VAASAN AMMATTIKORKEAKOULU

Teemu Haapoja

KESKITETTY KÄYTTÄJÄHALLINTO LDAP-HAKEMISTOLLA

Tekniikka ja liikenne
2005

ALKUSANAT

Tämä opinnäytetyö on tehty Vaasan ammattikorkeakoulun tietojärjestelmähallinnon käyttöön. Työ suoritettiin vuoden 2005 aikana oppilaitoksen tiloissa.

Työn tilasi Vaasan ammattikorkeakoulun tietojärjestelmävastaava Timo Pitkäranta. Työn valvojana toimi professori Riku Jäntti Vaasan yliopistosta.

Kiitokset edellämainituille henkilöille sekä lisäksi Hannu Teulahdelle (Tietoliikennesuunnittelija, Vaasan ammattikorkeakoulu) käytännön ohjeista ja opastuksesta.

Vaasassa 2.5.2006

VAASAN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma

TIIVISTELMÄ

Tekijä	Teemu Haapoja
Opinnäytetyön nimi	Keskitetty käyttäjähallinto LDAP-hakemistolla
Vuosi	2005
Kieli	suomi
Sivumäärä	38 + 7 liitettä
Ohjaaja	Riku Jäntti

Opinnäytetyön tarkoituksena on kehittää Vaasan ammattikorkeakoulun käyttäjähallintoa LDAP-hakemiston avulla.

Työ koostuu seitsemästä eri osa-alueesta: Olemassaolevan järjestelmän dokumentointi, Moodle-järjestelmän LDAP-autentikointiongelmien selvittäminen, käyttäjien luonti WebCT-järjestelmään, uuden työntekijän käyttäjätunnuksen lisäysprosessi, käyttäjätunnusten erääntymistoiminnot, kertakäyttötunnusten hallinta sekä koulun organisaatiokaavion luonti LDAP-hakemiston avulla.

Kukin osa-alue parantaa osaltaan koulun käyttäjätunnusten hallintaa niin ylläpidon kuin käyttäjienkin kannalta sekä saattaa koulun käyttäjähallintoa vastaamaan paremmin HAKA-hankkeen vaatimuksia.

Asiasanat LDAP, käyttäjähallinto

VAASA POLYTECHNIC

Tietotekniikan koulutusohjelma

ABSTRACT

Author	Teemu Haapoja
Topic	Centralized User Management With LDAP Directories
Year	2005
Language	Finnish
Pages	38 + 7 appendices
Supervisor	Riku Jäntti

The goal of this thesis is to develop the existing user account management and related LDAP procedures of Vaasa Polytechnic.

The thesis consists of seven distinct parts: Documenting selected parts of the current user management system, setting up the Moodle system for LDAP authentication, importing users into the WebCT system, software for creating new staff user accounts, expiration of user accounts, management of guest user accounts and creating an organizational chart of the school staff with the information stored in LDAP.

Each part improves the management of user accounts and improves the tasks of IT support staff as well as the users themselves. The project also improves readiness of the polytechnic's user management practices for the HAKA project.

Keywords LDAP, User Management

LYHENNELUETTELO

BIND	LDAP-hakemiston autentikointimenetelmä. Anonymous BIND tarkoittaa yhdistämistä ilman tunnistetietoja.
BNF	Backus-Naur Form, tapa esittää syntakseja
DC	Domain Component
DN	Distinguished Name, yksilöllinen nimi
Hakemisto	puumaisen rakenteen omaava tiedon säilytyspaikka
LDAP	Lightweight Directory Access Protocol (kevyt hakemistoprotokolla)
LDIF	LDAP Data Interchange Format (tiedostomuoto LDAP-datan esitykseen)
OID	Object Identifier
PAM	Pluggable Authentication Module
SSL	Secure Sockets Layer (salattu tiedonsiirtoprotokolla)
Schema	Tiedon rakenteen määritys
UNIX Timestamp	UNIX-järjestelmien eräs tapa ilmaista aikaa. Aika ilmaistaan sekunteina ajanhetkestä 1. tammikuuta 1970 kello 00:00.
URI	Universal Resource Identifier

Sisällys

1 JOHDANTO

Työn tarkoituksena on kehittää Vaasan ammattikorkeakoulun käyttöön ohjelmisto tai ohjelmistoja, sekä parantaa olemassa olevia sovelluksia, joiden avulla koulun sovellusten autentikointi ja käyttäjätunnusten hallinta voidaan hoitaa keskitetysti LDAP-hakemiston (Lightweight Directory Access Protocol) avulla. Näin käyttäjätunnusten hallinta helpottuu ja oikeuksia voidaan hallita yhtenäisesti. Lisäksi ammattikorkeakoulun tavoitteena on liittyä FUNET (Finnish University and Research Network) luottamussopimukseen, ja saattaa järjestelmä sopimuksen auditoinnin vaatimalle tasolle. Sopimus liittyy HAKA-infrastruktuurin toteutukseen, jonka ansiosta eri oppilaitokset voivat jakaa käyttäjätietoja. Luottamussopimuksessa määritellään muun muassa se, kuinka kauan talosta lähteneet ihmiset säilyttävät käyttöoikeuden koulun verkkoon.

Lainattua HAKA-hankkeen [1] sivuilta:

”Haka-infrastruktuuri on Suomen yliopistojen ja ammatti-korkeakoulujen yhteinen käyttäjien tunnistusjärjestelmä, johon liittyneiden korkeakoulujen loppukäyttäjät pääsevät yhdellä käyttäjätunnuksella korkeakoulusektorin palveluihin, riippumatta siitä kuka palvelun tuottaa.”

- Käyttäjällä on yksi käyttäjätunnus ja salasana korkeakoulumaailman kaikkiin palveluihin
- Palveluntarjoajan ei tarvitse ylläpitää käyttäjätunnuksia ja salasanoja. Käyttäjien ajantasaiset henkilötiedot tulevat suoraan kotikorkeakoulun rekistereistä

2 LDAP

LDAP on protokolla, jonka avulla voidaan hakea tietoa yhdestä tai useammasta hakemistosta.

2.1 Historiaa

LDAP:in esi-isä on X.500 [2] hakemistoprotokolla, jonka yhteydessä käytettiin DAP-protokollaa (Directory Access Protocol). Tämä protokolla oli kuitenkin tarpeettoman raskas internetin yli tehtäviin tiedonhakuihin. Tämän vuoksi kehitettiin LDAP-protokolla, joka on yksinkertainen mutta kuitenkin tehokas tapa käyttää hakemistopalveluja.

2.2 Hakemiston toiminta

LDAP-hakemistoa voitaisiin verrata lähinnä sähköiseen puhelinluetteloon. Suurin ero näiden välillä on se, että hakemiston tietoja voidaan lisätä, muokata ja poistaa milloin tahansa, ja nämä muutokset näkyvät välittömästi hakemistossa. Puhelinluettelo joudutaan painamaan uudestaan tietyin väliajoin, jotta muuttuneet tiedot saataisiin julkaistua. Dynaamisen luonteensa ja helposti tehtävien muutosten ansiosta LDAP-hakemistoa voidaan käyttää myös muihin tarkoituksiin kuten käyttäjätunnusten, salasanojen tai muiden tunnistetietojen tallennukseen. Tiedon hakeminen hakemistosta on myös nopeampaa ja kätevämpää kuin puhelinluettelosta, koska haku voidaan suorittaa haluttujen ehtojen perusteella. LDAP-hakemistoa hyödyntävän sovelluksen käyttäjän ei kuitenkaan tarvitse edes tietää käyttävänsä LDAP-hakemistoa, sillä hakemiston tekninen toiminta on normaalisti käyttäjiltä piilotettu. Hakemiston toimintaan liittyvät läheisesti myös tiedon rakenteen määrittelyt eli skeema (schema), joista lisää luvussa 2.9.

2.3 Miksi LDAP?

LDAP-hakemisto on muodostunut erääksi käytännön standardiksi tallentaa organisaation käyttäjätietoja sekä muita oleellisia, harvoin muuttuvia tietoja. Useat sovellukset nykypäivänä tukevat LDAP-autentikointia jollakin tasolla, joten hakemiston avulla voidaan suhteellisen pienellä vaivalla toteuttaa järjestelmä jossa jokaisella ihmisellä on vain yksi käyttäjätunnus ja salasana. Näin myös käyttäjien käyttökokemus paranee, kun heidän ei tarvitse muistaa kuin yksi tunnus ja salasana. LDAP-hakemiston mahdollisia käyttötarkoituksia käyttäjätunnusten hallinnan lisäksi voivat olla seuraavat:

- Käyttäjien mukautetut asetukset (sovelluskohtainen)
- Yksityiset ja julkiset osoitekirjat
- Yksityiset ja julkiset kalenterit
- Sovellusten konfiguraatiotiedot
- DNS-tietokanta (Domain Name System)
- Laitetietokanta
- PKI-avaintietokanta (Public Key Infrastructure)

2.4 Vaihtoehtoja LDAP-hakemistoille

On olemassa myös vaihtoehtoisia hakemistoratkaisuja ja -palvelinohjelmistoja. Sun Network Information Service (NIS) [3], oli yleinen hakemistojärjestelmä vielä viime vuosituhannella, mutta on nykyään melkein kokonaan syrjäytetty LDAP-hakemistoilla. Tunnettu myös nimellä Yellow Pages (yp)

2.5 LDAP-palvelinohjelmistoja

LDAP-protokollaa tukevia palvelimia on sekä avoimeen lähdekoodiin perustuvia että kaupallisia, suljetun lähdekoodin ohjelmistoja. Tässä luvussa tarkastellaan yleisimpiä ohjelmistoja.

2.5.1 OpenLDAP

OpenLDAP [4] on suosittu avoimen lähdekoodin LDAP-palvelin, ja se on myös pääosassa tämän työn toteutuksessa. Palvelin ei vielä tue kaikkia kaupallisten vaihtoehtojen kehittyneimpiä ominaisuuksia, mutta tarjoaa kuitenkin vakaan alustan lähes mihin tahansa tarkoitukseen.

2.5.2 Fedora Directory Server

Fedora/Red Hat Directory Server [5] on toinen avoimen lähdekoodin lisenssillä jaeltava palvelin, joka on aikaisemmin tunnettu Netscape Directory Server-nimellä. Palvelimen kehitys on lähtenyt aikanaan samasta lähdekoodista kuin OpenLDAP, mutta sen kehityksestä on vuosia vastannut pääasiassa Netscape. Red Hat kuitenkin osti sovelluksen Netscapelta, ja julkaisi sen myöhemmin avoimen lähdekoodin lisenssillä. OpenLDAP-palvelimeen verrattuna Fedora DS:stä löytyy lukuisia hyödyllisiä lisäominaisuuksia, joten on mahdollista että OpenLDAP tulee olemaan sivuun jäävä hakemistoratkaisu tulevaisuudessa. Lisäksi Fedora DS-palvelimen hallintatyökalut ovat paljon kehittyneempiä, joten hakemiston toimintakuntoon saaminen on helpompaa ja nopeampaa.

Tätä työtä kirjoitettaessa palvelin on hiljattain julkaistu. Lähdekoodia on toistaiseksi hankalahko saada toimimaan muissa linux-distribuutioissa kuin Redhatissa/Fedorassa, johtuen sen tarkoista vaatimuksista kirjastojen versioista ja distribuutiosta. Ajan kuluessa nämä ongelmat korjautunevat.

2.5.3 Novell eDirectory

Novellin eDirectory [6] on kaupallinen, hyvin tunnettu ja suosittu hakemistopalvelin. Hakemistosta voidaan hakea tietoa monilla eri protokollilla (mm. LDAP, XML, DSML, SOAP, OBDC, JDBC, JNDI, EJB, Active X sekä ADSI). Palvelin on käytössä useissa organisaatioissa.

2.5.4 Microsoft Active Directory

Active Directory [7] on Microsoft Windowsin hakemistopalvelu, joka toimii mm. autentikointipalveluna Windows-domaineille. Standardi LDAP-rajapinta käy sellaisenaan myös Active Directoryn käsittelyyn. Hakemiston rakenne ja tietotyypit eroavat kuitenkin hieman käytännön standardista, joten jos halutaan käyttää AD-hakemistoa autentikointiin, joudutaan usein muokkaamaan asiakasohjelman asetuksia.

2.6 Hakemiston käyttöönotto

LDAP-palvelinohjelmiston asentaminen ja käyttöönotto vaativat asiaan perehtymistä. Tässä työssä käytettävä OpenLDAP ei ole tunnettu helposta asennusprosessista, mutta hakemiston käyntiin saaminen on mahdollista nopeasti jos työkalut ovat tuttuja.

2.6.1 Alkutoimet

Kun palvelimen asennus on saatu suoritettua, ensimmäinen tehtävä on määrittää palvelimen konfiguraatiotiedostot. Tärkeimmät tiedostot ovat */etc/openldap/ldap.conf* (konfiguraatio paikallisille hallintatyökaluille) ja */etc/openldap/slapd.conf* (konfiguraatio palvelimelle). Lisäksi voidaan myös lisätä mahdollisesti tarvittavat laajennukset skeemaan (*/etc/openldap/schema/*).

2.6.2 ldap.conf

Tiedostoon määritellään vähintään seuraavat tiedot:

BASE - Juurisolmun LDAP-polku (esim. dc=puv,dc=fi)

URI - Palvelimen yhteydenottotavat URI-muodossa, esim. ldap://ldap.puv.fi ja/tai ldaps://ldap.puv.fi (ldaps = SSL-suojattu yhteys)

2.6.3 slapd.conf

Tämän tiedoston sisältö määrittelee palvelimen asetukset ja käyttöoikeudet. Tiedoston tärkeimmät direktiivit ovat:

```
include <filename>
```

Tällä komennolla palvelimeen sisällytetään schema-tiedostoja. Esim. include /etc/ openldap/schema/core.schema

```
access to <where> by <who> <level>
```

Tällä komennolla määritetään palvelimen käyttöoikeudet. Oikeudet tarkistetaan ylhäältä alaspäin niin, että vähiten rajoittava määritys täytyy määritellä viimeisenä. Esimerkiksi:

```
access to attrs=userPassword by group="cn=admin,dc=puv,dc=fi" write.
```

Viimeisenä access-rivinä voi olla esimerkiksi:

```
access to dn.base="" by * read
```

```
database <type>
```

Tämä komento määrittää palvelimen tietokannan tyypin. Kaksi yleisimmin käytettyä ovat bdb sekä ldbm.

```
suffix <dn>
```

Määrittää palvelimen juurisolmun

```
rootdn <dn>
```

Määrittää palvelimen superkäyttäjän DN-polun. Tämä on erikoisobjekti, jolla ei ole varsinaista tietuetta palvelimessa. Tällä tunnuksella on rajoittamaton oikeus suorittaa operaatioita palvelimessa.

```
rootpw <password>
```

Superkäyttäjän salasana voi olla joko selkokielinen salasana tai jollakin palvelimen tukemalla kryptausalgoritmilla kryptattu. Joka tapauksessa koska salasana voidaan hakea tiedostosta, tiedoston lukuoikeus tulisi olla vain pääkäyttäjällä.

```
directory <dir>
```

Palvelimen tietokannan sijainti tiedostojärjestelmässä. Esim:

```
directory /var/lib/openldap-data
```

Täydellinen esimerkki konfiguraatiotiedostoista liitteessä 1.

2.6.4 Tietokannan alustus

Seuraavaksi on luotava juurisolmu, jonka alle kaikki muut objektit sijoittuvat. Helpointa on tehdä LDIF-tiedosto, joka syötetään ldapadd-ohjelmalle. Tässä tiedostossa määritellään juurisolmun attribuutit.

```
# Esimerkki Yritys Oy:n LDAP-juurisolmu
dn: dc=esimerkki,dc=org
objectClass: dcObject
objectClass: organization
```

```
dc: esimerkki
o: Esimerkki Yritys
description: Esimerkki Yritys Oy.
```

2.6.5 Palvelimen käynnistäminen

Yleisimmissä Linux-distribuutioissa palvelin käynnistyy komennolla:

```
$ /etc/init.d/slapd start
```

Kun palvelin on käynnistetty, voidaan aloittaa hakemiston rakentaminen. Ensimmäiseksi täytyy luoda palvelimen juurisolmu, jonka määrittely on juuri tehdyssä LDIF-tiedostossa. Tiedosto ajetaan hakemistoon komennolla ldapadd. Esimerkki:

```
$ ldapadd -D "cn=Manager,dc=esimerkki,dc=org" -W -f tiedosto.ldif
```

Parametrien selitykset:

-D Määrittää, millä käyttäjällä operaatio halutaan suorittaa.

-W määrittää, että salasana kysytään käyttäjältä suorituksen yhteydessä.

-f Lukee operaation parametrit annetusta tiedostosta. Ilman tätä parametria ohjelma aloittaa operaation lukemisen näppäimistöltä.

2.6.6 Rakenteen luominen

Kun palvelimen perustiedot on saatu asetettua, voidaan aloittaa luomaan hakemiston rakennetta. Ensimmäisenä perustoimenpiteenä voi olla esimerkiksi Users-haaran luominen, mikäli hakemistoon on tarkoitus tallentaa käyttäjätietoja. Users-haaran luominen tapahtuu seuraavalla LDIF-tiedostolla:

```
dn: ou=Users,dc=esimerkki,dc=org
ou: Users
objectClass: top
objectClass: organizationalUnit
```

Tämä tiedosto ajetaan hakemistoon samalla tavoin kuin edellisessä kohdassa.

2.7 Hallintatyökalut

Hakemiston hallinnan mahdollistamiseksi ja helpottamiseksi on luotu useita työkaluja, joista yleisimmistä kerrotaan tässä luvussa.

2.7.1 Komentorivityökalut

OpenLDAP-palvelimen mukana tulee joukko komentorivipohjaisia työkaluja, joilla voidaan hallita palvelimen toimintaa perustasolla. Tärkeimpiä työkaluja näistä ovat:

- ldapsearch - hae tietoa hakemistosta
- ldapmodify - päivitä hakemiston objekteja
- ldapdelete - poista objekteja hakemistosta
- ldapadd - lisää objekteja hakemistoon

Esimerkki ldapsearch-komennon käytöstä:

```
$ ldapsearch -h 10.0.0.1 -b dc=esimerkki,dc=org uid=*
```

Tämä komento palauttaa kaikki objektit joilla on määritelty *uid*-attribuutti, palvelimelta, jonka IP-osoite on *10.0.0.1*, ja jonka hakemiston juurisolmu on *dc=esimerkki,dc=org*. Tiedot tulostetaan oletuksena LDIF-muodossa.

Palvelimen mukana tulee myös ns. offline-työkaluja. Nämä työkalut eivät tarvitse palvelinyhteyttä, vaan käsittelevät suoraan palvelimen tiedostojärjestelmässä sijaitsevaa tietokantaa. Tämän takia näitä komentoja tulisi käyttää vain silloin, kun palvelin on sammutettuna:

- slapcat - listaa hakemiston sisältö LDIF-muodossa
- slapadd - luo hakemiston sisällön LDIF-tiedoston perusteella

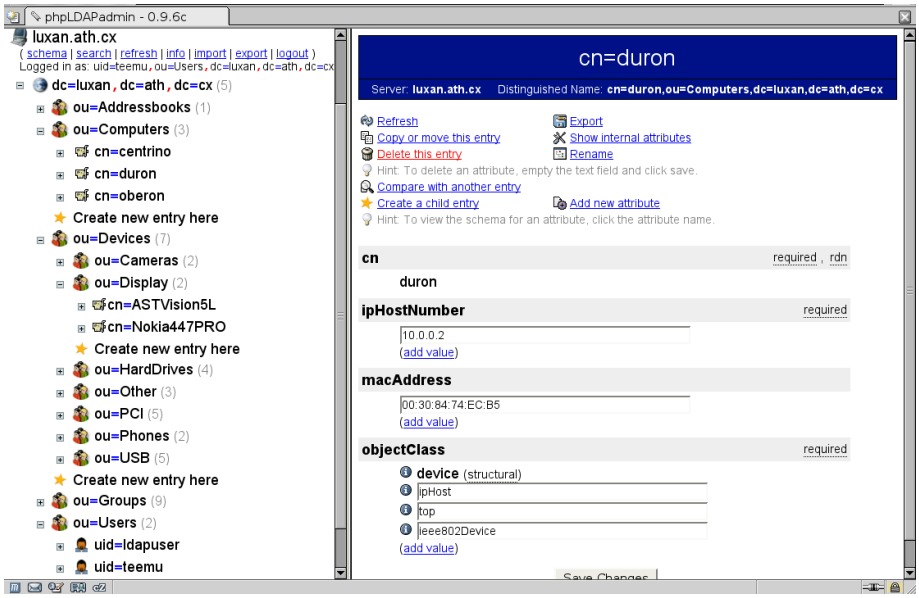
Slapcat ja slapadd ovat tärkeitä hakemiston varmuuskopioinnissa, sillä ne ottavat huomioon myös hakemiston toiminnalliset attribuutit. Toisaalta ne tuottavat tulostuksensa siinä järjestyksessä, missä tietokannassa olevat objektit ovat. Slapcat-ohjelman tuottama tuloste soveltuukin siksi lähinnä slapadd-ohjelman syötteeksi, koska slapadd ei suorita datan eheystarkistusta.

Hakemiston tietokannan varmuuskopiointi slapcat-ohjelmalla:

```
$ /etc/init.d/slapd stop
$ slapcat | gzip > /backup/varmuuskopio-ldap.gz
$ /etc/init.d/slapd start
```

2.7.2 Selainpohjaiset työkalut

Suosituin selaimella käytettävä LDAP-hallintatyökalu on phpLDAPadmin[8]. Sen avulla voidaan nopeasti suorittaa hakuja, lisäyksiä, päivityksiä ja poistoja (Kuva 1). Sovelluksen kirjastoilla voidaan myös luoda mukautettuja WWW-sovelluksia, jotka käsittelevät LDAP-hakemiston dataa (kts. kohta Henkilölomake).



Kuva 1. phpLDAPadmin-työkalun käyttöliittymä

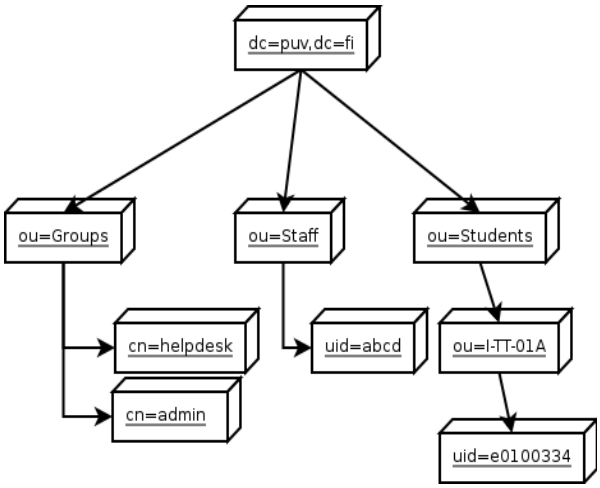
2.8 Hakemiston rakenne

LDAP-hakemiston rakenne suunnitellaan yleensä puumaiseksi, jolloin hakemiston ylimpänä objektina on juurisolmu, jonka alle hakemiston data sijoittuu. Yleensä juurisolmu muodostetaan organisaation domain-nimestä; ammattikorkeakoulun domain-nimi on puv.fi, joten LDAP-hakemiston juurisolmu on *dc=puv,dc=fi*. Yleensä LDAP-hakemisto on jaoteltu organisaatioyksikköihin (Organizational Unit, ou), esimerkiksi henkilökunnan käyttäjätunnukset ovat Staff-nimisen organisaatioyksikön alla (*ou=Staff,dc=puv,dc=fi*).

Sana ”yleensä” esiintyi edellisessä tekstissä useasti siksi, että hakemiston luontivaiheessa ei ole kuitenkaan pakollista tehdä minkäänlaista suunnittelua; kaikki objektit voidaan haluttaessa sijoittaa suoraan juurisolmun alle. Tällaisen "toteutuksen" rajoitukset tulevat kuitenkin hyvin pian vastaan, kun hakemistolta halutaan muutakin kuin yksinkertaisia autentikointipyyntöjä.

Tiedon jaotteleminen puumaisesti helpottaa myös hakuprosesseja, koska haluttaessa haku voidaan kohdistaa vain tiettyyn haaraan. Tällöin haun kannalta epäoleelliset tiedot eivät ole hidastamassa prosessia.

Esimerkki hakemiston rakenteesta (Kuva 2):



Kuva 2. Esimerkki hakemiston puurakenteesta

2.9 Tiedon hakeminen hakemistosta

Tiedon haku LDAP-hakemistosta tapahtuu hakulauseiden sekä muiden parametrien avulla. Tärkeimmät parametrit tiedon haussa ovat haun kohdehaara (base DN) sekä hakusuodin (search filter). Haun kohdehaara ilmaistaan DN-muodossa, esimerkiksi *dc=puv,dc=fi* jos halutaan hakea ammattikorkeakoulun koko hakemistosta. Jos halutaan hakea pelkästään opiskelijoista, voidaan kohdehaaraksi asettaa esimerkiksi *dc=Students,dc=puv,dc=fi*.

2.9.1 Yksinkertainen haku

Yksinkertainen hakusuodin määritellään seuraavasti:

```
parametri=haettava arvo
```

Esimerkiksi:

```
cn=Matti Meikäläinen
```

Jos halutaan yhdistää useampia hakuehtoja, ne sijoitetaan sulkuihin, joissa ilmaistaan haluttu yhdistämistoiminto:

```
(& (uid=mme) (cn=Matti Meikäläinen) )
```

Merkki & (looginen JA)tarkoittaa, että kaikkien kyseisten sulkujen sisällä olevien

hakuehtojen täytyy toteutua. Muita merkkejä ovat | (looginen TAI - jokin ehto täytyy toteutua) ja ! (looginen EI – negaatio). Hakuehtoja voidaan yhdistää myös monimutkaisemmiksi rakenteiksi tarvittaessa, kunhan noudatetaan syntaksin sulkusääntöjä.

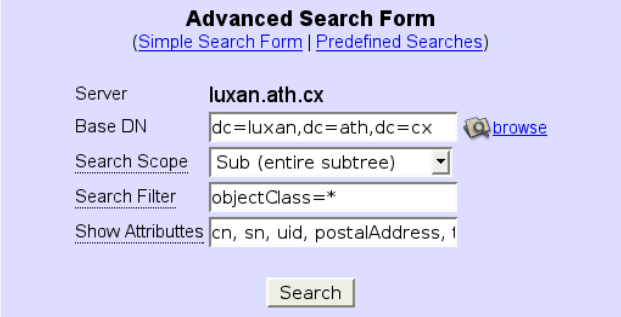
Lisää esimerkkejä hakuehdoista:

```
(& (uid=a*) (! (roomNumber=*)) )
```

Tällä hakulausekkeella voidaan hakea ne käyttäjät, joiden uid-attribuutti alkaa a-kirjaimella, ja joilla ei ole roomNumber-attribuuttia (tarkistus on negaation sulkujen sisällä). Erikoismerkki * vastaa mitä tahansa merkkijonoa.

Haun yhteydessä voidaan scope-parametrilla määrittää myös, haetaanko objekteja pelkästään kohdehaarasta (One), vai myös sen alihaaroista (Sub). Voidaan myös hakea pelkkä kohdehaara (Base), jolloin saadaan kohdehaaran objektin tiedot esiin. Tällöin voidaan käyttää hakuehtoa *objectClass=**.

PhpLDAPadmin tarjoaa helppokäyttöisen liittymän hakujen suorittamiseen (Kuva 3).



Kuva 3. Hakutoiminto phpLDAPadmin-työkalussa

2.9.2 Viittaukset

LDAP-viittaukset (referral) ovat eräs tapa luoda hajautettuja hakemistoja. Viittaus on yksinkertaistettuna palvelimen ohje asiakasohjelmalle ottaa yhteys toiseen palvelimeen ja suorittaa haku saadusta palvelimesta alkuperäisen palvelimen sijaan. Asiakasohjelmassa täytyy olla kuitenki tuki viittauksien käsittelemiseen.

2.10 Skeemat

LDAP-hakemiston skeemat (schema) määritetään yksinkertaisella kuvauskielellä. Tällöin voidaan kuvata eri objektiluokkien väliset yhteydet ja rajoitukset, sekä mitä attribuutteja näihin voi liittää. Objektiluokat voivat periä attribuutteja ylemmältä tasolta kuten olio-ohjelmoinnissakin.

Skeemat määritellään yksinkertaisella määrittelykielellä tiedostoihin, jotka palvelin lukee käynnistyessään. Jotkin palvelinohjelmistot tukevat skeeman muokkausta myös verkon yli. Tässä työssä käytettävä OpenLDAP ei tue tätä ominaisuutta, joten skeeman muokkaukset täytyy tehdä manuaalisesti palvelimen tiedostoihin, ja jokaisen skeeman muutoksen jälkeen palvelin on käynnistettävä uudelleen.

2.10.1 Skeemojen määrittelyt

Skeemojen kuvauksessa tarvitaan määrittelyt attribuuteille sekä objektiluokille. Näiden yhdistelmistä muodostuu toimiva skeema. Yleensä toisiinsa liittyvät attribuutit ja objektiluokat määritellään samassa tiedostossa, joten joitakin skeeman osia voidaan yksinkertaisesti jättää käyttämättä jos palvelimessa ei niitä tarvita.

Skeeman käytössä tärkein osa on OID-numero, joka on uniikki jokaiselle objektiluokalle ja attribuutille. Skeemaa suunnitellessa ei siis voida "arpoa" numeroita, vaan on haettava organisaatiolle oma OID-haara. Jos oman skeeman tekoon on arvottu numeroita, se rikkoo yhteensopivuuden muun maailman kanssa.

Kun organisaatiolle haetaan oma OID-haara, se sijoittuu alueen 1.3.6.1.4.1 alle (Vaasan ammattikorkeakoulun OID-haara on 1.3.6.1.4.1.20644). Käytössä on myös alue 1.3.6.1.3, joka on määritelty kokeelliseksi. Tätä aluetta voidaan vapaasti käyttää skeemojen suunnitteluvaiheesssa, odotettaessa oman OID-haaran saamista. Tätä haaraa ei kuitenkaan suositella tuotantokäyttöön, sillä sen rakennetta ei ole standardisoitu.

2.10.2 Skeeman määrittelykieli

Skeemat määritellään yksinkertaisella määrittelykielellä [9], joka on määritelty tarkemmin RFC2252-dokumentissa [10]. Eri LDAP-palvelimet tukevat yleensä tätä muotoa, mutta joskus saatetaan tarvita pieniä syntaksin muutoksia jos halutaan siirtää skeemamäärittelyjä eri palvelinten välillä. BNF-muodossa

määrittely menee seuraavasti:

```
AttributeTypeDescription = "(" whsp numericoid whsp ; attribuutin
oid
[ "NAME" qdescrs ] ; tyypin nimi
[ "DESC" qdstring ] ; seloste
[ "OBSOLETE" whsp ] ;
[ "SUP" woid ] ; mistä attribuuttityypistä peritään
[ "EQUALITY" woid ; millä metodilla arvoa verrataan
[ "ORDERING" woid ; millä metodilla arvo järjestetään
[ "SUBSTR" woid ] ; millä metodilla arvosta etsitään tietoa
[ "SYNTAX" whsp noidlen whsp ] ; Tyypin syntaksi, määrittelee
minkälaista tietoa attribuuttiin voidaan sijoittaa. Ilmoitetaan
OID-muodossa.
[ "SINGLE-VALUE" whsp ] ; onko tyypin tallennus rajoitettu yhteen
arvoon, vai voidaanko siihen sijoittaa useita arvoja
[ "COLLECTIVE" whsp ] ; default not collective
[ "NO-USER-MODIFICATION" whsp ]; voiko käyttäjä muuttaa arvoa
[ "USAGE" whsp AttributeUsage ]; mihin tarkoitukseen arvo on
tarkoitettu
whsp ")"
AttributeUsage =
"userApplications" / ; käyttäjäsovellukset
"directoryOperation" / ; hakemiston toiminnallisuus
"distributedOperation" / ; DSA-jaettu (Directory System Agent)
"dSAOperation" ; DSA-operaatio, arvo riippuu palvelimesta
```

2.10.3 Esimerkki skeeman attribuuttien määrittelystä

PuvPerson-objektiluokan attribuutteja:

Attribuutti 'puvPersonQuotaU' (U-levyn quota eli käyttäjän tilarajoitus):

```
attributetype (1.3.6.1.4.1.20644.1.1.1
NAME 'puvPersonQuotaU'
DESC 'quota for U-drive in megabytes'
EQUALITY integerMatch ; arvoa verrataan kokonaislukumetodilla
ORDERING integerOrderingMatch ; arvojen järjestys määräytyy
kokonaislukumetodilla
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 ; Käytetään integer-syntaksia
arvon tallennukseen.
SINGLE-VALUE ) ; voi olla vain yksi arvo
```

Attribuutti 'puvPersonAccountExpireWarningSent':

```
attributetype (1.3.6.1.4.1.20644.1.1.2
NAME 'puvPersonAccountExpireWarningSent'
DESC 'account expiration warning sent (unix time)'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE )
```

2.10.4 Esimerkki skeeman objektiluokan määrittelystä

PuvPerson-objektiluokan määrittely:

```
objectclass ( 1.3.6.1.4.1.20644.1.1
NAME 'puvPerson'
DESC 'members of PUV - Vaasa Polytechnic'
AUXILIARY ; luokka ei voi muodostaa yksinään objektia
MAY ( puvPersonQuotaU $ puvPersonAccountExpireWarningSent )) ;
luokkaan kuuluvat ei-pakolliset (MAY) attribuutit. Pakolliset
attribuutit määritellään sanalla 'MUST'
```

2.11 Objektit

LDAP-hakemiston data koostuu tietueista, joilla on vähintään yksi objektiluokka (objectClass). Objektiluokka määrittelee sen, mitä tietoja tietue voi sisältää. Tietueella voi olla myös useampi luokka, mutta näistä vain yksi voi olla rakenteellinen (structural), paitsi siinä tapauksessa että jokin käytetty strukturaalinen objektiluokka perii ylemmän strukturaalisen objektiluokan (person-> organizationalPerson-> inetOrgPerson). Henkilön tunnuksen objektiluokkina on yleisesti person (tai jokin perillinen, esim. inetOrgPerson) ja posixAccount+shadowAccount jos halutaaan ottaa LDAP-autentikointi käyttöön Unix/Linux-järjestelmässä.

2.11.1 Yleisimpiä objektiluokkia

organizationalUnit

OrganizationalUnit on objektiluokka, jonka avulla hakemisto voidaan jakaa eri osiin.

person/inetOrgPerson

Objektiluokkia person tai inetOrgPerson käytetään, kun halutaan tallentaa käyttäjästä mm. osoite, puhelinnumero, paikkakunta, ja muita yksilöllisiä tietoja.

Myös LDAP-hakemistoa hyödyntävät sähköpostiohjelmat yleensä käyttävät tätä tai jotakin perinnäistä.

groupOfNames

GroupOfNames on melko yleinen ryhmän määrittelyluokka. Käyttäjän osallisuus ilmoitetaan member-kentissä, joihin sijoitetaan ryhmän jäsenten täydelliset DN-polut. Tämä ryhmän tyyppi on myös tuettu palvelimen (OpenLDAP) oikeusmäärittelyiden yhteydessä.

posixAccount

PosixAccount on Unix/Posix-käyttäjätunnuksen objektiluokka. Kun halutaan käyttää LDAP-autentikointia Unix-järjestelmissä, täytyy käyttää tätä luokkaa tunnukselle, jotta autentikaatiojärjestelmä löytäisi käyttäjän. Yleensä käytetään myös shadowAccount-luokkaa täydentämään posixAccount-luokan puutteita (tunnuksen erääntyminen, tunnuksen voimassaolo jne.).

posixGroup

PosixGroup on objektiluokka Unix/Posix-käyttäjäryhmille. Käyttäjän osallisuus ilmoitetaan memberUid-kentissä, joihin sijoitetaan ryhmän jäsenten uid-tunnukset. Unix-autentikointijärjestelmä osaa täten sijoittaa käyttäjät oikeisiin ryhmiin. OpenLDAP ei tue tätä ryhmää hakemiston oikeuksien määrittelyyn.

3 AUTENTIKOINTIMENETELMÄT

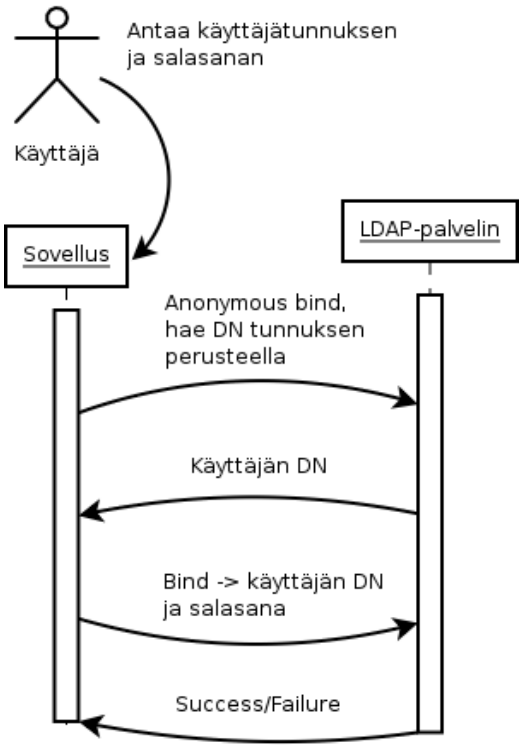
Autentikointi LDAP-hakemistoa vastaan on eräs yleisimmistä tavoista käyttää hakemistoa, ja siihen käyttötarkoitukseen tämäkin työ perustuu.

Yksinkertaisesti ajateltuna, voitaisiin olettaa että hakemiston autentikointi toimii perinteisellä tavalla; haetaan käyttäjän salasana hakemistosta ja verrataan sitä käyttäjältä saatuun salasanaan. Tämä on kuitenkin väärä tapa, sillä hakemisto voi tehdä tämän itsekin.

Edellämainittu prosessi ei sovi LDAP-autentikointiin, sillä tällä tavalla autentikointi ei ole "oikea", eikä palvelimeen määriteltyjä oikeuksia voida käyttää hyväksi. Hakemiston kannalta oikea tapa autentikoida on bind (suomeksi kytkeä, yhdistää). Operaatiossa käytetään siis hakemiston natiivia tapaa autentikointiin, jolloin palvelimeen määritellyt oikeudet ja rajoitukset ovat voimassa (edellyttäen että palvelin on konfiguroitu oikein). Tällöin käyttöön saadaan myös ne käyttäjän attribuutit, jotka on mahdollisesti piilotettu autentikoimattomilta käyttäjiltä.

Kun sovellus haluaa tarkistaa, ovatko käyttäjän antamat tunnistetiedot oikeita, yleinen tapa on käyttää seuraavaa prosessia:

1. Käyttäjä antaa tunnuksen ja salasanan sovellukselle.
2. Sovellus ottaa yhteyden LDAP-palvelimeen anonymous bindillä.
3. Sovellus käyttää palvelimen hakutoimintoa saadakseen selville käyttäjän DN-polun.
4. Jos DN löytyi, yhdistetään uudelleen tällä DN:llä ja käyttäjän salasanalla.
5. Palvelin palauttaa viestin siitä, onko salasana/DN-yhdistelmä oikein.



Kuva 4. Autentikointiprosessin kuvaus

Anonymous bind tarkoittaa käytännössä sitä, että hakemistoon yhdistetään ilman tunnusta ja salasanaa. Se on yleensä tarpeellista, koska aina ei voida tietää käyttäjän täydellistä DN-polkua etukäteen. Tällä tavalla hakemistosta voidaan hakea käyttäjän DN. Jos palvelimen ylläpito on päättänyt estää anonymous-yhteydet, on yleensä myös mahdollista määrittää sovellukseen jokin tietty käyttäjä, jolla on tarvittavat lukuoikeudet hakemistoon. Tämä käyttäjä täytyy määrittää täydellisen DN-polun ja salasanan avulla.

4 LINUX JA LDAP-HAKEMISTO

Linux-järjestelmissä LDAP-autentikointi on hyvin tuettu. PAM-kirjastoon on saatavilla moduulit, joilla paikallisiin tiedostoihin perustuvan autentikoinnin sijaan autentikointi voidaan suorittaa LDAP-hakemiston avulla. Myös ryhmien määrittelyt voidaan sijoittaa hakemistoon, jos luodaan ryhmä, jonka objektiluokkana on *posixAccount*.

4.1 Tarvittavat moduulit

pam_ldap

Tämän moduulin avulla voidaan tehdä liityntä järjestelmän PAM-kirjaston ja LDAP-palvelimen välille

nss_ldap

Tämän moduulin avulla LDAP-hakemistosta voidaan hakea lisätietoa, kuten ryhmätiedot.

4.2 LDAP-autentikoinnin käyttöönotto

Kun edellä olevat kirjastot on asennettu, niiden konfiguraatiotiedostoihin pitää tehdä tarvittavat muutokset, jotta ne pystyisivät kommunikoimaan LDAP-palvelimen kanssa. Alla on lyhyt selostus tarvittavista asetuksista. Tarkempaa tietoa löytyy esimerkiksi Gentoo Linux-distribuution [10] verkkosivuilta [11].

4.2.1 PAM-moduulin asetukset

LDAP-moduulin asetukset täytyy lisätä tiedostoon */etc/pam.d/common-auth* (tai *system-auth*, riippuen distribuutiosta). Esimerkkitiedosto (lisäykset lihavoitu):

```
auth    required    pam_env.so
auth    sufficient  pam_unix.so likeauth nullok shadow
auth    sufficient  pam_ldap.so use_first_pass
auth    required    pam_deny.so

account requisite  pam_unix.so
account sufficient pam_localuser.so
account required   pam_ldap.so

password    required pam_cracklib.so retry=3
```

```
password    sufficient pam_unix.so nullok use_authtok shadow md5
password    sufficient pam_ldap.so use_authtok use_first_pass
password    required pam_deny.so

session required    pam_limits.so
session required    pam_unix.so
session required    pam_mkhomedir.so skel=/etc/skel/ umask=0066
session optional    pam_ldap.so
```

4.2.2 NSS-moduulin asetukset

Tiedosto */etc/ldap.conf* täytyy asettaa niin, että NSS-kirjasto osaa hakea tarvitsemansa tiedot LDAP-hakemistosta:

```
ssl start_tls
ssl on
suffix          "dc=esimerkki,dc=org"

uri ldaps://ldap.esimerkki.org/
pam_password exop

ldap_version 3
pam_filter objectclass=posixAccount
pam_login_attribute uid
pam_member_attribute memberuid
nss_base_passwd ou=People,dc=esimerkki,dc=org
nss_base_shadow ou=People,dc=esimerkki,dc=org
nss_base_group  ou=Group,dc=esimerkki,dc=org
nss_base_hosts  ou=Hosts,dc=esimerkki,dc=org

scope one
```

Tämän lisäksi täytyy vielä asettaa */etc/nsswitch.conf* (muutokset lihavoituna):

```
passwd:         files ldap
group:          files ldap
shadow:         files ldap
```

Näin saadaan käyttöön normaalit UNIX-käyttäjien ominaisuudet LDAP-hakemiston kautta. Jos LDAP-autentikointia halutaan käyttää muissa Linux-koneissa, tulee niihin kopioida tiedostot */etc/ldap.conf* ja */etc/nsswitch.conf*.

5 KÄYTTÄJÄHALLINTO LDAP-HAKEMISTOLLA

Ammattikorkeakoululla on työtä aloitettaessa käytössä monta eri järjestelmää, joissa monessa on omat käyttäjälistansa ja autentikointimenetelmänsä. Ongelmana on, että näiden järjestelmien käyttäjätunnusten hallinta on hankalaa, koska ne eivät (kaikki) osaa käyttää LDAP-hakemistoa suoraan, tai ovat puutteellisia toteutukseltaan. Tämän lisäksi järjestelmä ja ohjelmakirjastot, joilla oppilaiden ja opettajien tunnukset synkronoidaan Winha-järjestelmän ja LDAP- sekä AD-hakemistojen välillä, on dokumentoimaton.

5.1 Hakemistorakenteen dokumentointi

Dokumentoitiin LDAP-hakemistoon Winha-järjestelmästä siirtyvät tiedot sekä jotkin LDAP-hakemistoa käsittelevät ohjelmat.

Ylläpitoon käytettävät ohjelmat on kirjoitettu Perl-kielellä. Seuraavassa on listattu eri tiedostot, niiden sisältämät toiminnot ja niiden käyttötarkoitukset.

5.1.1 PuvWinha.pm

Tällä moduulilla luodaan yhteys koulun Winha-Ville-tietokantaan. Moduulissa määritellään seuraavat funktiot:

pWinha_init

Tarkoitus:	Luo uuden yhteyden koulun Winha-tietokantaan.
Argumentit:	-
Palautusarvo:	Uusi tietokantayhteys.

pWinha_kill

Tarkoitus:	Sulkee tietokantayhteyden Winha-tietokantaan.
Argumentit:	pWinha_init-funktiolla luotu tietokantayhteys-objekti.
Palautusarvo:	-

pWinha_opiskelija

Tarkoitus:	Hakee Winha-tietokannassa olevat opiskelijat.
Argumentit:	-
Palautusarvo:	Hash-objekti opiskelijoista
Lisätietoa:	Funktio käyttää välimuistitiedostoa levyllä, jos sitä ajetaan useammin kuin kerran päivässä.

pWinha_henkilo

Tarkoitus:	Hakee Winha-tietokannassa olevat henkilöstön jäsenet.
Argumentit:	-
Palautusarvo:	Hash-objekti henkilökunnasta.

5.1.2 PuvLdap.pm

Tässä moduulissa on määritelty LDAP-hakemistoon liittyvät toiminnot.

pLdap_init

Tarkoitus:	Luo uuden yhteyden koulun LDAP-palvelimeen.
Argumentit:	-
Palautusarvo:	Uusi LDAP-yhteysobjekti.

pLdap_kill

Tarkoitus:	Sulkee yhteyden LDAP-palvelimeen.
Argumentit:	pLdap_init-funktiolla luotu LDAP-yhteysobjekti.
Palautusarvo:	-

pLdap_hen

Tarkoitus:	Hakee LDAP-hakemistosta henkilökunnan objektit.
Argumentit:	pLdap_init-funktiolla luotu LDAP-yhteysobjekti.
Palautusarvo:	Taulukko henkilökunnan tunnuksista.
Lisätietoa:	Hakemistosta saadulle listalle suoritetaan filtteröinti; vain pienistä kirjaimista koostuvat ja 2-7 merkkiä pitkät tunnukset palautetaan.

pLdap_opisk

Tarkoitus:	Hakee LDAP-hakemistosta opiskelijoiden objektit.
Argumentit:	pLdap_init-funktiolla luotu LDAP-yhteysobjekti.
Palautusarvo:	Taulukko opiskelijoiden tunnuksista.

pLdap_free_uid

Tarkoitus:	Hakee uuden vapaan UNIX uid-numeron.
Argumentit:	pLdap_init-funktiolla luotu LDAP-yhteysobjekti.
Palautusarvo:	Uusi uid-numero.
Lisätietoa:	Funktio käyttää hakemistosta löytyvää lastUidNumber-objektia numeroiden hallintaan. Tästä johtuen kaikkien ohjelmien, jotka luovat uusia UNIX-tunnuksia hakemistoon, olisi hyvä käyttää tätä funktiota.

pLdap_paivita_entry

Tarkoitus:	Päivittää objektin tiedot, mutta ei päivitä niitä hakemistoon.
Argumentit:	LDAP::Entry-objekti, Hash-objekti tunnuksen päivitetyistä tiedoista.
Palautusarvo:	Jos tietoja päivitettiin: päivitetty LDAP::Entry-objekti. Jos tietoja ei päivitetty: undef.

pLdap_make_ou

Tarkoitus:	Luo uuden organisaatioyksikön (OU) LDAP-hakemistoon.
Argumentit:	LDAP-palvelinyhteys, uuden OU:n DN-polku.
Palautusarvo:	-

5.1.3 PuvAd.pm

Tämä moduuli sisältää Active Directoryn käsittelyyn käytettävät funktiot.

pAD_init

Tarkoitus:	Luo yhteyden koulun AD-hakemistoon.
Argumentit:	-
Palautusarvo:	Uusi LDAP-yhteysobjekti.

pAD_kill

Tarkoitus:	Sulkee yhteyden AD-hakemistoon.
Argumentit:	pAD_init-funktiolla luotu LDAP-yhteysobjekti.
Palautusarvo:	-

pAD_hen

Tarkoitus:	Palauttaa AD-hakemistossa olevat henkilökunnan tunnukset.
Argumentit:	pAD_init-funktiolla luotu LDAP-yhteysobjekti.
Palautusarvo:	Taulukko AD-hakemistossa olevista henkilökunnan objekteista.

pAD_opisk

Tarkoitus:	Palauttaa AD-hakemistossa olevat henkilökunnan

tunnukset.
Argumentit: pAD_init-funktiolla luotu LDAP-yhteysobjekti.
Palautusarvo: Taulukko AD-hakemistossa olevista opiskelijoiden objekteista.

pAD_crypt

Tarkoitus: Kryptaa salasanan AD-hakemiston käyttämään muotoon.
Argumentit: Selkokielinen salasana.
Palautusarvo: Kryptattu salasana.

pAD_timeu2w

Tarkoitus: Muuntaa annetun ajan UNIX timestamp-muodosta AD:n käyttämään muotoon.
Argumentit: Aika UNIX timestamp-muodossa.
Palautusarvo: Aika AD-hakemiston käyttämässä muodossa.

pAD_timew2u

Tarkoitus: Muuntaa annetun ajan AD-hakemiston käyttämästä muodosta UNIX timestamp-muotoon.
Argumentit: Aika AD-hakemiston käyttämässä muodossa.
Palautusarvo: Aika UNIX timestamp-muodossa.

5.1.4 PuvPref.pm

Tämä moduuli sisältää joitakin käyttäjien hallinnointiin liittyviä funktioita ja asetuksia.

PUV_WINHA_LASNAOLO_OIKEUS

Tarkoitus: Tarkistaa oikeuttaako annettu läsnäolokoodi läsnäolo-oikeuteen.
Argumentit: Läsnäolokoodi.
Palautusarvo: '1' jos koodi oikeuttaa läsnäoloon, '0' jos koodi ei oikeuta läsnäoloon.

puvPref_winha2ldap

Tarkoitus:	Luo Winha-tietokannassa olevan käyttäjän LDAP-objektin perustiedot.
Argumentit:	Winha-käyttäjän objekti (Hash).
Palautusarvo:	Viittaus Hash-objektiin, joka sisältää luotavan LDAP-objektin perustiedot.

puvPref_winha2ad

Tarkoitus:	Luo Winha-tietokannassa olevan käyttäjän AD-objektin perustiedot.
Argumentit:	Winha-käyttäjän objekti (Hash).
Palautusarvo:	Viittaus Hash-objektiin, joka sisältää luotavan AD-objektin perustiedot.

pPref_uusi_ldap

Tarkoitus:	Luo uuden LDAP-objektin.
Argumentit:	LDAP-yhteysobjekti, luotavan tunnuksen Hash-objekti.
Palautusarvo:	Uusi LDAP::Entry-objekti.

pPref_uusi_ad

Tarkoitus:	Luo uuden AD-objektin.
Argumentit:	AD-yhteysobjekti, luotavan tunnuksen Hash-objekti.
Palautusarvo:	Uusi LDAP::Entry-objekti.

5.1.5 PuvUtil.pm

Tämä moduuli sisältää joitakin yleisesti käytettyjä apufunktioita.

pUtil_8bit_pois

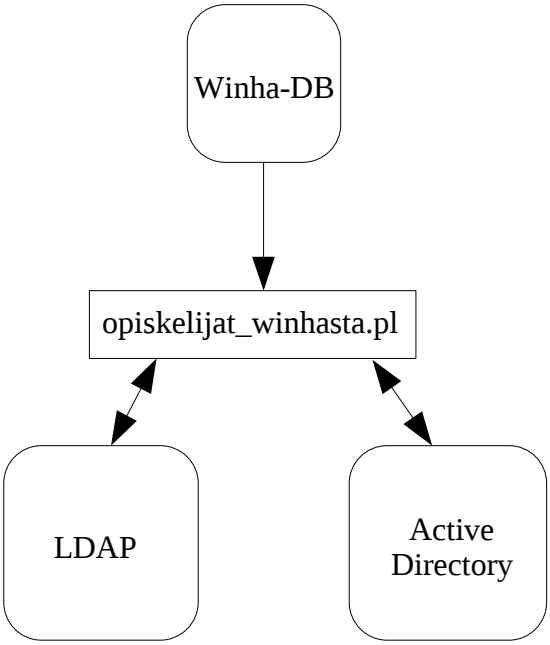
Tarkoitus:	Korvaa annetusta merkkijonosta 8-bittiset merkit niiden 7-bittisillä vastineilla, jotta se voidaan lisätä LDAP-hakemistoon. Tätä ominaisuutta tarvitaan silloin, kun tallennettava kenttä voi sisältää vain 7-bittisiä merkkijonoja.
Argumentit:	Merkkijono.
Palautusarvo:	Merkkijono josta 8-bittiset merkit on korvattu niiden 7-bittisillä vastineilla.

pUtil_assign_home

Tarkoitus:	Antaa annetulle tunnukselle mahdollisen UNIX-kotihakemiston annetun quotan perusteella.
Argumentit:	LDAP-yhteysobjekti, käyttäjätunnus, levyquotan määrä.
Palautusarvo:	Polku uuden käyttäjän kotihakemistoon.

5.1.6 opiskelijat_winhasta.pl

Tämä Perl-skripti luo uusia opiskelijoiden tunnuksia ja päivittää vanhojen tunnusten tietoja Winha-tietokannasta LDAP- ja AD-hakemistoihin ja niiden välillä.



Kuva 5. Tiedon siirtyminen järjestelmien välillä

5.1.7 ldap2ad_henkilokunta.pl

Tämä Perl-skripti luo uusia henkilökunnan tunnuksia ja päivittää vanhojen tunnusten tietoja LDAP-hakemistosta AD-hakemistoon.

5.1.8 Esimerkki kirjastojen käytöstä

Yksinkertainen esimerkki edellämainittujen kirjastojen käytöstä liitteessä 2.

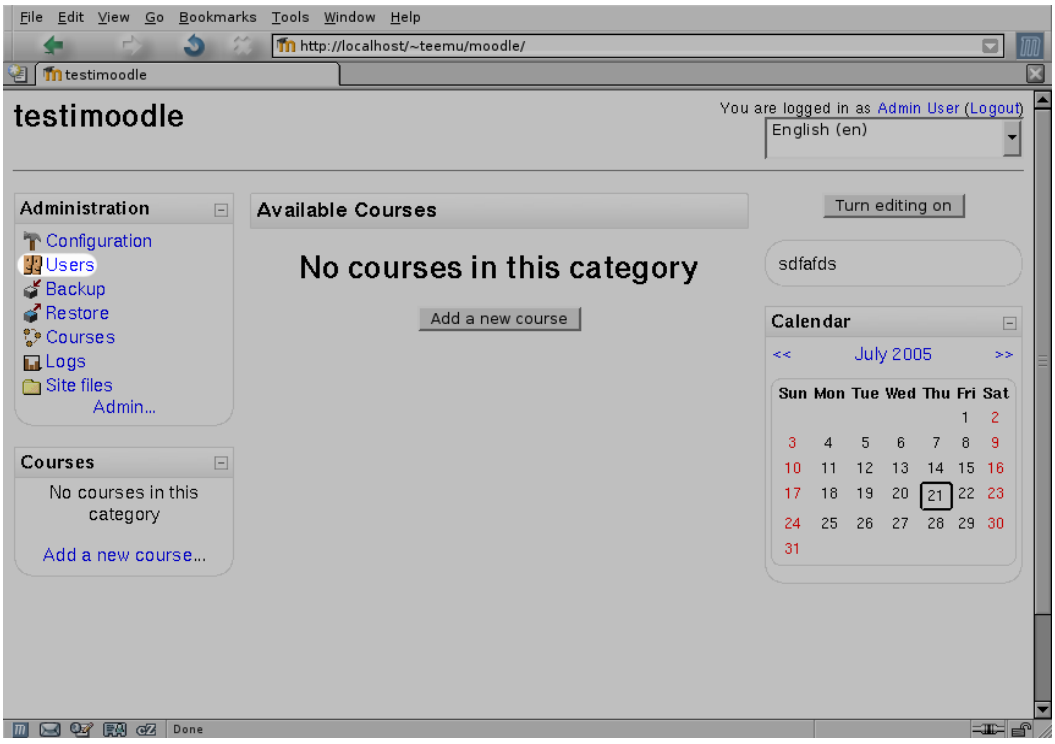
5.2 Moodle-järjestelmän LDAP-autentikointi

Moodle on Open Source-lisenssillä jaeltava virtuaalinen oppimisympäristö.

Nykyinen vakaa (1.4.x) versio hallitsee LDAP-autentikoinnin, mutta siinä on joitakin puutteita; opettajien oikeudet luoda kursseja häviävät heti kun he kirjautuvat sisään. Moodlesta on kuitenkin ilmestynyt uudempi versio (1.5), jossa tämä vika on korjattu. Työn aikana Moodlen LDAP-kirjastosta löytyi myös pieni virhe, joka esti tietyn tyyppisten ryhmien tunnistuksen. Tämä versio tukee myös opettajien määrittelyä LDAP-hakemiston kautta. Seuraavassa ohjeet, kuinka autentikointi voidaan ottaa käyttöön.

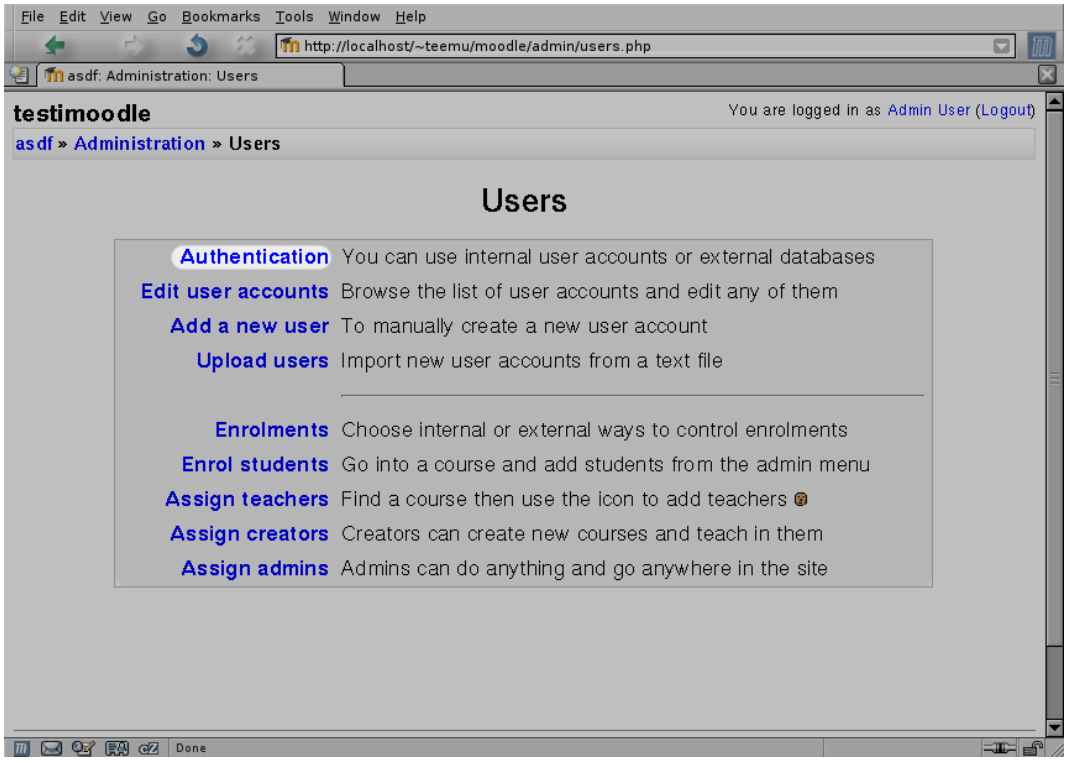
5.2.1 Moodlen autentikointiasetukset

Moodle tukee oletusasennuksessa hyvin monia eri autentikointitapoja, mutta tässä keskitytään LDAP-autentikoinnin asettamiseen. Asetuksen käyttöönottoa varten joudutaan Moodleen kirjautumaan admin-käyttäjän tunnuksella. Kirjautumisen jälkeen suoritetaan seuraavat toiminnot:



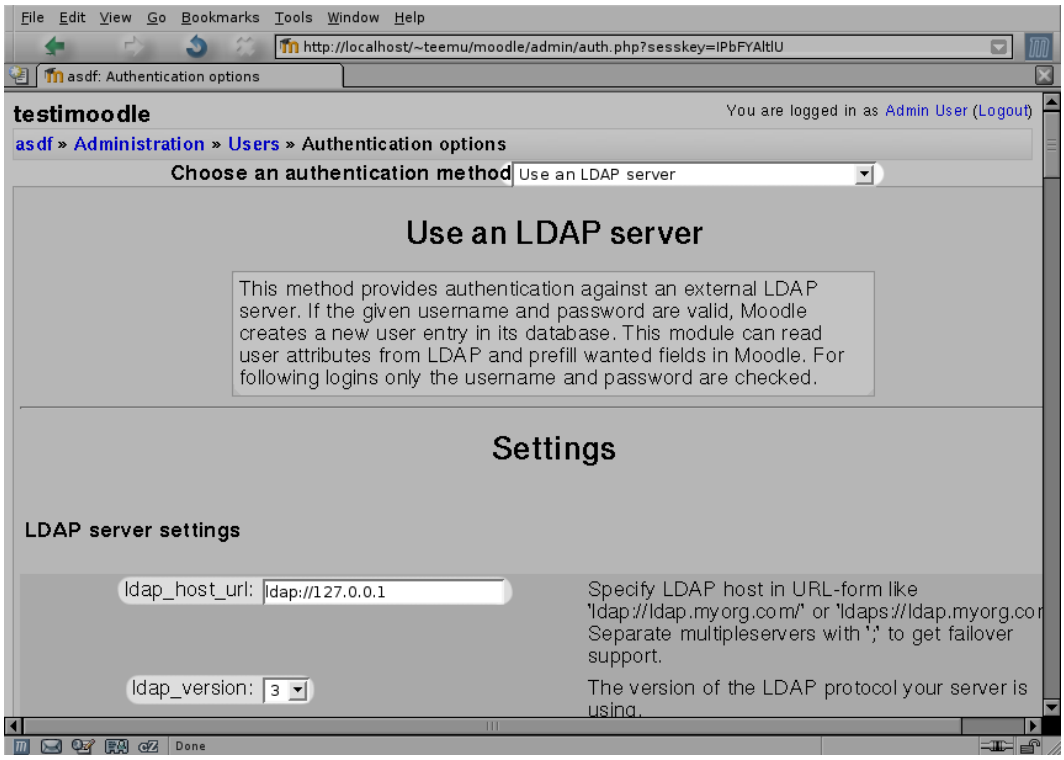
Kuva 6. Moodlen asetukset

Valitaan ylläpitovalikosta "Users"



Kuva 7. Moodlen asetukset jatkoa

Aukeavasta sivusta valitaan "Authentication".



Kuva 8. Moodlen asetukset, jatkoa

Ylhäällä olevasta alasvetovalikosta valitaan "Use an LDAP server".

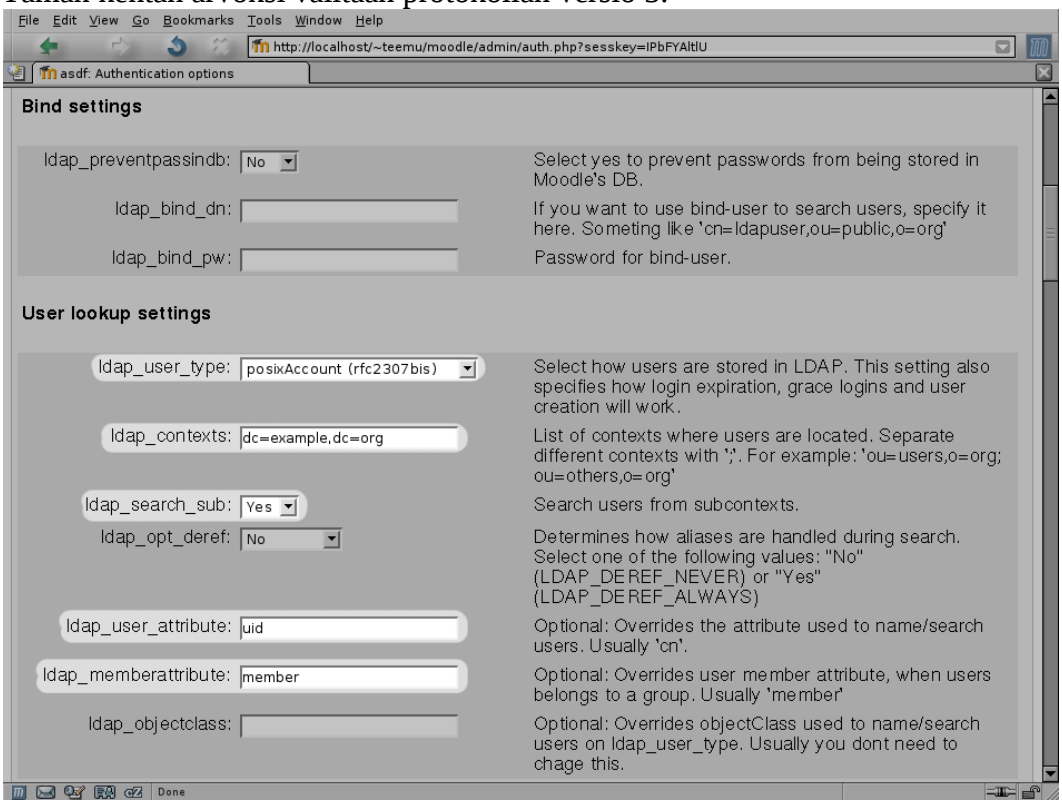
Kenttien asetukset:

ldap_host_url

Tähän kenttään sijoitetaan halutun LDAP-palvelimen URI-polku, esim. *ldaps://ldap.puv.fi*.

ldap_version

Tämän kentän arvoksi valitaan protokollan versio 3.



Kuva 9. Moodlen asetukset jatkoa

Kenttien asetukset:

ldap_user_type

Tämän kentän arvo riippuu siitä, onko opettajien ryhmän tyyppinä *posixGroup* vai *groupOfNames*. Jos tyyppinä on groupOfNames, täytyy valita vaihtoehto *posixAccount (rfc2307bis).* PosixGroup-ryhmän tapauksessa valitaan *posixAccount (rfc2307).*

ldap_contexts

Tähän kenttään sijoitetaan ne LDAP-polut, joista halutaan hakea käyttäjiä (esim. *dc=puv,dc=fi*).

ldap_search_sub

Tämän kentän arvoksi asetetaan "Yes". Arvolla "No" Moodlen LDAP-haku hakee käyttäjiä vain määritellystä kontekstista, eikä etene sen alikonteksteihin.

ldap_user_attribute

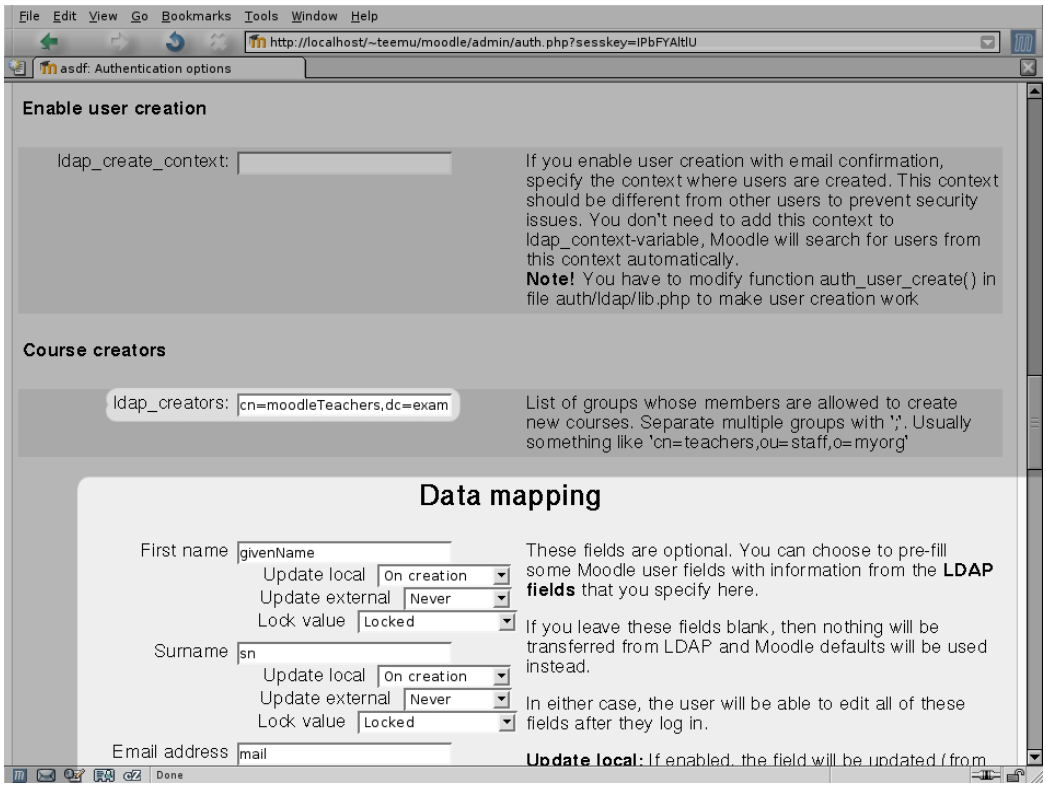
Tämän kentän arvoksi asetetaan "uid". Tämän arvon perusteella Moodle hakee käyttäjän objektia LDAP-hakemistosta.

ldap_memberattribute

Tämän kentän arvoksi asetetaan "member" jos käytössä on *groupOfNames*-ryhmä, "memberUid" jos käytössä on *posixGroup*-ryhmä.

ldap_opt_deref

Tämä kenttä määrittää, miten LDAP-aliakset käsitellään. Jos hakemistossa ei käytetä aliaksia, tämä asetus voidaan jättää pois päältä.



Kuva 10. Moodlen asetukset jatkoa

ldap_create_context

Tämä kenttä ilmaisee, minne uusia käyttäjiä luodaan, jos Moodle on asetettu niin, että käyttäjät pystyvät rekisteröimään itsensä Moodlen tietokantaan. Tämä asetus

ei ole tarpeellinen ammattikorkeakoulun ympäristössä.

ldap_creators

Tämä kenttä ilmaisee mihin ryhmiin kuuluvat käyttäjät saavat luoda Moodleen uusia kursseja. Ryhmän tyypistä riippuen myös muut asetukset pitää asettaa oikein (ylempänä). Jos kenttään laitetaan ryhmiä, joiden tyyppi hakemistossa ei vastaa muita tähän liittyviä asetuksia, tunnistus ei toimi oikein.

Jos ei haluta käyttää LDAP-hakemistoa ryhmän määrittelyyn, tämä kenttä tulee jättää tyhjäksi.

Data mapping

Näihin kenttiin voidaan asettaa ne LDAP-objektin kentät, joista halutaan tuoda tietoa Moodleen. Voidaan myös valita, milloin Moodlessa olevat tiedot päivittyvät ja voiko käyttäjä itse päivittää näitä tietoja Moodlen tietokantaan.

5.3 WebCT-järjestelmän LDAP-autentikointi

WebCT on kaupallinen virtuaalinen oppimisympäristö, joka on toiminnoiltaan vastaava Moodlen kanssa. LDAP-autentikoinnin käyttäminen vaatii lisenssin, jossa sen käyttäminen on sallittu. WebCT käyttää kuitenkin LDAP-autentikointia erittäin suppeasti; käyttäjät pitää joka tapauksessa luoda kumpaankin järjestelmään, koska WebCT ei luo automaattisesti uusia käyttäjiä (vrt. Moodle). Kehitettiin sovellus, joka luo LDAP-hakemistossa olevat tunnukset WebCT-tietokantaan. Sovellus käyttää WebCT:n Standard API-rajapintaa tunnusten luontiin. Sovelluksen lähdekoodi liitteessä 3.

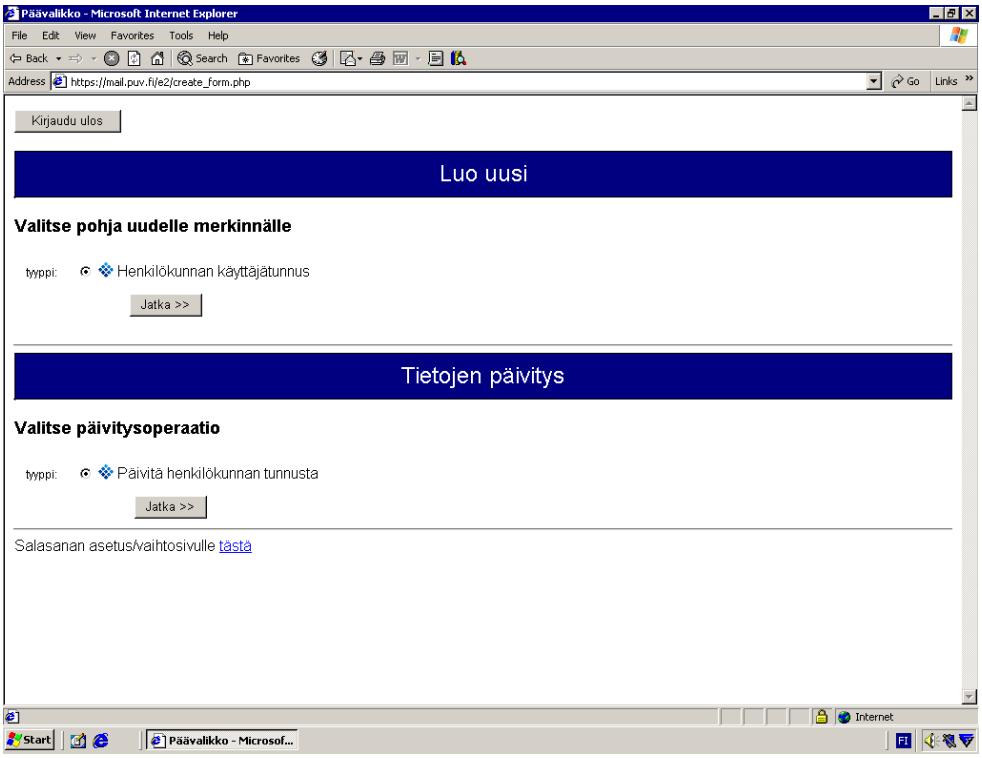
5.4 Työntekijän lisäys

Kun oppilaitoksen palvelukseen palkataan lisää henkilöstöä, tiedon kulkeutuminen koulun järjestelmässä on ollut hankalaa, ja on voinut kestää kauankin ennen kuin nämä henkilöt ovat saaneet käyttöoikeuden koulun tietoverkkoon. Tämän vuoksi kehitettiin WWW-lomake, jonka kautta siihen oikeutetut henkilöt voivat luoda uuden tunnuksen koulun verkkoon.

Lomakkeen suorituksen yhteydessä lähetetään sähköposti-ilmoitus niille, jotka tarvitsevat tietoa uusista työntekijöistä (palkanmaksu, mahdollinen Winha-tunnus, sähköposti). Tällöin uudet työntekijät pääsevät nopeasti työhönsä.

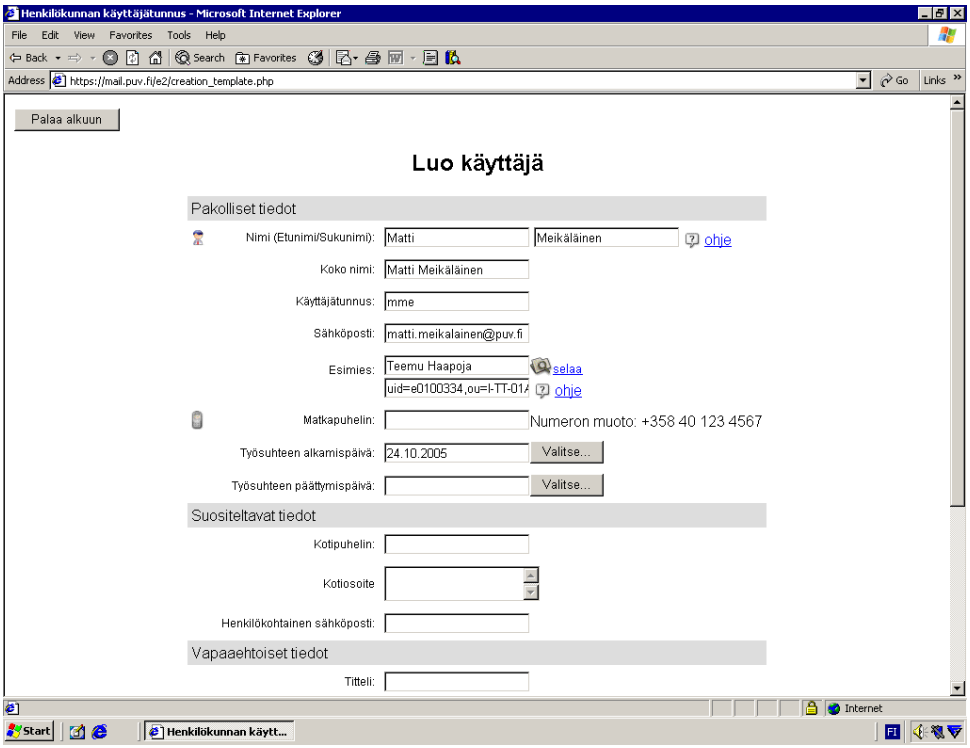
Lomakkeen perustana käytettiin GPL-lisenssillä jaeltavan phpLDAPadmin-työkalun kirjastoja. Tämä mahdollisti työpanoksen asettamisen itse lomakkeeseen ja tarvittaviin lisätoimintoihin, koska tarvittavat LDAP-funktiot olivat jo saatavilla. Toisaalta, joitakin tiedostoja jouduttiin muokkaamaan halutun toiminnallisuuden saavuttamiseksi.

Käyttöliittymä on tehty mahdollisimman yksinkertaiseksi (Kuva 11), ja suurin osa taustatoiminnoista onkin piilotettu käyttäjältä. Mukaan sisällytettiin myös yksinkertainen ohje-toiminto.



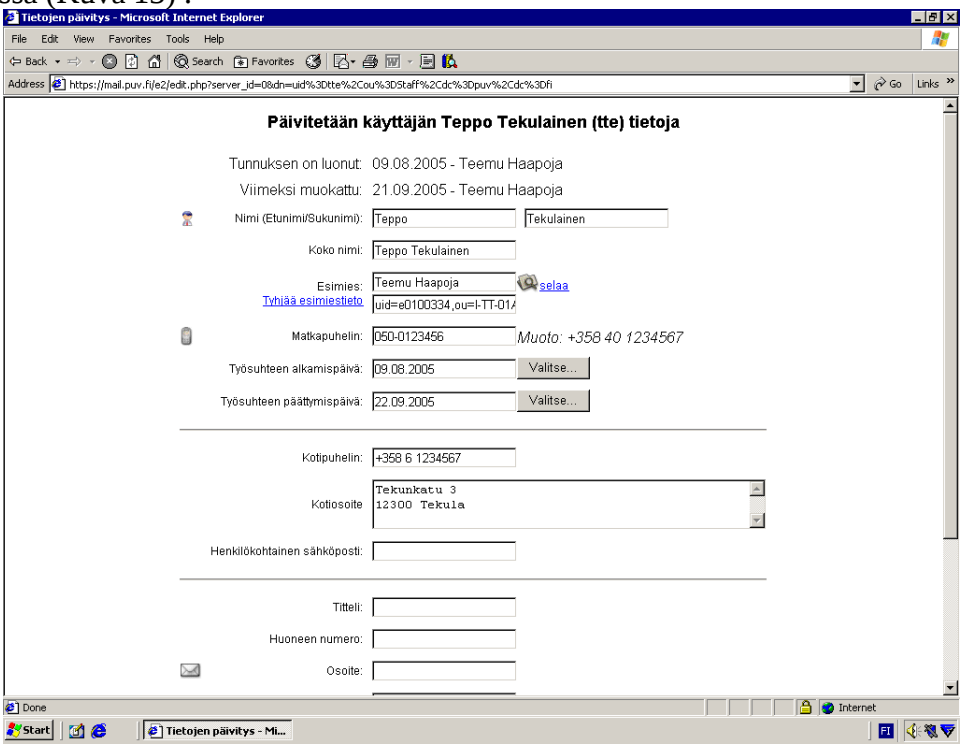
Kuva 11. Henkilölomakkeen aloitussivu

Lomakkeen yhteydessä kysytään myös työsuhteen aloitus- ja lopetusaikaa (Kuva 12). Koska standardi LDAP-skeemassa ei ollut tarvittavaa attribuuttia sellaisenaan, luotiin puvPerson-objektiluokalle kaksi uutta attribuuttia ilmaisemaan työsuhteen alku- ja loppupäivämäärää: puvEmployeeHireStartDate ja puvEmployeeHireEndDate. Tieto näihin kenttiin tallennetaan UNIX-timestamp muodossa.



Kuva 12. Uuden käyttäjän luonti lomakkeella

Samaan sovellukseen tehtiin myös käyttöliittymä henkilökunnan tietojen päivitystä varten. Käyttöliittymä on suurelta osin samanlainen lisäystoiminnon kanssa (Kuva 13) .



Kuva 13. Henkilökunnan tietojen päivitys

Lomake sijaitsee osoitteessa https://mail.puv.fi/e2. Kirjautumista varten käyttäjän

on oltava StaffAdmins-ryhmässä, joka määritellään LDAP-hakemistoon.

5.5 Tunnusten erääntymiset

Kun oppilas valmistuu tai muulla tavalla lopettaa opiskelunsa ammattikorkeakoulussa, hänen tunnustaan pidetään auki vielä tietyn ajan, ennenkuin se poistetaan koulun järjestelmistä. Oppilasta pitäisi muistuttaa, jotta hän ehtisi kopioida mahdolliset sähköpostinsa ja tiedostonsa talteen omalle tietokonelleen. Tähän tarkoitukseen kehitettiin ohjelma, joka tutkii LDAP-hakemistosta oppilaiden läsnäolokoodit ja lähettää varoituksen tunnuksen erääntymisestä hyvissä ajoin. Varoituksen lähetysaika tallennetaan hakemistoon, jotta samalle henkilölle ei lähetettäisi samaa varoitusta aina, kun ohjelma ajetaan päivittäin.

Kun tietty aika on kulunut, käyttäjän tili poistetaan käytöstä. Tällöin käyttäjän LDAP-objekti siirretään haaraan, josta LDAP-autentikointia käyttävät sovellukset eivät pysty sitä lukemaan, joten tunnus on käytännössä poistettu. Haaran piilotus toteutettiin LDAP-palvelimen konfiguraatiotiedoston oikeusmäärittelyjen avulla. Tämä osaltaan ratkaisee myös tilanteen, jossa käyttäjä ottaa valmistumisen jälkeen yhteyttä koululle, ja pyytää tunnustaan käyttöön saadakseen vanhat sähköpostinsa ja/tai tiedostonsa talteen. Tällöin tunnus voidaan aktivoida ylläpidon toimesta tietyksi ajaksi. Liitteessä 4 on listattu järjestelmän ohjelmakoodi.

5.6 Kertakäyttötunnusten Unix-oikeudet

Ammattikorkeakoulun kertakäyttötunnukset ovat tähän asti olleet käytössä vain Windows-järjestelmässä, joten päivitettiin tunnusten luonti koskemaan myös Linux/Unix-järjestelmiä. Näin väliaikaiset käyttäjät saavat oikeuden käyttää mm. kotihakemistoa väliaikaisten tiedostojen tallennukseen sekä oikeuden käyttää UNIX/Linux shell-konetta niin kauan, kuin tunnus on voimassa (Kuva 14).

Tunnusten luontia varten laajennettiin jo käytössä olevaa käyttöliittymää (https://mail.puv.fi/), jonka avulla ylläpito pystyy vaihtamaan käyttäjien salasanoja. Liittymään lisättiin uusi kohta, "Luo kertakäyttötunnuksia", jonka avulla voidaan helposti luoda haluttu määrä uusia kertakäyttötunnuksia. Tunnusten luonnin yhteydessä luodaan tulosteet, jotka voidaan antaa niitä tarvitseville ihmisille. Luoduilla tunnuksilla on myös viimeinen käyttöpäivä, jonka jälkeen ne poistetaan. Tämä on yksi vuosi luontipäivästä. Tämä helpottaa tilannetta, jossa tunnus on jo

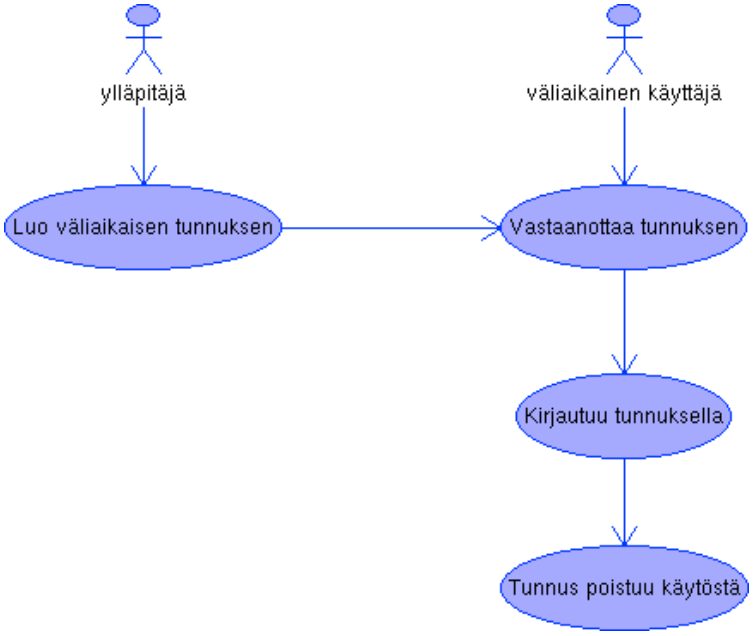
annettu jollekin, mutta sillä ei koskaan kirjauduta (esimerkiksi tilanne, jossa tuloste unohdetaan pöydänlaatikkoon tai se katoaa).

Ammattikorkeakoulun palvelimeen luotiin ohjelma, joka poistaa tunnukset, joita on käytetty. Tunnusten käyttö tunnistetaan LDAP-hakemiston osalta puvPersonAccountExpire-attribuutilla, jota käytettiin edelläolevassa opiskelijoiden tunnusten erääntymisten hallinnassa. Koska kertakäyttötunnukset luodaan eri organisaatioyksikköön kuin opiskelijat, niille ei lähetetä varoituksia tunnuksen erääntymisestä.

Active Directory-hakemistossa on valmiina logonCount-attribuutti kirjautumisen tunnistukseen. Tämän attribuutin perusteella poisto-ohjelma tunnistaa Windows-kirjautumiset. Tämä attribuutti ei kuitenkaan replikoidu eri AD-palvelimien välillä, joten poisto-ohjelman täytyy tarkistaa kaikki käytössä olevat palvelimet saadakseen oikean tiedon tunnuksen tilasta.

Kun kertakäyttötunnuksella kirjaudutaan esimerkiksi shell-koneeseen, siellä oleva PAM-kirjasto aktivoituu. Tähän kirjastoon asennettiin lisämoduli, pam_script, jonka avulla voidaan suorittaa mitä tahansa komentoja kirjautumisvaiheessa, käyttäjän sitä huomaamatta. Kertakäyttötunnusten osalta suoritetaan lyhyt Perl-ohjelma, joka asettaa LDAP-hakemistoon tunnuksen erääntymisajan. Ohjelma tunnistaa käyttäjän ryhmän ja suorittaa sen perusteella määritellyn kirjautumisohjelman.

Koska PAM-kirjastoa käyttävät monet Linux/UNIX-ympäristössä toimivat ohjelmat, sen avulla voidaan samalla tunnistaa myös koulun langattomaan verkkoon kirjautuminen, joka on yleinen käyttötarkoitus kertakäyttötunnuksille. Ne ohjelmat, jotka jotka käyttävät LDAP-hakemistoa itsenäisesti, eivät rekisteröidy PAM-kirjastolle.



Kuva 14. Kertakäyttötunnuksen elinkaari

5.7 Ammattikorkeakoulun organisaatiorakenne

Ammattikorkeakoulun organisaatiorakenteesta ei ole olemassa dokumentaatiota. LDAP-hakemisto onkin hyvä tapa tallentaa nämä tiedot, sillä organisaatiorakenne on suhteellisen harvoin muuttuvaa tietoa. Kun hakemistoon tallennetaan jokaiselle henkilölle esimies, voidaan tästä tiedosta pienehköllä vaivalla rakentaa organisaatiokaavio. Tieto esimiehestä tallennetaan inetOrgPerson-objektiluokan manager-attribuuttiin. Kehitettiin sovellus, joka muodostaa näistä tiedoista XML- tai ASCII-tulosteen, jota voidaan käyttää graafisen esityksen muodostamiseen. Sovelluksen avulla on myös mahdollista hakea hakemistosta ne käyttäjät, joilla ei ole esimiestä, tai joiden esimiestieto on virheellinen.

Esimiestietojen hallinnassa voidaan käyttää tässä työssä ennemmin mainittua henkilökunnan tunnusten luonti- ja muokkauslomaketta.

6 TULOKSET JA JOHTOPÄÄTÖKSET

LDAP-hakemisto on tehokas tapa keskittää käyttäjienhallinta ja -autentikointi organisaation sisällä. Yksinkertainen protokolla ja varma toiminta ovat LDAP-hakemiston vahvoja puolia. Hakemiston toteutukseen ja rakenteeseen on kuitenkin kiinnitettävä huomiota, jos halutaan säilyttää yhteensopivuus muiden järjestelmien kanssa.

LDAP-integraatio koulun järjestelmissä on nyt entistä tehokkaampi ja entistä tiukemmin keskitetty käyttäjähallinto helpottaa niin ylläpidon kuin käyttäjienkin elämää. Myös koulun tietojärjestelmän valmius HAKA-hanketta varten on parantunut.

LÄHDELUETTELO

[13] Donley, Clayton; LDAP programming, management and integration; ISBN: 1930110405

[5] Fedora Directory Server <URL:http://directory.fedora.redhat.com>

[12] Gentoo Guide to OpenLDAP Authentication <URL:http://www.gentoo.org/doc/en/ldap-howto.xml>

[11] Gentoo Linux <URL:http://www.gentoo.org>

[1] HAKA-hanke <URL:http://www.csc.fi/suomi/funet/middleware/haka/>

[6] Novell eDirectory <URL:http://www.novell.com/products/edirectory/>

[4] OpenLDAP <URL:http://www.openldap.org>

[9] OpenLDAP Administrator's guide: Schema specification <URL: http://www.openldap.org/doc/admin22/schema.html >

[8] phpLDAPadmin <URL:http://phpldapadmin.sourceforge.net/>

[10] RFC 2252 <URL:http://www.rfc-editor.org/rfc/rfc2252.txt>

[3] Sun Network Information Service <URL:http://en.wikipedia.org/wiki/Network_Information_Service>

[7] Windows Server 2003 Active Directory - <URL:http://www.microsoft.com/windowsserver2003/technologies/directory/activedirectory/default.mspx>

[2] X.500 <URL:http://en.wikipedia.org/wiki/X.500 >

LIITELUETTELO

ldap.conf:

```
#
# LDAP Defaults
#

# See ldap.conf(5) for details
# This file should be world readable but not world writable.

# SSL/TLS-yhteyden käyttöönotto
TLS_REQCERT  allow

BASE  dc=centrino,dc=luxan,dc=ath,dc=cx
URI     ldap://127.0.0.1 ldaps://127.0.0.1

SIZELIMIT    0
#TIMELIMIT 15
#DEREF                never
```

slapd.conf:

```
# Sisällytetään skeemamäärittelyt

include         /etc/openldap/schema/core.schema
include         /etc/openldap/schema/cosine.schema
include         /etc/openldap/schema/inetorgperson.schema
include         /etc/openldap/schema/nis.schema
include         /etc/openldap/schema/corba.schema
include         /etc/openldap/schema/java.schema
include         /etc/openldap/schema/openldap.schema
include         /etc/openldap/schema/dyngroup.schema
include         /etc/openldap/schema/funeteduperson.schema
include         /etc/openldap/schema/puvperson.schema
include         /etc/openldap/schema/winhaperson.schema
include         /etc/openldap/schema/automount.schema
include         /etc/openldap/schema/eduperson.schema
include         /etc/openldap/schema/trust.schema

# SSL-sertifikaattien määrittelyt
TLSCertificateFile /etc/ssl/ldap.pem
TLSCertificateKeyFile /etc/openldap/ssl/ldap.pem
TLSCACertificateFile /etc/ssl/ldap.pem

# Define global ACLs to disable default read access.


# tiedosto johon palvelimen pid (process id) kirjoitetaan
```

```
pidfile         /var/run/openldap/slapd.pid

# tiedosto josta voidaan lukea palvelimen saamat komentoriviparametrit
argsfile        /var/run/openldap/slapd.args



# Sample access control policy:
#       Root DSE: allow anyone to read it
#       Subschema (sub)entry DSE: allow anyone to read it
#       Other DSEs:
#               Allow self write access
#               Allow authenticated users read access
#               Allow anonymous users to authenticate
#       Directives needed to implement policy:
# access to dn.base="" by * read
# access to dn.base="cn=Subschema" by * read
# access to *
#       by self write
#       by users read
#       by anonymous auth
#
# if no access controls are present, the default policy
# allows anyone and everyone to read anything but restricts
# updates to rootdn.  (e.g., "access to * by * read")
#
# rootdn can always read and write EVERYTHING!
#       by dn="ou=Administrators,dc=centrino,dc=luxan,dc=ath,dc=cx" write

# vain Helpdesk-ryhmän henkilöt voivat lukea käyttäjien salasanakenttiä
access to attrs=userPassword
        by group="cn=Helpdesk,dc=centrino,dc=luxan,dc=ath,dc=cx" write
        by anonymous auth
        by self write
        by * none

# piilotettujen objektien haara joka näkyy vain Helpdesk-ryhmän jäsenille
access to dn.subtree="ou=inactiveUsers,dc=centrino,dc=luxan,dc=ath,dc=cx"
        by group="cn=Helpdesk,dc=centrino,dc=luxan,dc=ath,dc=cx" write
        by * none

# annetaan kaikille oikeus lukea palvelimen skeemaa
access to dn="cn=Subschema"
        by * read


# rajoitetaan luettavia attribuutteja
```

```
access to
attrs=givenName,sn,cn,entry,gecos,gidNumber,homeDirectory,loginShell,member
,memberUid,objectClass,uniqueMember,uid,uidNumber,mail
        by group="cn=Helpdesk,dc=centrino,dc=luxan,dc=ath,dc=cx" write
        by * read

# kaikki voivat lukea palvelimen juurisolmusta, mutta vain Helpdesk-ryhmän
jäsenet voivat kirjoittaa
access to dn.subtree="dc=centrino,dc=luxan,dc=ath,dc=cx"
        by group="cn=Helpdesk,dc=centrino,dc=luxan,dc=ath,dc=cx" write
        by users read

# annetaan kaikille oikeus lukea perus
access to dn.base=""
        by * read


# tietokannan tyyppi
database        bdb
checkpoint      32      30 # <kbyte> <min>
suffix          "dc=centrino,dc=luxan,dc=ath,dc=cx"
rootdn  "cn=Manager,dc=centrino,dc=luxan,dc=ath,dc=cx"
# superkäyttäjän salasana (voi olla myös kryptattu)
rootpw salainen

# The database directory MUST exist prior to running slapd AND
# should only be accessible by the slapd and slap tools.
# Mode 700 recommended.
# tietokannan sijainti tiedostojärjestelmässä
directory       /var/lib/openldap-data

# indekseillä voidaan nopeutta tiettyjen kenttien perusteella tehtäviä hakuja
index   objectClass     eq
index   uid eq
```

Esimerkki koulun LDAP-kirjastojen käytöstä.

```
#!/usr/bin/perl -w
# Esimerkki:
# Päivitetään henkilökunnan huoneiden numerot
# Winha-tietokannasta LDAP- ja AD-hakemistoihin.

use strict;
use lib '/home/teemu/loppari/lib/perl/pm/';
use PuvLdap;
use PuvWinha;
use PuvAD;
use PuvPref;

my $ldap = pLdap_init();
my $ad = pAD_init();
my $huoneet;

my $winha_hen = pWinha_henkilo;
my @ldap_hen = pLdap_hen($ldap);
my @ad_hen = pAD_hen($ad);

for my $uid (keys $winha_hen) {
  # laitetaan käyttäjän huoneen numero listaan
  if ( defined $winha_hen->{$uid}->{puh4} ) {
    $huoneet->{$uid} = $winha_hen->{$uid}->{puh4}
  }
}

# paivitetaan huoneet LDAP-hakemistoon
for my $entry (@ldap_hen) {
  my $uid = $entry->get_value('uid');
  if ( defined $huoneet->{$uid} ) {
    # jos huoneen numero on saatu winhasta, päivitetään
se
    $ldap->modify($entry, replace =>
{'roomNumber',$huoneet->{$uid} } );
  } else {
    # muussa tapauksessa poistetaan huoneen numero
    $ldap->modify($entry, delete => 'roomNumber');
  }
}

# päivitetaan huoneet AD-hakemistoon
for my $entry (@ad_hen) {
```

```
  my $uid = $entry->get_value('CN');
  if ( defined $huoneet->{$uid} ) {
    $ldap->modify($entry, replace =>
    {'physicalDeliveryOfficeName',$huoneet->{$uid} } );
  } else {
    $ldap->modify($entry, delete =>
'physicalDeliveryOfficeName');
  }
}
```

WebCT-synkronisointiskripti

```
#!/usr/bin/perl -w

use strict;
use Net::LDAP;
use Net::LDAP::Util qw(ldap_error_text);
use Unicode::String qw(latin1 utf8);
use LWP;
use HTTP::Request::Common;

our $ldap_bind_dn = 'uid=admin,ou=System,dc=puv,dc=fi';
our $ldap_bind_pw = 'salasana';
our $ldap_bind_host = 'ldap://ldap.puv.fi';
our $ldap_base = "ou=Students,dc=puv,dc=fi";

my $remote_host = 'http://localhost:8900';
my $remote_path = '/webct/public/serve_webctdb';
my $SECRET_FILE = '/home/webct/generic/api/api_secret';
my $md5maker = '/home/webct/generic/api/get_md5';

my $ldap = connect_ldap() or die "Cannot connect LDAP";

my $search = $ldap->search (
  base => $ldap_base,
  filter =>  '(uid=*)',
  scope => 'sub'
);

if( $search->is_error )
{
  die "Search error: ",$search->error, "\n";
}

for ($search->entries) {
  if (!find_webct_user($_->get_value('uid'))) {
    create_webct_user($_);
  }
}

exit;
#####################################################
sub create_webct_user {
  my $entry = shift;
```

```
  my %params;
  $params{'OPERATION'} = 'add';
  $params{'DB'} = 'global';
  $params{'Password'} = '*';
  $params{'WebCT ID'} = $entry->get_value('uid');
  $params{'First Name'} = $entry->get_value
('givenName');
  $params{'Last Name'} = $entry->get_value('sn');

  my $get = "OPERATION=".$params{'OPERATION'};
  $get .= "&DB=".$params{'DB'};
  $get .= "&WebCT%20ID=".$params{'WebCT ID'};
  $get .= "&Password=".$params{'Password'};
  $get .= "&First%20Name=".$params{'First Name'};
  $get .= "&Last%20Name=".$params{'Last Name'};


  my $data_string = $params{'OPERATION'};
  $data_string .= $params{'DB'};
  $data_string .= $params{'WebCT ID'};
  $data_string .= $params{'Password'};
  $data_string .= $params{'First Name'};
  $data_string .= $params{'Last Name'};

  $params{'AUTH'} = `$md5maker $SECRET_FILE
$data_string`;

  $get .= "&AUTH=".$params{'AUTH'};
  my $ua = LWP::UserAgent->new;
  $get = "$remote_host$remote_path?$get";

  my $request = GET $get;

  my $response = $ua->request($request);
  my $content = $response->content();
  if(!$content) {
    print "Connection to $remote_host failed";
  } else {
    if ($content =~ /.*Error.*/) {
      die $content;
    } elsif ($content =~ /.*Success.*/) {
      print "Created WebCT user ".$entry->get_value
('uid')."\n";
      return 1;
    }
```

```
  }
  return 0;
}



sub find_webct_user {
  my $username = shift;
  my %params;
  $params{'OPERATION'} = 'find';
  $params{'DB'} = 'global';
  $params{'COURSE'} = 'global';
  $params{'WebCT ID'} = $username;
  $params{'IMS ID'} = '*';
  $params{'USER_TYPE'} = '0';
  $params{'SEPARATOR'} = ',';



  my $get = "OPERATION=".$params{'OPERATION'};
  $get .= "&DB=".$params{'DB'};
  $get .= "&COURSE=".$params{'COURSE'};
  $get .= "&WebCT%20ID=".$params{'WebCT ID'};


  my $data_string = $params{'OPERATION'};
  $data_string .= $params{'DB'};
  $data_string .= $params{'COURSE'};
  $data_string .= $params{'WebCT ID'};

  $params{'AUTH'} = `$md5maker $SECRET_FILE
$data_string`;

  $get .= "&AUTH=".$params{'AUTH'};

  my $ua = LWP::UserAgent->new;
  my $request = GET "$remote_host$remote_path?$get";

  my $response = $ua->request($request);
  my $content = $response->content();
  if(!$content) {
    print "Connection to $remote_host failed";
  } else {

    if ($content =~ /.*Error.*/) {
```

```
      return 0;
    } elsif ($content =~ /.*Success.*/) {
      return 1;
    }
  }
  return 0;
}
########################

sub connect_ldap {
  # connect ldap
  my $ldap = new Net::LDAP($ldap_bind_host);
  my $result = $ldap->bind( $ldap_bind_dn,
password=>$ldap_bind_pw );
  if( $result->code )
  {
    die "Cannot bind: ",$result->error,"\n";
  }
  return $ldap;
}
```

Tunnusten erääntymisjärjestelmän lähdekoodi

```
#!/usr/bin/perl -w
#
# ldapexpiration.pl
#
# Ohjelma hakee hakemistosta valmistuneet opiskelijat, ja
lähettää heille
# varoituksen tunnuksen erääntymisestä.
#
# Tunnus poistuu käytöstä 7 päivän kuluttua,
# ja poistetaan lopullisesti 180 päivän kuluttua
#
use strict;
use Net::LDAP;
use Net::LDAP::Util qw(ldap_error_text);
use Unicode::String qw(latin1 utf8);

# tarvittavia muuttujia
our $day = (60*60*24);
our $month = $day*30;
our $today_unix = int(time());

# asetukset
our $from = "root\@puv.fi";
our $subject = "Tunnuksesi vanhenee / your account expires";

our $contact = "helpdesk\@puv.fi";

# kuinka kauan odotetaan kunnes tunnus sulkeutuu
our $archive_time = $day*7;

# kuinka pitkaan suljettu tunnus sailyy hakemistossa
our $delete_time = $month*6;

our $ldap_bind_dn = 'uid=Admin,ou=System,dc=puv,dc=fi';
our $ldap_bind_pw = 'salasana';
our $ldap_bind_host = 'ldap://ldap.puv.fi';
# mista kayttajat haetaan
our $ldap_user_base = "ou=Users,dc=puv,dc=fi";
# minne kayttajat arkistoidaan
our $ldap_archive_base = "ou=ArchivedUsers,dc=puv,dc=fi";

our $ad_bind_dn =
```

```
'CN=Administrator,CN=Users,DC=puv,DC=fi';
our $ad_bind_pw = 'salasana';
our $ad_bind_host = 'dcwolf.puv.fi';
our $ad_base = "CN=Users,DC=puv,DC=fi";

########################################
#
my ($ldap,$ad) = connect_ldap();

our $warnings_sent = 0;
our $users_archived = 0;
our $users_deleted = 0;

our $dryrun = 0;

if ( ($#ARGV >= 0) and ($ARGV[0] eq '-dry') ) {
  $dryrun = 1;
  print "Dry run\n";
}
########################################
#

# poisto
my @deleteusers = findUsersForDeletion($ldap);
for (@deleteusers)
{
  if ($dryrun == 0) {
    deleteUser($ldap,$ad,$_);
    $users_deleted++;
  }
  else
  {
    print "User ".$_->get_value('uid')." would be deleted.\n";
  }

}

# arkistointi
my @archive_users = findUsersForArchiving($ldap);
for (@archive_users) {
  if ($dryrun == 0)  {
    archiveUser($ldap,$ad,$_);
    $users_archived++;
  }
```

```
  else {
    print "User ".$_->get_value('uid')." would be archived.\n";
  }
}

# varoitus
my @warn_users = findUsersForWarning($ldap);
for (@warn_users)
{
 my $entry = $_;
 my $uid = $entry->get_value('uid');
 my $warning_sent = $entry->get_value
('puvPersonAccountExpireWarningSent');


 if( !$warning_sent ) {
    if ($dryrun == 0) {
    sendWarning($ldap,$ad,$entry);

    $warnings_sent++;
    } else {
      print "User $uid would get a warning\n";
    }

 }
}


print "Expire warnings sent: ",$warnings_sent,"\n";
print "Users archived: ",$users_archived,"\n";
print "Users deleted: ",$users_deleted,"\n";
##########################################
######################
sub deleteUser {
    my $ldap = shift;
    my $ad = shift;
    my $entry = shift;

    my $uid = $entry->get_value('uid');
    system "./poista_tunnus.pl $uid";
}


sub archiveUser {
```

```
    my $ldap = shift;
    my $ad = shift;
    my $entry = shift;

    # ldap
    my $uid = $entry->get_value('uid');
    my $cn  = utf8($entry->get_value('cn') )->latin1;
    my $dn  = $entry->dn();

    # move entry
    ###
    my $newrdn = "uid=${uid}";
    print "Archiving user $uid ($cn)\n";
    my $mesg = $ldap->moddn(    $dn,
                            newrdn => $newrdn,
                            newsuperior => $ldap_archive_base,
                            deleteoldrdn => '1',
                            );
    $mesg->code && warn "Failed to move entry: ", $mesg-
>error ;


    # active directory
    my $adsearch = $ad->search(
      base => $ad_base,
      filter =>  "(CN=$uid)",
      scope => 'sub'
    );

    if ($adsearch->is_error ) {
        die "AD search error: ",$adsearch->error,"\n";
    }
    for ($adsearch->entries) {
        my $ad_expire = $_->get_value('userAccountControl');
        if (int($ad_expire) & 0x2) {
            print "User already disabled on AD";
        }
        # Aseta 2. bitti päälle tunnuksen deaktivoimiseksi
        $ad_expire = $ad_expire | 0x2;
        $ad->modify($_,  replace => {'userAccountControl' =>
$ad_expire} );
    }

}
```

```
# send a warning to user, modify this to add new features
sub sendWarning {
    my $ldap = shift;
    my $ad = shift;
    my $entry = shift;

my @finnish_days = (  'sunnuntaina',
                      'maanantaina',
                      'tiistaina',
                      'keskiviikkona',
                      'torstaina',
                      'perjantaina',
                      'lauantaina');
my @english_days = (  'Sunday',
                      'Monday',
                      'Tuesday',
                      'Wednesday',
                      'Thursday',
                      'Friday',
                      'Saturday');
# my @swedish_days = ('måndag',
#                     'tisdag',
#                     'onsdag',
#                     'torsdag',
#                     'fredag',
#                     'lördag',
#                     'söndag' );
  # aseta erääntymisaika
  my $expire = int(time()+$archive_time);

  if (!$entry->get_value('puvPersonAccountExpire')) {
    $ldap->modify($entry, add => { puvPersonAccountExpire =>
$expire } );
  }

  $ldap->modify($entry,  add =>
{ puvPersonAccountExpireWarningSent => time() } );

  # get some necessary data
  my $mail = $entry->get_value('mail');
  my $uid = $entry->get_value('uid');
  my $cn = utf8( $entry->get_value('cn') )->latin1;
  my $phonenumber = $entry->get_value('mobile');
```

```
  if (!$phonenumber) { $phonenumber=""; }


  print "Sending warning to $uid ($cn) $mail $phonenumber\n";

  my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst)
= localtime($expire+$delete_time);
  $mon += 1;

  $year += 1900;


  my $msg2 = "uid: $uid\nmail: $mail\nexpire: ${mday}.
${mon}.${year}.";

  my $msg = <<EOT;
Käyttäjätunnuksesi $uid on merkitty poistettavaksi
$finnish_days[$wday] ${mday}.${mon}.${year}.
Muista kopioida tärkeät sähköpostisi ja tiedostosi talteen!

Tämä on automaattinen ilmoitus, johon ei tarvitse vastata.

-------------------------------------------------------------------------------

Your user account $uid has been marked for deletion on
$english_days[$wday] ${mday}.${mon}.${year}.
Remember to back up your important emails and files!

This is an automatic announcement, please do not reply.

EOT
  print $msg;
  # laheta sahkoposti / tekstiviesti

  send_mail($from,$mail,$subject,$msg);
}

sub connect_ldap {
    # connect ldap
    my $ldap = new Net::LDAP($ldap_bind_host);
    my $result = $ldap->bind( $ldap_bind_dn,
password=>$ldap_bind_pw );
    if( $result->code )
    {
```

```
        die "Cannot bind: ",$result->error,"\n";
    }

    # connect ad
    my $ad = new Net::LDAP($ad_bind_host);
    my $ad_result = $ad->bind( $ad_bind_dn,
password=>$ad_bind_pw);
    if ( $result->code ) {
        die "Cannot bind to AD: ",$ad_result->error,"\n";
    }


    return ($ldap,$ad);
}

sub findUsersForDeletion {
    my ($ldap) = @_;

  my @userlist;
  my $usercount=0;

  my $searchdel = $ldap->search (


  base => $ldap_archive_base,
    filter =>  '(&(objectClass=puvPerson)
(puvPersonAccountExpire=*))',
    scope => 'sub'
  );

  if( $searchdel->is_error )
  {
    die "Search error: ",$searchdel->error, "\n";
  }

  for ($searchdel->entries) {
    my $expire = $_->get_value('puvPersonAccountExpire');
    my $estimate = int( ($delete_time+$expire)-$today_unix);
    if ( $estimate < 0 ) {
      print "User: ",$_->get_value('uid')," marked for deletion\n";
      $userlist[$usercount] = $_;
    }
    else {
      print $_->get_value('uid')," will be deleted in ".int
```

```
($estimate/$day)." days\n";
    }
  }
  return @userlist;
}


sub findUsersForArchiving {
  my ($ldap) = @_;
  my @user_list;
  my $archivecount = 0;

  my $search = $ldap->search (
    base => $ldap_user_base,
    filter => '(& (objectClass=puvPerson)
(puvPersonAccountExpire=*) )',
    scope => "sub" );

  if( $search->is_error )
  {
    die "Search error: ",$search->error, "\n";
  }

  for ($search->entries) {
    my $warned_timestamp = $_->get_value
('puvPersonAccountExpire');
    if ($warned_timestamp) {
      my $estimate = int( ($warned_timestamp-$today_unix) /
$day);
      if ($estimate < 0)
           {
        $user_list[$archivecount] = $_;
        $archivecount++;
      }
      else {
        print $_->get_value('uid')," will be archived in $estimate
days\n";
      }

    }
  }
  return @user_list;
}
```

```
sub findUsersForWarning {
   my ($ldap) = @_;
    my $search_filter = '(& (winhaAttendanceCode=*)
(objectClass=puvPerson) )';
    my $scope = 'sub';
    #my $filter = "(shadowExpire=$today)";
    my @user_list;
    my $usercount = 0;

    my $search = $ldap->search (
        base => $ldap_user_base,
        filter => $search_filter,
        scope => "sub" );

    if( $search->is_error )
    {
        die "Search error: ",$search->error, "\n";
    }

    for ($search->entries) {
        my $attendance = utf8( $_->get_value
('winhaAttendanceCode') )->latin1;
        if (    $attendance &&
                (0 == PUV_WINHA_LASNAOLO_OIKEUS
($attendance) ) ) {
                $user_list[$usercount] = $_;
                $usercount++;
            }
    }
    return @user_list;
}

sub send_mail {
  my $from = shift;
  my $to = shift;
  my $subject = shift;
  my $msg = shift;

  my $smtp_host = "localhost";
  my $smtp = Net::SMTP->new($smtp_host) ;
  die "Couldn't connect to server" unless $smtp ;
  $smtp->mail($from);
  $smtp->to($to);
```

```
  $smtp->data();
  $smtp->datasend("From: $from\n");
  $smtp->datasend("To: $to\n") ;
  $smtp->datasend("Subject: $subject\n") ;
  $smtp->datasend("\n") ;
  $smtp->datasend("$msg\n\n") ;
  $smtp->dataend() ;

  $smtp->quit() ;
}

##############################
# TODO: vaihda tama toimimaan moduulista
sub PUV_WINHA_LASNAOLO_OIKEUS {

 my $koodi  = shift;
 my %koodit = (
  'PE'  => 0,
  'ER'  => 0,
  'VP'  => 0,
  'LÄ'  => 1,
  'T3'  => 1,
  'TH'  => 1,
  'OV'  => 1,
  'XX'  => 1,
  'T1'  => 1,
  'T2'  => 1,
  'T4'  => 1,
  'T5'  => 1,
  'PO'  => 1,
  'LO'  => 1,
  'ESR' => 1,
  'YA'  => 1,
  'YK'  => 1,
  'YV'  => 1,
  'KU'  => 1
 );
 if ( defined( $koodit{$koodi} ) ) {
  return $koodit{$koodi};

 } else {
  die "Tunnistamaton läsnäolokoodi: ", $koodi, "\n";
 }
```

```
}
```

```
#!/usr/bin/perl -w
#
# palauta_tunnus.pl
#
# Palauttaa käyttäjätunnuksen käyttöön annetuksi ajaksi
#
use strict;
use Net::LDAP;
use Net::LDAP::Util qw(ldap_error_text);
use Unicode::String qw(latin1 utf8);

our $temp_ou = "ou=RestoredUsers,dc=puv,dc=fi";
our $inactive_ou = "ou=Archive,dc=puv,dc=fi";

our $ldap_bind_dn =
'uid=Admin,ou=System,dc=centrino,dc=luxan,dc=ath,dc=cx';
our $ldap_bind_pw = 'salasana';
our $ldap_bind_host = 'ldap://ldap.puv.fi';

our $ad_bind_dn =
'CN=Administrator,CN=Users,DC=puv,DC=fi';
our $ad_bind_pw = 'salasana';
our $ad_bind_host = 'dcwolf.puv.fi';
our $ad_base = "CN=Users,DC=puv,DC=fi";


################################
my ($ldap,$ad) = connect_ldap();


if ($#ARGV < 0) {
   listDisabledUsers($ldap,$ad);
   exit(0);
}
elsif ($#ARGV < 1) {
   print "Anna palautuksen kestoaika päivinä.\n";
   exit(1);
}

my $user = getUserEntry($ldap,$ad,$ARGV[0]);
enableUser($ldap,$ad,$user,$ARGV[1]);
```

```
#########################################
#
sub round {
   my ($number) = @_ or return undef;
   return int($number);
}

sub connect_ldap {

   # connect ldap
   my $ldap = new Net::LDAP($ldap_bind_host);
   if (!$ldap) {
     die "ERROR: Cannot connect to ",$ldap_bind_host."\n";
   }
   my $result = $ldap->bind( $ldap_bind_dn,
password=>$ldap_bind_pw );
   if( $result->code )
   {
      die "Cannot bind: ",$result->error,"\n";
   }

   # connect ad
   my $ad = new Net::LDAP($ad_bind_host);
   if (!$ldap) {
     die "ERROR: Cannot connect to ",$ad_bind_host."\n";
   }
   my $ad_result = $ad->bind( $ad_bind_dn,
password=>$ad_bind_pw);
   if ( $result->code ) {
      die "Cannot bind to AD: ",$ad_result->error,"\n";
   }


   return ($ldap,$ad);
}

sub getUserEntry {
   my ($ldap,$ad,$uid) = @_;
   my $search_base = "uid=${uid},${inactive_ou}";
   my $search_filter = "objectClass=posixAccount";
   my $scope = "base";

   my $search = $ldap->search (
```

```
      base => $search_base,
      filter => $search_filter ,
      scope => $scope );

    if( $search->is_error )
    {
      die "Out of luck: ",$search->error, "\n";
    }

    if( $search->count < 1)
    {
      die "Käyttäjää ei löydy\n";
    }
    return $search->entry(0);
}

sub enableUser {
    my $ldap = shift;
    my $ad = shift;
    my $entry = shift;
    my $grace = shift;

    my $today = int(time());
    my $day = (60*60*24);

    my $uid = $entry->get_value('uid');
    my $dn = $entry->dn();
    my $newrdn = "uid=${uid}";
    print "Enabling user $dn\n";

    my $new_expire = int( $today+($day*$grace) );

    print "Grace: $grace days\nToday: $today\nexpire:
$new_expire\n";
    if ($entry->get_value('puvPersonAccountExpire')) {
        $ldap->modify($entry,  delete =>
['puvPersonAccountExpire']  );
    }
    $ldap->modify($entry,  delete =>
['puvPersonAccountExpireWarningSent']  );
    $ldap->modify($entry, add => { puvPersonAccountExpire =>
($new_expire) } );
    my $mesg = $ldap->moddn(    $dn,
                            newrdn => $newrdn,
```

```
                        newsuperior => $temp_ou,
                        deleteoldrdn => '1',
                        );
    $ldap->modify($entry, delete => { shadowExpire =>
($new_expire) } );

    $mesg->code && warn "Virhe: ", $mesg->error ;

    # aseta userAccountControl 2. bitti pois (esim. 66050 ->
66048)
    # active directory
    my $adsearch = $ad->search(
      base => $ad_base,
     filter =>  "(CN=$uid)",
     scope => 'sub'
    );

    if ($adsearch->is_error ) {
        die "AD search error: ",$adsearch->error,"\n";
    }
    for ($adsearch->entries) {
        my $ad_expire = $_->get_value('userAccountControl');
        $ad_expire = $ad_expire & ~0x2;
        $ad->modify($_,  replace => {'userAccountControl' =>
$ad_expire} );
    }
}

sub listDisabledUsers {
    my ($ldap,$ad) = @_;
    print "List of disabled users:\n";
    my $search = $ldap->search (
      base => $inactive_ou,
      filter => "(objectClass=posixAccount)" ,
      scope => "sub"
    );
    for ($search->entries) {
        print $_->get_value('uid')." (".$_->get_value('gecos').")\n";
    }
    return;
}
```

Kertakäyttötunnusten luonti- ja poistojärjestelmän lähdekoodi

```
#!/usr/bin/perl -w
#
# onauth.pl
#
# pam_script-moduulin ajama tiedosto, josta voidaan
kutsua
# haluttuja ohjelmia käyttäjien kirjautuessa.

use strict;
use User::grent;

my $make_log = 0;
my $logfile = ">>/tmp/logins";

my $aika = localtime;

if ( 1 == $make_log ) {
  open(LOGFILE, $logfile) or $make_log = 0;
}

doGroupLogin("TempUsers", "sudo /
etc/scripts/TempUsers_login.pl");

if( 1 == $make_log) {
  #print LOGFILE $aika . "-" . $ARGV[1] . "-" . $ARGV
[0] . "\n";
  close(LOGFILE);
}

# TÄRKEÄÄ: Jos pam_script on määritelty required tai
sufficient,
# skriptin tulee palauttaa arvo joka ei ole 0!
# Muuten autentikointi toimii millä tahansa
salasanalla.
exit(0);

############################################################
##############

# Suorittaa login-skriptejä ryhmäjäsenyyden perusteella
#
# Kutsuttava skripti saa argumentteina ryhmän nimen ja
```

```
tunnuksen
sub doGroupLogin {

 my $groupname = shift;
 my $script = shift;
 my $group = getgrnam($groupname);

 if ( ! $group ) {
  return;
 }

 for ( @{ $group->members } ) {
  if ( $ARGV[0] eq $_ ) {
    print $_ . " is a member of " . $groupname . "\n";
    my $command = $script . " " . $groupname . " " .
$ARGV[0];
    #print $command . "\n";
    system $command;
    if ( 1 == $make_log ) {
  print LOGFILE $aika . " - Executed '" . $script . "'
for user " . $ARGV[0] . "\n";
    }
  }
 }
}
```

```
#!/usr/bin/perl -w
#
# TempUsers_login.pl
#
# Tiedosto, joka ajetaan kertakäyttäjien
kirjautumishetkellä

use strict;
use PuvLdap;

if ( $#ARGV < 1 ) {
 exit(1);
}

my $today = time();
my $uid = $ARGV[1];
```

```
my $ldap           = pLdap_init();
my $ldap_user_base = "ou=TempUsers,dc=puv,dc=fi";

set_ldap_expire($ldap, $uid);

exit(0);

###################################
sub set_ldap_expire {

 my $ldap     = shift;
 my $username = shift;
 my $search = $ldap->search(
  base   => $ldap_user_base,
  filter => '(uid=' . $username . ')',
  scope  => "sub"
 );

 if ( $search->is_error or ($search->count == 0) ) {
  #die "Search error: ",$search->error, "\n";
  return 0;
 }

 my $ldap_entry = $search->entry(0);

 # tarkista LDAP-expiret
 my $ldapexpire = $ldap_entry->get_value
('puvPersonAccountExpire');

 if ( $ldapexpire && $ldapexpire < $today ) {
  return 1;
 }


 # paivita LDAP expire
 my $msg = $ldap->modify($ldap_entry,  replace =>
 {'puvPersonAccountExpire' => $today} );
 if (!$msg->is_error) {
  print "Käyttämäsi tunnus erääntyy huomenna.\n";
 }
 return 1;
}
```

```
#!/usr/bin/perl -w
```

```
#
# poista_tempusers.pl
#
# Huolehtii kertakäyttötunnusten poistamisesta
# Ajetaan kerran päivässä

use strict;
use PuvLdap;
use PuvAD;

my @ad_servers = ( 'ldaps://dcwolf.ad.puv.fi',
'ldaps://dcraas.ad.puv.fi' );

my $ad_deletor_dn =
'CN=sysLdapUseradmin,CN=Users,DC=ad,DC=puv,DC=fi';
my $ad_deletor_pw = 'cVU5VM6z3FAE8UwJjzQw';

my $ldap_group        = "ou=Groups,dc=puv,dc=fi";
my $onetime_unixgroup = "TempUsers";

my $onetime_base_ldap = "ou=TempUsers,dc=puv,dc=fi";
my $onetime_base_ad   =
"OU=TempUsers,DC=ad,DC=puv,DC=fi";

my $today = time();

my %deletelist;

my $ldap = pLdap_init();
my $ad = pAD_init();

# hae LDAP-otukset ensin (näistä saa tarvittavat tiedot
poistoa
# varten)
my %ldap_entries = get_ldap_users($ldap);
my %ad_entries = get_ad_users($ad);

# tarkista logonCount AD-servereiltä
for (@ad_servers) {
 # hae ad-tunnukset palvelimelta
 my $ad_connection = connect_ad($_);
 my %ad_entries = get_ad_users($ad_connection);
 for my $uid ( sort keys %ad_entries ) {
  if ( $ad_entries{$uid}->get_value('logonCount') ) {
   # AD-hakemistosta haetaan attribuutti 'logonCount',
```

```
josta
   # ilmenee onko tunnuksella kirjauduttu.
   if ( int( $ad_entries{$uid}->get_value('logonCount')
) > 0 ) {
    $deletelist{$uid} = 1;
   }
  }
 }
}

# tarkista shadowExpire LDAP-serveriltä
for my $uid (sort keys %ldap_entries) {
 if ( $ldap_entries{$uid}->get_value
('puvPersonAccountExpire') ) {
  if ( $ldap_entries{$uid}->get_value
('puvPersonAccountExpire') < $today ) {
  $deletelist{$uid} = 1;
  }
 }
}

for my $uid ( sort keys %deletelist ) {
 delete_user($ldap, $ad, $ldap_entries{$uid},
$ad_entries{$uid});
}

pLdap_kill($ldap);
pAD_kill($ad);
##########################################################
##########
sub connect_ad {

 my $hostname = shift;
 my $ad       = new Net::LDAP($hostname);
 my $ad_msg   = $ad->bind( $ad_deletor_dn, password =>
$ad_deletor_pw );

 if ( $ad_msg->code ) {
  die "Ei voitu yhdistaa " . $hostname . ": " .
$ad_msg->error . "\n";
 }
 return $ad;
}
```

```
sub delete_user {
 my $ldap = shift;
 my $ad = shift;
 my $ldap_entry = shift;
 my $ad_entry = shift;
 my $uid = $ldap_entry->get_value('uid');

 # halutaan poistaa vain kertakäyttäjät...

 if ( $uid =~ /^User\d{4}/ ) {
  my $homedir = $ldap_entry->get_value
('homeDirectory');
  #print "Deleting " . $uid . "\n";
  #print "Homedir: " . $homedir . "\n";
  #print "LDAP DN: " . $ldap_entry->dn() . "\n";
  #print "AD DN: " . $ad_entry->dn() . "\n\n";

  # poistetaan kotihakemisto
  my $command = "rm -rf " . $homedir;
  #print $command . "\n";
  my $ret = (system $command) >> 8;
  if ( 0 != $ret ) {
   die "Virhe käyttäjän kotihakemiston " . $homedir . "
poistossa!";
  }

  # poistetaan AD-objekti
  my $delmsg_ad = $ad->delete( $ad_entry );
  if ($delmsg_ad->is_error ) {
   die "Virhe käyttäjän poistossa (AD):" . $delmsg_ad-
>error;
  }

  # poistetaan LDAP-objekti
  my $delmsg_ldap = $ldap->delete( $ldap_entry );
  if ( $delmsg_ldap->is_error ) {
   die "Virhe käyttäjän poistossa (LDAP): " .
$delmsg_ldap->error;
  }

  print "Tunnus $uid poistettu.\n";
 }

}
```

```
sub get_ldap_users {
 my $ldap = shift;
 my %ldap_entries;

 # hae kayttajat LDAP:ista
 my $search = $ldap->search(
  base   => $onetime_base_ldap,
  filter => "uid=*",
  scope  => 'sub',
 );


 for ( $search->entries ) {
  $ldap_entries{ $_->get_value('uid') } = $_;
 }
 return %ldap_entries;
}

sub get_ad_users {

 my $ad = shift;

 # hae kayttajat AD:sta
 my $adsearch = $ad->search(
  base   => $onetime_base_ad,
  filter => "(CN=*)",
  scope  => 'sub'
 );

 #print "AD results: " . $adsearch->count . "\n";
 my %ad_entries;


 for ( $adsearch->entries ) {
  $ad_entries{ $_->get_value('CN') } = $_;
 }

 return %ad_entries;
}
```