**OPINNÄYTETYÖ**

**Tapani Poikela 2010**

# TIETOKANTAPOHJAISEN VERKKOSIVUN TIETOTURVA KÄYTTÄJÄN SYÖTTEIDEN POHJALTA

Rovaniemen ammattikorkeakoulu
University of Applied Sciences

**TIETOJENKÄSITTELYN KOULUTUSOHJELMA**

Rovaniemen
ammattikorkeakoulu
University of Applied Sciences

ROVANIEMEN AMMATTIKORKEAKOULU

KAUPPA JA HALLINTO

Tietojenkäsittelyn koulutusohjelma

Opinnäytetyö

# TIETOKANTAPOHJAISEN VERKKOSIVUN TIETO-TURVA KÄYTTÄJÄN SYÖTTEIDEN POHJALTA

Tapani Poikela

2010

Toimeksiantaja Rovaniemen kristillisen koulun kannatusyhdistys ry

Ohjaaja Jortikka Aarre

Hyväksytty _____ 2010 _____

Rovaniemen ammattikorkeakoulu
University of Applied Sciences

Luonnontieteiden ala
Tietojenkäsittelyn koulutusohjelma

Opinnäytetyön tiivistelmä

**Tekijä** Tapani Poikela Vuosi 2010

**Toimeksiantaja** Rovaniemen kristillisen koulun kannatusyhdistys ry
**Työn nimi** Tietokantapohjaisen verkkosivun tietoturva käyttäjän syötteiden pohjalta
**Sivu- ja liitemäärä** 35

Opinnäytetyön aiheena oli tehdä Rovaniemen kristilliselle koululle tietokantapohjaiset verkkosivut, jotka sisältävät jäsen- ja oppilasrekisterin sekä mahdollisuuden julkaista sivustolle kirjoituksia.

Opinnäytetyön tutkimusongelmana oli tietoturva tietokantapohjaisissa verkkosivuissa käyttäjän syötteitä silmälläpitäen. Työssäni huomioidaan käyttäjän antamat syötteet ja miten niihin tulee suhtautua.

Työssä käydään läpi käyttäjän syötteitä yleisesti sekä erilaisten tietoturvauhkien kautta. Työssä ei käsitellä verkkosivujen tekemistä vaan siihen liittyvää tietoturvaa ja tietoturvan uhkia. Työn tarkoitus on auttaa tekemään tietoturvallisempia verkkosivuja ja verkkosovelluksia.

Avainsana(t) PHP, SQL- injektio, Cross site scripting, Local file inclusion, Remote file inclusion

Muita tietoja

Rovaniemen ammattikorkeakoulu
University of Applied Sciences

School of Business and Administration
Business Information Technology Programme

Abstract of Thesis

---


**Author** Poikela Tapani Year 2010


| | |
|---|---|
| **Commissioned by** | Rovaniemi Christian School Support Association (Rovaniemen kristillisen koulun kannatusyhdistys ry) |
| **Subject of thesis** | Database- based website security solutions based on the user's feeds. |
| **Number of pages** | 35 |

---


The purpose of this thesis was to make a database- based web pages that contain a member and student data as well as opportunities are published for the writings. The Web page was commissioned by the Rovaniemi Christian School Support Association.

The main point of the thesis was security of database-based websites based on the user's feed. The Thesis has taken into account the user feeds and how they must be treated.

The thesis examines the user input in general and also different kind of security threats. This thesis did not deal with what web sites do but the related information security and information security threats. The purpose was to help make more secure web pages and web applications for the Rovaniemi Christian School Support Association.



Key words PHP, SQL- injection, Cross site scripting, Local file inclusion, Remote file inclusion

Special remarks

SISÄLTÖ

# 1 JOHDANTO

Opinnäytetyön aiheen valitseminen ei ollut hankalaa, koska tiesin mitä haluan tehdä opinnäytetyöksi. Tavoitteena oli löytää itselleni sellainen projekti, jossa pääsen tekemään verkkosovelluksen, joka sisältää tietokannan ja verkkosivut. Tällaisen työn sain Rovaniemen kristillisen koulun kannatusyhdistykseltä.

Työni tavoitteena oli tehdä Rovaniemen kristilliselle koululle tietoturvalliset tietokantapohjaiset verkkosivut, jotka sisältävät mahdollisuuden kirjautua sisään ja lisätä uutisia ja tekstejä sivulle. Työn toimeksiantajana toimi Rovaniemen kristillisen koulun kannatusyhdistys ry. Kannatusyhdistys halusi sivustolleen myös jäsen- ja oppilasrekisterin.

Työni tavoitteisiin kuului myös oma syvällinen perehtyminen tietokantapohjaiseen verkkosovellukseen ja tietokantapohjaisen verkkosovelluksen tietoturvaan käyttäjän syötteiden pohjalta. Halusin myös oppia suunnittelemaan ja toteuttamaan verkkosivut, joissa on paljon erilaista tietoturvallisuutta vaativaa sisältöä, kuten tietokantoja ja helppoja mahdollisuuksia verkkosovelluksen käyttäjälle muokata sivustoa haluamansa mukaan.

Tietoturva on nykyaikana suuri asia verkkosivuilla. Halusin perehtyä tietoturvaan ja testata käytännössä erilaisilla tekniikoilla, miten tietoturva toimii verkkosivuilla ja sen eri rakenteissa. Työn tarkoitus ei ole käydä läpi sitä, miten verkkosivut tehdään vaan keskittyä verkkosivujen tietoturvaan käyttäjän syötteiden kautta. Työ käsittelee erilaisia syötteisiin perustuvia mahdollisia hyökkäystekniikoita ja sitä, miten niitä vastaan voidaan suojautua.

Työn tarkoitus on selvittää, miten verkkosivut voidaan suojata mahdollisilta hyökkääjiltä. Työn sisältö on suunnattu ohjelmoijille. Työn sisällön ymmärtämiseen on hyvä tietää PHP- ohjelmoinnin perusteet ja SQL- perusteet. Myös yleistietämys ohjelmoinnista auttaa tekstin ymmärtämisessä.

## LYHENNELUETTELO

**Avoin lähdekoodi**

On avointa ohjelmistokoodia, joka tarjoaa mahdollisuuden jokaiselle käyttäjälle tutustua ohjelman lähdekoodiin, muokata ja levittää sitä haluamansa mukaan. Jotta sovellusta voidaan levittää avoimena lähdekoodina, niin sovelluksen kehittäjän pitää julkaista sovellus lisenssin alla, joka mahdollistaa sovelluksen avoimeksi. Tekijänoikeudet säilyvät sovelluksen kehittäjälle ellei niistä erikseen ole luovuttu. (Wikipedia 2010a.)

**JavaScript**

On ohjelmointikieli, joka on kehitetty tuomaan verkkosivuihin interaktiivisuutta. JavaScript toimii ehtojen mukaisesti eli se toimii siten, että koodia pitää komentaa tekemään jotain, jotta se toimii. (W3school 2010a.)

**MySQL**

On suosittu, nopea ja helppokäyttöinen avoimeen lähdekoodiin perustuva SQL-tietokannan hallintajärjestelmä, joka on asennettu yli kuuteen miljoonaan tietokoneeseen. (MySQL 2010b.)

**MySQL INFORMATION_SCHEMA**

On informatiivinen tietokanta, joka sisältää tiedot kaikista tietokannoista ja tietokantojen sarakkeista ja riveistä, jotka sijaitsevat kyseisen verkkopalvelun palvelimella. (MySQL 2010a.)

**SLL ja TSL salaus**

Tällaisia salaustekniikoita käytetään liikenteen salaamiseksi www-palvelimen ja käyttäjän selaimen välillä. Salauksia käytettäessä on ulkopuolisen käytännössä mahdotonta saada tietoa www-sivulla liikkuvasta sisällöstä. Aina kun käyttäjän henkilötietoja kysytään, niin tulisi käyttää salaustekniikkaa. (Elisa 2010.)

**PHP(Hypertext Preprocessor)**

On HTML-koodiin liitettävä, avoimeen lähdekoodiin perustuva ohjelmointikieli, joka mahdollistaa dynaamisempien verkkosivujen tekemisen. (PHP Manual 2010a.)

**PHP GET($_GET)**

Tiedonvälitystapa, jolla selain lähettää tietoa palvelimelle. Tämä tiedonvälitys tapa on kaikille avoin ja sitä ei tulisi käyttää silloin kun siirretään arkaluontoista dataa. (W3school 2010b.)

**PHP INCLUDE()**

Tällä funktiolla voidaan lisätä olemassa olevalle verkkosivulle tiedosto. Esimerkiksi sivustoon voidaan lisätä sivun yläosa ja alaosa INCLUDE()- toiminnolla. (PHP Manual 2010c.)

**SQL(Structured Query Language)**

On IBM:n kehittämä standardisoitu tietokantojen kyselykieli, jolla voidaan tehdä relaatiotietokantaan erilaisia hakuja ja kyselytä sekä sillä voidaan muokata, lisätä ja poistaa tietoja tietokannasta. (IBM 2006.)

**Traceroute**

Tällä työkalulla voidaan selvittää, mitä kautta tieto menee tietokoneelta palveluun. Windows pohjaisissa käyttöjärjestelmissä työkalua kutsutaan nimellä tracert ja UNIX- pohjaisissa koneissa sitä kutsutaan käskyllä traceroute. Työkalu toimii lähettämällä pieniä paketteja kohteeseen ja laskemalla aikaa milloin paketti tulee takaisin. (Tracert 2010.)

# 2 LÄHTÖKOHDAT

## 2.1 Rovaniemen kristillinen koulu

Viime keväänä Rovaniemelle perustettiin Rovaniemen kristillisen koulun kannatusyhdistys. Yhdistykseen kuuluu jäseniä eri seurakunnista ja se toimii yhteiskristilliseltä pohjalta. Yhdistys hakee valtionneuvostolta koulun aloittamislupaa syksyksi 2011. Mikäli lupa saadaan, koulun tarkoituksena on aloittaa esikoulu ja luokka-asteet 1-6. (Rovaniemen kristillinen koulu 2010.)

Koulu noudattaa valtakunnallista opetussuunnitelmaa ja koulun toimintaperiaate perustuu kristilliseen arvopohjaan. Kristilliset arvot korostuvat eettisen kasvatuksen alueella. Koulussa opetetaan hyväksyvää asennetta erilaisia kristillisiä oppikäsityksiä kohtaan. Opetuksessa painotetaan kristillisyyttä. Koulu palvelee myös erityistukea tarvitsevia lapsia. (Rovaniemen kristillinen koulu 2010.)

Rovaniemen kristillisen koulun kannatusyhdistys halusi luoda tulevalle Rovaniemen kristilliselle koululle verkkosivut, jotka minä näin sopivana haasteena lähteä tekemään opinnäytetyöni. Myös saamani kristillinen kasvatus edesauttoi valinnassa. Näen kristillisen koulun Rovaniemellä tarpeellisena ja hyödyllisenä.

Mahdollisuus kasvattaa lapsi myös koulun puolesta kristillisesti on todella hyvä asia ja haluan omalla työllä myös edesauttaa yhdistystä ja koulua saamaan kannatusjäseniä ja kouluun oppilaita. Nykyisin tietoverkossa esiintyminen on todella tärkeää ja tiedon saaminen verkosta on tärkeää. Sivuston avulla on tarkoitus saada uusia jäseniä ja oppilaita koululle. Ennen kaikkea sivuston tehtävä on markkinoida koulua.

## 2.2 Asiakkaan vaatimuksen mukaiset verkkosivut

Työn lähtökohtana toimi asiakkaan toivomus saada verkkosivut, jotka sisältävät jäsenrekisterin, mahdollisuuden luoda sivustolle uutisia ja muokata sivustoa helposti. Opinnäytetyötäni valitessa olin kiinnostunut tekemään tällaisen

verkkosivuston ja halusin oppia käyttämään koulussa oppimiani asioita hyödyksi käytännössä. Sivustoa suunnitellessani piti ottaa huomioon seikkoja, jotka ovat verkkosivujen suunnittelussa tärkeitä.

Sivusto sisältää yhdistyksen jäsenrekisterin ja koulun oppilasrekisterin, joten tietoturvallisuus on suuressa osassa sivuston suunnittelua, toteutusta ja testatusta. Tietoturvallisuuden lisäksi halusin suunnitella käyttäjille helpon ja selkeän käyttöliittymän, jolla voidaan lisätä sivustolle kätevästi uutisia. Halusin myös, että pääkäyttäjän on helppo hallinnoida jäsen-, ja oppilasrekisteriä.

Sivustolle kannatti tehdä käyttäjätasot, joilla saadaan rajattua käyttäjiä, niin etteivät kaikki käyttäjät voi toimia jäsen- ja oppilasrekisterin puolella, koska tällainen valinta on tietoturvan kannalta vaarallista. Sivuston tekemisessä piti tietoturva pitää koko ajan mielessä.

Työn tekemistä hidasti se, että ei ollut olemassa varsinaista pohjaa tai mitään vanhoja verkkosivuja, mistä lähteä suunnittelemaan sivustoa. Aloitin sivuston työstämisen nollasta ja päädyin lopuksi yhden navigointivalikon järjestelmään, joka sisälsi tarvittavat sisällöt. Sivuston ulkoasun idea syntyi koulun liitutaulusta, jossa on rauhalliset värit. Sivusto on yksinkertainen ja helppokäyttöinen.

Sivustoa suunnitellessa mietin eri vaihtoehtoja toteutukselle: lähdenkö tekemään sivua jollakin mahdollisella avoimeen lähdekoodiin perustuvalla julkaisujärjestelmällä vai rakennanko sivuston kokonaan itse. Päätin lähteä rakentamaan järjestelmän itse. Oman oppimiseni ja syventymiseni lisäksi päätökseeni vaikutti muutama pinnalla oleva tietoturvaan liittyvä uhkakuva, joita oli avoimen lähdekoodin verkkojulkaisujärjestelmissä, ja joista Cert-fi tietoturvaviranomainen uutisoi. (Cert-fi 2010a.) Kuvaan niitä neljännessä luvussa.

### 2.3 Oma perehtyminen

Tietokantapohjaisella WWW-julkaisujärjestelmällä tarkoitetaan WWW-palvelimelle asennettavaa ohjelmistoa, jonka avulla hallitaan sivuston sisäl-

töä, tarvitsematta ohjelmointitaitoja. (Ubinet 2010) Markkinoilla on todella paljon erilaisia avoimeen lähdekoodiin perustuvia julkaisu- ja sisällönhallintajärjestelmiä, joista olisi voinut lähteä hakemaan verkkosivustolle pohjaa ja käyttämään niitä sivujen tekoon. Pääsin aloittamaan kaiken puhtaalta pöydältä, joka toi osaltaan myös vaikeuksia sivuston tekemiseen.

Sivuston lopullisen ulkoasun löytämiseen meni aikaa. Rakensin CSS- tyyleillä sivun ulkoasun ja asettelun. Sivuston ulkoasun rakentaminen oli mielekäs projekti, ja siinä samalla tein myös sivuston pohjaa. Aikataulu oli työlle sopiva, joka mahdollisti kaikenlaista kokeilua sivustolla ja myös sivuston sisällön suunnitteluun jäi riittävästi aikaa. Työn aloittamisen ajankohta oli kesällä 2010, joten sivustoa sai tehdä kesän ja syksyn 2010 ajan. Aikaa sivuston tekemisen harjoitteluun oli riittävästi.

Sivuston sisältö rakentui PHP- ja HTML-koodilla. MySQL-tietokanta toimi tietokannan hallintajärjestelmänä, jolla sai rakentaa verkkosivuston tietokannan kokonaan itse. Koulussa oppimiani asioita yhdistämällä sain toimivan yhdistelmän tietokantojen käsittelyä varten.

Työssäni pääsin perehtymään hyvin sekä ohjelmointiin että ulkoasulliseen puoleen. Opinnäytetyö on rajattu kyseisten ohjelmointikielten ja tekniikoiden sisältöön, funktioihin ja niiden tietoturvaan. Työn edettyä riittävän pitkälle, pääsin perehtymään syvällisemmin tutkimusongelmaan, joka käsittelee tietokantapohjaisen sivuston tietoturvaratkaisuja käyttäjän syötteiden näkökulmasta.

Käyttäjän syötteisiin perustuva tietoturvallisuus on todella tärkeää kyseisessä työssä etenkin, kun tietokantaan syötetään paljon tietoja, niin käyttäjistä, jäsenistä kuin oppilaistakin. Uskon, että syvällinen perehtyminen SQL- ja PHP-tietoturvaan ja sen tietoturvauhkiin auttaa ohjelmistojen koodien ymmärtämistä ja omaksumista. Tämä lisää taas ammatillista valmiutta ja parantaa omaa ohjelmointia huomattavasti. Syvällinen perehtyminen tietoturvallisuuteen tuo ammatillista valmiutta ja parantaa työnhaun mahdollisuuksia. Verkkosivujen tietoturvallisuus on oman osaamisen kannalta tärkeää, koska pyrin myös töihin ohjelmointipuolelle.

# 3 Tietoturvaratkaisuja tietokantapohjaiselle verkkosivulle

## 3.1 Perusidea tietoturvallisuudessa

Perusajatuksena voidaan pitää sitä, että ei ole olemassa täysin tietoturvallista verkkosovellusta. Kaikki mitä ihminen kehittää, niin ihminen voi sen myös murtaa. Vaikka keksisit minkälaisen turvaratkaisun, niin aina löytyy innokkaita ihmisiä purkamaan salaukset. Tietoturvallisuudessa voidaan myös käyttää hyvänä esimerkkinä sitä, että vaikka sinulla on tietokoneesi kiinni, niin se voidaan varastaa. Verkkosivujen kriittisimmät kohdat koostuvat palvelusta itsestään, palvelun käyttäjistä sekä verkosta näiden välillä. Ideana on, että mikään näistä kolmesta ei saa luottaa toisiinsa. Käyttäjän näkökulmasta katsottuna verkossa liikkuva data on turvatonta ja palveluunkaan ei voida luottaa. (Mureakuha 2006b.)

Traceroute-toiminnolla voidaan selvittää käyttäjän ja palvelun välissä olevat järjestelmät (tietokoneet tms.). Mitkä tahansa näistä koneista pystyvät seuraamaan käyttäjän ja palvelun välistä tietoliikennettä. Traceroute-toimintoa voidaan testata kirjoittamalla käyttöjärjestelmän komentoriville esimerkiksi: tracert www.sonera.fi. Tällä toiminnolla saadaan selville koneet jotka ovat oman koneen ja www.sonera.fi palvelun välissä. (Tracert 2010.)

Välitettävän tiedon tai informaation pitämiseksi salassa on kehitetty salaus metodeja, joilla saadaan tieto vaikeasti luettavaan muotoon. Vaikka suojausmetodeja on, niin tieto pystytään tallentamaan ja aukaisemaan jossakin ajassa, siksi pankeilla on käytössä vaihtuvat tunnusluvut, joilla estetään ajallisesti tämä tapahtuma. Myös palvelun ja käyttäjän välinen liikenne on turvattava esimerkiksi, niin ettei käyttäjä pääse tekemään peruttamattomia vahinkoja palveluun. (Mureakuha 2006b.)

Verkossa olevan palvelun tulisi suhtautua kaikkiin käyttäjiin varauksellisesti. Luotettu käyttäjä voi olla murrettu ja näin käyttäjä ei olekaan se joksi palvelu sitä luulee. Käyttäjistä pitää ottaa huomioon sellaiset käyttäjät, jotka haluavat aiheuttaa vahinkoa tahallaan ja poistavat tietoja ilman syytä. Käyttäjille sattuu vahinkoja, joten huomioon on otettava myös vahingossa poistetut tiedot. (Mureakuha 2006b.)

Tietomurtojen ehkäisemiseksi on tärkeää ottaa huomioon sekä käyttäjä että käyttäjän syötteet. Kuten aikaisemmin mainitsin, niin käyttäjään ei saa eikä voi luottaa. Jos käyttäjään ei voida luottaa palvelussa, niin sen toimintaa pitää pyrkiä rajoittamaan palvelussa niin, ettei käyttäjä voi tehdä syötteillään sellaista, mitä sen ei pitäisi pystyä tekemään.

Seuraavien kappaleiden tarkoituksena on avata tietokantapohjaisen verkkosivun ohjelmointia. Kolmannen kappaleen tarkoituksena on käsitellä tietoturvallisuutta käyttäjää, käyttäjän asetuksia ja oikeuksia ja käyttäjän syötteitä silmällä pitäen.

### 3.2 WWW-palvelimen tietoturva käyttäjää silmällä pitäen

Palvelin on tietokone, joka tarjoaa siinä ajettavien palvelinohjelmistojen välityksellä erilaisia palveluja ohjelmille. Käyttäjät voivat sijaita samalla tai eri koneilla. (Linux 2009.) WWW-palvelin on tarkoittaa tietokonetta tai ohjelmistoja joka jakaa HTTP-protokollalla dokumentteja asiakkaille. HTML-kielellä kirjoitetut dokumentit muodostavat WWW-sivuja, jotka voidaan näyttää asiakaskoneissa selainohjelman avulla. (Wikipedia 2010e.)

WWW-palvelimen tietoturvaan voidaan vaikuttaa palvelinvarmenteella. Palvelinvarmenteella varmennetaan palvelun tarjoaja. Väestörekisteri myöntää palvelinvarmenteita, joilla voidaan tunnistaa palvelun tarjoaja. Suomessa näitä varmenteita myöntää väestörekisterikeskus ja ne ovat automaattisesti luotettuja. Palvelinvarmenteen avulla palvelun käyttäjä voi varmistua palvelun tarjoajan aitoudesta. Palvelinvarmenne mahdollistaa selaimen ja palvelimen välille SSL – suojatun tietoliikenteen. (Väestörekisterikeskus 2010.)

Palvelinvarmenne voidaan määritellä käyttökohteen mukaisesti: palvelimen tunnistamiseen, asiakkaan tunnistamiseen ja molemmat samanaikaisesti. Näitä varmenteita voidaan käyttää verkkopalveluissa kolmella eri tavalla. Ensimmäisenä käyttötapana on pelkkä palvelinvarmenne, jossa sivut käyttävät osittain tai kokonaan suojattua tietoliikennettä. Tällöin tietoliikenne on suojat-

tu käyttäjän selaimen ja palvelun välillä. Palvelussa voidaan käyttää perinteistä käyttäjätunnus- ja salasana-yhdistelmää. (Väestörekisterikeskus 2010.)

Toinen tapa missä voidaan käyttää varmenteita, on palvelun ja ennalta tunnistamattoman käyttäjän välillä olevat varmenteet. Palvelun tarjoaja tarjoaa palvelun käyttäjälle oman palvelinvarmenteen kuten kortin, jolla käyttäjä voidaan varmentaa palvelussa. Käyttäjä tulee palveluun, se kirjautuu kortin varmenteella palveluun. Tällä saadaan haettua kortin haltijan henkilöllisyys palvelun sovelluskyselyn avulla. Sovelluskyselyssä tulee käyttäjän vielä hyväksyä tunnistaminen palvelun ilmoittamalla kysymyksellä. Tämän pohjalta pystytään toteuttamaan laajoja palveluita ennalta määrittelemättömälle käyttäjäryhmälle. Tällaisia palveluita ovat tyypillisesti verkkokaupat. Tämä menetelmä mahdollistaa myös sähköisen allekirjoituksen. (Väestörekisterikeskus 2010.)

Kolmantena tapana on ennalta määritetyn käyttäjän ja palvelimen varmenteet. Kyseinen käyttötapa eroaa edellisistä siten, että se on linkitetty johonkin esimerkiksi tietokannan käyttäjätunnukseen. Toisin sanoen voidaan lukea kortti, jolla kirjaudutaan järjestelmään ja saadaan kortin avulla oikeudet käyttää kortin oikeuksien mukaisia oikeuksia järjestelmässä. (Väestörekisterikeskus 2010.)

Kyseisillä kolmella tekniikalla voidaan luoda laaja verkkopalvelu, jolla voidaan varmentaa tietoliikennettä palvelun ja käyttäjän välillä. Näillä tekniikoilla saadaan salassapitoa vaativia tietoja liikuteltua turvallisesti ilman ulkopuolisia käyttäjiä. Jos kyseisiä varmenteita ei ole verkkopalvelussa käytössä, niin verkossa liikkumisesta voidaan tehdä tietoturvallisempaa tulevissa kappaleissa esitetyin tavoin. Seuraavan kappaleen tarkoituksena on opastaa verkkosivuston käyttäjätunnuksen ja käyttäjätasojen rakentamisessa ja antaa ohjeita niiden toteuttamiseen.

### 3.3 Käyttäjätasot ja käyttäjätunnukset

Suunnitellessa verkkosivuston käyttäjätasoja löytyi WordPress-julkaisujärjestelmästä hyvä pohja mistä lähteä suunnittelemaan sivuston

käyttäjätunnuksia ja käyttäjätasoja. WordPresss-julkaisujärjestelmässä on käyttäjätasot rajattu 0-tasosta 10-tasoon. 0-tasolla on kaikista vähiten oikeuksia ja 10-tasolla on sivuston pääkäyttäjän oikeudet ja mahdollisuudet tehdä sivustolle mitä haluaa. Sivuston 10-tason käyttäjä voi siis lisätä, muokata, poistaa ja säätää käyttäjätasoja jokaisen käyttäjän osalta. WordPressin käyttäjätasot etenevät loogisesti: mitä isompi taso käyttäjällä on niin sitä enemmän ominaisuuksia ja oikeuksia käyttäjällä on. (WordPress 2010.)

Esimerkiksi 0- tason käyttäjä pystyy kirjautumaan sisään, näkemään työtilan, luomaan oman profiilin ja muokkaamaan sitä. 0- tason käyttäjällä ei ole muita oikeuksia.1- tason käyttäjällä on 0-tason oikeuksien lisäksi kirjoittamaan kirjoituksia sivulle mutta ei julkaisemaan niitä. 1- tason oikeuksilla pystyy myös muokkaamaan ja poistamaan omia kirjoituksia. Käyttäjätasojen noustessa ominaisuudet lisääntyvät mahdollistavat käyttäjälle lisää toimintoja ja mahdollisuuksia verkkosivujen käytössä. (WordPress 2010.)

Opinnäytetyön tietoturvallisuuden vuoksi oli hyvä luoda käyttäjätunnuksille käyttäjätasot ja rajoittaa niitä niin, etteivät kaikki käyttäjät pääse käsiksi kaikkeen tietoon. WordPressin käyttäjätasojen pohjalta kehitin omille sivuille riisutun version käyttäjätasoista. Perusajatuksena oli luoda kolme tasoa, jossa sivustolla olisi korkeimman tason kaikki käyttäjäominaisuudet omaava pääkäyttäjä, toisena tasona olisi sihteeritason käyttäjä. Sihteeritason käyttäjää olisi riisuttu hieman. Sihteeritason käyttäjäominaisuudet ovat rajattu niin, että pysyvät poisto tapahtumat tapahtuisi pääkäyttäjän toimesta, myös käyttäjätunnusten mahdollinen muokkaaminen tapahtuisi pääkäyttäjän toimesta.

WordPressin pohjaa apuna käyttäen alimmalla tasolla olisi opettajatason käyttäjät. Tällä käyttäjätasolla on mahdollista lisätä sivustolle kirjoituksia, muokata niitä ja merkitä niitä poistettavaksi. Lopullisen poistamisen tulee tekemään päätason käyttäjä. Opettajatason käyttäjällä on mahdollisuus muokata omaa profiiliaan. Käyttäjillä on mahdollisuus valita oman profiilin käyttäjätunnus, mutta tietyin ehdoin.

Tietoturvallisuuteen liittyvä rikollisuus on kasvussa ja siksi on hyvä, että verkossa pyritään liikkumaan niin turvallisesti kuin mahdollista. Kun käyttäjä

saapuu palveluun, niin hänen tulee tehdä käyttäjätunnus. Käyttäjätunnuksien luominen verkkosivustolle tulisi tehdä siten, että se perustuu tiettyihin ehtoihin. Käyttäjätunnuksen tulee olla tietyn mittainen, useasti verkkosivuilla esiintyvä minimivaatimus on kuusi merkkiä. Käyttäjätunnuksen olisi hyvä sisältää muitakin kuin kirjaimia. Käyttäjätunnus saa sisältää merkkejä kuten ”.”, ”-” ja ”_”. Oma käyttäjätunnus tulisi pitää salassa ja suojata se ulkopuolisilta. Sen tulee olla lisäksi uniikki ja itselle helposti muistettava. (Charter 2010.)

Käyttäjätunnuksille asetettujen ehtojen on tarkoitus lisätä tietoturvallisuutta huomattavasti, jotta käyttäjätunnukset eivät olisi ennalta arvattavia. Näitä käyttäjätunnusehtoja on soveltaen käytetty työssä tehdyillä verkkosivuilla. Vahva käyttäjätunnus ei yksin riitä vaan vahvan käyttäjätunnuksen lisäksi käyttäjän tunnistaminen pitää tapahtua tietoturvallisesti.

### 3.4 Käyttäjän tunnistaminen

Käyttäjän tullessa palveluun on hänen todistettava henkilöllisyytensä. Tämä tapahtuu useimmiten käyttäjätunnuksella ja salasanalla. On olemassa erilaisia käyttäjän tunnistamismahdollisuuksia. Fyysinen tunnistus tapahtuu esimerkiksi sormenjälki tunnistamisella tai pankkikortilla. Toinen vaihtoehto tunnistukselle on salasanatunnistus. Kolmantena tunnistustapana on vuorovaikutteinen tapa. Vuorovaikutteisella tunnistustavalla tarkoitetaan esimerkiksi pankkien vaihtuvia avainlukuja, joilla pankkien tunnistusjärjestelmä toimii. (Mureakuha 2006b.)

Kuten käyttäjätunnukselle, on salasanalle asetettava ehtoja, joilla siitä saadaan vahva ja tietoturvallinen. Tällaisia ehtoja ovat salasanan pituuden määrittäminen ja salasanan sisältämien merkkien määritteleminen. Salasana ei saa sisältää merkkejä kuten ”\” ja ” ” ”. Salasanan tulisi olla sellainen että sen muistaa helposti mutta se on vaikeasti arvattava. Myös esimerkiksi o korvaaminen nollalla edesauttaa salasanan vahvuutta. Eri salasanojen käyttäminen eri palveluissa on suositeltavaa ja salasanan vaihtaminen 30-60 päivän välein lisää tietoturvallisuutta. (Charter 2010.)

PHP-ohjelmointikieli sisältää funktion, jolla saadaan testattua salasanan vahvuutta ja se voidaan näyttää käyttäjälle esimerkiksi juuri mittarina. PHP – funktio on nimeltään Crack, joka käyttää hyväkseen PHP: n CrackLib- kirjastoa, joka testaa tämän kirjaston avulla kuinka vahva salasana on. Funktio käy läpi seuraavia asioita: sisältääkö salasana helposti arvattavia sanoja, sisältääkö salasana isoja ja pieniä kirjaimia ja sisältääkö salasana numeroita. Täten muodostaa arvion salasanan vahvuudesta. Myös salasanalle asetettu salasanan toistaminen tai uudelleen kirjoittaminen verkkopalvelussa on hyvä lisä salasanan turvallisuuteen. Tämä kenttä vertaa salasanaa ja varmistussalasanaa keskenään ettei tule lyöntivirheitä salasanassa. Tällä tavoin saadaan käyttäjälle turvallinen tapa tehdä hyvä ja vahva salasana ja myös varmistaa että salasana on juuri sitä mitä käyttäjä itse haluaa. (PHP Manual 2010c.)

Tällaisten käyttäjän tunnistamisen ehtojen asettaminen ja palvelun käyttäjälle suunnattujen apuvälineiden kautta saadaan paljon tietoturvallisempi verkkosivusto. Opinnäytetyön verkkosivujen tietoturvallisuutta tehdessä näistä ehdoista, funktiosta ja apuvälineistä oli paljon hyötyä, jotta sivustosta saatiin tietoturvallisempi ja helpompi käyttää.

### 3.5 Käyttäjän syötteet

Käyttäjän syötteet luovat verkkosovellukselle tietoturvallisuuden kannalta suuria tietoturvauhkia, kuten SQL-injektio, Local- ja remote file inclusion ja Cross site scritping-hyökkäystekniikoita. Näihin hyökkäystekniikoihin perehdytään neljännessä kappaleessa. Tämän kappaleen tarkoituksena on antaa lisää keinoja, joilla saadaan käyttäjän syötteitä turvallisemmaksi ja selkeämmäksi. Tulevissa kappaleissa esittelen muutaman keinon miten lisätä tietoturvallisuutta hyökkäysten estämisen lisäksi.

#### 3.5.1 Käyttäjän syötteiden varmentaminen

SQL- injektion estämisessä käytettyjen menetelmien lisäksi, käyttäjän syötteitä voidaan lisäksi varmistaa erilaisilla määrityksillä. Käyttäjän syötteiden varmentaminen on erityisen tärkeää tilanteissa, joissa pyydetään käyttäjältä esimerkiksi puhelinnumeroa tai sähköpostiosoitetta.

Puhelinnumeroa pyydettäessä on hyvä määritellä tekstikenttään asetuksia, joilla ehkäistään haitallisia syötteitä. Puhelinnumero kenttä saa sisältää vain numeroita, ”+” ja ”-” merkkejä ja sulkuja. Myös tallennusmuoto on hyvä määritellä. Sähköpostin varmentaminen on hyvä tehdä käyttäjän rekisteröityessä palveluun. Se toteutuu siten, että käyttäjä saa rekisteröityessään sähköpostiinsa aktivointi viestin, jolla hänen pitää varmistaa se, että sähköposti on oikeasti olemassa. On myös hyvä lisätä sähköpostin syöttöön tarkoitetun kenttään ominaisuuden, joka vaatii käyttäjältä @ - merkin käyttöä. (Microsoft 2010.) Käyttäjän syötteitä voidaan myös varmistaa määrittelemällä jokaiselle tietoa syötettävälle kentälle ominaisuudet.

Kaikki mihin käyttäjä voi syöttää tekstiä, niin niillä tulee olla ennalta määritetyt ominaisuudet, jotka vaikuttavat syötettävään tekstiin. Jokaisen kentän syötettävien merkkien määrittelemisen lisäksi on tärkeää määritellä kenttien pituudet ja tyyppi. Kenttiin syötettävän datan minimipituus ja maksimipituus tulee määritellä, jotta sivustolle ei voida syöttää haitallista koodia. Toinen mahdollinen häirintätapa on syöttää kenttiin niin paljon tekstiä, että se tukkii palvelun. Kentän tyypin määritteleminen estää sen, ettei mahdollinen hyökkääjä voi lisätä tekstikenttään kuvan sisältävää koodia. Sivustoa tehdessä kannattaa testata paljon sellaisia hyökkäyskoodeja, jotka ovat vaarallisia omalle sivustolle. Kaikki data, mitä sivusto käsittelee, tulee olla varmistettua ja turvallista. Lisäksi merkit kuten ”;”, ” ’ ”, ”--”, ”/*…*/” ja ”xp_” tulee olla estettynä ellei niitä erikseen tarvita. (Microsoft 2010)

### 3.5.2 Virheiden raportointi

Verkkosivuilla olevat virheilmoitukset eivät ole tarkoitettu käyttäjiä varten vaan sivuston ylläpitoa varten. Jos käyttäjä näkee verkkosivulla virheilmoituksen, esimerkiksi SQL virheen niin se vaarantaa sivustoa, koska käyttäjä saa tiedon, miksi virhe tulee. Sen sijaan tulisi tehdä ilmoitus siitä, että on tullut jonkinlainen virhe ja tästä on ilmoitettu sivuston ylläpitäjille. Näin voidaan saada tietoa käyttäjän toimista ennen virhettä. Käyttäjälle esiintyvä virheilmoitus voisi olla esimerkiksi ”Tietokantavirhe. Ylläpidolle on ilmoitettu.” Tällaisella virheilmoitustyylillä voidaan saada tietoon käyttäjä, joka aiheuttaa tahallaan paljon virheitä sivustolla ja näin voidaan myös mahdollisesti sulkea käyttäjä

pois sivustolta. Käyttäjän pois sulkemisessa kannattaa olla varovainen, koska virheet voivat olla myös palvelussa. (Mureakuha 2006b.)

### 3.5.3 Tietojen poistaminen

Kun käyttäjä, joilla ei ole pääkäyttäjän valtuuksia haluaa poistaa esimerkiksi henkilön jäsenrekisteristä, niin sitä ei poisteta suoraan vaan merkitään poistettavaksi. Tällä voidaan vähentää vahingossa sattuneita poistotapauksia ja ne voidaan palauttaa ennen kuin vahinkoa pääsee tapahtumaan. Kun poistotapahtuma on ollut tietyn ajan merkittynä poistettavaksi, voidaan se ylläpidon toimesta poistaa pysyvästi. Tietojen muokkaamisen ja poistamisen on hyvä tapahtua vaiheittain, koska tällä voidaan vähentää vahingossa tapahtuvien muokkaus ja poisto tapahtumien määrää. Käyttäjältä kysytään varmistus jokaiseen muokkaus ja poisto tapahtumaan. (Mureakuha 2006b.)

## 3.6 Huomion arvoista PHP ohjelmoinnissa

Kun PHP-ohjelmoinnilla tehdään tietoturvallisuutta vaativaa ohjelmakoodia, on hyvä ottaa huomioon PHP: n ominaisuuksia, jotka vaikuttavat paljon tietoturvallisuuteen. Kun PHP: llä tehdään koodia, niin kannattaa kytkeä niin sanotut PHP: n vahingolliset ominaisuudet pois päältä. Esimerkiksi Register globals on PHP: n ominaisuus, joka mahdollistaa alustamattomien muuttujien syöttämisen suoraan esimerkiksi GET- tai POST-parametrina. Ohjelmistojen kehittäjien onneksi tämä ominaisuus on oletuksena kytketty pois päältä. Tuntemattomalla palvelimella tämä asetus on aina hyvä tarkistaa. (Auralinna 2009.)

Toinen huomion arvoinen PHP: n ominaisuus on Magic quotes. Se on tehty helpottamaan SQL-injektion estämistä, jottei kaikkea dataa tarvitsisi tehdä vaarattomaksi escape- funktiolla. Magic quotes sisältää kuitenkin muutamia ongelmia, joten se on parempi pitää pois päältä. (Auralinna 2009.)

Seuraavassa kappaleessa käsitellään käyttäjän syötteitä hyökkäystekniikoiden kuten SQL- injektion, local ja remote file inclusion sekä Cross site scriptingin-tekniikoiden kautta. Hyökkäystekniikat käsitellään yksitellen ja yksin-

kertaisesti niin, että niitä vastaan voidaan puolustautua ja niiden avulla on mahdollista parantaa omaa osaamistaan tietoturvallisen verkkosovelluksen rakentamisessa.

## 4 Käyttäjän syötteisiin perustuvia tietoturvauhkia

Tässä luvussa käydään läpi seuraavat vaaralliset hyökkäystekniikat. Alla esitettyjä hyökkäystekniikoita vastaan ei auta virustorjunta järjestelmät eivätkä minkäänlaiset palomuurit vaan itse ohjelmoitu koodi ja sen tarkistaminen ja testaaminen. Alla esitetyt tekniikat ovat käyttäjän syötteisiin perustuvia hyökkäystekniikoita. Työ käsittelee tulevissa kappaleissa SQL-injektiota, local- ja remote file inclusion ja cross site scripting hyökkäystekniikoita. Mielenkiinto aiheesta heräsi opinnäytetyön aiheen valintaa tehdessä alla olevasta uutisesta.

Uutisessa kerrottiin jatkuvasti löytyvistä uusista haavoittuvuuksista, joiden vuoksi palvelut ja käyttäjät ovat vaarassa altistua hyökkäyksille. Hyväksikäyttömenetelmiä, jotka kohdistuva sisällönhallintajärjestelmiin ja verkkokauppaohjelmistoihin, on havaittu kymmeniä. Useimmiten hyökkäystekniikat olivat SQL-injektio, local- ja remote file inclusion hyökkäykset ja suuri osa hyökkäyksistä suoritetaan näitä menetelmiä käyttäen. SQL-injektio hyökkäyksen avulla tietokannasta voidaan hakea tietoa, kuten käyttäjätunnuksia ja salasanoja. Local file inclusion-hyökkäyksellä voidaan päästä käsiksi palvelimen tiedostoihin, joihin www-palvelun prosessilla on käyttöoikeus. Remote file inclusion hyökkäystä käyttäen voidaan huijata palvelin suorittamaan toiselta palvelimelta haettavaa ohjelmakoodia. (Cert-fi 2010a.)

Murrettuja palvelimia voidaan käyttää hyväksi esimerkiksi haitallisten tiedostojen levittämiseen tai tiedostojen vakoilemiseen. Paras tapa suojautua tällaisia murrettuja palvelimia vastaan satunnaiselle verkkosivujen käyttäjälle on pitää oman selaimen lisäosat kunnossa ja oman tietokoneen virustorjunta valmiudessa ja päivitettynä. Palveluiden ylläpitäjien pitäisi huolehtia siitä, että järjestelmät ovat ajan tasalla ja käytössä on aina uusimmat versiot. Perusajatuksena voidaan myös pitää sitä, että ellei lisäosaan ole päivitystä, niin otetaan se kokonaan pois käytöstä tai rajoitetaan se niin, ettei vahinkoja pääse tapahtumaan. (Cert-fi 2010a.)

Uutisessa mainittujen hyökkäystekniikoiden lisäksi Cross site scripting-hyökkäystekniikka on käyttäjän syötteisiin perustuvaa, joten tämä menetelmä

on lisätty kappaleeseen. Seuraavissa luvuissa on tarkoitus käydä läpi yllä-esitettyjä tekniikoita, niin että tällaiset hyökkäystekniikat tiedostetaan olevan, ne pystytään havaitsemaan ja niitä vastaan osataan suojautua. Nämä kolme hyökkäystekniikkaa perustuvat pitkälti käyttäjän syötteisiin, jotka johtavat tietynlaiseen lopputulokseen verkkopalvelussa.

### 4.1 SQL- Injektio

SQL- injektio on vanha mutta suhteellisen yleinen hyökkäys tietokantaa käyttäviä sovelluksia vastaan ja on varsin yleinen WWW-pohjaisissa sovelluksissa. Hyökkäys tapahtuu virheellisesti rakennetun tietokantakyselyn kautta. Injektiolla voidaan syöttää virheelliseen kyselyyn erilaisia kyselyn osia, joilla voidaan päästä hakemaan arkaluonteista tietoa tai jopa muokkaamaan tietokannan tietoja, joita ei pitäisi päästä muokkaamaan. Tietokantoja voidaan jopa muokata niin, ettei ylläpitäjällä ei ole mitään havaintoa asiasta. Hyökkäys tapahtuu useimmiten puuttuvan tai väärin toteutetun syöttötiedon tarkistuksen kautta ja joskus tietokantarajapinnan väärästä tiedon käsittelystä. Kaikki käyttäjältä tuleva tiedon oikeellisuus tulee tarkistaa ja tietotyyppien kohdalla pitää varmistaa että tieto on oikeassa muodossa. Esimerkiksi numerokenttään syötettävät numerot ovat todella numeroita. (PHP Manual 2010e.)

Oletetaan, että verkkosivuilla kirjautumiskohta on se, mihin käyttäjä syöttää käyttäjätunnuksen ja salasanan. Kun käyttäjä syöttää käyttäjätunnuksen ja salasanan ja kirjautuu sisään, niin syntyy kysely tietokantaan, jossa tarkistetaan että käyttäjätunnus ja salasana löytyvät taulusta kayttajat. Kysely on esitetty alla:

”SELECT * FROM kayttajat WHERE kayttäjätunnus =”$tunnus” AND salasana= ”$salasana”;

Jos kysely on yllä esitetyssä muodossa, niin mahdollinen hyökkääjä voi käyttää kyselyä hyväksi ja syöttää tietokantaan haluamiaan tietoja käyttäen esimerkiksi kayttajatunnus- kenttää hyväkseen. Hyökkääjä kirjoittaa kayttajatunnus - kenttään komennon ’OR 1=1- -’, jolloin hyökkääjä ei tarvitse salasanaa

tutkiakseen tietokantaa, koska vaatimukset salasanasta poistuvat ”- -” merkein tehden salasanakyselystä kommentin eikä kyselyn. (Sqlbook 2010a.)

Huomion arvoista on myös se, ettei hyökkääjän tarvitse tietää taulujen tai sarakkeiden nimiä, jotta tämä hyökkäys onnistuisi. Tätä hyökkäysmenetelmää voidaan käyttää kaikissa kentissä, mihin käyttäjä voi syöttää tietoja verkkosivuilla ja aiheuttaa suuriakin ongelmia. SQL-injektion löytyminen verkkosivuilta mahdollistaa jopa koko tietokannan tuhoamisen. (Sqlbook 2010a.)

Hyökkääjä pystyy hakemaan erilaisilla komennoilla tietokannasta taulujen nimet, ja tämä mahdollistaa taulujen poistamisen. Verkkosivuilla oleva hakukenttä, mistä käyttäjä voi hakea sivujen sisältöä. Hyökkääjä voi hakea samasta kentästä taulujen ja kenttien nimiä, jos tietokantakyselyä ei ole tehty oikein. Hyökkääjä pystyy käyttämään INFORMATION_SCHEMA tauluja haun apuna. INFORMATION_SCHEMA taulut sisältävät verkkosivujen taulujen ja niiden kenttien ja rivien tiedot. (Sqlbook 2010a.)

Hyökkääjä käyttää toiminnossaan INFORMATION_SCHEMA taulujen lisäksi UNION lausetta. UNION lausekkeella voidaan ketjuttaa SELECT toimintoa niin, että haku jatkuu toisesta taulusta edellisen haun päätyttyä. Hyökkääjä käyttää hyväksi INFORMATION_SCHEMA tauluja ja toimii hakukenttään kirjoittamalla esimerkiksi seuraavanlaisen haun:

'UNION SELECT TABLE_CATALOG, TABLE_SCHEMA,
TABLE_NAME, 1FROM INFORMATION_SCHEMA. Tables –

Tällä haulla hyökkääjä saa selville tietokannan nimen(TABLE_CATALOG), taulun käyttäjän tai omistajan(TABLE_SCHEMA), taulun nimen(TABLE_NAME) ja kaiken sen mitä taulun kentät ja rivit sisältävät voi hyökkääjä hakea kyselyillä. Jos tällainen hyökkäys on mahdollinen, niin hyökkääjä voi halutessaan kirjoittaa esimerkiksi kayttajatunnus- kenttään komennon DROP TABLE esimerkkitaulu ja täten tuhota taulun. Koko tietokannan tuhomaiseen ei mene käytännössä katsoen kuin muutama minuutti. (Sqlbook 2010a.)

Lisäksi huomioon kannattaa ottaa se, että SQL-injektio ei ole mitenkään riippuvainen käytettävistä tietokantajärjestelmistä (SQL Server, Oracle, yms.), eikä myöskään käyttöjärjestelmistä (Windows, Linux, yms.) tai verkkosovelluksen ohjelmointikielestä (PHP, ASP, yms.). Se toimii myös Java, C++, yms. ohjelmistoissa. Mikäli hyökkääjä pystyy hyödyntämään tätä menetelmää, niin hän voi tehdä sivustolla mitä haluaa. (Sqlbook 2010a.)

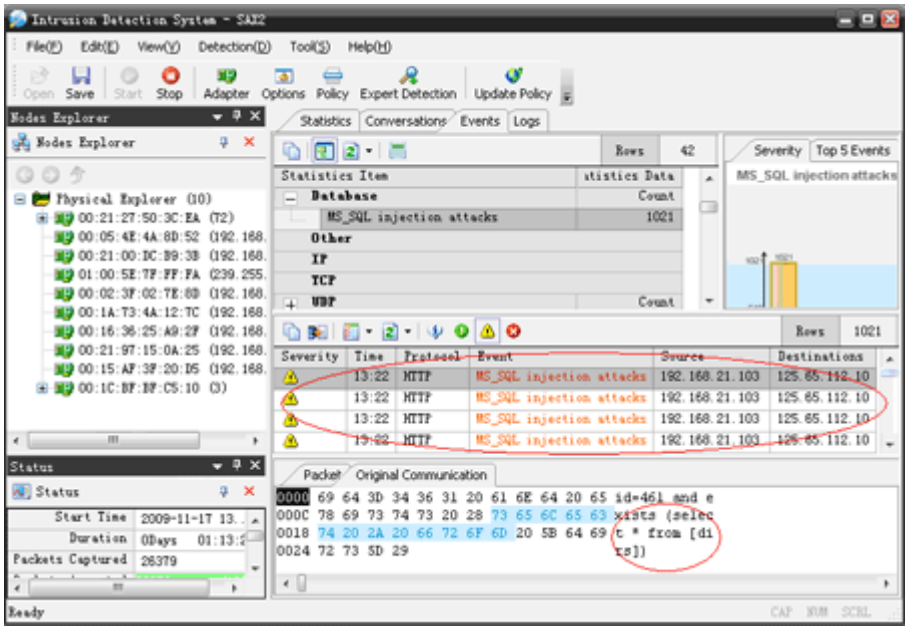
Aiemmin kuvattiin SQL-injektion toimintapa, mihin sitä käytetään ja miten sitä voidaan käyttää. Koodien pätkillä esiteltiin miten hyökkääjä voi toimia ja mitä hän voi tehdä huonosti tai virheellisesti ohjelmoidulla tietokannalla. Seuraavan kappaleen tavoitteena on esittää, miten SQL- injektio haavoittuvuudet voidaan havaita.

### 4.1.1 SQL- injektion havaitseminen

SQL-injektion havaitsemiseen on olemassa useita ilmaisia työkaluja. Merkkejä SQL-injektion haavoittuvuuksien hyödyntämisyrityksistä voi etsiä esimerkiksi www-palvelimen lokeista. Tämä tapahtuu etsimällä lokista HTTP-pyyntöjä, joissa pyydetty URL sisältää yhden tai useampia seuraavista: heittomerkki ja sen heksadesimaaliesitykset (', % 27, ...), perättäiset väliviivamerkit (--) ja "risuaita"-merkki ja sen heksadesimaaliesitykset (#, %23, ...). Lisäksi varsinaisten SQL- komentojen ja loogisten operaattoreiden esiintyminen voi olla merkki hyökkäysyrityksestä. Uhan aiheuttavat merkit ja merkkijonot riippuvat käytetystä tietokantasovelluksesta, tietokannan rakenteesta ja nimeämiskäytännöistä. Huomioitavaa HTTP- palvelimella tutkittavasta sovelluksesta löytyvistä esiintymistä saattaa olla vääriä hälytyksiä mutta ylläpitäjä havaitsee ne helposti. (Cert-fi 2007c.)

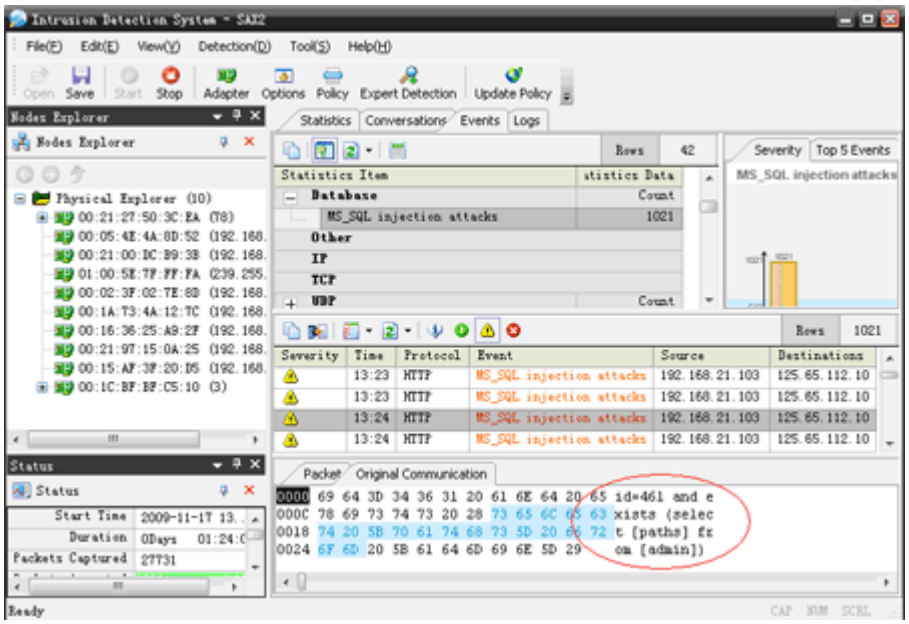
SQL- injektio voidaan havaita tunkeutumisen havainnointijärjestelmällä (Intrusion Detection Systems (IDS)). Tällä järjestelmällä voidaan nimensä mukaisesti havaita hyökkäysyrityksiä mutta ei suojautua niitä vastaan. Koska järjestelmä hakee hyökkäysyrityksiä lokien tiedoista, palvelimen lokikirjoitus täytyy olla päällä. Järjestelmä seuraa verkon liikennettä ja tallentaa niistä tietoja. (Wikipedia 2010c.) Esimerkissä on käytetty SAX 2-ohjelmistoa.

Kun SQL – injektio hyökkäystekniikkaa halutaan käyttää, niin käydään läpi SQL – injektion vaiheet. Ensimmäisenä käydään läpi se, mistä voidaan löytää haavoittuvuus. Seuraavaksi käydään läpi tietokannan tyyppi ja sen jälkeen haetaan taulun nimen, kentät ja sarakkeet, joista halutaan tietoa. Tämä hyökkäysyritys voidaan havaita SAX 2 ohjelmistolla, joka toimii verkossa oikeassa ajassa. (IDS-SAX2 2010.) Alla olevassa kuvassa SAX 2 ohjelmisto on havainnut hyökkäysmahdollisuuden esimerkki verkkosivulla:




Kuvio 1. SQL- injektion havaitseminen. (IDS-SAX2 2010)

Kuvassa on ympyröity hyökkäysyritys, joka on osoitettu IP- osoitteeseen 125.65.112.10 ja hyökkäys tulee IP- osoitteesta 192.168.21.103. Tällä hyökkäyksellä on havaittu tietokannan taulun nimi. Alemmasta ympyrästä näkee, että hyökkääjä on saanut selville taulun nimen, jota on käytetty verkkosivuston tietokantakyselyssä. Kun tällainen mahdollisuus on havaittu, hyökkääjä voi alkaa hakemaan tietokannan sarakkeiden ja kenttien nimiä. (IDS-SAX2 2010.) Alla olevassa kuvassa hyökkääjä hakee oikeaa sijaintia Admin tunnusten löytämiseksi.




Kuvio 2. SQL- injektion havaitseminen. (IDS-SAX2 2010)

Hyökkäysyritys on ympyröity punaisella ja hyökkääjä hakee tietoja muuttamalla hakua niin, että löytää loppujen lopuksi oikean sijainnin tunnuksille. Hyökkääjä löytää tällä tekniikalla kentän pituuden ja voi sitä kautta arvata mitä kenttä sisältää. Myös mahdollinen salasanan kryptauksen purkaminen on mahdollista. (IDS-SAX 2 2010.)

SAX2 ohjelmisto on hyvä ja toimiva SQL- injektion havaitsemiseen tarkoitettu ohjelmisto, jota kannattaa käyttää hyödyksi verkkosovellusten tarkistus ja testaus vaiheessa. Yllä käsiteltiin miten ja esimerkiksi millä ohjelmalla voidaan havaita SQL- injektio hyökkäysyritykset. Seuraavan kappaleen tarkoituksena on opastaa miten SQL- injektiota vastaan voidaan suojautua.

### 4.1.2 SQL-injektiolta Suojautuminen

Tämän kappaleen tarkoituksena on avata sitä, miten puolustaudutaan SQL-injektiota vastaan. Kappaleessa käydään läpi yleisimmät tavat, joilla voidaan estää potentiaalisia hyökkäysyrityksiä. Kappaleen tarkoituksena on parantaa ohjelmointitaitoja ja -osaamista.

SQL- injektiolta suojautumisen tärkeimmät kohdat ovat erikoismerkkien muuntaminen ja arvojen sulkeminen heittomerkein (’). Erikoismerkkien

muuntamisessa kannattaa ottaa esiin muutama seikka, jotka liittyvät kenttiin joihin syötetään tietoja. Kun erikoismerkkejä muutetaan, tulee ottaa huomioon se, että esimerkiksi käyttäjätunnus voi sisältää kyseisiä merkkejä. Myös tietokannoissa on käytössä erilaisia erikoismerkkejä, jotka voivat olla haitallisia. Esimerkiksi % ja _ merkit toimivat jokerimerkkeinä joidenkin operaattoreiden kanssa, joten nämäkin on tarkastettava, ellei toisin olla haluttu. Muuntamisen voi tehdä manuaalisesti tai automaattisemmin tietokannan kirjastossa olevilla funktioilla. (mureakuha 2010a.)

Koska manuaalinen tapa lisätä jokaisen erikoismerkin eteen kenoviivan(\) ja jokaisen arvon sulkemiseen käytettävän heittomerkkien lisääminen olisi turhan työlästä, perehdymme pelkästään tietokantakirjaston funktioihin. Jos manuaalisen arvojen käsittelyn virheen mahdollisuus on suuri, kannattaa käyttää automaattista arvojen muuttamista. Yleisin toteutus tälle on hakujen esikäsittely. Tällä saadaan pidettyä itse kysely ja haku erillään toisista, joten itse kyselyyn ei tule muuttujien arvoja ollenkaan. MySQL ja PHP: tä käytettäessä tämä tulisi tehdä mysql_real_escape_string- funktiolla. (Mureakuha 2010a.)

Oletetaan, että mahdollinen hyökkääjä yrittää hyökätä sivustolle käyttäen edellä esitettyjä tapoja ja sivustolla on käytetty mysql_real_escape_string-funktiota suojaamaan verkkosivuilla olevia tietokantakyselyjä. Kyseinen funktio kutsuu tietokantakirjastoa, joka käy syötteen läpi ennen tietokantakyselyä, ja jos syöte sisältää vaarallisia merkkejä niin syöte palautuu turvallisessa muodossa. Funktio muuttaa nämä erikoismerkit kenoviivaksi (\) ja sulkee arvot heittomerkein. Tätä funktiota tulisi käyttää lähes aina ennen kuin kysely tehdään tietokantaan. (PHP Manual 2010d.)

Mysql_real_escape_string-funktiota voidaan tarkastella helpoiten koodin avulla. Koodissa käydään läpi, kuinka mahdollinen hyökkääjä kirjoittaa tunnus kenttään hyökkäysyrityksen ” ’OR 1 ’ ” ja yrittää päästä tuhomaan tietokantaa DELETE FROM komennolla.

$tunnus=”’OR 1’”;

```
$tunnus= mysql_real_escape_string($tunnus);

$tunnuskysely= “SELECT * FROM kayttajat WHERE tunnus=’ $tunnus’”;
```

Tässä koodissa hyökkääjän komento estyy mysql_real_escape_string ($tunnus) funktiolla, joka palauttaa hyökkäysyrityksen muotoon SELECT * FROM kayttajat WHERE tunnus = ’\’ OR 1’\’. Eli käytännössä katsoen kyselyssä haetaan kayttajat taulusta ” ’\’ OR 1 ’ ” -nimistä käyttäjää. (Tizag 2010a.)

Ohjelmakoodin tulisi olla sellaista, jotka estävät hyökkäyksien tekemistä. Käyttäjän syötteiden rajaaminen ja käyttäjän syötteiden ominaisuuksien rajaaminen estää SQL-injektiota. Näitä rajoitteita tulisi käyttää, jotta verkkosivun tietoturvallisuus olisi parempi.

### 4.2 Local- ja remote file inclusion

Tämän kappaleen tavoitteena on avata, mitä Local- ja remote file inclusionit ovat, miten voit havaita Local- ja remote file inclusion hyökkäyksiä ja miten voit omalla ohjelmoimisella suojata omaa verkkosovellusta näiltä kahdelta hyökkäykseltä. Tulevissa kappaleissa käydään läpi yksinkertainen esimerkki hyökkäystekniikoista, miten hyökkäyksiä voidaan havaita omassa ohjelmistossa ja miten yksinkertaisia local- ja remote file inclusion hyökkäyksiä vastaan voidaan puolustautua.

#### 4.2.1 Local file inclusion

Local file inclusion-hyökkäyksellä voidaan päästä käsiksi palvelimen tiedostoihin joihin www-palvelun prosessilla on käyttöoikeus. Useimmissa tapauksissa LFI on syötetty verkkosivujen URL: ään. Tämä sen takia, koska ohjelmoijat käyttävät paljon GET-metodia silloin, kun joku toinen sivu sisällytetään verkkosivuun käyttäen INCLUDE() toimintoa. (Ackack 2010.)

Tyypillinen esimerkki haavoittuvaisesta PHP-koodista seuraavana:

```
<? PHP
$file= $_GET [’file’];
IF (ISSET ($file))
```

```
{
        INCLUDE (“pages/$file”);
}
ELSE
        INCLUDE (”index.php”);
}
?>
```

Yllä olevan koodin kutsuminen menisi esimerkiksi näin http://esmerkki.fi/index.php?file=esimerkkitiedosto.php. Tämänlainen koodinpätkä mahdollistaa mahdollisen hyökkääjän pääsemisen tiedostoihin, jotka eivät ole verkkosivuston kansiossa vaan esimerkiksi palvelimen juuressa. Näihin tiedostoihin käsiksi pääsemiseen mahdollinen hyökkääjä käyttää local file inclusion-hyökkäystä. Yksinkertainen esimerkki tämän käytöstä on syöttää esimerkki.fi sivuston URL:ään http://esimerkki.fi/index.php?file=../../etc/salasana. (Ackack 2010.)

Tällä hyökkäyksellä saadaan haettua palvelimen kaikki salasanat, jotka voidaan myöhemmin murtaa ja näin saadaan myös tiedostot haettua palvelimelta. Yksinkertaisiin local file inclusion-hyökkäyksiin on olemassa yksinkertainen ratkaisu, jota kaikkien PHP- ohjelmoijien suotaisiin käyttää koodissaan. Yllä olevaan koodin tehdään pieni muutos, jolla saadaan estettyä hyökkäys, uusi koodi menee näin:

```
<? PHP
$file= STR_REPLACE (‘.. /’,’’, $_GET [’file’];
IF (ISSET ($file))
{
        INCLUDE (“pages/$file”);
}
ELSE
        INCLUDE (”index.php”);
}
?>
```

Tämän koodin muutoksen ansiosta yksinkertainen LFI-hyökkäys ei enää toimi. (Ackack 2010.) Yllä esitettiin yksinkertainen esimerkki Local file inclusion-hyökkäyksestä. Seuraavan kappaleen tarkoituksena on opastaa mikä on Remote file inclusion.

### 4.2.4 Remote file inclusion

Remote file inclusion -haavoittuvuuksia esiintyy useimmiten PHP-pohjaisissa verkkosovelluksissa. Tämä haavoittuvuus esiintyy syötteen tarkistuksen yhteydessä ja sillä pyritään suorittamaan kohteena olevalla palvelimella hyökkääjän omia komentokoodeja. Hyökkäysmenetelmä on vanha mutta sitä käytetään edelleen paljon. Hyökkäysyrityksiä tekevät usein toiset kaapatut palvelimet ja ne kohdistuvat satunnaisesti valittuihin www-palvelimiin. Www-palvelinten lokeja on suositeltavaa tutkia ajoittain, jotta ylläpitäjällä pysyisi riittävän tarkka kuva hänen palveluunsa kohdistuvista ja liikkeellä olevista hyväksikäyttöyrityksistä. (Cert-fi 2008b.) Seuraavaksi käydään esimerkin avulla läpi, kuinka tällainen haavoittuvuus toimii.

Esimerkkinä oletetaan, että meillä on verkosta löydetty haavoittuva sivusto. Sen osoite menee näin: www.esimerkkisivusto.com/index.php?page=home. Kuten osoiterivistä huomataan, haavoittuva sivusto hakee palvelimelta tekstitiedoston include toimintoa käyttäen, joka sisältää sivuston koodin. Tämä avaa mahdollisuuden liittää oman kooditiedoston osoiterivin loppuun ja mahdollistaa näin hyökkäyksen. Hyökkäys menisi näin: www.esimerkkisivusto.com/index.php?page=hyökkäystiedosto.txt. Hyökkäystiedosto.txt sisältää koodin, jolla voidaan päästä käsiksi palveluun. Mikäli tätä haavoittuvuutta ei ole estetty, niin palvelin on hyökkääjän hallussa ja hän voi käyttää palvelinta haluamansa mukaan. (Wikipedia 2010d.)

## 4.4 Cross site scripting

Cross site scripting(XSS) on tietoturva- aukko, joka esiintyy useimmiten verkko ohjelmissa. XSS-aukko mahdollistaa koodin syöttämisen ja sen avulla mahdollisen hyökkäyksen verkkosivuille ilman, että selain varmistaa sitä. Haavoittuvuutta voidaan käyttää esimerkiksi asiakaspäässä suoritettavilla

JavaScripteillä tai HTML-koodilla. Erityisen vaarallisia nämä aukot ovat sivustoilla, jotka sisältävät arkaluonteista tai henkilökohtaista tietoa. XSS-tietoturva-aukot ovat verkkosivuilla olevia haavoittuvuuksia, joilla hyökkääjä voi ohittaa yhtenäisen turvallisuuskäytännön, joka tuli JavaScriptin turvallisuusriskin myötä. Turvallisuuskäytäntö sallii objektien ja sivujen vuorovaikutuksen vain samalla yhteydellä ja saman verkkotunnuksen sisällä. Turvallisuuskäytännöllä pyritään estämään selaimen toisessa ikkunassa olevat salaiset tiedot. XSS-aukon myötä hyökkääjä voi päästä käsiksi salattaviin tietoihin, evästeisiin ja muihin kohteisiin. (Owasp 2010c.)

Cross site scripting-haavoittuvuuksia on kolmea erilaista, jotka ovat Dom-pohjaiset haavoittuvuudet, ei pysyvät haavoittuvuudet ja pysyvät haavoittuvuudet. Dom-pohjainen haavoittuvuus on paikallinen haavoittuvuus, joka löytyy sivuston koodista itsessään. Haavoittuvuus löytyy useimmiten sivuston JavaScriptistä. Sivusto on haavoittuva silloin, jos koodia ei käsitellä ennen sivulle lisäämistä. Vanhemmilla Internet Explorer selaimilla tällaiset koodit voivat jopa aiheuttaa etäkäyttöhaavoittuvuuden. Käytännössä katsoen tällainen paikallisesti sivua selaavan käyttäjän kovalevylle ajettavan koodin turvallisuusaukko on korjattu jo Internet Explorer 6 Service Pack 2: ssa. (Wikipedia 2010b.)

Toisena Cross site scripting-haavoittuvuuksista on selaimella käytettävät haavoittuvuudet. Tällainen haavoittuvuus esiintyy selaimen osoiterivistölle syötetyssä osoitteessa. Tämä haavoittuvuus on todella yleinen. Tätä haavoittuvuutta voidaan käyttää siten että huijataan sivuston käyttäjä painamaan huijauslinkkiä esimerkiksi sähköpostissa. Huijauslinkki voi johtaa sivustolle, joka näyttää samalta kuin alkuperäinen mutta onkin hyökkääjän oma sivusto, johon käyttäjä syöttää tietoja ja hyökkääjä saa ne itselle. Näitä huijauslinkkejä esiintyy paljon erilaisissa sosiaalisissa ja yhteisöllisissä medioissa. (Wikipedia 2010b.)

Kolmantena haavoittuvuuksista on pysyvät haavoittuvuudet. Tällainen haavoittuvuus löytyy sivuston tietokannasta, johon hyökkääjä on sen laittanut. Hyökkääjän ei tarvitse houkutella käyttäjiä painamaan linkkiä vaan ajaa hyökkäyksiä suoraan käyttäjien selaimia apuna käyttäen. Tällaisia haavoittu-

vuuksia löytyy useasti sivustoilla, joilla voidaan tuottaa sisältöä, kuten yhteisöpalvelut ja keskustelufoorumit. Tällä haavoittuvuudella mahdollistetaan käyttäjien tunnusten ja salasanojen hakemisen niin, ettei käyttäjä huomaa syöttäneensä niitä vahingossa väärään paikkaan. (Wikipedia 2010b.)

### 4.4.1Cross site scripting- haavoittuvuuden havaitseminen

Paras tapa havaita haavoittuvuus on suorittaa turvallisuustarkistus käymällä oma koodi läpi. XSS-aukkoja voidaan löytää sellaisista paikoista missä sisään tuleva http-pyyntö löytää tiensä ulostulevaan HTML-koodiin. Huomioitavaa on myös se, että erilaiset HTML-elementit voivat sisältää hyökkäysmahdollisuuden. On olemassa erilaisia ohjelmia kuten Nessus ja Nikto, joilla voidaan havaita tällaisia haavoittuvuuksia. Mutta jos sivustolla on esimerkiksi HTML-elementti haavoittuvuus, voidaan olettaa myös, että sivusto on kauttaaltaan haavoittuvainen. (Owasp 2010a.)

### 4.4.2 Cross site scripting- haavoittuvuudelta suojautuminen

Jotta XSS-haavoittuvuudelta voidaan suojautua, on hyvä tietää sääntöjä mitä pitää ottaa huomioon verkkosivuilla haavoittuvuuden estämiseksi.
Nämä säännöt kannattaa huomioida siten, että säännöt palvelevat oman sivuston tarkoitusta. Nämä säännöt ovat luotu suojaamaan verkkosivustoa ja näitä sääntöjä ei tulisi ohittaa silloin kun luodaan verkkosivuja tai verkkosovelluksia. XSS-haavoittuvuuksia on paljon ja näillä säännöillä saadaan niitä kitkettyä (Owasp 2010b).

Sääntö numero yksi on: syötä epäluotettava data sellaiseen paikkaan, missä se on sallittua. Epäluotettava data on sellaista, joka tulee HTTP-pyynnöistä, lomake kyselyistä, URL-parametreistä tai evästeistä. Kuten aikaisemmissa kappaleissa on käsitelty, niin syötettävä data pitää käsitellä ja se pitää suojata niin, että hyökkäyksiltä voidaan välttyä. Aina tulee ajatella että epäluotettava data on mahdollinen hyökkäysyritys. Aikaisemmin olemme käsitelleet tietokannataan menevän datan estääksemme SQL-injektion, joten nyt käsitellään lähetettävää dataa. (Owasp 2010b.)

Ensimmäisen säännön ideana on estää kaikki epäluotettava data vääristä paikoista. Tällaisia paikkoja ovat HTML-koodin sisältävän <script></script> merkkien välissä, HTML-koodin kommentissa, HTML-koodin ominaisuuden nimeksi tai merkin nimeksi. Tärkeää on myös huomioida se että ohjelmistokoodi ei sisällä epäluotettavaa dataa, joka kutsuu JavaScriptiä ja ajaa sitä väärässä paikassa. Toisena sääntönä tulee tarkistaa, että epäluotettava data on tarkistettu ennen kuin se menee HTML-koodin sisällöksi. Koodin tulee olla tarkistettu sellaisissa paikoissa, kuten <body></dody> ja  välissä ja myös muiden HTML-elementtien välissä. (Owasp 2010b.)

Kolmantena tulee huomioida se, että elementin kuten  ominaisuudet eivät sisällä epäluotettavaa dataa. Esimerkiksi elementin nimessä oleva epäluotettava data pitää olla tarkistettua, jotta se ei mahdollista haavoittuvuutta, etenkin jos elementin ominaisuudet sisältävät JavaScriptiä. Neljäntenä tulee huomioida se, että JavaScriptin käsittelypyynnöt ovat tarkistettuja HTML-elementin sisällä. Ainut turvallinen paikka JavaScriptin käsittelypyynnölle on ”datan arvo”- kohta. Muussa tapauksessa se voidaan vaihtaa helposti toiseksi käsittelypyynnöksi ja siten saadaan aikaan hyökkäys sivustolle. Huomioitavaa JavaScriptissä on se, että kaikki JavaScriptin toiminnot eivät ole turvallisia vaikka ne onkin tarkistettu. (Owasp 2010b.)

Viidentenä haavoittuvuusmahdollisuutena tulee Cascading styling sheet- sivuton tyyli asettelut. CSS on vaarallinen ja monipuolisia hyökkäysmenetelmiä mahdollistava, jos sitä ei ole käsitelty ennen kuin se lisätään sivuston tyylielementtiin. CSS-tiedostoissa epäluotettava data tulee sijoittaa vain ja ainoastaan ominaisuuksien arvoihin ja niin ettei se sisällä JavaScriptiä. Kuudentena tulee huomioida URL: n tarkistaminen ennen kuin se sisältää epäluotettavaa dataa. Local file Inclusion-hyökkäyksen estämisessä käytettyjen keinojen avulla voidaan saada tällainen epäluotettava data pois URL- parametreistä. (Owasp 2010b.)

## 5 Yhteenveto

Yhteenvetona voidaan tiivistää opinnäytetyöprosessin aikana vastaan tulleita asioita, mitä tulee huomioida, kun tehdään verkkosivujen ohjelmointia ja siihen liittyvää tietoturvaa.

Ensimmäisenä kannattaa tarkistaa käyttäjien syötteet: älä luota käyttäjään ja käyttäjän syötteisiin, äläkä päästä syötteitä sellaisenaan palveluun. Tarkista, että syötteet ovat mitä niiden kuuluukin olla. Toiseksi, estä salasanoja ja muita tärkeitä tietoja sisältävien tiedostojen lukeminen. Tiedostojen lukemisen estämiseksi on monia tapoja. Tiedostojen nimeäminen PHP-tarkentimella auttaa tietojen lukemisen estämistä. Muista myös kytkeä PHP: n vahingolliset ominaisuudet pois päältä. Tärkeämpiä asoita on estää SQL-injektio. Käytä mysql_real_escape_string tai vastaavaa funktiota tehdäksesi syötteet, joita aiot käyttää SQL-kyselyssä turvallisiksi. Estä myös yksinkertaiset local- ja remote file inclusion-hyökkäykset ja varaudu myös cross site scripting hyökkäyksiin (Auralinna 2009).

Estä virheilmoituksien näkyminen käyttäjille. Virheilmoitukset voivat olla päällä kehitysvaiheessa mutta ne eivät saa näkyä missään vaiheessa loppukäyttäjällä. Muista, että salasanoja ei saa koskaan tallentaa suoraan vaan ne pitää ajaa jonkin, esimerkiksi kryptauksen läpi. Aseta salasana vähintään 6 merkkiä pitkäksi ja sen tulee sisältää kirjaimia ja numeroita. Nykyisin käytettävät vahvuusmittarit sivustolla on hyvä apu saada salasanoista vahvoja. Muista asettaa käyttäjille rajat käyttäjätunnuksia luodessa. Käytä lokeja paikantamaan virheet sivustolla ja pysyttele ajan tasalla tietoturvan ja tietoturvauhkien tasolla. Verkkosivujen ja verkkosovellusten tekemisessä tietoturva on todella tärkeää ja ylläesitettyjen asioiden läpikäyminen ja tarkistaminen parantaa verkkosivujen ja sovellusten tietoturvaa huomattavasti. (Auralinna 2009.)

Opinnäytetyö oli mielenkiintoinen ja haastava prosessi, minkä suorittamiseen meni paljon aikaa. Opinnäytetyössä pääsi hyödyntämään paljon koulussa ja työharjoittelussa oppimiani asioita. Ohjelmointipuoli oli minulle mieluinen prosessi ja tällaisen projektin tekeminen auttaa paljon opittujen asioiden sisäis-

tämistä. Uskon että tällaisen projektin läpikäyminen edesauttaa opiskelijoita työnhaussa ja työn saamisessa. Tekstiosuutta tehdessä opin paljon uutta mitä tulee ottaa huomioon verkkosivujen ohjelmoinnissa ja tietoturvassa.

Uskon, että tällaisen tekstin kirjoittaminen ja läpikäyminen edesauttaa alalla opiskelevia ja alalla olevia henkilöitä. Alan nopea kehittyminen vaikeuttaa tällaisten töiden tekemistä, koska lähteet ja tiedot vanhenevat nopeasti. Kirjoja on vaikea käyttää esimerkiksi työssä esiteltyjen hyökkäystekniikoiden tutkimisessa. Verkossa olevat lähteet ovat useasti ajan tasalla, mutta toisaalta ne tulee käydä läpi ja varmistaa, että niissä oleva sisältö on sellaista, mitä sen kuuluukin olla.

Opinnäytetyö oli työntäyteinen ja sain prosessista paljon irti. Opin paljon uutta asiaa ja prosessissa tuli kerrattua paljon vanhaa asiaa. Neljännessä kappaleessa käydyistä asioista opin todella paljon uutta, ja josta uskon olevan hyötyä tulevaisuudessa. Mielestäni tällaiset tietoturvallisuuden uhkakuvat tulee ottaa todella vakavasti. Nykyisen yhteiskunnan kehittyessä yhä enemmän tietoverkkojen ympärille ja verkkolaitteiden kasvaessa tietoturvallisuus on entistä tärkeämpää.

Uskon, että tietokantapohjaisten verkkosivujen tietoturvallisuuteen perehtyminen ja siihen liittyvien uhkakuvien tiedostaminen edesauttaa tulevaisuudessa työn haussa ja työn tekemisessä. Mielestäni opinnäytetyöprosessi oli mielenkiintoinen ja onnistunut. Opinnäytetyöhön haastavuutta toi se, ettei opinnäytetyöni ollut raportti tehdyistä verkkosivuista vaan tutkimus tietoturvallisuuteen käyttäjän syötteiden kautta. Tällaisten haastavien projektien tavoitteena on edistää ammatillista osaamista.

**Lähteet**

Ackack 2010. Local File Inclusion. Osoitteessa http://downloads.ackack.net/LocalFileInclusion.pdf. 10.10.2010.

Auralinna 2009. Huomion arvoista PHP ohjelmoinnissa. Osoitteessa http://www.auralinna.fi/2009/08/07/turvallisten-www-sovellusten-toteutus-phplla/. 7.8.2009.

Cert-fi 2010a. Haavoittuvat sisällönhallintajärjestelmien ja verkkokauppasovellusten lisäosat. Osoitteessa https://www.cert.fi/tietoturvanyt/2010/04/ttn201004131507.html. 13.4.2010.

Cert-fi 2008b. Remote file inclusion- haavoittuvuuksia hyväksikäytetään runsaasti. Osoitteessa http://www.cert.fi/tietoturvanyt/2008/09/ttn200809132036.html. 13.9.2008

Cert-fi 2007c. SQL injection -hyökkäysten havaitseminen. Osoitteessa http://www.cert.fi/tietoturvanyt/2007/10/P_14.html. 9.11.2007.

Charter 2010. Using a Strong Username and Password. Osoitteessa http://www.myaccount.charter.com/customers/Support.aspx?SupportArticleID=1777#tips_password. 2.10.2010

Elisa 2010. SSL- salaus. Osoitteessa http://www.elisa.fi/asiakastuki/199.49/elisa-yritysweb/6299/ssl-salaus/. 18.10.2010

IBM 2006. Structured Query Language(SQL). Osoitteessa http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp?topic=/com.ibm.db2.udb.admin.doc/doc/c0004100.htm. 27.10.2006.

IDS-SAX2 2010. Detect SQL Injection Attacks with Sax2. Osoitteessa http://www.ids-sax2.com/articles/DetectSQLInjectionAttacks.htm 1.10.2010.

Linux 2009. Palvelin. osoitteessa http://linux.fi/wiki/Palvelin. 14.9.2009.

Microsoft 2010. Validating User Input. Osoitteessa http://msdn.microsoft.com/en-us/library/aa174437(SQL.80).aspx. 15.11.2010.

MySQL 2010a. INFORMATION_SCHEMA. Osoitteessa http://dev.mysql.com/doc/refman/5.0/en/information-schema.html. 28.10.2010

MySQL 2010b. What is MySQL? Osoitteessa http://dev.mysql.com/doc/refman/5.0/en/what-is-mysql.html. 28.10.2010

Mureakuha 2007a. SQL – injektio. Osoitteessa http://wiki.mureakuha.com/wiki/SQL-injektio. 25.7.2007.

Mureakuha 2006b. Turvallisten palvelujen tuottaminen. Osoitteessa http://wiki.mureakuha.com/wiki/Turvallisempien_palvelujen_toteuttaminen. 12.12.2006.

Owasp 2010a. Cross site Scripting. Osoitteessa http://www.owasp.org/index.php/Cross-site_Scripting_(XSS). 1.11.2010.

Owasp 2010b.Cross site Scripting Flaw. Osoitteessa http://www.owasp.org/index.php/Cross_Site_Scripting_Flaw. 22.4.2010.

Owasp 2010c. Cross site scripting Prevention Cheat Sheet. Osoitteessa http://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet#XSS_Prevention_Rules. 1.11.2010.

PHP Manual 2010a. General Information. Osoitteessa http://fi2.php.net/manual/en/faq.general.php. 12.10.2010.

PHP Manual 2010b. PHP- Crack. Osoitteessa http://www.php.net/manual/en/intro.crack.php. 21.10.2010.

PHP Manual 2010c. PHP- INCLUDE(). Osoitteessa http://php.net/manual/en/function.include.php. 12.10.2010

PHP Manual 2010d. PHP- mysql_real_escape_string- funktio. Osoitteessa: http://php.net/manual/en/function.mysql-real-escape-string.php. 12.11.2010.

PHP Manual 2010e. SQL- injection. Osoitteessa: http://php.net/manual/en/security.database.sql-injection.php. 12.11.2010.

Rovaniemen kristillinen koulu 2010. Koulu. Osoitteessa http://rovaniemenkristillinenkoulu.fi/koulu.php. 10.10.2010.

Ubinet 2010. Julkaisujärjestelmät. osoitteessa http://www.ubinet.fi/julkaisujarjestelma. 7.10.2010.

Sqlbook 2010. SQL Injection Attacks - are your databases secured? Osoitteessa http://www.sqlbook.com/SQL/SQL-Injection-Attacks-29.aspx. 14.10.2010.

Tizag 2010a. mysql - sql injection prevention. Osoitteessa http://www.tizag.com/mysqlTutorial/mysql-php-sql-injection.php. 1.11.2010.

Tizag 2010b. SQL Union. Osoitteessa
http://www.tizag.com/sqlTutorial/sqlunion.php 2.10.2010

Tracert 2010. What is Traceroute? Osoitteessa
http://tracert.com/traceexplain.html. 13.10.2010.

Väestörekisterikeskus 2010. Palvelinvarmenne. Osoitteessa
http://www.vaestorekisterikeskus.fi/vrk/fineid/home.nsf/pages/20ED3CF022000595C2256FFF0038FA36. 28.10.2010.

Wikipedia 2010a. Avoin Lähdekoodi. Osoitteessa
http://fi.wikipedia.org/wiki/Avoin_l%C3%A4hdekoodi. 10.10.2010.

Wikipedia 2010b. Cross Site Scripting. Osoitteessa
http://en.wikipedia.org/wiki/Cross-site_scripting. 25.10.2010.

Wikipedia 2010c. Intrusion Detection System. Osoitteessa
http://en.wikipedia.org/wiki/Intrusion_Detection_System.
7.11.2010

Wikipedia 2010d.Remote File Inclusion.Osoitteessa
http://en.wikipedia.org/wiki/Remote_File_Inclusion. 23.10.2010

Wikipedia 2010e. WWW- Palvelin. Osoitteessa
http://fi.wikipedia.org/wiki/WWW-palvelin. 2.6.2010.

WordPress 2010. User Levels. Osoitteessa
http://codex.wordpress.org/User_Levels. 20.10.2010

W3schools 2010a. JavaScript Introduction. Osoitteessa
http://www.w3schools.com/js/js_intro.asp. 15.10.2010

W3schools 2010b. PHP $_GET Function. Osoitteessa
http://www.w3schools.com/PHP/php_get.asp. 20.10.2010