OPINNÄYTETYÖ

ANTTI LUUSUA 2011

TIETOTURVALLISUUS PHP- JA MYSQL-POHJAISESSA WWW-SOVELLUKSESSA

Rovaniemen
ammattikorkeakoulu
University of Applied Sciences

TIETOJENKÄSITTELYN KOULUTUSOHJELMA

Rovaniemen
ammattikorkeakoulu
University of Applied Sciences

ROVANIEMEN AMMATTIKORKEAKOULU

LUONNONTIETEIDEN ALA

Tietojenkäsittelyn koulutusohjelma

Opinnäytetyö

TIETOTURVALLISUUS PHP- JA MYSQL-POHJAISESSA SOVELLUKSESSA

Antti Luusua

2011

Toimeksiantaja Kuusipekan sukuseura ry

Ohjaaja Aarre Jortikka

Hyväksytty _____ 2011 _____

Rovaniemen ammattikorkeakoulu
University of Applied Sciences

Luonnontieteiden ala
Tietojenkäsittelyn koulutusohjelma

Opinnäytetyön tiivistelmä

Tekijä	Antti Luusua	Vuosi	2011
Toimeksiantaja	Kuusipekan sukuseura ry		
Työn nimi	Tietoturvallisuus PHP- ja MySQL-pohjaisessa sovelluksessa		
Sivu- ja liitemäärä	27		

Opinnäytetyön tavoitteena oli luoda Kuusipekan sukuseura ry:lle tietokantapohjaiset verkkosivut ja jäsenrekisterisovellus tietoturvallisuus huomioon ottaen. Verkkosivuilta toivottiin erityisesti mahdollisuutta julkaista uutisia ja valokuvia suvun tapahtumista. Jäsenrekisterin tulisi sisältää perustiedot jokaisesta suvun jäsenestä ja toimia apuvälineenä jäsenmaksujen käsittelyssä.

Työkaluiksi sivujen rakentamiseen valitsin PHP-kielen ja MySQL-tietokannan. Opinnäytetyön tutkimusongelmana oli WWW-sovelluksen tietoturva PHP-kielen sekä MySQL-tietokannan kannalta. Työn tarkoitus on selittää, mikä tietokanta on, käydä läpi tärkeimpiä ja yleisimpiä tietoturvauhkia sekä kertoa suojautumismenetelmistä. Tietoturvasyistä työssä ei kuvata tarkasti Kuusipekan sukuseura ry:n verkkosivujen toteutusta.

Kuusipekan sukuseura ry on tyytyväinen verkkosivujen ulkoasuun ja helppokäyttöisyyteen. Verkkosivujen päivitysvastuu tullaan siirtämään seuran hallituksen jäsenille.

Avainsanat PHP, MySQL, tietokanta, tietoturva
Muita tietoja Työhön sisältyvä verkkosivu osoitteessa http://www.kuusipekka.net

Rovaniemen ammattikorkeakoulu
University of Applied Sciences

School of Business and Administration
Business Information Technology Programme

Abstract of Thesis

Author	Antti Luusua	Year	2011
Commissioned by	Kuusipekan sukuseura ry		
Subject of thesis	Web Application Security		
Number of pages	27		

The aim of the thesis was to create webpages and a web-based membership database application for Kuusipekan sukuseura ry considering the information security. Especially it was hoped for the webpages for the opportunity to publish news and photos of family events. The membership database application should include basic information on each family member and to be an effective tool in the handling of membership fees.

I chose PHP language and MySQL database to be the tools to build the site with. The research problem of this thesis was the security of web application regarding PHP and MySQL techniques. The purpose of the thesis is to explain what a database is, go through the most important and common security threats and tell about protection methods. For security reasons a thesis does not describe precisely how webpages of Kuusipekan sukuseura ry were made.

Kuusipekan sukuseura is pleased with the appearance of webpages and user-friendly interface. In the future the pages will be updated by the members of Kuusipekan sukuseura ry.

Key words PHP, MySQL, database, information security
Special remarks Thesis includes web site in address: http://www.kuusipekka.net

SISÄLTÖ

KUVIOLUETTELO

1 JOHDANTO

Aloittaessani opintoja Rovaniemen ammattikorkeakoulussa tiesin haluavani tehdä opinnäytetyönäni jotain ohjelmointiin liittyvää. Opintojen edetessä kiinnostuin erityisesti PHP-ohjelmointikielestä ja tietokannoista. Päädyin valitsemaan opinnäytetyöni aiheeksi WWW-tietokantasovelluksen, jonka toteuttaisin PHP-ohjelmointikielellä sekä MySQL-tietokannalla.

Varsinainen aihe oli pyörinyt mielessäni jo ensimmäisen opiskeluvuoden syksystä lähtien. Kuusipekan sukuseura ry tarvitsi omat verkkosivut sekä jäsenrekisterin, ja koin aiheen sopivaksi omaan opinnäytetyöhöni.

Työni tavoitteeksi asetin turvalliset ja käyttäjäystävälliset verkkosivut sekä jäsenrekisterin. Päädyin rakentamaan sivut sovelluksen alusta lähtien itse enkä lähtenyt tutkimaan vaihtoehtoja käyttää valmiita ratkaisuja, esimerkiksi julkaisujärjestelmiä. Vaikka julkaisujärjestelmien osaaminen on valtti työmarkkinoilla, halusin perehtyä projektin toteuttamiseen perusteista lähtien.

Toinen tavoittelemani asia on saada aikaiseksi helppokäyttöiset, turvalliset verkkosivut ja WWW-sovellus, jota voi myöhemmin käyttää esimerkiksi työn haussa todisteena osaamisesta. Tulen jatkamaan sivujen kehitystä jatkossakin.

Toteutin työni PHP-kielellä sekä MySQL-tietokannalla. Raporttini keskittyy tutkimaan tietoturvaa näiden kahden tekijän kannalta. Raportin keskipiste ei ole itse rakentamani tuote ja sen tietoturva, vaan tarkastelen WWW-sovelluksen tietoturvaa ja sen parantamista yleiseltä kannalta. Tähän ratkaisuun päädyin, koska en halua asettaa rakentamaani sivustoa ja sen turvallisuutta vaakalaudalle. Työn sisäisen ristiriidankin uhalla valitsin tietoturvan aiheekseni, koska uskon sen olevan osa-alue josta minulla on eniten opittavaa WWW-ohjelmoinnin parissa. Otsikon mukaisesti raporttini keskittyy tarkastelemaan WWW-sovelluksen tietoturvaa PHP-kielen sekä MySQL-tietokannan näkökulmasta, eikä muihin ohjelmointikieliin tai tekniikoihin oteta, pieniä sivuhuomautuksia lukuunottamatta, kantaa.

Raportista saa eniten irti lukija, jolla on perustietämystä WWW-ohjelmoimisesta, eritoten PHP-kielestä sekä MySQL-tietokannasta. Koska tutkittavana ongelmana on tietoturvallisuus, raportissa ei ohjata lukijaa esimerkiksi MySQL-tietokannan käyttöönotossa tai PHP-kielen perusteissa.

2 TOIMEKSIANTAJA

2.1 Kuusipekan sukuseura ry

Sodankylässä perustettiin 1992 Kuusipekan sukuseura. Seuran alkuperäinen tarkoitus oli edesauttaa Kuuselan suvun sukututkimusta ja järjestää kokoontumisia, joissa suvun jäsenet tutustuisivat toisiinsa. Kuusipekan suku jakautuu 12 eri sukuhaaraan. Vuonna 2010 seuraan kuului noin 1200 henkilöä. Jäsenistä valtaosa asuu Suomessa, mutta esimerkiksi Venäjällä, Saksassa sekä Ruotsissa asuu useita jäseniä. Kuusipekan sukuseura on rekisteröity yhdistys, jonka hallituksen muodostaa 12 henkilöä siten, että yksi jäsen edustaa kutakin sukuhaaraa.

Verkkosivujen ja jäsenrekisterin sisältö, ulkoasu ja käyttöliittymä suunniteltiin yhteistyössä hallituksen sihteerin kanssa ja hänen toiveidensa pohjalta. Ulkoasusta toivottiin hillitty, Sodankylän järvimaisemiin sopiva, ja käyttöliittymästä helppokäyttöinen.

2.2 WWW-sovellus ja verkkosivut

Kuusipekan sukuseura perii jäseniltään vuosittaista jäsenmaksua. Jäsenmäärän kasvaessa maksujen tarkkailu on käynyt työlääksi ja on langennut seuran sihteerin harteille. Tilannetta korjaamaan Kuusipekan sukuseuran hallitus tilasi WWW-sovelluksen, jonka avulla voisi paitsi tarkkailla maksuja, myös ylläpitää helppolukuista tietokantaa jäsenistä. Sovelluksen tulisi olla helppokäyttöinen ja tietoturvallinen.

Toinen seuralta puuttuva palvelu oli WWW-sivut. Sivuille haluttiin muun muassa mahdollisuus päivittää uutisia, hallituksen yhteystietoja sekä kuvia järjestetyistä tapahtumista.

Päätin toteuttaa sekä WWW-sovelluksen että WWW-sivut käyttäen PHP-kieltä ja MySQL-tietokantaa. Molemmat näistä ovat ilmaisia. Toinen syy valintaan oli aikaisempi kokemukseni sekä kiinnostukseni molempia kohtaan. Uskon, että näiden kahden työkalun osaamisesta on hyötyä myös työmarkkinoilla.

Kuusipekan sukuseura ry:n verkkosivut toimivat webhotellissa ja tästä

johtuen minulla ei henkilökohtaisesti ollut pääsyä kaikkiin MySQL-tietokannan sekä PHP-kielen asetuksiin. Esimerkiksi tietokannan päivämäärän muoto on oletuksena vuosi-kuukausi-päivä, kun suomalainen tapa olisi merkitä päivä-kuukausi-vuosi. Webhotellin asiakkaana kyseiseen asetukseen ei voi tehdä muutosta, joka taas aiheuttaa pientä lisävaivaa WWW-sovelluksen kehittämisessä. Osa raportissa läpi käytävistä asioista voivat vaatia ylemmän tason oikeuksia sekä MySQL-tietokannan että PHP-kielen osalta ja kaikkiin näihin asetuksiin ei ainakaan itse käyttämässäni webhotellissa ole pääsyä.

3 TIETOKANTA

3.1 Relaatiotietokannat

Tietokanta on moniselitteinen käsite. Yleisesti ottaen tietokanta on loogisesti yhteenkuuluvien, tallennettujen tietojen joukko, jota voidaan helposti käsitellä tietokantakielellä (kuten SQL). Tietokannassa olevia tietoja hallinnoi erityinen ohjelmisto, tietokannan hallintajärjestelmä eli TKHJ (Database Management System, DBMS). Tunnettuja esimerkkejä näistä ovat Oracle, DB2, Microsoft SQL Server, MySQL ja Access. (Hovi–Huotari–Lahdenmäki 2005, 4.)

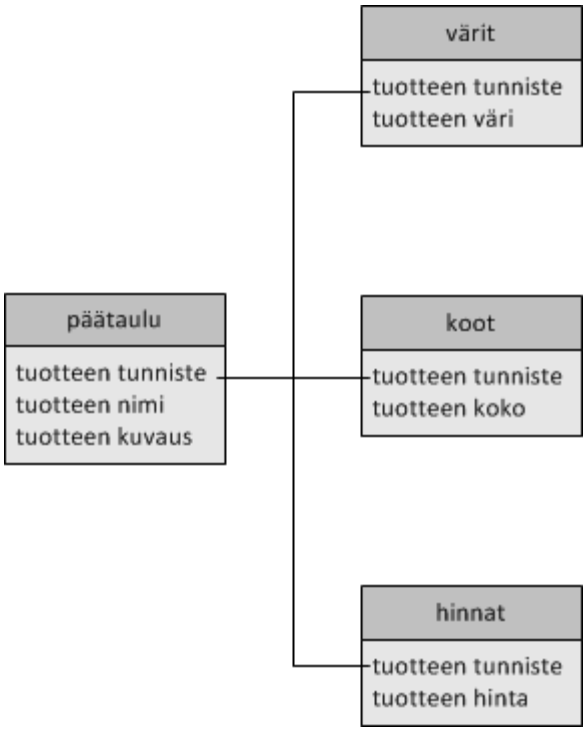
Mikä on relaatiotietokanta? Lyhyesti sanottuna relaatiotietokanta on joukko yhteen liitettyjä tauluja, jotka koostuvat sarakkeista ja riveistä. Nämä taulut ovat suhteessa toisiinsa tiettyyn sarakkeeseen merkittyjen arvojen pohjalta. (Meloni 2003, 8.)

Kun WWW-sovelluksia kehitetään oikeasti, yksi tavallisimmista tehtävistä on rakentaa tuoteluettelo verkkokauppaan – kauppa on ikään kuin sovellus, joka on tehty pienemmistä datapaloista (Meloni 2003, 8). Muita hyviä esimerkkejä WWW-sovelluksista ovat mm. erilaiset kuvapankit (Flickr), videotietokanta (YouTube, Vimeo) sekä sosiaaliset mediat (Facebook, Twitter), jotka säilövät monenlaista dataa ja koostuvat massiivisista tietokannoista.

Kuvittele, että teet tuoteluettelon urheiluvälineistä. Tuloksena voi esimerkiksi olla seuraavanlainen lista:

- Tuotteen tunniste
- Tuotteen nimi
- Tuotteen väri
- Tuotteen koko
- Tuotteen hinta
- Kuvaus tuotteesta

Näillä tiedoilla voisi luoda helposti yhden suuren taulukon johon tuotteet on listattu. Tästä taulukosta kasvaisi kuitenkin nopeasti todella valtava ja hankalasti luettava. Valtavan luettelon sijasta voisit tehdä useita pieniä, toisiinsa liittyviä tauluja. (Meloni 2003, 8.)



Kuvio 1. Taulujen väliset suhteet. (Meloni 2003, 10)

3.2 MySQL

MySQL on laajimmin käytetty avoimen lähdekoodin tietokanta (Learn Web Tutorials 2011). Sillä on useita miljoonia käyttäjiä. Joukossa on niin yksittäisiä henkilöitä, jotka käyttävät tietokantaa omien henkilökohtaisten verkkosivujensa ryydittämiseen, kuin isoja yrityksiäkin, joissa MySQL vauhdittaa laajalti käytössä olevia sivustoja. Esimerkki jälkimmäisestä on Yahoo! Finance, jossa MySQL:llä prosessoidaan satoja kyselyitä sekunnissa ja miljoonia kyselyitä päivässä. (Meloni 2003, 11.) Muita esimerkkejä isoista MySQL-tietokannoista ovat mm. Facebook, YouTube sekä Wikipedia (Learnwebtutorials 2011).

MySQL-tietokannan suosio perustuu pitkälti sen yhteensopivuuteen minkä tahansa ohjelmointikielen kanssa. Esimerkkejä näistä kielistä ovat PHP, Perl, C/C++, Java, Python ja Tcl. Toinen etu on helppo siirrettävyys käyttöjärjestelmien välillä. MySQL sisältää valmiit työkalut tietojen siirtämiseen eri käyttöjärjestelmien tai ohjelmointikielten välillä (Meloni 2003, 12.)

Suurin MySQL:n myyntivaltti kuitenkin on, että sen saa useimmiten ilmaiseksi. Se on ilmainen myös liiketoimintaan. MySQL on alun perin

ruotsalaisen MySQL AB:n kehittämä tuote, jonka Sun Microsystems osti vuonna 2008. Vuosi tästä eteenpäin ohjelmistojätti Oracle osti Sun Microsystemsin ja sai tällä kaupalla haltuunsa myös MySQL:n oikeudet. Oracle on kuitenkin toistaiseksi luvannut pitää MySQL:n avoimena ja ilmaisena kaikille käyttäjille. (Sharon Machlis 2009.) Viimeisenä etuna MySQL:ssä haluan todeta sen helpon käyttöönoton sekä laajan yhteisön tuen eri keskustelufoorumeiden kautta. Lisäksi useat webhotellit tarjoavat valmiiksi asennetut ja konfiguroidut MySQL-tietokannat.

4 PHP:N JA MYSQL:N TIETOTURVAUHKIA

4.1 Tietoturva verkkosovelluksissa

Tietoturva on WWW-sovelluksen tukipilari. Mikäli tukipilari ei tarpeeksi vakaa ja luotettava, kaikkein hienoimmatkin rakennelmat voivat sortua. Tästä johtuen se on myös mielenkiintoinen näkökulma WWW-sovelluksen kehittämisessä ja sitä myöten päätyi raporttini viitekehykseksi.

Internet on vihamielisin kehitysympäristö sovelluksille (Howard-LeBlanc 2004, 307). Väite on ymmärrettävä ja järkeenkäypä, kun ajattelee mahdollisten väärinkäyttäjien määrää. Mikäli sovellus ei toimi avoimessa verkossa, on siihen myös vaikeampi murtautua. WWW-sovellukset voivat tapauksesta riippuen olla käytössä joka puolella maailmaa, kenen tahansa päätteeltä. Sovelluksen tietoturvallisuuden varmentaminen tulisi olla suunnitteluvaiheessa tärkeässä roolissa.

Kuusipekan sukuseura ry:n WWW-sivujen tietoturvan suunnittelu oli haastavaa. Esimerkkinä kerron WWW-sivuilla olevasta uutisten lisäystoiminnosta. Kaikki uutiset tallennetaan tietokantaan ja myöhemmin haetaan sieltä. Uutisten lisäys sallitaan vain muutamalle henkilölle jotka kirjautuvat WWW-sivuille omilla tunnuksillaan. Varsinainen ongelma oli uutisten hakeminen ja näyttäminen ei-kirjautuneille henkilöille, WWW-sivun vierailijoille. Miten voidaan sallia pääsy haavoittuvaan tietokantaan ilman mitään selvitystä käyttäjän henkilöllisyydestä? Tässä vaiheessa raportin sisäinen ristiriita ilmiintyy – vaikka raporttini aiheenvalinta liittyy tietoturvallisuuten, juurikin tietoturvasyistä en voi omaa ratkaisuani ongelmaan tässä raportissa esittää.

Mitä tietoturva sitten tarkoittaa? Kysymys on toimenpiteistä, jotka takaavat yhtiön tai yksilön tietojen koskemattomuuden. Tietoturvalle on asetettu tiettyjä tavoitteita; niitä ovat yksilön tai organisaation annettujen tietojen luottamuksellisuus, eheys, kiistämättömyys, pääsynvalvonta, saatavuus ja tarkastettavuus. Jotta tiedot ovat luottamuksellisia, niiden on oltava vain niihin oikeutettujen käytössä. (Suomen Internetopas 2011.) Riittävä tietoturva tavoiteltavana tasona on mielestäni sopiva, koska täydellisen murtovarmaa

sovellusta, oli se sitten WWW-sovellus tai jokin muu, ei pystytä rakentamaan.

Tietokantasovelluksen suunnittelussa tärkeä vaihe on miettiä tarkasti mitä tietoa halutaan säilyttää, kuka sovellusta käyttää ja mihin tarkoitukseen sekä mitä sovelluksen halutaan tekevän. WWW-sovellusta suunniteltaessa tulee miettiä mitä tietoa ja ominaisuuksia sovellukseen halutaan. Mitä laajempi sovellus, sitä enemmän tietoturva-aukkoja.
Toinen tärkeä asia on ottaa huomioon sovelluksen käyttämien tekniikoiden ajantasaisuus. Vanhentunut tekniikka mahdollistaa usein uusia tietoturva-aukkoja. CERT-FI verkkosivusto kertoo Älypää-pelipalvelun tietovuototapauksesta seuraavasti:

”Maaliskuun 2010 lopulla tuli ilmi tapaus, jossa alypaa.com- pelisivustolta oli onnistuttu varastamaan käyttäjätunnukset, salasanat ja palveluun rekisteröityessä käytetyt sähköpostiosoitteet.

Yli 125 000 käyttäjän tiedot sisältävä tiedosto oli saatavilla useiden tiedostonjakopalvelujen kautta ja tieto siitä levisi nopeasti internetissä.

CERT-FI:n tietojen mukaan Älypää-palvelun salasanojen urkkimisessa on voitu käyttää hyväksi palvelun vanhentunutta ja tietoturvallisuudeltaan puutteellista toteutusta. Salasanat ovat ilmeisesti olleet selväkielisinä palvelun taustajärjestelmän tietokannassa. Tietokantaan on voitu päästä käsiksi ns. SQL-injektio tyyppisen puutteelliseen syötteentarkistukseen perustuvan haavoittuvuuden kautta.” (CERT-FI 2010a.) Tuntematta tapauksen taustoja, voidaan todeta, että ohjelmistojen päivitys sekä panostus turvallisuuteen tietokannan suunnittelussa olisi voinut estää tietovuodon.

Raporttini seuraavat osat keskittyvät kertomaan yleisimmistä tietoturva-aukoista ja uhista, siitä miten niiltä voidaan suojautua ja omista kokemuksistani asian tiimoilta. En kuitenkaan paljasta suoraan omia ratkaisujani, vaikka ne pohjaavatkin vahvasti tulossa oleviin kappaleisiin.

4.2 MySQL-tietokannan salaaminen

Oletuksena tietokantaan tallennetaan kaikki tiedot selkokielisenä, luettavassa muodossa. Mikäli tietokanta sisältää salaista tietoa, esimerkiksi salasanoja, käyttäjätunnuksia, henkilötunnuksia tai mitä tahansa muuta tietoa mitä ei haluta ulkopuolisille näyttää, on suositeltavaa salata tietokannan sisältö.

Tietojen salaus suojelee paitsi murtautujilta, myös tietokannan lukuoikeuden omaavilta ylläpitäjiltä.

Tietokannan salaus perustuu vanhaan keksintöön, salakirjoitukseen. Salakirjoituksen tavoite on muuttaa teksti tai viesti sellaiseen muotoon, että ainoastaan vastaanottaja pystyy sen lukemaan. Vastaanottaja tarvitsee viestin luettavaan muotoon palauttamiseen tiedon, jota kutsutaan salakirjoitusavaimeksi. Mikäli salakirjoitettu viesti pystytään muuntamaan luettavaan muotoon ilman tuota kyseistä avainta, on salakirjoitus murrettu. (Ruohonen 2002, 264.)

Tietokannassa olevan tiedon salaamiseen käytetään Hash-funktioita. Näitä kutsutaan yksisuuntaisiksi funktioiksi, jotka luovat viestistä tiivisteen. Viesti on funktion lähtöarvo, ja tiiviste on funktion vastaus. Viestin jokainen merkki vaikuttaa funktion lopputulokseen. Toisin sanoen yhtä merkkiä muuttamalla tiiviste muuttuu ratkaisevasti erilaiseksi. (Ruohonen 2002, 264 - 265.)

MD5 eli Message Digest 5 on suosittu hash-funktio. Tämä kyseinen funktio tuottaa viestistä 128-bittiä eli 16 tavua pitkän tiivisteen. Tiiviste on viestistä riippumatta aina saman mittainen merkeissäkin mitattuna. Tästä johtuen on mahdoton päätellä viestin sisältöä pelkästään tiivisteen pituutta tarkastelemalla.

SHA eli Secure Hash Algorithm on MD5-funktiota uudempi. Sen toimintaperiaate on samankaltainen kuin MD5-funktiossa, mutta SHA tuottaa suuremman tiivisteen. (Ruohonen 2002, 265.) SHA funktiosta on olemassa useita eri versioita, seuraavassa kuvattuna SHA-1 funktion ja MD5-funktion tuottamat tiivisteet.

Viesti	Tiiviste (MD5)	Tiiviste (SHA-1)
Salakirjoitus	42ffe2d5a22ae6f8b0cce489b5051bbc	46962b1705bf5ea9423e59d3b849f6b1abe95c12
salakirjoitus	2837950bdb0c873169644591ffdb3dce	9f2fe27c43e09da269226e8391068cdad4ad0417
Useampi kuin yksi sana	b37a069e31c848e7a42a82702855aea2	23666901eac2359e54bc5eeffbe0cead1e31f9a7

Kuvio 2. Hash-funktioiden tiivisteitä

4.3 Käyttäjätunnukset ja oikeudet

WWW-sovelluksen tietoturvan kannalta on hyvä pitää mielessä seuraava seikka: käyttäjien oikeuksissa vähemmän on parempi. Liian laajat

käyttäjäoikeudet voivat aiheuttaa, joko tahallisesti tai tahattomasti, suurta vahinkoa. (Howard-LeBlanc 2004, 119.) Hyvä ensimmäinen vaihe on kytkeä käyttäjiltä kaikki oikeudet pois ja lisätä niitä yksi kerrallaan tarpeen näin vaatiessa. Varmuuden vuoksi tulisi kuitenkin säilyttää yksi tai useampi laajaoikeuksinen käyttäjätili. (Kofler 2005, 265.) Oletuksena MySQL-tietokanta luo valmiita käyttäjätunnuksia, mukaan lukien anonyymeille käyttäjille tarkoitetun tunnuksen ilman salasanaa. On suositeltavaa tietoturvan kannalta poistaa tämä tili kokonaan. (MySQL 2011.)

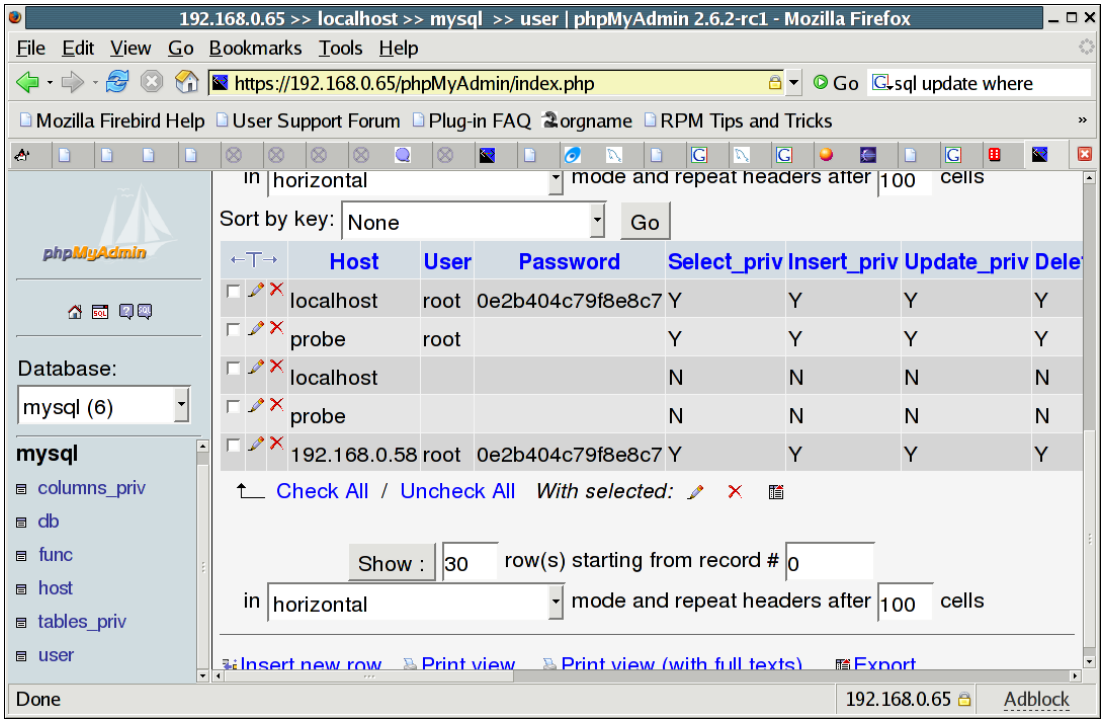
Mielestäni MySQL tarjoaa helppokäyttöisen ja laajan tavan hallita käyttäjätunnuksia ja niiden oikeuksia. Oikeuksia voidaan asettaa jopa yksittäisille taulun sarakkeille tai riveille. Lähtökohtana tietokannan oikeudet tulisi asettaa siten, ettei kaikilla ole oikeuksia tutkia dataa, eikä todellakaan hallita sitä poistamalla ja / tai lisäämällä tietueita (Kofler 2005, 263).

MySQL:n käyttäjäntunnistus toimii kahdella tasolla. Ensimmäisellä tasolla tarkistetaan onko käyttäjällä oikeuksia kommunikoida MySQL-tietokannan kanssa lainkaan. Oikeuksien puuttuessa yhteys katkaistaan eikä mitään dataa tuoda käyttäjän silmäiltäväksi. Mikäli kyseisellä käyttäjätunnuksella on oikeudet kommunikoida MySQL-tietokannan kanssa, tunnistautumisprosessi etenee toiselle tasolle. Tällä tasolla määritetään mitä oikeuksia (esimerkiksi poisto ja lisäysoikeudet) käyttäjällä on mihinkin tietokantaan ja sen tauluihin, riveihin ja sarakkeisiin. (Kofler 2005, 265.)

Käyttäjäoikeuksia on mahdollista asettaa useilla eri tavoilla:

- Graafinen käyttöliittymä, esimerkiksi MySQL Administrator- tai phpMyAdmin-ohjelma (katso kuvio 3)
- Muutoksien tekeminen suoraan MySQL-tauluun UPDATE ja INSERT komennoilla
- SQL komennoilla GRANT ja REVOKE

Edellä mainitut kolme tapaa ovat kaikki omalla tavallaan hyvin yksinkertaisia, mutta kaikki vaativat tietoa eritasoisista oikeuksista. (Kofler 2005, 265.)



Kuvio 3. mysql.user-taulu phpMyAdmin-ohjelmassa

Toisin kuin esimerkiksi käyttöjärjestelmien käyttäjätunnukset, jotka koostuvat käyttäjänimestä sekä salasanasta, MySQL-tietokanta lisää kolmannen tekijän tunnukseen, isäntäkoneen nimen. Tällä tavoin voidaan määrätä kuka ja mistä kyseinen henkilö voi yhdistää tietokantaan. (Kofler 2005, 265–266.) Seuraavassa selitetään nämä kolme yhdistävää tekijää, joista käyttäjätunnus muodostuu.

Käyttäjänimi on yksilöllinen tunniste, jolla jokainen henkilö tunnetaan MySQL:n parissa. Nimi voi olla 16 merkkiä pitkä ja se on merkkikokoriippuvainen. Nimen ei ole pakko muodostua pelkistä ASCII-merkeistä, mutta tämä on kuitenkin suositeltavaa koska eri käyttöjärjestelmät tulkitsevat erikoismerkit eri tavoin, joka voi johtaa ongelmiin. (Kofler 2005, 265.)

Salasana turvaa käyttäjätunnuksen. Käyttäjänimen tavoin salasanat ovat myös merkkikokoriippuvaisia, mutta niiden pituutta ei määrätä. Salasana tallennetaan kryptatussa muodossa MySQL-tietokannan käyttäjätauluun (user). Vaikka tätä kyseistä taulua voidaan tutkia, siitä ei voida suoraan päätellä käyttäjän salasanaa kryptauksen eli salauksen vuoksi. (Kofler 2005, 265.) Mielestäni ei ole järkevää käyttää samaa salasanaa esimerkiksi käyttöjärjestelmään sekä MySQL-tietokantaan, koska se lisää vahinkojen

suuruutta tietomurron tapahtuessa. MySQL-tietokannan salasanoja joudutaan joskus käyttämään niiden selkokielisessä muodossa, esimerkiksi koodin seassa (Kofler 2005, 264). Tällöin salasana voi olla helppo varastaa sovelluksen lähdekoodia tarkastelemalla.

Isäntänimi (engl. host name) on määritys, joka kertoo koneen nimen, jolta käyttäjä ottaa yhteyden tietokantaan. Tämän määrityksen avulla voidaan asettaa suoja, ettei tietty käyttäjätunnus voi yhdistää kuin esimerkiksi yhdestä IP-osoitteesta. Isäntänimien käyttö voi kuitenkin aiheuttaa ongelmia, sillä palvelimen mihin MySQL-tietokanta on asennettu, täytyy pystyä näkemään kone joka on asetettu isännäksi.

Alla olevassa kuviossa on listattu eri oikeudet ja niiden sallimat toiminnot.

Oikeus	Toiminta, jonka oikeus sallii
ALTER	Taulujen rakenteen muokkaus
CREATE	Uuden tietokannan, taulun tai indeksin luontioikeus
DELETE	Rivien poistaminen taulusta
DROP	Taulujen tai tietokantojen poistaminen
INDEX	Indeksien luonti ja poisto
INSERT	Rivien lisääminen tauluihin
SELECT	Tietojen valitseminen taulusta
UPDATE	Taulujen tietojen päivittäminen

Kuvio 4. MySQL-tietokannan oikeudet ja niiden sallimat toiminnot

4.4 Käyttäjien syötteet

Eräs yleisimmistä, ellei yleisin, PHP tietoturva-aukko on käyttäjien syötteiden tarkastamatta jättäminen. Käyttäjien syöttämään dataan ei yksinkertaisesti voi luottaa. Sinun tulee olettaa jokaisen WWW-sovelluksesi käyttäjän olevan vahingontekijä, koska on varmaa että osa heistä on. (Dickinson 2005.) PHP-kielellä kirjoitettujen WWW-sovelluksien tietoturvariskit eivät ole lähtöisin

ohjelmointikielestä itsestään, vaan siitä, ettei ohjelmia lähdetä aina suunnittelemaan tietoturvan kannalta. Sovellusta suunnitellessa tulisi kiinnittää erityistä huomiota paikkoihin joissa käyttäjät voivat kommunikoida sovelluksen kanssa ja itse syöttää dataa. (PHP Manual 2010a.)

4.4.1 SQL-injektio

Eräs helpoimmista reiteistä sovellukseen ja tietoihin hyökkääjän kannalta ovat vapaamuotoiset tekstikentät. Näihin kenttiin voi halutessaan syöttää millaista tietoa tahansa, ja tarkistamattomana ne aiheuttavat huikean tietoturva-aukon. Tästä syystä käyttäjän syöttämä data tulisi aina tarkistaa eli validoida ennen varsinaisten operaatioiden suorittamista tämän datan perusteella. Tällaisia vahingollisia syötteitä kutsutaan SQL-injektioiksi. (PHP Manual 2010b.)

SQL-injektio (engl. SQL injection) on tekniikka tietoturva-aukkojen hyödyntämiseksi järjestelmiin tunkeutumisessa. Niitä esiintyy tietokantapohjaisissa sovelluksissa, joissa käyttäjät käyttävät tietokantaa WWW-rajapinnan yli, mutta SQL-injektiot eivät sinällään ole WWW-sidonnaisia. (PHP Manual 2010b.) Toisin sanoen niitä voi esiintyä missä tahansa muussakin sovelluksessa. Kuvio 5 on esimerkki injektiosta komentorivin kautta, suoraan MySQL-tietokantaan.

SQL-injektiossa hyökkääjä antaa tietokantapalvelimelle SQL-komentoja, joita hänen ei pitäisi pystyä antamaan. Tämä hyökkäys tapahtuu useimmiten puuttuvan tai väärin toteutetun syöttötiedon tarkistuksen kautta ja joissain tapauksissa myös itse tietokantarajapinnassa tapahtuvan tiedon väärästä käsittelystä. SQL-injektioiden torjumiseksi kaiken käyttäjältä tulevan tiedon oikeellisuus pitää tarkistaa: merkkijonojen SQL-erikoismerkit, kuten lainausmerkki, pitää merkitä erikoismerkeiksi (useimmiten tietokantarajapinnassa valmiiksi löytyvällä quote-operaatiolla) ja muiden tietotyyppien kohdalla pitää varmistaa, että ne ovat oikeassa muodossa (esimerkiksi numeroarvo on todella numeroarvo). (PHP Manual 2010b.)

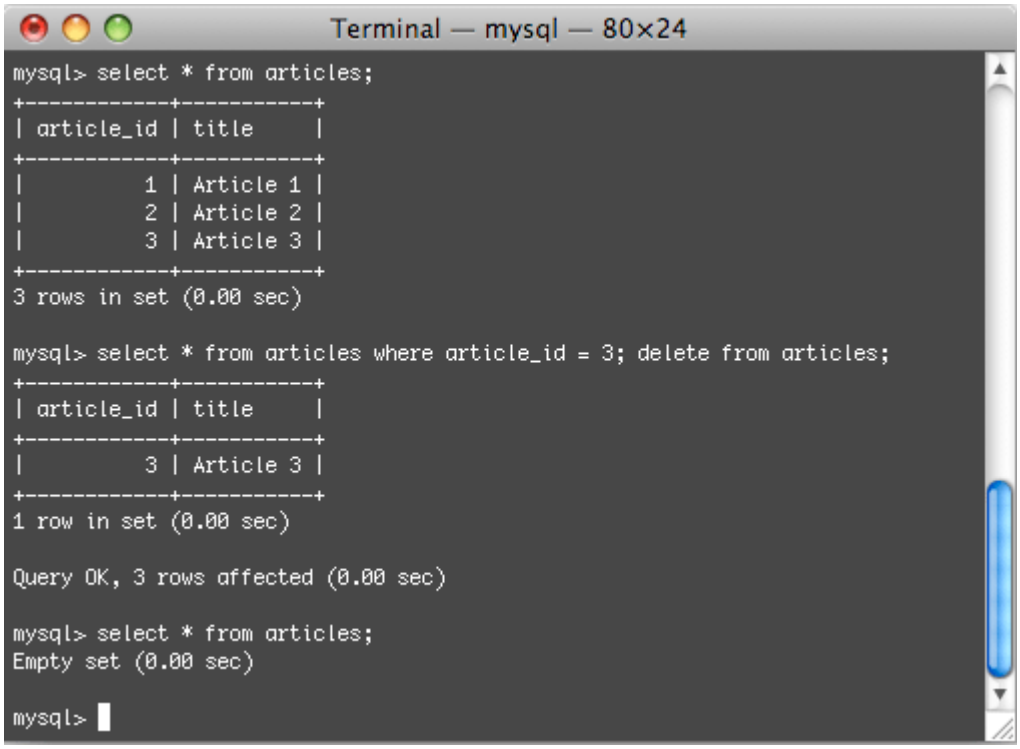
Otetaan esimerkki SQL-injektiosta. Oletetaan sovelluksessa olevan hakukenttä, jolla voidaan hakea henkilöitä sukunimen perusteella, jossa SQL-lause on seuraavanlainen:

```
”SELECT * FROM kayttajat WHERE sukunimi = ’$sukunimi’”;
```

Yllä oleva komento hakee kaikki ne tiedot kayttajat-nimisestä taulusta, jossa sukunimi-kenttä vastaa käyttäjän asettamaa $sukunimi PHP-muuttujaa. Mikäli käyttäjän syötettä ei ole tarkistettu ennen haun suorittamista, syöte voi olla mitä tahansa. Mikäli käyttäjän syöte on "'; DROP TABLE KAYTTAJAT "'; muodostuu tällöin palvelimelle lähtevästä SQL-lauseesta hieman erilainen.

”SELECT * FROM kayttajat WHERE sukunimi = ’ ’; DROP TABLE kayttajat;”

Vaikka alun perin haluttiin suorittaa ainoastaan hakukysely, jossa sukunimen perusteella tuodaan tietoja tietokannasta, vahingollinen kysely suorittaa tyhjän kyselyn, jonka jälkeen kayttajat-taulu tuhotaan. Onneksi on olemassa useita eri tapoja estää edellä mainittujen kaltaisten SQL-injektiohyökkäysten tapahtuminen.



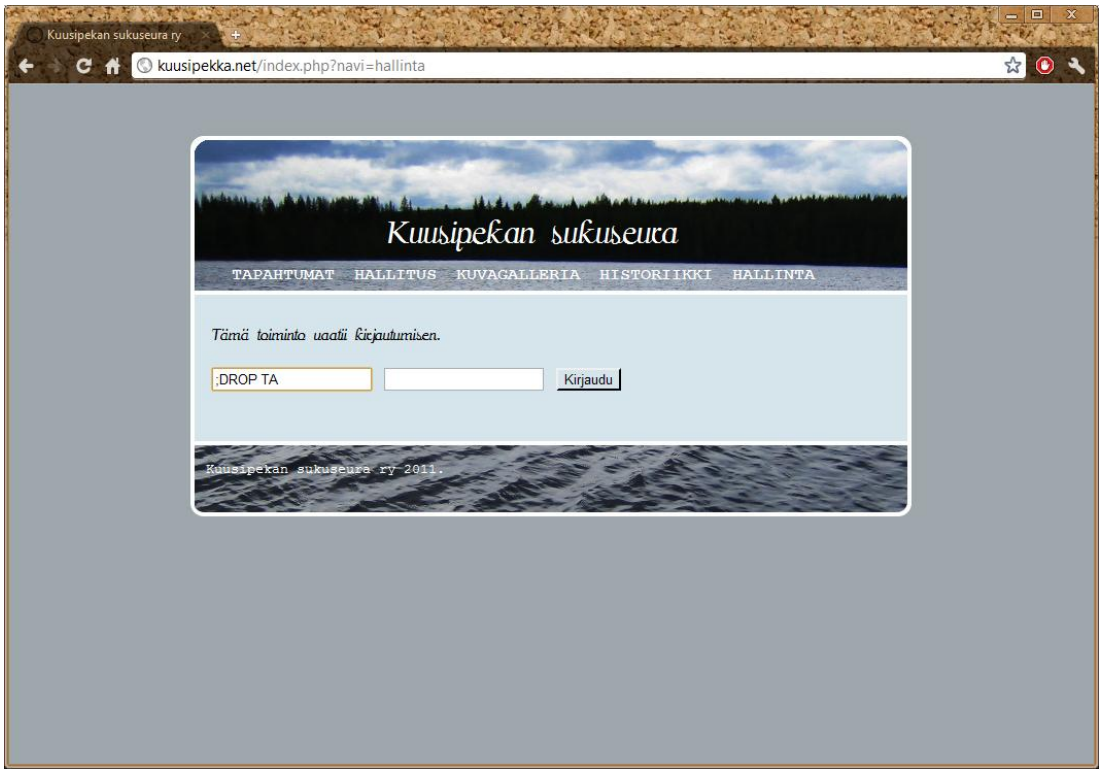
Kuvio 5. SQL-injektio komentorivin kautta (Of Zen and Computing 2008)

Käyttäjien haitalliset syötteet tulevat useimmiten tekstikenttien kautta, joten ensimmäinen askel kohti turvallisempaa sovellusta on rajata kenttiin syötettyjen merkkijonojen pituutta (Learn PHP Online 2009). Merkkijonojen pituuden säätely ei kuitenkaan ole kovinkaan turvallinen tai murtamaton ratkaisu, etevimmät murtautujat voivat luoda oman lomakkeen ja tämän avulla päästä käsiksi tietokantaan. Syötteiden pituuden säätelyn lisäksi kaikki

sovellukseen saapuvat syötteet tulisi tarkastaa.

Joten miten tulisi tarkastaa lomakkeelta lähetetyt syötteet? Kaikki käyttäjän muuttujat tulisi tarkastaa PHP-kielen funktiolla htmlspecialchars(). Tämä kyseinen funktio korvaa käyttäjän mahdollisesti syöttämät HTML-erikoismerkit kuten < ja > niiden HTML turvallisilla versioilla. (My PHP Form 2009.) Turvalliset HTML merkinnät eivät anna käyttäjän ajaa omia komentojaan. Tämän kaltainen suojaus estää itsessään hyökkääjää lisäämästä lomakkeen kautta sovellukseen erillisiä JavaScript tai HTML skriptejä. Ilman tarkastusta etevä käyttäjä voi ajaa monimutkaisia skriptejä tai jopa kokonaisia sovelluksia lomakkeen kautta ja täten saada haltuunsa sovelluksen.

Kuusipekan sukuseura ry:n verkkosivuille sisäänkirjautuessa on rajoitettu käyttäjänimen maksimipituutta kahdeksaan merkkiin. Sama rajoitus on voimassa kentässä, johon kirjautumisnimi kirjoitetaan.



Kuvio 6. Käyttäjänimen maksimipituus on 8 merkkiä.

PHP:ssä on olemassa erityisesti SQL-injektiota vastaan suojaava funktio nimeltään mysql_real_escape_string. Tämä funktio vastaanottaa merkkijonon jota aiotaan käyttää MySQL- kyselyssä ja palauttaa saman merkkijonon ilman

uhkaa SQL injektiosta. Funktio yksinkertaisesti korvaa haitalliset lainausmerkit (’) jotka käyttäjä on syöttänyt ja asettaa niiden tilalle turvalliset, käsitellyt lainausmerkit. (Tizag 2010.) Käsitelty syöte on huomattavasti turvallisempi kuin raakana lähetetty.

Edellä mainitut PHP-funktiot sekä lomakkeiden kenttien pituuksien säätely auttavat turvaamaan sovellusta haitallisilta käyttäjiltä ja heidän syötteiltään.

4.4.2 Cross Site Scripting

Cross-Site Scripting-hyökkäykset ovat injektiotyyppisiä ongelmia, joissa haitallista koodia syötetään muutoin hyvätahtoiselle ja luotetulle websivustolle. Cross-site scripting (XSS) hyökkäykset tapahtuvat hyökkääjän käyttäessä websovellusta haitallisen koodin lähettämiseen toiselle käyttäjälle. Viat, jotka mahdollistavat tämänkaltaisten hyökkäyksien toiminnan, ovat melko laajalle levinneitä ja vikoja löytyy usein paikoista joissa WWW-sovellus lähettää käyttäjän antaman syötteen eteenpäin ilman validointia. (Oswap 2010.)

XSS-hyökkäykset eivät ole riesana ainoastaan pienten kehittäjien sovelluksissa. CERT-FI kertoo haavoittuvuustiedotteessaan 076/2010 Microsoft Sharepointin kärsivän XSS-tyyppisestä haavoittuvuudesta. Haavoittuvuuden avulla hyökkääjä voi suorittaa kohdejärjestelmässä esimerkiksi Javascriptiä järjestelmään kirjautuneen käyttäjän oikeustasolla. Haavoittuvuutta voidaan hyväksikäyttää lähettämällä sähköpostia käyttäjälle, joka on kirjautuneena Sharepoint-järjestelmään ja houkuttelemalla käyttäjä avaamaan hyökkääjän muotoilema linkki. (CERT-FI 2010b.)

On siis todettava XSS-hyökkäysten olevan uhka myös isommille ohjelmistoille ja sovelluksille. Yksityisten ja pienempien sovellusten kehittäjien keskuudessa suositussa phpMyAdmin-ohjelmistossa on myös havaittu XSS-haavoittuvuus joulukuussa 2010. Tietokantahaun puutteellisesta syötteentarkistuksesta johtuva phpMyAdmin-ohjelmiston haavoittuvuus voi mahdollistaa Cross-Site scripting -tyyppisen hyökkäyksen, jolloin hyökkääjän HTML-koodin tai skriptien suorittaminen käyttäjän selaimessa on mahdollista. (CERT-FI 2010c.)

Cross-Site scripting hyökkäyksiltä suojautuminen onnistuu parhaiten rajaamalla käyttäjien syötteiden sisältöä. Täytyy sulkea pois mahdollisuus

syöttää Javascriptiä mihinkään osaan sovellusta. Sivuston ja WWW-palvelimen tulisi aina tarkistaa huolellisesti kaikki syötteet. Tämä koskee sekä palvelimelle syötettäviä URL-osoitteita, että sivujen kautta syötettäviä tietoja. Pelkästään tiettyjen merkkien tai merkkiyhdistelmien suodattamista ei voi pitää riittävänä, sillä tällaisen suodatuksen voi useimmiten kiertää.

Myös palvelimen käyttäjälle tulostamien sivujen oikeellisuus olisi syytä tarkistaa. Sivujen tulisi olla ehjää HTML- tai XML-sisältöä ja sivuilla käytettävän merkistökoodauksen mukaista. (CERT-FI 2008.)

4.5 Tiedostojen turvallinen käsittely

PHP-kieli tekee tiedostojen lataamisen palvelimelle ja sitä kautta sovelluksen saataville helpoksi. Tiedosto voi olla minkä tyyppinen tahansa, ja maksimikokoa voi säätää mielensä mukaan. (Web Cheat Sheet 2009.)

Yksinkertaisimmillaan tiedostojen lataaminen palvelimelle käy HTML-omakkeen ja muutaman PHP-komennon avulla. HTML-lomake vastaanottaa käyttäjän antamat tiedot, jonka jälkeen PHP suorittaa taustalla tiedoston siirtämisen palvelimelle. (Calin 2009.)

Alla olevassa esimerkkilomakkeessa tiedoston koko rajoitetaan 100 kilobittiin (100 000 tavua) käyttäen maksimiKoko- nimistä piilokenttää.

```
<form action=”siirto.php” method=”post”>
<input type=”hidden” name=”maksimiKoko” value=”100000” />
Valitse tiedosto minkä haluat ladata:
    <input type=”file” name=”tiedosto” />
    <input type=”submit” value=”Siirrä tiedosto” />
</form>
```

Toinen yleinen virhe tiedostojen latauslomakkeita luotaessa on jättää tiedoston tyyppi tarkistamatta (Calin 2009.) Hyvä tapa suojata WWW-sovellusta ja -palvelinta on rajoittaa lisättävien tiedostojen tyyppi vain niihin, mitkä voivat olla harmittomia tiedostotyyppejä. Jos käyttäjien halutaan pystyvän lisäämään palvelimelle kuvatiedostoja, tulisi tiedostojen päätteet rajata muutamaan yleisimpään muotoon, esimerkiksi jpg, png ja gif. (Hungred 2009.) Tiedoston päätteen rajoittaminen PHP-kielellä tapahtuu esimerkiksi

listaamalla sallitut tiedostopäätteet taulukkoon ja käymällä siirtohetkellä läpi, täsmääkö käyttäjän lähettämän tiedoston pääte sallittujen päätteiden listaan.

Kirjoitushetkellä Kuusipekan sukuseura ry:n verkkosivuilla ei ole vielä mahdollisuutta siirtää palvelimelle mitään tiedostoja. Tulevaisuudessa on kuitenkin tarkoitus rakentaa kuvapankki, johon vanhat valokuvat voidaan digitaalisessa muodossa siirtää. Kuvapankki tulee hyväksymään ainoastaan JPG- sekä PNG-päätteiset tiedostot. Nämä molemmat ovat kuvatiedostoja. Kuvien kokoa tullaan myös rajoittamaan, noin yhteen megatavuun per kuva. Tämä johtuu enimmäkseen webhotellin tarjoamasta kovalevykapasiteetista. Kuvapankki ei tule olemaan kaikille avoin, vaan se käyttää samoja tunnuksia kuin muukin osa sivuista.

4.6 Istunnon salaaminen

Sovelluksen tietoturvan parantamiseksi käyttäjän sekä palvelimen välille voi luoda salatun yhteyden. Salatun yhteyden tunnistaa selaimen osoiteriviltä, kun osoite alkaa ”https://”. Tässä kappaleessa kerron istunnoista sekä niiden salaamisesta SSL-protokollan avulla.

Verkkosivujen perusolemukseen kuuluu kommunikoinnin puute selaimen ja palvelimen välillä. Jokainen selaimen tekemä pyyntö palvelimelle käsitellään yksilönä. Vanhoja pyyntöjä ei tallenneta muistiin. Toisin sanoen tällaisten verkkosivujen käyttäminen ei ole interaktiivista. Sivustoihin jotka vain tarjoavat informaatiota tai esimerkiksi dokumentteja ilman hakutoimintoja, interaktiivisuuden puute sopii (Lane–Williams 2004, 338.)

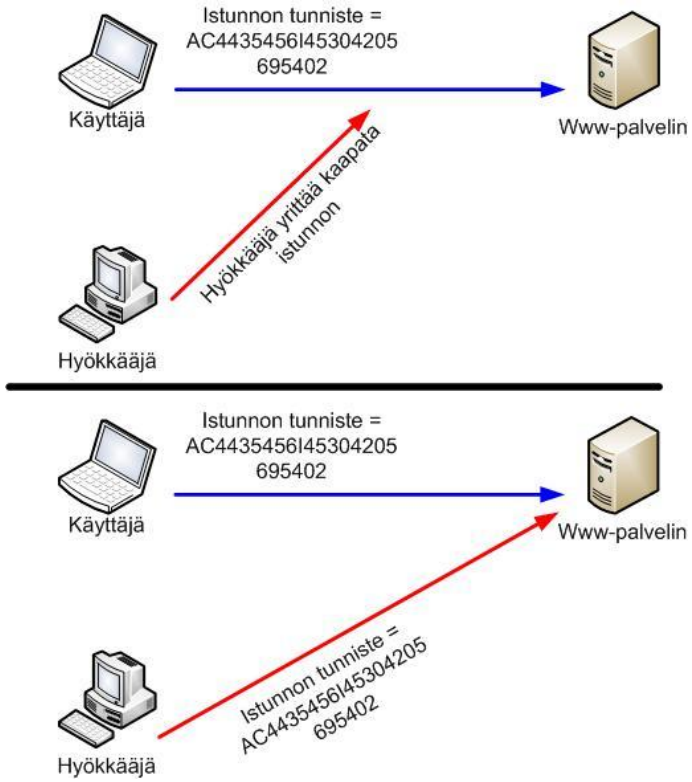
Verkkosivuissa tai sovelluksissa, joissa tarvitaan monimutkaista vuorovaikutusta käyttäjän kanssa, ei voida käyttää yksittäisiä, toisistaan irrallaan olevia sivuja. Tämänkaltaisesta sovelluksesta esimerkkinä toimii verkkokauppa. Kaupan sivuilla voidaan surffata ja etsiä tuotteita, lisätä tuotteita ostoskoriin ja jatkaa etsimistä. Valitut tuotteet säilytetään ostoskorissa verkkokaupan käytön ajan. Ostoskorin sisällön verkkosivu muistaa istuntomuuttujan avulla (Lane–Williams 2004, 338).

Käyttäjäntunnistus verkkosivuilla tai WWW-sovelluksessa voidaan toteuttaa

istunnoilla. Pääpiirteissään tunnistus etenee seuraavasti:

- käyttäjä syöttää tunnuksen ja salasanan lomakkeen kautta
- syötetyt tiedot käsitellään ja varmistetaan PHP koodissa
- tiedot tallennetaan istuntomuuttujaan, jossa ne ovat käytettävissä koko selailun ajan
- istuntomuuttujan tiedot tyhjennetään tai hävitetään joko istunnon maksimiajan umpeutuessa tai käyttäjän tyhjentäessä muuttujan itse (Lane–Williams 2004, 338.)

Käyttäjien tietojen tallentamisessa istuntoon on heikkoutensa. Käyttäjätunnusta ja salasanaa ei salata, vaan ne pysyvät selkokielisessä muodossa lähetettynä selaimesta palvelimelle. Lisäksi on mahdollista kaapata toisen henkilön istunto ja täten saada haltuunsa salaisia tietoja (Lane–Williams 2004, 338.) Seuraava kuvio havainnollistaa kaappaamisprosessin.



Kuvio 7. Istunnon kaappaaminen (Oswap 2010)

Istuntomuuttujien käyttämisessä avautuu mahdollisuus kaapata rehellisen käyttäjän tiedot. Käyttäjänimen ja salasanan kaappaamisen sijaan hyökkääjä

voi onkia itselleen istunnon tunnisteen (session ID) ja ottaa käyttöön istunnon (ja tätä myötä kaikki istuntoon tallennetut tiedot). Salaamattoman istunnon kaappaaminen on luonnollisesti helpompaa kuin salatun (Lane–Williams 2004, 387.)

Itse koen mieleiseksi tavaksi tunnistaa käyttäjä vertaamalla syötettyjä tunnuksia suoraan MySQL-tietokantaan. Tätä tapaa käytän myös Kuusipekan sukuseura ry:n verkkosivuilla. Mikäli annetuilla tunnuksilla on oikeudet kantaan, tiedot tallennetaan istuntomuuttujiin salattuina. Lisäksi muuttujaan tallennetaan kirjautumishetkellä käytössä ollut IP-osoite. Kun käyttäjä tarvitsee tietoa jostain taulusta, tarkistetaan ovatko tunnukset sekä IP-osoite pysyneet alkuperäisinä. Mikäli muutoksia havaitaan, tapahtuu istunnon katkaisu ja käyttäjä ohjataan takaisin etusivulle ja istuntomuuttujat tuhotaan.

Mikäli sovellus tai verkkosivu sisältää salaisia tietoja, on järkevää käyttää salattua SSL-yhteyttä (Secure Sockets Layer Protocol). Tällöin yhteys toimii salatussa tilassa, eikä ulkopuolinen henkilö pysty seuraamaan liikennettä eikä kaappaamaan istuntoa. (Philbin 2010.) SSL yhteyden kolme tavoitetta ovat

- Yksityisyys ja luottamuksellisuus
- Viestit eheässä ja alkuperäisessä muodossa lähettäjältä vastaanottajalle
- Sekä lähettäjä että vastaanottaja voivat koko ajan olla varmoja toistensa identiteetistä

Secure Sockets Layer-protokollasta on olemassa eri versioita, joista uusin on versio 3.3 (Lane–Williams 2004, 398). Tässä raportissa ei eritellä versioiden välisiä eroja.

MySQL-tietokanta pyrkii olemaan yhteyksissään mahdollisimman nopea. Tästä johtuen MYSQL-tietokanta ei oletuksena ota käyttöön SSL-protokollaa. Yhteyden suojaaminen kuormittaa palvelinta, joka taas hidastaa muita MySQL-tietokannan toimintoja (MySQL 2011b.)

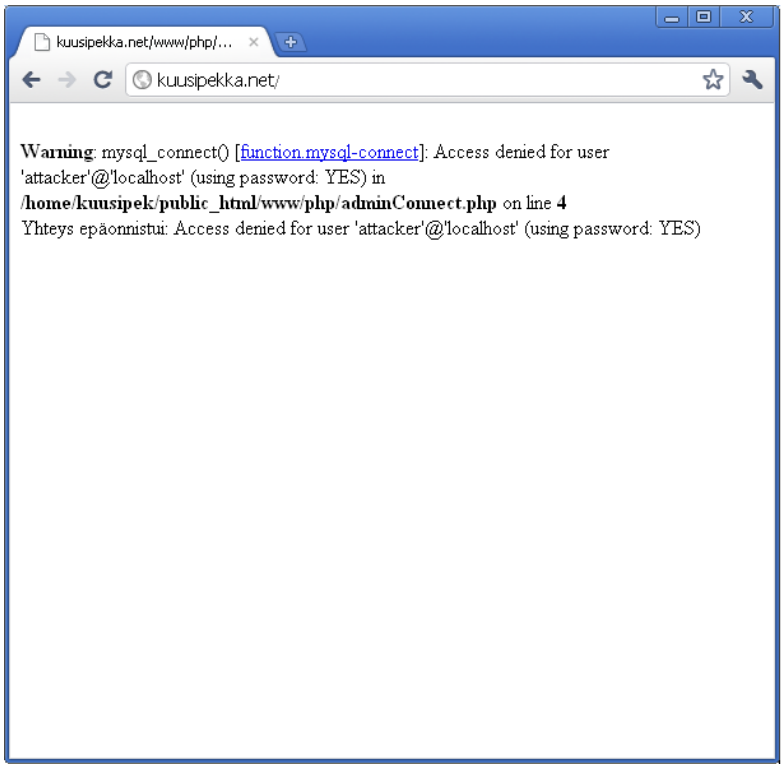
Oletuksena suojaamaton MySQL-yhteys asiakkaan ja palvelimen välillä on täysin selväkielistä. Hyökkääjä voi seurata kaikkea liikennettä näiden kahden välillä. Tällöin tietojen muuttaminen on myös mahdollista, joten esimerkiksi palvelimelta lähetetty viesti tuleekin asiakkaalle aivan eri muodossa kuin oli tarkoitettu ja päinvastoin (MySQL 2011c.)

Suojattua yhteyttä eli SSL-protokollaa käytettäessä voidaan varmistaa viestin alkuperä ja eheys. Protokolla käyttää useita eri salausalgoritmeja tiedon varmistamiseen (MySQL 2011c.)

4.7 Tiedostorakenteen piilottaminen

Mikäli potentiaalisella hyökkääjällä on hallussaan tietoa verkkosivun tiedosto- ja kansiorakenteesta, helpottaa hyökkäyksen suunnittelu ja toteutus valtavasti. Tästä syystä tiedostojen ja kansioiden nimet tulisi pitää mahdollisimman salaisina. Mikäli hyökkääjä voi nähdä suoraan jonkin tiedostopolun, voi tämä suunnitella esimerkiksi skriptin tuon tiedostopolun avulla.

Helppo tapa selvittää sivuston kansiorakenne on väärinkäyttää virheilmoituksia. Nämä virheilmoitukset ovat ensisijaisesti sivuston kehittäjien työkaluja. Toiminnassa olevilla sivustoilla esille jätetyt virheilmoitukset ovat kuitenkin merkittävä tietoturvariski niiden tarjoamien yksityiskohtaisten tietojen vuoksi. Tästä johtuen virheilmoitukset tulisi piilottaa. (Miller, R. 2009.)



Kuvio 8. Piilottamatta jätetty virheilmoitus.

Kuusipekan sukuseura ry:n sivuilla piilottamatta jätetty virheilmoitus sisäänkirjautumisvaiheessa paljasti sivustolla käytetyn kansiorakenteen ja tiedoston nimen, jossa sisäänkirjautuminen käsitellään. Virheilmoitukset tulisi korvata esimerkiksi ohjaamalla käyttäjä erityisesti luodulle virhesivulle tai näyttämällä itse asetettu virheviesti. (Miller 2009).

5 LOPPUPÄÄTELMIÄ

Tietoturva WWW-sovelluksissa koostuu useasta alueesta. Tämä on käynyt selväksi aihetta käsitellessäni ja tutkiessani. Tärkein asia on hyvin yksinkertaisesti ilmoitettuna seuraavanlainen: älä luota yhteenkään käyttäjään. Valtaosa käyttäjistä ei sovelluksen kimppuun hyökkää, mutta erityisesti WWW-sovelluksien kehittäjän tulisi pitää kaikkia potentiaalisina hyökkääjinä.

Projektin aikana Kuusipekan sukuseura ry:n verkkosivu koki monta muodonmuutosta, eikä lopullinen versio ole kirjoitushetkellä vielä valmis. Kuusipekan sukuseura ry:n verkkosivut on kuitenkin julkaistu ja nähtävissä osoitteessa www.kuusipekka.net. Tärkein on jo olemassa, eli perusturvallinen runko, jonne on kutsumattomilla vierailla hankala päästä. Mahdotonta se ei ole, sillä kuten aikaisemmin raportissa on todettu, sataprosenttisesti murtovarmaa sovellusta ei nykypäivän ihminen tai kone pysty luomaan.

LÄHTEET

Calin, B. 2009. Why File Upload Forms are a major security threat. Osoitteessa http://www.acunetix.com/websitesecurity/upload-forms-threat.htm

CERT-FI 2008. Cross site scripting- haavoittuvuudet. Osoitteessa http://www.cert.fi/tietoturvanyt/2008/03/P_6.html

CERT-FI 2010a. Älypää-pelisivuston salasanoja levitettiin julkisesti. Osoitteessa http://www.cert.fi/katsaukset/2010/tietoturvakatsaus_1_2010.html

CERT-FI 2010b. XSS-haavoittuvuus Microsoft Sharepointissa. Osoitteessa https://www.cert.fi/haavoittuvuudet/2010/haavoittuvuus-2010-076.html

CERT-FI 2010c. phpMyAdmin-ohjelmiston cross site scripting – haavoittuvuus. Osoitteessa https://www.cert.fi/haavoittuvuudet/2010/haavoittuvuus-2010-183.html

Dickinson, P. 2005. Top 7 PHP Security Blunders. Osoitteessa http://articles.sitepoint.com/article/php-security-blunders

Gilfillan, I. 2004. An introduction to MySQL permissions. Osoitteessa http://www.databasejournal.com/features/mysql/article.php/3311731/An-introduction-to-MySQL-permissions.htm

Howard, M. – LeBlanc, D. 2004. Ohjelmoijan tietoturvaopas. 1. painos. Helsinki: Edita Prima

Hovi, A. – Huotari, J – Lahdenmäki, T. 2005. Tietokantojen suunnittelu ja indeksointi. Helsinki: Docendo

Hungred 2009. Secure File Upload Check List with PHP. Osoitteessa http://hungred.com/useful-information/secure-file-upload-check-list-php/

Kofler, M. 2005. The Definitive Guide to MySQL 5. Berkley: Apress

Lane, D. – Williams, H. 2004. Web Database Applications with PHP and MySQL, Second Edition. 1. painos. O'Reilly Media.

Learn Web Tutorials.com 2011. MySQL is the most popular open source database. Osoitteessa http://learnwebtutorials.com/mysql-most-popular-open-source-database 2.5.2011

Learn PHP Online 2009. SQL Injection: How To Prevent Security Flaws In PHP / MySQL. Osoitteessa http://www.learnphponline.com/security/sql-injection-prevention-mysql-php

Machlis, S. 2011. Oracle-Sun: What happens to MySQL. Osoitteessa http://blogs.computerworld.com/oracle_sun_what_happens_to_mysql

Meloni, J. 2003. MySQL Trainer Kit. 1. painos. Helsinki: Edita Prima

Miller, R. 2009. PHP Security Guide. Osoitteessa http://php.robm.me.uk/#toc-ErrorReporting. 2.5.2011

My PHP Form 2009. Validating forms with PHP. Osoitteessa http://myphpform.com/validating-forms.php

MySQL 2011a. Securing the Initial MySQL Accounts. Osoitteessa http://dev.mysql.com/doc/refman/5.0/en/default-privileges.html

MySQL 2011b Using SSl for Secure Connections. Osoitteessa http://dev.mysql.com/doc/refman/5.1/en/secure-connections.html

Of Zen and Computing 2008. Protect Your Web Applications from SQL Injection. Osoitteessa http://www.ofzenandcomputing.com/zanswers/1159/

Oswap 2010. Cross site scripting (XSS). Osoitteessa http://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

Philbin, S. 2010. Setting Up a Secure (SSL) Connection. Osoitteessa http://www.htmlgoodies.com/beyond/security/article.php/3774876/Setting-Up-a-Secure-SSL-Connection.htm. 18.03.2011

PHP Manual 2010a. User Submitted Data. Osoitteessa
http://php.net/manual/en/security.variables.php

PHP Manual 2010b. SQL-injection. Osoitteessa
http://php.net/manual/en/security.database.sql-injection.php

Ruohonen, M. 2002. Tietoturva. 1. painos. Helsinki: Docendo

Suomen Internetopas 2011. Tietoturva. Osoitteessa
http://www.internetopas.com/yleistietoa/tietoturva

Tizag. 2010. MySQL – SQL injection prevention. Osoitteessa
http://www.tizag.com/mysqlTutorial/mysql-php-sql-injection.php

Web Cheat Sheet 2009. Secure File Upload with PHP. Osoitteessa
http://www.webcheatsheet.com/PHP/file_upload.php