovat tunnettuja pyörivästä tanssistaan, koska Dirvish tallentaa varmuuskopionsa pyöriville magneettikiekoille (eli kovalevyille). Tämä soveltuu hyvin meidän tarkoituksiimme, ja sen tähden tähän ohjelman toimintaan tullaan perehtymään perusteellisemmin kuin muihin tähän mennessä läpikäytyihin. [17.]

6.3.1 Kovat linkit

Jos Dirvishin toinen tukipilari on rsync, joka takaa tiedostojen nopean siirron verkon yli, niin toinen on UNIXin kovat linkit. Niiden ansiosta tiedostot eivät vie turhaa tilaa kovalevyltä. Dirvish luo joka varmuuskopiointia varten täydellisen tiedostorakenteen, joka on suoraan palautettavissa asiakkaalle. Jos jokin tiedosto ei ole muuttunut edellisestä versiosta (hyvin tavallinen tilanne), niin Dirvish ei tee tiedostosta kopiota, vaan luo ainoastaan kovan linkin siihen.

UNIXissa jokaisella tiedostolla tiedostojärjestelmässä on oma uniikki numeronsa, jota kutsutaan indeksisolmuksi. Käyttöjärjestelmä pystyy viittaamaan kaikkiin tiedostoihin niiden indeksisolmujen kautta. Kovat linkit toimivat siten, että samalle indeksisolmulle voidaan luoda useampia viittauksia, jotka toimivat samaan malliin kuin osoittimet C-kielessä. Kaikki tiedostoon tehdyt muutokset, näkyvät kaikissa kovissa linkeissä, koska kyseisestä datasta on vain

olemassa yksi ainoa instanssi. Tiedostojärjestelmä pitää kirjaa kovien linkkien määrästä jokaiselle tiedostolle. Kun tiedosto (= kova linkki tiedostoon) poistetaan, niin kyseinen kova linkki poistetaan, linkkien määrää dekrementoidaan. Jos määrä on nolla, niin myös se alue kovalevystä, missä data sijaitsee, julistetaan vapaaksi. [18.]

Yksi tämän järjestelmän hienous on siinä, että kovat linkit eivät käytännössä vie tilaa lainkaan. Jos on neljän gigatavun tiedosto, jolle on neljä kovaa linkkiä, niin näyttää siltä, että on neljä tiedostoa, jotka vievät yhteensä 16 GT, mutta oikeasti tilaa ei ole mennyt kuin 4 GT. Tätä ominaisuutta Dirvish hyödyntää.

6.3.2 Rajoitukset

Kovien linkkien käytöstä seuraa muutama rajoitus, joista yksikään ei kuitenkaan ole erityisen vakava. Ensimmäinen rajoitus tulee siitä, että kovien linkkien käyttö sillä tavalla kuin Dirvish niitä käyttää, edellyttää UNIX-tyyppisen tiedostojärjestelmän (ja sitä myöten UNIX-tyyppisen käyttöjärjestelmän). Windowsin käyttämä NTFS tukee jossain määrin myös kovia linkkejä, mutta ei ole olemassa työkalua, jolla niitä voisi hallita samalla tavalla kuin UNIXin ln:llä pystyy. Koska aikomus on käyttää Linuxia, niin tämä ei ole ongelma. Asiakaspuolella on kuitenkin mahdollista käyttää NTFS:ää, tai FAT32 -pohjaista järjestelmää, kunhan asiakkaalta löytyy rsync. Windowsilla tämän voi hoitaa esimerkiksi cygwinin avulla.

Toinen rajoitus tulee siitä, että indeksisolmujen määrä on joissain tiedostojärjestelmissä rajoitettu määrään, joka lyödään kiinni, kun järjestelmä luodaan. Tavallisessa käytössä tämä raja ei todennäköisesti tule ikinä vastaan. Mutta kun kovia linkkejä käytetään niin paljon kuin Dirvish niitä käyttää, niin solmut saattavat loppua ennen kuin levytila loppuu. Helpoin tapa kiertää tämä rajoitus on käyttää tiedostojärjestelmää, jossa indeksisolmujen määrää voi dynaamisesti kasvattaa, esimerkiksi reiserfs. [19.]

Lopuksi kovien linkkien käyttö edellyttää, että tiedostot sijaitsevat samalla tiedostojärjestelmällä. Koska kyseessä on tiedostojärjestelmän ominaisuus, niin linkittäminen kahden järjestelmän välillä on mahdotonta. Tämä tarkoittaa sitä, että mikäli kovalevytilaa tarvitsee lisää, niin ei voi pelkästään lisätä kovalevy, formatoida ja käskeä Dirvishiä jatkamaan siitä. Joko täytyy jakaa toiset varmuuskopiot toiselle levylle ja toiset toiselle manuaalisesti tai mieluummin käyttää dynaamisesti laajennettavaa järjestelmää – esimerkiksi RAID5 + LVM. Suurehkon RAIDin luominen heti alkuun auttaa tietysti myös. Päämäärä on luoda järjestelmä, jota ei tarvitse laajentaa sen ennustetun eliniän aikana. Sopivankokoisella RAIDilla ja arkistointisuunnitelmalla, missä vanhoja varmuuskopioita siirretään säännöllisesti ulkoisille kovalevyille, tähän lienee mahdollista päästä.

6.3.3 Sisäinen logiikka ja käyttö

Dirvish varastoi varmuuskopionsa ”pankki → holvi → kuva”-logiikan mukaan. Pankki on paikka, joka sisältää n kappaletta holvia. Holvi on paikka, joka sisältää n kappaletta kuvaa samasta kohteesta. Kuva on taas yhden varmuuskopiointioperaation tulos. Pankit (joita voi olla periaatteessa rajattomasti) määritellään pääkonfiguraatiotiedostossa `/etc/dirvish/master.conf` . Ne ovat vain tavallisia hakemistoja. Holvit määritellään luomalla pankiksi määritellyssä paikassa hakemiston ja sen alle hakemiston sekä tiedoston nimeltä `dirvish/default.conf` . Tämä tiedosto voi sisältää asiakaskohtaista konfiguraatiotietoa, joka ajaa oletusarvojen yli. Kaikki hakemistot, jotka täyttävät kyseiset ehdot, lasketaan holveiksi. Kuvia ei voi itse luoda, vaan ne syntyvät joka kerta, kun Dirvish tekee kyseisestä holvista varmuuskopio-operaation. [17.]

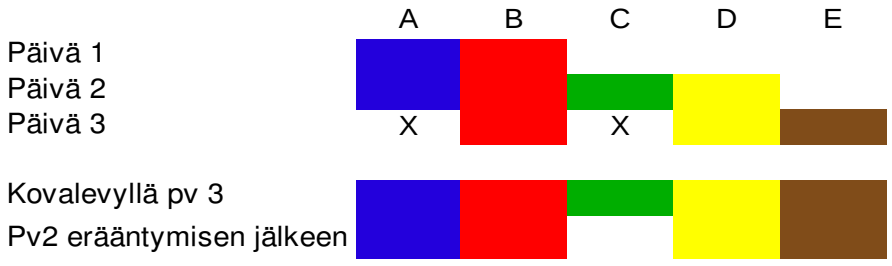
Kun rsync muodostaa yhteyden asiakkaalle, niin tämä tapahtuu ssh-verkkoprotokollan avulla. Kaikki data siirtyy kryptattuna, mikä on hyvä asia. Ssh ei kuitenkaan (hyvästä syystä) päästä ketään sisään root-oikeuksilla noin vaan, vaikka mielessä olisikin hyödyllinen varmuuskopiointioperaatio. Koska

varmuuskopiointia halutaan mielellään suorittaa automaattisesti ja mielellään öisin, kun muu käyttö on vähäistä, niin ei ole hyväksyttävää, että järjestelmänvalvoja joutuu kirjoittamaan salasanan joka kerta, kun Dirvish haluaa suorittaa varmuuskopioinnin. Tästä syystä on suotavaa luoda palvelimella root-tunnuksilla ssh-avainpari ja asettaa tämän parin julkinen osa luotetuksi ssh-avaimeksi jokaiselle asiakkaalle.

6.3.4 Vanhojen kuvien erääntyminen

Vanhoja varmuuskopioita ei säilytetä itse palvelimella ikuisesti, vaan jossain vaiheessa niiden annetaan erääntyä, mahdollisesti arkistokopion ottamisen jälkeen. Erääntyminen tarkoittaa Dirvishin kannalta yksinkertaisesti poistoa. UNIX-pohjaiset tiedostojärjestelmät pitävät huolen siitä, että vain sellaiset tiedostot, joihin ei enää osoita yhtään kovaa linkkiä oikeasti, poistetaan. Se, kuinka paljon levytilaa vapautuu erääntymisen yhteydessä, riippuu siis siitä, kuinka paljon yhteisiä tiedostoja kyseisellä kuvalla on muiden samaan holviin kuuluvien kuvien kanssa. [17.]

Seuraavaa kuva havainnollistaa tilannetta. Ensimmäisenä päivänä otetaan varmuuskopio, johon kuuluvat tiedostot A ja B. Sitä seuraavana päivänä otetaan varmuuskopio, johon sen lisäksi kuuluvat tiedostot C ja D. Kolmantena päivänä on lisätty tiedosto E, mutta poistettu tiedostot A ja C.



Kuva 1. Kovalevytilan kuvaus

Tässä vaiheessa kyseisessä holvissa tulee olemaan kaikki tiedostot A-E. Jos yhdenkin niistä poistaisi, niin jokin kuva menisi pieleen siten, että sitä ei voisi enää täydellisesti palauttaa. Jos tässä vaiheessa päätetään antaa toi-

sen päivän kuva erääntyä, niin tiedosto C hävitetään ja sen viemä tila vapautuu käyttöön, koska yksikään kuva ei enää sitä tarvitse. [20.]

6.3.5 Muita ominaisuuksia

Yksi Dirvishin kiinnostavia ominaisuuksia on mahdollisuus lisätä niin sanottuja koukkuja (engl. hook scripts). Näitä koukkuja on neljä kappaletta. Niitä ajetaan joko palvelin- tai asiakas-puolella, ja joko ennen tai jälkeen varmuuskopioinnin. Näitä voi räätälöidä jokaisen asiakkaan mukaan ja esimerkiksi käyttää ongelmakohtien varmuuskopiointia varten. Näitä ongelmakohtia voivat olla mitkä tahansa tiedostot, jotka eivät välttämättä kopioidu ”järjellisessä tilassa” normaalin kopiointioperaation seurauksena. Tietokannat ja versiohallintajärjestelmät ovat hyviä esimerkkejä, mutta niistä lisää myöhemmin.

Tiedostojen haku ja palautus on tärkeä osa jokaista varmuuskopiointijärjestelmää. Dirvishissä haku onnistuu komentorivityökalulla käyttäen perlin säännöllisiä lausekkeita. Niillä voi tehokkaasti ilmaista asioita, mutta ne olettavat melko korkeaa tietämystasoa käyttäjältä. Tiedostojen palautus onnistuu suoraan rsyncillä, koska kaikki tiedostot esiintyvät samassa polkurakenteessa, ja samoilla oikeuksilla kuin alkuperäisessäkin. Tämä edellyttää kuitenkin, että palautuksen tekijä pääsee varmuuskopiointipalvelimelle root-käyttäjänä. Kaiken kaikkiaan Dirvish on kuitenkin hyvin lupaava ehdokas järjestelmällemme. [17.]

6.4 BackupPC

BackupPC on varmuuskopiointiohjelmisto, jolla on hyvin paljon yhteistä Dirvishin kanssa, ja siten se on myös lupaava ehdokas. Ensimmäinen versio BackupPC:stä julkaistiin vuonna 2001 ja sitä kehitetään edelleen aktiivisesti. Aivan kuten Dirvish, sen toiminta perustuu kovalevyihin, rsynciin ja koviin linkkeihin. Toisin kuin Dirvishillä BackupPCllä on valmis webpohjainen käyttöliittymä. Tämä on jo paljon pidemmälle kehitetty kuin mitä voisi realistisesti olettaa, että voisi itse rakentaa tämän projektiin nähden järkevässä ajassa,

mikäli lähtisi nollasta. Tämä liittymä toimii minkä tahansa http-palvelimen kanssa, joka ymmärtää perliä. Suurin osa dokumentaatiosta kuitenkin olettaa, että käytössä on Apache.

6.4.1 Tiedonsiirto

BackupPCn manuaali suosittelee käyttämään joko rsynciä tai rsyncin daemonia tiedostonsiirtoon. Muita vaihtoehtoja on kuitenkin tarjolla. BackupPC pystyy myös käyttämään taria, jos asiakkaalla ei jostain syystä ole rsync käytettävissä. Kolmas mahdollisuus on pystyttää asiakkaan puolelle FTP-palvelin, johon varmuuskopiointipalvelin ottaa yhteyttä. Kumpikaan näistä menetelmistä ei tue sitä, että siirretään vain se osa tiedostosta, joka on muuttunut samalla tavalla kuin rsync. Tämä johtaa tietenkin ylimääräisen datan siirtämiseen ja siten huonompiin nopeuksiin, mutta valinnanvara ei ikinä ole pahitteeksi. [21.]

Tämän lisäksi dataa voi myös siirtää samban kautta. Metodi on lähinnä tarkoitettu Windows-työasemia varten. Se on metodeista hitain, mutta se on helpompi ottaa käyttöön kuin UNIX-työkalujen käyttö cygwinin alla. Tämä voi olla suurikin hyöty, jos työasemia on paljon eikä käyttäjäkunnan osaamisesta ole tietoa.

6.4.2 Tiedon tallennus

Kuten jo kävi ilmi, niin BackupPC käyttää UNIXin kovia linkkejä hyödykseen ja kaikki, mikä tässä pätee Dirvishille, pätee BackupPCllekin. BackupPC kuitenkin vie konseptin vielä pidemmälle. Se ei tyydy linkittämään samaan holviin kuuluvat samanlaiset tiedostot yhteen, vaan se linkittää *kaikkien tekemiensä varmuuskopioiden* samanlaiset tiedostot yhteen globaalissa varastotilassa. Jos varmuuskopioi paljon samanlaisia asiakkaita, niin tilansäästö voi olla hyvinkin huomattava.

Tilaa säästyy myös siksi, että BackupPC pakkaa varmuuskopioidun datan käyttäen gzipin zlib-kirjastoa. [21.] Pakkaustason voi valita itse (tai kytkeä

pois päältä) ja siis päättää, onko aika vai kovalevytila tärkeämpi. Oletusarvoisesti pakkaus on päällä tasolla 3 (max = 9), jonka arvioidaan olevan hyvä kompromissi.

6.4.3 Tiedon palautus

Palautus onnistuu helposti BackupPCn graafisen käyttöliittymän avulla. Koska liittymä tukee useampia käyttäjiä ja oikeuksien jakamista, niin jokaiselle käyttäjälle voi sallia pääsyn omille varmuuskopioilleen. Tämä tarkoittaa, että käyttäjän ei tarvitse pyytää palautusta varmuuskopiojärjestelmän ylläpitäjältä, vaan hän voi itse palauttaa omat varmuuskopionsa. Palautettavat tiedostot valitaan laittamalla rasti ruutuun. Palautusmetodina voi valita joko palautuksena suoraan tiettyyn paikkaan (jossa oletuksena on se paikka mistä varmuuskopiot otettiinkin) tai sitten palvelin voi lähettää tiedostot pakattuna joko zipillä tai tarilla.

BackupPCssä on myös arkistointitoiminto. Palvelimia voi määritellä arkistointipalvelimiksi. Sen jälkeen niihin voi arkistoida asiakkaiden viimeisin varmuuskopio tar-pakettina. Arkistointipalvelimien määrä ei ole rajattu, ja se voi myös olla sama palvelin, jossa sovellus pyörii, eli voi arkistoida esimerkiksi USB-kovalevylle tai nauha-asemalle. Tällä tavoin voi myös tehdä off-site varmuuskopioita. BackupPCtä voi myös käyttää erillisen sovelluksen (esimerkiksi AMANDAn) kanssa, joka arkistoi magneettinauhalle. Kaiken kaikkiaan BackupPC on hyvin pätevä sovellus, joka näyttää soveltuvan tarkoituksiimme erinomaisen hyvin. Seuraavaksi tulenkin asettamaan pääehdokkaat Dirvishin ja BackupPC:n vastakkain ja vertaamaan niiden vahvat ja heikot puolet. Tämän jälkeen parempi ohjelma valitaan, implementoidaan ja testataan.

7 DIRVISH VASTAAN BACKUPPC

Tässä luvussa verrataan vain näitä kahta sovellusta toisiinsa projektimme näkökannalta. Päämääränä on selvittää, kumpi sovellus sopii paremmin tarkoituksiimme punnitsemalla molemman ohjelman vahvat ja heikot puolet

sekä arvioida, millaisia toimenpiteitä joudutaan tekemään paremman ohjelman heikkojen puolien paikkaamiseksi.

7.1 Asennus ja käyttöönotto

Perusasennus sujuu molemmilla ohjelmilla varsin kivuttomasti. Molemmat ohjelmat löytyvät suoraan melkein kaikkien Linux-jakeluiden pakettivarastoista. Asennus on sikäli lastenleikkiä.

Konfiguraatiokaan ei juuri tuota ongelmia kummallakaan ohjelmalla. Dirvishiä konfiguroidaan luomalla ja muokkaamalla tekstitiedostoja. BackupPCtä voi konfiguroida joko webkäyttöliittymän avulla tai sitten tekstitiedostoja muokkaamalla. Usein käytettiin jälkimmäistä vaihtoehtoa, koska tiedostot olivat hyvin kommentoituja ja niiden esimerkeillä oli helppo saada toivottu tulos aikaan. Kommenteissa – ja dokumentaatiossa yleensäkin BackupPC vetää pidemmän korren. Dirvishissä itsessään ei löydy juuri lainkaan dokumentaatiota (yksi suppea man-sivu) ja ilman Internetiä ja hakukonetta sen käytössä ei pääsisi kovin pitkälle.

Apachen käyttöönottoa https:llä ei ole laskettu mukaan tähän, koska se ei varsinaisesti liity BackupPChen. Jos Dirvishille rakentaa webliittymän, niin sille joutuu tekemään saman. Webpalvelimen säädöistä tulee lisää myöhemmin.

Tässä kategoriassa BackupPC vie niukan voiton laajemman dokumentaationsa ansiosta.

7.2 Ominaisuudet

Ominaisuuslistaa ei tarvitse tuijottaa kovin kauan ennen kuin huomaa, että BackupPC näyttää olevan vahvempi tässä. Suurin ominaisuus, joka Dirvishiltä puuttuu kokonaan, on pitkälle kehitetty webliittymä. Monet muut asiat, jotka Dirvish tekee hyvin, BackupPC tekee myös, ja usein tarjoaa myös lisävaihtoehtoja, joita Dirvish ei tarjoa. Dirvish pystyy käyttämään rsynciä, sa-

moin BackupPC. Sen lisäksi se voi käyttää sambaa, taria ja ftptä. Dirvish käyttää kovia linkkejä säästääkseen kovalevytilaa, samoin BackupPC. Sen lisäksi se pystyy linkittämään tiedostoja myös täysin eri asiakkailta säästäen vielä enemmän tilaa. Se voi myös halutessaan pakata kyseisen datan.

Taulukko1: Ominaisuusvertailu

Ominaisuus	BackupPC	Dirvish
Varmuuskopioi kovalevylle	✓	✓
Komentorivipohjainen käyttö	✓	✓
Käyttö graafisen webkäyttöliittymän avulla	✓	✗
Tuki rsyncille	✓	✓
Tuki tarille	✓	✗
Tuki samballe	✓	✗
Tuki ftplle	✓	✗
Tuki koukkukomennoille	✓	✓
Tuki koukkuskripteille	✗	✓
Pystyy palauttamaan tiedostot suoraan asiakkaalle	✓	✓
Pystyy lataamaan tiedostoja suoraan selaimella	✓	✗
Hallitsee arkistoinnin paikalliselle medialle	✓	✗
Hallitsee arkistoinnin ulkoiselle palvelimelle	✓	✗
Käyttöoikeushallinta monelle käyttäjälle	✓	✗
Voi käsitellä asiakkaita, joilla on DHCP-osoite	✓	✗
Joustava käsittely ajoittain kytketyille asiakkaille	✓	✗
Sähköpostin lähetys virhetilanteissa	✓	✗
Sallii jokaisen asiakkaan erillisen konfiguraation	✓	✓
Jatkuva aktiivinen kehitys	✓	✗
Vapaa lähdekoodi	✓	✓

Kuten taulukosta näkyy, niin BackupPC:ssä on huomattavasti enemmän ominaisuuksia. Kaikki mainitut ominaisuudet eivät tietenkään ole yhtä tärkeitä. Esimerkiksi ftp-tuen puute Dirvishillä ei paljoa paina, mutta web-käyttöliittymän puute painaa sitäkin enemmän. Käyttöoikeushallinta, DHCP-asiakkaat ja mahdollisuus valita ja ladata varmuuskopioituja tiedostoja itselleen

suoraan selaimella ei yksinkään kaada Dirvishiä, mutta kallistaa kylläkin vaakalautaa jonkin verran BackupPC:n suuntaan.

Dirvishin tulevaisuus näyttää myös epävarmemmalta kuin BackupPCn. Dirvishin pääkehittäjä Jonathan Schultz kuoli maaliskuussa 2004 [22], ja viimeisin vakaa versio ohjelmasta on tätä kirjoitettaessa toukokuulta 2005. BackupPC puolestaan on aktiivisemmin kehitetty, ja viimeisin vakaa versio on heinäkuulta 2010.

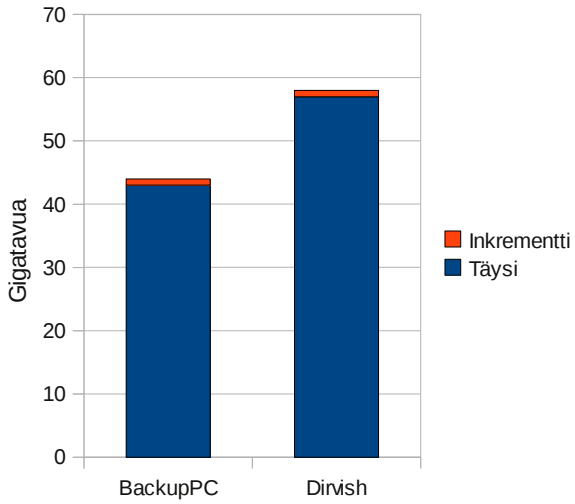
BackupPCllä on yksi punainen ruksi johtuen kokkuskriptien puutteesta. Tämä ominaisuus olisi hyödyllinen esimerkiksi tietokantojen käsittelyä varten. Tämän ominaisuuden puute on kuitenkin mahdollista kiertää esimerkiksi koukkukomennolla ja asiakaspuoleisella skriptillä.

7.3 Suorituskyky

Ominaisuuksien lisäksi on aina mukava tietää, kuinka tehokkaasti ohjelma suorittaa jonkun tehtävän. Kun puhutaan varmuuskopiointisovelluksista niin ”tehokkaasti” tarkoittaa lähinnä, kuinka *nopeasti* ja kuinka suurella *tilankäytöllä*. Seuraavat testit on ajettu palvelimella, jossa on yksiytiminen AMD Sempron LE-1200 -suoritin ja gigatavun verran muistia. Itse varmuuskopiot on otettu ulkoiselle 3,5” USB 2.0-kovalevylle, joka oli tyhjennetty ja formatoitu reiserfsllä juuri ennen testien aloittamista.

7.3.1 Tilankäyttö

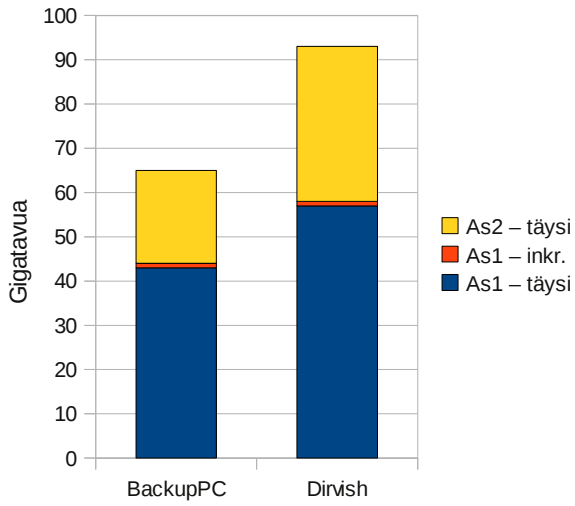
Dirvishin ja BackupPCn käytännön eron selvittämiseksi, tehtiin yksi täydellinen varmuuskopio työpöytäkäytössä olevalta koneelta ja pari päivää myöhemmin yksi inkrementaalinen varmuuskopio. Varmuuskopioitavan datan määrä oli ensimmäisellä kerralla 57 gigatavua ja toisella kerralla noin gigatavun verran enemmän. BackupPCllä käytössä oli rsync tiedonsiirtoon ja pakkausasetus oli oletuksella, eli 3:lla.



Kuva 2: Tilavaatimusvertaus

Dirvish vei 57 + 1 gigatavua tilaa. BackupPC vei 43 + 0,8 gigatavua eli noin 75% siitä mitä Dirvish.

BackupPC käyttää yhteistä tiedostovarastoa kaikille säilyttämille tiedostoille en ja käyttää kovia linkkejä kaikkiin identtisiin tiedostoihin riippumatta siitä, miltä asiakkaalta tiedosto on kotoisin. Tästä syystä olisi kiinnostavaa tietää, miltä näyttäisi, jos vielä laittaisi toisen asiakkaan mukaan. Otettiin varmuuskopiot toiselta työasemalta, jolla oli sama käyttöjärjestelmä (Kubuntu 10.04) kuin ensimmäiselläkin. Toisella asiakkaalla oli 35 gigatavua varmuuskopioitavaa dataa ja kaikki asetukset kuten ensimmäisessäkin testissä.



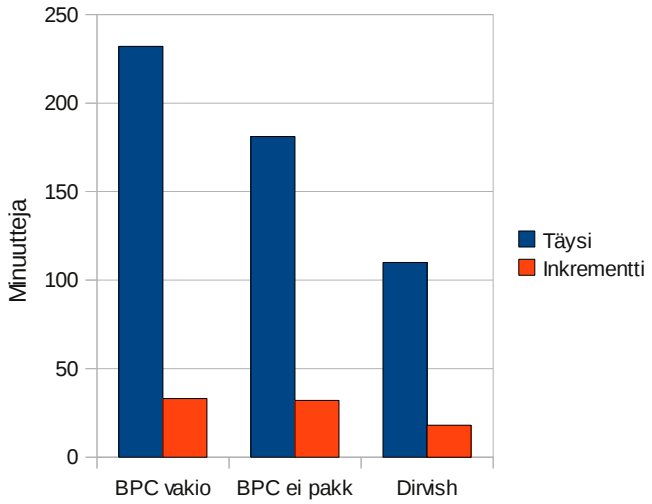
Kuva 3: Tilavaatimusvertaus kahdella asiakkaalla

Dirvishin tilanvaraus oli – odotetusti – 35 gigaa suurempi, kun 35 gigaa lisättiin. BackupPCn varaus kasvoi vain 21 gigaa. Tulos ei ollut yllättävä, mutta kuitenkin huomattava. Jos katsotaan uutta asiakasta, niin BackupPC vie vain 60 % niin paljon kovalevytilaa kuin Dirvish. Jos kokonaisuutta katsotaan, luku on 70 % (viiden prosenttiyksikön lasku). Tästä huomataan myös, että samanlaisten asiakkaiden määrän kasvaessa, BackupPCn etumatka voi kasvaa hyvinkin suureksi. Tämä voi olla oleellista varsinkin, jos varmuuskopioidaan työasemia. Myös samaa käyttöjärjestelmää käyttävät palvelimet hyötyvät tästä.

7.3.2 Nopeus

BackupPC voitti selkeästi tilankäyttötaistelun: osittain pakkauksensa ansiosta, osittain tiedostojen yhteiskäyttönsä ansiosta. Pakkaaminen kuitenkin vie oman aikansa ja identtisten tiedostojen löytäminen isosta poolista on myös vaativampaa kuin jos otetaan huomioon vain jokaisen tietokoneen muuttumattomat tiedostot. Millä tavalla tämä näkyy nopeuserona? Voisi olettaa, että Dirvish on nopeampi, mutta oleellinen kysymys tässä kuuluu: ”Kuinka paljon?”

Onneksi sekä Dirvish että BackupPC tallentavat nämä tiedot lokeihinsa, joten on helppo tarkistaa, kuinka kauan edellisessä testeissä käytetyt operaatiot kestivät. Tämän lisäksi kiinnostaa tietää, kuinka suuri osa nopeuserosta riippuu pakkaamisen käytöstä ja kuinka suuri osa riippuu muista tekijöistä, joten yksi lisätesti, jossa saman työaseman data otetaan BackupPC:llä talteen ilman minkäänlaista pakkausta, on tarpeen. Kaikki nämä testit on suoritettu nopeassa gigabitin lähiverkossa, jotta kaistanleveys ei muodostuisi pullonkaulaksi, vaan todella pystyttäisiin mittaamaan sovellusten välistä nopeuseroa.



Kuva4: Nopeuserot

Dirvish oli odotetusti nopeampi. Ainoa yllätys oli, että se oli niin paljon nopeampi. BackupPC suoritti oletusasetuksilla 57 gigatavun kopioinnin ajassa 3 tuntia 52 minuuttia. Ilman pakkausta aikaa kului minuutin päälle 3 tuntia. Dirvish sen sijaan käytti aikaa vain 1 tunti 50 minuuttia, eli selvisi urakasta alle puolet siitä ajasta, mitä BackupPC käytti, kun pakkaus oli päällä. Osasyynä tähän saattaa olla, että käytetty palvelin on kaikkea muuta kuin vauhtihirmu. Yksiytiminen 2,1 GHz:n Sempron rokottaa sitä sovellusta, joka tarvitsee enemmän prosessoritehoa.

Inkrementaalisissakin kopioinnissa Dirvish oli nopeampi – joskaan ei tuplasti nopeampi. Dirvishiltä toimitukseen meni 18 minuuttia. BackupPCltä taas 33 minuuttia pakkauksen kanssa ja 32 ilman. Tässä tapauksessa huomataan, että pakkauksen poiskytkemisen merkitys pienenee huomattavasti. Tämä ei ole yllättävää, koska inkrementaalisessa varmuuskopiossa muutoksia ei yleensä ole kovin paljon. Siten pakattavaa ei ole kovin paljon, vaan suurin osa työstä tulee tiedostojen läpikäynnistä ja vertailemisesta.

Tuloksista nähdään myös, että siirtonopeudet eivät ole päätä huimaavia ja että sisäverkon nopeus tuskin tulee vaikuttamaan oleellisesti järjestelmään. Paras siirtonopeus nopeammalla ohjelmalla ei ollut kuin 8,6 MB/s (eli noin 69 Mb/s). Rajoittavaksi tekijäksi jää lähinnä laajakaistan nopeus, mikäli var-

muuskopiointi tapahtuu Internetin välityksellä. Kokeilematta jäi mahdollisen tiedostojärjestelmän vaikutus asiaan. Dirvishin käyttäjät ovat kuitenkin yleisesti sitä mieltä, että ext2 ja ext3 eivät ole hyviä vaihtoehtoja, vaan suosivat reiserfsää. [23.]

7.4 Nopeustestejä BackupPCllä

Tässä vaiheessa BackupPC vaikuttaa selkeästi lupaavammalta ehdokkaalta. Se hävisi kuitenkin selkeästi nopeustaistelun. Tämä herättää kysymyksen: ”Onko BackupPC *tarpeeksi* nopea meille?” Jos vastaus tähän kysymykseen on kyllä, niin voimme turvallisin mielin ottaa sen käyttöön. Jos vastaus on ei, niin pitää itse implementoida ne tärkeät ominaisuudet, jotka Dirvishiltä puuttuu. Lisäksi pitää jollain tavalla kiertää ne puuttuvat ominaisuudet, jotka eivät ole aivan välttämättömiä.

Vastauksen saamiseksi tehtiin kaksi testiä. Ensimmäisessä testissä otettiin viikon ajan varmuuskopioita Internetin kautta Saksassa sijaitsevalta palvelimelta, jossa varmuuskopioitavaa oli noin 20 gigatavua. Toisessa testissä otettiin varmuuskopioita paikalliselta palvelimelta, jossa oli paljon (165 gigatavua) dataa. Tarkoitus oli tehdä varmuuskopiointi lähiverkon kautta, mutta inhimillisen erehdyksen takia *tämäkin* toiminto suoritettiin Internetin kautta. Tällä tavalla saatiin ainakin pätevän vastauksen kysymykseen: ”Voiko Internetiä käyttää tällaisiin datamääriin?”

Taulukko2: Viikon varmuuskopiot kahdelta palvelimelta

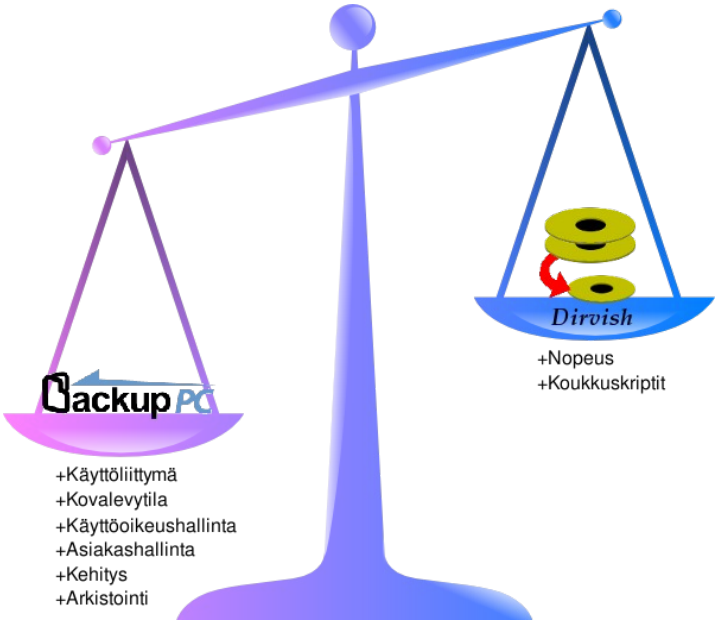
Päivä	Kesto (Saksa, 20 GT)	Kesto (Suomi, 165 GT)
Ma	4h 46min	42h 12min
Ti	30 min	Edellinen vielä menossa
Ke	20 min	41 min
To	59 min	1h 13min
Pe	1 h 46 min	1h 15min
La	16 min	43 min
Su	20 min	43 min

Ensimmäinen varmuuskopio vie tietysti kauan, koska kaikki data on siirrettävä. Tämä on kuitenkin kestettävä vain kerran, koska sen jälkeen pärjätään muutosten siirrolla. Kun tämä otetaan huomioon, niin ajat ovat oikein hyviä ja jopa perjantain päivitys on täysin hyväksyttävän rajoissa. Ajoista näkee myös, milloin suurimmat muutokset ovat tapahtuneet palvelimella. Suuret muutokset tarkoittavat enemmän siirrettävää dataa eli pitempiä aikoja. Viikonloppuna ei paljon töitä tehdä. Siten muutoksetkin ovat pienet.

Testin lopputulos on, että BackupPC on tarpeeksi nopea meidän käyttöömme. Se riittää kattamaan nykyiset tarpeemme sekä kaikki lähitulevaisuuden tarpeemme.

7.5 Vertailun lopputulos

BackupPC nousi tämän vertailun voittajaksi huomattavasti laajempien ominaisuuksiensa ansiosta. Dirvish on myös pätevä ohjelma, mutta se ei yllättänyt meitä positiivisesti yhtä useasti kuin BackupPC.



Kuva5: Vertailun lopputulos

Dirvishin nopeus ei ollut tarpeeksi painava syy valita se BackupPCn yli. Valmis webkäyttöliittymä, arkistointitoiminto, käyttöoikeushallinta sekä kovalevyn tilankäyttö painoivat vaakakupissa enemmän.

8 PALVELINPUOLEN IMPLEMENTAATIO

Tämän luvun tarkoitus on dokumentoida, millaisia ratkaisuja on käytetty ja millä tavalla ne on toteutettu käytännössä, jotta olisi helppoa laajentaa, korjata tai implementoida sitä uudelleen tarpeen niin vaatiessa.

8.1 Laitteisto

Palvelimeksi valittiin HP:n Proliant ML110 G6 sen kohtuullisen hinnan ja hyvän laajennettavuuden takia. Kaksiytiminen 2,33 GHz:n prosessori ei ole mikään vauhtihirmu, mutta varmuuskopiopalvelimelle kuitenkin riittävä. Muistin määrä laajennettiin 4 gigatavuun, lisättiin HP:n RAID-ohjain johon asennettiin 4 kappaletta 2 teratavun kovalevyä RAID5-moodissa. Palvelimen mukana tuli myös 250 GT:n kokoinen kovalevy jolle asennettiin käyttöjärjestelmä. Tämä konfiguraatio arvellaan pystyvän täyttämään funktionsa usean vuoden ajan, vaikka datan määrä kasvaakin.

8.2 Ohjelmisto

Käyttöjärjestelmäksi valittiin Debianin viimeisin vakaa versio (tätä kirjoittaessa, ”Lenny”). Koska mikään koneella käytettävä ohjelmisto ei vaadi 32-bittistä käyttöjärjestelmää, valittiin 64-bittinen sen nopeuden takia. RAID-levyt konfiguroitiin käyttämään LVMää. Aluksi tallennusta varten varattiin 1 terabyten pala LVMstä ja tätä alustettiin reiserfs:llä.

BackupPC sekä pari sen toimintaan liittyvää pakettia asennettiin suoraan Debianin pakettivarastosta komennolla

```
apt-get install backuppc libfile-rsync-perl par2
```

Tämä toiminto asentaa myös kaikki riippuvuudet, kuten Apachen, sekä ajaa konfigurointiskriptin, joka asettaa ja kertoo BackupPC:n pääkäyttäjän (jonka käyttäjänimi on ”backuppc”) salasanan. Tätä salasanaa ei tarvitse ottaa ylös, koska sen korvaa helposti omalla.

8.3 Asetukset

BackupPC toimii periaatteessa tämän jälkeen. Mutta mikäli haluaa sen toimivan optimaalisesti, niin voi olla syytä muuttaa konfiguraatiota. Eräs konfiguraatiomuunnos, joka usein mainitaan on mod_perl:in käyttöönotto. Dokumentaation mukaan se voi parantaa suorituskykyä huomattavastikin. Kun sitä testattiin kokeiluympäristössä, tulokset olivat kuitenkin niin heikot, että sitä ei varsinaiseen versioon otettu mukaan.

8.3.1 Käyttäjien luonti

BackupPCllä ei voi varsinaisesti luoda käyttäjiä sen graafisen käyttöliittymän kautta, koska BackupPC ei itse hoida käyttäjien tunnistusta, vaan on delegoinut tämän tehtävän Apachelle. Uuden käyttäjän voi luoda pääkäyttäjä komennolla:

`htpasswd -s /etc/backuppc/htpasswd` *`käyttäjänimi`*

Tällä tavalla voi myös resetoida jo olemassa olevan käyttäjän salasanan. Voi siis olla hyvä idea aloittaa korvaamalla käyttäjän ”backuppc” automaattisesti asetettu salasana. Valitsin -s saa htpasswd:n tallentamaan salasanan SHA1-hajautusalgoritmin avulla, -m puolestaan saa sen käyttämään MD5-algoritmiä. Jos kumpaakaan ei käytä, niin htpasswd käyttää vanhaa crypt-menetelmää, jolloin vain ensimmäiset 8 merkkiä salasanasta otetaan mukaan. Nämä salasanat ovat myös erittäin helposti murrettavissa, jos pahaa aikova ihminen pääsee niihin käsiksi. Tätä ei siis suositella. [24.]

8.3.2 Tiedostojärjestelmän asetukset

BackupPC tallentaa oletuksena tiedostonsa `/var/lib/backuppc` -hakemistoon. Tämän oletusarvon voi muuttaa muokkaamalla konfiguraatiotiedostosta `/etc/backuppc/config.pl` arvoa `$Conf{TopDir}`. Vaihtoehtoisesti voi liittää tiedostotallennusvolyymin suoraan oletuspaikkaan. Tämä toinen lähestymistapa valittiin. Hakemistossa olevat alahakemistot pitää en-

sin siirtää tallennusvolyymin juureen. Jotta tämä onnistuisi, pitää backuppc-daemoni hetkeksi sulkea komennolla:

```
/etc/init.d/backuppc stop
```

Kun tiedostot on siirretty, niin tiedostoon `/etc/fstab` pitää lisätä seuraavanlainen rivi:

```
/polku/tallennusvolyymille /var/lib/backuppc reiserfs \
defaults,notail,noatime 0 2
```

Noatime-optiota käytetään levyoperaatioiden nopeuttamiseksi, samoin notailia suositellaan käytettäväksi nimenomaan reiserfsn kanssa nopeuden takia. Tämän jälkeen volyymin voi liittää paikoilleen ja backuppc-daemoni käynnistää uudelleen. [25.]

8.3.3 SSL sertifikaatti -avainparin luonti

Kun salasanoja kirjoitetaan selaimeen ja lähetetään verkon yli, on aina turvallisempaa, jos yhteys tapahtuu käyttäen SSLää, joten haluamme konfiguroida Apachea käyttämään tätä.

Ensityöksemme tarvitsemme sertifikaatin ja avaimen. Koska päämäärämme on ainoastaan saavuttaa salattu yhteys eikä verifioida palvelua, niin itse allekirjoitettu sertifikaatti riittää. Sertifikaatin luonti tapahtuu seuraavasti. Ensin luodaan salasanasuojattu avain

```
openssl genrsa -des3 -out server.key 4096
```

Tämän jälkeen luodaan sertifikaatin allekirjoituspyyntö vastaamalla kysymyksiin, joita esitetään, kun ajetaan komento

```
openssl req -new -key server.key -out server.csr
```

Kannattaa huomioida, että kysymykseen "Common Name (eg, YOUR name)" pitää vastata joko palvelimen domain-nimellä tai IP-osoitteella. Muut

kysymykset ovat hyvin selkeitä. Tämän jälkeen allekirjoitetaan sertifikaatti komennolla

```
openssl x509 -req -in server.csr -signkey server.key \
-out server.crt
```

Tämän jälkeen pitää tehdä avaimesta versio, joka ei vaadi salasanaa. Salasanaa vaativa avain pistetään talteen ja kaikille luoduille tiedostoille asetetaan sellaiset oikeudet, että vain pääkäyttäjä pääsee niiden käsiksi:

```
mv server.key ./server.key.secure
openssl rsa -in server.key.secure -out server.key
chown root: ./server.*
chmod 600 ./server.*
```

Näin luotu sertifikaatti sallii SSLn käytön. [26.]

8.3.4 Apachen konfigurointi

Tämän jälkeen Apachea pitää konfiguroida niin, että se käyttää äsken luotua sertifikaatti -avainparia. Mikäli hakemistoa `/etc/apache2/ssl` ei ole olemassa, se luodaan ja tiedostot `server.key` ja `server.crt` kopioidaan siihen. Tämän jälkeen SSL otetaan käyttöön komennoilla:

```
a2enmod ssl
ln -s /etc/apache2/sites-available/default-ssl \
/etc/apache2/sites-enabled/000-default-ssl
```

Luodaan hakemistot `/var/www` ja `/var/www-ssl` , mikäli ne eivät ole olemassa. Tiedostosta `/etc/apache2/sites-available/default-ssl` muokataan seuraavat rivit:

```
DocumentRoot /var/www-ssl/
SSLEngine on
SSLCertificateFile /etc/apache2/ssl/server.crt
SSLCertificateKeyFile /etc/apache2/ssl/server.key
```

Lopuksi tiedostosta `/etc/backuppc/apache.conf` pitää poistaa kommentti riviltä

```
SSLRequireSSL
```

Tämän jälkeen Apachea voi käynnistää uudelleen. Nyt BackupPCn pitäisi toimia vain ja ainoastaan SSLn kanssa.

8.3.5 SSHn konfigurointi

Suositeltu tapa käyttää BackupPCtä on rsyncin avulla. Koska rsync toimii SSHn avulla tämä täytyy konfiguroida. Ensimmäinen askel on avainparin luonti (ilman salasanaa) komennolla

```
ssh-keygen -t rsa
```

Tämä luo tiedostot `id_rsa` ja `id_rsa.pub` hakemistoon `~/.ssh` . Nämä tiedostot täytyy myös kopioida käyttäjälle ”backuppc” vastaavalle paikalle ja pääkäyttäjälle. Omistusoikeudet täytyy lisäksi asettaa omistajan mukaiseksi toimivuuden takaamiseksi.

Jokaiselle asiakkaalle, joka halutaan BackupPCn piiriin, pitää kopioida tiedoston `id_rsa.pub` sisältö (eli palvelimen avainparin julkinen osa) ja liittää se tiedostoon nimeltä `/root/.ssh/authorized_keys`, jotta SSH-yhteys voidaan avata ilman salasanaa.

Tämän varmistuksen lisäksi, kun palvelin ensimmäisen kerran ottaa yhteyden asiakkaalle, SSH varoittaa, että asiakas on tuntematon ja kysyy, mikäli halutaan lisätä asiakasta tunnettujen listalle. Ongelma on, että tämä varoitus / vahvistus ei siirry SSHn ja BackupPCn välillä. Lopputulos on, että kaikki varmuuskopiot epäonnistuvat timeout-virheellä, kunnes joku manuaalisesti käy lisäämässä asiakkaan. Koska tämä ei ole toivottua, muutetaan SSHn toimintaa muokkaamalla tiedostoa `/etc/ssh/ssh_config` lisäämällä tiedoston loppuun rivi

```
StrictHostKeyChecking no
```

Lisätä voi myös rivin

```
Ciphers arcfour,blowfish
```

Rivi saa SSHn käyttämään Arcfour-nimistä salausta, jonka etu muihin nähden on sen nopeus. SSHn käyttämistä salauksista vain Blowfish pääsee nopeudessa lähellekään. Vaikka Arcfouria ei ole testattu yhtä paljon kuin Blowfishiä, sen tietoturvan pitäisi olla täysin riittävä. [27.]

8.3.6 Postfixin konfigurointi

BackupPC tukee sähköpostien lähetystä virhetilanteissa, mikä on oikein mukava asia, jos ei halua säännöllisesti väijyä sen tekemisiä. Sähköpostien lähetykseen se käyttää Postfixiä tai sen korviketta ja tämä on konfiguroitava, ennen kuin lähetys toimii. Debian käyttää oletuksena Exim4:ää suorittamaan Postfixin työtä ja sen konfiguraatiota muuttaa helpoiten ajamalla

```
dpkg-reconfigure exim4-config
```

On monta eri tapaa konfiguroida Exim4sta, jotta se toimisi sähköpostien lähettämiseen. Nyt esitelty tapa hyödyntää palveluntarjoajan SMTP-palvelinta. Konfiguraatio-ohjelma esittää sarjan kysymyksiä, joihin kannattaa vastata seuraavalla tavalla:

Taulukko 3: Exim4:n konfigurointi

Kysymys	Vastaus
General type of mail configuration:	mail sent by smarthost; received via SMTP or fetchmail
System mail name:	Oletus on OK
IP-addresses to listen on for incoming SMTP connections:	127.0.0.1
Other destinations for which mail is accepted:	Oletus on OK
Machines to relay mail for:	Aliverkko, josta postia saa lähettää
IP address or host name of the outgoing smarthost:	Palveluntarjoajan SMTP-palvelimen osoite

Hide local mail name in outgoing mail?	Yes
Visible domain name for local users:	Joku ulospäin näkyvä oma palvelin
Keep number of DNS-queries minimal?	No
Delivery method for local mail:	mbox format in /var/mail/
Split configuration into small files?	No

Tämän jälkeen järjestelmän toimivuutta kannattaa kokeilla esimerkiksi komennolla

```
echo ”Testaus” | mail –s ”Testi” –v sähköpostiosoite
```

Jos kaikki toimii, niin postilaatikkoon pitäisi tulla viesti. Jos ei, niin -v optio antaa vihjettä siitä, mikä meni pieleen.

8.3.7 Arkistoinnin implementointi

BackupPCn mukana tulee valikoima komentorivityökaluja, joita voi käyttää cronjobien avulla. Arkistointiin sopiva työkalu `BackupPC_archiveStart` löytyy hakemistosta `/usr/share/backuppc/bin` . Se ei ihan sellaisenaan sovi käyttöömme, sillä oletuksena se ottaa vain ”raakoja” ei-pakattuja arkistoja. (Tämä on bugi joka on korjattu versioon 3.2.0, mutta Debian Lennyn mukana tulee versio 3.1.0.) Tämän asian korjaamiseksi tiedosto pitää avata ja rivi

```
compression => $bpc->{Conf}{CatPath}
```

pitää muokata muotoon:

```
compression => '/bin/gzip'
```

Lisäksi päätteen määrittelevä muuttuja `compext`, joka oletuksena on `.raw`, pitää muuttaa niin, että se vastaa käytettyä pakkausta (`.bz2` tai `.gz`). Kokeillessani näitä eri asetuksia tulin siihen johtopäätökseen, että gzip tarjoaa parhaan kompromissin nopeuden ja pakkaustehokkuuden välillä.

Tämän jälkeen tiedosto `/etc/backuppc/config.pl` tulee avata ja tarkistaa, että kaikki arkistointiin liittyvät muuttujat ovat kohdallaan. Muuttuja

`$Conf{ArchiveDest}`, pitää osoittaa siihen paikkaan, mihin ulkoinen kovalevy on liitetty. Sekä `$Conf{ArchivePar}` että `$Conf{ArchiveSplit}` tulisivat olla '0'.

Tämän jälkeen pitää vain luoda skripti, joka ajaa jokaiselle asiakkaalle komennon ja cronjobi, joka käynnistää sen kerran kuukaudessa. Koska arkistointi vie aikaa, niin arkistointiajankohdaksi valittiin jokaisen kuukauden ensimmäinen lauantai. Skripti on seuraavanlainen:

```
#!/bin/bash
ARCHIVE_HOST=”localhost”
clients=`cat /etc/backuppc/hosts | sed 's/#.*//;/^$/d' \
| tail -n +2 | grep -v ^$ARCHIVE_HOST$b | \
sed 's/[ \t].*//' | tr '\n' ' '`
su backuppc -c ”/usr/share/backuppc/bin/BackupPC_archiv\
eStart $ARCHIVE_HOST backuppc $clients”
exit 0
```

Lopuksi pitää varmistua siitä, että arkistot ovat salattuja, jotta tiedostoihimme ei pääsisi käsiksi pelkästään nappaamalla ulkoinen kovalevy povitaskuun. Kovalevyn salauksen voi toteuttaa monella eri tavalla. Koska LVM-kryptaus on meillä jo yleisessä käytössä oli luontevaa käyttää se tähänkin. Tämän käyttöönotto tapahtuu seuraavasti:

Ulkoisen kovalevyn kiinnittämisen jälkeen se tulee ensin osioida yhdeksi osioksi käyttämällä komentoa `fdisk`. Sen jälkeen voi luoda volyymiryhmän komennolla

```
vgcreate ryhmän_nimi /kovalevyn/osio
```

ja heti perään loogisen volyymin komennolla

```
lvcreate --extents 100%FREE -n volyyminimi ryhmän_nimi
```

Salausavaimen voi asettaa komennolla

```
cryptsetup luksFormat /dev/ryhmän_nimi/volyyminimi
```

Lopuksi salattu volyymi pitää avata (äsken luodulla avaimella), sovittaa tiedostojärjestelmään, alustaa esimerkiksi ext3:lla ja liittää siihen kohtaan, johon määriteltiin, että arkistot otetaan

```
cryptsetup luksOpen /dev/ryhmän_nimi/volyyminimi sovitus
mkfs.ext3 /dev/mapper/sovitus
mount /dev/mapper/sovitus /arkistojen/liitoskohta
```

Kun levy halutaan poistaa, pitää ensin poistaa liitos ja sitten sovitus

```
umount /dev/mapper/sovitus
cryptsetup remove sovitus
```

8.3.8 Gitin implementointi

Koska git toimii tavalla, joka ei välttämättä tee hyvin yhteistyötä rsync-kaltaisten ohjelmien kanssa, siitä päätettiin tehdä täysin erillinen implementaatio. Tämän tekemiseen on kaksi vaihtoehtoa: asiakaspuolelta käynnistetty `git push --mirror` tai palvelinpuolelta käynnistetty `git fetch`, joka suoritetaan `git clone --bare` -komennon jälkeen. Jälkimmäinen vaihtoehto vaatii vähemmän asennusta ja konfiguraatiota palvelimella, ja se jättää palvelimen tehtäväksi ajoittaa varmuuskopiot, joten tämä katsottiin parhaaksi.

Palvelimelle luotiin ”gitmirror”-niminen käyttäjä, jonka kotihakemistoon liitettiin 100 gigatavun looginen LVM-volyymi. Käyttäjälle luotiin myös ssh-avainpari, jonka avulla varsinainen git-palvelin tunnistaa käyttäjän. Tämän jälkeen kirjoitettiin skripti, joka ottaa varmuuskopiot. Skriptin toiminta on yksinkertainen, koska sen ei tarvitse välittää muista asioista kuin siitä, onko kyseessä uusi vai vanha tietovarasto. Pseudokoodina se näyttäisi tältä:

```
get all repositories from gitserver
for each repo in repositories do
  if repo exists then
    git fetch repo
  else
    git clone repo
done
```

Lopuksi asetettiin cron ajamaan skriptiä joka toinen tunti päivisin gitmirror -käyttäjänä.

9 ASIAKASPUOLEN IMPLEMENTAATIO

Kuten kohdassa 8.3.5 jo mainittiin, niin palvelimen julkinen ssh-avain pitää lisätä asiakkaan `/root/.ssh/authorized_keys` -tiedostoon. Tämä ei kuitenkaan vielä riitä, mikäli asiakkaalla on käytössään jonkinlainen tietokanta, versionhallintajärjestelmä tai jos se toimii LDAP -palvelimena. Tätä varten pitää kopioida näitä huomioon ottava bashskripti `backup.sh` (katso liite 1) hakemistoon `/usr/local/sbin` sekä konfiguroida se niin, että asetukset vastaavat asiakkaan tarpeita.

Mikäli erikoistoimenpiteitä vaativia ohjelmia ei asiakkaalla ole käytössä eikä halua yksilöllistää asiakkaan käsittelyä BackupPCn puolesta, niin skriptin tilalle voi myös laittaa ”tyhjän” skriptin, joka ei tee muuta kuin lopettaa itseään statuksella 0.

9.1 Skriptin ominaisuudet

Skriptin idea on, että varmuuskopiointi ja palautus pitäisi sujua ongelmitta ja ilman manuaalista työtä niin pitkälle kun on mahdollista. Tärkein ongelma, mitä tietenkin pyritään välttämään, on datan korruptointi. Mikäli skripti on oikein konfiguroitu, kaikki varmuuskopiot pitäisi pystyä ottamaan ja palauttamaan yhtä helposti riippumatta siitä, onko asiakkaalle asennettu jotain ”ongelmaohjelmia” (joita käsiteltiin luvussa 4) vai ei.

Tällä hetkellä skripti pystyy käsittelemään MySQL-tietokannat, PostgreSQL-tietokannat, OpenLDAP ja sen BerkeleyDB-tietokannat sekä subversionin tietovarastot. Kaikkia paitsi subversionia varten on lisäksi olemassa kaksi eri varmuuskopiointistrategiaa: looginen dumppi tai LVM-valokuva. Subversionin tietovarastot kopioidaan aina loogisen dumpin avulla.

9.1.1 MySQL

Skripti pystyy käsittelemään MySQL-tietokannat riippumatta siitä, mitä tietokantamoottoria kyseisen kannan taulut käyttävät. Se pystyy käyttämään optimoitua transaktioihin perustuvaa strategiaa loogisen dumpin tekemiseen, jos kaikki tietokannan taulut ovat InnoDB-tauluja. Muuten se pelaa varman päälle ja käyttää taulujen lukkoja. Mikäli käytössä on LVM valokuva, niin kaikista tauluista otetaan lukulukko, joka vapautetaan heti kuvan valmistumisen jälkeen.

MySQLää varten pitää konfiguroida neljä muuttujaa:

Taulukko 4: Skriptin MySQL -kohtaiset asetukset

Muuttuja	Toiminto
`MYSQL`	Onko MySQL käytössä vai ei?
`MYSQL_PASSWORD`	MySQL:n rootsalasana, jota tarvitaan tämäntapaisiin tehtäviin. (On hyvä idea päästä vain pääkäyttäjä skriptiin käsiksi.)
`MYSQL_INIT`	MySQL:n käynnistysskriptin polkunimi. Jakelukohtainen asetus joka on yleensä: `/etc/init.d/mysql`
`MYSQL_DUMPDIR`	Vapaasti valittava paikka, mihin dumppi luodaan (jos luodaan).

Tätä viimeistä muuttuja (sekä kaikkia muita `_DUMPDIR` -muuttujia) valitessa pitää huomioida muutama seikka:

- Sen täytyy osoittaa paikkaan, jossa on niin paljon vapaata tilaa, että dumppi mahtuu siihen.
- Sen täytyy osoittaa paikkaan, joka todella varmuuskopioidaan.
- Hakemiston ei tarvitse olla olemassa – se luodaan tarpeen mukaan.
- Hakemiston ei tulisi sisältää mitään (muuta) dataa – se poistetaan tarpeen mukaan.

- Samaa hakemistoa ei tulisi käyttää kahdelle `_DUMPDIR` -muuttujille. Se sekoittaa hakemistojen poiston.

- Näihin hakemistoihin luotuja dumppeja tulisi palauttaa asiakkaalle, jos – *ja vain jos* – halutaan palauttaa tietokanta dumppiin säilytettyyn tilaan.

9.1.2 PostgreSQL

PostgreSQLssä ei ole sitä ongelmaa, että käytössä saattaisi olla erilaisia tietokantamoottoreita, ja dumppityökalukin toimii yksinkertaisesti ja suoraviivaisesti. LVM-valokuvaa varten ei kuitenkaan löydy mitään yksinkertaista MySQLmäistä tapaa lukita taulut, vaan koko kanta on otettava alas valokuvan ajaksi, jotta kanta olisi varmuudella järkevässä tilassa myös mahdollisen palautuksen jälkeen.

PostgreSQLn varmuuskopiointia ohjaavat seuraavat muuttujat:

Taulukko 5: Skriptin PostgreSQL -kohtaiset asetukset

Muuttuja	Toiminto
`POSTGRE`	Onko PostgreSQL käytössä vai ei?
`POSTGRE_USER`	PostgreSQLää käyttää aina tietty käyttäjä, (yleensä ”postgres” mutta saattaa vaihdella).
`POSTGRE_INIT`	PostgreSQL:n käynnistysskriptin polku. Jakelukohtainen asetus. Usein: `/etc/init.d/postgresql`
`POSTGRE_DUMPDIR`	Vapaasti valittava paikka mihin dumppi luodaan (jos luodaan). Samat rajoitukset kuin muissa `_DUMPDIR` -muuttujissa pätevät.

9.1.3 LDAP

LDAP ja sen tietokannat eivät ole läheskään yhtä yleisiä tuttavuuksia kuin edellä mainitut. Sen huomaa myös tiedon tarjonnasta. Suurimmat erimielisyydet näyttäisivät tulevan aiheesta: ”Voiko slapcat (loogisen) dumpin suorit-

taa kuumana vai ei?” Vastaus tähän on kuitenkin: ”Kyllä, nykyään (OpenLDAP versio 2.2 tai uudempi) voi.” [28.] Aivan kuten PostgreSQL:llä, valokuvan ottaminen vaatii kannan pysäyttämisen.

LDAP:lla on skriptissä seuraavat asetukset:

Taulukko 6: Skriptin LDAP -kohtaiset asetukset

Muuttuja	Toiminto
`LDAP`	Onko LDAP käytössä vai ei?
`LDAP_CONFIG`	LDAP:lla on konfiguraatiotiedosto (yleensä nimellä slapd.conf), jonka sijainti voi vaihdella jakeluiden välillä.
`LDAP_INIT`	LDAP:n käynnistysskriptin polku. Jakelukohtainen asetus joka on yleensä: `/etc/init.d/slapd`
`LDAP_DUMPDIR`	Vapaasti valittava paikka mihin dumppi luodaan (jos luodaan). Samat rajoitukset kuin muissa `_DUMPDIR` -muuttujissa pätevät.

9.1.4 Subversion

Subversion ja sen nykyään yleensä käytetty FSFS -tiedostotallennusmenetelmä eroaa edellisistä tietokannoista sillä, että se ei pyöritä mitään daemonia, jota voidaan sammuttaa tai lukita ja sillä aikaa ottaa LVM-kuva. Vaikka on periaatteessa mahdollista väliaikaisesti estää tavallisimmat yhteydenottotavat (ssh ja apache + https), niin tämä on kömpelö tapa ja aiheuttaa muita ongelmia, koska varmuuskopiointisovelluksen itse käyttämä rsync luottaa myös ssh:n toimivuuteen. Tämä tapa ei myöskään estä mahdollista paikallista käyttäjää koskemasta tietovarastoon.

Toinen huomioon otettava asia on, että tietovarastot eivät välttämättä sijaitse missään keskitetyssä paikassa, vaan niitä voi periaatteessa luoda kuinka monta tahansa ihan mihin vaan. Näistä syistä subversionin tietovarastoista otetaan dumppi käyttäen svnadmin-työkalun hotcopy-toiminnallisuutta riippumatta siitä, onko käytössä LVM-valokuvastrategia vai ei.

Taulukko 7: Skriptin subversionkohtaiset asetukset

Muuttuja	Toiminto
SVN	Onko subversion käytössä vai ei?
SVN_REPOS	Lista kaikkien talteen otettavien tietovarastojen poluista.
SVN_DUMPDIR	Vapaasti valittava paikka mihin dumppi luodaan. Samat rajoitukset kuin muissa _DUMPDIR -muuttujissa pätevät.

9.1.5 LVM valokuva

Kuten on jo selvinnyt, on mahdollista käyttää loogisen dumpin sijan LVM-valokuvastrategiaa. Tämän strategian suurin etu on nopeus. Mitä suuremmaksi tietokannat kasvavat, sitä suuremmaksi tämä etu muuttuu. Sen haittapuolena on lähinnä työläämpi konfiguraatio ja tiukemmat vaatimukset kyseisen asiakkaan konfiguraatioon. Jos haluaa käyttää tätä menetelmää, niin se vaatii, että asiakas käyttää LVMää ja että vapaata tilaa on niin paljon, ettei se lopu kesken varmuuskopioinnin aikana.

Taulukko 8: Skriptin LVM-valokuvakohtaiset asetukset

Muuttuja	Toiminto
SNAPSHOP	Käytetäänkö LVM vai dumppistrategiaa?
SNAPSHOT_NAME	Vapaasti valittava nimi, joka ei ole jo käytössä.
SNAPSHOT_SIZE	Kuvan koko. Pitää olla niin iso, että tila ei lopu kesken ja niin pieni, että mahtuu vapaaseen tilaan.
VOLUME_GROUP_DIRECTORY	Hakemisto missä loogiset volyymiryhmät sijaitsevat. Oletuksena /dev
VOLUME_GROUP_NAME	Volyymiryhmän nimi, johon valokuva liitetään. (Tällä pitää olla riittävästi vapaata tilaa.)
LOGICAL_VOLUME_NAME	Sen loogisen volyymin nimi, josta valokuva tulee ottaa.
SNAPSHOT_MOUNTPOINT	Paikka, johon kuva liitetään. Voi olla mikä paikka tahansa. Oletus: /mnt/snapshot

On huomioitavaa, että asiakkaan konfiguroinnin lisäksi myös varmuuskopiointipalvelimen puolelle pitää tehdä yksi muutos, mikäli tätä halutaan käyt-

tää. Muuttuja `$Conf{RsyncClientCmd}` määrittelee, millä komennolla varmuuskopiot otetaan. Se pitää muuttaa muotoon (muutos punaisella)

```
$sshPath -q -x -l root $host chroot /mnt/snapshot \
$rsyncPath $argList+
```

Tämä takaa sen, että rsync näkee LVM valokuvan ja että oikeat tiedostot kopioidaan oikeisiin paikkoihin niin, että palautuksen voi tehdä suoraan ilman kikkailua polkujen kanssa.

9.2 Skriptin rajoitukset

Tällä hetkellä skripti pystyy käsittelemään melkein kaikkia järjestelmiä, joita meillä on tällä hetkellä käytössä. Ainoa poikkeus ovat git-tietovarastot. Koska gitillä ei ole mitään yksinkertaista tapaa, jolla voisi tehdä dumpin tai jäädyttää tietovarastot LVM-valokuvan ajaksi, niin gitin varmuuskopiointi päätettiin suorittaa täysin tästä järjestelmästä riippumatta git-mirrorin avulla.

LVM-valokuvauksessa skripti pystyy toistaiseksi käsittelemään vain yhtä kuvaa kerralla. Jos tiedostojärjestelmä on hajautettu usealle loogiselle volyymille ja täydellisen valokuvan ottaminen vaatisi usean eri kuvan ottamista ja niiden uudelleenkokoaminen mountilla, niin skripti ei tähän pysty. Mikäli käy ilmi, että tämä ominaisuus olisi hyvin tärkeä, niin skriptiä voi laajentaa niin, että tämäkin onnistuu.

Skripti ei myöskään käsittele tietokantoja, joita ei ole täällä mainittu. Mikäli tarve muiden tietokantojen varmuuskopiointiin syntyy, niin skriptiä voi laajentaa. Sen rakenne on pyritty tekemään sellaiseksi, että laajennus onnistuisi suhteellisen vähällä vaivalla.

10 JOHTOPÄÄTÖKSET

Edellinen varmuuskopiointisovellus ei enää täyttänyt yrityksen tarpeita. Tavoitteena oli luoda varmuuskopiointijärjestelmä, joka on helppokäyttöinen,

kattaa yrityksen tämänhetkiset tarpeet ja joka on laajennettavissa mahdollisia tulevaisuuden tarpeita varten. Näissä tavoitteissa myös onnistuttiin hyvin. Ensin analysoitiin tarpeet, sitten kartoitettiin olemassa olevat sovellukset. Lopuksi päädyttiin BackupPChen sen monipuolisten ominaisuuksiensa ansiosta, ja implementoitiin se. Lisäksi kirjoitettiin skripti (katso liite 1), joka palvelinkohtaisesti huolehtii tietokannoista ja muista ongelmakohdista. Yhdessä nämä takaavat, että myös tietokannat saadaan talteen muodossa, jonka pystyy myöhemmin palauttamaan.

Varmuuskopiointi, jos jokin on alue joka kuuluu sanonnan ”Tämä kuulostaa helpolta, mutta anna kun minä selitän...”-piiriin. Ensisilmäyksellä ongelma vaikuttaa yksinkertaiselta. Meillä on tiedostoja. Niitä pitää kopioida. Kuitenkin, mitä enemmän aiheeseen paneutuu, sitä enemmän huomaa kaikenlaisia miinoja, hankaluuksia ja pikkujuttuja, jotka eivät menekään niin helposti ja jotka vaativat varmuuskopioinnilta jonkinlaisia erikoisjärjestelyitä.

Erikoisjärjestelyt eivät tietenkään ole mikään hyvä asia. Tietotekniikassa on yleensä parempi, että asiat toimivat samalla tavalla, että eri konfiguraatioita on mahdollisimman vähän ja että kaiken pystyy automatisoimaan samalla tavalla. Erilaisuus ja erikoisjärjestelyt tuottavat helposti lisää työtä. Varmuuskopiointipuolella tämä erikoistilanteiden välttäminen näyttää kuitenkin pysyvän idealistisena haaveena niin pitkälle kuin tietotekniikassa yleensä uskaltaa ennustaa asioita. Suurin syy tähän on se, että varmuuskopiointi on vain tukitoiminto ja tulee aina sellaisena pysymään. Yrityksissä, jotka ovat jollain muulla alalla kuin IT-alalla, niin se on itse asiassa tukitoimintojen tukitoiminto.

Tästä seuraa, että varmuuskopiointi tulee aina tapahtumaan kaikkien muiden sovellusten ehdoilla, eikä päinvastoin. Tietokannat tullaan aina suunnittelemaan sen mukaan, miten ne pystyvät saavuttamaan eheyden ja pysyvyyden sekä miten ne mahdollisimman tehokkaalla tavalla pystyvät suorittamaan monimutkaisia hakuja, eikä sen mukaan, että olisi olemassa jonkinlai-

nen yhteinen standardi, jonka mukaan varmuuskopioita hoidetaan aina samalla tavalla. Jos tällainen joskus luodaan, niin järjestelmävalvojat ympäri maailman olisivat ikuisesti kiitollisia. Siihen asti varmuuskopiointi tulee olemaan haastava – joskaan ei suinkaan mahdoton tehtävä.

VIITELUETTELO

[1] Magnetic_tape_data_storage. Wikipedia. [Viitattu 19.9.2010].

[2] Toigo, Jon William. The Holy grail of data storage manage ment. Upper Saddle River: Prentice Hall PTR, 2000.

[3] Do ACLs follow a file from one machine to another? [Verkkodo-kumentti] 9.5.2009. [Viitattu 2.11.2010]. Saatavissa: http://stackowerflow.com/questions/839625/where-does-windows-store-acls-and-do-acls-follow-a-file-from-one-machine-to-anoth .

[4] ACLBit – ACL Backup and Inspect Tool. [Verkkodokumentti] [Viitattu 27.9.2010]. Saatavissa: http://aclbit.sourceforge.net .

[5] Preston, W Curtis. Backup and Recovery. Sebastopol: O'Reilly Media Inc, 2007.

[6] Lewis AJ. LVM HOWTO. [Verkkodokumentti] 27.11.2006. [Viitattu 27.9.2010]. Saatavissa: http://tldp.org/HOWTO/LVM-HOWTO .

[7] Nilfs2. [Verkkodokumentti] [Viitattu 1.10.2010]. Saatavissa: http://www.mjmwired.net/kernel/Documentation/filesystems/nilfs2.txt .

[8] Mullins, Craig S. Database Administration : the complete guide to practices and procedures. Indianapolis: Pearson Education Inc, 2002.

[9] Bartholomew, Daniel. Easy Database Backups with Zmanda Recovery Manager for MySQL. Linux Journal. 2010 Sep ; 197.

[10] LDAP. Wikipedia. [Viitattu 19.10.2010].

[11] Svnadmin hothopy. [verkkodokumentti] [viitattu 15.10.2010]. Saatavissa: http://svnbook.red-bean.com/en/1.0/re33.html .

[12] Dump_(program). Wikipedia [Viitattu 12.10.2010].

[13] Rsync. Wikipedia [Viitattu 12.10.2010].

[14] Randal, K Michael. Mastering UNIX Shell Scripting: Bash, Bourne, and Korn Shell Scripting for Programmers, System Administrators, and UNIX Gurus, Second Edition. Indianapolis: Wiley Publishing Inc, 2008.

[15] BackupPC: Network Backup with De-Duplication. [Verkkodokumentti] [Viitattu 14.10.2010]. Saatavissa: http://www.zmanda.com/backuppc.html .

[16] Friedman, Rich. Open source backup tools Amanda, BackupPC and Bacula compared. [Verkkodokumentti] 21.7.2009 [Viitattu 10.10.2010]. Saatavissa: http://searchstorage.techtarget.com.au/news/2240020810/Open-source-backup-tools-Amanda-BackupPC-and-Bacula-compared .

[17] Dirvish FAQ. [Verkkodokumentti] [Viitattu 10.10.2010]. Saatavissa: http://www.dirvish.org/FAQ.html .

[18] Indiana University Knowledge Base: Unix hard link. [Verkkodokumentti] 13.5.2009 [Viitattu 11.10.2010]. Saatavissa: http://kb.iu.edu/data/aibc.html .

[19] Morris, James. New SELinux features for 2.6.12: name_connect, working reiserfs xattrs, checkreqprot etc. [Verkkodokumentti] 12.4.2005 [Viitattu 19.10.2010]. Saatavissa: http://james-morris.livejournal.com/3580.html .

[20] Lofstrom, Keith. Dirvish and Rsync for Disc-to-Disc Backups. [Verkkodokumentti] 1.9.2007 [Viitattu 6.10.2010]. Saatavissa: http://wiki.dirvish.org/plugin/attachments/Presentations/2007Sep.pdf .

[21] BackupPC Documentation. [Verkkodokumentti] [Viitattu 14.10.2010]. Saatavissa: http://backuppc.sourceforge.net/faq/BackupPC.html .

[22] In memoriam Jonathan Schultz. [Verkkodokumentti] [Viitattu 5.10.2010]. Saatavissa: http://wiki.dirvish.com/index.cgi?JonathanSchultz .

[23] WhatFilesystem. [Verkkodokumentti] [Viitattu 5.10.2010]. Saatavissa: http://wiki.dirvish.com/index.cgi?WhatFilesystem .

[24] htpasswd:n virallinen man-sivu, versio 2007-04-24.

[25] Speed up backups. [Verkkodokumentti] 7.8.2009 [Viitattu 15.10.2010]. Saatavissa: http://sourceforge.net/apps/mediawiki/backuppc/index.php?title=Speedupbackups .

[26] Bramscher, Paul. Creating Certificate Authorities and self-signed SSL certificates. [Verkkodokumentti] 29.5.2008 [Viitattu 16.10.2010]. Saatavissa: http://www.tc.umn.edu/~brams006/selfsign.html .

[27] O'Reilly & Associates. SSH: The Secure Shell The Definitive Guide. [Verkkodokumentti] 2002 [Viitattu 16.10.2010] Saatavissa: http://docstore.mik.ua/orelly/networking_2ndEd/ssh/ch03_09.htm .

[28] Performing Backups and Recovery with Berkeley DB. [Verkkodokumentti] [Viitattu 12.10.2010] Saatavissa: http://docs.hp.com/en/5991-7503/ar01s07.html .

```
#!/bin/bash
# backup.sh Copyright (C) 2010 EfiCode Oy
# daniel.suni@eficode.fi
# Version 1.0.0 (01.11.2010)

# Exit codes:
# 0 -> Successful execution
# 1 -> Error in configuration and/or usage
# 2 -> Database error
# 3 -> LVM snapshot error
# 4 -> Subversion error

################################################################################
# BEGIN CONFIGURATION
#
# Directories defined here should not end with a slash.
#
# Regarding dump directories (*_DUMPDIR):
#
# -They must point to volumes with enough free space to hold the dumps
# -They must be in a path that is actually backed up by BackupPC
# -They don't have to exist, they will be created as needed
# -They shouldn't contain any other data, they will be deleted as needed
# -You shouldn't use the same directory for different dumps (It will
#  mess up the deletion schema.)
# -You shouldn't restore one of these directories unless you want to
#  roll the system in question back to that version
################################################################################

# Set this to 1 if client uses LVM snapshot backup strategy, else to 0.
# If set to 0, all other snapshot related items become irrelevant.
SNAPSHOT=0

#This can be any name that is not used for another volume
SNAPSHOT_NAME="snap"

# Set the size of the snapshot volume. M = megabytes, G = gigabytes et.c.
# Please make sure that the volume group defined by $VOLUME_GROUP_NAME
# has at least this much unallocated space.
SNAPSHOT_SIZE="10G"

# The directory where the volume groups reside.
VOLUME_GROUP_DIRECTORY="/dev"

# The name of the volume group to snapshot.
VOLUME_GROUP_NAME="storage"

#The name of the logical volume to snapshot.
LOGICAL_VOLUME_NAME="data"

# The mountpoint for the snapshot
SNAPSHOT_MOUNTPOINT="/mnt/snapshot"

# Set to 1 if you have a MySQL database you want to back up, else 0.
MYSQL=0

# MySQL root password. Make sure that this script is owned by root
# and has permissions set to 700. If root password is not used, set
# it to "".
MYSQL_PASSWORD="mysecretpassword"

# MySQL init script
MYSQL_INIT="/etc/init.d/mysql"

# Where to create MySQL dumpfile in case LVM snapshot isn't used.
MYSQL_DUMPDIR="/tmp/mydump"

# Set to 1 if you have a PostgreSQL database you want to back up, else 0.
POSTGRE=0
```

```
# User to use when running dumping or restoring PostgreSQL.
POSTGRE_USER="postgres"

# PostgreSQL init script
POSTGRE_INIT="/etc/init.d/postgresql-8.4"

# Where to create PostgreSQL dumpfile in case LVM snapshot isn't used.
POSTGRE_DUMPDIR="/tmp/postdump"

# Set to 1 if you have an LDAP database to back up, else to 0.
LDAP=0

# LDAP init script
LDAP_INIT="/etc/init.d/slapd"

# LDAP config file
LDAP_CONFIG="/etc/openldap/slapd.conf"

# Where to create LDAP dumpfile in case LVM snapshot isn't used.
# MAKE SURE THIS DIRECTORY IS AMONG THOSE BEING BACKED UP!!!
LDAP_DUMPDIR="/tmp/ldapdump"

# Set to 1 if you have subversion repositories to back up, else 0.
SVN=0

# The paths to the svn repositories. One repository on each line.
SVN_REPOS="/var/svn/foo
/var/svn/bar
/var/svn/baz"

# Where to make the SVN dumpfiles go. Since SVN repositories can't
# be reliably locked, dumpfiles will always be used.
SVN_DUMPDIR="/tmp/svndump"

############################################################
# END CONFIGURATION
############################################################

# Make sure only root can run this script
if [ "$(id -u)" != "0" ] ; then
  echo "Backup preparation script must be run as root." 1>&2
  exit 1
fi

# Make sure that only values 0 & 1 are used for the options that require it
if [ $SNAPSHOT -lt 0 -o $SNAPSHOT -gt 1 -o $MYSQL -lt 0 -o $MYSQL -gt 1 -o
$POSTGRE -lt 0 -o $POSTGRE -gt 1 -o $LDAP -lt 0 -o $LDAP -gt 1 -o $SVN -lt
0 -o $SVN -gt 1 ] ; then
  echo "Only vaues 0 and 1 are accepted for activation variables." 1>&2
  exit 1
fi

# Make sure that options are sane
if [ $# -ne 2 ] ; then
  echo "Backup preparation script must be run either with 2 parameters --[b
ackup/restore] --[start/stop]" 1>&2
  exit 1
fi
start=0
backup=0
if [ "$1" == "--start" -o "$2" == "--start" ] ; then
  start=1
elif [ "$1" == "--stop" -o "$2" == "--stop" ] ; then
  start=0
else
  echo "Backup preparation script must be run either with --start or --stop
 parameter." 1>&2
  exit 1
fi
```

```
if [ "$1" == "--backup" -o "$2" == "--backup" ] ; then
  backup=1
elif [ "$1" == "--restore" -o "$2" == "--restore" ] ; then
  backup=0
else
  echo "Backup preparation script must be run either with --backup or --res
tore parameter." 1>&2
  exit 1
fi

# Make sure that MySQL is installed if backup requested
if [ $MYSQL -eq 1 ] ; then
  if ! which mysql mysqldump &>/dev/null ; then
    echo "Could not find database MySQL, or some of its files." 1>&2
    exit 1
  fi
  if [ ! -f $MYSQL_INIT ] ; then
    echo "Could not find the MySQL init script." 1>&2
    exit 1
  fi
fi

# Make sure PostgreSQL is installed if backup requested
if [ $POSTGRE -eq 1 ] ; then
  if ! which psql pg_dumpall &>/dev/null ; then
    echo "Could not find database PostgreSQL, or some of its files." 1>&2
    exit 1
  fi
  if [ ! -f $POSTGRE_INIT ] ; then
    echo "Could not find the PostgreSQL init script." 1>&2
    exit 1
  fi
fi

# Make sure that LDAP is installed if backup requested
ldap_datadir=""
if [ $LDAP -eq 1 ] ; then
  if ! which slapcat slapadd slapindex &>/dev/null ; then
    echo "Could not find LDAP or its tools." 1>&2
    exit 1
  fi
  if [ ! -f $LDAP_INIT -o ! -f $LDAP_CONFIG ] ; then
    echo "Could not find LDAP environment files." 1>&2
    exit 1
  fi
# The first sed strips out all the comments. Then we grep for "directory"
# and should be left with a line that looks like this:
# directory        "/some/ldap/path/"
# The second sed strips out the quotes, then everything before the first
# slash, then the trailing slash. /some/ldap/path should remain.
  ldap_datadir=‘cat $LDAP_CONFIG | sed ’s/#.*//’ | grep directory | sed ’s/
\"//g;s/.*[ \t]\//\//;s/\"//;s/\/$//’‘
  if [ -z ‘echo $ldap_datadir | sed ’s/[ \t]//g’‘ ] ; then
    echo "Unable to extract LDAP datadir from "$LDAP_CONFIG" Aborting." 1>&
2
    exit 1
  fi
fi

# Make sure that svn is installed, if backup requested
if [ $SVN -eq 1 ] ; then
  if ! which svn svnadmin &>/dev/null ; then
    echo "Could not find the appropriate svn tools." 1>&2
    exit 1
  fi
# Check that the repositories are valid. This is checked only on backup,
# as we might want to restore into a place without a valid repo.
  for repo in $SVN_REPOS ; do
    if [ ! -f $repo/format -a $backup -eq 1 ] ; then
      echo "Invalid svn repository: $repo" 1>&2
```

```
      exit 1
    fi
  done
fi

# Modify the MYSQL_PASSWORD parameter so that it works with MySQL
# regardless of whether it uses a password or not. (I.e. if a password
# exists, add "--password=" in front of it.)
if [ ! -z $MYSQL_PASSWORD ] ; then
  MYSQL_PASSWORD="--password="$MYSQL_PASSWORD
fi
innodb="" # Databases containing *only* InnoDB tables
myisam="" # Databases containing MyISAM tables

function snapshot_fail {
  if [ $MYSQL -eq 1 ] ; then
    echo "UNLOCK TABLES;" | mysql -u root $MYSQL_PASSWORD
  fi
  if [ $POSTGRE -eq 1 ] ; then
    $POSTGRE_INIT start &>/dev/null
  fi
  if [ $LDAP -eq 1 ] ; then
    $LDAP_INIT start &>/dev/null
  fi
  echo "Could not create or mount LVM volume." 1>&2
  exit 3
}

####################
# Backup start part
####################
if [ $backup -eq 1 ] ; then
  if [ $start -eq 1 ] ; then

# Always copy the svn repos if they exist
    if [ $SVN -eq 1 ] ; then
      for repo in $SVN_REPOS ; do
        svndir=$SVN_DUMPDIR/‘echo $repo | sed ’s/.*\///’‘
        mkdir -p $svndir
        svnadmin hotcopy $repo $svndir || exit 4
      done
    fi

# Create LVM snapshot if requested
    if [ $SNAPSHOT -eq 1 ] ; then
      mkdir -p $SNAPSHOT_MOUNTPOINT || exit 1
# Shut down LDAP until snapshot is completed.
      if [ $LDAP -eq 1 ] ; then
        $LDAP_INIT stop &>/dev/null || exit 2
      fi
# Lock MySQL tables until snapshot is completed.
      if [ $MYSQL -eq 1 ] ; then
        echo "FLUSH TABLES WITH READ LOCK;" | mysql -u root $MYSQL_PASSWORD
 || exit 2
        echo "FLUSH LOGS;" | mysql -u root $MYSQL_PASSWORD || ‘echo "UNLOCK
 TABLES;" | mysql -u root $MYSQL_PASSWORD ; exit 2‘
      fi
# Shut down PostegreSQL until snapshot is completed.
      if [ $POSTGRE -eq 1 ] ; then
        $POSTGRE_INIT stop &>/dev/null || exit 2
      fi
# Take snapshot.
      lvcreate -L$SNAPSHOT_SIZE -s -n $SNAPSHOT_NAME $VOLUME_GROUP_DIRECTOR
Y/$VOLUME_GROUP_NAME/$LOGICAL_VOLUME_NAME &>/dev/null || snapshot_fail
      mount --read-only $VOLUME_GROUP_DIRECTORY/$VOLUME_GROUP_NAME/$SNAPSHO
T_NAME $SNAPSHOT_MOUNTPOINT || snapshot_fail
# Bring PostgreSQL back online.
      if [ $POSTGRE -eq 1 ] ; then
        $POSTGRE_INIT start &>/dev/null || exit 2
      fi
```

```
# Unlock MySQL.
      if [ $MYSQL -eq 1 ] ; then
        echo "UNLOCK TABLES;" | mysql -u root $MYSQL_PASSWORD || exit 2
      fi
# Reactivate LDAP
      if [ $LDAP -eq 1 ] ; then
        $LDAP_INIT start &>/dev/null || exit 2
      fi

# What to do in case of non-snapshot backup
    else
      if [ $MYSQL -eq 1 ] ; then
        mkdir -p $MYSQL_DUMPDIR || exit 1

# Check which *databases* are using InnoDB
        for base in `echo 'SELECT table_schema FROM information_schema.tabl
es WHERE engine="InnoDB";' | mysql -u root $MYSQL_PASSWORD | tail -n +2 | s
ort -u` ; do
          innodb=$innodb" "$base
        done

# Check which *databases* are not using InnoDB.
# (They will all be treated like MyISAM.)
        for base in `echo 'SELECT table_schema FROM information_schema.tabl
es WHERE NOT engine="InnoDB" AND NOT table_schema="information_schema" AND
NOT table_schema="mysql";' | mysql -u root $MYSQL_PASSWORD | tail -n +2 | s
ort -u` ; do
          myisam=$myisam" "$base
        done

# Compare the two and check for Frankenstein databases (containing both
# InnoDB & MyISAM tables) These will also be treated like MyISAM.
        for innobase in $innodb ; do
          for isambase in $myisam ; do
            if [ "$innobase" == "$isambase" ] ; then
              innodb=`echo $innodb | sed "s/ $isambase//"`
            fi
          done
        done

# Clean up the lists
        innodb=`echo $innodb | sed 's/^ //'`
        myisam=`echo $myisam | sed 's/^ //'`

# Back up the MyISAM & Frankenstein databases
        for database in $myisam ; do
          mysqldump -u root --lock-all-tables --flush-logs --add-drop-table
 --add-locks --disable-keys $MYSQL_PASSWORD $database > $MYSQL_DUMPDIR/$dat
abase.sql || exit 2
        done
# Back up the InnoDB databases
        for database in $innodb ; do
          mysqldump -u root --single-transaction --flush-logs --add-drop-ta
ble --add-locks $MYSQL_PASSWORD $database > $MYSQL_DUMPDIR/$database.sql ||
 exit 2
        done
# Back up the mysql database
        mysqldump -u root --lock-all-tables --flush-logs --add-drop-table -
-add-locks --disable-keys --flush-privileges $MYSQL_PASSWORD mysql > $MYSQL
_DUMPDIR/mysql.sql || exit 2
      fi

# What to do in case of PostgreSQL and no snapshot
      if [ $POSTGRE -eq 1 ] ; then
        mkdir -p $POSTGRE_DUMPDIR || exit 1
        su $POSTGRE_USER -c pg_dumpall > $POSTGRE_DUMPDIR/postgre.dump || e
xit 2
      fi

# What to do in case of LDAP and no snapshot
```

```
      if [ $LDAP -eq 1 ] ; then
        mkdir -p $LDAP_DUMPDIR
        slapcat -f $LDAP_CONFIG -l $LDAP_DUMPDIR/ldap.ldif || exit 2
      fi
    fi
  fi
fi

###################
# Backup stop part
###################
if [ $backup -eq 1 ] ; then
  if [ $start -eq 0 ] ; then
# Remove LVM snapshot if one exists
    if [ $SNAPSHOT -eq 1 ] ; then
      umount $SNAPSHOT_MOUNTPOINT
      lvremove -f $VOLUME_GROUP_DIRECTORY/$VOLUME_GROUP_NAME/$SNAPSHOT_NAME
 &>/dev/null || exit 3

# Remove dump directories
    else
      if [ $MYSQL -eq 1 ] ; then
        rm -rf $MYSQL_DUMPDIR
      fi
      if [ $POSTGRE -eq 1 ] ; then
        rm -rf $POSTGRE_DUMPDIR
      fi
      if [ $LDAP -eq 1 ] ; then
        rm -rf $LDAP_DUMPDIR
      fi
    fi
    if [ $SVN -eq 1 ] ; then
      rm -rf $SVN_DUMPDIR
    fi
  fi
  exit 0
fi

####################
# Restore start part
####################

# For now, nothing needs to be done here.

###################
# Restore stop part
###################
if [ $start -eq 0 ] ; then
# Svn repos must always be imported if they exist
  if [ $SVN -eq 1 ] ; then
    for repo in $SVN_REPOS ; do
      if [ -d $SVN_DUMPDIR/‘echo $repo | sed ’s/.*\///‘‘ ] ; then
        rm -rf $repo/* || exit 4
        cp -a $SVN_DUMPDIR/‘echo $repo | sed ’s/.*\///‘‘/* $repo || exit 4
# Restore the permissions (that svnadmin hotcopy loses)
        chown --recursive ‘ls -ld $repo | awk ’{print $3}‘‘.‘ls -ld $repo |
 awk ’{print $4}‘‘ $repo
      fi
    done
    if [ -d $SVN_DUMPDIR ] ; then
      rm -rf $SVN_DUMPDIR
    fi
  fi

# If we’re not using snapshots the databases must be imported
  if [ $SNAPSHOT -eq 0 ] ; then
    if [ $MYSQL -eq 1 ] ; then
      $MYSQL_INIT start &>/dev/null
      if [ -d $MYSQL_DUMPDIR ] ; then
        for database in ‘ls $MYSQL_DUMPDIR/*.sql‘ ; do
```

```
          mysql ‘echo $database | sed ’s/\.sql$//;s/.*\///’‘ -u root $MYSQL
_PASSWORD < $database || exit 2
        done
        rm -rf $MYSQL_DUMPDIR
      fi
    fi
    if [ $POSTGRE -eq 1 ] ; then
      if [ -f $POSTGRE_DUMPDIR/postgre.dump ] ; then
        $POSTGRE_INIT start &>/dev/null
        su $POSTGRE_USER -c "psql -f $POSTGRE_DUMPDIR/postgre.dump &>/dev/n
ull" || exit 2
        rm -rf $POSTGRE_DUMPDIR
      fi
    fi
    if [ $LDAP -eq 1 ] ; then

# Restore operation is dangerous. Perform it only if backup exists.
      if [ -f $LDAP_DUMPDIR/ldap.ldif ] ; then
        rm -rf $ldap_datadir/*
        slapadd -c -l $LDAP_DUMPDIR/ldap.ldif -f $LDAP_CONFIG || exit 2
        slapindex
        rm -rf $LDAP_DUMPDIR
      fi
      $LDAP_INIT start &>/dev/null
    fi
  fi
fi

exit 0
```