Verkkosovellusten toteuttamisympäristö

Jyrki Hällström

28.5.2004

Joensuun yliopisto
Tietojenkäsittelytiede
Kandidaatintutkielma

Tiivistelmä

Verkkosovellusten toteuttamiseksi on useita käyttöympäristöjä. Tässä työssä käsitellään ilmaisohjelmistojen ja komponenttien hyödyntämistä WWW-palvelinympäristön toteuttamisessa. Tutkielman liitteenä on yksinkertainen tietokantasovellus, joka on toteutettu Apache-, PHP- ja MySQL-ohjelmistoilla.

ACM-luokat (ACM Computing Classification System, 1998 version): C.2.4, H.2.4, H.2.7

Avainsanat: Verkkosovellus, palvelinympäristö, Apache, PHP, MySQL

Sisältö

1 JOHDANTO 1

2 PALVELIMEN TOTEUTUS 2

2.1 LINUX 3
2.2 APACHE 5
2.3 APACHEN KONFIGUROINTI 6
2.4 PHP 8
2.5 ISTUNNONHALLINTA 9
2.6 MYSQL 11
2.7 MYSQL:N TURVALLISUUS 12

3 YMPÄRISTÖN KÄYTTÖÖNOTTO 13

3.1 NIMIPALVELIN 13
3.2 WEBMIN 14
3.3 APACHEN HAKEMISTORAKENNE 15
3.4 TIEDOSTOJEN JA HAKEMISTOJEN KÄYTTÖOIKEUDET 15

4 YLLÄPITO 16

5 ESIMERKKISOVELLUS 17

6 YHTEENVETO 19

VIITTEET 20

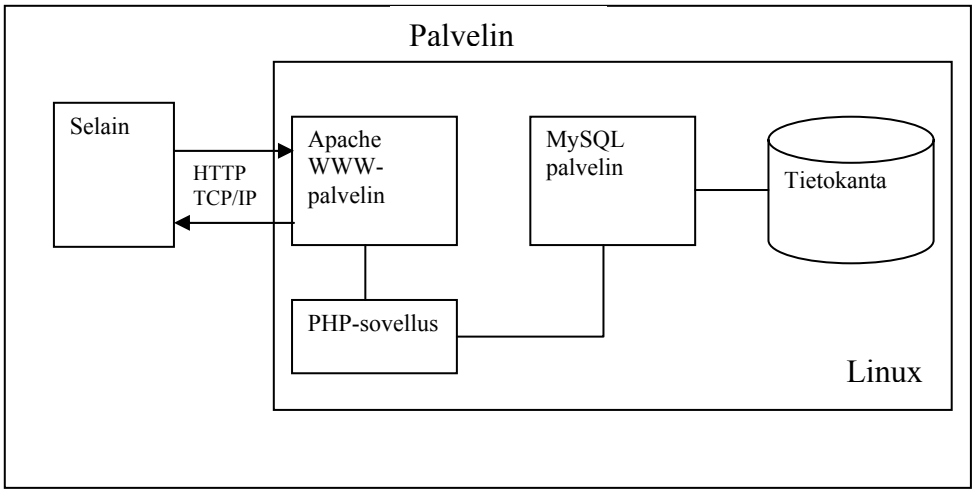
LIITE 1: ESIMERKKISOVELLUKSEN LÄHDEKOODI KÄYTTÄJÄT-OSIOLLE 22

LIITE 2: ESIMERKKISOVELLUKSEN LÄHDEKOODI HALLINTA-OSIOLLE 25

1 Johdanto

Tietoyhteiskunnassa valtava määrä tietoa liikkuu sähköisessä muodossa. Tietoa voidaan hallita ja hakea *verkkosovellusten* avulla. Verkkosovellus voi toimia yrityksen sisäisessä tai julkisessa verkossa. Julkisella verkolla tarkoitetaan tässä tutkielmassa Internet-verkkoa. Verkkosovellus tarvitsee toimiakseen palvelimen, jonka toteuttaminen voi tulla monessa tapauksessa taloudellisesti kannattavaksi mikäli palvelujen ulkoistamiskustannukset ovat korkeat tai palveluilta halutaan tiettyjä räätälöityä ominaisuuksia. Palvelin voi toimia esimerkiksi posti-, tulostus- tai WWW-palvelimena. Kaikkia palveluita, joita käytetään selaimen kautta voidaan kutsua verkkosovelluksiksi. Lahtosen (2002) mukaan suuri osa maailmalla olevasta informaatiosta on tallennettuna tietokantoihin. Tietokantojen hyöty on merkittävä, kun verkkosovelluksesta halutaan tehdä tehokas ja monikäyttöinen.

Verkkosovellusten vaatiman palvelimen toteutukseen on olemassa useita erilaisia välineitä ja ohjelmistoja. Tässä tutkielmassa tarkastellaan sellaisia ohjelmia, jotka ovat kaikkien vapaasti saatavilla olevia ilmaisohjelmia. Teknisesti toimiva palvelin voidaan toteuttaa jonkin sopivan WWW-ohjelmointikielen ja tietokantaratkaisun yhdistelmällä kohtuullisen helposti. Ilmaisohjelmistoista mahdollisia vaihtoehtoja tarjoavat muun muassa Apache, PHP sekä MySQL ja kaupallisista vaihtoehdoista esimerkkeinä mainittakoon ORACLE ja PL/SQL (Silvén, 1999). Kaupallisena WWW-palvelinratkaisuna voidaan mainita Microsoftin Windows Server System, joka on Netcraftin (2004) mukaan Apachen jälkeen toiseksi suosituin palvelinjärjestelmä ja SunONE on kolmanneksi suosituin ympäristö.



Kuva 1: Verkkosovelluksen eräs malli.

Apachen alustaksi voidaan käyttää joko Linux- tai Windows-perusteista käyttöjärjestelmää. Mikäli palvelin toimii yrityksen sisäisessä verkossa ilman ulkoista liitäntää, voidaan tietoturvaratkaisut toteuttaa suljetun järjestelmän ehdoilla. Tällöin ei tarvitse ottaa huomioon ulkoisten liitäntöjen aiheuttamaa tietoturvariskiä. Sisäisistä verkoista voidaan mainita esimerkiksi pankkiautomaatit. Julkisessa verkossa WWW-palvelimen tulee pystyä vastaamaan ulkoisiin HTTP-pyyntöihin. Kaikki Internet-yhteydessä olevat palvelimet tuovat mukanaan turvallisuusongelmia, joiden huomioonottaminen on tärkeää tiedon ja järjestelmän turvaamiseksi. Tietoturvan takaamiseksi eivät kuitenkaan riitä pelkät tekniset turvatoimet, vaan järjestelmän käyttäjien toimiminen hyväksyttyjen tietoturvaperiaatteiden mukaan ja ohjeistuksen noudattaminen ratkaisevat suojaustoimien onnistumisen.

WWW-palvelimen toiminnan hyödyntäminen vaatii usein dynaamisia palveluita. Siten sivuista saadaan mielekkäitä ja järkevästi toimivia interaktiivisia kokonaisuuksia. Mikäli suunnittelutyö tehdään järkevästi voidaan PHP, MySQL ja Apache yhdistelmällä toteuttaa laajojakin kokonaisuuksia, joiden konfiguroiminen eli järjestelmän asetusten valitseminen on helppoa.

Tutkielman rakenne on seuraava. Aluksi esitellään palvelimen toteuttamisympäristöjä siten, että luku 2 esittelee kaupallisen ja vapaan lähdekoodin erilaiset lähestymistavat. Käytetyt ilmaisohjelmat sekä komponentit tarkentuvat kohdissa 2.1 – 2.7. Luku 3 käsittelee palvelinympäristön kayttöönottoa. Luvussa 4 selvitetään palvelimen ylläpitoprosessia. Luku 5 sisältää esimerkkisovelluksen yksinkertaisesta autotietokannasta, joka on toteutettu Apache, PHP ja MySQL ohjelmilla. Tutkielman päättää lyhyt yhteenveto luvussa 6.

2 Palvelimen toteutus

Palvelin voidaan toteuttaa joko kaupallisilla tai vapaan lähdekoodin eli GPL-ohjelmistoilla. Molemmilla on vahvuutensa ja heikkoutensa. Kaupallisten sovellusten hankintaa tukee kattavampi tukiorganisaatio sekä tuotteiden yhtenäisyys esimerkiksi Microsoft-ympäristössä, mutta kertainvestointi on selkeästi suurempi kuin GPL-ohjelmistoissa. Usein verkkopalvelut ja palvelimen kapasiteetti mitoitetaan liian suureksi ja kaupallisten sovellusten tuoma lisäarvo jää helposti käyttämättä. Sama palvelinjärjestelmä olisi mahdollista toteuttaa siis myös

GPL-ohjelmistojen avulla. GPL-toteutus on hyvä vaihtoehto, varsinkin jos yrityksellä on käytettävissä mikrotukea oman henkilökunnan joukossa.

Ammattitaitoinen henkilö pystyttää toimivan palvelimen muutamassa tunnissa, kun hän on tehnyt etukäteissuunnitelman palvelimen toteutuksesta ja hankkinut tarvittavat komponentit sekä ohjelmat. Kaikki tarvittavat ohjelmistotuotteet löytyvät ilmaisohjelmistoina, jolloin kustannussäästö on merkittävä. Seuraavissa kohdissa tarkastellaan Linux-pohjaista alustaa käyttävää palvelimen toteutusympäristöä.

2.1 Linux

Linux on tehokas 32-bittinen moniajokäyttöjärjestelmä. Linux-käyttöjärjestelmän perustan loi suomalainen Helsingin yliopistossa opiskellut Linus Torvald. Järjestelmän on Unix-pohjainen ja sen ensimmäinen versio oli nimeltään minix. Ensimmäinen virallinen Linux-versio julkaistiin vuoden 1991 lopulla. Torvald järjesti lähdekoodin vapaasti saataville ja antoi muille mahdollisuuden kehittää Linuxia eteenpäin (Pitts ja Ball, 1999). Linuxissa on kaikki samat ominaisuudet mitä voisi odottaa Unix-järjestelmästä: moniajo, virtuaalimuisti, jaetut tiedostokirjastot, muistinhallinta ja TCP/IP verkkotoiminta. Linuxin johdolla tapahtunut avoimen lähdekoodin suosion kasvu on saanut monet organisaatiot siirtymään GPL-ohjelmistoihin.

Linux on erittäin vakaa, turvallinen ja täydellisesti verkkoympäristöön sopiva käyttöjärjestelmä. Koska kaikki lähdekoodi on saatavilla, on myös jokaiseen turvallisuusaukkoon saatavissa korjaus yleensä 24 tunnin sisällä siitä, kun turvallisuusaukko on huomattu (Linux Online, 2004). Tietoturvan kannalta Linux on usein parempi kuin kaupalliset suljetut järjestelmät. Vapailla ohjelmilla on yleensä riittävät tietoturvaratkaisut, kun ne on otettu käyttöön. Esimerkiksi palomuurityökalut- ja toteutukset ovat turvallisia, mikäli ne ovat oikein konfiguroitu. Peltomäen ja Linjaman (1999) mukaan koko Linux-tietoturva rakentuu kahden peruskomponentin mukaan: 1) tiedostojen suojaukset eli käyttöoikeudet, jotka määrittelevät eri resurssit vain niiden tahojen käyttöön, joille ne on tarkoitettu ja 2) käyttäjätunnukset ja niihin liittyvät salasanat varmentavat käyttäjän oikeudet määrättyihin resursseihin. Linux-järjestelmän hakemistorakenne on kuvattu taulukossa 1. Tiedostot ja hakemistot ymmärre-

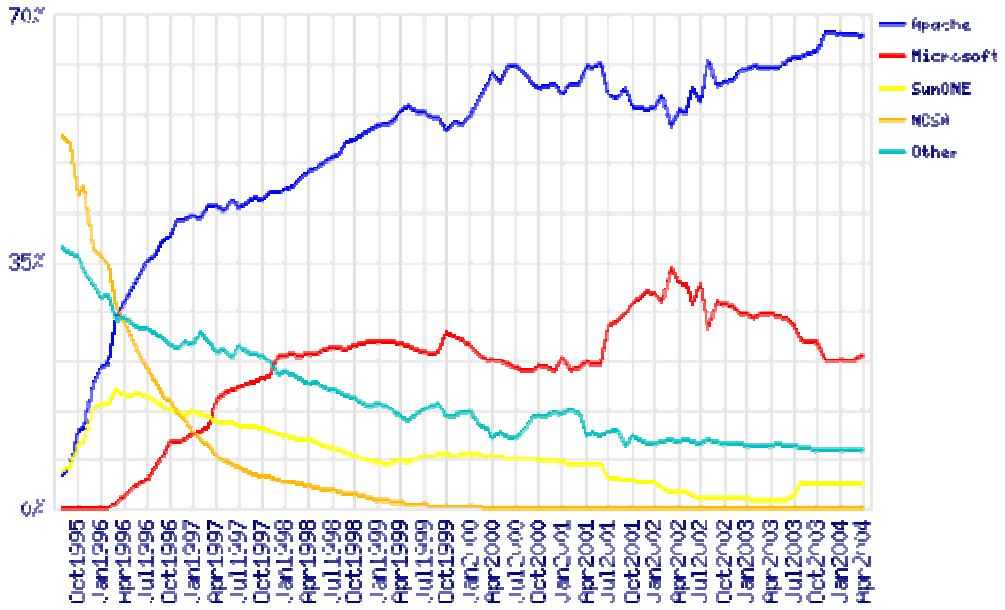
tään Linux-tiedostojärjestelmässä samaksi asiaksi. Linuxin tiedostojärjestelmä on hierarkkinen puurakenne, johon kaikki tiedostot tallennetaan.

Taulukko 1: Linuxin hakemistorakenne.

Hakemisto	Kuvaus
/boot	Kansio joka sisältää bootloaderin (esimerkiksi Lilo).
/bin	Kansio, johon on tallennettu tärkeät binääritiedostot (kaikille käyttäjille).
/sbin	Kansio, johon on tallennettu tärkeät binääritiedostot (root).
/lib	Kansio, joka sisältää välttämättömät jaetut kirjastot sekä kernelin moduulit.
/etc	Kansio, johon tallennetaan konfiguraatiotiedostot.
/tmp	Kansio, joka sisältää väliaikaiset tiedostot.
/var	Kansio, jossa sijaitsee muuttuva data, esimerkiksi sähköposti, uutisryhmät ja tulostusjonot.
/dev	Kansio, johon on tallennettu laitetiedostot (device files), jotka vastaavat koneen komponentteja.
/mnt	Kansio, johon on tarkoitettu tiedostojärjestelmien väliaikaiseen mounttaamiseen(liittämiseen).
/home	Kansio, jossa sijaitsevat käyttäjien kotihakemistot.
/root	Järjestelmänvalvojan (root) kotihakemisto.
/usr	Toissijainen hakemistopuu.
/opt	Kansio, johon on tallennettu ohjelmistoja.
/proc	Kansio, joka on tarkoitettu illusionääri tiedostoja varten.

2.2 Apache

Yksi avoimen ohjelmistokehityksen lippulaivoista on Apache ja se on laajimmin levinnyt WWW-palvelinalusta maailmassa (Laurie ja Laurie, 1999). Apache on WWW-palvelin, joka on vastuussa ylivoimaisesti suurimmasta osasta Internetin WWW-palveluista. Apachella pyöritetään eräitä maailman suurimpia sivustoja. Se on hyvin mukautuva ja toimii myös vanhemmilla tietokoneilla. Esimerkiksi verkkokauppa Amazon (http://www.amazon.com) toimii Apache-pohjaisella järjestelmällä. Netcraftin (2004) mukaan Apache on vakaa ja luotettava tuote kuvan 2 mukaisesti. Sen osuus maailman WWW-palvelimista on lähes 70 %. Se on stabiili, turvallinen ja tehokas palvelinohjelmisto, mikä tarjoaa runsaasti erilaisia skriptaus- ja konfigurointimahdollisuuksia. Apache ohjelmaa voidaan ajaa useilla eri käyttöjärjestelmillä. Apache on suosittu, koska se on ilmainen ja helposti konfiguroitavissa oleva WWW-palvelin ja se on myös yksi Linux-järjestelmien tyypillisimmin tarjoamista palveluista (FCS partners, 2004). Apachen vahvuuksia ovat joustavuus ja sopeutuvuus liittyä erilaisiin käyttöjärjestelmiin ja alustoihin.



Ympäristö	Maaliskuu 2004	Prosenttia	Huhtikuu 2004	Prosenttia	Muutos
Apache	32280582	67.20	33329879	66.99	-0.21
Microsoft	10099760	21.02	10691683	21.49	0.47
SunONE	1651575	3.44	1661229	3.34	-0.10

Kuva 2: Apachen osuus palvelinalustana ajalla elokuu 1995 - huhtikuu 2004.

Apachea on mahdollisuus käyttää monikielisenä versiona. Tämä antaa käyttäjille valinnan mahdollisuuden eri kieliversioiden välillä. Yleisesti standardikielenä on englanti, mutta esimerkiksi virheilmoitukset voidaan tulostaa käyttäjän äidinkielellä.

2.3 Apachen konfigurointi

Palvelinohjelmiston konfiguroiminen on tärkein vaihe Apache-palvelimen saattamisessa toimintakuntoon. Apachen asetukset määritellään tekstitiedostoihin, jotka sijaitsevat oletusarvoisesti palvelimen `/usr/local/apache/conf`-alihakemistossa. Taulukosta 2 ilmenevät tärkeimmät konfigurointitiedostot. Apachen kaikkein tärkein tiedosto on `httpd.conf`. Se on tiedosto, joka sisältää kaikki olennaisimmat palvelimen asetukset. Tiedostoa on helppo muokata tekstieditorilla. Apachen tiedostot ovat yleisesti erittäin hyvin kommentoituja, joten asetusten muokkaaminen on helppoa. Tiedosto `srm.conf` sisältää ja määrittelee palvelimella olevia resursseja, kuten HTML- ja XML-dokumentteja, CGI-ohjelmia sekä muita käytettäviä resursseja. Palvelimen hakemistojen ja niissä olevien tiedostojen oikeuksia voidaan muokata editoimalla `access.conf`-tiedostoa. Apache-palvelimella sijaitseva hakemisto on helppo suojata luomalla suojattavaan hakemistoon `.htaccess` tiedosto, joka määrittelee mistä tiedostosta käyttäjätunnus- ja salasana haetaan (kuva 3). `htpasswd`-ohjelmalla voidaan lisätä käyttäjiä ja muuttaa salasanoja hakemiston käyttöoikeuksiin (kuva 4).

```
AuthUserFile
/home/tko/jhalls/th.txt
AuthType Basic
AuthName "Autotietokanta ylläpito"
require valid-user
```

Kuva 3: Esimerkki tiedostosta `.htaccess`

Ohjelmalla `htpasswd` kirjoitetaan käyttäjätunnus- ja salasana. Optio `-c` luo uuden salasanatiedoston ellei sitä ole jo olemassa. Mikäli tiedosto on luotu aiemmin, korvataan sen sisältö uusilla tunnuksilla.

```
htpasswd -c th.txt matti
Adding password for matti
New password: mainio
Re-type new password: mainio
```

Kuva 4: Käyttäjätunnuksen ja salasanan antaminen `htpasswd` ohjelmalla.

Apache kryptaa automaattisesti salasanan, joten salasanatiedoston `th.txt` sisältö salasanan `mainio` kryptauksen jälkeen on `matti:ZKMVo9NNxaFU2.` Salasana on kryptattu DES-algoritmilla (Data Encrypting Standard). Tämän avulla salasanasta saadaan 4096 (2^12) erilaista kryptattua versiota. Kryptaus toteutetaan sen vuoksi, että salasanojen listoista ei voi päätellä mitkä salasanoista ovat samoja (Rantala 2003). Tiedostoa `etc/shadow` saa lukea ainoastaan `root`-käyttäjä, koska siellä sijaitsevat järjestelmän salasanat.

Taulukko 2: Apachen konfigurointitiedostot.

Tiedosto	Selitys
`httpd.conf`	Tärkein konfigurointitiedosto.
`srm.conf`	Resurssien määrittelytiedosto.
`access.conf`	Käyttöoikeuksien määrittelytiedosto.

Apache-palvelinta voidaan ajaa kahdessa tilassa, joko `standalone`- tai `inetd`-tilassa. Kun palvelinta ajetaan `inetd`-tilassa, niin Linux huolehtii palvelinohjelmiston käynnistämisestä aina, kun hakupyyntö saapuu tiettyyn TCP-porttiosoitteeseen. Prosessit muodostavat hierarkkisen järjestelmän siten, että `initd`-prosessi on kaikkien prosessien äiti. Suorituskyvyn optimoimiseksi Apachea kannattaa ajaa `standalone`-tilassa, koska tällöin Apachen palvelinprosessi on jatkuvasti päällä odottamassa uusia hakupyyntöjä (Rantala, 2002).

Apachen tietoturvaa voidaan parantaa asentamalla palvelimelle ainoastaan tarvittavat ohjelmistot ja komponentit. Kaikki käyttämättä olevat portit kannattaa sulkea mahdollisten murtoyritysten varalta. Linuxin ja Apachen tallentamia lokitiedostoja kannattaa säännöllisesti tarkkailla sekä automaattisesti että CGI-skriptein manuaalisesti (Peltomäki, 1999).

2.4 PHP

PHP on HTML-koodin yhteyteen sijoitettava skriptauskieli, jolla saadaan WWW-sivuille dynamiikkaa varsin helposti. Syntaksiltaan PHP muistuttaa hyvin paljon C-, Java- ja Perl-kieliä, joten se on tehokasta ja helposti opittavissa. Kielen tavoitteena on tarjota nopeasti hyödynnettävä kehitysympäristö (Akkanen *& al.*, 2002). PHP-kokonaisuuden kehittäminen alkoi ideasta saada tietää kuka lukee WWW-palvelimella olevia dokumentteja. Kun muut käyttäjät ottivat käyttöön kehitetyn ohjelmapaketin pyytäen siihen lisää ominaisuuksia, syntyi alkuperäinen PHP eli Personal Home Pages Tools (Kujala, 2000). Alkuperäisen PHP:n kehitystyön aloitti Rasmus Lerdorf vuonna 1994. PHP:n suosio lisääntyi, kun PHP3 julkaistiin kesäkuussa 1998. Viimeisin PHP:n stabiili versio on 4 (Rantala, 2002).

Nykyisin PHP:tä käytetään laajasti erilaisissa WWW-toteutuksissa. PHP on avoimen lähdekoodin ohjelmisto, joten sen käyttäminen on ilmaista. PHP on toimiva kieli WWW-pohjaisten sovellusten tekemiseksi, koska komentojonoja voidaan upottaa suoraan HTML-koodiin. Jos ohjelmoijalla on kokemusta esimerkiksi C- tai Java-kielestä niin PHP:n omaksuminen on kohtuullisen helppoa kielten samankaltaisen syntaksin vuoksi. PHP:llä voidaan rakentaa dynaamisia WWW-sivustoja, lähettää kuitteja (cookie) tai kytkeytyä erilaisilla Internet-protokollilla erilaisiin palvelimiin. PHP tukee valmiiksi esimerkiksi SMTP- (*Simple Mail Transfer Protocol*), POP3- (*Post Office Protocol 3*), FTP- (*File Transfer Protocol*) ja tietenkin HTTP-protokollaa (*HyperText Transfer Protocol*), joten se soveltuu hyvin esimerkiksi WWW-pohjaisten palveluiden ohjelmointiin (Hakala *& al.,* 1999). Palvelimella suoritettavien sovelluksien merkittävä etu on niiden toiminnan varmistaminen erilaisilla järjestelmäalustoilla. Esimerkiksi eri valmistajien selaimet, PDA-laitteet ja matkapuhelimet hyötyvät palvelimella suoritettavista ohjelmista. PHP:llä vältetään yhteensopivuusongelmia, joita voi ilmetä esimerkiksi JavaScript-sovelluksissa.

Mannermaan (1999) mukaan PHP:n erinomainen ominaisuus on sen hyvä tuki eri tietokannoille. Tällä hetkellä löytyy ajurit niin Oracle- ja dBase-tietokannoille kuin maksuttomille MySQL- ja PostgreSQL-kannoille. Mikäli kantaa ei löydy suoraan tuettavien tietokantojen listasta voi käyttää geneerisiä ODBC-ajureita, jotka löytyvät melkein jokaiseen relaatiotietokantaan.

2.5 Istunnonhallinta

PHP:n turvallisuutta ja hallittavuutta voidaan parantaa istunnonhallinnalla. Rantalan (2002) mukaan istunnolla tarkoitetaan sellaisia yhteenkuuluvia HTTP-pyyntöjä, joita voidaan tarkastella selkeän aloitus- ja lopetustapahtuman avulla. WWW-sovelluksessa istunnon luominen on hankalaa, koska HTTP on tilaton protokolla, jonka avulla ei ole mahdollista ylläpitää tilatietoa eri siirtotapahtumien välillä. Istunnonhallinnan avulla pystytään ylläpitämään tilatietoa sovelluksen tilasta asiakaspäässä. PHP mahdollistaa sisäänrakennetut komponentit istunnonhallintaan versiosta 4 alkaen. PHP tukee istuntojen hallintaa pysyvien tietojen tallennuksessa (PHP Manual, 2004). Istunto on tietty joukko muuttujia, jotka säilytetään palvelupyyntöjen välillä. Istunto tulee pystyä tunnistamaan tietyn tunnistekoodin avulla. Tunnistekoodi välitetään HTTP-kuittien avulla (Leponiemi, 2004).

Istunto luodaan PHP:ssä funktiolla `session_start`. Istunnolle asetetaan muuttujia käsittelemällä `$_session`-taulukkoa. Kuvassa 5 on esimerkki istunnon luomisesta.

```
<?php
  session_start();
  $_SESSION["Muuttuja"] = "Arvo";
?>
```

Kuva 5: Istunnon luominen.

Istuntoon lisätään muuttujia sijoittamalla niitä taulukkoon `$_SESSION`. Taulukko on käytössä ainoastaan funktion `session_start` kutsun jälkeen. Muuttuja poistetaan funktion `unset` avulla, esimerkiksi `unset($_SESSION["Muuttuja"])` (Leponiemi, 2004). Istunnon voi tarvittaessa tuhota komennolla `session_destroy()`

Leponiemen (2004) mukaan kuitit tarjoavat mahdollisuuden määritellä verkkopalvelua asiakkaan valintojen mukaan. Yksinkertainen esimerkki kuittien käytöstä on ylläpitää tietoa asiakkaan vierailusta määrätyllä WWW-sivulla. Asetetaan kuitti kun asiakas ensimmäisen kerran lataa WWW-sivun. Kuitti on lyhyt tekstimuotoinen tieto, joka tallennetaan käyttäjän työasemalla selaimen avulla. Kuitti muodostuu nimi–arvo parista ja sille voidaan määrittää

vanhenemisaika, eli aika jolloin selain poistaa kuitin. Kuitin avulla voidaan mukauttaa WWW-palvelua käyttäjän valintojen mukaan. Tämä on yleistä esimerkiksi pankkien verkkopalveluissa, jotka tunnistavat käyttäjän ja hänen aiemman profiilinsa kuitin avulla. Kuitteja on kahdenlaisia: istuntokohtaisia ja pysyviä. Istuntokohtaiset kuitit säilyvät selaimen muistissa niin kauan kuin selain on auki. Pysyvät kuitit tallentuvat levylle ja säilyvät siellä kunnes käyttäjä menee samaan palveluun uudestaan ja selain palauttaa kuitin palvelimelle. Kuvassa 6 on esimerkki kuitin luomisesta. Esimerkin rakenne on seuraava: Muuttuja `$k` alustetaan arvoon 1. Kuitti nimeltä ”`kaynnit`” asetetaan siten, että sen vanhenemisaika on 1 tunti. Kun käyttäjä lataa WWW-sivun uudelleen, niin muuttuja `$k`:n arvo kasvaa yhdellä. Otsaketiedoissa kerrotaan selaimelle, että tietoja ei saa tallentaa välimuistiin.

```
<?php // ASETETAAN KUITTI
  $k = 1;
  if ($_COOKIE[“kaynnit”]) {
      $k = $_COOKIE[“kaynnit”] + 1;
  }
  setcookie(“kaynnit”, $k, time() + 3600);
  header(“Pragma: No-cache”); ?>
```

Kuva 6: Kuitin luominen.

Myöhemmin voidaan tulostaa käyttäjälle hänen vierailujensa määrä viimeisen tunnin aikana esimerkiksi seuraavasti:

```
Olet ladannut tämän sivun viimeisen tunnin aikana
<b><?= $k ?></b> kertaa!
```

Kuittien toimintaan ei tule täysin luottaa, koska käyttäjällä on mahdollisuus kieltää niiden käyttö. Kun halutaan varmistaa WWW-sovellusten toiminta, niin kuittien tulisi oletusarvoisesti olla sallittuja.

2.6 MySQL

Hakalan *& al.* (1999) mukaan tietokantojen käyttö perustuu yleensä relaatiotietokantajärjestelmään ja standardeihin kyselykieliin sekä rajapintoihin. Sovellusohjelmilla ei ole yleensä suoraa pääsyä tietokantoihin, mikä mahdollistaa tiedon suojaamisen ja korruptoimisen väärinkäytöksiltä. Toimiva SQL-tietokanta tarvitsee rajapinnan tietokannan ja ohjelmointikielten välille.

MySQL on ilmainen vapaan lähdekoodin relaatiotietokantasovellus (MySQL, 2004). Ruotsalainen T.c.X Ab yhtiö kehitti sen aluksi omiin yhtiön sisäisiin käyttötarkoituksiinsa. WWW-ohjelmoinnin lisääntyminen on saanut aikaan rajapintojen kehittymisen itse palvelinohjelmiin, kuten Apache-palvelimeen. MySQL:n noudattama sisäinen asiakas–palvelin-arkkitehtuuri tekee siitä vakaan ja helposti hallittavan ohjelmiston, jossa sovellukset eivät koskaan käsittele suoraan tietokantaa, vaan käsittely tapahtuu aina palvelinohjelman kautta. MySQL-palvelimeen voidaan ottaa yhteys erilaisten rajapintojen kautta, kuten ODBC ja JDBC. Java-kielen lisäksi tietokantaa on mahdollista käsitellä myös PHP-, Perl-, Python-kielillä sekä C- kielellä käyttäen Tcl-tekniikkaa. ODBC on rajapinta SQL-ohjelmointiin Windows ympäristössä. Lähes jokaiseen tietokannanhallintajärjestelmään on saatavilla ajuri, joka tulkitsee ODBC API:n (*Application Programming Interface*) kautta suoritetut SQL-komennot. JDBC on standardi Java-ohjelmoinnissa tietokantayhteyksien ja tietokantojen välillä. JDBC API mahdollistaa SQL-kyselyiden suorittamisen (Sun Microsystems, 2004). Haluttaessa myös C++ kieltä voidaan hyödyntää (Heinisuo, 2001). MySQL toimii myös Tomcat-palvelimella servlet-ohjelmointia hyödyntäen. Tomcat on ilmainen WWW-palvelin, joka sisältää muun muassa tuen JSP-sivustoille ja Java-servleteille. Tomcat-ohjelmisto on ladattavissa asennusohjeineen osoitteesta: http://jakarta.apache.org/tomcat.

MySQL:n nykyinen versio on 4.0.12 ja se sisältää SQL-standardin ANSI SQL99 tuen. SQL (Structured Query Language) on standardoitu relaatiotietokantojen yhteydessä käytettävä kieli, joka sisältyy lähes kaikkiin tietokannanhallintajärjestelmiin (Lahtonen, 2002). Lahtosen (2002) mukaan kehittyneisiin tietokannanhallintajärjestelmiin on rakennettu ominaisuuksia, kuten monipuoliset tietoturvatoiminnot, elvytystekniikat ja transaktiot. Transaktio on myös MySQL:n tukema looginen toimintakokonaisuus, joka joko tapahtuu kokonaan tai ei ollenkaan. Esimerkki transaktiosta on toiminta pankkiautomaatilla. Koko toiminta määritellään transaktioksi joka tapahtuu kun kaikki toiminnot on todettu virheettömiksi. Käyttäjällä

on mahdollisuus vahvistaa toimintoketju komennolla `commit` tai peruuttaa toimintoketju komennolla `rollback`. Kuvassa 7 on esimerkki transaktiosta, käyttäjä Matti siirtää rahaa käyttäjälle Jussi.

```
UPDATE tili
SET summa = summa – 1000
WHERE kayttaja = 'Matti';

UPDATE tili
summa = summa + 1000
WHERE kayttaja = 'Jussi';

COMMIT;
```

Kuva 7: Transaktio.

MySQL sisältää ominaisuudet tietoturvan ja käyttöoikeuksien hallintaan. Jokaiselle taululle määritellään omistaja `owner`. Myös valtuuksien edelleen ohjaus onnistuu `grant`-optiolla. Valtuuksia voidaan evätä `revoke` toiminolla. Näkymien käsittelyyn voidaan määritellä oikeudet samalla tavalla kuin tavallisten taulujen yhteyteen (Lahtonen 2002).

2.7 MySQL:n turvallisuus

MySQL tarjoaa käyttäjien oikeuksien (priviledge) määrittelyn sekä pääsynvalvonnan (access control) (MySQL, 2004). Oikeuksien määrittely tarkoittaa SQL:n GRANT-määrittelyjen suorittamista. Pääsynvalvonta tarkoittaa käyttäjätunnuksia ja näihin liittyviä salasanoja. Versiosta 4.1 alkaen PASSWORD()-funktio on tarkoitettu MySQL:n käyttäjätilien hallintaan, mutta sovellusohjelmoija voi omiin tarkoituksiinsa käyttää SHA1()- ja MD5()-funktioita. MD5 (Message Digest 5) ja SHA (Secure Hash Algorithm) ovat niin sanottuja hash-funktiota. Tällaiset tiivistealgoritmit eivät tarvitse erillistä avainta sanoman tiivistämiseen. Tiivistefunktiot salaavat syötteensä yksisuuntaisesti siten, että saadusta tuloksesta ei voida päätellä mitään syötteestä. Turvallisuuden lisäämiseksi MySQL-palvelimen ja asiakasohjelmien välille on mahdollista rakentaa myöskin SSL-salaus (MySQL, 2004). Turvalli-

suutta voidaan parantaa myöskin hyvillä käytännöillä, esimerkiksi sallimalla tietokannan konfiguroiminen ainoastaan root-oikeuksilla. Kun MySQL-palvelin käynnistetään asennuksen jälkeen ensimmäisen kerran, root-tunnukselle ei ole vielä määritelty salasanaa. Antamalla komento `SET PASSWORD root@localhost=PASSWORD('uusi_salasana')` luodaan salasana pääkäyttäjälle `root`.

3 YMPÄRISTÖN KÄYTTÖÖNOTTO

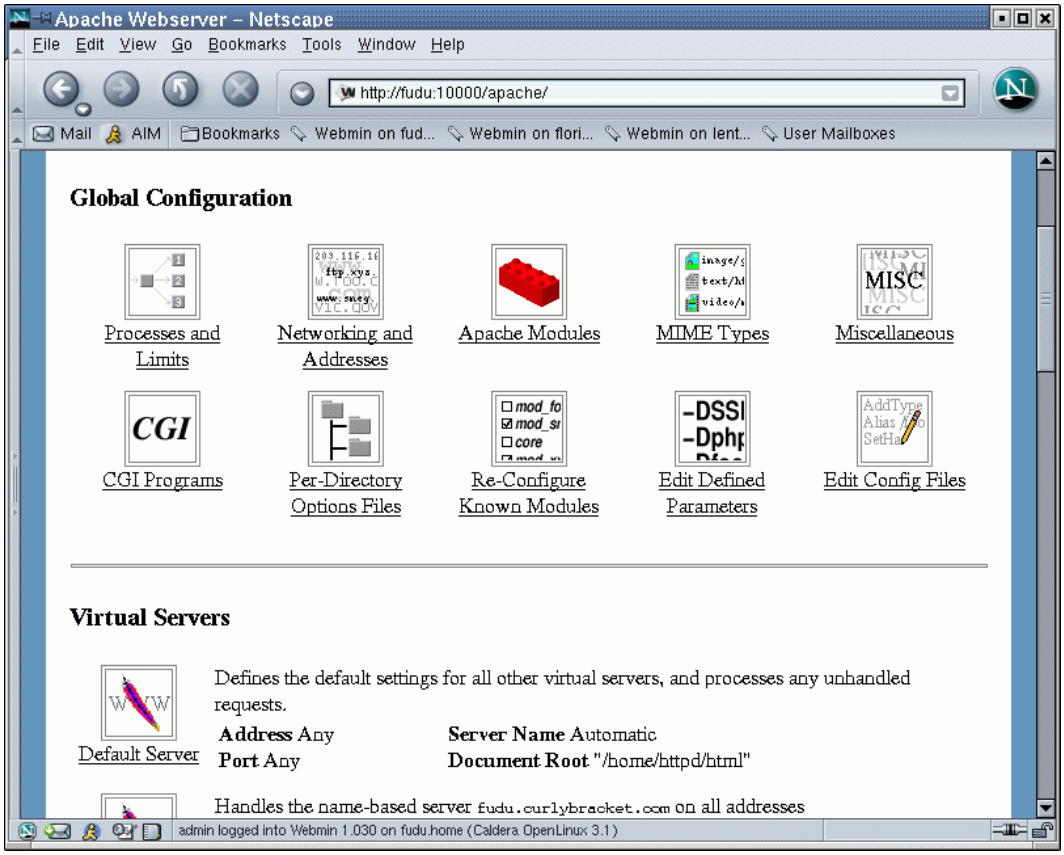
Hyvä WWW-palvelin koostuu selkeästä ja nopeasta ohjelmistosta sekä tarkoituksenmukaisesta laitteistoista, jolloin ohjelmat ja laitteisto saadaan optimoitua vakaaseen yhteistyöhön. On varmistettava, että kaikki komponentit ovat riittävän tehokkaita ja ettei palvelimen toimintaan vaikuttavia pullonkauloja pääse syntymään. Lahtosen (2002) mukaan dynaamisten sivujen tuottaminen ei tuota ongelmia, koska WWW-palvelin suorittaa varsinaisen PHP-koodin ja huolehtii yhteyden ottamisesta tietokantaan. Dynaamisten sivujen generointi on kuitenkin huomattavasti raskaampaa kuin tavallisten staattisten dokumenttien tuottaminen.

3.1 Nimipalvelin

Palvelimen kytkeytyminen julkiseen Internet-verkkoon tapahtuu IP-osoitteen avulla. Domain-nimipalvelut (Domain Name Services, DNS) tarkoittaa järjestelmää, jolla tietokoneet löytävät toisensa Internetissä. Jokainen palvelin omistaa oman IP-osoitteensa (The Linux Documentation Project, 2001). TCP/IP-protokollan ja Internet-tietoliikenteen toiminnan kannalta IP-osoitteet ovat keskeisessä roolissa. IP-osoite on nelitavuinen luku, esimerkiksi `193.167.42.127`, joka on Joensuun yliopiston tietojenkäsittelytieteen laitoksen CS-palvelimen osoite. Nimipalvelimen avulla IP-osoite muutetaan muotoon `www.yritys.fi` ja päinvastoin. Pittsin ja Ballin (1999) mukaan DNS on laaja, jaettu tietokanta, joka sisältää isäntäkoneiden nimien ja IP-numeroiden yhdistelyjä. Linux-alustassa käyttökelpoinen nimipalvelin on Bind. Nimipalvelimia tarvitaan aina kaksi kappaletta, ensisijainen ja toissijainen palvelin, jotka sijaitsevat fyysisesti eri palvelimilla. Tämä varmistaa sen, että mikäli toinen palvelin ei vastaa pyyntöön niin haettu HTTP-pyyntö ohjautuu automaattisesti toissijaiselle palvelimelle ja yhteys haluttuun palvelimeen voidaan toteuttaa. Nimipalvelimen konfiguroiminen tapahtuu muokkaamalla tiedostoa `named.conf`, joka löytyy hakemistosta `/etc/bind/named.conf`.

3.2 Webmin

Webmin on monipuolinen etähallintaohjelma Linux-palvelimelle. Sen avulla voidaan hallita selainpohjaisella liittymällä Apache-palvelinta. Webmin on yksinkertainen WWW-palvelin joka hyödyntää cgi-sovelluksia etähallintatoiminnoissa. Webmin koostuu useista moduuleista. Standardimoduuleilla voidaan hallita järjestelmän peruspalveluja sekä monia verkkopalveluita, kuten Apachea, SQL-palvelimia, DNS-palvelinta ja CVS-versionhallinta palvelinta. Java-tuella varustetulla WWW-selaimella voidaan käsitellä myös järjestelmän tiedostoja suoraan (Webmin, 2004). Oletusarvoisesti Webmin asentuu porttiin 10000. SSL-salaus varmistaa hallinnan tietoturvan käytettäessä Webmin-ohjelmaa verkon yli etähallintana. Mikäli tietoturvaa halutaan parantaa, kannattaa oletusportti vaihtaa esimerkiksi portiksi 20000. Kuvassa 8 on esimerkki Webmin-käyttöliittymästä.



Kuva 8: Webmin käyttöliittymä.

3.3 Apachen hakemistorakenne

Apachen hakemistorakenne on jaettu selkeisiin helposti hallittaviin osiin. Taulukossa 3 on lyhyt esimerkki hakemistoista ja kuvaus niiden sisällöistä.

Taulukko 3: Apachen hakemistorakenne.

Hakemisto	Kuvaus
bin	Binääritiedostot.
cgi-bin	Kansio, johon oletusarvoisesti tallennetaan CGI-ohjelmat.
conf	Kansio, johon oletusarvoisesti tallennetaan konfiguraatiotiedostot.
error	Kansio, johon oletusarvoisesti tallennetaan käyttäjälle näytettävät virheilmoitukset.
htdocs tai web-docs	Kansio, johon oletusarvoisesti tallennetaan julkaistavat HTML-dokumentit. Hakemiston nimi voi olla myös public_html.
icons	Kansio, johon tallennetaan muun muassa dokumentaation käyttämät kuvakkeet.
logs	Kansio, johon oletusarvoisesti tallennetaan erilaiset lokitiedostot.
manuals	Kansio, johon on tallennettu Apachen dokumentaatio.
modules	Kansio, johon tallennetaan kaikki moduulit.

3.4 Tiedostojen ja hakemistojen käyttöoikeudet

Järjestelmän turvallisuuden ja vakauden perustan luovat tiedostojen ja hakemistojen käyttöoikeudet. Tiedostojen käyttöoikeudet jaetaan kolmeen osaan: tiedoston omistaja, tiedoston ryhmä ja kaikki muut. Tiedostojen oikeudet voi tarkistaa komennolla `ls-l` joka kertoo tiedostojen oikeudet koodiriveinä. Hakemistojen käyttöoikeudet ilmaistaan samalla tavalla kuin tiedostojenkin. Lukuoikeus antaa mahdollisuuden listata hakemiston tiedostojen nimet, kirjoitusoikeus antaa mahdollisuuden kirjoittaa hakemistoon uusia tiedostoja ja suoritusoikeus antaa mahdollisuuden tehdä hakemistoja nykyiseen hakemistoon.

Käyttäjät jaetaan kolmeen joukkoon taulukossa 4 ilmenevällä tavalla, jossa joukot ovat toisensa poissulkevia.

Taulukko 4: Hakemistojen ja tiedostojen omistajat sekä ryhmät.

Symboli	Selitys
u	user, omistaja
g	group, omistajan ryhmä
o	others, muut.

Jokaiseen joukkoon liittyy suojauskenttä, joka ilmoittaa mitkä oikeudet kyseisellä joukolla on. Suojauskentän kirjainten merkitykset ilmenevät taulukosta 5.

Taulukko 5: Käyttöoikeudet hakemistolistauksessa.

Symboli	Selitys
r	read, luku
w	write, kirjoitus
x	execute, suoritus.

Jos suojauskentän kohdalla on viiva, niin kyseinen suojaus puuttuu (Hietanen 1996).

4 Ylläpito

Järjestelmän pitäminen ajan tasalla tarkoitta Rantalan (2002) mukaan kaikkien virallisten ohjelmapäivitysten asentamista. Eri Linux-jakeluissa on useita menetelmiä järjestelmän päivittämisen automatisoimiseksi. Etenkin julkisessa verkossa olevien palvelimien pitäminen ajan tasalla tietoturvapäivitysten osalta on tärkeää. MySQL-tietokanta tulee varmuuskopioida säännöllisin väliajoin mahdollisten järjestelmähäiriöiden tai levyrikkojen varalta. Yksinkertainen tapa suorittaa varmuuskopiointi on käyttää `mysqldump` komentoa. Antamalla komento `mysqldump –u –matti –p autokanta > varmuuskopio.sql` luodaan identtinen kopio `autokanta` tietokannasta.

5 Esimerkkisovellus

Esimerkkisovellus perustuu MySQL-tietokantaan, jota käytetään ja hallitaan PHP-lomakkeilla. WWW-selaimella käytettävällä sivustolla pystytään lisäämään, muokkaamaan ja poistamaan tietoa tietokannasta. Käyttäjä-osio listaa tietokannassa olevat autot järjestettynä auton merkin mukaan. Hallinta-osiolla pystytään lisäämään ja poistamaan sekä muokkaamaan autojen tietoja. Tietokanta voidaan luoda `root`-oikeuksilla seuraavan esimerkin mukaan:

Luodaan tietokanta nimeltä **"**autokanta**"**

```
CREATE database autokanta;
```

Annetaan kaikki oikeudet käyttäjänimelle **"**`matti`**"** ja salasanalle **"**`mainio`**"**

```
grant all on * to matti@localhost identified by 'mainio';
```

Kuvassa 9 luodaan taulu nimeltä `henkiloauto`

```
CREATE TABLE henkiloauto (
  id int(11) NOT NULL auto_increment,
  merkki varchar(255),
  rekisteri_numero varchar(11),
  PRIMARY KEY(id)
);
```

Kuva 9: Tietokantataulun luominen.

Sovelluksen hallinta-osio tulee olla ainoastaan tietokannan ylläpitäjän käytettävissä. Hallinta-hakemisto on suojattu käyttämällä `.htaccess` tiedostoa. PHP:n avulla hallinta-osio olisi mahdollista suojata esimerkiksi sallimalla niiden käyttö ainoastaan tietystä IP-osoitteesta. Mikäli ylläpitäjänä on ainoastaan yksi henkilö, on tarkoituksenmukaisempaa käyttää `.htaccess` menetelmää. Tällöin suojaus tapahtuu käyttöjärjestelmätasolla ilman, että varsinaiseen PHP-koodiin tarvitsee tehdä muutoksia. Myös MySQL-tietokantaa on mahdollista

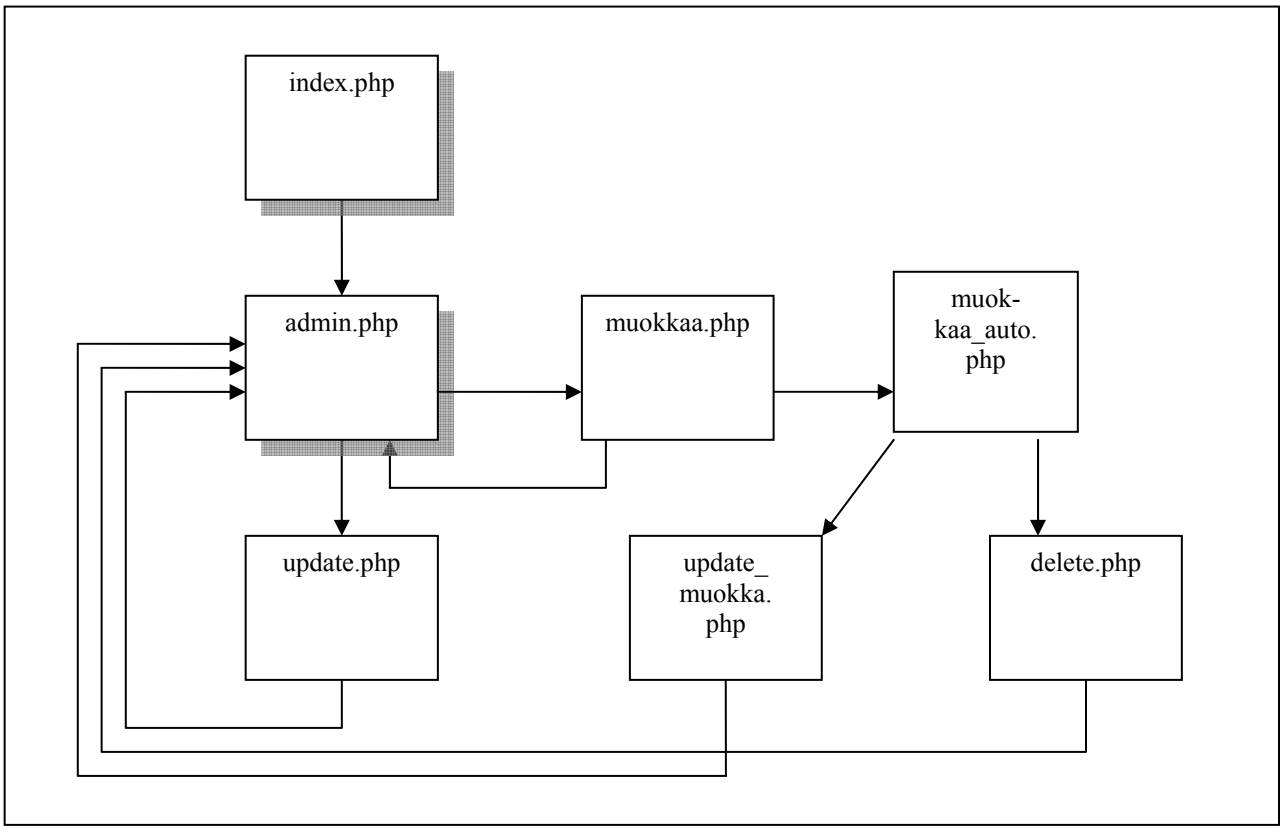
hyödyntää käyttäjätunnistuksessa luomalla erityinen `kayttajat` taulu, joka ylläpitää sallittuja käyttäjätunnuksia ja salattuja salasanoja.

Taulukosta 6 ilmenee sovelluksessa käytettyjen tiedostojen nimet ja tehtävät.

Taulukko 6: Esimerkkisovelluksessa käytettävien tiedostojen kuvaukset.

Tiedoston nimi	Tehtävä
index.php	Toimii sovelluksen etusivuna.
style.inc	Sisältää tyylitiedoston määritte-lyt.
functions.inc	Tässä tiedostossa on funktio(t), joita kutsutaan sovelluksen eri kohdista.
admin.php	Hallinta-osion etusivu, joka si-sältää lomakkeen uuden auton li-säämiseksi tietokantaan.
avaakanta.php	Tiedosto, joka sisältää tietokan-nan avausrutiinit.
update.php	Suorittaa lisäystoiminnon.
muokkaa.php	Hallinta-osion näkymä, jossa lis-tataan kaikki tietokannassa olevat autot. Auton nimi toimii linkkinä muokkaustilaan.
muokkaa_auto.php	Sivu, jossa valitun auton tietoja voidaan muokata tai auto voidaan poistaa tietokannasta.
update_muokkaa.php	Suorittaa pyydetyn SQL-toiminnon.
delete.php	Poistaa auton tietokannasta.
.htaccess	Suojaa hallintakansion käyttöjär-jestelmätasolla.
style.inc	Tiedosto, joka sisältää tyylimää-ritykset.

Sovelluksen sivurakenne on kuvan 10 mukainen.



Kuva 10:Esimerkki sovelluksen sivurakenne.

Liitteet 1 ja 2 sisältävät autotietokanta-sovelluksen lähdekoodin.

6 Yhteenveto

Laajenevassa tietoyhteiskunnassa WWW-palvelimelta vaaditaan kustannustehokkuutta ja joustavuutta. Yritykset ovat alkaneet panostaa verkkosovelluksiinsa yhä enemmän. Dynaamisten WWW-palveluiden avulla voidaan toteuttaa useanlaisia verkkopalveluita erilaisiin käyttötarpeisiin. Palvelimen tulisi olla tehokas, vakaa ja helposti ylläpidettävä. Tähän kehykseen sopivat Linux-pohjaiset palvelinratkaisut. Tutkielmassa on tarkasteltu palvelinympäristön toteutusta ilmaisohjelmien ja komponenttien avulla. Linux, Apache, PHP ja MySQL ovat komponentteja, joilla voidaan toteuttaa toimiva ja vakaa palvelin. Kaikki komponentit voi ladata niiden valmistajien kotisivuilta ilmaiseksi.

Kaupallisten palvelinjärjestelmien ja ilmaistuotteiden valinnan ratkaisee mielestäni kustannustehokkuus ja käyttäjien asenne palvelinratkaisua kohtaan. Kaupallisissa tuotteissa tulee usein mukana sellaisia ominaisuuksia, joita ei koskaan tarvita. Jos ylläpitäjällä on aktiivisuutta perehtyä ilmaistuotteiden konfiguroimiseen, lopputulos on yleensä vakaampi ja helpommin hallittava palvelinkokonaisuus. Tutkielman lopuksi esitetty yksinkertainen tietokantasovellus on esimerkki Apache, PHP ja MySQL ohjelmien helposta yhteensovittamisesta.

Viitteet

Akkanen A., Gröhn A., Hällström J., Jaatinen P., Korhonen A., Korhonen P., Kuittinen J., Lehikoinen J., Okkonen A., Pakarinen P., Tammivuori S. (2002) *PHP-opas.* http://cs.joensuu.fi/~jhalls/php_opas/php_index.html (5.4.2004).

FCS partners (2004) *Apache-palvelimen pystyttäminen.* http://www.fcspartners.fi/koulutus/kurssit/309.html (27.3.2004).

Hakala, M., Kurki-Suonio, J., Kurki-Suonio, K., (1999) *Linux yrityksen avoin vaihtoehto*, Edita, Helsinki.

Heinisuo, R. (2001) *PHP ja MySQL. Tietokantapohjaiset verkkopalvelut.* Satku, Jyväskylä.

Hietanen, R. (1996) *Unix-käyttäjän alkeisopas Helsingin yliopiston Unix-ympäristössä.* http://www.helsinki.fi/atk/oppaat/unix/unixopas.html (26.4.2004)

Kujala, P. (2000) *Dynaamisten WWW sivujen tuottaminen PHP-kielen avulla.* http://www.mit.jyu.fi/pjkujala/studies/LuK (10.4.2004).

Lahtonen, T. (2002) *SQL*. Docendo Finland Oy, Jyväskylä.

Laurie, B., Laurie. P. (1999) *Apache. The definitive guide.* 2nd Edition. O´Reilly & Associates, CA.

Linux Online (2004) WWW-sivusto

http://www.linux.org (8.5.2004).

Leponiemi, J. (2004) *Verkkopalvelut ja PHP*.
http://www11.uta.fi/~jl/php/index.php (8.5.2004).

Mannermaa, M. (1999) *PHP3.* Teknillinen Korkeakoulu.
http://www.tml.hut.fi/Studies/Tik-110.300/1998/Essays/php.html (13.4.2004).

MySQL (2004) *MySQL Reference Manual,*
http://dev.mysql.com/doc/ (10.5.2004).

Netcraft (2004) *Web Server Survey*.
http://news.netcraft.com/archives/web_server_survey.html (5.4.2004).

Peltomäki, J., Linjama, T. (1999) *Linux.* Teknolit Oy, Jyväskylä.

PHP Manual (2004)
http://www.php.net/manual/en/ref.session.php (10.5.2004).

Pitts, D. ja Ball, B. (1999) *Inside Linux.* IT Press, Helsinki.

Rantala, A. (2002) *PHP.* Docendo Finland Oy, Porvoo.

Silvén, O. (1999) *Oulun yliopisto intranet-työryhmän loppuraportti.*
http://herkules.oulu.fi/nbnfi-fe19991195/html/x92.html#AEN143 (24.3.2004).

Sun Microsystems (2004)
http://java.sun.com/products/jdbc/overview.html (17.5.2004).

The Linux Documentation Project (2001) *DNS HOWTO.*
http://www.tldp.org/HOWTO/DNS-HOWTO.html (8.5.2004).

Webmin (2004) http://www.webmin.com (6.5.2004).

Liite 1: Esimerkkisovelluksen lähdekoodi käyttäjät-osiolle

Tiedosto Index.php:

```
<html>
<head>
<title>Database</title>
<? include "style.inc"; ?>
<? include "functions.inc"; ?>
</head>
<body>

<h2>Autotietokanta</h2>
<a href="hallinta/admin.php">Ylläpito</a>
<?php
echo "<hr size=\"1\" noshade>";

include("hallinta/avaakanta.php"); // avataan tietokanta
mysql_connect("$palvelin","$kayttaja","$salasana");
@mysql_select_db("$tietokanta");

/* ------ suoritetaan SQL kysely ------ */
$taulu="henkiloauto";
$kysely="select merkki as 'Merkki',
                rekisteri_numero as 'Rekisterinumero' ";
$kysely .="from $taulu order by merkki ASC";
$tulos=mysql_query($kysely) or die("Kysely epäonnistui!");
if ($tulos != "")
?>

<!------ tulostus kysely taulukkoon ------>
<table width="400" border="0">
<tr>
<?php
while ($kentta=@mysql_fetch_field($tulos)) {
        echo "<th align=\"left\">";
        echo "$kentta->name"; // kentän nimen tulostus
        echo "</th>";
}
```

```
while ($rivi=@mysql_fetch_row($tulos)) {
        echo "<tr bgcolor=\"",vaihda_vari(),"\" valign=\"top\">";
        // tulostetaan joka toinen rivi värillä #33ccff
        for ($i=0;$i<@mysql_num_fields($tulos);$i++) {
          echo "<td>";
          echo "$rivi[$i]";
          echo "</td>";
        }
        echo "</tr>\n";
}
?>
</body>
</html>
```

Tiedosto functions.inc:

```
<?
function vaihda_vari()
{
  static $varikoodi;
  if($varikoodi =="#33ccff")
  {
    $varikoodi="#ffffff";
  }
  else
  {
    $varikoodi="#33ccff";
  }
  return ($varikoodi);
}
?>
```

Tiedosto style.inc:

```
<style type="text/css"><!--
 td,body,input,textarea {
 font-size:12px;
 font-family:Verdana,Arial,Helvetica,sans-serif;
 color:#000000}

 A:link {text-decoration: none; color: #336699;}
 A:visited {text-decoration: none; color: #336699;}
 A:active {text-decoration: none; color: #336699;}
 A:hover {text-decoration: none; color: #336699; text-decoration: under-
line}
--></style>
```

Liite 2: Esimerkkisovelluksen lähdekoodi hallinta-osiolle

Tiedosto admin.php:

```
<html>
<head>
<title>Database</title>
<? include "style.inc"; ?>

</head>
<body>
<h2>Autotietokanta</h2>
<p>
<a href="../index.php">Käyttäjien näkymään</a> |
<a href="muokkaa.php">Näytä kaikki & muokkaa</a>
<hr size="1" noshade></p>
<p>
Lisätään tietokantaan uusi auto
<form name="lisays_lomake" method="post" action="update.php">
  <p>
    <input name="merkki" type="text" id="merkki">merkki</p>
  <p>
    <input name="rekisteri_numero" type="text" id="rekisteri_numero">
    rekisterinumero</p>
  <p>
    <input type="submit" name="Submit" value="Lisää auto">
  </p>
</p>
</form>
</body>
</html>
```

Tiedosto avaakanta.php:

```
<?
// avataan tietokantayhteys
$palvelin=“localhost”;
$kayttaja=“matti”;
$salasana=“mainio”;
$tietokanta=“autotietokanta”;
  mysql_connect(“$palvelin”,”$kayttaja”,”$salasana”);
  @mysql_select_db(“$tietokanta”);
?>
```

Tiedosto update.php:

```
<html>
<head>
<title>database</title>
<? include “style.inc”; ?>

</head>
<body>

<?php
/* ------ avataan tietokantayhteys ------ */
include(“avaakanta.php”);
mysql_connect(“$palvelin”,”$kayttaja”,”$salasana”);
@mysql_select_db(“$tietokanta”);

if (($merkki ==“”) || ($rekisteri_numero ==“”))
echo”Tietojen lisäys epäonnistui, täytä kaikki kentät<br><br>
<a href=\”admin.php\”>Takaisin</a>“;

else {
/* ------ suoritetaan SQL toiminto ------ */
$taulu=“henkiloauto”;
$kysely=“insert “;
$kysely .=“into $taulu (id,merkki,rekisteri_numero)”;
$kysely .=“ values ('$','$merkki','$rekisteri_numero')”;
$tulos=mysql_query($kysely) or die(“Päivitys epäonnistui, anna tiedot uu-
delleen”);
```

```
if ($tulos != "")
echo"Tietojen lisäys onnistui<br><br><a href=\"admin.php\">Takaisin</a>";
}
?>
</body>
</html>
```

Tiedosto muokkaa.php:

```
<html>
<head>
<title>Database</title>
<? include "style.inc"; ?>
<? include "../functions.inc"; ?>
</head>
<body>

<h2>Autotietokanta</h2>
<?php
echo "<a href=\"admin.php\">Takaisin</a>";
echo "<hr size=\"1\" noshade>";

echo "<br>Auton merkki on linkki muokkaustilaan<br><br>";

/* ------ avataan tietokantayhteys ------ */
include("avaakanta.php");
mysql_connect("$palvelin","$kayttaja","$salasana");
@mysql_select_db("$tietokanta");

/* ------ suoritetaan haluttu SQL kysely ------ */
$taulu="henkiloauto";
$kysely="select id , merkki as 'Merkki', rekisteri_numero as 'Rekisterinu-
mero' ";
$kysely .="from $taulu order by merkki ASC";
$tulos=mysql_query($kysely) or die("Kysely epäonnistui");

if ($tulos != "")
?>
```

```
<!------ tulostus kysely taulukkoon ------>
<table width="400" border="0">
<tr>
<?php
while ($kentta=@mysql_fetch_field($tulos)) {
        echo "<th align=\"left\">";
        echo "$kentta->name"; // kentän nimen tulostus
        echo "</th>";
}

while ($rivi=@mysql_fetch_row($tulos)) {
        echo "<tr  bgcolor=\"",vaihda_vari(),"\" valign=\"top\">";
        for ($i=0;$i<@mysql_num_fields($tulos);$i++) {
          echo "<td>";
          if($i==1) echo"<a href=\"muokkaa_auto.php?id=$rivi[0]\">";

          echo "$rivi[$i]";
          echo "</a>";
          echo "</td>";
        }
        echo "</tr>\n";
}
?>
</body>
</html>
```

Tiedosto muokkaa_auto.php:

```
<html>
<head>
<title>Database</title>
<? include "style.inc"; ?>

</head>
<body>
<h2>Autotietokanta</h2>
<?php
echo "<a href=\"muokkaa.php\">Takaisin</a>";

include("avaakanta.php");
mysql_connect("$palvelin","$kayttaja","$salasana");
@mysql_select_db("$tietokanta");
```

```
/* ------ suoritetaan SQL kysely ------ */
$taulu="henkiloauto";
$kysely="select * ";
$kysely .="from $taulu where id = '$id'";

$tulos=mysql_query($kysely) or die("Anna muokattavan auton <b>id</b> nu-
mero <br><a href=\"admin.php\">Takaisin</a>");

if ($tulos != "")
?>
<?php
while ($rivi=@mysql_fetch_row($tulos)) {
        for ($i=0;$i<@mysql_num_fields($tulos);$i++){
                $merkki = "$rivi[1]";
                $rekisteri = "$rivi[2]";
        }
}
?>
<form name="muokkaus_lomake" method="post" action="update_muokkaa.php">
    <input type="hidden" name="id_numero" value="<? echo"$id"; ?>">
  <p>
    <input name="merkki" type="text" id="merkki" value="<? echo"$merkki";
?>">
    merkki</p>
  <p>
    <input name="rekisteri_numero" type="text" id="rekisteri_numero" va-
lue="<? echo"$rekisteri"; ?> ">
    rekisteri numero</p>
  <p>
    <input type="submit" name="Submit" value="Hyväksy muutokset">
  </p>
</form>
<form name="poisto_lomake" method="post" action="delete.php">
  <input type="hidden" name="id_numero" value=" <? echo $id ?> ">
  <p>
    <input type="submit" name="Submit" value="Poista auto">
  </p>
</form>
</body>
</html>
```

Tiedosto update_muokkaa.php:

```
<html>
<head>
<title>database</title>
<? include "style.inc"; ?>

</head>
<body>

<?php
/* ------ avataan tietokantayhteys ------ */
include("avaakanta.php");
mysql_connect("$palvelin","$kayttaja","$salasana");
@mysql_select_db("$tietokanta");

/* ------ suoritetaan SQL toiminto ------ */
$taulu="henkiloauto";
$kysely="update ";
$kysely .="$taulu set merkki=\"$merkki\",
            rekisteri_numero=\"$rekisteri_numero\"";
$kysely .=" where id='$id_numero'";
$tulos=mysql_query($kysely) or die("Päivitys epäonnistui");

if ($tulos != "")
echo "Päivitys onnistui <br><br><a href=\"admin.php\">Takaisin</a>";
?>
</body>
</html>
```

Tiedosto delete.php:

```
<html>
<head>
<title>database</title>
<? include "style.inc"; ?>

</head>
<body>

<?php
/* ------ avataan tietokantayhteys ------ */
include("avaakanta.php");
mysql_connect("$palvelin","$kayttaja","$salasana");
@mysql_select_db("$tietokanta");

/* ------ suoritetaan SQL toiminto ------ */
$taulu="henkiloauto";
$kysely="delete ";
$kysely .="from $taulu";
$kysely .=" where id = $id_numero";
$tulos=mysql_query($kysely) or die("Toiminto epäonnistui, anna tiedot uu-
delleen");

if ($tulos != "")

echo "Poisto onnistui<br><br>";
echo "<a href=\"admin.php\">Takaisin</a>";

?>
</body>
</html>
```